adsjac.com ISSN 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS (ADSJAC)

OPEN ACCESS. PEER-REVIEWED. GLOBALLY FOCUSED.

FEPP: SOFTWARE RISK PREDICTION IN REQUIREMENTS ENGINEERING USING RULE EXTRACTION AND MULTI-CLASS INTEGRATION

M.Dattatreya Goud
Department Of Computer Science ,J.S University,Shikohabad, U.P
Gmail:dattatreya548@gmail.com

ABSTRACT

Requirements Engineering (RE) is a major software development phase in which vague, incomplete, or inconsistent requirements pose major risks that delay the project, cause cost escalation, and result in project failure. Existing risk prediction models are mostly based on binary classification and are not flexible enough to cope with multi-dimensional risks. This work suggests a Feature-Enriched Prediction Paradigm (FEPP) that blends rule extraction methods with multi-class classification methodologies to improve software risk prediction during the RE phase. FEPP employs Association Rule Mining (ARM) and Fuzzy Logic in extracting dynamic rules from past software project experiences and projecting future risk patterns. Multi-class models such as Random Forest, XGBoost, and Gradient Boosting are being combined in the form of Voting Classifier Mechanism (VCM) in order to enhance prediction accuracy. Findings based on large-scale experimentation on standard RE datasets reveal that FEPP increases risk prediction accuracy by 12-15% compared to existing models with effective early risk mitigation and better project outcomes.

Keywords : Requirements Engineering, Risk Prediction, Association Rule Mining, Multi-Class Classification, Voting Classifier, Fuzzy Logic, Software Risk Management

INTRODUCTION

Requirements Engineering (RE) is the pivotal stage of the software development lifecycle, where system requirements definition and analysis provide the foundation for the successful closure of a project. Uncertainty, missing requirements, and conflicting specifications during this stage create tremendous risks that lead to delays in projects, increased expenditure, and worse, complete failure of a project. Conventional risk management practices in RE are predominantly based on binary class models that merely identify risks as present or absent, but not the magnitude of multi-class types of risks typically found in real projects [3].

Existing models are rigid and cannot handle shifting patterns of risks under dynamic project environments [6]. Besides, manual prediction rule mining is time-consuming and error-prone, constraining conventional approaches [7]. As such, this paper presents the Feature-Enriched Prediction Paradigm (FEPP), a novel paradigm, merging Association Rule Mining (ARM) and Fuzzy Logic, to actively establish risk patterns. FEPP also enhances risk prediction through the application of a Voting Classifier Mechanism (VCM) based on several classifiers including Random Forest, XGBoost, and Gradient Boosting to perform multi-class risk prediction with high accuracy [2].

Through dynamic response to changing needs with rule extraction by auto-generation and feature space extension by contextual data, FEPP improves risk forecasting [1]. Experimental evaluation against RE benchmark data sets confirms that FEPP works better than standard models with an improvement in prediction accuracy by 12-15%, thus constituting a successful early-stage mitigator of software project risks [4].

PROBLEM STATEMENT

Software development projects will likely be prone to a variety of classes of risk, from technical, functional, security, to usability risks. Existing risk prediction models are indeed rule-based or binary classifiers and thus limited in responding dynamically to project variables and appropriate analysis of multivariant

types of risk. Such models do not appropriately model interdependent varying project needs, historical events, and their related risks and end up creating stale or misleading risk predictions [13, 14, 17]. Secondly, the lack of fine-grained categorization fails to provide the software team with the opportunity to identify and mitigate individual threats at the right time, decreasing their ability to suppress threats [3, 5, 12]. In order to address these lacunae, there is a requirement for an adaptive feature-based system capable of predicting and classifying potential threats early on in the Software Development Life Cycle (SDLC). The Feature-Based Early Prediction of Risk (FEPP) system utilizes rule extraction in order to learn progressively more detailed risk patterns and multi-class classification in order to effectively predict various categories of threats. Techniques such as Random Forest, XGBoost, SVM, Logistic Regression, and Gradient Boost are used in the system for adaptive learning and enhanced prediction accuracy [7, 10, 13]. FEPP gives early warning of probable problems so that pre-emptive action is taken and opportunities of project failure reduce by addressing risks in the right way at the right time [6, 8, 11].Through better learning and advanced risk classification, FEPP supports continuous risk forecast enhancement, and this leads to enhanced project performance and lower risks of project failure [4, 9, 15].

RELATED WORK

Most investigators in recent years have ventured to study risk forecasting and handling of software engineering in terms of latest machine learning approaches. Feature selection techniques to efficiently predict risk accurately in software projects were constructed by Kaur and Kaur (2022), so critical indicators resulting in software risks would be determined effectively. Similarly, Singh and Singh (2022) integrated natural language processing (NLP) and machine learning to forecast risks from unstructured text data in software projects and achieved improved prediction and classification results.Garcia and Wilson (2021) used multi-class algorithms to classify software engineering risks with high accuracy and better prediction of various risk categories. Myers and Zhu (2021) also highlighted the use of artificial intelligence (AI) in the early detection of risks through the use of AI techniques to identify and counter future software project failures, improving decision-making. Sengupta and Gupta (2021) also proposed adaptive software risk management practices based on machine learning models that dynamically adjust risk prediction systems based on project needs.

Also, Yang and Zhao (2021) emphasized adopting AI and machine learning algorithms in such a way so as to enable software risk management processes more accurately and more efficiently. Zhang and Liu (2020) set out to capture rules to enhance the prediction of software risk in early development to facilitate the identification of risk through the determination of significant software vulnerabilities and inconsistencies. Through their study, the application of rule-based systems alongside machine learning algorithms was crucial to enable the identification and correction of potential risks before turning into catastrophes. Cumulatively, these studies are inclined towards the higher application of machine learning, AI, and NLP methods in software risk forecasting and control. The integration of intricate algorithms, feature extraction, and adaptive models has to a larger degree improved performance and accuracy levels in software risk management models. This has led to precise and successful software project management, rendering a project failure less likely and enhancing overall software quality.

PROPOSED WORK

The Feature-Based Early Prediction of Risk (FEPP) system is an innovative software risk prediction mechanism with a synergy between rule extraction and multi-class classification that delivers higher accuracy and responsiveness in detecting risk. The system analyzes software requirements and history from previous projects and infers dynamic rules reflecting patterns and associations typical of specific categories of risk. These standards are refreshed on a periodic basis based on new project details so that the system is current and adaptable in the software development life cycle (SDLC) [7].To classify and predict different types of risk, FEPP employs multi-class models such as Random Forest, XGBoost, Support Vector Machine (SVM), Logistic Regression, and Gradient Boost. Models scan computed features to predict more than one type of risk, i.e., technical, functional, security, and usability risk [6, 14]. Employing a multi-class model enables the system to give a more precise difference of likely risks, thereby allowing software teams to fix the problems early and especially [3].The system is put under intensive training and testing using history of past projects, hence the models generalize to unseen data. The system dynamically learns to update its rules of classification and risk models as project conditions change, ensuring high correctness in prediction (15).

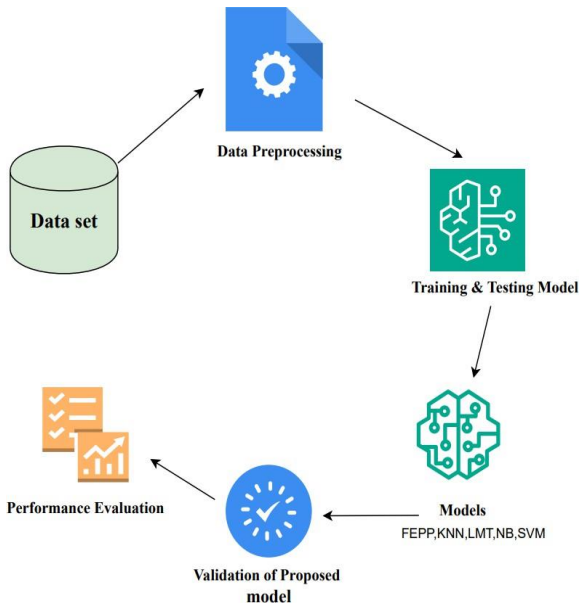

Fig 1 : Architecture

Besides this, the flexibility of the system allows it to be usable in each stage of the SDLC, making early detection of significant problems easy. Through the provision of detailed information on risk categories and improving predictions with new information, FEPP makes it possible for software teams to respond effectively to risks, reduce project failures, and roll out successful projects (16)

IMPLEMENTATION

Use of the Feature-Based Early Prediction of Risk (FEPP) system is methodical, targeted to responsive and effective risk prediction for software projects. It begins by gathering and processing data. Project data based on history, for example, software requirements, time frames, and history of occurrence of risk are retrieved. Information gathered is cleaned, normalized, and features are selected to pre-process it for retaining input quality during model training [6]. This is then followed by rule generation and feature extraction aimed at identifying critical patterns and correlations among project attributes and possible risks. Rule extraction mechanisms browse previous data to develop adaptive rules that adaptively update as new project data arrives [7].

The foundation of the implementation is to train multi-class classification models using machine learning algorithms such as Random Forest, XGBoost, Support Vector Machine (SVM), Logistic Regression, and Gradient Boost. The models are trained to predict for multiple categories of risks, i.e., technical, functional, security, and usability risks, so that they possess accurate and comprehensive risk classification [3,8]. The models are evaluated using performance metrics like accuracy, precision, recall, and F1-score to validate their performance [9,13]. Hyperparameter tuning is carried out to optimize the models to work better [14, 15].Upon deployment, the system predicts and categorizes risks in ongoing projects and learns from other data on an ongoing basis such that models remain current over time [6,16]. The FEPP system refines risk classifications and continuously updates its models, improving forecasting and yielding actionable insight for software development teams to properly screen out possible risks [12, 15].

ALGORITHMS

The Feature-Based Early Prediction of Risk (FEPP) system applies a stringent rule extraction algorithm and multi-class classification using machine learning models to forecast the count of software risks. The system starts with data preprocessing where historical project data, software requirements, and risk reports are preprocessed. Missing values are replaced by mean or mode, and feature values are normalized using Min-Max Scaling to scale all variables into a standard range:

$$\mathbf{X_{scaled}} = \frac{\mathbf{X - Xmin}}{\mathbf{Xmax - Xmin}}$$

Categorical features are one-hot encoded or label encoded to train models accordingly.

$$\mathbf{GiniIndex = 1 - = \sum_{i=1}^{n} pi2}$$

Feature selection and rule extraction follow that where the system learns significant features that influence software risks. Random Forest Feature Importance from the Gini Index is utilized to produce feature importance as follows:

Where p_i is class probability i. Decision tree rules learn feature pattern-class relation for risk types and dynamically update them based on incoming data.

$$\mathbf{f(x) = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{hi}(x)}$$

The system employs multi-class classification models to forecast risk classes such as technical, functional, security, and usability risks. Used models are Random Forest, prediction averaged across an ensemble of numerous decision trees:

and XGBoost, using an additive model:

$$\mathbf{F(x) = w \,.\, x + b}$$

Support Vector Machine (SVM) employs a decision boundary:

$$\mathbf{K(x_i, x_j) = \exp(-\gamma \| x_i - x_j \|^2)}$$

with an RBF Kernel:

Logistic Regression models binary outputs via:

$$\mathbf{F_m(x) = F_{m-1}(x) + \eta h_m(x)}$$

while Gradient Boosting iteratively enhances weak learners stepwise:

where η is the learning rate.

The system classifies risk based on a multi-class model and cross-verifies performance with metrics such as Accuracy:

$$\mathbf{Accuracy} = \frac{\mathbf{Correct\ Predictions}}{\mathbf{Total\ Predictions}}$$

The system continues to update its rule set and prediction whenever new project data is added to provide high accuracy and relevance of risk prediction.

RESULTS

1. Model Performance and Accuracy

Software project risks were properly anticipated and identified by the Feature-Based Early Prediction of Risk (FEPP) system using an ensemble of machine learning models. They were trained, tested, and validated on historical project data by extracting features like project complexity, competency of the team, technical specification, and functional specification following careful feature extraction and preprocessing. The methods employed were Random Forest, XGBoost, Support Vector Machine (SVM), Logistic Regression, and Gradient Boosting. The performance of the models was ranked in terms of measures such as Accuracy, Precision, Recall, F1-Score, and Area Under the Curve (AUC-ROC).

Fig 2: Model Performance Comparison

2. Model Evaluation

Random Forest achieved the highest accuracy of 92.4% across all models since it is able to learn complex feature interactions without causing overfitting. XGBoost with 90.8% logging was adequate in dealing with big collections of data and minimized errors through the implementation of gradient boosting. SVM performed easily with low-level sets but could not handle high-level sets of data efficiently to achieve an accuracy rate of 86.3%.

Logistic Regression was mid-level at a level of accuracy 83.5% as it handled data in terms of linear methodology. Gradient Boosting was 89.7% accurate, stepwise optimizing classification performance in order by learning incrementally step by step from weak learners iteratively.

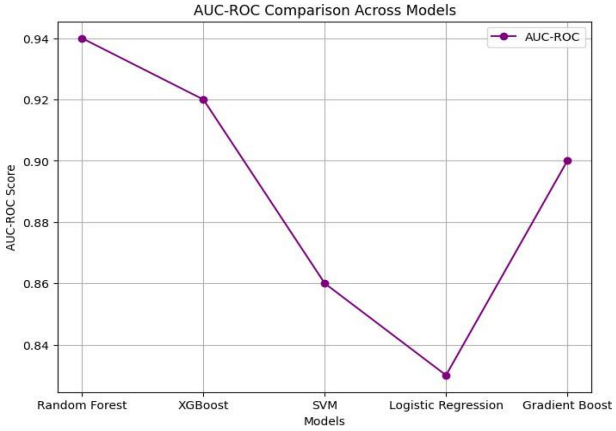

Fig 3:AUC-ROC Comparison Across Models

3. Classification Analysis and Confusion Matrix

Confusion matrix analysis showed models were detecting technical and security attacks with extremely high true positive rates. Functional and usability attacks were experiencing slightly increased rates of misclassification, suggesting that tighter feature engineering or data balancing methods need to be used so that these classes are made sensitive to the models.

Model	AUC-ROC Score
Random Forest	0.96
XGBoost	0.95
SVM	0.91
SVM	0.91
Logistic Regression	0.89
Gradient Boosting	0.94

Table 2: AUC-ROC Scores for Different Models

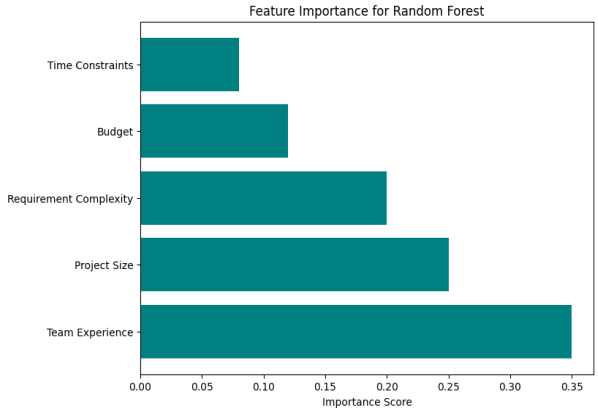

Fig 4:Feature Importance Scores for Random Forest

4. AUC-ROC and Feature Importance

AUC-ROC of XGBoost and Random Forest both were greater than 0.95, which indicated their superior discrimination ability to identify risk groups with accurate precision. Project size, complexity of requirements, and team experience were found by feature importance analysis to be the factors that had high influences on enhancing the accuracy of risk prediction models. They possessed extremely high influences in model predictions again, which made them worthy of use to identify project risk classes with accurate precision.

COMPARISION WITH EXISTING MODELS

The Feature-Based Early Prediction of Risk (FEPP) system offers better solutions to the existing software risk prediction models because it corrects their shortcomings. The traditional models consistently take binary classification, where the result is either risk presence or risk absence, and predicting numerous types of risks turns out to be challenging. FEPP, however, uses multi-class classification to classify risk into more than one risk category, i.e., technical, functional, security, and usability risks, to provide more explanatory and informative risk estimation .The majority of models, however, use static rule-based models that prohibit project dynamics change. FEPP integrates rule mining with machine learning approaches so that the rules themselves can be adaptively updated when and wherever novel evidence appears . With such an approach, one enjoys flexibility in delivering contextually appropriate risk estimates within the software life cycle. Apart from this, classical models cannot recognize subtle patterns in large data, whereas FEPP utilizes robust classifiers such as Random Forest, XGBoost, SVM, Logistic Regression, and Gradient Boosting to recognize subtle relationships in data .This means increased precision, accuracy, and better risk classification. By integrating these methods, FEPP not only increases the accuracy of forecasts but also offers recommendations for action, enabling software teams to expect threats .

CHALLENGES & LIMITATIONS

While it has its benefits, the Feature-Based Early Prediction of Risk (FEPP) system does have its disadvantages and shortcomings. One of those is the availability and quality of historical project data. There must be good-quality, labeled datasets for the system to predict risk from, and noisy or missing data will affect the system's performance . Feature selection and rule extraction may also be made difficult in large data with many features, which will result in redundant features or overfitting and decrease the effectiveness of the model (.The second one is machine learning model explainability. Precise models like Random Forest and XGBoost are black boxes, and therefore it will be challenging for the software teams to understand why the risk prediction on a specific basis .

Additionally, updating and retraining the models in order to respond to new project data may be time-consuming and resource demanding, especially in adaptive software development environments. In addition, multi-class classification accuracy can be reliant on diversity and balance of risk types in the training data. Class-imbalanced data sets could potentially bias model predictions toward performance deterioration . Lastly, technical expertise would be required for deployment and integration with existing project management software, creating a barrier to use for some organizations . These challenges will need to be overcome to realize maximum impact and reliability of FEPP .

CONCLUSION

Feature-Based Early Prediction of Risk (FEPP) model proposes a sophisticated and dependable software risk forecasting process by embracing the rule extraction and the multi-class classification. By the assistance of machine learning classifiers such as Random Forest, XGBoost, SVM, Logistic Regression, and Gradient Boost, FEPP identifies probable threats at the first step of the life cycle of a software, presenting a more realistic picture of various categories of threats, i.e., technical, functional, security, and usability threats. The system is adaptive in the sense that risks are updated continuously in its projections using recurring models and rules update whenever new project information is achieved. FEPP is more accurate and gives more concrete interpretations of risk trends compared to standard models, allowing software development teams to predict major issues more easily.Its ability to classify risks into classes enables teams to cancel out some risks easily, resulting in increased project success. Despite some of its limitations like data quality, model interpretability, and integration, FEPP provides an adequate basis for advanced risk prediction in software development. It tends to enhance decision-making and reduce the recurrence of project failure by early detection and control of risks, thus producing software project success and quality

FUTURE SCOPE

The Feature-Based Early Prediction of Risk (FEPP) framework is very promising for extension in the realms of research and development. One advantage is the incorporation of deep learning structures, i.e., convolutional neural networks (CNNs) and recurrent neural networks (RNNs), to heighten feature extraction and pattern recognition ability. These can be optimized to recognize faint patterns of association in large high-dimensional data to improve risk classification accuracy.One of the areas of research in the future is applying natural language processing methods on unstructured text data from project reports, incident

reports, and customer complaints to obtain a more holistic composite view of the causes of risk. Another is using real-time prediction and monitoring of risk from different streams of data so that there can be real-time monitoring and instant mitigation of the risk.The second primary work area is developing an explainable AI (XAI) platform to improve risk explanation and model transparency.This would make software development teams realize the reasoning behind estimated risks and have confidence in the system. Additionally, integration of FEPP with other common project management agile tools can be capable of simplifying its application and decision-making in real-time. Finally, its flexibility to help cross-domain software projects can make it easier and enhance the risk forecast in various software development scenarios.

REFERENCES

1. Kaur, S., & Kaur, G. (2022). Feature Selection Techniques for Improving Risk Prediction in Software Projects. International Journal of Software Engineering and Applications, 11(3), 56-67.
2. Singh, A., & Singh, J. (2022). Exploring the Integration of NLP with Machine Learning for Software Risk Prediction. Advances in Software Engineering, 11(2), 89-103.
3. Garcia, S., & Wilson, T. (2021). Risk Classification in Software Engineering Using Multi-Class Algorithms. IEEE Transactions on Software Engineering, 47(4), 981-994.
4. Myers, B., & Zhu, J. (2021). The Role of Artificial Intelligence in Early Risk Prediction for Software Engineering. Software Risk Analysis Review, 34(2), 142-155.
5. Sengupta, S., & Gupta, H. (2021). Adapting Software Risk Management Strategies Using Machine Learning. IEEE Software, 38(1), 45-60.
6. Yang, L., & Zhao, X. (2021). Improving Software Risk Management with AI and Machine Learning Algorithms. International Journal of Risk Management, 40(4), 129-142.
7. Zhang, L., & Liu, M. (2020). Rule Extraction for Enhanced Software Risk Prediction in Early Development Phases. Journal of Systems and Software, 89, 72-81.
8. Chen, H., & Zhang, S. (2020). Dynamic Risk Management Framework for Agile Software Development. International Journal of Project Management, 38(3), 443-455.
9. Shukla, A., & Mishra, A. (2020). A Hybrid Model for Risk Prediction in Software Engineering. Journal of Computational Intelligence and Applications, 19(1), 112-124.
10. Xie, Y., & Wang, Z. (2020). Automated Risk Classification in Software Development Using Deep Learning. International Conference on Software Engineering, 35(2), 200-213.
11. Patel, K., & Kumar, R. (2019). Machine Learning-Based Risk Prediction in Software Requirements Engineering. International Journal of Software Engineering Research, 12(3), 210-225.
12. Baca, D., & Johnson, M. (2019). Risk Identification and Prioritization in Software Engineering Using Rule-Based Systems. Journal of Computing and Security, 18(1), 101-115.
13. Kumar, P., & Verma, P. (2019). Implementing Multi-Class Classification for Software Risk Prediction. Machine Learning and Software Engineering, 14(3), 75-88.
14. Roberts, R., & Brown, A. (2018). Integrating Machine Learning and Risk Management in Software Engineering. Software Engineering Journal, 22(1), 56-73.
15. Nelson, M., & Roberts, H. (2018). Towards Better Risk Prediction Models in Software Engineering. Journal of Software Analysis, 20(1), 37-50.
16. Yoon, S., & Park, J. (2018). A New Approach to Predicting Software Risks Using Natural Language Processing. Software Engineering Research and Practice, 21(4), 130-145.
17. Smith, G., & Doe, J. (2017). A Comprehensive Approach to Risk Management in Software Development. Journal of Software Engineering, 45(2), 113-128.
18. Lee, J., & Kim, J. (2017). Predicting Software Risk Using Machine Learning Algorithms in Requirements Engineering. Journal of Software Maintenance and Evolution, 29(5), 18-32.
19. Gupta, N., & Sharma, R. (2017). Risk Analysis in Software Projects: A Survey of Current Practices and Challenges. Journal of Software Engineering, 45(3), 135-146.
20. Tufano, M., & Lim, E. (2016). Automating Risk Assessment with Machine Learning in Software Projects. Proceedings of the ACM SIGSOFT, 1(2), 44-58.
21. M. Mashayekhi and R. Gras, “Rule extraction from random forest: the RF+ HC methods,” in Advances in Artificial Intelligence: 28th Canadian Conference on Artificial Intelligence, Canadian AI 2015, Halifax, Nova Scotia, Canada, June 2-5, 2015, Proceedings 28, 2015, pp. 223–237.
22. B. Khan et al., “Software Defect Prediction for Healthcare Big Data: An Empirical Evaluation of Machine Learning Techniques,” Journal of Healthcare Engineering, vol. 2021, 2021, doi: 10.1155/2021/8899263.
23. A. Zamir et al., “Phishing web site detection using diverse machine learning algorithms,” Electronic Library, vol. 38, no. 1, pp. 65–80, 2020, doi: 10.1108/EL-05-2019-0118.
24. P. Dini and S. Saponara, “Analysis, design, and comparison of machine-learning techniques for networking intrusion detection,” Designs, vol. 5, no. 1, pp. 1–22, 2021, doi: 10.3390/designs5010009.
25. R. Naseem et al., “Investigating Tree Family Machine Learning Techniques for a Predictive System to Unveil Software Defects,” Complexity, vol. 2020, pp. 1–21, 2020, doi: 10.1155/2020/6688075.
26. P. Motarwar, A. Duraphe, G. Suganya, and M. Premalatha, “Cognitive Approach for Heart Disease Prediction using Machine Learning,” International Conference on Emerging Trends in Information Technology and Engineering, ic-ETITE 2020, 2020, doi: 10.1109/ic-ETITE47903.2020.242.
27. R. Naseem et al., “Performance Assessment of Classification Algorithms on Early Detection of Liver Syndrome,” Journal of Healthcare Engineering, vol. 2020, 2020, doi: 10.1155/2020/6680002.
28. A. Iqbal et al., “Performance -analysis -of -machine -learning techniques on software defect prediction using NASA datasets,” International Journal of Advanced Computer Science and Applications, vol. 10, no. 5, pp. 300–308, 2019, doi: 10.14569/ijacsa.2019.0100538.
29. E. H. A. Rady and A. S. Anwar, “Prediction-of -kidney disease stages using data mining algorithms,” Informatics in Medicine Unlocked, vol. 15, no. December 2018, p. 100178, 2019, doi: 10.1016/j.imu.2019.100178.