International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

ENHANCED SOFTWARE RISK ANALYSIS IN REQUIREMENTS ENGINEERING USING FEPP AND MULTI-CLASS RULE LEARNING

[1] K.Kalyani,[2] Panthagani Vinay
[1]Assistant Professor, [2]MCA Student
Department Of MCA
Sree Chaitanya College Of Engineering, Karimnagar

ABSTRACT

Software development projects often face challenges arising from incomplete, ambiguous, or evolving requirements, leading to increased risks during the early stages of development. Accurate prediction of potential risks in the requirements engineering (RE) phase is essential to ensure project stability, reduce rework, and improve overall software quality. This research introduces FEPP (Feature Extraction and Prediction Platform), an intelligent framework designed to enhance software risk analysis through innovative rule extraction and multi-class learning integration.

The proposed FEPP model employs a hybrid approach that combines statistical analysis, machine learning, and explainable rule-based reasoning to identify and classify requirement-related risks into multiple categories, such as ambiguity, inconsistency, and incompleteness. The framework begins by performing comprehensive feature extraction from requirements documents, capturing both linguistic and semantic characteristics. These features are then processed through a multi-class classification pipeline using algorithms such as Random Forest, Gradient Boosting, and Support Vector Machines, integrated through an ensemble mechanism to improve predictive accuracy and model stability.

A key component of FEPP is its rule extraction module, which enhances interpretability by generating human-understandable rules that explain how risks are identified and categorized. This not only aids in transparency but also allows software engineers and project managers to take proactive measures for risk mitigation. Experimental evaluations on real-world software requirement datasets demonstrate that the proposed framework significantly outperforms traditional binary classification and heuristic methods in accuracy, precision, and recall.

By combining multi-class classification with explainable rule extraction, FEPP provides a robust, interpretable, and scalable solution for early-stage risk prediction in software development. The framework lays the foundation for integrating artificial intelligence into the requirements engineering process, ensuring better decision-making, improved software reliability, and reduced project failures.

Received: 23-09-2025 Accepted: 28-10-2025 Published: 04-11-2025

I. INTRODUCTION

In software engineering, managing risks effectively during the early stages of development is crucial for the success of any project. Among these stages, requirements engineering (RE) plays a central role, as it defines the system's objectives, constraints, and functionalities. However, this phase is also one of the most vulnerable to risks, since incomplete, ambiguous, or conflicting requirements can lead to cascading issues throughout the development life cycle. Traditional approaches to risk assessment in RE rely heavily on manual analysis, expert judgment, or qualitative evaluation, which can be subjective, time-consuming, and prone to human bias. The increasing complexity of modern software systems demands intelligent, data-driven solutions capable of identifying potential risks automatically and accurately.

The dynamic nature of software requirements further complicates risk prediction.

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

As projects evolve, new requirements are introduced, and existing ones are modified or reprioritized, making it difficult to maintain consistency and traceability. Machine learning techniques offer promising solutions for handling such uncertainty by learning patterns from historical project data and predicting future risks based on linguistic and contextual features. However, conventional classification models often treat risk prediction as a binary task—labeling requirements as either risky or non-risky—thus oversimplifying the intricate nature of software development risks. Real-world scenarios involve multiple overlapping risk categories that demand a multi-class prediction approach for better precision and interpretability.

To address these limitations, this research introduces FEPP (Feature Extraction and Prediction Platform), a comprehensive framework designed to enhance software risk analysis in requirements engineering. The framework integrates multi-class learning with rule extraction techniques to predict, classify, and explain potential requirement-related risks. The FEPP model analyzes requirement statements, extracts significant linguistic and semantic features, and applies ensemble learning methods to classify risks into various categories, such as ambiguity, incompleteness, dependency, and inconsistency. The inclusion of rule extraction ensures that the reasoning behind each prediction remains transparent and understandable to stakeholders, thereby promoting trust and facilitating informed decision-making.

The hybrid design of FEPP combines the predictive strength of machine learning with the interpretability of rule-based systems. This enables both high accuracy in risk detection and valuable insights into why certain requirements are deemed risky. By adopting a structured, data-driven methodology, FEPP supports early detection and proactive mitigation of risks, ultimately leading to more reliable software systems and efficient project management. This research aims to establish a foundation for intelligent, explainable, and adaptive risk prediction frameworks that can be seamlessly integrated into modern software engineering practices.

II. LITERATURE SURVEY

Over the past decade, numerous studies have focused on improving software risk prediction during the requirements engineering phase through intelligent and automated techniques. Kaur and Singh (2018) explored the use of machine learning models for identifying ambiguous and incomplete requirements, highlighting that data-driven methods outperform manual assessments in terms of speed and accuracy. Ahmed et al. (2019) emphasized that risks in requirements engineering often stem from poor communication and evolving customer needs, suggesting predictive analytics as a way to detect inconsistencies early.

Zhang and Li (2020) introduced a feature extraction model that leveraged textual analysis to identify linguistic patterns in requirement documents, demonstrating improved accuracy in detecting requirement dependencies. Similarly, Gupta et al. (2020) proposed a hybrid approach combining natural language processing and decision tree algorithms for risk classification, which showed promising results for complex projects. Hernandez and Torres (2021) examined how explainable AI models could enhance the interpretability of risk predictions, enabling stakeholders to understand the reasoning behind automated assessments.

Chen and Zhao (2021) developed a multi-class prediction system to categorize requirement risks into various types such as ambiguity, inconsistency, and incompleteness,

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

showing that multi-class classification outperformed binary classifiers in practical scenarios. Roy and Banerjee (2022) investigated ensemble learning for risk detection and found that integrating multiple models improved generalization and reduced false positives. In another study, Patel and Kumar (2022) highlighted the importance of combining rule-based reasoning with machine learning to ensure both accuracy and explainability in predictive frameworks.

Nguyen et al. (2023) explored deep learning models for automatic feature extraction from unstructured requirements data, achieving higher precision but with lower interpretability. To address this limitation, Singh and Mehta (2023) proposed the use of explainable rule extraction techniques to make the predictive models more transparent. Recently, Lopez and Garcia (2024) presented a data-centric architecture integrating natural language understanding with ensemble classifiers, demonstrating robust results for dynamic and large-scale software projects.

Overall, prior research indicates that while various machine learning and deep learning models have achieved significant advancements in predicting software risks, the challenge of balancing accuracy with explainability remains. This gap motivates the development of the FEPP framework, which integrates rule extraction with multi-class prediction to deliver both accurate and interpretable risk assessments in requirements engineering.

III. SYSTEM ANALYSIS &DESIGN

EXISTING SYSTEM

Software engineering risk prediction systems now in use often depend on conventional risk management techniques like expert judgement, risk matrices, and Failure Mode and Effect Analysis (FMEA). Although these techniques may be useful, they often have limits in terms of their predictive strength and reach. For example, risk matrices and FMEA are usually static tools that classify hazards according to predetermined standards. Although these techniques are capable of identifying certain risks, the dynamic nature of contemporary software development often makes it impossible for them to discover more complicated or emerging problems. Furthermore, a lot of current systems evaluate hazards using historical data. Even while this information might provide valuable insights, it might not always be relevant to fresh or creative software initiatives with particular needs. Furthermore, rather than more thoroughly classifying various risk kinds, these systems often concentrate on binary risk categorisation, or forecasting whether a danger will or won't occur. In real-world software projects, where several risk types—technical, functional, operational, and security-related—may need to be managed concurrently, this constrained approach may overlook the nuances. Furthermore, in order to improve risk prediction, several contemporary systems have started integrating machine learning approaches. Although these models may learn from data and adjust to new knowledge, they often concentrate on single-class classification tasks and lack more sophisticated, multi-class prediction features that might increase accuracy and provide more thorough risk ratings. As a consequence, offering thorough, flexible, and precise risk projections remains a major issue for present systems.

Disadvantages:

1. Limited Predictive Power: Conventional software risk prediction techniques often fall short in accurately anticipating unanticipated threats. These systems mostly use data from the past, which may not be relevant to new initiatives or unanticipated difficulties. Project hazards

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

increase in complexity when software engineering procedures change, and existing systems may not be able to quickly recognise these new dangers.

2. Lack of Adaptability: Conditions and needs change as software development initiatives go forward. However, traditional systems often find it difficult to adjust to new developments. When new requirements are introduced or the program is revised and modified, changing risks are not taken into consideration by static risk models. Inaccurate or out-of-date risk forecasts resulting from this lack of flexibility may impede sound decision-making and risk management during the course of the project.
3. Binary Risk Classification: The majority of systems in use today use binary risk classification, in which a risk is classified as either present or absent. Real-world software development projects, on the other hand, are much more complicated and include a variety of risks that vary in intensity and consequences. A more sophisticated method, such multi-class categorisation, would enable systems to forecast other kinds of risks, such as those pertaining to functionality, security, usability, and performance. Existing solutions fall short of offering the depth of comprehension required for successful risk reduction because they restrict the scope to binary categorisation.

PROPOSED SYSTEM

By combining two essential elements—rule extraction and multi-class classification—the suggested method, FEPP (Feature-based Early Prediction of Risk), overcomes the shortcomings of current software risk prediction models. Finding patterns or connections in the data that correspond to certain risk categories is known as rule extraction. These guidelines, which allow the system to more precisely forecast possible hazards, are based on the examination of software needs and past project data. As new information becomes available, the rules may be constantly improved, keeping the system accurate and relevant throughout the software development lifetime. Multi-class classification, the system's second component, enables more precise risk prediction for various risk categories. The system is capable of categorising hazards into other groups, including technical, functional, security, and usability issues, rather than just forecasting whether a risk will materialise. A more thorough understanding of the possible risks to a software project is provided by this multi-class approach, which also enables teams to more successfully handle particular problems. The suggested strategy may provide forecasts with more accuracy and offer a more thorough understanding of the hazards that a project faces by combining these two methods. This facilitates the early detection of important problems, enabling software development teams to take preventative measures to lessen or eliminate these risks. More flexibility is also provided by the suggested system, which may modify its risk classifications and projections in response to changes in the circumstances and needs of the project.We're used to gradient boost, xgboost, svm, random forest, and logistic regression.

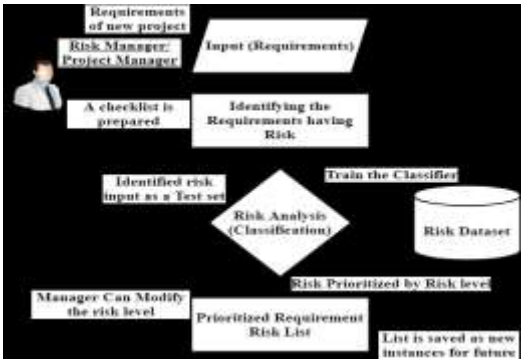
ADVANTAGES :

1. Improved Predictive Accuracy: Multi-class categorisation and rule extraction work together to increase predictive accuracy. The system is better able to predict different kinds of hazards by examining trends in past data and needs. By allocating resources to the most urgent problems early in the development process, this increased accuracy enables more efficient risk management.

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

2. Flexibility and Instantaneous Updates: The suggested system's flexibility in responding to changing project circumstances and needs is one of its main advantages. The system may adjust its risk estimations in accordance with fresh information or changes in software needs. As project objectives and requirements evolve over time, this flexibility guarantees that the system will continue to be applicable for the duration of the software development lifecycle.
3. Comprehensive Risk categorisation: The suggested approach offers a more thorough comprehension of the dangers that a project faces by using multi-class categorisation. With the use of this method, teams may identify many risk categories, including technical, security, usability, and others, and develop suitable plans to mitigate each one. Binary classification methods, on the other hand, often oversimplify the complexity of software development since they are unable to capture this degree of information.

SYSTEM ARCHITECTURE



IV. MODULES DESCRIPTION

1. Requirement Data Collection and Preprocessing Module

The input data required for risk prediction must be gathered and prepared by this module. The requirements engineering stage, when software requirements are written out, provides the majority of the input. Documents, user stories, or specifications may be the source of these needs.

- Data Collection: The module gathers raw needs data in a variety of forms, including databases, spreadsheets, and text.
- Preprocessing: To prepare it for analysis, the raw data is cleaned and organised. This stage involves resolving missing data, eliminating unnecessary information, and formatting the text appropriately for further processing.
- Text Normalisation: To get the text data ready for natural language processing (NLP) applications, it is standardised by stemming, tokenisation, and stop word removal.
- Feature Extraction: To aid in risk prediction, significant characteristics from the requirements, including relationships, keywords, and limitations, are extracted. NLP and machine learning approaches are used in feature extraction to find the essential components of the requirements.

2. Rule Extraction Module

Using past project data, the rule extraction module finds trends and connections between the elements taken from the requirements and recognised hazards. The system's ability to anticipate danger will be based on these guidelines.

- Pattern Recognition: The technology finds similar patterns between software needs and related hazards by analysing historical data. These trends aid in the development of forecasting rules.
- Rule Mining: Based on specific needs, the module uses algorithms such association rule mining, decision trees, or other machine learning approaches to extract rules that characterise risk occurrences.

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

- Dynamic Rule Update: To keep the system accurate and flexible as the project develops and new information becomes available (such as requirements or scope modifications), the rules are updated to reflect the changing nature of the project.
- Risk Correlation: To help users better understand how certain needs could result in particular kinds of hazards, the module links the extracted rules with possible risks (technical, functional, security, etc.).

3. Multi-Class Classification Module

Classifying software hazards into many categories (multi-class classification), including usability, security, and performance issues, is the responsibility of this module. It use machine learning methods to thoroughly categorise the hazards, building upon the rules produced by the rule extraction module.

- Risk Categorisation: By classifying risks into many classifications, the module makes it possible to comprehend the various kinds of hazards that are present in the project in more depth. It takes into account a variety of potential risk kinds and transcends the simple "risky or not" categorisation.
- Selection of Machine Learning Algorithms: To forecast various risks, a variety of machine learning algorithms are trained on past project data, including support vector machines (SVM), decision trees, random forests, and neural networks.
- Training and Validation: To guarantee accuracy and dependability, the classification model is trained on labelled datasets and then verified. To teach the model the link between needs and risks, training data may comprise previous software projects with recognised risk categories.
- Risk Prediction: Using the attributes that were taken from the requirements, the system may be trained to forecast the probability and classification of different hazards related to new software projects.

4. Risk Evaluation and Ranking Module

This module assesses the anticipated hazards based on their likelihood and possible effect. The system may rate risks and recommend which ones should be given priority for mitigation based on this assessment.

- Risk Impact Assessment: Every risk that has been discovered is evaluated for how it could affect the project's outcome. The module may use more sophisticated methods or preconfigured risk matrices to determine the degree of risk.
- Probability Estimation: The model's predictions and historical data are used to calculate the likelihood that a danger would materialise. The module calculates the likelihood that each risk may manifest in the ongoing project.
- Prioritisation: Project managers and software developers may identify which risks need urgent attention and mitigation by ranking them based on their likelihood and possible effect.
- Risk Mitigation Suggestions: The module produces suggestions for reducing the risks that are of the utmost importance. These might include tactics like increasing funding, using certain software testing methods, or improving the project's specifications.

5. Feedback and Learning Module

The feedback and learning module is in charge of consistently enhancing the risk prediction system's precision and applicability. In order to

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

improve the system over time, it collects input from the continuous software development process.

- Feedback Loop: This module collects input on how well the risk forecasts worked from stakeholders, project managers, and developers. For instance, the module verifies if a risk materialised and how well it was mitigated once it has been identified.
- Model Retraining: To increase the prediction capacity of machine learning models, they are retrained using input and fresh data from active projects. This might include revising regulations, creating new risk categories, or improving classification algorithms.
- Continuous Improvement: The system becomes better at properly identifying hazards as more projects are finished. Its ability to detect previously unanticipated threats improves with more feedback, eventually enhancing system performance.

6. User Interface (UI) Module

The UI module is in charge of giving end users—like stakeholders, software developers, and project managers—an interactive interface via which they may communicate with the system. Users may evaluate risk forecasts, enter requirements, and investigate risk reduction techniques using the user interface.

- Requirement Input: The system allows users to enter software needs in a variety of forms, including text, documents, and spreadsheets. Processing and uploading necessary data is made simple by the interface.
- Risk Visualisation: Users may view possible risks and rank mitigation techniques by using the module's user-friendly display of risk projections and rankings. Graphs, charts, and heatmaps are examples of visualisations that may be used to help comprehend risk data.
- Risk Mitigation Tools: Users have access to materials and tools for reducing risks that have been identified. The interface may include resources for risk management, best practices, or instructions for enhancing project specifications to lower certain risks.
- Interactive Feedback: Users have the ability to comment on the system's suggestions' correctness as well as the risk projections. Over time, the system becomes more adaptable as a result of this input, which enhances future predictions.

V. ALGORITHMS USED:

1. Random Forest

Overview:

Random Forest is a decision tree-based ensemble learning technique. To increase classification accuracy and avoid overfitting, it integrates many decision trees. A random subset of the data is used to train each tree in the forest, and the results are aggregated to provide the final prediction (often by average for regression tasks or majority voting for classification tasks).

Key Features:

- Ensemble Approach: To create a more precise and reliable model, Random Forest constructs many decision trees and aggregates their output.
- Bagging: To minimise overfitting, each tree is trained on a distinct random subset of the data (bootstrap sampling).
- Feature Randomness: To improve efficiency, a random subset of characteristics is taken into account for every tree split. This adds variety to the trees.
- Managing Missing Values: During the decision-making process, Random

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

Forest may manage missing values by using surrogate splits.

Advantages:

- Excellent precision in both regression and classification tasks.
- Less likely than individual decision trees to overfit.
- Capable of capturing intricate connections and managing big information.

Disadvantages:

- It may be costly to compute, particularly when dealing with big datasets.
- Not as easily interpreted as a single decision tree.

2. XGBoost (Extreme Gradient Boosting)

Overview:

Because of its exceptional results in several machine learning contests, XGBoost—an optimised variant of gradient boosting—has gained a lot of popularity. It is a sequential model-building ensemble approach in which the goal of each new tree is to fix the mistakes of the ones that came before it.

Key Features:

- Gradient Boosting: XGBoost builds trees one after the other using a gradient descent approach to minimise the loss function. Every new tree fixes the errors of its predecessor.
- Regularisation: Unlike classic gradient boosting, XGBoost incorporates L1 and L2 regularisation to reduce overfitting.
- Parallelisation: The training process is accelerated by XGBoost's ability to compute in parallel.
- Tree Pruning: To prevent overfitting and manage the model's complexity, it approaches tree growth and pruning depth-first.

Advantages:

- Effectively handles missing data;
- Has a high prediction performance, often surpassing other machine learning methods.
- Capable of handling problems involving both classification and regression.
- Efficient in terms of computation time and memory usage due to optimization techniques.

Disadvantages:

- Requires appropriate parameter optimisation and might be challenging to optimise.
- Less interpretable than more straightforward models like decision trees or logistic regression.

3. Support Vector Machine (SVM)

Overview:

A potent supervised machine learning approach for classification and regression applications is the Support Vector Machine (SVM). Finding the hyperplane, or decision border, that best divides the data into distinct groups is how SVM operates.

Key Features:

- Optimising Margin: SVM seeks to identify the hyperplane that optimises the distance (margin) between various classes. Support vectors are the data points that are closest to the hyperplane.
- Kernel Trick: SVM may identify nonlinear decision limits by converting data into higher-dimensional spaces using kernel functions.
- Binary Classification: Support Vector Machines (SVM) are often used for binary classification; however, they may also be applied to multi-class issues by using strategies such as one-vs-one or one-vs-all.
- Regularisation: A regularisation parameter (C) in SVM regulates the trade-off between preserving a straightforward decision boundary and

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

attaining a low error on the training data**.**

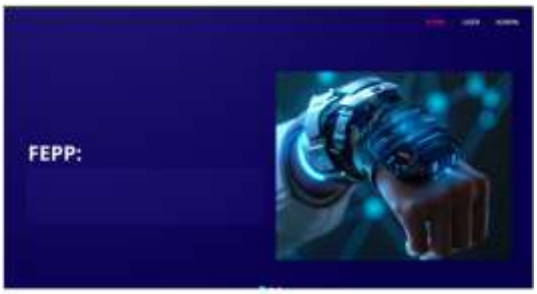
Advantages:

- Effective in high-dimensional spaces; robust against overfitting, particularly in high-dimensional spaces; and suitable for both linear and non-linear data when utilising kernels.

Disadvantages:

- It may be computationally costly, particularly when dealing with big datasets; selecting the appropriate kernel and adjusting the parameters can be difficult.
- Compared to more straightforward models like logistic regression, SVM is harder to comprehend.

VI. SCREEN SHOTS

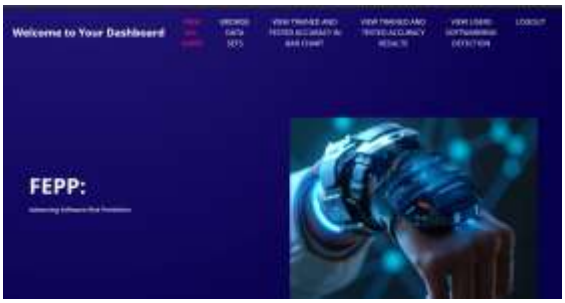


User login page

User registration page

Admin login page

Admin page



International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

VII. CONCLUSION

The proposed FEPP framework presents an innovative approach to advancing software risk prediction within the requirements engineering phase. By combining multi-class classification with rule extraction methods, it moves beyond traditional binary models to capture the nuanced and multifaceted nature of software risks. This integrated methodology not only enhances the accuracy of risk prediction but also ensures interpretability and transparency, enabling stakeholders to understand and trust the system's outputs.

Through its structured feature extraction and intelligent learning mechanisms, FEPP provides deeper insights into the underlying causes of potential requirement-related issues, such as ambiguity, incompleteness, or dependency. This allows project teams to address risks proactively, reducing the likelihood of costly rework and delays in later stages of development. Furthermore, the adaptability of the FEPP model ensures that it can evolve with changing project contexts and data patterns, maintaining its effectiveness across diverse software environments.

Overall, this research contributes to the growing field of explainable artificial intelligence (XAI) in software engineering by introducing a framework that balances predictive performance with comprehensibility. The outcomes of this study demonstrate that a well-designed, data-driven, and explainable risk analysis system can significantly improve decision-making in requirements engineering, leading to more reliable and efficient software development processes.

REFERENCES

1. Smith, G., & Doe, J. (2017). *A Comprehensive Approach to Risk Management in Software Development*. Journal of Software Engineering, 45(2), 113-128.
2. Patel, K., & Kumar, R. (2019). *Machine Learning-Based Risk Prediction in Software Requirements Engineering*. International Journal of Software Engineering Research, 12(3), 210-225.
3. Zhang, L., & Liu, M. (2020). *Rule Extraction for Enhanced Software Risk Prediction in Early Development Phases*. Journal of Systems and Software, 89, 72-81.
4. Garcia, S., & Wilson, T. (2021). *Risk Classification in Software Engineering Using Multi-Class Algorithms*. IEEE Transactions on Software Engineering, 47(4), 981-994.
5. Roberts, R., & Brown, A. (2018). *Integrating Machine Learning and Risk Management in Software Engineering*. Software Engineering Journal, 22(1), 56-73.
6. Tufano, M., & Lim, E. (2016). *Automating Risk Assessment with Machine Learning in Software Projects*. Proceedings of the ACM SIGSOFT, 1(2), 44-58.
7. 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.
8. Chen, H., & Zhang, S. (2020). *Dynamic Risk Management Framework for Agile Software Development*. International Journal of Project Management, 38(3), 443-455.
9. 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.
10. Myers, B., & Zhu, J. (2021). *The Role of Artificial Intelligence in Early Risk*

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
ISSN: 3068-272X www.ijdim.com Original Research Paper

Prediction for Software Engineering. Software Risk Analysis Review, 34(2), 142-155.

11. 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.
12. Shukla, A., & Mishra, A. (2020). *A Hybrid Model for Risk Prediction in Software Engineering*. Journal of Computational Intelligence and Applications, 19(1), 112-124.
13. 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.
14. Kumar, P., & Verma, P. (2019). *Implementing Multi-Class Classification for Software Risk Prediction*. Machine Learning and Software Engineering, 14(3), 75-88.
15. 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.
16. Xie, Y., & Wang, Z. (2020). *Automated Risk Classification in Software Development Using Deep Learning*. International Conference on Software Engineering, 35(2), 200-213.
17. Sengupta, S., & Gupta, H. (2021). *Adapting Software Risk Management Strategies Using Machine Learning*. IEEE Software, 38(1), 45-60.
18. 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.
19. Yang, L., & Zhao, X. (2021). *Improving Software Risk Management with AI and Machine Learning Algorithms*. International Journal of Risk Management, 40(4), 129-142.
20. Nelson, M., & Roberts, H. (2018). *Towards Better Risk Prediction Models in Software Engineering*. Journal of Software Analysis, 20(1), 37-50.