IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-7, Issue-3, March 2019 E-ISSN: 2347-2693

Performance Evaluation of Classification Algorithms Using MCDM and Rank Correlation Method Applied on Software Defect Prediction Datasets

Ankit Mehta[1*], Sandeep Upadhyay[2]

[1,2]Department of Computer Science, Techno NJR College of Engineering, Udaipur, Rajasthan, India

[*]*Corresponding Author: rmehtaankit@gmail.com, Tel.: +91-9610665222*

DOI: https://doi.org/10.26438/ijcse/v7i3.615620 | Available online at: www.ijcseonline.org

Accepted: 13/Mar/2019, Published: 31/Mar/2019

***Abstract*—** Software Defect Prediction is one of the important research areas of the software engineering. When developing new software from the existing prototype a software defect handling is one the major factor. In order to improve the quality of the software various data mining techniques are being used and applied to obtain predictions regarding the failure of particular software component by using the past datasets or logs consisting of various software measures related to the software defects. The main objective of the research was to rank & identify the most appropriate data mining classifier algorithms from the fifteen selected algorithms such as Lazy-IBK, Lazy-K Star, Function-SMO, Function-Multilayer Perceptron,Rules-ZeroR,Rules-OneR,Rules-PART,Tree-REP,Tree-Decision stump, J48, Naïve Bayes, BayesNet, Meta- AdaBoostM1,Misc-HyperPipes & Misc-VFI. In this particular research study firstly, 15 classifiers were applied to four datasets and the classification results were measured using 12 performance measures. Second, five MCDM methods (i.e., TOPSIS, GRA, VIKOR, PROMETHEE II, and ELECTRE III) were used to rank the classification algorithms based on their performances. So finally it can be concluded that the TOPSIS & VIKOR shows strong negative correlation which depicts that there is association between the two sets and the results were found in accordance. The best algorithm for software defect prediction datasets was found to be Lazy-IBK with highest overall score of 0.8023.

***Keywords*—** J48, IBK, TOPSIS, VIKOR, GRA, PROMETHEE II and ELECTRE III

I. INTRODUCTION

Software Defect Prediction (SDP) is presently one of the most challenging and important task in software companies particular when it comes to software development and maintenance, quality & reuse processes of the system development life cycle, which reflects the overall successes factor of the software. This is because predicting the software faults in earlier phase improves the software quality, reliability, efficiency and reduces the software cost. However, developing robust bug prediction model is a challenging task and many techniques have been proposed in the literature. According previous researches various software bug prediction model have been proposed based on machine learning (ML) algorithms or data mining algorithms. In this research study for better results different datasets related to software defects prediction were being considered along with different classifier algorithms such as Naïve Bayes (NB), BayesNet, Decision Tree (DT), SMO, IBK, KStar, AdaBoostM1, One R, PART etc. The performance evaluation process reveals that data mining algorithms with MCDM methods can very effective with high accuracy, reliability & quality measures. The comparative study between various algorithms at different n-fold crosses validation testing configuration was being done in this particular research work.

II. RELATED WORK

According to Feidu A., Ermiyas B. & Bahir S, 2017 the software applications are utilized to help the specialty units of different enterprises including human services, training, fabricating, protection, banking, etc. The software quality is assumes an imperative job. Software quality can be characterized as parameter to assess the plan and its implementation. To gauge software quality different traits are there, for example, product quality, precision (rightness), scalability, mistake free and so on. In spite of the fact that the quality measure received by one association may contrast from the other association so programming measurements standard measures ought to be utilized for quality evaluation. The software defect indicator can utilize the software metrics parameters as contribution to get the software quality. The examination covers the utilization of AI calculations to foresee the software quality. [1]

Peng He et al, 2015 in his exact examination chipped away at software defect forecast utilizing the software metrics or measures for assessment .The investigation was basically led on 10 open activities (34 releases) of PROMISE archive. As per the examination the significant discoveries recommends that top-k measurements or measure gives better yield when contrasted with benchmark indicators. [2]

The study led by Amit Kumar Jakhar and Kumar Rajnish, 2018 depended on highlight choice model in which chose highlights were incorporated and test was connected on NASA datasets from PROMISE information storehouse. In the proposed model diverse order calculations were being utilized for defect prediction with two test gatherings. In the main test aggregate all highlights were incorporated and analyze was led while in other set just chosen highlights were incorporated, the choice was being founded on weight doled out by different choice techniques. In first set 23 highlights were incorporated while in second gathering just 4 highlights were being utilized. The different execution measure, for example, Accuracy, Recall, Precision, MAE, TP, FP rate, F-measure and so on were utilized for assessment and it was reasoned that second test assemble indicates better execution in blame forecast when contrasted with first gathering. [3]

The study by Balogun, Abdullateef and O Bajeh, Amos and An Orie, Victor and W Yusuf-Asaju, Ayisa, 2018 recommends that software defect prediction is essentially the procedure of defects prediction in software modules right off the bat discover the modules for defects by utilizing different testing arrangements. The most generally utilized instrument for software defect prediction is classifier algorithms which assume a vital job in software development process. The specialist centres around the utilization of groups classifiers when contrasted with single classifier. As per the creator the execution of single classifiers (Function SMO,MLP, KNN and Decision Tree) were being contrasted and groups like Bagging, Stacking, Voting and so forth. The MCDM method utilized was Analytic Network Process (ANP). The test was led on 11 software defect datasets with 11 execution measures or parameters. It was reasoned that troupe strategies were smarter to single classifier as the positions accomplished by group were having higher need with 0.0493 most elevated when contrasted with others. [4]

Hammouri A., Hammad M., Alnabhan M., Alsarayrah F., 2018 Software bug prediction is a strategy in which a prediction model is made so as to foresee the future programming flaws dependent on recorded information. Different methodologies have been proposed utilizing diverse datasets, distinctive measurements and diverse execution measures. The investigation assessed the utilizing of AI calculations in software bug prediction issue. Three AI methods have been utilized, which are NB, DT and ANNs. The assessment procedure is executed utilizing three genuine testing/debugging datasets. Test results are gathered dependent on exactness, accuracy, review, F-measure, and RMSE measures. Results uncover that the ML strategies are proficient ways to deal with foresee the future programming bugs. The examination results demonstrated that the DT classifier has the best outcomes over the others. In addition, exploratory outcomes demonstrated that utilizing ML approach gives a superior act to the expectation show than different methodologies, for example, linear AR and POWM display. [5]

As per Singh and Chug, 2017 research work five most prominent Machine Learning algorithms were being utilized for software defect prediction that were Artificial Neural Networks (ANNs), Decision Tree (DT), Particle Swarm Optimization (PSO),) Linear Classifiers (LC) and Naïve Bayes (NB. The investigation displayed imperative outcomes including that the ANN has least blunder rate pursued by DT, however the linear classifier is superior to different algorithms in term of defect prediction accuracy, the most well known strategies utilized in software defect prediction are: DT, BL, ANN, SVM, RBL and EA, and the regular measurements utilized in software defect prediction examines are: Line Of Code (LOC) measurements, object situated measurements, for example, union, coupling and inheritance, additionally different measurements called cross breed measurements which utilized both item arranged and procedural measurements, besides the outcomes demonstrated that most programming deformity expectation contemplated utilized NASA dataset and PROMISE dataset.[6] According to Parameswari A.,2015 Software defect expectation (SDP) assumes an essential job in lessening the expenses of software development and keeping up the high caliber of software systems. Data mining Techniques, for example, 1) Feature Extraction 2) Classification are utilized to discover the imperfections that are available in the product item amid testing of every stage. Distinctive measurable techniques or calculations are utilized in highlight extraction stage to improve the precision of the imperfection expectation. Consequently the examination lessens cost by discovering blunders in past procedure as opposed to after consummation. While improving the precision of imperfection expectation the examination can locate the most appropriate calculation by looking at changed data mining classification calculation. [7]

The study led by Saiqa Aleem et al., 2015 uncovers that when similar investigation of AI strategies was accomplished for publically accessible NASA MDP dataset utilizing software prediction model at 10-fold cross validation testing arrangement. As indicated by the investigation the accompanying grouping algorithms were being utilized, for example, MLP, SVM, Naïve Bayesian, AdaBoost, Bagging Decision Tree, Random forest and KNN. The mean precision for Software fault prediction model for the given 15 property

dataset following outcomes were acquired with qualities SVM (89.29 %), Bagging (89.38 %) and Random forest (89.08 %).The best algorithm was observed to be MLP with absolute execution score of 99.52 %. [8]

As indicated by Dwivedi V.K., Singh M.K., 2016 today the software managers centre more around quality control, time and cost for which different distinctive assets are being utilized like better master designers, effective apparatuses, and better improvement strategies and so on. For these issues to be tended to software error or fault or defect prediction models are being utilized. The authors utilized two models named as model 1 and model 2 the proposed model 1 comprise of classification algorithm NN (Neural Network) while display 2 comprise of stacking as preparing set. The test was being led on DATATRIEVETM dataset which contain 130 instances and 9 attributes (counting class attribute).The results infers that model 1 was having precision of about 91.54% when contrasted with model 2. [9]

As per Gang Kou, Yanqun Lu, Yi Peng and Yong Shi, 2012 the MCDM methods can be considered as suitable devices for choosing the best classification algorithms for various datasets as the issue which incorporates different execution measures as basis and can be spoken to in MCDM issues structure. There are diverse MCDM methods which are having distinctive angles based on which the classifiers create unique rankings. While clashing positions were recognized between the MCDM methods the Spearman's rank correlation coefficient method was being utilized for positioning the classification algorithm. The essential methodology in the examination was to discover the weight for each MCDM method and after that dependent on similitude the position for other MCDM methods were being created. The examination was directed in WEKA condition in which 17 classification algorithms were being utilized for investigation at 10-fold cross validation connected on 11 binary UCI characterization datasets, for example, magic gamma telescope data, adult data and so on. Initially 10 performance measures were utilized and besides 5 MCDM methods, for example, TOPSIS, GRA, VIKOR, PROMETHEE II and ELECTRE III were utilized for the investigation and assessment of classification algorithms. Thirdly the Spearman's rank correlation coefficient was determined for finding the loads and producing the auxiliary positioning. The best algorithm with most noteworthy score was observed to be Bayes net. [10]

III. METHODOLOGY

The proposed study consists of following steps:
Step 1: Identification & data collection related to software defect prediction datasets which consists of attributes Total_LOC, Blank_LOC, Comment_LOC, Unique_operands Code_and_comment_LOC, Unique_operators etc. related to software measures.

Step 2: The data pre-processing & transformation will be done which basically converts the various different formats of datasets into required ones as an input to the system and analyzing tool here we will use Weka tool
.
Step 3: Different classification algorithms would be executed on the different datasets at different testing configurations including 10-fold, 15-fold & 5 fold cross validation would be taken to have comparison. In the proposed work 15 classifier algorithms would be used on four different software defect prediction datasets.

Step 4: Different accuracy measures like percent correctly classified, TP rate, FP rate, precision, recall, F-measure etc. would be determined for further analysis. The results were determined at different testing configuration that is randomly sampled partitions (i.e., 10-fold, 15-fold & 5-fold- cross-validation) using WEKA 3.6.9.

Step 5: Evaluate classification algorithms using TOPSIS, GRA, VIKOR and ELECTRE III. MCDM methods are implemented using MATLAB 7.0.

Step 6: Generate the first ranking of classification algorithms provided by each MCDM method. If there are disagreements among MCDM methods, go to Step 7; otherwise, end the process.

Step 7: Apply the Spearmen's Rank Correlation method for finding the correlation between different MCDM methods.

Step 8: Finally the best algorithms would be selected based on MCDM method, Spearmen's Rank Correlation method and percentile method.

Step 9: The selected data mining classifier would be best and most suitable for the software defect prediction.

Multiple-criteria decision-making (MCDM) is a well ordered technique which joins together the execution of the different choice choices over repudiating, quantitative and additionally subjective criteria and whose assessment results in a trade off arrangement. There are numerous MCDM methods been created and used to rank different choices in various ways. The specific rankings of options as being given by MCDM procedures may result in understanding yet while there are specific circumstances where different MCDM methods may produce altogether different and negating rankings. For this the other factual strategies dependent on subjective angles incorporate the utilization of Spearman's rank correlation coefficient so as to locate the rank correlation coefficient and create the last outcomes with summed up positions which

may decreases the distinctions among different MCDM positioning techniques. The investigation especially examinations a few MCDM methods so as to rank classifiers calculations initially then further so as to sum up if the outcomes are not in understanding the Spearman's rank correlation coefficient strategy is utilized to discover the connection coefficient between the diverse MCDM methods positioning sets and furthermore whenever required decide loads for MCDM methods to get auxiliary rankings of classifiers. The four MCDM methods used in the study are:

1. TOPSIS
2. ELECTRE III
3. Grey relational analysis
4. VIKOR

In order to evaluate and study the performance of classification data mining algorithms four software defect prediction (AR1, JM1, CM1 & KC1) datasets were taken from PROMISE Software Engineering Repository [11,12,13,14] into consideration and for experimental analysis. The purpose and reason for the particular choice is also discussed. It also includes the number of instances being used with details regarding attributes.

IV. RESULTS AND DISCUSSION

As per the outcomes acquired and broke down especially at adjustment of the algorithms were being dissected taking in thought different parameters or estimating perspectives, for example, accuracy, recall, precision, true positive rate, false positive rate, mean absolute error, AUC, F-measure and so forth. The tool utilized for examination was WEKA 3.6.9 environment. The outcomes for the dataset AR1 uncover that at 10-fold cross validation testing setup the algorithm demonstrating best execution was Lazy-IBK. The MCDM method TOPSIS and GRA likewise legitimizes it with most noteworthy execution score of 0.9139 and 0.8793.

At the point when the dataset AR1 was in effect further test on the testing configuration 15-fold cross validation it was discovered that near to 10-fold cross validation the execution scores were better particularly if there should arise an occurrence of exactness or percent effectively grouped occasions. The best three algorithms that were being recognized were Lazy-IBK, Function-SMO and Rules-PART with scores 0.5130, 0.5000 and 0.4863 as indicated by TOPSIS multiple criteria decision making technique. Additionally the MCDM method GRA proposes that the algorithm Lazy-IBK, Misc-HyperPipes and BayesNet were the best performing algorithms. Further the after-effects of VIKOR uncovers that Misc-VFI, Misc-HyperPipes and Function-Multilayer Perceptron were the best performing algorithms with performance score 1, 0.8549 and 0.7464 respectively.

Now applying TOPSIS technique on JM1 dataset at 10-fold cross validation testing design it was discovered that the best execution were appeared by the classifiers Lazy-IBK, Lazy-K Star and Function-SMO with execution score 0.7913, 0.7887 and 0.5000. Essentially as per GRA the best three best algorithms were Misc-HyperPipes, Lazy-IBK and Tree-REP with score esteems 0.7816, 0.7060 and 0.6879 individually.

As indicated by TOPSIS MCDM technique applied on JM1 dataset at 15-fold cross validation it very well may be reasoned that the Lazy-K Star, Lazy-IBK and Function-SMO were the best performers with score esteems 0.7888, 0.7787 and 0.5000. Similarly dependent on VIKOR technique the best algorithms were Misc-VFI, Misc-HyperPipes and Rules-ZeroR. The GRA results propose that the algorithms Misc-HyperPipes, Rules-PART and Lazy-IBK were having the best scores 0.7767, 0.6967 and 0.6942 individually.

Now considering TOPSIS method applied on the dataset CM1 it very well may be inferred that at 10-fold cross validation the algorithms demonstrating best outcomes were Lazy-K Star, Lazy-IBK and values 0.7971, 0.6020 and 0.5000. Similarly the consequences of GRA recommend that Misc-HyperPipes, Lazy-IBK and Naïve Bayes were the best performing algorithms. At the point when VIKOR method was applied it was discovered that Misc-VFI, Misc-HyperPipes and BayesNet were having the most elevated score esteems 1, 0.89 and 0.79 separately. The table above uncovers that dependent on the general normal score it can inferred that as indicated by TOPSIS the three best algorithms were observed to be Lazy-IBK, Lazy-K Star and Function-SMO with complete execution positioning score of 0.7686, 0.7101 and 0.5000.

Table 1: Performance of the algorithms with respect to the MCDM score based on measures applied on different datasets with the use of overall average of all testing configurations

Classifier	TOPSIS		GRA		VIKOR	
	Value	Rank	Value	Rank	Value	Rank
Lazy-IBK	0.7686	1	0.7641	2	0.1259	15
Lazy-K Star	0.7101	2	0.6328	9	0.1912	13
SMO	0.5000	3	0.6217	11	0.5978	5
Multilayer Perceptron	0.4133	9	0.6460	6	0.4572	9
Rules-ZeroR	0.3865	12	0.6070	13	0.6745	4
Rules-OneR	0.4351	5	0.6063	14	0.4529	10
Rules-PART	0.4279	6	0.6552	5	0.2973	12
Tree-REP	0.4153	8	0.6392	7	0.4738	8
Decision Stump	0.3975	11	0.6200	12	0.5859	6
J48	0.4260	7	0.6279	10	0.2973	11
Naïve Bayes	0.4415	4	0.6552	4	0.1331	14
BayesNet	0.3863	13	0.6640	3	0.7182	3
AdaBoostM1	0.4025	10	0.6343	8	0.5802	7
HyperPipes	0.2278	15	0.8056	1	0.8625	2
Misc-VFI	0.2473	14	0.5986	15	1.0000	1

So also dependent on MCDM method GRA it tends to be presumed that Misc-HyperPipes, Lazy-IBK and BayesNet were the best three performers first, second and third individually with all out execution positioning score of 0.8056, 0.7641 and 0.6640 separately with all out normal execution score of 1, 0.8625 and 0.7182. Further based on VIKOR MCDM method it was discovered that Misc-VFI, Misc-HyperPipes and BayesNet were the best three algorithms separately with absolute normal execution score of 1, 0.8625 and 0.7182. The VIKOR method was having the score esteems shifting to a great extent when contrasted with others.

Spearman's rank correlation coefficient estimates the similitude between two arrangements of rankings. The fundamental thought of the proposed methodology is to appoint a weight to each MCDM method as indicated by the likenesses between the positioning it created and the rankings delivered by other MCDM methods. An expansive estimation of Spearman's rank correlation coefficient shows a decent understanding between a MCDM method and other MCDM methods. The proposed methodology is intended to deal with clashing MCDM rankings through a determination of MCDM methods is connected to rank classification algorithms. On the off chance that there are solid differences among MCDM methods, the distinctive positioning scores created by MCDM methods are utilized as contributions for the second step. In the second step the Spearman's rank correlation coefficient is being utilized to discover the connection between two arrangements of rankings. A positive correlation coefficient shows a positive connection between the two factors (as estimations of one variable expands, estimations of the other variable likewise increments) while a negative correlation coefficient communicates a negative relationship (as estimations of one variable builds, estimations of the other variable decreases). A correlation coefficient of zero demonstrates that no relationship exists between the factors.

Table 2: Over all Best Ranked algorithms after MCDM & Spearman's Rank Correlation Coefficient

Classifier Algorithms (Best Three)	**Based on MCDM & Spearman's Rank Correlation Coefficient**	
	Average Performance Score	Rank Obtained
Lazy-IBK	0.8023	1
Lazy-K Star	0.7172	2
Naïve Bayes	0.6545	3

The correlation coefficient (r_{TG}) value between TOPSIS rank set & GRA rank set it was found to be 0.075 which shows weak positive correlation between the sets. Similarly the correlation coefficient ((r_{TV})) value between TOPSIS rank set and VIKOR rank set it was observed as -0.8464 which shows strong negative correlation between TOPSIS rank set and VIKOR rank set. Further taking in consideration the correlation coefficient (r_{GV}) value between GRA rank set and VIKOR rank set it was found to be -0.2714 which shows weak negative correlation between the two sets. So finally it can be concluded that the TOPSIS & VIKOR shows strong negative correlation which depicts that there is association between the two sets and the results were found in accordance. The best algorithm for software defect prediction datasets was found to be Lazy-IBK with highest overall score of 0.8023.

V. CONCLUSION AND FUTURE SCOPE

In the particular research study firstly, 15 classifiers were applied to four datasets and the classification results were measured using 12 performance measures. Second, five MCDM methods (i.e., TOPSIS, GRA, VIKOR, PROMETHEE II, and ELECTRE III) were used to rank the classification algorithms based on their performances.

It can be concluded that the TOPSIS & VIKOR shows strong negative correlation which depicts that there is association between the two sets and the results were found in accordance. The best algorithm for software defect prediction datasets was found to be Lazy-IBK with highest overall score of 0.8023.

.

REFERENCES

[1] Akmel, Feidu & Birihanu, Ermiyas & Siraj, Bahir. (2018). *"A Literature Review Study of Software Defect Prediction using Machine Learning Techniques"*. International Journal of Emerging Research in Management and Technology. 6. 300. 10.23956/ijermt.v6i6.286.

[2] He, Peng, et al. *"An empirical study on software defect prediction with a simplified metric set."* Information and Software Technology 59 (2015): 170-190.

[3] Amit Kumar Jakhar, K. R. (2018). *"Software Fault Prediction with Data Mining Techniques by Using"*. International Journal on Electrical Engineering and Informatics - Volume 10, Number 3 , 447-465.

[4] Balogun, Abdullateef & O Bajeh, Amos & A Orie, Victor & W Yusuf-Asaju, Ayisat. (2018). *"Software Defect Prediction Using Ensemble Learning: An ANP Based Evaluation Method"*.

[5] Hammouri, Awni & Hammad, Mustafa & Alnabhan, Mohammad & Alsarayrah, Fatima. (2018). *"Software Bug Prediction using Machine Learning Approach"*. International Journal of Advanced Computer Science and Applications Vol. 9, No. 2,78-83.

[6] Deep Singh, Praman & Chug, Anuradha.(2017). *"Software defect prediction analysis using machine learning algorithms"*. 775-781. 10.1109/CONFLUENCE.2017.7943255.

[7] A.Parameswari. (2015). *"Comparing Data Mining Techniques For Software Defect Prediction".* International Journal of Science and Engineering Research (IJ0SER),Vol 3 Issue 5 , 3221 5687, (P) 3221 568X.

[8] Saiqa A, Luiz F, and F. A, *"Benchmarking Machine Learning Techniques for Software Defect Detection".* International Journal of Software Engineering & Applications, vol. 6, pp. 11-23, May 2015.

[9] Dwivedi, V.K., & Singh, M.K. (2016). *"Software Defect Prediction Using Data Mining Classification Approach".*

[10] Gang Kou, Y. L. (2012). *"Evaluation of Classification Algorithms Using MCDM And Rank Correlation".* International Journal of Information Technology & Decision Making Vol. 11, No. 1 , 197-225.

[11] Ayse Tosun (2009) . AR1/Software defect prediction.The PROMISE Repository of Software Engineering Databases. School of Information Technology and Engineering, University of Ottawa, Canada . Creator : Software Research Laboratory (Softlab), Bogazici University, Istanbul, Turkey .Available: http://promise.site.uottawa.ca/SERepository.

[12] Tim Menzies (2004). CM1/Software defect prediction.The PROMISE Repository of Software Engineering Databases. School of Information Technology and Engineering, University of Ottawa, Canada . Creator : NASA. Available: http://promise.site.uottawa.ca/SERepository.

[13] Tim Menzies (2004). JM1/Software defect prediction. The PROMISE Repository of Software Engineering Databases. School of Information Technology and Engineering, University of Ottawa, Canada . Creator : NASA.Available: http://promise.site.uottawa.ca/SERepository.

[14] Tim Menzies (2004). KC1/Software defect prediction. The PROMISE Repository of Software Engineering Databases. School of Information Technology and Engineering, University of Ottawa, Canada .Creator : NASA.Available: http://promise.site.uottawa.ca/SERepository.

Authors Profile

Mr. Ankit Mehta pursed Bachelor of Technology from Visvesvaraya Technological University of Belgaum, Karnataka, India and presently pursuing Master of Technology from Techno NJR college of Engineering, Udaipur, Rajasthan, India.

Mr. Sandeep Upadhyay is being Assistant Professor in Department of Computer Science & Engineering from Techno NJR college of Engineering, Udaipur, Rajasthan, India. He is having total experience of more than 10 years in the field of Business Analytics.

