ISSN NO: 9726-001X

Volume 13 Issue 02 2025

JOURNAL OF CURRENT SCIENCE

Enhancing Software Effectiveness and Defect Prediction with Machine Learning

[1]Muddala Likhitha Kanaka Sri Sai, [2] Kella Madhuri, [3] Maddala Narendra, [4] Nelaparthi Kamal Raj, [5]Mr. L.D.R. Kishore,

[1,2,3,4] Student, Dept. of CSE, DNR College of Engineering & Technology, Balusumudi, Bhimavaram, India.

[5] Professor, Dept. of CSE, DNR College of Engineering & Technology, Balusumudi, Bhimavaram, India.

Abstract—

While contributing to both business outcomes and development mistakes, software defect prediction gives development teams measurable outputs. It is possible to aid developers in finding flaws and organizing their testing efforts by forecasting where code may be problematic. For early identification, the proportion of classification that provides the correct prediction is crucial. Additionally, because of their massive size, software-defect data sets are acknowledged and supported to some extent. Using the weka simulation tool, a hybridized strategy is used to solve this problem. The technique incorporates the PCA, random forest, naive bayes, and the SVM Software Framework. The five datasets included in the study are PC3, MW1, KC1, PC4, and CM1. By measuring and comparing the parameters of confusion, precision, recall, identification accuracy, etc., with the prevalent schemes, a systematic research study is carried out. According to the results of the investigation, the suggested method will provide better answers for predicting device failures. Software for predicting defects; metrics for software; prediction defect models; quality management systems; machine learning techniques;

Introduction

Over the last decade, people have increasingly put their emphasis on software-based systems, where the quality of the software is seen as the most important factor in user functioning. Due to the large volume of application software being produced, the issue of software quality is still not addressed, resulting in insufficient output for both private and industrial applications. Many sectors rely on designs for defect prediction. These models are useful for many tasks like as fault prediction, effort estimation, software reliability testing, hazard analysis, and more throughout the growth stage. Using a set of predetermined training data, a supervised machine learning system may make predictions. After the algorithm learns the ropes on the training dataset, it comes up with rules to forecast the class label on fresh data. During the learning phase, the predictor function is generated and strengthened using mathematical techniques. Both the input value and the specified output value are part of the training data utilized in this procedure. The output that is commonly known is compared with the predicted quality of the ML algorithm. Until the maximum number of loops is exhausted or the ideal prediction accuracy is achieved, this is repeated in several iterations of the training data. Unsupervised learning methods work in a world where the data value of the class label output is unknown. On the other hand, the program receives a data cluster, which it then processes using an algorithm to find patterns and correlations. Computer Science and Engineering at India's Thiagarajar College of Engineering in Madurai is the primary focus. On a social networking site, for instance, one may choose a circle of friends. Predicting faulty modules helps improve software quality. One of the first steps in identifying potentially flawed systems, such as units or classes, is defect prediction, which entails building models. The modules may be categorized as either defect prone or not prone to accomplish this. Support vector classifiers (SVCs), random forests, and naive bayes are among the most popular ways to determine the classification module. During the progress testing stages, the malfunction prone modules are prioritized, while the non-defect prone modules are tested when time and money allow. The classifier approach involves establishing and examining the feature of classification, which is the link between

ISSN NO: 9726-001X

Volume 13 Issue 02 2025

JOURNAL OF CURRENT SCIENCE

characteristics and the class label in the training dataset, using formulas for target categorization. In order to provide the labels for future datasets, those rules are also required. Classification patterns and a classifier may therefore be used to classify the unknown datasets. Due to the enormous deployment of software, researchers have repetitious job defining software problems, detecting the issue, and recognizing it. Sorting the software dataset into a defective and non-defective set is the primary objective when using the dataset as a model for bug prediction. In this approach, the user knows the real class values and feeds them into the classifier as input into the software dataset. Prior to this plan, metric approaches focused on requirements and design had shown promising outcomes. However, there is more work to be done in terms of algorithm design and forecast accuracy.

RELATED WORK

The software defect prediction model put forth by Wang et al. [3] for expanding the number of application software systems makes use of machine learning, a potent tool for prediction. Databases with bugs in them contain imbalanced data that generates haphazard patterns. Developing a trustworthy scenario classifier for use in academia and business is motivated by this dilemma. According to Xu et al. [4], who studied "software defect prediction strategies," the conventional wisdom is that in order to improve the effectiveness of defect prediction strategies, one must first reduce the number of superfluous features. A maximum data point, piece of information

The imbalanced nature of software defect outcomes was recently addressed by Duksan et al. [5], and throughout the prediction process, relatively few occurrences exhibit characteristics that correspond to the faulty class. Because this stage reduces software industry efficiency, it need a particular categorization system. To fix it, we turn it into a multi-objective optimization problem and use a multi-goal learning system by studying a diverse cross-project environment. "Relied on a popular approach in machine learning, namely SVM (support vector machine)" was the technique used by Shan et al. [6]. Additionally, attribute predictability is addressed by combining a support vector classifier with a locally linear embedding method. In fact, this method configures SVM constraints with a grid search technique and a tenfold cross-validation procedure. The LLE-SVM is effective in fault detection, according to the experimental results.

The Predicting Software Deficiencies using a neural network technique, which incorporates the neural network idea together with the Bayesian methodology as a radial foundation, was introduced by Yang et al. [7]. While the motivation-minimization strategy is often used for weight realization, enhancing the weight update framework with one or two Gaussian structures may improve the radial neural network's efficiency. In their suggested model for stable program quality estimate based on software development, Han et al. [8] say that... To improve prediction outcomes, our method incorporates a computer-assisted software safety estimate, a Rayleigh model, a system building forecast model, and an enhanced software reliability template. A model that describes the symptoms of design uncertainty based on an aspect-oriented method of evaluating uncertainty has been put forth by Parthipan et al. [9]. It has been noted that most defect prediction models are created at the design process or at the code level. These models are used for binary classification, which involves determining which faults are trustworthy and which are not, or for regression analysis, which involves estimating the total number of flaws. According to Panichella et al. [10], "a unified predictor of defects that incorporates into account the clusters supplied by diverse methodologies of machine learning" improved the ability to identify software projects that are prone to faults. The authors Felix et al. [2] have suggested research into software fault prediction utilizing a neural network–centric machine learning approach. For the purpose of studying defect prediction, this paper considers Github datasets. The use of a NN is used to gain classification and prediction by using registry linkages between software programs and their errors. Feature section and reduction will enhance performance in machine learning-based classification and prediction strategies. The way we talk about something is as an important part. In order to investigate how a semi-supervised learning method for software defect prediction was put into practice, Lu et al. [12] "used a version of the algorithm for self-study". The study's authors came to the conclusion that trust fitting might stand in for current supervised techniques. The semi-supervised technique outperformed a random forest model with dimensional reduction and training modules that included common faults. A meta-study of all the elements impacting output in predictions was carried out by Shepperd et al. [13]. By analyzing the factors that significantly affect the software defect classificatory's predictive abilities, they ensured that their defect prediction system was effective, according to calculations based on the Matthews correlation coefficient. Classifier choice, they found,

ISSN NO: 9726-001X

Volume 13 Issue 02 2025

JOURNAL OF CURRENT SCIENCE

had a little effect on output, but model building factors—that is, features unique to the research group—had a large one. This is due to the fact that the research team is responsible for doing preliminary data processing. Using an uneven dataset of software faults, Jayanthi et al. [1] "developed a Selection of characteristics for applications approach. Subsets of attributes are subsequently collected once the selection using wrapper-based attributes is executed. To mitigate the effects of the imbalanced dataset, the next step involves using random sampling.

RESEARCH METHODOLOGY

Included in the compilation are the most popular and widely used machine learning techniques. Here are several methods along with brief descriptions of what they include. As an applied training method for statistical technique knowledge grouping, Naive Bayes (ABN) is the first to be discussed. This method casually asserts that the properties of a certain class are autonomous, as the name suggests, which is naive. Characteristics take on strong or innocent seclusion. Assigned as a vector for drawing class descriptors from limited sets, it serves as a template that problematic items might utilize as class labels. Because to their lack of complexity and reliance on generalizations, naive bays fall within the category of real-world issues. The Random Forest B Building a framework that can forecast value functions from various inputs is the algorithm's main goal. The interior nodes each represent a different input parameter. There are limits on the progeny of each of these parameters for all possible values. The objective factor, represented by the leaf in the tree, is the one that the input factor's defined parameters may cross to, from the root to the leaf. The learning technique maps the item's analysis to the desired quality interpretation using the random forest statistical model. Mining, statistics, and machine learning all make use of this prediction method. A. SVC Data used for classification and regression analysis may be understood with the help of the learning approach. A support vector machine (SVM) model is a set of test samples spread throughout a range and partitioned as evenly as possible according to their distributional class. The fresh samples are categorized according to the side of the gap they fell into after mapping into a specific region. For classification and correlation, support vector machines (SVMs) build a set of hyperlens in a space without dimensions. The greatest distance between a group of points in a certain class to a hyperplane is called the functional margin. In the end, the difference between the generalization error and the functional margin was inversely proportional. Section D. Neural Sytem These neurons are part of a network that allows them to communicate and exchange data.

The following is a definition of how an ANN works. To begin, at the input layer node, the neural network takes the data variable values. The links that connect nodes are given weights. The numerical Weights are adjusted based on the NN's ability to learn and adapt. Moving across the network involves crossing nodes and determining the values of variables. Each connection's weight influences the parameter value. Parameter values are compared to goal values at the output node, where the predicted effect is calculated.

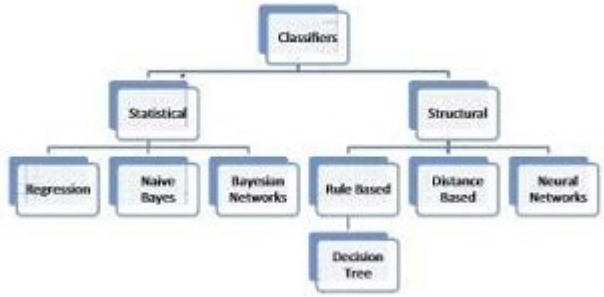


Fig 1. Machine learning techniques Classification

PROPOSED APPROACH

A crucial problem in software engineering is the prediction of software errors. Methods for defect prediction based on machine learning software were covered in the previous chapter. Researchers still have a long way to go before they can solve the problems of software bug mismatch, classification accuracy, and general efficiency using these approaches. An artificially dependent neural network technique for software defect prediction and a hybrid feature reduction scheme are introduced to address this issue. In the first part of the article, you'll find an improved principal component analysis (PCA) approach for dimensionality reduction and numerical modeling; in the second part, you'll get information on using the neural network in conjunction with the present PCA method. Part A: PCA Primary component analysis, or PCA, is a statistical method. Principal component analysis (PCA) is a way to

ISSN NO: 9726-001X

Volume 13 Issue 02 2025

JOURNAL OF CURRENT SCIENCE

reduce the size of large datasets while increasing the computing complexity and reducing information loss. Reducing the dataset's dimensionality is the objective of primary component analysis (PCA), a computer approach. The process of transforming the data into a new quaternion is often called a linear orthogonal transformation. The main issue is that principal component analysis (PCA) is more of a method for extracting items than a tool for feature selection. Characteristics are evolving into new ones as a result of linear variation. Features with the least variance should be used to execute the reduction. Such articles often made use of principal component analysis (PCA) to enhance the efficacy of their experiments. According to this, the principal component analysis (PCA) approach transforms n vectors {x1,x2,...,xn} from a d-dimensional space into n vectors {x1,x2,...,xn} in a new dimensional space. Feature selection technique in the suggested approach identifies and prioritizes the data set's most useful functionality for learning and prediction, which is a major benefit. The content is simplified and the learning approach may be made more efficient. Classifying and forecasting outcomes rely heavily on input data.

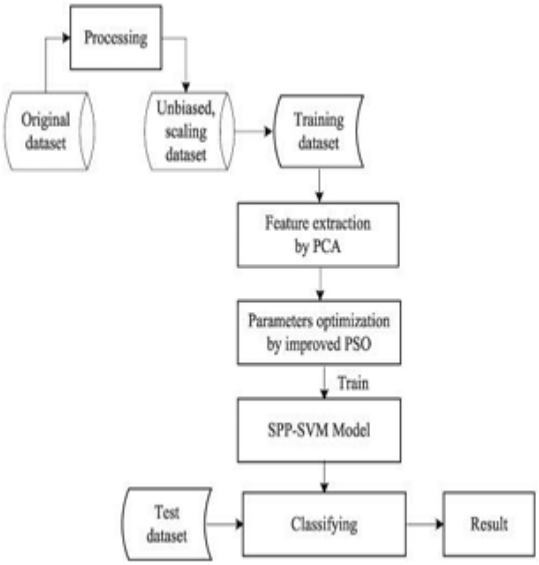


Fig 2. Software Defect Prediction Model PSO stands for Particle Swarm optimization.

This approach finds the best possible answer to the issue by making the first, candidate solution better. Each potential candidate in this technique is referred to as a particle locally. Based on its location and velocity, this particle uses a system to travel the search region. Finding the optimal placements in the search space is another parameter that relies on serialization. In large search spaces, PSO may locate a solution that is close to optimum. Using this approach to find the answer is not guaranteed. Applications of this approach span several domains, from artificial neural networks (ANN) and fuzzy controllers to optimization issues. If you're in the jungle with a group of buddies, you should know that there's just one way out. Their understanding of the way to departure is limited to only one dimension, relative distance. Everyone involved in this problem may be thought of as a particle, and just like any other particle, they will have a certain location and speed. Each particle's speed is proportional to its distance from the exit point, as stated by PSO. It is possible to make anything more appealing by adjusting certain factors in PSO. A single implementation with few tweaks may serve a wide range of purposes. A wide range of industries are making use of Particle Swarm Optimization, from those dealing with massively complex technology to those with one-of-a-kind applications that prioritize narrowly defined criteria. B. DataSet Access a number of databases online without paying a dime. Database of Kaggle promise problems. The data sets were backed by five datasets: KC1, PC3, PC4, MW1, and CM1. Numeric defect level KC1 classes and data retrieval (here referred to as AT) were both used. The datasets used in this research are from the Kaggle PROMISE database and include many characteristics; they are KC1, CM1, PC3, MW1, and PC4. A count of characteristics, a breakdown of useable components, a breakdown of defective components, and a defective percentage are all shown in the tables, which pertain to the hypothetical dataset.

ISSN NO: 9726-001X

Volume 13 Issue 02 2025

JOURNAL OF CURRENT SCIENCE

Dataset	Precision	Recall	F1-measure	Support
KC1	T-1.00 F-1.00	F-1.00 T-0.99	F-1.00 T-0.99	F-438 T-90
CM1	T-0.99 F-1.00	N-1.00 Y-0.88	N-1.00 Y-0.93	N-117 Y-8
PC4	N-1.00 Y-1.00	N-1.00 Y-0.94	N-1.00 Y-0.97	N-260 Y-18
MW1	N-0.89 Y-0.87	N-0.89 Y-0.89	N-0.96 Y-0.93	N-55 Y-9
PC3	N-0.87 Y-1.00	N-0.97 Y-0.94	N-1.00 Y-0.92	N-32 Y-238

as contrasted with other state-art approaches. There are a plethora of measures to consider, including recall, uncertainty matrices, accuracy, rate, false-positive rate, and precision. These classification tests may be used to create a matrix that contains the expected and actual class values.

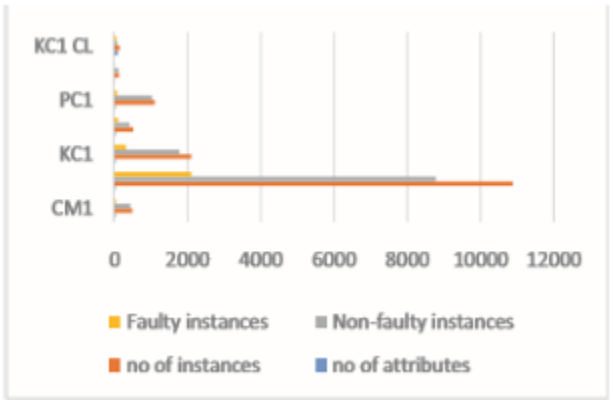


Fig 3.Characteristics of Dataset

In the above figure, we can see the data sets that pertain to the amount of defective, non-faulty, and real characteristics for each case type.

RESULTS AND ANALYSIS

All of the datasets were from the Kaggle promise dataset repository. We make use of five datasets: KC1, PC3, C4, MW1, and CM1. Naive Bayes, Random Forest, SVC (Linear Regression), and Principal Component study (PCA) are the methods that were selected for the study. Collecting the data sets in arff format from the Kaggle database is made easier with the help of the r studio tool. Consequently, data sets that were compatible with it were rendered by running the data processing section mast. The table below displays the summary of the test results. The accuracy value of each approach, expressed as % concordance, is shown. The most prevalent heuristic in a dataset is labeled as such, among other things. Table I: Evaluation of SVC Performance The results show that out of the five datasets that were tested, the linear classification approach has the highest accuracy in defect prediction, making it the most precise and accurate methodology. Neural networks, naive Bayes, and random forests were the other three algorithms, and they could only achieve maximum accuracy on a single dataset.

TABLE II. PERFORMANCE EVALUATION FOR RANDOM FOREST

Dataset	Precision	Recall	F1-measure	Support	Acc
KC1	T-0.96 F-0.75	F-0.99 T-0.88	F-1.00 T-0.97	F-438 T-90	0.99
CM1	T-0.88 F-0.92	N-0.85 Y-0.93	N-0.92 Y-0.76	N-117 Y-8	0.98
PC4	N-0.95 Y-0.86	N-1.00 Y-0.93	N-0.97 Y-0.79	N-260 Y-18	0.98
MW1	N-0.96 Y-0.85	N-0.94 Y-1.00	N-0.85 Y-0.84	N-55 Y-9	0.97
PC3	N-0.94 Y-0.84	N-0.85 Y-0.95	N-0.89 Y-0.99	N-32 Y-238	1.00

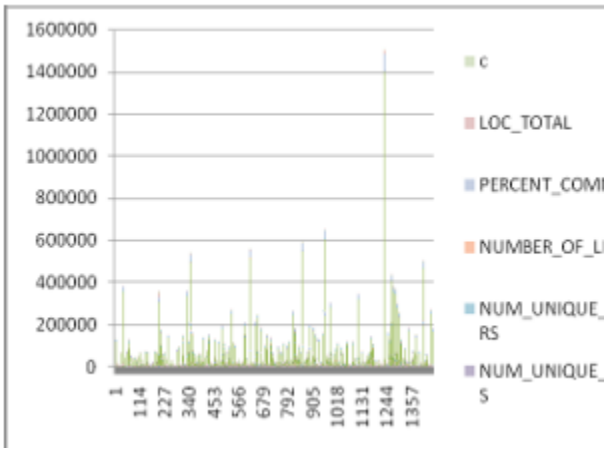
A description of each method's standard deviation loss is included in the table above. It reveals that each method's percentage phrases are incorrectly positive. Among many approaches, the dataset's top algorithm is marked to stand out. Table III: Assessment of

ISSN NO: 9726-001X

Volume 13 Issue 02 2025

JOURNAL OF CURRENT SCIENCE

Naïve Bayes Performance
You can see the test's standard deviation from the anticipated fault in the table up there. The neural network approach has the best success rate. Overcoming the tie would be helped by the failure rate. If two algorithms are tied in terms of defect prediction accuracy, the one with the less error will win.



Tdata stands for the datasets used for training our prediction algorithms, whereas Vdata stands for the datasets used for testing. Both the findings and the forecast were shown in the Test and Pred tables, respectively.

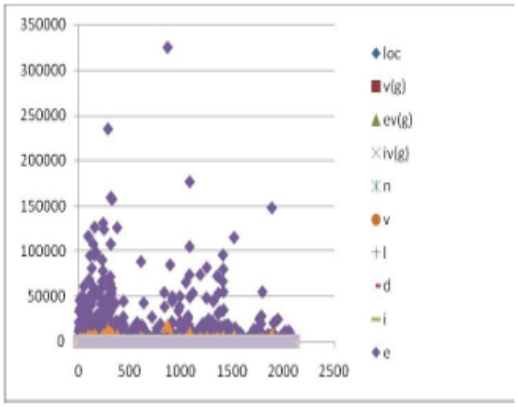


Fig 5.Standard Deviation for each Dataset From the analysis,

Based on the results of the research, it seems that neural networks, followed by random forests, have the lowest failure rates. In addition, a linear classification (SVC) has the highest accuracy value. To determine which technique works best when there is a disagreement over accuracy estimate, error rate parameters might be taken into account.

CONCLUSION AND FUTURE WORK

Predicting software flaws using information-mining methods is the primary goal of this study. In addition, this area has grown into a sizable research topic, with many different approaches used in an effort to find ways to make software defect detection and bug prediction more efficient. As part of our research, we developed a novel hybrid model by combining

Dataset	Precision	Recall	F1-measure	Support	Acc
KC1	T-0.99 F-1.00	F-1.00 T-0.88	F-1.00 T-0.93	F-117 T-8	0.96
CM1	T-0.98 F-0.86	N-0.96 Y-0.93	N-0.97 Y-0.89	N-104 Y-27	0.97
PC4	N-0.98 Y-0.86	N-0.96 Y-0.93	N-0.97 Y-0.89	N-109 Y-16	1.00
MW1	N-0.92 Y-0.96	N-0.94 Y-0.93	N-0.96 Y-0.98	N-131 Y-29	0.99
PC3	N-0.91 Y-0.86	N-0.93 Y-0.89	N-0.97 Y-0.81	N-32 Y-238	0.98

included in order to get a feature reduction template, with overall probability additionally used to reduce the data retrieved by PCA. Problems in code may also be found using the neural network classification technique. According to the research, the suggested strategy significantly outperforms several state-of-the-art models in terms of efficiency and obtains an AUC of 98.70 percent. In order to demonstrate, without bias, how important the feature selection impact is. Whenever feature selection procedures are used, or when no approach is used, there is no discernible variance in classifier accuracy. When using approaches for function selection without a set of features, classifier accuracy suffers. As a result, it is clear that feature selection strategies reduce the time and space challenge of defect prediction without compromising prediction accuracy. Using many

ISSN NO: 9726-001X

Volume 13 Issue 02 2025

JOURNAL OF CURRENT SCIENCE

datasets may enhance these insights even further. Improving the results is possible with more datasets. Additional methods may be compared as well. This research took into account the most popular and often utilized strategies. The problem of classification accuracy for large datasets will likely be addressed in the future, and new approaches will be shown and utilized for deep study of feature reductions and classification. PCA is

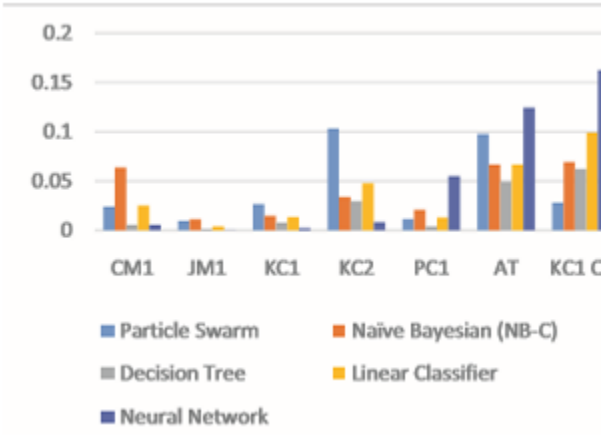


Fig 6.Overall Algorithm classification on Dataset.

The study's findings suggest that, after random forest, neural networks may have the lowest failure rate. The dimensional classification, on the other hand, has the highest detection rate. The failure rate parameter may be used to ascertain the proper result in the event of a tie accuracy forecast.

REFERENCES

[1]. Jayanthi, R. and Florence, L., 2019. Software defect prediction techniques using metrics based on neural network classifiers. Cluster Computing, 22(1), pp.77-88.

[2]. Felix, E.A. and Lee, S.P., 2017. Integrated approach to software defect prediction. IEEE Access, 5, pp.21524-21547.

[3]. Wang, T., Zhang, Z., Jing, X., Zhang, L.: Multiple kernel ensemble learning for software defect prediction. Autom. Softw. Eng. 23, 569–590 (2015).

[4]. Xu, Z., Xuan, J., Liu, J., Cui, X.: MICHAC: defect prediction via feature selection based on maximal information coefficient with hierarchical agglomerative clustering. In: 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER), Suita, pp. 370–381 (2016).

[5]. Ryu, D., Baik, J.: Effective multi-objective naïve Bayes learning for cross-project defect prediction. Appl. Soft Comput. 49, 1062 (2016).

[6]. Shan C., Chen B., Hu C., Xue J., Li N.: Software defect prediction model based on LLE and SVM. In: Proceedings of the Communications Security Conference (CSC ’14), pp. 1–5 (2014).

[7]. Yang, Z.R.: A novel radial basis function neural network for discriminant analysis. IEEE Trans. Neural Netw. 17(3), 604–612(2006).

[8]. K. Han, J.-H. Cao, S.-H. Chen, and W.-W. Liu, “A software reliability prediction method based on software development process,” in Quality, Reliability, Risk, Maintenance, and Safety Engineering (QR2MSE), 2013 International Conference on. IEEE, 2013, pp. 280–283.

[9]. S. Parthipan, S. Senthil Velan, and C. Babu, “Design level metrics to measure the complexity across versions of ao software,” in Advanced Communication Control and Computing Technologies (ICACCCT), 2014 International Conference on. IEEE, 2014, pp. 1708–1714.

[10]. A. Panichella, R. Oliveto, and A. De Lucia, “Cross-project defect prediction models: L’union fait la force,” in Software Maintenance,

[11]. Bautista, A.M., Feliu, T.S.: Defect prediction in software repositories with artificial neural networks. In: Mejia, J., Munoz,M., Rocha,Á., Calvo-Manzano, J. (eds.) Trends and Applications in Software Engineering.Advances in Intelligent Systems and Computing, vol.405. Springer, Cham (2016).

[12]. H. Lu, B. Cukic, and M. Culp, “Software defect prediction using

ISSN NO: 9726-001X

Volume 13 Issue 02 2025

JOURNAL OF CURRENT SCIENCE

semisupervised learning with dimension reduction," in Automated Software Engineering (ASE), 2012 Proceedings of the 27th IEEE/ACM International Conference on. IEEE, 2012, pp. 314–317.