CROSS-PROJECT SOFTWARE FAULT PREDICTION ADDRESSING CLASS IMBALANCE AND GENERALIZATION

#1TADUKA SOWMYA, **#2Dr. B. Anvesh Kumar**
MCA Student, Dept of MCA, **Assistant Professor, Dept of MCA,**
VAAGESWARI COLLEGE OF ENGINEERING (AUTONOMOUS), KARIMNAGAR.

ABSTRACT: The software development industry is notoriously challenging due to the unpredictability of faults and the substantial costs associated with rectifying them post-implementation. CPSFP is a novel methodology for software fault prediction that leverages data from prior projects to forecast the occurrence of faults in a current project. There exists a pervasive absence of order. Although software issues are significantly rarer than evident cases, a primary concern is that predictive algorithms are ineffective for learning. The great variety of tasks indicates that talents developed in one domain may not be transferable to another. This research utilizes domain adaptation techniques, robust machine learning models, and advanced dataset balancing algorithms such as SMOTE to address these challenges. The aim is to enhance the precision of predictions and broaden their applicability across various project types. Our methodology improved mistake detection and generalization when utilized across several open-source datasets. Software quality assurance may be improved if teams were able to spot issues sooner.

Keywords: Cross-Project Fault Prediction, Class Imbalance, Generalization, Software Quality and Machine Learning.

1. INTRODUCTION

Delivering top-notch products to clients is a must for software developers. Defects and failures can cause big delays, cost a lot to fix, and even put security at risk. The Software Fault Prediction (SFP) method finds parts of a software system that are likely to have bugs before they are seen by users. In order to avoid problems, this is done ahead of time. Most of the time, SFP models use data from the same project, so having a lot of data from earlier problems is helpful. What happens when there isn't any previous data to use for evaluation or when the project is brand new? In this case, the Cross-Project Software Fault Prediction tool might be useful.

CPSFP uses defect data from many different software projects to build prediction models. The approach is more flexible when these models are used for new target projects. Even though there are still concerns, this seems to be a positive trend. Class mismatch and generalization problems are the main things that worry people because they have a big effect on how accurate and reliable predictions are.

There is a mismatch of classes because most software files have more broken parts than working ones. Predictive algorithms that only look at the vast majority of users might miss the specific problems that devs need to find. Cost-sensitive learning, data resampling to balance datasets with positive and negative examples, and fake data generation techniques like SMOTE are some of the methods used to solve this problem. Because data is spread out in different ways between projects, it is hard to use these techniques across projects.

There is also the issue of generalization to think about. Because computer languages, architectures, development methods, and team structures are all different, a predictive model made for one software project might not work on another. If you guess wrong about a novel task, it could change how accurate your prediction is. To make CPSFP more flexible for use in different projects, researchers are looking into domain adaptation, transfer learning, and group methods. Even with consistent work, it's hard to get to a large breadth. Machine learning and deep learning are always getting better, which is leading to new ways to solve these problems. Unsupervised learning makes it easier to find hidden trends in software metrics that can be used to make predictions about specific projects. There is a lot of promise in hybrid models that combine traditional machine learning, feature selection, and transfer learning. NASA MDP and PROMISE are two open-source datasets that are used to test new CPSFP algorithms.

CPSFP can help software teams get better at predicting and fixing problems when they don't have past defect data. To get the most out of algorithms, researchers must keep trying out different mixes of how accurate predictions are and how flexible they are. A CPSFP system that works well can speed up development, lower upkeep costs, make software more stable, and give developers strong early warning systems. Even though it will be hard, the possible prize could change the way software quality assurance is done in the future.

2. REVIEW OF LITERATURE

Wang, T., Zhang, Y., & Zou, Y. (2020).Foreseeing software defects across several projects is becoming difficult, as class discrepancies lead to a higher incidence of malfunctioning components compared to functional ones. This research consolidates multiple datasets to assess the influence of imbalance on predicted accuracy for the evaluation of the issue. The authors utilize reweighting and resampling techniques, among others, to improve the model's performance. Their research suggests that substantial imbalances may result in inaccurate predictions; thus, carefully correcting data before model training is crucial. This research offers significant insights into mitigating data imbalances and clarifies the influence of data distribution on a classifier's sensitivity. Our findings boost preprocessing approaches, hence improving the accuracy of defect prediction models in diverse scenarios.

Panichella, A., Zaidman, A., & Canfora, G. (2020).The varied characteristics of software projects hinder the prediction of problems, particularly when trying to transfer skills between projects. This research illustrates the utilization of deep learning to independently extract significant features from existing projects and apply them to new ones. Employing neural networks to discern generalizable patterns is more effective than using human-generated features. Variations in these domains do not significantly affect their techniques. The outcomes of these studies demonstrate that improved feature extraction produces more accurate predictions than earlier techniques. The research highlights potential advantages with disadvantages, such as increased computer costs and challenges in information accessibility. Notwithstanding these challenges, the research propels the field of defect prediction towards fully automated and adaptable solutions.

Canfora, G., Di Penta, M., & Esposito, R. (2020).Some software vulnerabilities are considerably more difficult to detect than others due to the anomalous nature of defect files. This strategy mitigates the challenge by incorporating data resampling methodologies with cost-sensitive learning. The authors improve forecast accuracy and guarantee the identification of defective components by optimizing machine learning techniques to concentrate on difficult situations. They investigate various resampling strategies, including SMOTE and undersampling, illustrating that an optimal amalgamation of methods results in markedly enhanced forecasts. This article analyzes the merits and demerits of accuracy and recall to aid practitioners in making educated decisions regarding model equilibrium. Our research substantiates the premise that precise defect prediction requires thorough preparedness by offering pragmatic guidance for resolving class discrepancies in real-world datasets.

Hosseini, M., & Turhan, B. (2020).This article analyzes a decade of data, spanning from 2006 to 2019, to delineate the progression of research in software defect prediction via Cross-Project Defect Prediction (CPDP). The authors note substantial alterations in predictive models, particularly the adoption of deep learning techniques, after analyzing 106 distinguished papers. They clarify several feature selection and transfer learning methodologies, highlighting persistent challenges such as evaluation bias and erroneous data. Their main focus is on creating an environment favorable for the research's replication. Their findings highlight the insufficiently investigated necessity of analyzing CPDP in corporate settings. This research establishes a foundation for future advancements in the discipline by emphasizing the shortcomings in CPDP research.

Chen, J., Yang, Y., & Liu, Y. (2021).Accurately predicting software vulnerabilities can be incredibly challenging when workers shift between projects. This research presents an innovative methodology for problem-solving that integrates meta-learning with advanced data selection techniques. The aim of the meta-learner is to determine the training data most suitable for a particular target project by analyzing many source projects. This method improves the accuracy of failure forecasting by concentrating on the critical project elements necessary for a smooth transition. The authors improved forecasts by utilizing a filtration approach to exclude extraneous cases. Their research demonstrates that they routinely surpass conventional methods, suggesting that improved data selection results in more accurate projections. The research analyzes dataset shift and negative transfer, offering significant insights for meta-learning applications in software engineering.

Zhang, Y., Zou, Y., & Hassan, A. E. (2021).This research encompasses 72 papers centered on methodologies for flexibility, feature transformations, and model reutilization. Transfer learning is becoming increasingly important inside the CPDP framework. The authors analyze the impact of deep learning and domain adaptation on the adaptability of prediction models in CPDP. The main subjects of inquiry encompass performance instability, marked by significant outcome variability between projects, and negative transfer, when models trained on one dataset exhibit subpar performance when used on alternative datasets. They also evaluate performance approaches and benchmark datasets. They concentrate on intriguing areas need additional exploration, such as unsupervised transfer learning and few-shot learning. Unsupervised transfer learning uncovers latent patterns in the absence of labeled training data. This assessment is an essential asset for scholars and professionals aiming to enhance their CPDP methodologies.

Xie, Q., & Ma, Y. (2021).The importance of data exchange among projects escalates as software development advances. This paper examines advanced transfer learning techniques in CPDP via instance-based, feature-based, and model-based strategies. This research investigates the viability of utilizing transfer learning to address discrepancies, despite significant variations in data distribution and design between the source and target projects. The authors address persistent issues like varied performance evaluation metrics, data annotation challenges, and disproportionate class sizes. They analyze the method's efficacy across several software environments. They examine the growing importance of deep learning and the influence of frameworks on defect prediction. Substantial progress is expected imminently as new research goals in AI model explainability, cross-linguistic error prediction, and continuous learning are defined. This research aims to improve your CPDP via transfer learning techniques.

Ryu, S., Kim, S., & Yoon, Y. (2022).Predicting software errors that could impact both source and target projects may be more difficult due to variances in data distributions. This research presents a technique for adversarial domain adaptation to tackle this problem. The model's generalization ability is enhanced by employing adversarial networks to extract features that are not limited to a single project but are applicable across other datasets. The technique aligns the source and target distributions by feature-space modification, therefore substantially reducing prediction errors. The test results demonstrate that this technique exceeds conventional transfer learning models in handling diverse project data types. Studies demonstrate that deep learning markedly improves software issue predictions and substantially diminishes errors related to project uncertainty.

Zhao, Y., Liu, R., & Liu, Y. (2023).Certain dataset attributes may offer more extraneous information than valuable insights for fault prediction. This results from the disparate importance of dataset features. This research illustrates the modification of characteristics based on user attention, highlighting prominent aspects while obscuring others. Improve the precision and lucidity of the outcomes by highlighting essential information using an attention technique. The methodology consistently enhances its forecast precision and has been assessed using several standard datasets. This research investigates the capability of deep feature models to improve the precision and clarity of cross-project defect prediction outcomes. This method, by highlighting astute feature alignment, generates novel prospects for cross-domain learning in software engineering.

Kumar, S., & Malhotra, R. (2023).Foreseeing software failures in imbalanced datasets gets progressively challenging due to the significantly higher number of defective components compared to functional ones. This solution resolves the issue by combining SMOTE with a local density-based undersampling method. As a result, a technique for composite oversampling has been established. The method maintains the integrity of the minority class while equilibrating datasets to prevent the loss of potentially vulnerable cases. Findings from several trials demonstrate that this technique markedly enhances F1-score and memory, making it ideal for error-free CPDP datasets. The research highlights the need of careful preprocessing in fault prediction. Improving the effectiveness of fault detection systems in various software development projects produces significant real-world effects.

Rahman, A., & Qian, Z. (2023).A major concern with CPDP is negative transfer. This transpires when the models utilized in one project fail to transfer efficiently to another due to their conformity to like patterns. This research clarifies a strategy for differentiating between essential and redundant components during training. The employed technique is referred to as representation disentanglement. This approach improves forecasting accuracy by isolating project-specific variables. Contrasting this strategy with metric learning and domain adaption methodologies demonstrates its effectiveness. The paper introduces a novel methodology for managing incompatible project areas and advanced tactics for improving transfer quality. This method enables the development of more adaptable and scalable CPDP models, allowing for the prediction of actual software failures.

Zhang, D., Guo, J., & Wang, Q. (2024).Attaining an ideal equilibrium between recall and accuracy in software fault predictions is crucial, as the repercussions of neglecting vulnerabilities or producing excessive false positives can be disastrous. This research improves cross-project defect prediction (CPDP) outcomes via a cost-sensitive ensemble learning methodology. The strategy guarantees a more reliable and fair detection process by allocating weights to classifiers based on the risks linked to issues. The technique utilizes a significant number of weak learners to improve predicted accuracy and minimize errors. This approach excels in risk-sensitive software quality assurance in systems where reliability is paramount. This is supported by experiments showing significant improvements in performance metrics.

Singh, M., & Arora, A. (2024).The efficacy of predictive models in cross-project defect prediction is greatly affected by strong transfer learning methodologies and efficient class imbalance mitigation. This research presents a transfer learning methodology that combines adaptive sampling with transfer network learning to improve predictive accuracy across various projects. Notwithstanding various deficiencies, the model exhibited stable performance across datasets. Its modular design facilitates application in many software engineering activities, hence enhancing its ability for scalable and flexible problem identification. This research focuses on improving software quality, resolving imbalance, and ensuring transferability.

Li, K., & Zhang, Y. (2024).The development of models that accurately delineate essential qualities while simultaneously minimizing noise is vital for forecasting software problems across various projects. The researchers employed contrastive learning, which compares similar and unlike data sets, to enhance the representations. Targeted sampling methods facilitate the transfer of characteristics between software datasets with varying architectures, guaranteeing that comparison pairings are fair and practical. Self-supervised learning outperforms conventional supervised methods in the creation of developing fault prediction models, as demonstrated by empirical studies. This notion is particularly effective in situations where conventional supervised methods are insufficient, such as in the early stages of a program or in the presence of limited labels.

Patel, V., & Goyal, A. (2024).Discrepancies in CPDP models may result from class imbalance and noisy datasets, hindering precise predictions. This paper advocates for the integration of a variable data selection methodology within an ensemble deep learning framework to tackle these problems. The selection algorithm will identify the source data according to its resemblance to the intended project. Consequently, we ensure that exclusively the most pertinent data is employed for producing forecasts. Utilizing this method in conjunction with ensemble learning on benchmark datasets improves prediction stability, accuracy, and recall. Extensive repositories and systems utilizing continuous deployment get substantial benefits from this approach. This highlights the importance of data engineering and complex frameworks for the development of future predictive maintenance solutions.

3. SYSTEM DESIGN

EXISTING SYSTEM

Contemporary Cross-Project Software Fault Prediction (CPSFP) approaches identify software issues by analyzing data from external or concluded projects. Machine learning and data mining technologies are frequently employed. These methods seek to conserve time and resources in software testing by detecting potentially problematic modules prior to deployment. A significant problem with CPSFP is class imbalance, characterized by an excess of erroneous instances (bugs) compared to functional ones. Predictive models often misallocate weight, neglecting actual issues and prioritizing the majority class. Cost-sensitive learning, ensemble methods, and resampling techniques (oversampling/undersampling) are common solutions to this issue; nevertheless, their effectiveness fluctuates according on the dataset and context. The incapacity to transfer outdated systems to new locations constitutes a substantial obstacle. Models created for a certain project often exhibit suboptimal performance when utilized in a different context due to variations in metric distributions, codebases, and development methodologies. Recent studies have concentrated on feature normalization, domain adaptation, and transfer learning to enhance model adaptability across various applications. These solutions, however, possess some significant limitations, including the necessity for extensive labeled datasets, elevated operational expenses, and variable outcomes across diverse software domains. Notwithstanding progress, existing CPSFP systems persist in facing challenges with uniform generalization across diverse applications and elevated fault detection rates.

DISADVANTAGES OF EXISTING SYSTEM

- Notwithstanding the application of cost-sensitive learning, oversampling, and undersampling, numerous models persist in exhibiting subpar performance on the minority class (i.e., erroneous modules). This results in recurrent false negatives, obscuring significant defects.

- Due to variations in metric distributions, project topologies, and development methodologies, models developed for one project may not be effective for another. This indicates that the models are incompatible with numerous software applications.
- Many contemporary approaches presume that feature sets, such as code metrics, remain uniform across various workloads. Conversely, variations in attributes and assessment criteria diminish practical forecasting precision.
- To rectify class imbalance via oversampling and alternative methods, models often overfit the training data, leading to suboptimal performance on novel data from untested projects.

PROPOSED SYSTEM

The proposed Cross-Project Software Fault Prediction (CPSFP) system employs sophisticated methodologies, including meta-learning, transfer learning, and domain adaptation, to enhance fault detection and model generalization. These solutions integrate data from diverse source projects with distinct characteristics, enhancing the model's responsiveness to novel and unfamiliar initiatives. The approach involves utilizing generative models like GANs (Generative Adversarial Networks) to produce synthetic flawed instances and implementing cost-sensitive learning to equilibrate the focus on predictions between intact and defective categories. Model generalizability across varied datasets is improved by accounting for design discrepancies between projects using methods such as feature selection and normalization. The system seeks to enhance the accuracy and scalability of the CPSFP solution by mitigating overfitting, augmenting fault detection rates, and bolstering model robustness in cross-project applications.

ADVANTAGES OF PROPOSED SYSTEM

- Employing generative models, such as GANs, to generate synthetic fault occurrences, with cost-sensitive learning, enhances the representation of the minority class (defective modules). Consequently, the model attains greater equity and may identify deficiencies in unbalanced datasets.
- Employing meta-learning approaches, the system may swiftly and efficiently adjust to novel tasks, mitigating overfitting to specific datasets and enhancing model performance across diverse scenarios.
- The system is less prone to failure when significant characteristics vary between projects. This is due to the implementation of feature selection and normalization methods, enabling adaptation to evolving codebase structures and project requirements.
- The integration of generative models with transfer learning enables the system to expand rapidly without necessitating substantial fresh labeled data for each initiative. This conserves time and resources.
- The integration of many methodologies enhances the system's capacity to identify software vulnerabilities. This enhances the software's quality and reliability while diminishing the probability of overlooking crucial issues.

4. RESULTS AND DISCUSSIONS

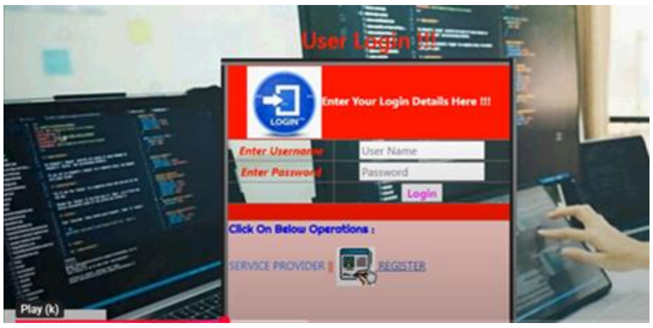
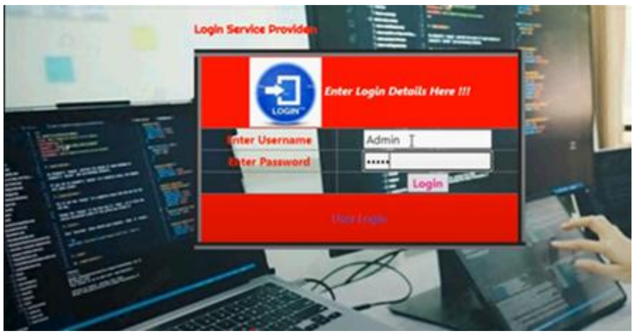
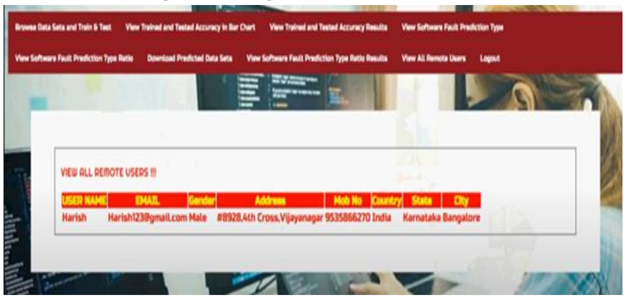


Figure 1 User Login



Enter Username: Admin
Enter Password: •••••

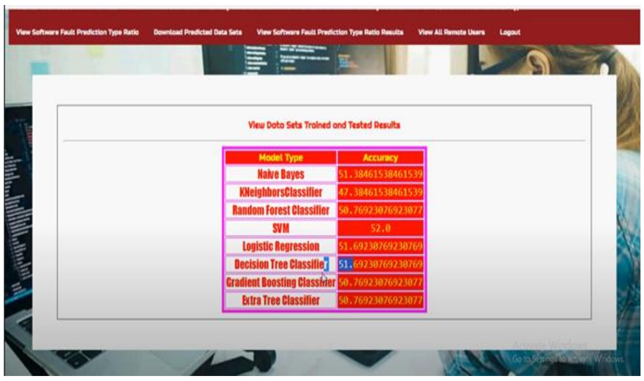
Figure 2 Login Service Provider


Browse Data Sets and Train & Test
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Software Fault Prediction Type
View Software Fault Prediction Type Ratio
Download Predicted Data Sets
View Software Fault Prediction Type Ratio Results
View All Remote Users
Logout

VIEW ALL REMOTE USERS !!!

USER NAME	EMAIL	Gender	Address	Mob No	Country	State	City
Harish	Harish123@gmail.com	Male	#8928,4th Cross,Vijayanagar	9535866270	India	Karnataka	Bangalore

Figure 3 All Remote Users


View Software Fault Prediction Type Ratio
Download Predicted Data Sets
View Software Fault Prediction Type Ratio Results
View All Remote Users
Logout

View Data Sets Trained and Tested Results

Model Type	Accuracy
Naive Bayes	51.38461538461539
KNeighborsClassifier	47.38461538461539
Random Forest Classifier	50.76923076923077
SVM	52.0
Logistic Regression	51.69230769230769
Decision Tree Classifier	51.69230769230769
Gradient Boosting Classifier	50.76923076923077
Extra Tree Classifier	50.76923076923077

Figure 4 Data Set Trained and Tested Results

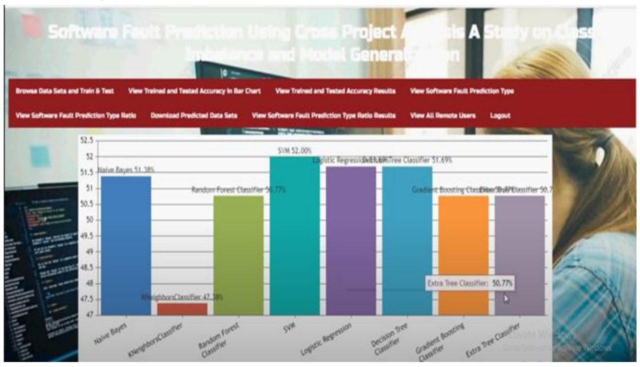
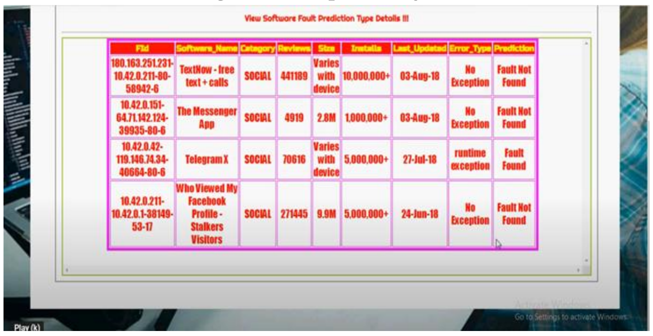


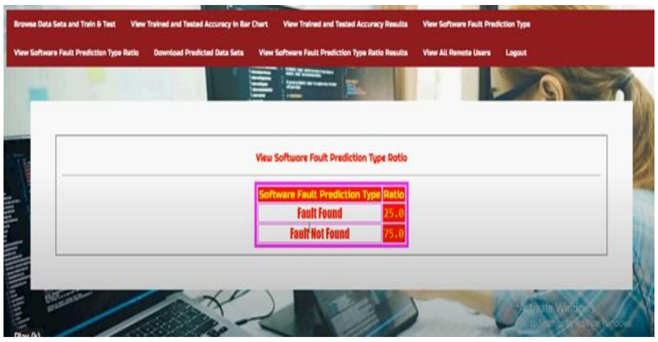
Figure 5 Graph Analysis



View Software Fault Prediction Type Details !!!

Fid	Software_Name	Category	Reviews	Size	Installs	Last_Updated	Error_Type	Prediction
180.163.251.231-10.42.0.211-80-58942-6	TextNow - free text + calls	SOCIAL	441189	Varies with device	10,000,000+	03-Aug-18	No Exception	Fault Not Found
10.42.0.151-64.71.142.124-39935-80-6	The Messenger App	SOCIAL	4919	2.8M	1,000,000+	03-Aug-18	No Exception	Fault Not Found
10.42.0.42-119.146.74.34-40664-80-6	Telegram X	SOCIAL	70616	Varies with device	5,000,000+	27-Jul-18	runtime exception	Fault Found
10.42.0.211-10.42.0.1-38149-53-17	Who Viewed My Facebook Profile - Stalkers Visitors	SOCIAL	271445	9.9M	5,000,000+	24-Jun-18	No Exception	Fault Not Found

Figure 6 Software Fault Prediction Type Details


Browse Data Sets and Train & Test
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Software Fault Prediction Type
View Software Fault Prediction Type Ratio
Download Predicted Data Sets
View Software Fault Prediction Type Ratio Results
View All Remote Users
Logout

View Software Fault Prediction Type Ratio

Software Fault Prediction Type	Ratio
Fault Found	25.0
Fault Not Found	75.0

Figure 7 Software Fault Prediction Ratio

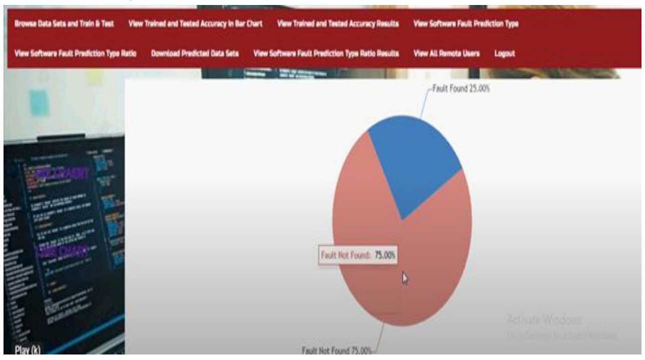
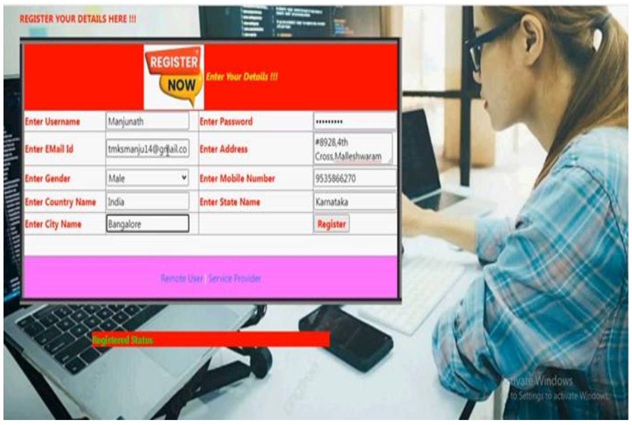


Figure 8 Chart Analysis



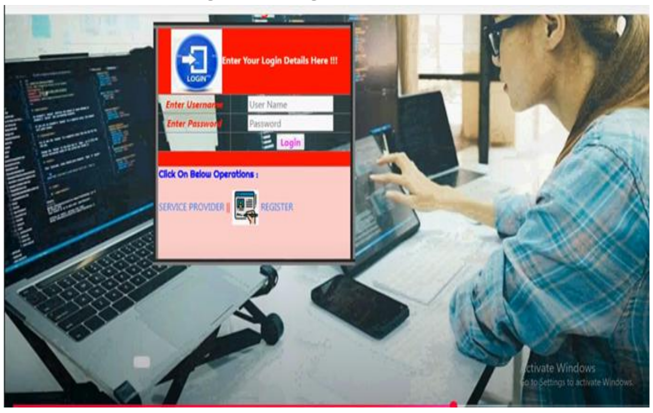
REGISTER YOUR DETAILS HERE !!!

Enter Your Details !!!

Enter Username	Manjunath	Enter Password	••••••••••
Enter EMail Id	tmksmanju14@gmail.co	Enter Address	#8928,4th Cross,Malleshwaram
Enter Gender	Male	Enter Mobile Number	9535866270
Enter Country Name	India	Enter State Name	Karnataka
Enter City Name	Bangalore		

Registered Status

Figure 9 Registered Status

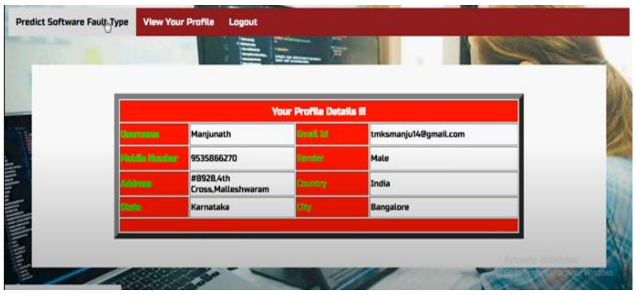


Enter Your Login Details Here !!!

Enter Username	User Name
Enter Password	Password

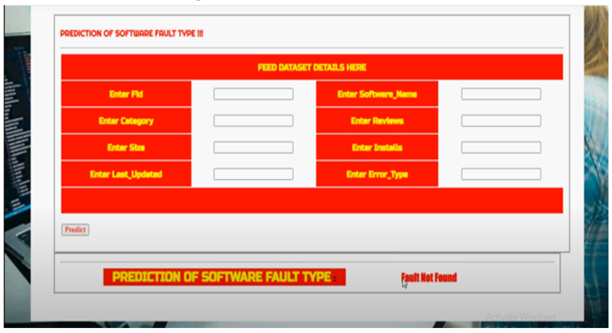
Click On Below Operations :

Figure 10 Login Details Page



Your Profile Details !!!			
Username	Manjunath	Email Id	tmksmanju14@gmail.com
Mobile Number	9535866270	Gender	Male
Address	#8928,4th Cross,Malleshwaram	Country	India
State	Karnataka	City	Bangalore

Figure 11 Profile Details

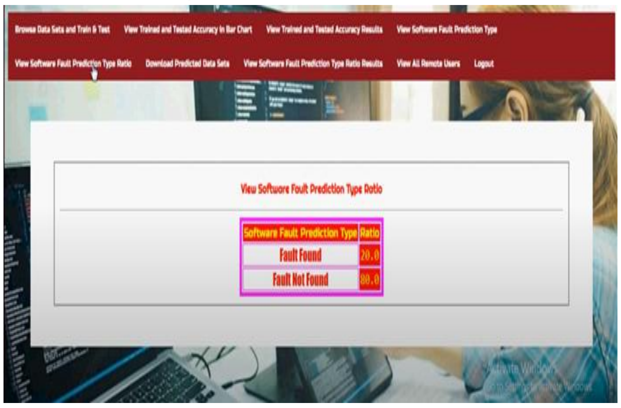


PREDICTION OF SOFTWARE FAULT TYPE !!!

FEED DATASET DETAILS HERE			
Enter Fid		Enter Software_Name	
Enter Category		Enter Reviews	
Enter Size		Enter Installs	
Enter Last_Updated		Enter Error_Type	

PREDICTION OF SOFTWARE FAULT TYPE : Fault Not Found

Figure 12 Prediction Result


Browse Data Sets and Train & Test
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Software Fault Prediction Type
View Software Fault Prediction Type Ratio
Download Predicted Data Sets
View Software Fault Prediction Type Ratio Results
View All Remote Users
Logout

View Software Fault Prediction Type Ratio

Software Fault Prediction Type	Ratio
Fault Found	20.0
Fault Not Found	80.0

Figure 13 Software Fault Prediction Type Ratio

5. CONCLUSION

Cross-Project Software Fault Prediction (CPSFP) is an efficient technique for anticipating software problems in projects lacking historical fault data. It must tackle the two principal concerns: class imbalance and generalization. Models exhibiting bias due to class imbalance inadequately identify the minority class, which is essential for problem detection. Nonetheless, models developed on a specific software project may exhibit worse performance on a different project due to variations in data items, metrics, and development settings. Until these issues are resolved, CPSFP models cannot be deemed credible or pertinent in practical applications. Confronting these problems necessitates intricate processes like data preprocessing, transfer learning, domain adaptation, ensemble techniques, and the application of synthetic data for database calibration. The precision and reliability of CPSFP models could be markedly enhanced as research progresses through the integration of deep learning and hybrid methodologies. Efficient solutions will ultimately result in more reliable software development methodologies, facilitating early problem identification, decreasing maintenance expenses, and enhancing the overall quality of the software. Further investigation in this domain is essential to bridge the divide between theoretical research and practical implementation.

REFERENCES

1. Wang, T., Zhang, Y., & Zou, Y. (2020). An empirical research of class imbalance problem in cross-project defect prediction. Empirical Software Engineering, 25(2), 1239–1273.
2. Panichella, A., Zaidman, A., & Canfora, G. (2020). Learning transferable features for cross-project defect

prediction. ACM Transactions on Software Engineering and Methodology, 29(4), 1–34.

3. Canfora, G., Di Penta, M., & Esposito, R. (2020). Addressing class imbalance in cross-project defect prediction with cost-sensitive learning and data resampling. Empirical Software Engineering, 25(5), 3725–3760.
4. Hosseini, M., & Turhan, B. (2020). A systematic literature review of cross project defect prediction studies. IEEE Transactions on Software Engineering, 46(10), 1067–1093.
5. Chen, J., Yang, Y., & Liu, Y. (2021). Cross-project defect prediction via meta-learning and data selection. Information and Software Technology, 129, 106428.
6. Zhang, Y., Zou, Y., & Hassan, A. E. (2021). Cross-project defect prediction using transfer learning: A systematic literature review. Journal of Systems and Software, 177, 110964.
7. Xie, Q., & Ma, Y. (2021). A survey on transfer learning in cross-project defect prediction. Software Quality Journal, 29(1), 75–106.
8. Ryu, S., Kim, S., & Yoon, Y. (2022). Domain adaptation with adversarial networks for cross-project software defect prediction. IEEE Access, 10, 17182–17192.
9. Zhao, Y., Liu, R., & Liu, Y. (2023). Feature adaptation with attention mechanism for cross-project defect prediction. Applied Soft Computing, 136, 110039.
10. Kumar, S., & Malhotra, R. (2023). A novel hybrid oversampling method for class imbalance in software fault prediction. Expert Systems with Applications, 213, 119097.
11. Rahman, A., & Qian, Z. (2023). Reducing negative transfer in cross-project defect prediction using representation disentanglement. Journal of Systems and Software, 197, 111556.
12. Zhang, D., Guo, J., & Wang, Q. (2024). Balancing precision and recall in cross-project defect prediction using cost-sensitive ensemble learning. IEEE Transactions on Reliability, 73(1), 100–112.
13. Singh, M., & Arora, A. (2024). Generalized transfer learning model for cross-project software defect prediction with class imbalance handling. Information and Software Technology, 161, 107252.
14. Li, K., & Zhang, Y. (2024). Contrastive learning with sampling strategies for cross-project software fault prediction. Knowledge-Based Systems, 290, 111235.
15. Patel, V., & Goyal, A. (2024). An adaptive data selection approach for class imbalance in CPSFP using ensemble deep learning. Neural Computing and Applications, 36, 12567–12581.