ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

Software Defect Estimation Using Machine Learning Algorithms

K. Venkataswara Rao[1], P. Dharahasini[2] , D.Hema[3] , N. Krishna Murthi [4] , Abdul Saleem[5] ,
[1]Ass.Professor, [2,3,4,5]Students

Dept of Computer Science And Engineering

Sri Vasavi Institute Of Engineering And Technology, Pedana, A.P, India

ABSTRACT:

Software defect prediction analysis is an important problem in the software engineering community. Software defect prediction can directly affect the quality and has achieved significant popularity in the last few years. This software prediction analysis helps in delivering the best development and makes the maintenance of software more reliable. This is because predicting the software faults in the earlier phase improves the software quality, efficiency, reliability and the overall cost in SDLC. Developing and improving the software defect prediction model is a challenging task and many techniques are introducing for better performance. Supervised ML algorithm have been used to predict future software faults based on historical data [1]. These classifiers are Naïve Bayes (NB), Support Vector Machine (SVM) and Artificial neural network (ANN). The evaluation process showed that ML algorithms can be used effectively with a high accuracy rate. The comparison is made with other machine learning algorithms to finds the algorithms which gives more accuracy. And the results show that machine learning algorithms gives the best performance. The existence of software defects affects dramatically on software reliability, quality, and maintenance cost. Achieving reliable software also is hard work, even the software applied carefully because most time there is hidden errors. In addition, developing a software defect prediction model which could predict the faulty modules in the early phase is a real challenge in software engineering. Software defect prediction analysis is an essential activity in software development. This is because predicting the bugs prior to software deployment achieve users satisfaction, and helps in increasing the overall performance of the software. Moreover, predicting software defects early improves software adaptation to different environments and increases resource utilization.

INTRODUCTION:

Software Defect Prediction is an important issue in software development and maintenance processes, which concerns with the overall of software success. Predicting and finding the bugs in the earlier phase in SDLC makes the software more reliable, efficient and better quality when compared with finding bugs in the later stages. However, developing a software defect prediction

Copyright @ 2022 Author
DOI : 10.36893.JK.2022.V12I01N01.994-1000

ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

model is not an easy task and many new tools and methods are introducing in the machine learning for better performance. These classifiers are Naïve Bayes(NB) and Support Vector Machine(SVM) and Artificial Neural Networks(ANN). The development procedure demonstrated that ML calculations can be utilized adequately with a high precision rate. A programming deformity is a blunder, bug, imperfection, issue, breakdown or errors in programming that makes it make a mistaken or unpredicted result. Issues are basic properties of a framework. They show up from structure or assembling, or outside condition. Programming blemishes are customizing mistakes which cause distinctive execution contrasted and expectation. The dominant parts of the flaws are from source code or plan, some of them are from the mistaken code producing from compilers. For programming designers and customers, programming deficiencies are a perilous issue. Programming abandons not only diminish programming quality, increment cost yet in addition postpone the advancement plan. Programming deficiency anticipating is proposed to settle this kind of trouble[4].

LITERATURE SURVEY:

Software Engineering is a comprehensive domain since it requires a tight communication between system stake holders and delivering the system to be developed within a determinate time and a limited budget. Delivering the customer requirements include procuring high performance by minimizing the system. Thanks to effective prediction of system defects on the front line of the project life cycle, the project's resources and the effort or the software developers can be allocated more efficiently for system development and quality assurance activities. The main aim of this is to evaluate the capability of machine learning algorithms in software defect prediction and find the best category while comparing seven machine learning algorithms within the context of NASA datasets obtained from public repository.

Software Quality is the most important aspect of a software. Software Defect Prediction can directly affect quality and has achieved significant popularity in last few years. Defective software modules have a massive impact over software's quality leading to cost overruns, delayed timelines and much higher maintenance costs. In this paper we have analysed the most popular and widely used Machine Learning algorithms — ANN (Artificial Neural Network), PSO(P article Swarm Optimization), DT (Decision Trees), NB(Naive Bayes) and LC (Linear classifier). The five algorithms were analysed using KEEL tool and validated using k-fold cross validation technique. Datasets used in this research were obtained from open source NASA Promise dataset repository. Seven datasets were selected for defect prediction analysis. Classification was performed on these 7 datasets and validated using 10

Copyright @ 2022 Author
DOI : 10.36893.JK.2022.V12I01N01.994-1000

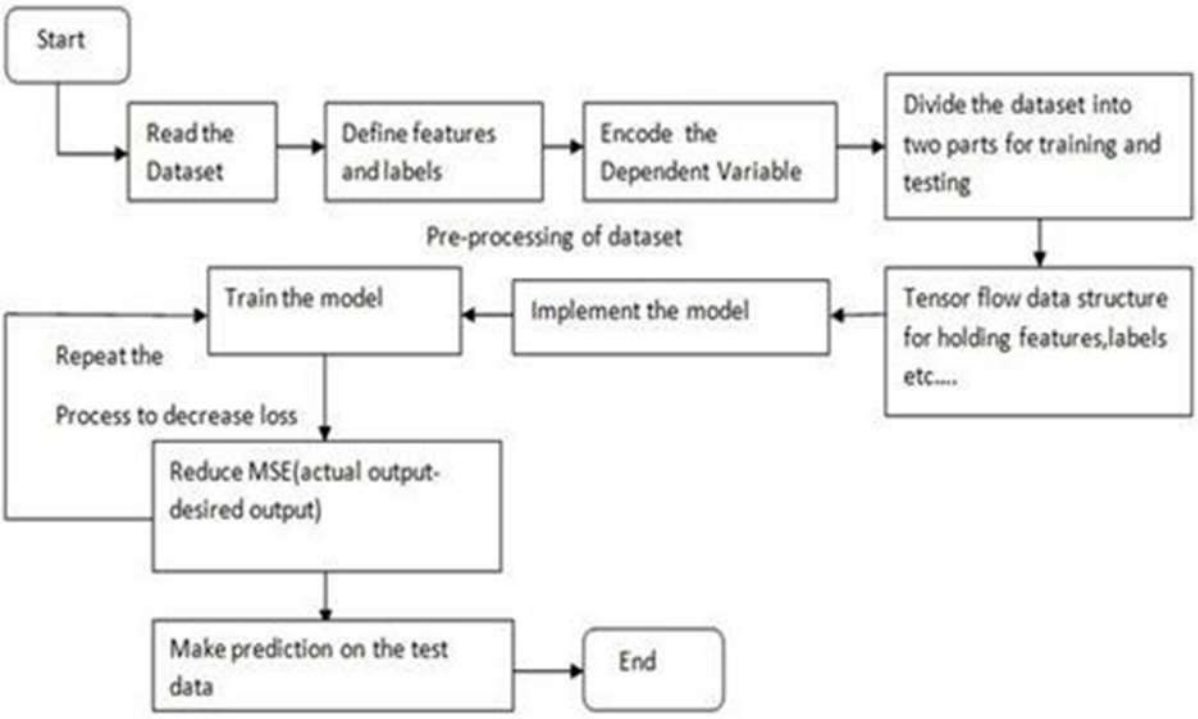
ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

fold cross validation. The results demonstrated the dominance of Linear Classifier over other algorithms in terms of defect prediction accuracy.

PROPOSED METHOD:

In this paper author is evaluating performance of various machine learning algorithms such as SVM, Bagging, Naïve Bayes, Multinomial Naïve Bayes, RBF, Random Forest and Multilayer Perceptron Algorithms to detect bugs or defects from Software Components. Defects will occur in software components due to poor coding which may increase software development and maintenance cost and this problem leads to dis-satisfaction from customers. To detect defects from software components various techniques were developed but right now machine learning algorithms are gaining lots of popularity due to its better performance. So in this paper also author is using machine learning algorithms to detect defects from software modules. In this paper author is using dataset from NASA Software components and the name of those datasets are CM1 and KC1. I am also using same datasets to evaluate performance of above mention algorithms.

ARCHITECTURE (Methodology):



Copyright @ 2022 Author
DOI : 10.36893.JK.2022.V12I01N01.994-1000

ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

DATASET:

Model	Accuracy	Precision	F-measure	Sensitivity	Specificity
#1	89.16%	0.9423	0.8981	0.8964	0.7674
#2	88.96%	0.9345	0.9048	0.9177	0.7650
#3	90.36%	0.9117	0.9072	**0.9256**	0.8311
#4	87.95%	0.9342	0.9003	0.9183	0.8154
#5	87.35%	0.9011	0.9187	0.9383	0.8276
#6	82.72%	0.9366	0.8624	0.8758	0.7963
#7	80.33%	0.8643	0.8802	0.8779	0.7841
#8	79.15%	0.9094	0.8771	0.8437	0.6340
#9	83.14%	0.8832	0.9015	0.9147	**0.8407**
#10	**90.45%**	**0.9507**	**0.9376**	0.9210	0.7761
#11	70.65%	0.9211	0.7831	0.6847	0.7063
#12	60.18%	0.8974	0.7786	0.7902	0.5474
#13	60.37%	0.9101	0.8771	0.8479	0.5341
#14	58.37%	0.9093	0.7354	0.6451	0.4228
#15	58.63%	0.8931	0.7527	0.6841	0.4276
#16	58.42%	0.8887	0.7372	0.6289	0.5494
#17	65.85%	0.8867	0.7563	0.6371	0.7702
#18	73.37%	0.9282	0.7694	0.6944	0.7582
#19	82.37%	0.8981	0.8993	0.9002	0.7353
#20	81.53%	0.8634	0.8759	0.9261	0.7254

IMPLEMENTATION:

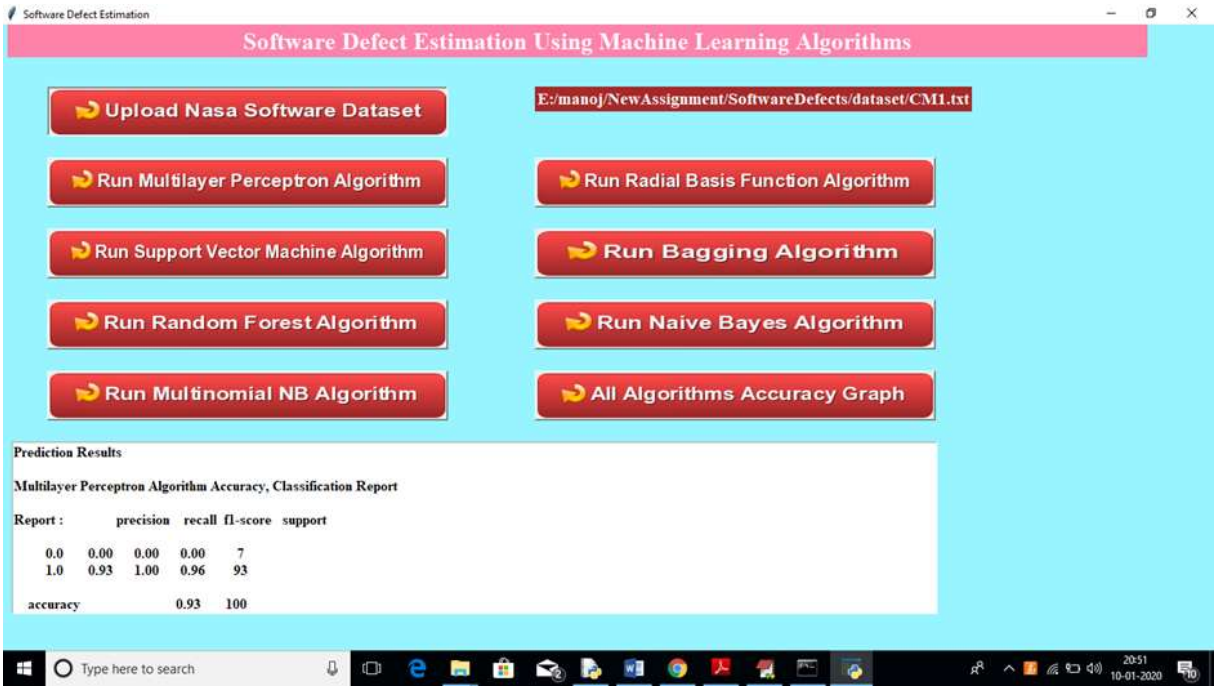
Steps:

- Gathering the dataset
- Pre-processing the dataset and analysis dataset
- For prediction apply the Linear Regression, Decision Tree, Random Forest, logistic regression, decision tree. models on the dataset by splitting the datasets in to 70 to 80 % of training with these models and 30 t0 20 % of testing for predicting
- Obtain the accuracy

Copyright @ 2022 Author
DOI : 10.36893.JK.2022.V12I01N01.994-1000

ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

IMPLEMENTATION RESULTS:

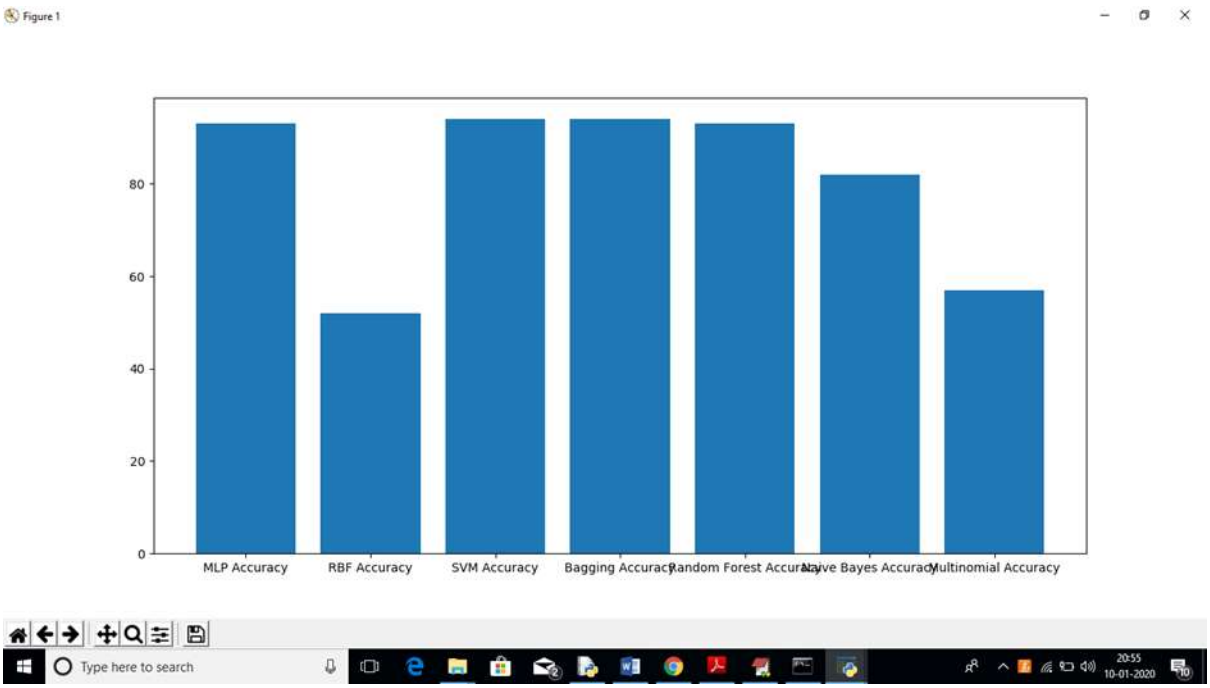


In above screen we can see multilayer perceptron fmeasure, recall and accuracy values and scroll down in text area to see all details.

In above screen we can see multilayer perceptron accuracy is 93%. Similarly you click on all other algorithms button to see their accuracies and then click on 'All Algorithms Accuracy Graph' button to see all algorithms accuracy in graph to understand which algorithm is giving high accuracy.

Copyright @ 2022 Author
DOI : 10.36893.JK.2022.V12I01N01.994-1000

ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022



In above graph x-axis represents algorithm name and y-axis represents accuracy of those algorithms. In all algorithms we can see MLP, Bagging is giving better accuracy.

CONCLUSION:

In this experimental study, seven machine learning algorithms are used to predict defectiveness of software systems before they are released to the real environment and/or delivered to the customers and the best category which has the most capability to predict the software defects are tried to find while comparing them based on software quality metrics which are accuracy, precision, recall and F-measure. We carry out this experimental study with four NASA datasets which are PC1, CM1, KC1 and KC2. These datasets are obtained from public PROMISE repository. The results of this experimental study indicate that tree-structured classifiers in other words ensemble learners which are Random Forests and Bagging have better defect prediction performance compared to its counterparts. Especially, the capability of Bagging in predicting software defectiveness is better. When applied to all datasets, the overall accuracy, precision, recall and FMeasure of Bagging is within 83,7-94,1%, 81,3-93,1%, 83,7- 94,1% and 82,4-92,8% respectively. For PC1 dataset, Bagging outperforms all other machine learning techniques in all quality metric. However, Naive Bayes outperforms Bagging in precision and F-Measure while Bagging outperforms it in accuracy and recall for CM1 dataset. Random Forests outperforms all machine learning techniques in all quality metrics for KC1 dataset. Finally, for KC2 dataset, MLP outperforms all machine learning techniques in all quality metrics for KC2 dataset.

Copyright @ 2022 Author
DOI : 10.36893.JK.2022.V12I01N01.994-1000

ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

REFERENCES:

[1] Victor R Basili, Lionel C. Briand, and Walcelio L Melo. ´ A validation of object-oriented design metrics as quality indicators. IEEE Transactions on software engineering, 22(10):751–761, 1996.

[2] Evren Ceylan, F Onur Kutlubay, and Ayse B Bener.Software defect identification using machine learning techniques.In 32nd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO'06), pages 240–247.IEEE, 2006.

[3] Karim O Elish and Mahmoud O Elish. Predicting defect-prone software modules using support vector machines. Journal of Systems and Software, 81(5):649– 660, 2008.

[4] Norman Fenton, Paul Krause, and Martin Neil. Software measurement: Uncertainty and causal modeling. IEEE software, 19(4):116–122, 2002.

[5] Lan Guo, Yan Ma, Bojan Cukic, and Harshinder Singh. Robust prediction of fault-proneness by random forests. In 15th International Symposium on Software Reliability Engineering, pages 417–428. IEEE, 2004.

[6] Taghi M Khoshgoftaar, Edward B Allen, and Jianyu Deng. Using regression trees to classify fault-prone software modules. IEEE Transactions on reliability, 51(4):455–462, 2002.

[7] Taghi M Khoshgoftaar, Edward B Allen, John P Hudepohl, and Stephen J Aud. Application of neural networks to software quality modeling of a very large telecommunications system. IEEE Transactions on Neural Networks, 8(4):902–909, 1997.

[8] Sunghun Kim, Hongyu Zhang, Rongxin Wu, and Liang Gong. Dealing with noise in defect prediction. In 2011 33rd International Conference on Software Engineering (ICSE), pages 481–490. IEEE, 2011.

[9] Yan Ma, Lan Guo, and Bojan Cukic. A statistical framework for the prediction of fault-proneness. In Advances in Machine Learning Applications in Software Engineering, pages 237–263. IGI Global, 2007.

[10] Ruchika Malhotra. A systematic review of machine learning techniques for software fault prediction. Applied Soft Computing, 27:504–518, 2015.

Copyright @ 2022 Author
DOI : 10.36893.JK.2022.V12I01N01.994-1000