International Journal of Engineering Trends and Technology
ISSN: 2231 – 5381 /doi:10.14445/22315381/IJETT-V69I4P203

Volume 69 Issue 4, 16-20, April 2021
© 2021 Seventh Sense Research Group®

TLHEL: Two Layer Heterogeneous Ensemble Learning for Prediction of Software Faults

Jyoti Goyal[1*], Bal Kishan[2]

[1]Department of Computer Science & Applications, Maharshi Dayanand University
[2]Department of Computer Science & Applications, Maharshi Dayanand University

[1]jyoti.goyal24@gmail.com, [2]balkishan248@gmail.com

Abstract
Software fault prediction is the most ubiquitous research concept in the domain of software engineering. Prior literature concedes the importance of ML techniques in the prediction of software faults, but the expedient method that gives consistently good results has still remained undetermined. So, to achieve high accuracy consistently, we have investigated many ensemble methods that advance the individual techniques and improves the performance of the fault prediction model. This paper proposed the novel TLHEL: two-layer heterogeneous ensemble model to predict software faults with less misclassification rate. The novelty of this model is that it combines the metric selection and training as a single process which reduces the computation overhead significantly, and performs feature selection with cross-validation, which particularly reduces the biasness of the model. The implementation of the TLHEL model will significantly increase the efficiency of the model.

Keywords: *Faults, heterogeneous ensembling model, Metrics, Stacking, Software fault prediction,*

I. INTRODUCTION

Software quality assurance is the important phase of every software industry because it ensures the desired quality of the software product. Predicting faults at an early stage significantly reduces the cost of fixing and correcting faults [1-4]. The idea behind the prediction of faults is to categorize the part of the software that is

Fault-prone from the part that is fault-free. This categorization is highly beneficial for the testing team because it is not possible to examine the whole system completely. So, it greatly maximizes the optimum utilization of scarce resources and also reduces the maintenance cost & effort. Software fault prediction using machine learning techniques is highly researched by many researchers. State-of-the-art research revealed that not a single classifier is always able to deal with all types of faults. To overcome this limitation, ensemble learning is introduced.

Ensemble learning is a machine learning technique where more than one single machine learning algorithm is trained and combined in order to produce a final output better than the outputs of individual algorithms [5-10]. There are two categories of ensemble learning; homogeneous and heterogeneous ensembles.[11] Inhomogeneous ensembling, every base learner is developed using the same kind of algorithm, but in the heterogeneous ensemble, every base learner is developed in a different way by employing different machine learning techniques. The final prediction is created by merging together each particular base learner prediction statistically [12,13]. Diversity and accuracy are two essential and vital conditions required to satisfy conditions and attain a decent ensemble. There are different ways to enhance the diversity of the models [14]:

1. Different machine learning algorithms
2. Different hyperparameter settings
3. Different feature sets
4. Different training sets
    - Bagging
    - Cross-validation

Stacking combines multiple base-level classifiers that are generated by using different learning methods (Wolpert, 1992). Similar to boosting, stacking classifiers use complex weighting schemes in comparison with bagging that uses simple uniform weighting schemes [15]. Several research papers showed that stacking could produce better performance in comparison with the voting [16,26]. So, here, we presented TLHEL: the two-layer heterogeneous ensembling learning that combines the output of different classifiers using stacking.
The main outline of the paper is. First, we present a few descriptions of the basics of fault prediction. Second, we define the problem that we observe in existing literature then we propose the solution. Third, we explain our proposed TLHEL: Two-layer heterogeneous ensemble learning with a flowchart. Fourth, we discuss the expected outcomes, and then finally, we conclude the findings of the paper.

II. LITERATURE REVIEW

In the 21st century, no one can imagine their life without technology. Without software, technology is like a human without a brain. So, good quality software plays the biggest role in everyone's life. To provide quality software, it is very important that the software should be fault-free. Many research works have been done in this field. Some of that recent work is discussed in this section. Ehsan Elahi et al. proposed an ensemble approach that is a model averaging method by employing different machine learning techniques with class balancing. The results prove that the model performs better than the other ensembling methods [17]. Thanh khuat et al. proposed an ensemble approach with data resampling techniques and data balancing techniques. The results show that the ensemble classifiers always perform better than the single classifiers [18]. Mohammad Zubair khan proposed a hybrid model and used eight datasets from the promise data repository to validate the results. The results state that Ada SVM and Bagging SVM are the best-performing classifiers [19].

Manu Banga et al. proposed a hybrid approach using PSO and a modified genetic algorithm for feature selection and classification of the faulty and non-faulty module. The algorithm is implemented on National Aeronautics and Space Administration Metric Data Program datasets [20]. Javad elmi et al. also proposed a multi-classifier system that proves the ensemble-based system is always better than standalone classifiers [21]. Mabayoje et al. design an approach where the wrapper feature selection method for selection of features and heterogeneous ensemble method is used to find the faulty modules. The results proved that naïve Bayes with genetic search feature selection gives the best results as compared to others [22]. Mohammad Zubair khan also performed a comparative study with individual classifiers and ensemble classifiers for fault prediction, and the results revealed that random forest is the best performing classifiers among all classifiers [23].

Abimbola G Akintola et al. performs the comparative analysis based on the heterogeneity of classifiers with feature selection techniques such as PCA, FSE, CFS for the prediction of defects. The outcome proves that the model will perform better when it is integrated with specific sets of features [24]. Zhiqiang L. et al. performed a study where the author focuses on two major problems is linearly inseparable data and highly imbalanced data; their proposed ensemble technique EMKCA outperformed the existing techniques [25]. Mohammad Akour et al. performed a comparative study between Bagging, Boosting, Stacking, and Base Learner classifiers. The empirical results prove that among 11 base classifiers, RF is the best performing classifier that can be blended with other classifiers to get an efficient fault prediction model [26].

All of the research work mentioned above uses different approaches for the prediction of faults, but no one focuses on all the dimensions collectively like data preparation, feature selection, data imbalance, etc.

III. PROBLEM DEFINITION

After doing a survey of the vast set of literature for software fault prediction, we find that almost all software fault prediction models are performing means similarly if they are producing similar performance figures, but the kind of defects detected by them are different. So, it leads to the following implications:

- First, we need to reconsider the performance metrics used for accessing the predictive capability of the model.
- Second, using only one model creates biasness in the model. So, we need to use ensembling.
- Third, advanced ensembling techniques need to be used, like stacking and blending.

In particular, the existing methods of combining outputs of different classifiers miss the contribution of the classifier that detects small but different defects not detected by the majority of classifiers. That's why the majority voting ensemble approach suffers from the problem of the sizeable property of defects. So, we require to enhance the combination strategies of ensembling [28].

Moreover, feature engineering performs a significant role in the efficiency of any defect prediction model. Mayhaps some models perform better when they are integrated with specific subsets of features or metrics. This is the reason why feature engineering plays a dominant role in increasing the predictive performance of the model.

So, we proposed new ways of building an enhanced defect prediction model that will address all these limitations.

IV. PROPOSED WORK

In this section, we propose a TLHEL approach that will address all the issues we discuss in the problem definition. The main objective of our proposed approach is to reduce the misclassification rate both in case of faulty or non-faulty. The efficiency of any model depends on how well it predicts the data, and its prediction depends on how well the model is trained. To effectively train the model, it is necessary that the data we use for training is good because it works on the simple principle GIGO: garbage in garbage out. So, our proposed approach, TLHEL, works on all the dimensions of data. The algorithm and architecture of our proposed approach TLHEL are mentioned in figure 1 and figure 2, where the whole dataset is divided into training and testing data. The training data is analyzed from various perspectives and prepare in such a way that it will efficiently train the model. It includes missing values treatment and outlier treatment. Outlier means all the values that are outside the range. Then we check the ratio of faulty and non-faulty classes. If it is not 1, it means the dataset is suffering from a class imbalance issue. So, to make it balance, we use a technique called SMOTE that generates artificial instances of the minority class. After that, we select those features that are highly correlated with the faults, then we train the base set of classifiers and validate it on the validation dataset. This process continues until we get the optimized model. The output of these classifiers will work as new instances for the meta-model. This meta-model is used to provide final predictions on real data.

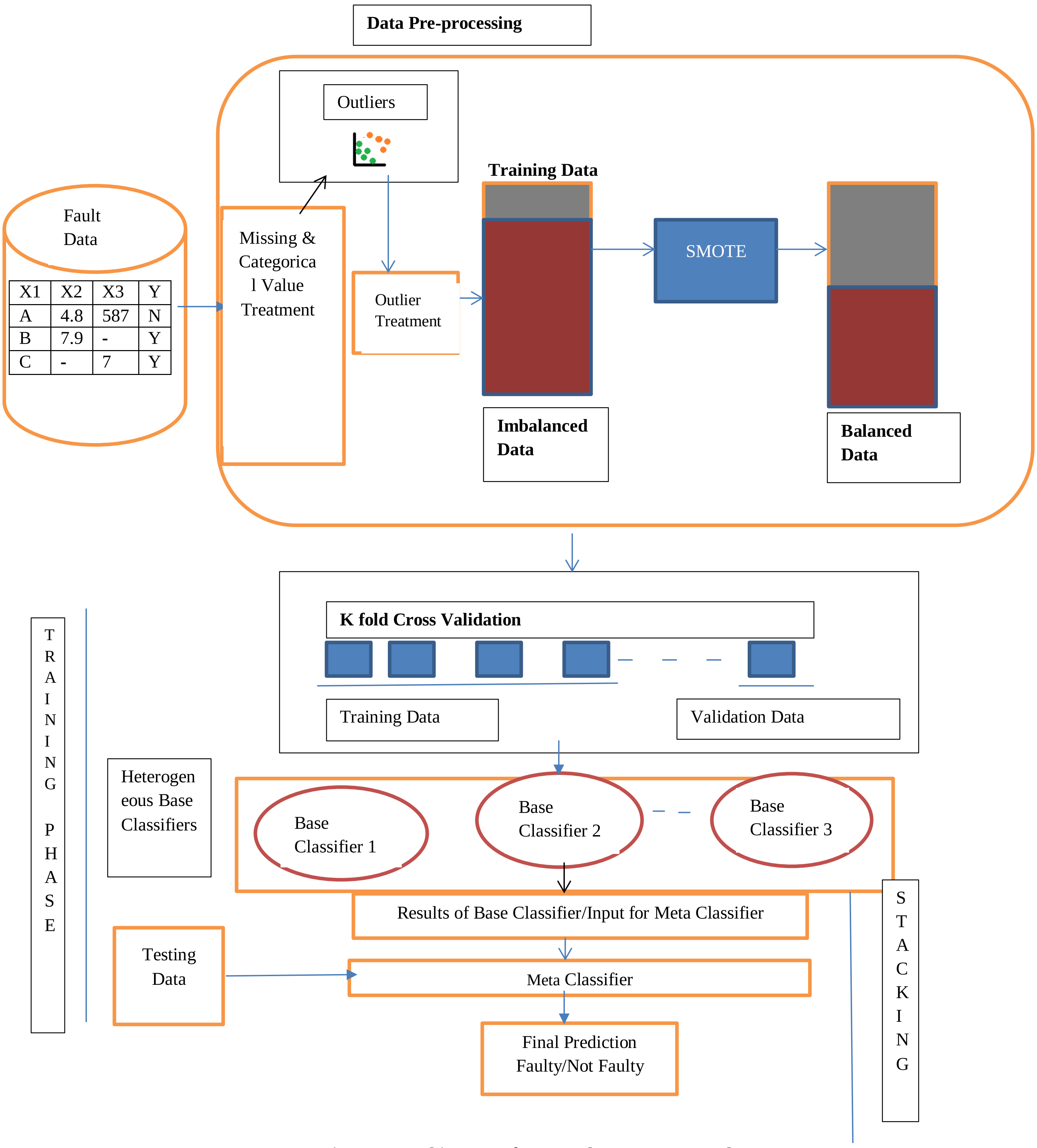

Figure 1: Architecture of Proposed TLHEL approach

The proposed TLHEL method given in the following figure is summarized here:

1. Input fault dataset with features and output labels.
2. Pre-process dataset by handling missing values and discretizing categorical data.
3. Perform feature engineering by using RFECV.
4. Perform class balancing using SMOTE
5. Normalize the features using the Minmax scaling method
6. Identify and treat outliers if they exist.
7. Divide the training dataset into training and testing datasets.
8. Train and Test the base learners with a reduced set of features with k fold cross-validation
9. The result of these base learners' classifiers will work as metadata for a meta learner.
10. Train the meta-learner with the newly constructed dataset.
11. Test the meta-model with the test dataset
12. Evaluate the novel TLHEL model in terms of F1-score, accuracy, ROC-AUC

These are the basic set of steps that we follow to build an efficient fault prediction model.

All these steps are majorly categorized into six major sections:

***A. Input fault data*:** The dataset for the prediction of faults is available in a Promise data repository. We can also use it from the GitHub repository. It can also be prepared using a bug tracking system.

B. Pre-processing of data: The data we extract from this repository is not always ready for model training. It sometimes contains missing values, outliers or it suffers from a class balancing problem. So, in this step, we handle all these problems.

C. Feature Engineering: It is the most crucial step of every machine learning model. So, we select those features that are highly correlated with faults. We also remove all those features that suffer from multicollinearity that unnecessarily increases the dimensions of the features. The model having fewer features always performs better.

***D. Base Classifiers*:** The classifiers used at layer 1 are base classifiers or weak learners. These base classifiers are selected on the basis of diversity so that they will be able to detect different faults. Diversity can be created in any of the ways mentioned in the above section.

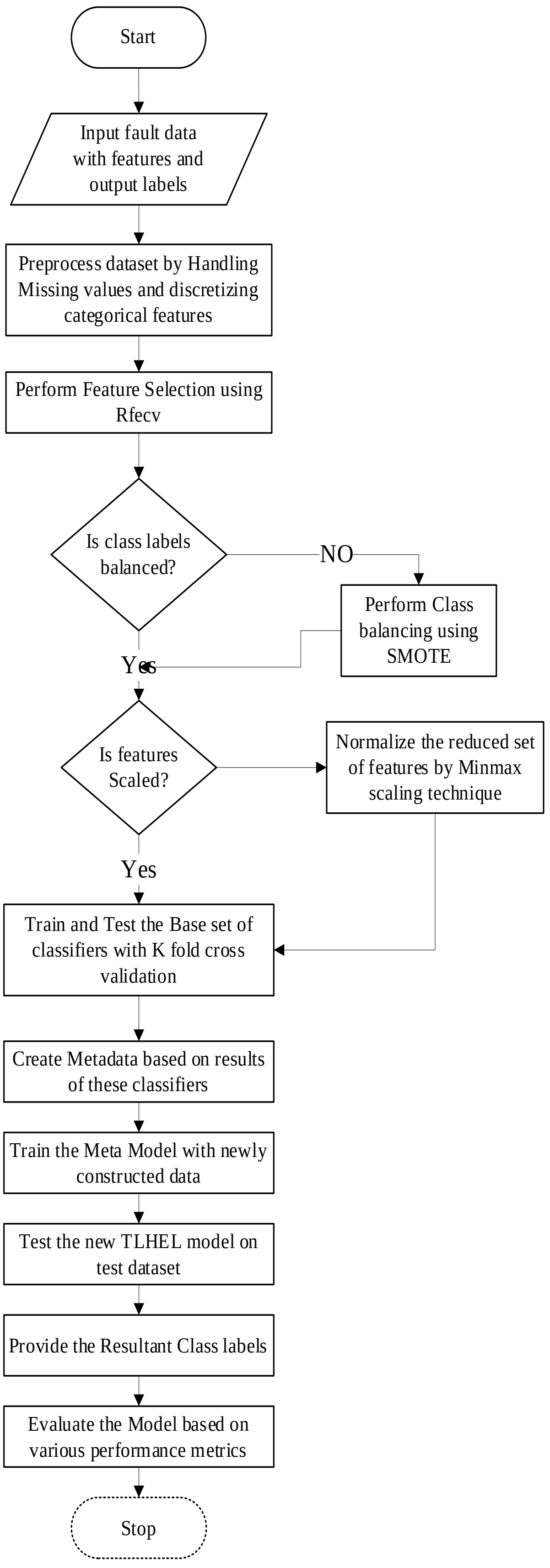
Start
Input fault data with features and output labels
Preprocess dataset by Handling Missing values and discretizing categorical features
Perform Feature Selection using Rfecv
Is class labels balanced?
NO
Perform Class balancing using SMOTE
Yes
Is features Scaled?
Normalize the reduced set of features by Minmax scaling technique
Yes
Train and Test the Base set of classifiers with K fold cross validation
Create Metadata based on results of these classifiers
Train the Meta Model with newly constructed data
Test the new TLHEL model on test dataset
Provide the Resultant Class labels
Evaluate the Model based on various performance metrics
Stop

Figure 2: Workflow of proposed TLHEL model

***E. Meta Learner*:** The classifier we select at layer 2 is called meta learner. The training of meta learner is performed on the output of the base learner. The output of the base learner will become the training data of meta learner. The final prediction of the real data will be provided by the meta learner.

All these modules are the major components of out TLHEL model. The implementation of this approach will appear in my next publication.

V. CONCLUSION AND FUTURE WORK

In this study, a TLHEL: two-layer heterogeneous ensemble learning is proposed that addresses the problem mentioned in the problem statement. We are not only concerned with improving the accuracy of the model but also with detecting the different faults that are possible only by utilizing different or heterogeneous or diverse classifiers. The diversity of the classifiers is achieved by selecting classifiers from different categories. But even after detecting different faults, they are not considered due to their small figure. So, we opt for the stacking technique that overcomes the problem of consideration of small but different defects. Therefore, it's a combined framework that manages the limitation of the data by using outlier removal techniques and data balancing techniques and provides an optimized fault prediction model. The findings of the proposed model will appear in my next publication.

REFERENCES

[1] Li, Ning, Martin Shepperd, and Yuchen Guo. A Systematic Review of Unsupervised Learning Techniques for Software Defect Prediction., Information and Software Technology 122(2020).

[2] Al-Shaaby, Ahmed, Hamoud Aljamaan, and Mohammad Alshayeb. Bad Smell Detection Using Machine Learning Techniques: A Systematic Literature Review., Arabian Journal for Science and Engineering 45(4)(2020) 2341–69. https://doi.org/10.1007/s13369-019-04311-w.

[3] Ensemble Approach to Code Smell Identification., (2019).

[4] Patchaiammal, P, and R Thirumalaiselvi., Software Fault Prediction Exploration Using Machine Learning Techniques., (6)(2019) 109–13.

[5] Zhou, Zhi-Hua.., Ensemble Learning., 1–5.

[6] Dietterich, Thomas G., Ensemble learning., The handbook of brain theory and neural networks 2(2002) 110-125.

[7] Polikar, Robi.., Ensemble learning., Ensemble machine learning. Springer, Boston, MA, (2012). 1-34.

[8] Yohannese, Chubato Wondaferaw, et al., Ensembles based combined learning for improved software fault prediction: A comparative study., 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE). IEEE, (2017).

[9] Yucalar, Fatih, et al., Multiple-classifiers in software quality engineering: Combining predictors to improve software fault prediction ability., Engineering Science and Technology, an International Journal 23.4(2020) 938-950.

[10] Sagi, Omer, and Lior Rokach., Ensemble learning: A survey., Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 8.4(2018) e1249.

[11] Petrakova, Aleksandra, Michael Affenzeller, and Galina Merkurjeva.., Heterogeneous versus Homogeneous Machine Learning Ensembles., (2015) 135–40.

[12] Laradji, Issam H., Mohammad Alshayeb, and Lahouari Ghouti.., Software Defect Prediction Using Ensemble Learning on Selected Features., Information and Software Technology 58(2015) 388–402. http://dx.doi.org/10.1016/j.infsof.2014.07.005.

[13] Wang, Tiejian, Zhiwu Zhang, Xiaoyuan Jing, and Liqiang Zhang., Multiple Kernel Ensemble Learning for Software Defect Prediction., Automated Software Engineering 23(4)(2016) 569–90.

[14] Bahler, Dennis, and Laura Navarro.., Methods for Combining Heterogeneous Sets of Classifiers., Proceedings of the 17th National Conference on Artificial Intelligence [American Association for Artificial Intelligence (AAAI) (2000) 1–6.

[15] Fan, David W, Philip K Chan, and Salvatore J Stolfo., A Comparative Evaluation of Combiner and Stacked Generalization., Proceedings of AAAI-96 Workshop on Integrating Multiple Learned Models: 40–46.

[16] Bowes, David, Tracy Hall, and Jean Petri. Software Defect Prediction: Do Different Classifiers Find the Same Defects?. (2017).

[17] Elahi, Ehsan., A New Ensemble Approach for Software Fault Prediction., 2020 17th International Bhurban Conference on Applied Sciences and Technology (IBCAST): (2020) 407–12.

[18] Tung, Thanh, Khuat My, and Hanh Le., Evaluation of Sampling-Based Ensembles of Classifiers on Imbalanced Data for Software Defect Prediction Problems., SN Computer Science 1(2)(2020) 1–16. https://doi.org/10.1007/s42979-020-0119-4.

[19] Education, I J Modern, Computer Science, and Mohammad Zubair Khan.., Hybrid Ensemble Learning Technique for Software Defect Prediction., (February)(2020) 1–10.

[20] Banga, Manu. 2020. Proposed Software Faults Detection Using Hybrid Approach., (2019) 1–14.

[21] Elmi, Javad, and Mahdi Eftekhari.., Dynamic Ensemble Selection Based on Hesitant Fuzzy Multiple Criteria Decision Making., Soft Computing. https://doi.org/10.1007/s00500-020-04668-3.,(2020).

[22] Mabayoje, M A et al.., Wrapper Feature Selection Based Heterogeneous Classifiers for Software Defect Prediction., (2019) (March).

[23] Alsaeedi, Abdullah, and Mohammad Zubair Khan.., Software Defect Prediction Using Supervised Machine Learning and Ensemble Techniques: A Comparative Study.,(2019) 85–100.

[24] Akintola, Abimbola, Abdullateef Balogun, Fatimah B Lafenwa-balogun, and Hammed Mojeed., "Comparative Analysis of Selected Heterogeneous Classifiers for Software Defects Prediction Using Filter-Based Feature Selection Methods., (2018) 1–6.

[25] Li, Zhiqiang, Xiao-yuan Jing, Xiaoke Zhu, and Hongyu Zhang., Heterogeneous Defect Prediction through Multiple Kernel Learning and Ensemble Learning., (2017).

[26] Akour, M, I Alsmadi, and I Alazzam., Software Fault Proneness Prediction: A Comparative Study between Bagging, Boosting, and Stacking Ensemble and Base Learner Methods." International Journal of Data Analysis Techniques and Strategies 9(1)(2017) 1–16. https://www.scopus.com/inward/record.uri?eid=2-s2.0-85018941954&doi=10.1504%2FIJDATS.2017.083058&partnerID=40&md5=20c0819792a0bee83561965240f9b392.

[27] D. I. George Amalarethinam, P. Mercy "MPTR_QoS: Multi-Path Trust Routing for Improving QoS in Heterogeneous IoT Based WSN" International Journal of Engineering Trends and Technology 69.3(2021):58-63.

[28] Ayidagn, Kassahun Azezew, and Shilpa Gite. , Analysis of Feature Selection Algorithms and a Comparative study on Heterogeneous Classifier for High Dimensional Data survey.