ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

IJESRT
INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

OPTIMIZED SOFTWARE FLAW FORECASTING SCHEME BASED ON DATA MINING AND DIFFERENTIAL EVOLUTION

Dr.A.R.Pon Periyasamy*, Mrs A.Misbahulhuda
Associate Professor, Dept. of Computer Science, Nehru Memorial College, Puthanampatti. Tamilnadu, India- 621 007
Research Scholar, Dept. of Computer Science, Nehru Memorial College, Puthanampatti. Tamilnadu, India- 621 007

DOI: 10.5281/zenodo.571738

ABSTRACT
Software flaws are affecting the software development process and performance which yields an unpredicted behavior and produce erroneous results. Software defects are expensive in terms of quality and cost. Detection and correction of these defects are significant task to ensure the quality of the software product. Numerous software repositories hold source code of large projects as many modules and hold data for the software metrics of these modules and the defective state of each module. In this paper, an optimized software flaw forecasting scheme is proposed based on Data Mining and Differential Evolution(DE). The data mining approach is used to select the attributes that forecast the defective state of software modules. The DE is utilized for optimization process. The experimental results are presented to exhibit the better prediction competence of the proposed scheme.
KEYWORDS: Data Mining, Differential Evolution, Forecasting, Optimization, Software flaw.

INTRODUCTION
A software flaw is a defect or fault in a computer program or system that produces an incorrect or unexpected result, or causes it to behave in unintended ways [32]. Software life cycle is a creature activity, so it is impractical to produce the software without defects. To deliver a defect free software it is essential to predict and fix the defects as many as possible before the product delivers to the customer. Software repositories have lots of information that is useful in assessing software quality. Data mining techniques and machine learning algorithms can be applied on these repositories to extract the useful information. Defect prediction and estimation models are used to predict the total number of defects and their distribution over time. Dynamic models require defect data and are used once the tracking of defects starts [15]. The number of remaining defects in an application under development may cause faults in the product that should be studied carefully to ensure reliability [19], [22], [23]. The most discussed problem is software defect prediction in the field of software quality and software reliability. As Boehm observed finding and fixing a problem after delivery is 100 times more expensive than fixing it during requirement and design phase. Additionally software projects spend 40 to 50 percent of their efforts in avoidable rework [33].

Data mining techniques and machine learning algorithms are useful in prediction of software bug estimation. Machine learning models and Data mining techniques can be applied on the software repositories to extract the defects of a software product. Common techniques include decision tree learning, Naïve Bayesian classification and neural networks, j48 and ONER. Software prediction model only works well when enough amount of data is available in software repository within the organization to initially feed the model. Extraction of defects from software bug repository accurately is not done without a good data mining model. There is a need of good data mining model to predict the software defects from a bug repository. A good data mining technique to build a better prediction model is an open issue [18].

In this paper, an optimized software flaw forecasting scheme proposed using data mining and differential evolution method. In this scheme software features are selected with help of data mining system and differential evolution method is used for optimization. The remainder of this paper begins with related work, followed by

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

materials and methods includes data mining and differential evolution techniques, then the subsequent sections are offering proposed optimized software flaw forecasting scheme through DE, experimental results and the conclusion.

RELATED WORK

Software Defect Prediction Model refers to those models that try to predict potential software defects from test data. There exists a correlation between the software metrics and the fault proneness of the software. A Software defect prediction models consists of independent variables (Software metrics) collected and measured during software development life cycle and dependent variable (faulty or non faulty). There are different data mining techniques for defect prediction. Laszlo.M., Mukherjee.S. have proposed Genetic Algorithm based Hyper-Quad-trees for Low-Dimensional K-means Clustering system. 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[13]. Shi Zhonget.al. have offered an 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[16].

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. Lessmann et al., have experimented the performance of classification algorithm. To compare the software defect prediction, experiments were conducted using 10 public domain datasets from NASA Metric Data repository, using 22 classifiers[17]. The system may work perfect in the beginning since the defect involved in it may be invoked at a later point of execution in its cycle. It may then induce irrelevant outputs from then. A system evolves from smaller to larger and the largest as per the requirements change and thus the complexity. It may be developed by combining different data sources to meet the specifications requested. It is always a tedious task to analyze a huge system and interpret the errors that could have incorporated in the system [20]. Partha Sarathi Bishnu et al. have exercised k-mean technique of clustering for Software Defect Prediction. K-mean clustering is a non-hierarchical clustering procedure in which items are moved among sets of clusters until the desired set is reached. It has certain drawbacks, so to overcome those drawbacks Quad Tree-based k-mean clustering method was proposed. The objective was initially, Quad-trees are applied for finding initial cluster centers for k-mean algorithm. Subsequently, the Quad tree-based algorithm is applied for predicting faults in program modules. They have evaluated the effectiveness of Quad tree-based k-mean clustering algorithm in predicting faulty software modules as compared to the original k-mean algorithm [21].

Ahmed H. Yousef has presented that the data mining models have reasonable accurate prediction of these defective modules. The best prediction performance measures achieved by the use of individual approach were for the Naive Bayes algorithm, then the Neural Network algorithm and the Decision Trees algorithms. The weighted voting rule approach is used to combine the decisions of the four models and achieved better results for accuracy, precision, recall and F-measure[24]. Golnoush Abaei et al. have offered a semi supervised hybrid self-organizing map (HYSOM) model in order to detect defects with a high accuracy and improve detection model. The generalization ability of HYSOM model will predict the label of modules in a semi-supervised manner. This is applied on eight industrial data sets from NASA and Turkish data set. It can also be used as an automated tool to predict defects in less time for project managers, software developers and Testers [25]. I. H.Laradji et al. have offered a two-variant ensemble learning classifier which shows that greedy forward selection is better than correlation forward selection. Further they proposed a model APE with seven different classifiers which results much better when compared to weighted SVM's and random forests. Further thy enhanced the version of APE with greedy forward selection to produce higher AUC measures for the different data sets. The results shown stronger robustness to redundant and irrelevant features [26].

Wen Zhang et.al have recommended Bayesian Regression Expectation Maximize algorithm for software effort prediction and two embedded strategies handle missing data. They used the method of ignoring the missing data in an iterative manner in the predictive model. Here they have used data sets such as ISBSG and CSBSG. When there are no missing data BREM outperforms CR, BR, and SVR& M5. When there are missing data BREM with MDT and MDI outperforms imputation technique includes MI, BMI, CMI and Mini & M5. BRM is used for software prediction and MDI used for finding missing values embedded with BREM[27]. Fahimeh Sadat FAZEL has proposed a system to predict software error via genetic algorithm. This system predicts software error with a higher precise and speed with easy. The achieved results indicate a desired performance of this system from time period for predicting error and output rate or recognition. The results show the recognition rates of suggested method more that 95 percent in best condition[28].Sonu Kumar Kushwaha et al. have analyzed the data mining techniques that are association mining, classification and clustering for software defect prediction [29].

Sajna.P, Shahad.P have conducted a survey regarding Data mining techniques for software defect prediction. Various classifiers are used to classify faulty or non-faulty modules. Then we compare our methods and find which method is better. The quality of software datasets can be improved by data preprocessing. In empirical studies, we chose datasets from real-world software projects, such as Eclipse and NASA[30]. Jyoti Devi and Nancy Seghal have analyzed about software prediction model and how it is used to control the classes of software which are often to change. Machine learning algorithms are used for predicting software[31].

MATERIALS AND METHODS

Data Mining Technique

Data mining has grasped the attention in the information industry and society due to the extensive availability of vast amounts of data and the looming requirement for turning such data into useful information and knowledge. Data mining is defined as extracting or "mining" knowledge from large amounts of data. The data mining is principally very helpful to by statisticians, database researchers, and the MIS and business communities. The term Knowledge Discovery in Databases (KDD) is normally used in large level to refer the process of determining constructive knowledge from data, where data mining is a particular step in this process. [5], [10].The supplementary steps in the KDD process, for instance data preparation, data selection, data cleaning, and proper interpretation of the results of the data mining process, guarantee that helpful knowledge is gained from the data.

Data Mining can be divided into two tasks: Predictive tasks and descriptive tasks. Predictive task is to predict the value of a specific attribute (target/dependent variable) based on the value of other attributes (explanatory). Descriptive task is to derive patterns (correlation, trends, and trajectories) that summarize the underlying relationship between data. Essentially, the two types of data mining approaches differ in whether they seek to build models or to find patterns. The first approach, concerned with building models is, apart from the problems intrinsic from the large sizes of the data sets, similar to predictable exploratory statistical methods. The purpose is to generate an overall summary of a set of data to identify and illustrate the most important features of the shape of the distribution. The second type of data mining approach is pattern detection which is utilized to search and identify small departures from the norm, to discover the unusual patterns of behavior. For instance include the unusual spending patterns in credit card usage, sporadic waveforms in EEG traces, and objects with patterns of characteristics unlike others. This division of strategies is useful to lead to the concept of data mining as seeking "nuggets" of information among the mass of data [7].

Data mining systems can be Classified according to various criteria such as
(i) Kinds of Databases, (ii) Kinds of Knowledge,
(iii)Kinds of Techniques Utilized,(iv)Applications Adapted.

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

The figure 1 shows the Architecture of Data Mining Process.

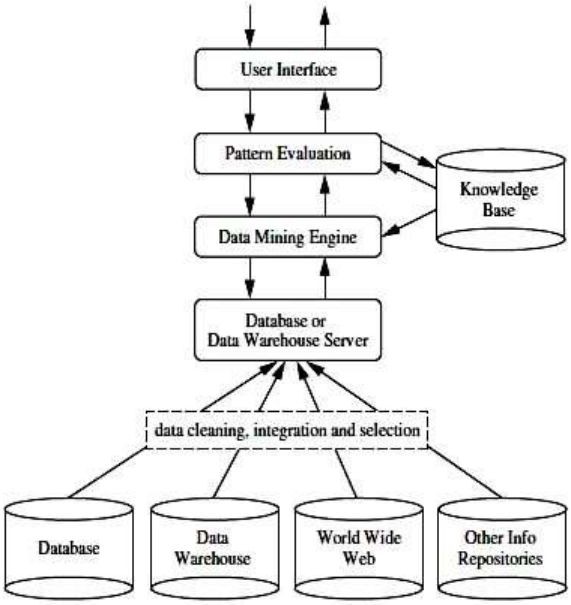


Figure 1: The Architecture of Data Mining.

Software bug repository is the main resource for fault prone modules. Different data mining algorithms are used to extract fault prone modules from these repositories. Software development team tries to increase the software quality by decreasing the number of defects as much as possible. The clustering, association mining, and classification of data mining techniques are used for software defect prediction system.

Clustering: Clustering is a form of unsupervised learning in which no class labels are provided. It is often the first data mining task applied on a given collection of data. In this, data records need to be grouped based on how similar they are to other records. It is a task of organizing data into groups such that the data objects that are similar to each other are put into same cluster[3][8]. The groups are not predefined. It is a process of partitioning a data in a set of meaningful sub-classes called clusters. Clusters are subsets of objects that are similar. Clustering helps users to understand the natural grouping or structure in a data set. Its schemes are evaluated based on the similarity of objects within each clusters.

Classification: Classification is a process of finding a set of models that describe and distinguish data classes or concepts. It is the organization of data in given classes known as supervised learning, where the class labels of some training samples are given. These samples are used to supervise the learning of a classification model[1]. Classification approaches normally use a training set where all objects are already associated with known class labels. The classification algorithm learns from the training set and builds a model. The model is used to classify new objects. Fraud detection and credit risk applications are particularly well suited to this type of analysis[1][2]. This approach frequently employs decision tree or neural network-based classification algorithms. The data classification process involves learning and classification.

Association: The Association mining task consists of identifying the frequent itemsets, and then forming conditional implication rules among them. It is the task of finding correlations between items in data sets. Association Rule algorithms need to be able to generate rules with confidence values less than one. Association rule mining is undirected or unsupervised data mining over variable-length data and it produces clear, understandable results. The task of association rules mining consists of two steps. The first involves finding the set of all frequent item sets. The second step involves testing and generating all high confidence rules among item sets.

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

Differential Evolution

Differential Evolution is a Stochastic, population-based optimization algorithm has been introduced by Storn and Price in 1996. DE is developed to optimize real parameter, real valued functions. Differential Evolution grew out of Ken Price's attempts to solve the Chebychev Polynomial fitting Problem that had been posed to him by Rainer Storn. A breakthrough happened, when Ken came up with the idea of using vector differences for perturbing the vector population. The "DE community" has been growing since the early DE years of 1994 - 1996 and ever more researchers are working on and with DE.

In evolutionary computation, differential evolution (DE) is a method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. Such methods are commonly known as metaheuristics as they make few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions. DE optimizes a problem by maintaining a population of candidate solutions and creating new candidate solutions by combining existing ones according to its simple formulae, and then keeping whichever candidate solution has the best score or fitness on the optimization problem at hand. DE two main stages: crossover and mutation. The crossover procedure takes two selected vectors and combines them about a crossover point thereby creating two new vectors. The mutation procedure modifies a certain vector subject to a mutation function, introducing further changing into the original vectors [6],[9]. Storn and Price have been published many books on theoretical and practical aspects of using DE in parallel computing, multiobjective optimization, constrained optimization, and the books also contain surveys of application areas.[11][12]

Differential Evolution (DE) begins through a population of NP candidate solutions which may be represented as Xi,G, i = 1, . . . ,NP, where i index denotes the population and G denotes the generation to which the population belongs. The working of Differential Evolution depends on the manipulation and efficiency of three main operators; *Mutation, Crossover and Selection*[4].

(i)Mutation: Mutation operator is the leading operator of DE and it is the implementation of this operation that makes DE different from other Evolutionary algorithms. The mutation operation of DE applies the vector differentials between the existing population members for determining both the degree and direction of perturbation applied to the individual subject of the mutation operation. The mutation process at each generation begins by randomly selecting three individuals in the population. The most often used mutation strategies implemented in the DE codes are listed below.

$$\text{DE/rand/1: } V_{i,g} = X_{r_1, g} + F * (X_{r_2, g} - X_{r_3, g}) \quad (1.1)$$

$$\text{DE/rand/2: } V_{i,g} = X_{r_1, g} + F * (X_{r_2, g} - X_{r_3, g}) + F * (X_{r_4, g} - X_{r_5, g}) \quad (1.2)$$

$$\text{DE/best/1: } V_{i,g} = X_{best,g} + F * (X_{r_1, g} - X_{r_2, g}) \quad (1.3)$$

$$\text{DE/best/2: } V_{i,g} = X_{best,g} + F * (X_{r_1, g} - X_{r_2, g}) + F * (X_{r_3, g} - X_{r_4, g}) \quad (1.4)$$

$$\text{DE/rand-to-best/1: } V_{i,g} = X_{r_1, g} + F * (X_{best,g} - X_{r_2, g}) + F * (X_{r_3, g} - X_{r_4, g}) \quad (1.5)$$

Where, i = 1, . . . , NP, $r_1, r_2, r_3 \in \{1, \ldots, NP\}$ are randomly selected and satisfy: $r_1 \neq r_2 \neq r_3 \neq i$, $F \in [0, 1]$, F is the control parameter proposed by Storn and Price [1]. In this scheme the basic strategy (1.1) is used.

(ii) Crossover: Thus, each individual of *Crossover*: once the mutation phase is complete, the crossover process is activated. The perturbed individual, $V_{i,G+1} = (v_{1,i,G+1}, \ldots, v_{n,i,G+1})$, and the current population member, $X_{i,G}$ =

$(x_{1,i,G}, \ldots, x_{n,i,G})$, are subject to the crossover operation, that finally generates the population of candidates, or "trial" vectors,$U_{i,G+1} = (u_{1,i,G+1}, \ldots, u_{n,i,G+1})$, as follows:

$$u_{j,i.G+1} = \begin{cases} v_{j,i.G+1} & if \ \ rand_j \le C_r \vee j = k \\ x_{j,i.G} & otherwise \end{cases} \quad (2)$$

Where, j = 1. . . n, k ∈{1, . . . , n} is a random parameter's index, chosen once for each *i*, and the crossover rate, Cr ∈ [0, 1], the other control parameter of DE, is set by the user. The temporary (trial) population is compared with its counterpart in the current population. The one with the lower objective function value will survive from the tournament selection to the population of the next generation. As a result, all the individuals of the next generation are as good or better than their counterparts in the current generation. In DE trial vector is not compared against all the individuals in the current generation, but only against one individual, its counterpart, in the current generation.

(iii) Selection: The selection process of DE is differs from other evolutionary algorithms. The population for the next generation is selected from the individual in current population and its corresponding trial vector according to the following rule:

$$X_{i.G+1} = \begin{cases} U_{i.G+1} & if \ \ f(U_{i.G+1}) \le f(X_{i.G}) \\ X_{i.G} & otherwise \end{cases} \quad (3)$$

PROPOSED OPTIMIZED SOFTWARE FLAW FORECASTING SCHEME

The proposed data mining and differential evolution based optimized software error prediction system is developed for identify the defects in the software and improve the software quality. In this system, data mining WEKA tool is used to select the software attributes and DE algorithm is employed to optimize the results.

Differential Evolution Algorithm:

Step 1. The foremost step is the random initialization of the parent population. Randomly generate a population of NP vectors, each of n dimensions:
$x_{i,j} = x_{min,j} + rand(0, 1)(x_{max,j}-x_{min,j})$, where $x_{min,j}$ and x_{max} are lower and upper bounds for j^{th} component respectively, rand(0,1) is a uniform random number between 0 and 1.

Step 2. Evaluate the objective function value $f(X_i)$ for all X_i.

Step 3. *C*hoose three points from population and generate perturbed individual V_i using equation (1.1).

Step 4. Recombine the each target vector x_i with perturbed individual generated in step 3 to generate a trial vector U_i using equation (2).

Step 5. Check whether each variable of the trial vector is within range. If yes, then go to step 6 else make it within range using $u_{i,j} = 2* x_{min,j} - u_{i,j}$,if $u_{i,j} < x_{min,j}$ and $u_{i,j} = 2* x_{max,j} - u_{i,j}$, if $u_{i,j} > x_{max,j}$, and go to step 6.

Step 6. Evaluate the objective function value for vector U_i.

Step 7. Choose better of the two (function value at target and trial point) using equation (3) for next generation.

Step 8. Ensure that whether the convergence criterion is met if yes then stop; otherwise go to step 3.

DE generates new candidates by adding a weighted difference between two population members to a third member. If the resulting candidate is superior to the candidate with which it was compared, it replaces it; otherwise, the original candidate remains unchanged.

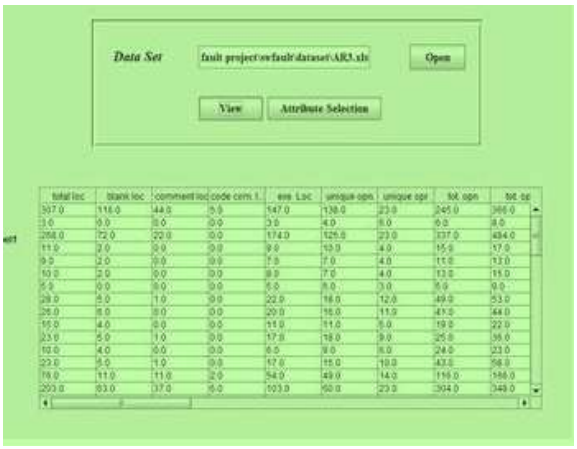
RESULTS AND DISCUSSION

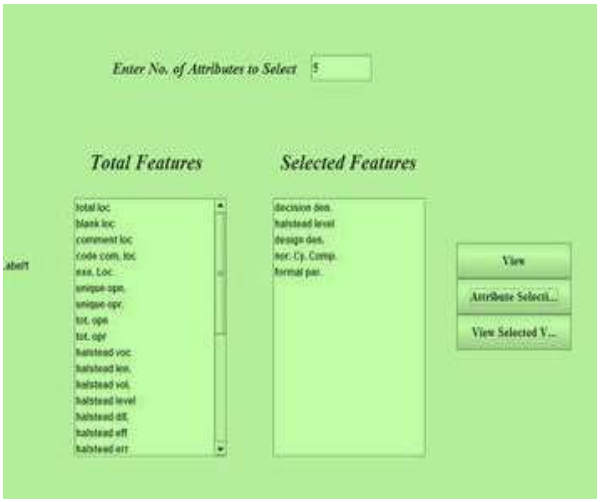
The Java program is considered as input to the system where the code is converted in to datasets using Halsted. The significant of optimized error detection system is select some attributes and process data sets through WEKA tool. The most important purpose of using WEKA tool is to improve the efficiency and speed. Initially a threshold value will be fixed in DE algorithm. After that classification is done on 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. The table1 shows the AR3 sample dataset. The figure2 shows the results obtained through the proposed system.

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

Table 1: The AR3 Sample Dataset

Executable LOC	Non-executable LOC	Blank LOC	Code com. LOC	Unique Operator.	Total Operator.	Unique Operation.	Total Operation.	Total LOC
147	44	116	5	23	366	138	245	307
3	0	0	0	6	8	4	6	3
174	22	72	0	23	484	125	337	268
9	0	2	0	4	17	10	15	11
7	0	2	0	4	13	7	11	9
8	0	2	0	4	15	7	13	10
5	0	0	0	3	9	5	5	5
22	1	5	0	12	53	18	49	28
20	0	6	0	11	44	16	41	26
11	0	4	0	5	22	11	19	15
17	1	5	0	9	36	18	25	23
6	0	4	0	6	23	9	24	10





2(a) ***2(b)*** ***2(c)***

Figure 2(a) Dataset and Attributes, 2(b) Selected Attributes, 2(c) Selected Attributes with Values

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

CONCLUSION

Software flaw forecasting scheme plays vital role in improving the quality and effectiveness of software development. The Standard dataset is collected from AR3, which contains large number of attributes. The weka tool is used to reduce the attributes and DE algorithm is utilized to optimize the results. The observed results indicate that the proposed system is highly effective in defect prediction. However, a combination of different data sources can employ to get more improved prediction results.

REFERENCES

[1] J. McQueen, "Some Methods for Classification and Analysis of Multivariate Observations," Proc. Fifth Berkeley Symp. Math. Statistics and Probability, pp. 281-297, 1967.
[2] H. Spath, Cluster Analysis Algorithms for Data Reduction and Classification of Objects. Chichester: Ellis Horwood,1980.
[3] A.K. Jain and R.C. Dubes, Algorithms for Clustering Data. Englewood Cliffs, N.J.: Prentice Hall, 1988.
[4] R. Storn, K. Price, "DE-a Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Space", Technical Report TR-95-012, ICSI, March 1995.
[5] Fayyad, U., Piatetsky-Shapiro, G., and Smyth, R (1996). "The KDD Process for Extracting Useful Knowledge from Volumes of Data," Communications of the ACM, (39:11), pp.27-34.
[6] Storn, R. and Price, K. (1997), 'Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces', Journal of Global Optimization, 11, pp. 341–359.
[7] Hand, D. J. (1998), "Data Mining: Statistics and More?", The American Statistician, May (52:2), 112-118.
[8] A.K. Jain, M.N. Murty, and P.J. Flynn, "Data Clustering: A Review," ACM Computing Surveys, vol. 31, no. 3, 1999.
[9] Price, K.V. (1999), 'An Introduction to Differential Evolution' in Corne,D., Dorigo, M. and Glover, F. (eds), New Ideas in Optimization, McGraw-Hill, London.
[10] Han, J., Kamber, M. (2001), Data Mining: Concepts and Techniques, Morgan-Kaufmann Academic Press, San Francisco.
[11] Price, K.; Storn, R.M.; Lampinen, J.A. (2005). Differential Evolution: A Practical Approach to Global Optimization . Springer. ISBN 978-3-540-20950-8.
[12] Feoktistov, V. (2006). Differential Evolution: In Search of Solutions . Springer. ISBN 978-0-387-36895-5.
[13] 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.
[14] Michael Laszlo and Sumitra Mukherjee, Member, IEEE, "A Genetic Algorithm Using Hyper-Quadtrees for Low-Dimensional K-means Clustering", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 28, no. 4, April 2006.
[15] Laird LM, Brennan MC. Software measurement and estimation: a practical approach. IEEE Comput Soc 2007.
[16] 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.
[17] S. Lessmann,B.Baesens, C.Mues, and S. Pietsch,"Benchmarking Classification Models for Software Defect Prediction: A Proposed Framework and Novel Findings", IEEE Transactions on Software Engineering, (2008).
[18] P.Singh," Comparing the effectiveness of machine learning algorithms for defect prediction", International Journal of Information Technology and Knowledge Management, 2009 ,pp. 481-483.
[19] Arisholm E, Briand LC, Johannessen EB. A systematic and comprehensive investigation of methods to build and evaluate fault prediction models. J Syst Softw 2010;83(1):2–17.
[20] Azeem, N., Usmani, S.: Analysis of Data Mining Based Software Defect Prediction Techniques. Global Journal of Computer Science and Technology 11(16) Version 1.0, September 2011.
[21] Partha Sarathi Bishnu and Vandana Bhattacherjee, "Software Fault Prediction Using Quad Tree-Based K-Means Clustering Algorithm", IEEE Transactions on knowledge and data engineering, Vol. 24, no. 6, June 2012.
[22] Hall T, Beecham S, Bowes D, Gray D, Counsell S. A systematic literature review on fault prediction performance in software engineering. Software Engineering, IEEE Transactions on 2012;38(6):1276–304.
[23] Radjenovic´ D et al. Software fault prediction metrics: a systematic literature review. Inf Softw Technol 2013;55(8):1397–418.

[24] Ahmed H. Yousef , "Extracting software static defect models using data mining", Ain Shams Engineering Journal (2015) 6, PP 133–144.
[25] G.Abaeia, A.Selamata, H.Fujitab, "An empirical study based on semi-supervised hybrid self-organizing map for software fault prediction", Knowledge-Based Systems, vol. 74, (2015), pp. 28-39.
[26] I. H. Laradji, M.Alshayeb, L.Ghouti, "Software defect prediction using ensemble learning on selected features. Information and Science Technology", vol. 58, (2015), pp. 388-402.
[27] W. Zhang, Y. Yang, Q. Wang, "Using Bayesian Regression and EM algorithm with missing handling for software effort prediction", Information and software technology, vol. 58, (2015), pp. 58-70.
[28] Fahimeh Sadat Fazel, A New Method to Predict the Software Fault Using Improved Genetic Algorithm, Bulletin de la Société Royale des Sciences de Liège, Vol. 85, 2016, pp 187 – 202.
[29] Sonu Kumar Kushwaha, Sunil Malviya, "A Study on Software Defect Prediction by Data Mining Techniques", IJMERT, Vol-3, Issue-5 ,pp 284-286, 2016.
[30] Sajna.P, Shahad.P., "Software Fault Prediction Methods : A Brief Survey", Recent Trends in Computational Intelligence & Image Processing - RICP 2017, Calicut , International Journal of Science and Research, pp 58-62.
[31] Jyoti Devi and Nancy Seghal, "A Review of Improving Software Quality using Machine Learning Algorithms", International Journal of Computer Science and Mobile Computing, Vol.6 Issue.3, March-2017, pp. 148-153.
[32] www. en.wikipedia.org/wiki/Software_bug
[33] www.cs.umd.edu/projects/SoftEng/ESEG/papers/82.78.pdf.

CITE AN ARTICLE:

Periyasamy, A. R., Dr, & Misbahulhuda, A., Mrs. (2017). OPTIMIZED SOFTWARE FLAW FORECASTING SCHEME BASED ON DATA MINING AND DIFFERENTIAL EVOLUTION. *INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY, 6*(5), 116-124. doi:10.5281/zenodo.571738