Attribute selection for Software Defect Prediction

ASHA LATHA NAMA, HARSHINI BADAM

Abstract

Software defect prediction is used to help developers identify probable flaws and decide where to focus their testing efforts to increase the reliability and quality of software. Software quality have a large impact on the software which leads to higher maintenance costs. In the software development life cycle, early detection of defects results in quick corrections and providing a maintainable software for the client. As the demand for the software systems is increasing day by day, it is necessary to control and reduce the defects. Most of the previous studies focused on finding the accuracy to predict defect or not by applying different machine learning algorithms. This paper identifies the features or attributes that are contributing for detecting whether there is a defect in the software. These attributes were analyzed using a classification algorithm and validated using k-fold cross validation. A dataset with 22 general attributes of a source code metrics were considered for the classification. The results demonstrated that, there are some specific attributes that are contributing more to predicting a defective code.

1.Introduction

Software is becoming more and more important in modern life, knowingly or unknowingly people are using software in every task they perform in their day-to-day activities. Software helps users to achieve greater results like accomplishing tasks faster, easier. But there are many risks in designing a software, one of them is defects.

A software defect is an error or fault in the design, implementation and operation of software system which may impact functionality, performance, undesired results or behaving in an unexpected way. Defects in the software are due to human errors, software errors or unclear specifications of the client which may affect the quality of the software. Predicting the defect is an important process in the software designing lifecycle. Mostly used technique to predict software defects is by building prediction models which measures the quality of the code as defective or non-defective. This approach it is not possible because of its cost within the available resources.

Also, with the help of software attributes quality of the product in the software development can be predicted. We concentrate more on code level attributes which has more impact in designing the defect prediction models. Our way of approach is search based i.e., by exploring the attributes available in the dataset and presenting which attributes contribute more in predicting the defect with the help of feature selection algorithms.

2.Limitations

In this paper, we are analyzing the attributes which contributes more to predicting the software defects by analyzing the data and with the help of feature extraction model. For example, we have used two methodologies in analyzing the attributes that contributed more to predicting the software defect and got almost same results. Due to less attributes in the dataset, it is difficult for finding patterns. The approach followed in our model is reactive in nature, in the future approaches should be proactive in nature, i.e., performing predictions, during the software development process instead of waiting for it to be completed.

Detecting a defect using the new market, many previous approaches and the approach followed in this paper are detecting the software using the source code metrics or process metrics. These are the old approaches.

This approach cannot be followed for the mobile applications. Some of the defects in these mobile applications can be predicted from the review of users i.e., the performance defects from the client side. This technique only tells where the defect is, but it won’t give any information about how to resolve the defect. For the future software defect prediction models, we should not only defects, we also need to provide the possible ways to resolve the defect.

In this paper, the attribute effort is not contributing more for the defect prediction, in the further analysis effort should be considered as the main attribute because the main question is for the code review, how much effort are these prediction models applying. In the future this has a lot of impact and many previous models have failed to explain this.

3.Methodology

Many defects that occur in a software are generally due to some metrics that are length of the program, complexities of the program. These have an impact on the software quality which in turn effects the client. In this paper we focus on whether these metrics are really contributing for a defect considering the dependent variable as defect and Independent Variables as line code of count, complexities, program length, difficulty (predicted using correlation matrix), with the use of logistic regression a classification algorithm we are predicting the accuracy and finding how much percentage they contribute for a predicting a software is defect or not. We are also using the k-fold cross validation to validate our results with the accuracy we found with the logistic regression.

This paper also finds what other attributes including with the general metrics are going to help for predicting a defect. Data was collected from a Kaggle website, the size of the data is 2109 and consists of 22 attributes. Extra Tree classifier is used for feature extraction and ranking the features, which gives the top features that have high rank and the accuracy for the three top five, four and three features were calculated, based on the accuracies we are going to find, which are the attributes that contribute for predicting a defect. The libraries we made use of in this paper are NumPy, scikit learn and matplotlib.

The dataset contains 3 McCabe metrics, 12 Halstead measures, 5 different lines of code measure, a branch count and 1 target i.e., defects. McCabe software metrics are related to the complexities i.e., essential complexity(ev(g)),
cyclomatic complexity(v(g)),
design complexity(iv(g))
lines of code,

where he argues that these metrics will give a complicated paths that might produce the defect. Halstead metrics are
total operators and operands(n),
volume(v),
programlength(l),
difficulty(d),
intelligence(i),
effort(e),
time estimator(t),
line count(locode),
count of lines of comment(locomment), count of blank lines(loblank)
argues that when the code is
hard to read it is lkely to be fault prone.

The other attributes are:

locodeAndComment,
uniqueoperators(unit_Op)
uniqueoperators(uniq_Op),
uniqueoperands(uniq_Opnd),
totaloperators(total_Op),
totaloperands(total_Opnd),
branch count,
defects (False/True).

- Cyclomatic Complexity(v(g)): It determines the level of confidence in a program and the number of linearly independent paths through the program.
- Essential Complexity(ev(g)): It measures the structure of a program i.e., the number of entry points, termination points.
- Design Complexity(iv(g)): Measure of achieving the required specifications i.e., functional requirements and constraints which is a time independent complexity.
- n: Total number of operands and operators
- v: It determines the size of implementation of an algorithm
- l: length of program
- d: effort required to maintain a program
- Defect: Categorical Variable (0/1)

3.1 Logistic Regression

It is a supervised machine learning algorithm which is used to predict the probability of target variable, we use this algorithm for the categorical variables. It is used for various classification such as spam detection, fake news detection, Cancer detection etc. It can easily determine the effective variables for classification by observing different types of data. For the algorithm the dependent variable must be categorical value. In our dataset the categorical variable is Defects which is considered as dependent variable.

3.2 Extra Tree Classifier

It is machine learning algorithm, combines the predictions from many decision trees. It splits each node based on the Gini index, selects the split value in a random way. It is based on a top down model approach. It uses all the records of the sample. It is faster than random forest.

For performing the feature selection, we compute a Gini index value for each feature. This value is called as Gini importance of feature. After these all the features are ordered in descending order based on the Gini index value and then the user can select the top features that is required. It minimizes over learning from the data and over fitting.

The three advantages of feature selection is reducing training time, improving accuracy and decreases over fitting. It also increases the power of machine learning and gives good results.

3.3 K-fold cross validation

It is an evaluation method in machine learning, it finds how well the model performs on the unseen data. The data will be splitted into k samples. K means number of folds, we use this to build a generalized model. Here we split the data into training, testing and validation sets where the amount of data varies, and accuracy will be found in each fold. In each fold, the dataset will be splitted differently, which gives accurate model.

It helps to avoid overfitting, in each fold, one set will be given for the testing during the entire process. In k-fold validation, for the training purpose we give (k-1) folds for the training purpose. If the k value is 10, it indicates that the data was used of good size. The k value should be as small as possible, if the value of k is large, increases the running time and leads to low variance in the training set. From this we can say that the k fold number and the size of the data are indirectly proportional. If the size is too small, number of k folds increases.

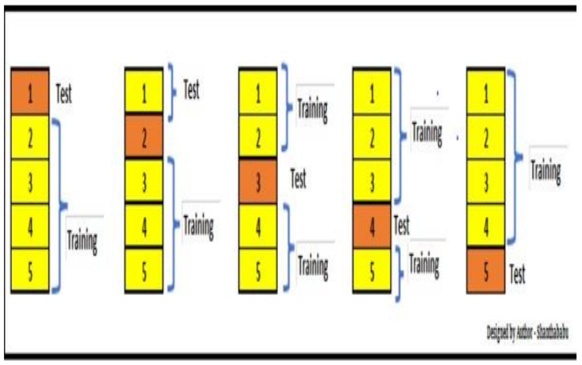

Fig 3.3.1 K-fold cross validation process

3.4 Correlation matrix

It is a matrix table which represents the correlation coefficient between different variables. The values in the matrix have all possible pair of values. For large datasets, this will be very useful and easy to visualize and summarize the patterns in the data. To visualize the correlation between the variables, we can use heatmap. For the regression techniques the correlation matrix is very useful. In the correlation matrix, if the value is 1, the features are strongly related, 0 means neutral and -1 is the relationship is not strong.

4. Literature Review

Praman Deep Singh, Anuradha Chag[1], analyzed different machine learning algorithms Artificial Neural Network, Decision Trees, Naïve Bayes and Linear classifier on the KEEL tool and cross validated using k-fold. The author used 7 datasets and classification was performed on all the 7 datasets. The results showed that, linear classifier has highest accuracy in all the datasets.

Anuradha Chug, Shafali Dhall[2], discussed prediction of defect using the data mining techniques. Here 3 data mining algorithms were used i.e., J48, Random Forest and Naïve Bayesian classifier. To evaluate this ROC, Precision, MAE, RAE etc were considered. They also used clustering techniques, and these were evaluated using some criteria like time taken, number of iterations. The results showed that, in the data mining techniques random forest produced higher accuracy and for the clustering techniques k-means developed more relevant information.

D. Rodr´ıguez1, R. Ruiz, J. Cuadrado-Gallego1, J. Aguilar-Ruiz [3], in this paper feature selection was applied on different software engineering databases and applied datamining classification techniques to detect the faulty modules. The results showed that, smaller datasets with lower attributes have better accuracy. Also, it uses a model called wrapper method, which is better than the filter model. Wrapper model, to evaluate a criteria uses a predetermined mining algorithm and use its performance, it searches features that are suitable for the mining algorithm. Filter model, it selects the features without any data evaluation.

Bavisi, Mehta and Lopes [4], they proposed that any single learning algorithm cannot perform better than different learning algorithm as both are being compared with the same performance measure and also, they stated that from domain-to-domain performance varies. So, they came to a conclusion that to find the accuracy of an algorithm there are other factors also this will depend on nature of the problem, dataset used in the experiment.

Arisholm, E., Briand, L. C., & Johannessen, E. B [5], this paper mainly focuses on three aspects on how to build and evaluate fault prediction models, the aspects are comparing data mining models, impact of using different sets of source metrics and comparing models with different performance metrics.

Ajeet Kumar Pandey & N. K. Goyal [6], for early fault prediction this model used fuzzy expert system to detect the number of faults before testing and using the software metrics these number of faults will be calculated at the end of software development phase. For predicting a defect in software, they also analyzed the impact of each individual phase of software development life cycle, from three different software projects metrics were considered and results were evaluated. This model gives an idea how metrics have impact on the defects during the development process.

Norman Fenton, Paul Krause and Martin Neil[7], For the prediction of software quality, many approaches have failed. In this paper, the author's goal is to achieve a single model of software development which is the combination of diverse forms. The graphical probability model which is also the Bayesian Belief Network is used for the proper representation of the evidence. By using the outputs of the experienced project managers for the construction of the probability model and using this model for the estimation of the software quality in the lifecycle development. This paper mainly works on the one type of model that is developed for the Philips Software Centre (PSC). This model is used to identify the defect rates at various testing and operational phases. For the software quality managers to use the model they developed the tool (AID).

For the development of software products many important decisions need to be made among them when to release the software product is the most important one. If any wrong decision is made this will affect the name of the product or the supplier. If we want to get good software quality development, we need to perform the following: They are:

Defect Prevention,

Defect Detection

Defect Correction.

The only disadvantage of the reliability model is the amount of data that is required to do the validation study. Besides the disadvantage the outcome of the study is positive.

Asst.Prof.Dr. Abdul Syukor Mohamad[8], the author is comparing the neural network based predictions to the fuzzy fundamental logics. Defects are found using various methods. In the context of neural networks, computer aided engineering is applied. To assess the quality of the result, gradient and regression analysis. The results showed that, ANN based strategy performed better than the fuzzy rules and the also concluded that, the prediction metrics are based on the fuzzy rules.

TAGHI M. KHOSHGOFTAAR, JOHN C. MUNSON[9], they are building models to provide a relationship between the errors and software complexity metrics. The common aspects of the program complexity are measured by the existing complexity metrics. Factor analysis is performed to describe the relationship between the variables. It is an extension of principal component analysis. This can be applied for detecting the defects.

The regression models that were performed on the raw complexity metrics gives a relationship between the lines of code and errors. From the second methodology it was showed that, there is a relationship between the program structure and volume. They also showed that software metrics are highly interrelated. The high correlation indicates we can develop predictive models. The R^2 statistic only increase if there is any new data and they contribute to the overall predictive quality.

C.Lakshmi Prabha, Dr.N.shivakumar[10], for the early identification of prediction proper classification is necessary. To reduce the enorumous dimension, hybrid approach has been used where this approach includes methods like PCA, random forest, naïve bayes and SVM. In this paper a research has been conducted on the performance metrics confusion, recall, precision, accuracy, these are measured and compared the metrics in this paper. The dataset used in this was collected from the Kaggle, this analysis has performed on 5 datasets namely KC1, PC3, MW1 AND CM1. For the PC3 dataset, the accuracy is 100% compared to others .

K. Saravana Kumar, R. B. Misra[11], discussed that, there are many previous models present to predict the reliability of a software system based on the metrics, due to fuzziness in the data models predicted early are not that accurate. They proposed a model which will help the software developers to idenfity which phase an action is required. They also explained how the software engineering metrics are effecting the defect in prediction in different phases of software development life cycle. The analysis performed at the requirement stage, Design stage and Coding Stage. This method provide an approach to detect the errors in the early phase of the cycle.

MuhammadAdnanKhan,NouhSabri Elmitwally,SagheerAbbas,ShabibAftab,MunirAhmad,MuhammadFayazand Faheem Khan[12], this paper discussed about the use of artificial neural networks for software defect prediction. A systematic research has been conducted on how current trends of software defect prediction are focusing on ANN. For the literature extraction, three online libraries were used they are IEEE, Elsevier and Springer. MATLAB has been used as a common tool. In this paper, the ensemble techniques and feature selection are investigated for the problem of detecting a defect in software defect prediction.

5.Main Body

Software defect prediction has been the main concern these days. To assure the quality faults need to be detected. In the software development life cycle, the defect can occur in any phase of the cycle like requirements, design phase or development phase. When defect occurred in the development phase it is difficult to find and also it is will be tough to resolve the defect as it takes a lot of time and this phase reaches to the clients. So, detecting a defect in the early is the best way defect prediction.

Many previous researchers were focused on the accuracy and on the machine learning models, but there are very few researchers who focused on the attributes which are involved occurring the default. When compared with the other software tools, like software testing and reviews, software defect prediction is cost-effective.

Software defects are might be due to the faults in the source code like the length of the program might be high, the complexities of the code is high, etc., the general attribute of the source code, will effect the software.

In this paper, the research question we are focusing is what attributes are contributing for predicting a defect in the software and also we are finding whether the attributes which we considered like the general attributes where every person thinks that these might contribute for finding a defect, accuracy was found for those attributes also. Firstly we started our modelling on the attributes which we considered.

We have used the dataset kc1 from the Kaggle website which is a combination of Halstead and McCabe attributes. After loading the dataset, we have done some preprocesssing steps like data cleaning. We have tried to find the duplicates and null values in our table, we found that there are no duplicates but has null values. These null rows are dropped and the dataset reduced to 1212 rows with 22 attributes.

Data has been normalized i.e. all the column values were changed to the same type. Visualized the defects column which contained categorical values(0 or 1), using histogram plot. It shows the frequency distributions and number of observations in each interval. The attributes we considered for our model are the loc[line count of code], v(g)[cyclomatic complexity], ev(g)[essential complexity], l[program length]. Considering these attributes are contributing more, we have trained our models on this attributes.

Data was splitted into training and testing sets 80 and 20% respectively, then logistic regression a supervised machine learning was applied on our dataset, we got an accuracy of 74%. To cross validate our results, we have used k-fold cross validation method, where we get an array of our accuracies, mean of the accuracies is considered. After the cross validation we found that the accuracy we have achieved for the attributes we considered is 74%.

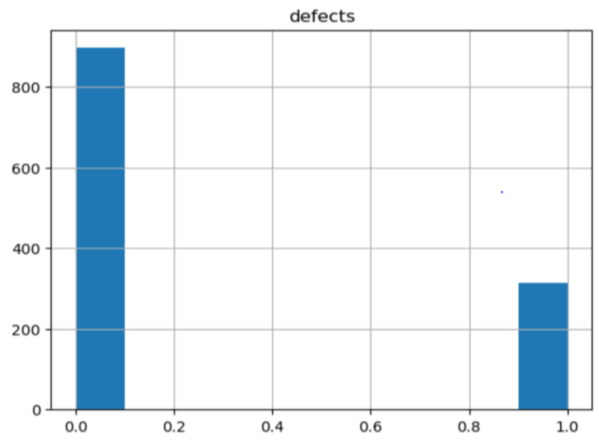

Fig 5.1 histogram for defects label

For the other research question, we have followed the below steps. All the 21 attributes are considered for predicting the accuracy in detecting a defect. We found that, for all the attributes we have achieved an accuracy of 75.3%. Considering all the attributes for model prediction is not a best way of training a model, so we tried to remove the attributes which are not important using the correlation matrix. We get a matrix, representing how many features are related to each other and how well they are related to each other. Each matrix value contains the correlation coefficient. In this matrix, we have an upper matrix and lower matrix, ignoring the diagonals. For our modelling, we have considered the upper matrix, ignoring the lower matrix values. After the correlation matrix results, dropped the features that are less related. By dropping the features, we have got 12 attributes. The below figure represents the correlation matrix.

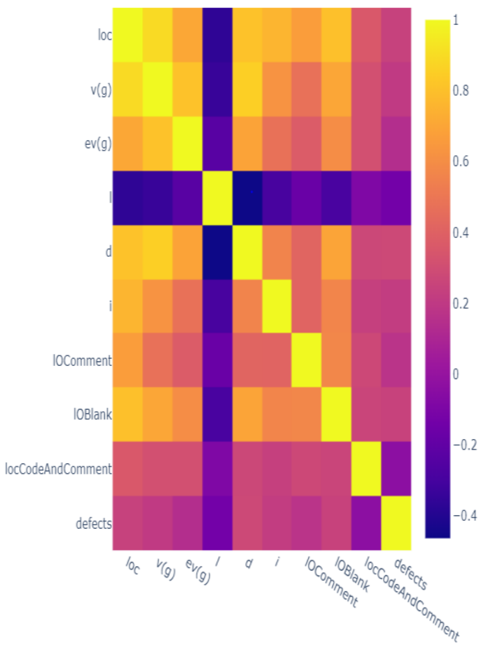

Fig 5.2 heat map for the attributes

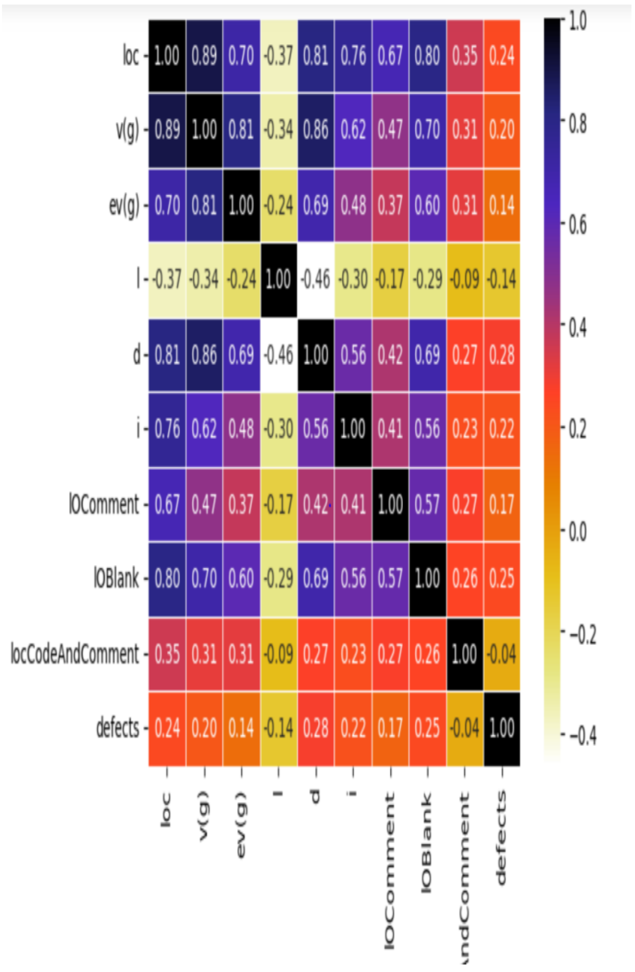

Fig 5.3 correlation matrix for attributes

Now by reducing the attributes to 12, we can train the model. By applying logistic regression and cross validating our model, we have got an accuracy of 74.8% for the 12 attributes. To increase the prediction power and accuracy of a model, we are applying feature selection on our attributes(12). The feature ranking helps in identifying the subset of input attributes that contributes for the target variable. Here our target variable is the Defect column, We need to find the subset of input attributes. Using the Extra tree classifier, we are finding the top inputs for predicting the defect. Feature ranking helps in giving importance for the features based on the classifier where it uses Gini index value and gives ranking for the attributes.

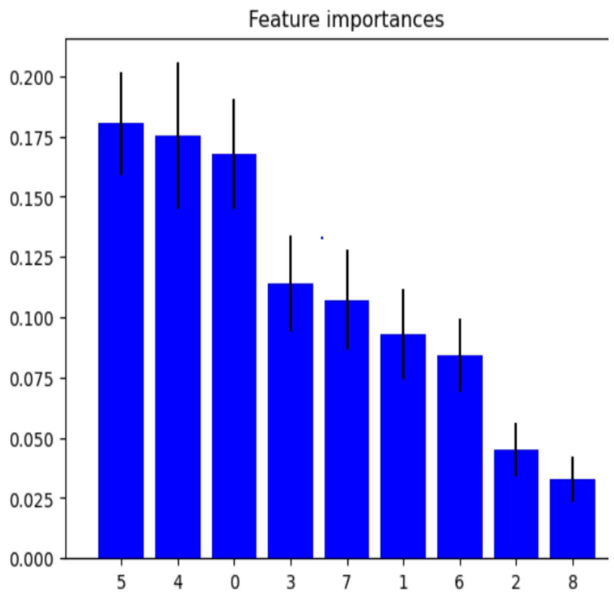

Fig 5.4 Feature Importance ranking

The above figure shows a graphical representation of the feature and their ranking respectively. From the figure we can say that, feature 5 has more importance and the top 5 features are feature 5, feature 4, feature 0, feature 3, feature 7. We have calculated accuracy for the top 5, top 4 and top 3 features using logistic regression and the accuracy for the 3 subset of features are 75.7%, 74.8% and 74.8%. Top 5 features have more frequency when compared with the other subsetof features.

6.Conclusion:

Qualitative software is one of the main concern for today's software developers. Due to the increase in complexities of software and it is difficult to deliver a model with high quality and also the costs increases. These difficulties are due to the defects in the software, to deliver a high quality software these defects should be identified and removed. The attributes like complexities, length of program contribute for detecting a defect in a software.

Both the research questions we considered for analysis were answered. For the first research question, we have achieved an accuracy of 74.3% trained with logistic regression for 5 attributes that were considered manually. For the second research question, after the correlation, the accuracy was found to be 74.8% accuracy. After the feature ranking using the extra tree classifier, considering the top 5 features we have achieved an accuracy of 75.7%, for the top 4 features the accuracy is 74.8% and for the top 3 features also it has the same accuracy of 74.8%.

From this we can conclude that, top 5 features are contributing more for predicting the defect in software. Also, we can say that, attributes which are considered initially i.e. the loc[line count of code], v(g)[cyclomatic complexity], ev(g)[essential complexity], l[program length] the accuracy. Hence, we can say that the features which are contributing more in the software defect prediction are v[Halstead volume], n[Halsteadtotaloperators+operands, loc[McCabe'slinecountofcode], iv(g)[McCabe'sdesigncomplexity], d[Halstead difficulty].

There is a lot of scope for these attributes for the further analysis. The other future analysis are mentioned as below:

One of the future analysis is, combining various source codes and detecting exactly why the defect is occurring and ways to mitigate the defect. Making the approaches proactive can be done in the future, instead of waiting until the software development process is completed, predicting a defect during the software development process is a best approach. Another future work is combining the automatic program repair with software defect prediction mechanism, for the predicted defects it automatically generates the repair and the ways.

References

1. Singh, Praman & Chug, Anuradha. (2017). Software defect prediction analysis using machine learning algorithms. 775-781. 10.1109/CONFLUENCE.2017.7943255.

2. Chug, Anuradha and Shafali Dhall. "Software Defect Prediction Using Supervised Learning Algorithm and Unsupervised Learning Algorithm." (2013).

3. Rodriguez, Daniel & Ruiz, Roberto & Cuadrado-Gallego, Juan & Aguilar-Ruiz, Jesús. (2007). Detecting Fault Modules Applying Feature Selection to Classifiers. 667-672. 10.1109/IRI.2007.4296696.

4. Bavisi, Shrey, Jash Mehta, and Lynette Lopes. "A Comparative Study of Different Data Mining Algorithms." International Journal of Current Engineering and Technology 4.5 (2014).

5. Arisholm, Erik & Briand, Lionel & Johannessen, Eivind. (2010). A systematic and comprehensive investigation of methods to build and evaluate fault prediction models. Journal of Systems and Software.83. 2-17. 10.1016/j.jss.2009.06.055.

6. Pandey, Ajeet Kumar & Goyal, Neeraj. (2009). A Fuzzy Model for Early Software Fault Prediction Using Process Maturity and Software Metrics. International Journal of Electronics Engineering. 1. 239-245.

7. Fenton, Norman & Krause, Paul & Neil, Martin & Lane, Crossoak. (2001). A Probabilistic Model for Software Defect Prediction.

8. diwan, sinan, and A. S. Mohamad. "Machine Learning Empowered Software Prediction System". *Wasit Journal of Computer and Mathematics Sciences*, vol. 1, no. 3, Oct. 2022, pp. 54-64, doi:10.31185/wjcm.61

9. T. M. Khoshgoftaar and J. C. Munson, "Predicting software development errors using software complexity metrics," in IEEE Journal on Selected Areas in Communications, vol. 8, no. 2, pp. 253-261, Feb. 1990, doi: 10.1109/49.46879.

10. C. L. Prabha and N. Shivakumar, "Software Defect Prediction Using Machine Learning Techniques," 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), 2020, pp. 728-733, doi: 10.1109/ICOEI48184.2020.9142909.

11. K. S. Kumar and R. B. Misra, "An Enhanced Model for Early Software Reliability Prediction Using Software Engineering Metrics," 2008 Second International Conference on Secure System Integration and Reliability Improvement, 2008, pp. 177-178, doi: 10.1109/SSIRI.2008.32.

12 . Muhammad Adnan Khan, Nouh Sabri Elmitwally, Sagheer Abbas, Shabib Aftab, Munir Ahmad, Muhammad Fayaz, Faheem Khan, "Software Defect Prediction Using Artificial Neural

Networks: A Systematic Literature Review", Scientific Programming, vol. 2022, Article ID 2117339, 10 pages, 2022. https://doi.org/10.1155/2022/2117339

13. Kotsiantis, Sotiris. "Feature selection for machine learning classification problems: a recent overview." Artificial intelligence review 42.1 (2011): 157-176.

14. K. Gao and T. Khoshgoftaar, "Software defect prediction for high-dimensional and class-imbalanced data," in *Proceedings of the 23rd SEKE*, vol. 2, pp. 89–94, Miami Beach, FL, USA, July 2011.

15. Kotsiantis, Sotiris. "Feature selection for machine learning classification problems: a recent overview." *Artificial intelligence review* 42.1 (2011): 157-176.

16. Alfian, Ganjar, et al. "Predicting Breast Cancer from Risk Factors Using SVM and Extra-Trees-Based Feature Selection Method." *Computers* 11.9 (2022): 136.

17. Sharaff, Aakanksha, and Harshil Gupta. "Extra-tree classifier with metaheuristics approach for email classification." *Advances in computer communication and computational sciences*. Springer, Singapore, 2019. 189-197.

18. Brownlee, Jason. "How to choose a feature selection method for machine learning." *Machine Learning Mastery* 10 (2019).

19. Cai, Jie, et al. "Feature selection in machine learning: A new perspective." *Neurocomputing* 300 (2018): 70-79.

20. Menzies, T., Ammar, K., Nikora, A., and Stefano, S., "How Simple is Software Defect Prediction?" Submitted to Journal of Empirical Software Engineering, October (2003)