IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-9, Sept. 2018 E-ISSN: 2347-2693

Optimized Machine Learning Approach for Software Defect Prediction using K-means with Genetic Algorithms

Manjula C[1*], Lilly Florence[2]

[1]MCA Department, PESIT- BSC, Karnataka.
[2]MCA Department, Adiyamman College of Engineering, Tamil Nadu

*Corresponding author: manjulaprasad@pes.edu

Available online at: www.ijcseonline.org

Accepted: 22/Sept/2018, Published: 30/Sept/2018

Abstract- Software defect prediction is one of the most active research areas in software engineering. Machine learning approaches are good in solving these. A predictive model is constructed by using machine learning approaches and classified them into defective and non-defective modules. Clustering is an unsupervised classification method aims at creating groups of objects, or clusters, in such a way that objects in the same cluster are very similar and objects in different clusters are quite distinct. In this paper we proposed a new hybrid approach of K-means clustering algorithm combined with Genetic Algorithm to get the optimum no of clusters. From the present studies it is shown that the performance of the proposed optimized hybrid algorithm is better than the conventional k-means algorithm without optimization.

Keywords: Unsupervised classifier, Clustering, K-means, Genetic Algorithm, Software Defect Prediction

1. INTRODUCTION

Software Defect Prediction is one of the most active research areas in software engineering [1], [2], [3], [4], [5], [6]. Since defect prediction models provide the list of bug-prone software artifacts, quality assurance teams can effectively allocate limited resources for testing and investigating software products [2], [3], [6]. The advancement in software technology causes an increase in the number of software products, and their maintenance has become a challenging task. More than half of the life cycle cost for a software system includes maintenance activities. With the increase in complexity in software systems, the probability of having defective modules in the software systems is getting higher [7].

Machine learning techniques can be used to analyse data from different perspectives and enable developers to retrieve useful information. The machine learning techniques that can be used to detect bugs in software datasets can be classification and clustering. It involves categorization of software modules into defective or non-defective that is denoted by a set of software complexity metrics by utilizing a classification model that is derived from earlier development projects data [8]. The metrics for software complexity may consist of code size [9], McCabe's Cyclomatic Complexity [10] and Halstead's Complexity [11].

Unsupervised classifiers make use of clustering methods. Clustering is classified under unsupervised learning approach because no class labels are provided. Clustering is a kind of non-hierarchal method that moves data points among a set of clusters until similar item clusters are formed or a desired set is acquired. The data is grouped together on the basis of their similarity. K-mean clustering is based on non-hierarchical clustering procedure and item are moved within sets of clusters until the desired set is reached. The method of K-means clustering is a partition-clustering algorithm that puts together a set of objects into k clusters by means of optimizing a standard function [12][13].

Genetic algorithm[GA] which proposed early in 1989 [14],[15] is search heuristic usually applied in the optimization problems[16] guided by the principles of evolution and natural genetics [17-24]. GA belongs to the larger class of evolutionary algorithms which engender solutions to optimization problems using techniques inspired by natural evolution such as inheritance, mutation, selection and crossover [25] .In GA, the populace of candidate solutions to an optimization problem is evolved towards better solutions [26],[27]. Each candidate solution has properties which can be mutated and altered [27]. The basic elemental artistry of the GAs are designed to mimic processes in natural systems necessary for evolution, the principle stated by Charles Darwin "Survival of Fittest"[17],[22]. GAs imitates the survival of the fittest individual over successive generations for solving any problem [17]. Each generation consist of string of characters which are similar to the chromosome of the DNA. Each individual is the possible solution of the problem basis of fitness of each individual the individual with maximum fitness forms the solution of the problem. The GAs is analogous to the biological genetic structure [15]. Rest of the article is organized as follows…Section 2 describes recent studies in the field of software defect prediction, proposed model is presented in section 3, experimental analysis is demonstrated in section 4 and finally concluding remarks are given in section 5.

2. REVIEW OF LITERATURE WORK

There are a few software fault prediction studies which do not use prior fault data for modeling. Researchers [28], [29] conduct to combine k-means along with genetic algorithm to succeed in getting optimal solution and to come out of the local optimum. Zhanqing Lu et al. [30] presented a modern Algorithm which brings K-means and genetic algorithm to find solution for the problem of Multiple Traveling Salesman. Babaie et al. [31] provided a novel combining genetic algorithm and K-means having cultural goods within the limits of budget for the family.K-means algorithm is an iterative algorithm to get the highest quality clusters of objects, but determination over the variety on clusters is the subject for which GA execute keep useful. Rahman and Islam. [28] Presented a novel clustering technique that combines both K –means and genetic algorithm together called GenClust technique aims to gain better quality clusters without any need for user inputs like number of clusters k.

3. METHODOLOGY

This section presents details about the methodology used such as Clustering using K-means, Genetic Algorithm and finally proposed K-means-GA optimized hybrid model for software defect prediction.

3.1 Clustering:

Clustering is an unsupervised learning approach. It locates in indirect data mining/machine learning group and classification area locates in direct data mining/machine learning group. While classification uses class labels for training, clustering does not use class labels and tries to discover relationships between the features . Clustering methods can be used to group the modules having similar metrics by using similarity measures or distances. After the clustering phase, the mean values of each metric within cluster can be checked against industrial metrics thresholds. If the limits are exceeded, the cluster can be labelled as fault-prone.

K-means: One of the simplest clustering algorithms is K-means clustering method.

The pseudo code of this algorithm is given as follows [32]:

```
"Require: Dataset D, number of clusters k, Dimension d:
{ Ci is the ith cluster }

{ 1. Initialization Phase}
1: (C1, C2, …, Ck} = Initial partition of D.
{ 2. Iteration Phase}
2: repeat
3: dij = distance between case i and cluster j;
4: ni = argmin dij;
5: Assign case i to cluster ni;
6: Recompute the cluster means of any changed clusters;
7: until no further changes of cluster membership occur
8: Output results".
```

In the initialization phase, clusters are initialized with random instances and in the iteration phase, instances are assigned to clusters according to the distances, computed between the centroid of the cluster and the instance. This iteration phase goes on until no changes occur in the clusters.

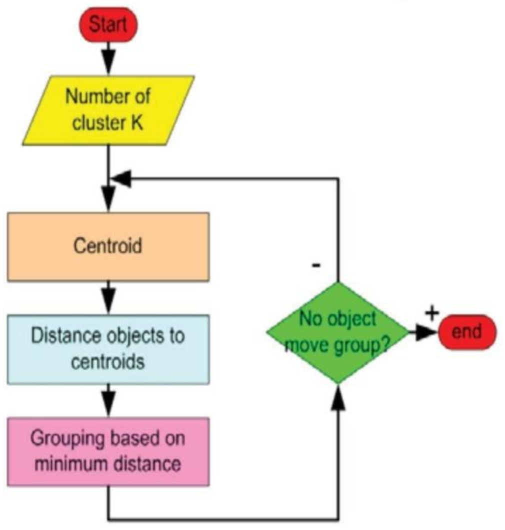
Start
Number of cluster K
Centroid
Distance objects to centroids
Grouping based on minimum distance
No object move group?
-
+
end

Figure 1: Flowchart of K-means

3.2 Genetic Algorithm:

Genetic algorithm works on the notion of coding parameter sets rather than parameters themselves [16]. The encoded parameter sets are known as chromosomes. GAs computes the optimization problems using population of fixed size known as population size [16]. A solution consists of string of symbols quintessential binary symbols. The more fit members of this population are more likely to mate and produce the next generation. As the generation pass, the members of the population get closer and closer to the solution.

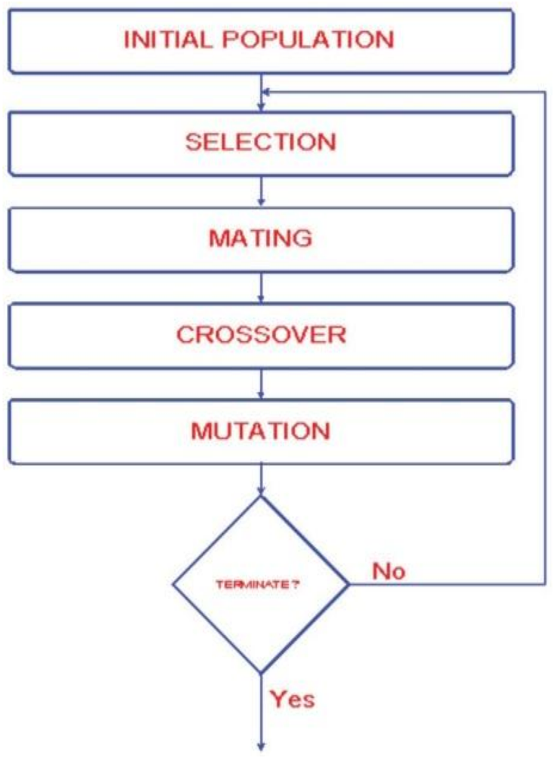

Figure2: Flowchart of Genetic Algorithm

The pseudo code of this algorithm is given as follows:

```
1. t=0
2. While t<T or termination criteria not meet
   do
3. Compute the fitness factor of P (t).
4. Select $P_b$ the fitted solution for next generation.
5. t=t+1
6. Perform crossover to generate new solution.
7. Perform mutation to the solutions.
8. End while.
```

3.3 Proposed Methodology- GA-K-means:

The function of GA-K-means is to determine the optimal weights of the attributes and cluster centers of clusters that are needed to classify the dataset. The selection is biased toward more highly fit individuals, so the average fitness of the population to improve from one generation to the next. In general GA generates an optimal solution by means of using reproduction, crossover, and mutation operators [33],[34]. The genetic algorithm initially start with population generated, population is the collection of chromosomes, chromosome is the collection of genes, the fitness for the population is calculated by using a suitable fitness function accordingly. In GA-K-means the result of K-means algorithm is used for setting the objective function of GA. If fitness value is satisfied, the best solution is obtained. Otherwise the GA parameters (reproduction, crossover, mutation) are apply for obtain a optimal no. of cluster.

Algorithm for GA-K-means is given below..

```
1.  Solution string, s*;
2.  { Initialize the population, P;
3.  geno = MAX-GEN;
4.  s* = P1; (Pi is the length in P)
5.  While (geno> 0)
6.  s* = P1; (Pi is the ith string in Pi)
7.   P = Selection (P);
8.  for i = 1 to N, Pi = Mutation (Pi);
9.  for I = 1 to N, K-means (Pi);
10. S = string in P such that the corresponding weight
    matrix Ws has the minimum SE measures;
11.  If (S(Ws)) > S(Ws)), s* = S;
        Geno =  geno-1;
            }
            Output s*;
            }
```

4. PERFORMANCE MEASUREMENT PARAMETERS:

For performance measurement, a well known technique is used. This technique is called as confusion matrix analysis. According to this matrix, actual class and predicted class are stored to obtain the classification results. A sample representation of confusion matrix is given in Table1.

Table 1. Confusion Matrix

Actual class	Predicted class	
	Non defective	Defective
Non Defective	False negative (FN)	True Positive (TP)
Defective	True Negative (TN)	False Positive (FP)

This confusion matrix helps us to compute total accuracy, precision, specificity, sensitivity and

F-measure of the proposed approach.

Accuracy is a measurement of rate of correct classification which is denoted by Acc. It is computed by taking the ratio of correct prediction and total number of prediction. It can be expressed as:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \qquad (1)$$

Another parameter is known as sensitivity analysis of the model. This is the measurement of true positive rate which can be computed by identifying the correctly classified non-defective modules. This can be expressed as:

$$Sensitivity = \frac{TP}{TP + FN} \qquad (2)$$

Next parameter is computed as true negative rate which shows the measurement of correct classified defective software modules and can be expressed as:

$$Specificity = \frac{TN}{TN + FP} \qquad (3)$$

Then, we compute Precision of the proposed approach. It is computed by taking the ratio of True Positive and (True and False) positives.

$$P = \frac{TP}{TP + FP} \qquad (4)$$

Finally, F-measure is computed which is the mean of precision and sensitivity performance. It is expressed as:

$$F = \frac{2 * P * Sensitivity}{P + Sensitivity} \qquad (5)$$

5. EXPERIMENTAL RESULTS & DISCUSSIONS

We applied our proposed approach to public repository datasets, since we wanted to compare our proposed approach with the conventional approach and it is easy to conduct replication or future research. We have considered AEEEM[35] software defect dataset repository.

We present the performance analysis of various parameters such as precision, sensitivity, specificity, F-measure and accuracy of conventional K-means algorithm in Table3. We also present the performance analysis of various parameters such as precision, sensitivity, specificity, F-measure and accuracy of the proposed optimized K-means algorithm with genetic algorithm in Table4. This comparative analysis shows that proposed approach achieves better performance when compared with conventional software defect technique.

Table 2:Data Set

Database	Total Class	Total versions	Transactions	Post release defect
Eclipse JDT core	997	91	9135	463
Eclipse PDE UI	1562	97	5026	401
Equinox framework	439	91	1616	279
Mylyn	2196	98	9189	677
Lucene	779	99	1915	403

Table 3 Performance analysis of Eclipse datasets using conventional K-means Algorithm.

Measurement Parameter	Eclipse JDT core	Equinox framework	Eclipse PDE UI	Mylyn	Lucene
Precision	0.45	0.48	0.51	0.57	0.47
Sensitivity	0.48	0.46	0.56	0.35	0.43
Specificity	0.51	0.48	0.48	0.41	0.39
F-Measure	0.38	0.33	0.51	0.47	0.41
Accuracy%	64	67	59	60	65

Table 4 Performance analysis of Eclipse datasets using Proposed optimized K-means with Genetic Algorithm.

Measurement Parameter	Eclipse JDT core	Equinox framework	Eclipse PDE UI	Mylyn	Lucene
Precision	0.51	0.48	0.56	0.48	0.53
Sensitivity	0.57	0.49	0.58	0.48	0.49
Specificity	0.48	0.52	0.53	0.51	0.46
F-Measure	0.43	0.49	0.57	0.56	0.49
Accuracy%	75	79	65	66	71

6. CONCLUSION

In this work, we have focused on software defect prediction using machine learning techniques. A novel approach optimization of K-means with Genetic algorithm for early defect prediction is presented. Genetic algorithm is used along with k-means clustering to produce the fittest result. The results shows that the genetic K-means clustering outperforms the k-means clustering in terms of predicting and classifying . Performance study of the proposed model is carried out using AEEEM public dataset repository using MATLAB. Proposed approach performance is compared with the conventional K-means in terms of classification accuracy. This experimental analysis shows that proposed approach is capable of achieving enhanced classification performance.

REFERENCES

[1] M. D'Ambros, M. Lanza, and R. Robbes. An extensive comparison of bug prediction approaches. In Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on, pages 31 –41, May 2010.

[2] T. Lee, J. Nam, D. Han, S. Kim, and I. P. Hoh. Micro interaction metrics for defect prediction. In SIGSOFT '11/FSE-19: Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering, 2011.

[3] T. Menzies, J. Greenwald, and A. Frank. Data mining static code attributes to learn defect predictors. IEEE Trans. Softw. Eng., 33:2–13, January 2007

[4]J. Nam, S. J. Pan, and S. Kim. Transfer defect learning. In Proceedings of the 2013 International Conference on Software Engineering, ICSE '13, pages 382–391, Piscataway, NJ, USA, 2013. IEEE Press.

[5] F. Rahman, D. Posnett, A. Hindle, E. Barr, and P. Devanbu. Bugcache for inspections: Hit or miss? In Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering, ESEC/FSE '11, pages 322–331, New York, NY, USA, 2011. ACM.

[6] T. Zimmermann and N. Nagappan. Predicting defects using network analysis on dependency graphs. In Proceedings of the 30th international conference on Software engineering, ICSE '08, pages 531–540, 2008.

[7] F. Akiyama. An Example of Software System Debugging. In Proceedings of the International Federation of Information Processing Societies Congress, pages 353–359, 1971.

[8] R. Spiewak & K. McRitchie (2008) "Using software quality methods to reduce cost and prevent defects", Journal of Software Engineering and Technology, pp. 23-27.

[9] D. Shiwei (2009) "Defect prevention and detection of DSP-Software", World Academy of Science, Engineering and Technology, Vol. 3, Issue 10, pp. 406-409.

[10] P. Trivedi & S. Pachori (2010) "Modelling and analyzing of software defect prevention using ODC", International Journal of Advanced Computer Science and Applications, Vol. 1, No. 3, pp. 75- 77.

[11] T. R. G. Nair & V. Suma (2010) "The pattern of software defects spanning across size complexity", International Journal of Software Engineering, Vol. 3, Issue 2, pp. 53- 70.

[12]Lloyd, S. (1982). Least squares quantization in PCM. Information Theory, IEEE Transactions on 28(2): 129-137.

[13] Forgy, E. W. (1965). Cluster analysis of multivariate data: efficiency versus interpretability of classifications. Biometrics 21: 768-769

[14]Pal, Sankar K., Dinabandhu Bhandari, and Malay K. Kundu. "Genetic algorithms for optimal image enhancement." Pattern Recognition Letters, Vol.15 (3), pp. 261-271, 1994

[15] Zheyun Feng "Data Clustering using Genetic Algorithm" Evolutionary Computation: Project Report, CSE484, 2012.

[16] Dash, B., Mishra, D., Rath, A., & Acharya, M., "A hybridized K-means clustering approach for high dimensional dataset", International Journal of Engineering, Science and Technology, Vol.*2* (2), pp.59-66, 2010.

[17]D.E. Goldberg "Genetic Algorithms in Search Optimization and Machine Learning", Addison-wesley, New York-1989.

[18]L. Davis (Ed.), Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, 1991.

[19]Z. Michalewicz "Genetic Algorithms Data Structure" Evolution Programs, Springer, New York, 1992.

[20]Ribeiro Filho, José L., Philip C. Treleaven, and Cesare Alippi. "Genetic- algorithm programming environments." Computer 27, Vol. 6, pp. 28-43. 1994.

[21] Pal, Sankar K., Dinabandhu Bhandari, and Malay K. Kundu. "Genetic algorithms for optimal image enhancement." Pattern Recognition Letters, Vol.15 (3), pp. 261-271, 1994.

[22]Maulik, Ujjwal, and Sanghamitra Bandyopadhyay. "Genetic algorithm- based clustering technique." Pattern recognition, Vol.33 (9), pp.1455- 1465, 2000.

[23] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann Publisher, San Francisco, USA,2001.

[24] Chiou, Yu-Chiun, and Lawrence W. Lan. "Genetic clustering algorithms." European journal of operational research, Vol.135 (2), pp. 413-427, 2001.

[25] Xu, Rui, and Donald Wunsch. "Survey of clustering algorithms." IEEE Transactions on neural networks, Vol.16(3), pp. 645-678, 2005

[26] Maulik, Ujjwal, and Sanghamitra Bandyopadhyay. "Genetic algorithm- based clustering technique." Pattern recognition, Vol.33 (9), pp.1455- 1465, 2000.

[27] Dash, Rajashree, and Rasmita Dash. "Comparative analysis of k-means and genetic algorith based data clustering." International Journal of Advanced Computer and Mathematical Sciences, Vol.3 (2), pp.257-265, 2012.

[28] Anon., *Investigating the Performance of Parallel Genetic Algorithms*.

[29] Wu, F.-X., W. Zhang, and A. Kusalik, *A genetic k-means clustering algorithm applied to gene expression data*. Advances in Artificial Intelligence, 2003: p. 994-994.

[30] Graña, M., et al., *International Joint Conference SOCO'16-CISIS'16-ICEUTE'16: San Sebastián, Spain, October 19th-21st, 2016 Proceedings*. Vol. 527. 2016: Springer

[31] Lu, Z., et al. *Applying K-means Clustering and Genetic Algorithm for Solving MTSP*. in *Bio-Inspired Computing-Theories and Applications*. 2016. Springer.

[32] G. Gan, C. Ma, J. Wu, "Data clustering: theory, algorithms, and applications", Society for Industrial and Applied Mathematics, Philadelphia, 2007

[33] D. Goldberg, Genetic Algorithm in Search , Optimization and Machine Learning, Addison Wesley, 1989.

[34] Z. Michalewicz, Genetic Algorithms + Data Structures =Evolution Programs, 3rd ed., Springer-Verlag, 1999.

[35] M. D'Ambros, M. Lanza, and R. Robbes, "An extensive comparison of bug prediction approaches," 2010 7th IEEEWorking Conference on Mining Software Repositories (MSR 2010), pp.31–41, IEEE, 2010.

Author's Profile

Manjula.C, from Bangalore, Karnataka India. Has completed Bachelor of Science, Master of Computer Applications and MPhil in Computer Science. Having 17 years of Teaching experience and 5 years of Industry experience. Currently working as an Associate Professor at PES Institute of Technology Bangalore South Campus, Bangalore, Karnataka, India. Has 7 publications in National/ International Journals. Has organized more than 10 workshops and seminars.

M. Lilly Florence, from Hosur, Tamil Nadu has completed her Bachelor Degree in Mathematics and Master degree MCA at Manonamaniam Sundarnar Unviersity and did her M.Tech.(IT) at Punjabi University and completed her Doctorate in Computer Science during at Mother Theresa Womens University. She has 17 years of teaching experience. She is good in teaching all Programming Languages. Prof. Lilly Florence has published 20 research papers in National and International Journals, also she has published 24 Papers in various National and International Conferences. She is an author of three text books namely, Operating Systems, Computer Graphics and Multimedia and Computer Architecture and Organization. Prof. Lilly has organized more than 20 workshops, seminars for various groups of audience. She has visited more than 20 colleges as a Technical Resource Person. She has received grants from DRDO, DST, ISRO, etc to organize FDP and seminars. She is acting as a Computer Society of India Student Branch Counselor for Adhiyamaan College of Engineering. In Research, she is a recognized supervisor of Periyar University and Bharathiyar University. She has produced one Ph.D Scholar and currently she is guiding 6 Ph.D scholars. Dr. Lilly has undertaken 2 research projects funded by Department of Science and Technology for Rs. 23.00 lakhs. Prof. Lilly is a life member of Computer Society of India and ISTE. Also she is a BOS member of MCA board.