Journal of Information Systems Engineering and Management
2025, 10(25s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

A Fuzzy-Based XGBoost Approach for Classification and Prioritization of Cost Overhead Factors in Agile Software Development

Jitesh R. Neve[1], Dr. Sohit Agarwal[1]
[1]*Dept of Computer Engineering and IT, Suresh Gyn Vihar University, Jaipur, India*

ARTICLE INFO

Received: 26 Dec 2024
Revised: 12 Feb 2025
Accepted: 22 Feb 2025

ABSTRACT

As a result of these concerns with regard to cost overhead reduction, this paper examines the problematics of exact classification and prioritizing in Agile software development. Although using Agile iterative approach may add the levels of needed complexity in making smart decisions to control cost and resources. The work proposes an approach integrating fuzzy logic with XGboost to address these difficulties. Thus, while XGBoost bases categorisation and prioritisation of cost overhead variables on its gradient-boosting method, proven to reliably classify and rank factors, fuzzy logic is employed to respond to uncertain conditions characteristic of Agile environments. Compared to other approaches, the suggested hybrid model shows significant improvement on both counts. Greater accuracy in categorisation ensures more uniformity in the identification of cost overhead elements, which proves useful for decisions in Agile projects. Proper prioritisation also enables planers in the team to reduce time wastage and allocate available resources in the best ways possible. The criteria which serve to evaluate paves for the strength of the model. As revealed with better accuracy in training, handling complex data, the hybrid model does have higher accuracy than Decision Tree, Random Forest and SVC. That is why the optimisation of feature selection in XGBoost and the ability to handle uncertainty in fuzzy logic increase the accuracy of the latter. Through recall, the process is much boosted and as such ensures all elements are noted. Amongst the measures, the pure handicapped loss less exact assignment of the F1 score assesses that carries a proposing of both recollect and precision that establishes the general organization of the model through studying the Agile cost overhead control method.

Keywords: agile cost overhead control, agile software development, cost prioritization, fuzzy logic, xgboost.

I. INTRODUCTION

Because agile software development can effectively produce high-quality software while allowing quick modifications, it has become the most often used approach in modern software engineering [1][8]. Agile development is often difficult, especially with regard to cost overheads, even with its advantages [14],[7]. Maintaining project sustainability and guarantee of effective resource allocation depend on giving these cost overhead elements top priority. The importance of the research topic is discussed, the prioritising of cost overheads in agile development is given an overview in this study, and the integration of machine learning for performance improvement is investigated. In agile projects, cost overheads from technological debt, poor communications, insufficient resource management, and frequent scope changes may come from a number of sources [10]. These elements may greatly influence project costs and schedules, therefore influencing the final quality. Although agile techniques are meant to be flexible, inefficiencies often result from the absence of methodical means for spotting and ranking cost-related issues [11]. Dealing with these inefficiencies calls for a thorough awareness of the many elements involved and their respective weight. Since it closes the gap between cost control techniques and agile development methods, the research done in this study is important. The research intends to improve decision-making procedures and allow teams to actively handle high-impact problems by concentrating on the categorisation and prioritising of cost overhead elements. Particularly in the context of prioritising, the incorporation of sophisticated machine learning approaches provides a strong foundation for obtaining actionable insights, hence matching agile's iterative and value-driven approach [12].

Automating difficult decision-making processes and finding trends in data that may not be immediately obvious to human analysts offers enormous promise for machine learning [15]. Within the framework of agile development, machine learning may be used to examine past project data, project cost overrun prediction, and cost component

prioritising based on influence [17]. This work suggests to reach these goals using a fuzzy-based XGBoost technique. Combining XGBoost with fuzzy logic offers a strong means of managing uncertainty and ambiguity inherent in agile settings [13]. While XGBoost's gradient-boosting system guarantees accurate and scalable predictions, fuzzy logic helps to describe imprecise aspects, like team dynamics and stakeholder communication. These methods used together allow the efficient grouping of cost overhead aspects and their prioritising depending on established criteria.

Several important research questions are answered in this work: the identification and classification of main cost overhead factors in agile software development, the efficacy of a fuzzy-based XGBoost approach in prioritising these factors, and the general effect of such prioritising on the efficiency and performance of agile projects. The paper proposes that compared to standard classification methods, a fuzzy-based XGBoost methodology will show better accuracy in identifying cost overhead variables. Furthermore, it suggests that by giving cost overhead elements top priority, project results and resource allocation would be much improved.

The paper's framework starts with an introduction that summarises the difficulties in agile development and the requirement of giving cost top priority overheads top importance. It is followed by an analysis of the body of current research on cost control in agile projects and machine learning application. The fuzzy-based XGBoost strategy is described in the section on methodology along with preprocessing, data collecting, and model implementation. The results and discussion part offers the conclusions and assesses the performance of the suggested strategy. At last, the section on the conclusion and future work summarises the contributions of the study and suggests possible routes for further investigation. In order to make a contribution to the development of agile approaches, the objective of this study is to examine these components in order to do this. As a consequence of this, it will provide useful insights that can be used to efficiently control cost overheads and improve the performance of the project simultaneously.

II. BACKGROUND AND RELATED WORK

The authors in [2] suggest a few desired features of priority vectors, including consistency, coherency, and intensity, and then establish sufficient conditions for the existence of priority vectors that possess those properties. In general, the Eigenvector Method and the Geometric Mean Method, which are the two most prominent approaches for deriving the priority vector, do not always give priority vectors that also possess these desired features. In this section, they provide a novel method for solving the issue of deriving the priority vector based on a particular optimisation problem that satisfies the required qualities under the right assumptions. The method is shown with two instances, which are provided and discussed in this article.

The authors in [18] suggest a DRL-based stock trading system using the CLSTM-PPO Model to uncover hidden information in daily stock data. Our model uses a cascaded structure with two stages of deep LSTM networks. The first stage extracts time-series features from daily stock data, which is then fed to the agent for training in the reinforcement learning algorithm. The actor and critic also use LSTM networks. Researchers test stock market datasets from four main indices: the Dow Jones Industrial Index (DJI) in the US, the Shanghai Stock Exchange 50 (SSE50) in China, the S&P BSE Sensex Index (SENSEX) in India, and the Financial Times Stock Exchange 100 (FTSE100) in the UK. They evaluate their model against several benchmarks.

This research [3] proposes combining expert and algorithmic judgements to reduce bias and enhance decision-making flexibility. This study introduces the novel ordered pair of normalised real numbers (OPNs) to the AHP approach and proposes the fuzzy analytic hierarchy process (OFAHP). THE OFAHP combines expert judgements with machine learning algorithms to make choices using OPNs. The suggested strategy yields good judgement outcomes in experiments on actual data sets. Additionally, machine learning algorithms may rectify incorrect or inaccurate expert judgements for appropriate decision-making outcomes.

The authors in [6] propose a three-stage framework for strategic marketing planning, utilising AI benefits: mechanical AI for automation, thinking AI for data processing, and feeling AI for analysing human emotions and interactions. This framework describes how AI may be used in marketing research, strategy (STP), and activities. Mechanical AI may be utilised for data collecting, thinking AI for market analysis, and feeling AI for consumer comprehension in marketing research. In marketing strategy (STP), mechanical AI may be utilised for segmentation, thinking AI for targeting, and feeling AI for positioning (segment resonance). During the marketing action stage, mechanical AI can standardise, thinking AI can personalise, and feeling AI can build relationships. The framework is used to numerous marketing domains, organised by 4Ps/4Cs, to demonstrate the strategic usage of AI.

This researchin [3] aims to examine how team characteristics interact in large-scale software development, with an emphasis on stable and dynamic teaming techniques. The research included a literature review, semi-structured interviews with 19 engineers from two businesses, and validation workshops with two companies from unrelated industries. The research indicates that industry seldom addresses the subject of stable vs dynamic methods to software engineering team formation, with stable teams being the default choice. Both stable and dynamic teams have strengths and disadvantages, requiring careful assessment of the best strategy for each circumstance. This research proposes a model to examine the relationship between team stability, completeness, and smallness. Practitioners find this model relevant, accurate, generalisable, and valuable.

The major goal of this work [9] is to explore the development of hybrid software development techniques and underline the main challenges with information systems (IS) audits. Although modern technology companies are under continual pressure to provide software quicker due to growing requirements globally, this ongoing endeavour results in supposedly motivated by practice novel development methodologies. Modern software development is neither pure linear stages progression nor agile, so choosing the suitable mix of methodologies that help to meet objectives and guarantee value generation for companies becomes difficult.

Agile techniques were established in [5] to reduce issues with conventional software development processes. There are various Agile techniques used in software development, such as Scrum, Extreme Programming, and Kanban. An Agile methodology emphasises customer-developer cooperation and self-organization. Project teams employ various Agile methods to accomplish this. Some teams utilise one practice, some use many. This paper analyses data from Agile software development teams and finds that adopting more practices improves team communication, project requirements, and priorities, leading to better project outcomes.

Here in [4] a methodical mapping research is established to analyse the presence of many empirical views of productivity throughout the several business sectors and knowledge fields covered by the industrial practice of SE, thereby pointing out their similarities. This paper uses DBLP and Scopus search engines to find bibliographic references on software productivity ranging from 1987 to 2021. The analyses eliminate references that do not line up with full not-later-subsumed papers published in peer-reviewed journals and meetings. These studies contain only publications reflecting industry practitioner opinions or empirical investigations based on software industry data. 99 articles in all are examined. Particularly with relation to the effects of agile development methods on software production, the mapping revealed substantial variation in research results. The methodical mapping also included methodological suggestions to assist sector professionals in tackling this topic and advancing further studies.

III. RESEARCH METHODOLOGY

The study's approach is based on the methodical structure developed for managing the Agile project cost control's cardinality, The process begins with the formulation of the topic of the study, which focuses on the difficulties in precise categorization and priority setting of the cost overhead elements inherent in the flexible and highly unpredictable nature of the Agile environment. This stage involves a close review of the body of current literature to assess the drawbacks of traditional approaches and identify the gaps which are targeted with the proposed hybrid model. Data collecting comes next, with focus on real Agile processes to ensure the data collected would be useful and practical for Agile programmes. Data from project management files, industry case studies and polls are used.

Preprocessing makes any alteration on this unprocessed data to allow it to be modelled. Missing or inconsistent values, data normalisations and selecting the most relevant attributes for cost overhead factors present core preprocessing steps. These actions influence the quality and dependability of the dataset, they are therefore directly related to model performance. The model building stage involves integrating of XGBoost with Fuzzy logic. The uncertainties that are associated with any agile processes are dealt with by the use of fuzzy logic which is a rule base system that is capable of dealing with ambiguous or imprecise data. Next, based on the obtained fuzzy logic outputs, XGBoost decision tree—a gradient boosting workhorse that earns its recognition for its accuracy and speed is used to classify and rank cost overhead components. The qualities of both methods create a basis for the hybrid methodology; XGBoost ensures scalability and high resilience performance while fuzzy logic is somewhat more interpretable.

For exploring the effectiveness of the proposed model various performance metrics such as accuracy, precision, recall rate, F1 score are analyzed. The results of the model are indicated as superior to other traditional approaches such Support Vector Classifier, Random Forest, and Decision Tree. The general significance of the findings is supported by statistical analysis. In determining its practical relevance in Agile project management the model is subjected to

consequences analysis. Further discussed are more practical aspects including the ability to scale the approach and the effect that subsets of the features have on the model. Finally, we discuss areas of limitations such as need of prior knowledge for creating fuzzy rule base and computational complexity. The set of future works is recommended for addressing such constrains and enhancing the practicality of the model through the automation of fuzzy rule generation and exploring its relevance to other disciplines.

Data Collection and Pre-processinng

The data collection stage involves the identification of data necessary for establishing cost overhead factors for Agile software development. Of this information, current and historical data from surveys of experts, repositories of previous projects, reports from the industry or scientific papers can be used. The collected dataset has characteristics like project size and team structure, tool usage, high requirement volatility, and dependencies which can result into cost overheads. For issues of validity to enhance the reliability of the data, the data is tested through the recommendation of different literatures and experts in the domain field.

- In this step, the data is prior to analysis conducted to undergo processing by using data enhancement techniques to be suitable for the fuzzy based-XGBoost model. The noise is addressed by deleting any duplicate and unnecessary features. Unfilled values are handled using imputation methodologies such as average or median insertion or even the sophisticated k-nearest neighbour (KNN). Outliers or any other inconsistencies in recording data are also treated based on statistical or field standards. Missing value imputation for numerical data is given as

$$\text{xi} = \frac{1}{n} \sum_{j=1}^{n} x_j \quad (1)$$

where xi denotes missing value and xj indicates observed values

Normalisation factor (x') is given below as:-

$$\text{x'} = \frac{x - min(X)}{max(X) - min(X)} \quad (2)$$

X denotes feature vector and x is original value

- Standardization is used to standardise the attributes, so that the variables with large ranges do not dominate the model. Feature transformation is performed in the process of feature pre-processing in which methods like sort/Limit Convert, one hot encoding of 'category' type features etc. Last of all, the classes are discretized and if there is a class imbalance, methods such as SMOTE are used to balance the classes in order to avoid the model settling on a single class due to a large number of instances on that side. Z-score denoted by z, μ denotes mean and σ indicates standard deviation for the feature

$$\text{z} = \frac{x - \mu}{\sigma} \quad (3)$$

Feature Selection and Engineering

Feature selection and engineering activities are considered in the context of improving the predictive capability of the fuzzy-based XGBoost architecture. Feature selection deals with selecting the meaningful attributes that have strong impact on cost overhead factors in Agile software development to minimize the number of inputs to be processed, enhance the computational speed, and avoid model complexity. Feature selection methods such as mutual information analysis, RFE, and from the domain expert's input are used to screen features that may be unimportant for analysis. Furthermore, data structures such as correlation analysis are applied to filter out multicollinearity, which would make the other features irrelevant and redundant.

- While feature engineering tries to overcome data preprocessing limitations by elaborating raw data and adjusting it to be more suitable for modeling cost overhead factors. This process involves generation of new features from the knowledge base of the domain. For instance, it is much more meaningful to sum up attributes such as "team size" and "experience level" into something called "team efficiency index"

$$\text{I(X;Y)} = \sum_{x \in X} . \sum_{y \in Y} p(x,y) \log \frac{p(x,y)}{P(x)p(y)} \quad (4)$$

In above equation I(X;Y) is mutual information between feature X and desired variable Y

- Weaknesses of linear models, such as inability to handle non-linear patterns and interactions between features, are addressed and polynomial feature generation or interaction terms are considered. Transformations of the logs are used where necessary, which puts the data in the right place concerning the basic assumption of the fuzzy-XGBoost model. In the equation, r denotes linear relation between x and y

$$r = \frac{\sum(x_i-\bar{x})(y_i-\bar{y})}{\sum(x_i-\bar{x})^2\Sigma(y_i-\bar{y})^2} \quad (5)$$

- In the enhanced workflow, the fuzzy logic component is in the feature engineering phase and includes linguistic variables and fuzzy rules. For instance, some elements such as “requirement volatility” may be fuzzified to such linguistic terms as low, medium, or high because of subjectiveness. In conjunction with XGBoost, these fuzzy variables enhance interpretability of output data and enhance the model to reason about uncertainty and imprecision that is part of agile projects.
- Finally, the stage guarantees that the model relies on suitable and high-quality features when it comes to cost overhead aspect classification and prioritization in Agile software development. This approach not only increases the reliability of forecasting but also increases the model readability and usability.

Class Imblance

The class imbalance stage addresses a critical challenge in developing an effective fuzzy-based XGBoost model: the imbalance of classes in the data set. Due to skewed class representation, a number of cost overhead factors may be encountered significantly more often while practicing Agile software development. This imbalance may render the model overly sensitive to the majority class and less sensitively to the minority classes even where they depict rare but resource-critical costs.

- In the light of this a good strategy for handling imbalance in class is put in place. With this, oversampling methods, for instance Synthetic Minority Oversampling Technique (SMOTE), are employed in order to create synthetic data from the minority classes. SMOTE synthesises new samples through the intermediation of the existing minority class data points in a normally distributed feature space while optimally migrating the intrinsic distribution of new samples. SMOTE for synthetic sample analysis (xnew) is given as

$$xnew = xminority + \lambda(xnearest - xminority) \quad (6)$$

  λ is random number between 0 and 1, xnearest is the closest neighbour of xminority

- Another sampling technique considered here is undersampling in which examples in the majority class are removed selectively in order to balance the class. Nevertheless, this type of processing is applied carefully to avoid the possible deletion of critical data. However, a combination of oversampling and undersampling can also be used in order to obtain the best outcome.

  In over-sampling,

$$Ratiooversample = \eta majority/\eta minority \quad (7)$$

  In under-sampling,

$$Ratioundersample = \eta minority/\ \eta majority \quad (8)$$

- Another adaptive solution is about integrating cost-sensitive learning into the fuzzy based XGBoost model. The misclassification penalties here have been designed to be high for minority classes in order to encourage the model to pay particular attention to them.
- Further, stratified sampling is applied during the data division step as the independent sets are created to reproduce the class distribution. Common evaluation techniques consist of precision, recall, and F1 score instead of accuracy metrics to retain a fair coverage of imbalanced data conditions.

Data Splitting and Standardization

It is crucial for further stages named Data Splitting and Standardization in order to provide the readiness of the dataset for the learning and evaluation of the fuzzy-based XGBoost model while removing the prejudicial influence from the data fluctuation.

- *Data Splitting* is a process of dividing the selected dataset into different part to use in training of the models and also in their evaluation. Typically, the data is divided into training, validation, and testing sets, with a common ratio being 70:15:15 or 80:10:10. When the evaluation is made across these subsets the stratified sampling is used to ensure that each class has a proper representation in case of class imbalance. This helps to guarantee that the model is learning more or less the real distribution from the training data and the validation and testing data sets are more balanced.
- *Standardisation* is normally done in order to ensure all feature have the relative values in their range and are transformed into a standard range. In Agile datasets thus, features like size of the team, size of the project or the frequency of tool usage may come in a different units and range and hence can skew the model significantly. Standardization means to altering each feature so that it has the same mean value of zero and a standard deviation of one by using normalization where z score is employed is used. This helps to keep all attributes equally involved in making judgements during model training, not only when establishing distances in distance-based computations and the gradients when used by XGBoost.
- Also, any transformation applied on the training set like standardization, normalization, and the like is used on the training set only, and is applied uniformly to the validation set as well as the testing set. This helps to exclude biased results due to a leakage of the evaluation and accredits the results to the model.

Model Development, Training and Validation

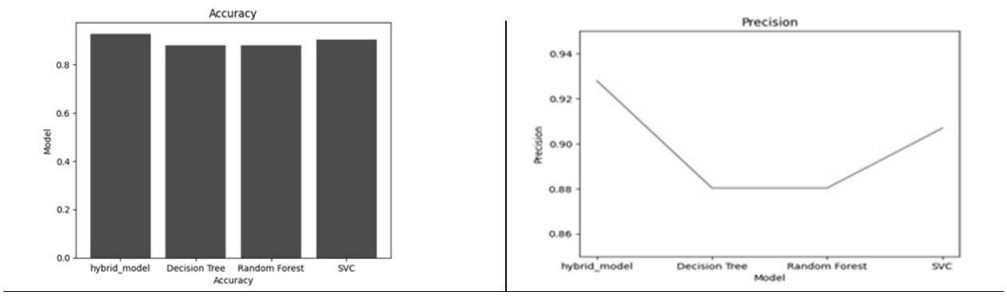
XGBoost based fuzzy Model Development, Training and Validation stage was performed to build a fuzzy-based framework to classify and prioritize cost overhead factors in the Agile software development. The model is as a combination of XGBoost and Fuzzy logic to handle uncertainty and imprecision. Linguistic variables, say such as 'high requirement volatility' are defined with fuzzy rules and membership functions that map them to meaningful numerical representation.

- The XGBoost algorithm first iteratively optimizes decision trees for training to minimize a loss function using gradient boosting to be more accurate and robust. Interpretability is enhanced by fuzzy engineered features, and the model better deals with ambiguity in the cost factors. Grid or random search methods are used to fine tune hyper parameters like learning rate, tree depth and regularization terms to get the best performance possible.
- The holdout or k-fold cross validation validation is performed to validate the model applied, and it is shown to generalize well for unseen data. Classifi¬cation effectiveness is measured with evaluation metrics like precision, recall,F1score, the Area Under the Receiver Operating Characteristic Curve (AUC-ROC).

This stage results in a well trained validated model to accurately identify and prioritize cost overhead factors in the Agile environment.

IV. EXPERIMENTAL RESULTS

In this section, an evaluation of the proposed model is conducted by using meaningful measurements such as accuracy, precision, recall, the F1-score, as well as the AUC-ROC. Categorizations that define disadvantageous features of the base model help to define the benefit from the use of the proposed model. Such techniques as confusion matrix and feature importance or ranking and ROC curves also aids interpretability. Furthermore, proving the applicability of the presented model in concrete Agile projects makes the suggested concept more relevant.



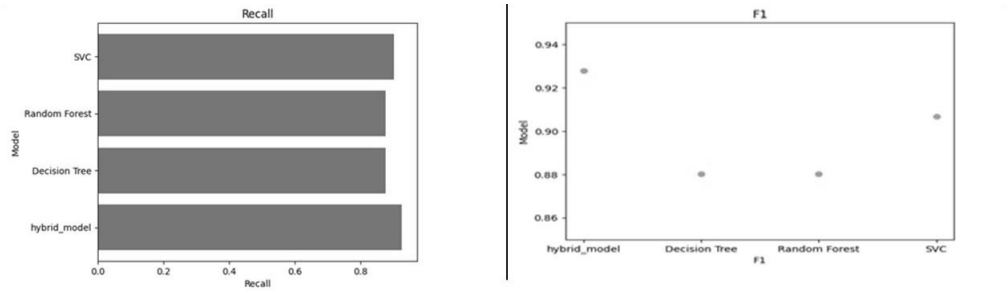


Figure 1: Experimental Results

Accuracy

Superior accuracy is shown by the hybrid fuzzy-based XGBoost model over conventional models like SVC, Random Forest, and Decision Tree. Although Decision Tree offers interpretability, it is not always strong for complicated data. Random Forest uses ensemble learning to increase stability, however with noisy data it may have declining benefits. SVC suffers with scalability and hyperparameter adjustment but shines on smaller datasets. The hybrid solution uses XGBoost's precision and fuzzy logic's uncertainty management to provide better classification and prioritising accuracy for cost overhead considerations.

Precision

The results show that the hybrid fuzzy-based XGBoost model prediction yields better accuracy compared to Decision Tree, Random Forest, and SVC. What is more, Random Forest increases the accuracy by approach based on the ensemble but can miss the focus on the minority class, as well Decision Tree has a drawback that its precision is limiting due to overemphasizing problem. SVC performs well in balanced data set; in imbalanced ones it tends to struggle. The integration of the exactness of feature significance of the XGBoost with the incorporation of the fuzzy logic concept of uncertainty assurance promises more accuracy in the prioritisation of the cost overhead factors in Agile development.

$$\text{Precision} = TP/(TP+FP) \quad (9)$$

TP = True Positives, FP = False Positives

Recall

Recall of the hybrid fuzzy-based XGBoost model over the Decision Tree, Random Forest, and SVC is established Because Decision Tree is inclined to over fit small data sets, it occasionally compromises recall. While it suffers of distorted data, Random Forest reduces variance in order to increase recall. Classroom interaction with another class has a negative impact on SVC's performance. The above integration enhances recall through equalising the uncertainty management of fuzzy logic and gradient boosting of XGBoost, thereby ensuring identification of all facets of cost overheads of Agile development.

$$\text{Recall} = TP/(TP+FN) \quad (10)$$

FN = False Negatives

F1 Score

Our proposed fuzzy based XGBoost hybrid model among Decision Tree, Random Forest and SVC yields the maximum F1 score hence indicating that it has provided the best compromise between accuracy and recall. Incorporated from an ensemble of trees, Random Forest slightly improves the F1 score although Decision Tree tends to overfit and may not perform well on the minority class datasets. SVC can achieve high F1 score in balanced datasets while it fails to do well in imbalanced complex data. The pairing of the rough logic system with feature importance optimisation in XGBoost assures a high F1 score for costs of overhead.

$$\text{F1 score} = 2.\ \text{Precision}.\ \text{Recall} / (\text{Precision} + \text{Recall}) \quad (11)$$

Discussion

The results reveal that the proposed fuzzy-based XGBoost method significantly enhances the classification performance and prioritising efficiency to meet critical concerns for enhancing the control of cost overhead components. This hybrid approach assists Agile teams in arriving at better decisions over data, optimise resource consumption, and minimise waste by employing XGBoost for high performance classification and fuzzy logic for managing uncertainty. The results highlight the rational advantages of the proposed approach. Greater reliability and sustainability offer reliable identification and categorization of cost-associated factors; right away advancing the project outcomes. It could prove beneficial in helping agile workers enhance their team performances for the projects they are handling and find ways to satisfy its needs during the project implementation as well as improve some processes related to decision-making.

V. CONCLUSION AND FUTURE WORK

This research emphasizes the benefits of the fuzzy-based XGBoost approach in Agile software development, showcasing its contribution to addressing key challenges such as precise classification and prioritization of cost overhead factors. The integration of fuzzy logic with XGBoost enables the model to handle uncertainties inherent in Agile environments while delivering high accuracy and effectiveness. This hybrid approach enhances decision-making in resource allocation and cost management, significantly improving Agile project outcomes. The study offers several research contributions. It introduces an innovative method that combines fuzzy logic's adaptability with XGBoost's powerful classification capabilities, providing a systematic solution to a critical problem in Agile development. Additionally, the model's superior performance across metrics such as accuracy, precision, recall, and F1 score demonstrates its practical applicability in real-world scenarios. Theoretical advancements are also evident, as the approach bridges gaps in existing literature by addressing both uncertainty and classification efficiency simultaneously. Future research directions could explore further enhancements, such as integrating other machine learning techniques, extending the approach to other domains, or refining the fuzzy rules for specific Agile contexts. The study's findings underscore its theoretical and practical significance, offering a robust foundation for advancing Agile practices and inspiring further innovation in cost overhead management.

REFRENCES

[1] Abusaeed, S., Khan, S. U., & Mashkoor, A. (2023). A fuzzy AHP-based approach for prioritization of cost overhead factors in agile software development. Applied Soft Computing, 133, 109977. https://doi.org/10.1016/j.asoc.2022.109977

[2] Bartl, D., & Ramík, J. (2022). A new algorithm for computing priority vector of pairwise comparisons matrix with fuzzy elements. Information Sciences, 615, 103–117. https://doi.org/10.1016/j.ins.2022.10.030

[3] Cui, H., Zhang, H., Zhou, L., Yang, C., Li, B., & Zhao, X. (2023). Fuzzy analytic hierarchy process with ordered pair of normalized real numbers. Soft Computing, 27(17), 12267–12288. https://doi.org/10.1007/s00500-023-08232-7

[4] Duarte, C. H. (2022). Software productivity in practice: A systematic mapping study. Software, 1(2), 164–214. https://doi.org/10.3390/software1020008

[5] Ghimire, D., & Charters, S. (2022). The impact of Agile Development Practices on project outcomes. Software, 1(3), 265–275. https://doi.org/10.3390/software1030012

[6] Huang, M.-H., & Rust, R. T. (2020). A strategic framework for artificial intelligence in marketing. Journal of the Academy of Marketing Science, 49(1), 30–50. https://doi.org/10.1007/s11747-020-00749-9

[7] Huss, M., Herber, D. R., & Borky, J. M. (2023). Comparing measured agile software development metrics using an agile model-based software engineering approach versus Scrum only. Software, 2(3), 310–331. https://doi.org/10.3390/software2030015

[8] Javanbarg, M. B., Scawthorn, C., Kiyono, J., & Shahbodaghkhan, B. (2012). Fuzzy AHP-based multicriteria decision making systems using particle swarm optimization. Expert Systems with Applications, 39(1), 960–966. https://doi.org/10.1016/j.eswa.2011.07.095

[9] Kirpitsas, I. K., & Pachidis, T. P. (2022). Evolution towards hybrid software development methods and information systems audit challenges. Software, 1(3), 316–363. https://doi.org/10.3390/software1030015

[10] Lee, J. Y., Burton, H., & Lallemant, D. (2018). Adaptive Decision Framework for civil infrastructure exposed to evolving risks. Procedia Engineering, 212, 435–442. https://doi.org/10.1016/j.proeng.2018.01.056

[11] Liu, L., & Ranji Ranjithan, S. (2010). An adaptive optimization technique for dynamic environments. Engineering Applications of Artificial Intelligence, 23(5), 772–779. https://doi.org/10.1016/j.engappai.2010.01.007

[12] Liu, Y., Eckert, C. M., & Earl, C. (2020). A review of Fuzzy AHP methods for decision-making with subjective judgements. Expert Systems with Applications, 161, 113738. https://doi.org/10.1016/j.eswa.2020.113738

[13] Martínez-López, J. A., García, F., Ruiz, F., & Vizcaíno, A. (2023). Contributions of Enterprise Architecture to Software Engineering: A systematic literature review. Journal of Software: Evolution and Process, 36(4). https://doi.org/10.1002/smr.2572

[14] Paschen, J., Paschen, U., Pala, E., & Kietzmann, J. (2020). Artificial Intelligence (AI) and value co-creation in B2B sales: Activities, actors and resources. Australasian Marketing Journal, 29(3), 243–251. https://doi.org/10.1016/j.ausmj.2020.06.004

[15] Shi, H., Lin, Z., Zhang, S., Li, X., & Hwang, K.-S. (2018). An adaptive decision-making method with Fuzzy Bayesian reinforcement learning for Robot Soccer. Information Sciences, 436–437, 268–281. https://doi.org/10.1016/j.ins.2018.01.032

[16] Ståhl, D. (2023). The dynamic versus the stable team: The unspoken question in large-scale agile development. Journal of Software: Evolution and Process, 35(12). https://doi.org/10.1002/smr.2589

[17] Wiesmann, D. (2023). Avoidance of the term agile in software engineering: Necessary and possible. Journal of Software: Evolution and Process, 35(12). https://doi.org/10.1002/smr.2566

[18] Zou, J., Lou, J., Wang, B., & Liu, S. (2024). A novel deep reinforcement learning based Automated Stock Trading System using cascaded LSTM Networks. Expert Systems with Applications, 242, 122801. https://doi.org/10.1016/j.eswa.2023.122801