New Dataset for Software Defect Prediction Model

1st Jameel Alsaraireh
Management Information Systems Department
Cyprus International University
Haspolat-Lefkosa, Turkey
21908602@student.ciu.edu.tr

2nd Assoc. Prof. Dr. Mary AGOYI
Management Information Systems Department
Cyprus International University
Haspolat-Lefkosa, Turkey
magoyi@ciu.edu.tr

***Abstract*—Software defect prediction is one of the critical fields that is related to software quality. The accuracy of prediction models relies on how the features are related to class value. However, this field has limitations of the lack of robust dataset in previous studies. Therefore, this research paper aims to develop new dataset to fill the gap of research area. So, we train the J48, NB, and MLP algorithms on our new dataset, JM1and KC1. The results indicated that our proposed new dataset performed better in terms of accuracy, recall, and precision.**

Keywords—Software Defect, Machine Learning, metrics

I. INTRODUCTION

Software defect prediction is one of the most important areas of research in the field of software quality. Defect prediction in software is the technique of detecting areas of a software system that may have flaws. Utilizing Defect Prediction models early in the software lifecycle allows practitioners to focus their testing personnel so that areas of the software system that are known to be resistant to defects are tested more thoroughly than other parts of the software system.[1]. This reduces the cost of resources during production while also easing the maintenance effort. Software Metrics, which are observable aspects of the software system, and fault data from a linked software project are used to develop defect prediction models in two methods. The defect prediction model can then be used to forecast problems in future software projects, allowing practitioners to identify elements of a software system that are vulnerable to defects [2].

Today, as more complex software systems evolve, the incidence of software defects can be said to be increasing proportionally. These flaws have the potential to trigger serious issues in critical projects. Not only is software defect detection a time-consuming operation, but it also lacks a standard tool. There have been attempts to standardize software quality measurements, and ISO/IEC 9126 is one of them is introduced to define software quality attributes. Furthermore, such techniques may be used to evaluate the set of metrics that should be considered when determining whether a software flaw exists. The overall cost of the project would be lower if faulty modules could be predicted, and the project's performance rate would be higher [3].

Depending at the defect prediction goals, the machine-learning strategies used to differ. The prediction module's defect orientation is centered after the researchers have hooked up numerous is about to first-class grain (for example, record stage or class), that is generally finished the use of category methods together with LR, Bayesian networks, and DM trees. Setting the module to coarse grain (for instance, on the packet or subsystem stage) reduces the variety of mistakes or flaws within the prediction module, which is generally carried out through regression analysis. In addition to conventional learning strategies, lively learning and semi-supervised learning are gaining popularity [4].

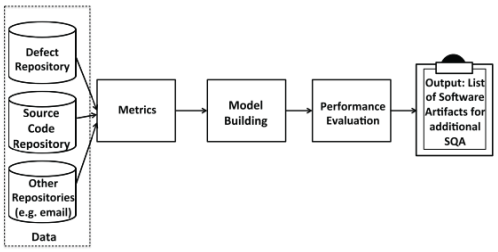

Fig. 1. Overview of Software Defect Prediction

II. RELATED WORKS

Many studies have been investigated in field of SDP to improve the results of such prediction models. In [5], eenhanced software prediction was achieved by a built-in machine learning algorithm focused on the regression method developed by using a collection of Factor predictors. By the amount of faults, They defined the combination of each predictor variable, they used 10 PROMISE datasets including a total of 22,838 samples. The regression method induces a p-value of < 0.001 with a modified R-square of 98.6 percent. We often apply a systematic design simulation approach to predict the number of computer defects. We find various combinations of predictor factors, including amount defect velocity, amount defect intake period, and amount defect intensity.

This study [6] presented the features with automatic predicting systems from a viewpoint of controls reliability and supervised learning and the analysis of imbalanced NASA sets of data (JM1, KC3, MC1). They were used for research on Bagging Bayesian Belief Network, LWL, Random Forest, C4.5, Multilayer Feedforward Neural Network, NB-K and SVM algorithms, the SMO algorithm has an optimal value of 0.716, BBN of 0.704, and Random forest of 0.656, Bagging + Random Forest (classifier) of 0.707.

In [7], a hyper quadtree-based K-means algorithm was used to predict program module faults. This paper is divided into two parts. First, the hyper-quadtree is used to initialize the K-means clustering algorithm on the software fault prediction dataset. The initial number of clusters and cluster centers is controlled by an input parameter D. Second, the initialization algorithm's cluster centers and number of cluster centers are used as input for the K-means clustering algorithm, which predicts faults in software modules. We propose a hyper-quadtree-based K-means algorithm for predicting software flaws. The overall goal of this paper is to demonstrate how to use K-means without specifying the number of clusters or the

initial cluster centers to predict software faults with a low error rate. We can get the initial cluster centers and the number of clusters by varying the value of D. The proposed algorithm is evaluated in comparison to various existing techniques. The overall error rate of the HQDK algorithm's software fault prediction approach is comparable to other existing algorithms, and HQDK has a lower error rate. At the same time, HQDK's accuracy is superior to other techniques summarize the main related works.

Database	Classifier	Label	Result	Reference
PROMISE dataset	Regression	Defective or non-defective	R-square=98.6%, and a p-value <0.001	[5]
(JM1, KC3, MC1)	Random forest, Bagging, SMO	Defect or No defect	The ensemble-based learning algorithm Bagging+Randomforest (classifier) has strong classification capabilities.	[6]
AR3, AR4, and AR5. PROMISE Software Engineering Repository	a hyper quadtree-based K-means algorithm has been applied for predicting the faults in the program module	Faulty or non-faulty	The total error rate of software failure prediction using the HQDK approach is comparable to other algorithms.	[7]

III. Drawbacks and Limitations

By reviewing the literature, we found that authors had limitations in their papers, and they recommended to overcome them in future works and those limitations are as follows:

First: future studies can validate this method for estimating the quantity of faults in an imminent product launch the use of the maximum latest datasets from any software program company, at the same time as additionally deliberating extra predictor variables [8], Trying out more attribute selection techniques and classifiers to see how they stack up against their learner. examine their student in a variety of areas, including company credit score and breast cancer disease.

Second: they want to enhance our DBN-based methodology to produce semantic characteristics for method-level defect prediction, which can assist predict defective methods in software projects [9].

For this, by reviewing and analyzing previously used datasets such as JM1, CM1, and KC1 which are mostly used by authors and researchers in this filed. Then, we select the main and the common attributes of those dataset to develop new dataset. The new dataset was extracted from a software company in Jordan by advising from 3 experts' developers.

Our new dataset consists of 17 attributes in addition to class label as shown in Tabel II. Therefore, these features were selected from different datasets that resulting in increased the correlation between the attributes and class label.

TABLE I. NEW DATASET MAIN FEATURES

Attribute	Attribute Information
# Of linearly independent paths	"Cyclomatic Complexity"
Reduced flow graph	"Essential complexity"
DC	"Design complexity"
X	"Number of operators + operands"
NOO	"volume"
PL	"Program length"
D	"difficulty"
Int	"intelligence"
EFF	"effort"
B	"Numeric"
TE	"Time estimator"
LC	"Count of lines of comments"
BC	"Flow graph"
T_O	"Total operands"
T_OR	"Total operators"
U_O	"Unique operands"
U_OR	"Unique operators"

Then, the correlation between the features and class label was tested using SPSS method. The results indicated that the feature "U_O" is the most correlated attribute with class label of 0.2478. Nevertheless, the "EFF" and "TE" features are the least correlated to class label of 0.0948 for both (see Table III and Fig. 2).

TABLE II. CORRELATION RESULTS BETWEEN THE MAIN FEATURES AND CLASS LABEL

Attribute	Correlation
# Of linearly independent paths	0.2017
Reduced flow graph	0.1542
DC	0.1779
X	0.2328
NOO	0.2129
PL	0.1728
D	0.2058
Int	0.2368
EFF	0.0948
B	0.2131
TE	0.0948
LC	0.1316
BC	0.2141
T_O	0.2325
T_OR	0.229
U_O	0.2478
U_OR	0.2116

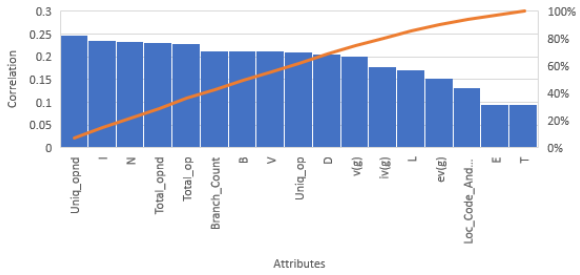

Fig. 2. The main results of attributes' correlation

IV. Results and Discussion

To evaluate and analyses our dataset, this study has used the main classical algorithms that have used in previous works. Therefore, we trained the JM1and KC1 datasets by using J48, NB, and MLP algorithms. Also, this study has trained the new dataset on the same algorithms. Then, we compare the results of J48, NB, and MLP algorithms on such datasets. The results show that our proposed new dataset outperformed the previous ones in the term of accuracy with value of 81.06%,81.19%, and 81.72% respectively for J48, NB, and MLP as shown in the tables below.

TABLE III. JM1 DATASET

	Precession	Recall	Accuracy
J48	0.759	0.795	79.504 %
NB	0.765	0.804	80.423 %
MLP	0.769	0.810	80.956 %

TABLE IV. KC3 DATASET

	Precession	Recall	Accuracy
J48	0.790	0.805	80.5 %
NB	0.773	0.785	78.5 %
MLP	0.752	0.775	77.5 %

TABLE V. NEW DATASET

	Precession	Recall	Accuracy
J48	0.766	0.811	81.06%
NB	0.772	0.811	81.19%
MLP	0.779	0.817	81.72%

V. Conclusion and Future Works

Artificial Intelligence (AI) is the science and engineering of enabling machines to demonstrate intelligence in areas such as visual identification, speech recognition, and decision-making. In essence, it is the artificial version of human intelligence done by machines, in particular computer systems [9]. This study aims to propose new dataset in order to enhance the accuracies of prediction models in SDP field. The new dataset contains 17 features as long as class label. Then we applied the classical algorithm on our dataset and compare the results with others dataset from previous works. The results have showed that our new dataset outperformed the others previous datasets. In the future works, this study recommends using other advanced techniques such as ensemble methods [10] or deep learning techniques [11].

References

[1] S. S. Rathore and S. Kumar, "A study on software fault prediction techniques," *Artificial Intelligence Review*, vol. 51, no. 2, pp. 255–327, 2019, doi: 10.1007/s10462-017-9563-5.

[2] L. H. Son, N. Pritam, M. Khari, R. Kumar, P. T. M. Phuong, and P. H. Thong, "Empirical study of software defect prediction: A systematic mapping," *Symmetry (Basel)*, vol. 11, no. 2, 2019, doi: 10.3390/sym11020212.

[3] M. Fatih Adak, "Software defect detection by using data mining based fuzzy logic," *6th International Conference on Digital Information, Networking, and Wireless Communications, DINWC 2018*, pp. 65–69, 2018, doi: 10.1109/DINWC.2018.8356997.

[4] Y. Yang, J. Ai, and F. Wang, "Defect Prediction Based on the Characteristics of Multilayer Structure of Software Network," *Proceedings - 2018 IEEE 18th International Conference on Software Quality, Reliability, and Security Companion, QRS-C 2018*, pp. 27–34, 2018, doi: 10.1109/QRS-C.2018.00019.

[5] E. A. Felix and S. P. Lee, "Integrated Approach to Software Defect Prediction," *IEEE Access*, vol. 5, pp. 21524–21547, 2017, doi: 10.1109/ACCESS.2017.2759180.

[6] J. Ge, J. Liu, and W. Liu, "Comparative study on defect prediction algorithms of supervised learning software based on imbalanced classification data sets," *Proceedings - 2018 IEEE/ACIS 19th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, SNPD 2018*, pp. 399–406, 2018, doi: 10.1109/SNPD.2018.8441143.

[7] G. Sai Sundara Krishnan, R. Anitha, R. S. Lekshmi, M. Senthil Kumar, A. Bonato, and M. Graña, "Computational intelligence, cyber security and computational models: Proceedings of ICC3, 2013," *Advances in Intelligent Systems and Computing*, vol. 246, pp. 107–118, 2014, doi: 10.1007/978-81-322-1680-3.

[8] E. A. Felix and S. P. Lee, "Integrated Approach to Software Defect Prediction," IEEE Access, vol. 5, no. c, pp. 21524–21547, 2017, doi: 10.1109/ACCESS.2017.2759180.

[9] ZAIM ZULKIFLY, KYAIRUL AZMI BAHARIN, CHIN KIM GAN, Improved machine learning model selection technique for solar energy forcastingapplications, International journal of renewable energy research vol 11, no1,2021,https://doi.org/10.20508/ijrer.v11i1.11772.g8135

[10] Ammar Almasri, Erbug Celebi, Rami S. Alkhawaldeh, "EMT: Ensemble Meta-Based Tree Model for Predicting Student Performance", Scientific Programming, vol. 2019, Article ID 3610248, 13 pages, 2019. https://doi.org/10.1155/2019/3610248.

[11] Alkhawaldeh, R.S., Alawida, M., Alshdaifat, N.F.F. et al. Ensemble deep transfer learning model for Arabic (Indian) handwritten digit recognition. Neural Comput & Applic 34, 705–719 (2022). https://doi.org/10.1007/s00521-021-06423-7.