INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

Analysis of Traditional and Agile Software Development Process for Developing Recommender Model using Machine Learning

Purvi Sankhe[1]*, **Mukesh Dixit**[2]

1 Research Scholar, Sanjeev Agrawal Global Educational University, Bhopal, Madhya Pradesh, India
2 Associate Professor, Sanjeev Agrawal Global Educational University, Bhopal, Madhya Pradesh, India

OPEN ACCESS

Received: 19-03-2024
Accepted: 12-04-2024
Published: 09-05-2024

Citation: Sankhe P, Dixit M (2024) Analysis of Traditional and Agile Software Development Process for Developing Recommender Model using Machine Learning . Indian Journal of Science and Technology 17(19): 1983-1993. https://doi.org/10.17485/IJST/v17i19.833

* **Corresponding author**.

purvisankhe@gmail.com

Funding: None

Competing Interests: None

Copyright: © 2024 Sankhe & Dixit. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objective: To create an AI-powered recommendation system that is designed for IT professionals to help them choose the best software development approaches. Through the use of specified data parameters. **Methods:** The recommendation system will make use of machine learning algorithms and data analysis methods to examine team dynamics, project needs, and other variables. The technology will enable developers to improve the quality of products and speed up the development process by recommending suitable development methodologies. Data parameters considered for the development of the recommendation model fall into four categories: requirements, user involvement, development team, type of project, and risk associated with it. **Findings:** Existing recommendation systems developed by different researchers are applicable for only requirement elicitation and to recommend different phases of the development process, whereas systems that will help select development methodology are not available in the existing systems. Among the five machine learning algorithms applied in the recommender system building process, the DecisionTree Classifier and RandomForest Classifier exhibit superior performance, achieving 100% accuracy, while the Kneighbors Classifier indicates 94.74% accuracy. **Novelty:** This study of systems introduces a novel approach to software development methodology, a recommender system, which helps IT developers select the best appropriate development approach for the development of a software product or project based on the type of project to be built and other data parameters.

Keywords: Agile; Development; Requirements; Methodology; User; Customer

1 Introduction

The Software Development Process involves applying engineering principles to software development, with a focus on designing, building, testing, and maintaining high-quality

software systems. It is a collaborative and multidisciplinary field that requires a combination of technical skills, project management expertise, and effective communication. The application of engineering principles to software development helps create reliable and maintainable software systems that meet the needs of users and stakeholders. Two standard approaches are used for the development of software products, namely the traditional software development process and an agile software development process[(1)]. The traditional waterfall model depicts water cascading down a cascade as a sequential flow through several stages. First comes analysis, followed by design, implementation, testing, and maintenance. While the Agile software development process emphasizes adaptability, teamwork, and iterative development, each step builds on the one before it to create a logical approach to software development[(2)]. It is an approach that prioritizes the client's satisfaction and modifies the process as development progresses to meet changing needs. The concepts that drive agile development are: focusing on individuals and their relationships instead of procedures and tools; a working version of software is preferred over extensive documentation; the involvement of clients in contract negotiations; and being easily adaptable to any changes[(3)].

Software engineering recommendation systems now in use are used to suggest various stages of development as well as the requirement elicitation process. Studies and research at present lack a recommendation mechanism that would suggest a development method[(4)] [(5)]. IT firm developers are constantly faced with a dilemma while choosing a software development approach to create higher-quality software that fulfills all the requirements of customers. So this problem of deciding which software development approach is best for a particular project while taking team dynamics and user involvement into account Risk management and project needs are resolved by offering a recommender model. Teams can improve project success, increase efficiency, and make better decisions with the help of this tool. For a number of reasons, developing a recommender model for software development methodologies can be a highly satisfying project. There are many different approaches available for the software development process, which is frequently complex and diversified. Creating a recommender model gives developers and teams the chance to address a practical issue by guiding them through a complex variety of approaches to select the one that works best for their project. The efficiency and efficacy of software development teams may be enhanced by the developed recommender model.

The first section of this research paper addresses the data parameters that should be taken into account while choosing a software development methodology. The algorithms used to recommend development processes are covered in the next portion of the approach. The results of each algorithm are displayed, along with an accuracy plot for each algorithm, and a comparative analysis is shown in the results section. The conclusion is included in the next section.

2 Methodology

In this research paper, software Development Methodology Recommender Model is built using five different machine learning algorithms. This model will help IT professionals to select best appropriate methodology to develop better quality software product.

2.1 Analysis of Dataset Parameters used for the selection of development methodology

2.1.1 Waterfall Model

One of the most traditional and widely used approaches to software development is the waterfall model. It represents a methodical process for developing software. Determining the program's requirements is the first step in the process, which is followed by design, construction, testing, and eventually user delivery. It's been around for a long time and was the first way ever used to create software. Software is created using a sequential and linear process, as seen in Figure 1, where each project step is finished before going on to the next. Due to the lack of alternate software development approaches in the past, the model's construction was driven by developments in hardware development[(6)].

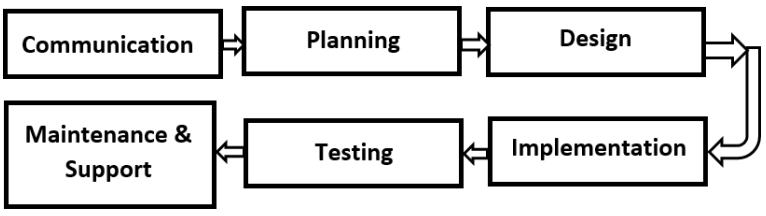

Fig 1. Traditional Software Development Process

2.1.2 Agile Software Development Process

Agile development uses iterative approach as shown in Figure 2. It has a number of major advantages, including quicker delivery times, enhanced quality, more cooperation and communication, and greater adaptability to changing requirements [7]. Agile, however, might not be appropriate for all projects or organizations and for teams to really embrace it, there must be a fundamental transformation in mindset and culture [8].

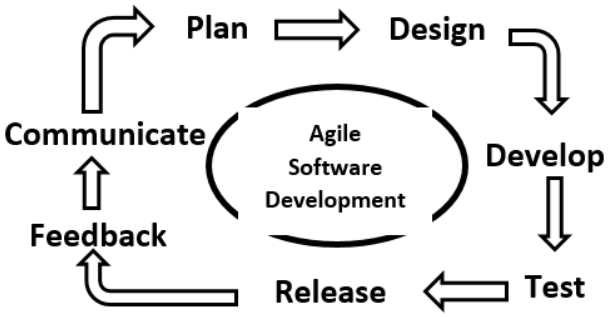

Fig 2. Agile software development approach

The Agile concept is centered around the Manifesto, which outlines the four fundamental values of Agile. While there has been discussion regarding the manifesto's continuous significance, it still serves as the foundation for the organization. Four values of agility are shown in Figure 3. Tools and processes are not as essential as individual relationships, a preference for functional software above comprehensive documentation, working together as opposed to negotiating contracts, and an emphasis on adapting to change.

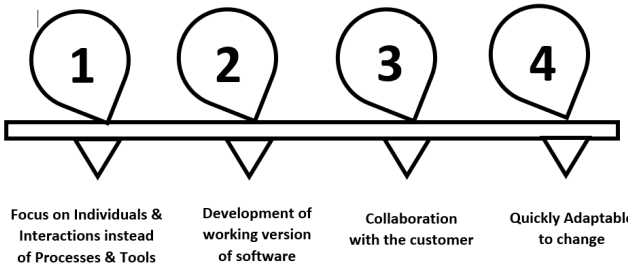

Fig 3. Important Values to the Agile Software Development Model

As shown in Figure 3 the concepts that drive agile development are: focusing on individuals and their relationships instead of to procedures and tools, working version of software is preferred over extensive documentation, involvement of clients in contract negotiations & easily adaptable to any changes [9].

Each iteration of agile development entails communication, plan, design, development, testing, and releasing a tiny, working piece of software [10]. This approach is iterative and incremental. This strategy enables ongoing input and progress throughout the entire process [11] [12].

2.1.3 Dataset Parameters used for the selection of development methodology

The following are the primary parameters that are crucial in determining which software development approach should be applied to create software that fully meets client needs.

- Requirements of the Project,
- User Involvement,
- Development Team,
- Type of Project & Risk Associated with it.
- **Requirements of the Project:** Open communication and a common objective are essential for the identification, clarification, and classification of both functional and non-functional requirements. While functional requirements

outline what the system must be able to do, non-functional requirements address qualities like performance, usability, security, and scalability. Upon gathering & validating all the requirements of the project it is necessary to check whether requirements are defined early in the SDLC, requirements are easily defined and understandable, whether Requirements are changed frequently & whether changes in requirements will be easily manageable or not[13].

- **User Involvement:** In the process of assessing the software development process's applicability, it is important to pay particular attention to the following important user involvement factors: Limited User Involvement, User Participation in All Phases No experience participating in similar projects and development teams[14].
- **Development Team:** The experience, expertise, and skill set of the team as a whole can have a significant impact on the project's results as well as the suitability of the selected methodology. Some key factors to consider when assessing the development team's readiness to adopt the software development process are: team experience, domain knowledge, experience with the tools needed for the project, and the and the need and availability of training[15].
- **Type of Project and Risk Associated:** It focuses on factors that give detailed information about the type of project based on whether the project is an improvement of an old system: stable funding, highly reliable requirements, a reliable requirement, a tight schedule, reusable components, and scarce resources. So projects involving the enhancement of an outdated system, steady funding, extremely dependable requirements, a strict timeline, the use of reusable components, and limited resources are ideal candidates for the software development methodology[16].

Table 1 gives briefing of data set required for the selection of waterfall approach for the development of software product.

Table 1. Data Parameters of Waterfall & Agile Software Development Process

Parameters	Traditional Software Development Process	Agile Software Development Process
Requirements of the Project		
All the Requirements are completely defined early in the SDLC	YES	NO
Requirements are easily defined and understandable	YES	NO
Requirements are changed frequently	NO	YES
User Involvement		
Limited User Involvement	YES	NO
User Participation in All Phases	NO	YES
Development Team		
Little Domain Knowledge- new to Technology	YES	NO
Little experience on Tools	YES	NO
Training Availability when needed	NO	YES
Availability of Tester from the start of Project	NO	YES
Availability of technical leadership skilled professionals	NO	YES
Type of Project		
Small project size	YES	NO
Medium to Large project size	NO	YES
Improvement of an Old System	NO	YES
Stable Funding	YES	NO
Tight Schedule	NO	YES
Slower Deployment time	YES	NO
Faster Delivery	NO	YES
Reusable Components	NO	YES
Scarce Resources	YES	NO
Faster Deployment times	NO	YES
Adaptive to Customer needs	NO	YES
Need of Documentation	YES	NO
Flexibility	NO	YES
Continuous Feedback	NO	YES
Adaptive to Customer needs	NO	YES
Continuous Feedback	NO	YES
Risk Associated		
Risk Analysis & Management	NO	YES
Possibility of risk association	YES	NO
Security	NO	YES

Continued on next page

Table 1 continued

Less Guarantee of success	YES	NO
Successful guaranteed product delivery on time	NO	YES

2.2 Implementation Methodology

Machine learning algorithms used for the development of recommendation systems are Logistic Regression, Kneighbors Classifier, Gaussian NB, DecisionTree Classifier, and Random Forest Classifier. The figure shows the flow of the implementation methodology used for the development of the software development methodology recommendation system. Steps included in the methodology are as shown in Figure 4.

Step 1: Understand the Problem: The problem we're attempting to address must first be precisely defined. This involves figuring out what needs to be accomplished and drawing some conclusions from the problem description.

Step 2: Gather Data: Next, information is gathered from various sources. Depending on the project, this information may come from files, databases, or other sources. The quantity and quality of the gathered data have a direct bearing on the accuracy of our system.

Step 3: Prepare the Data:Once the data is gathered, it needs to be cleaned up. This involves removing any information that is inconsistent, missing, or duplicated. Raw data is converted into a clean dataset because it isn't ready for immediate use.

Step 4: Train the Model: The model undergoes training to enhance its functionality. A training set and a testing set are created from the dataset; typically, 80% is used for training and 20% is used for testing. The testing set enables us to assess the model's performance, while the training set assists in the model's learning from the data.

Step 5: Evaluate the Model: The model is evaluated by testing. A testing set is used to evaluate its performance on newly acquired data. Metrics like recall, accuracy, and precision are used to evaluate its effectiveness. It may be necessary to modify certain settings or parameters if the model performs poorly.

Step 6: Use the Model: Finally, after we are satisfied with the model's functionality, we may start using it to solve real-world issues. This can entail applying the model's acquired knowledge to resolve other problems, formulate forecasts, or categorize data.

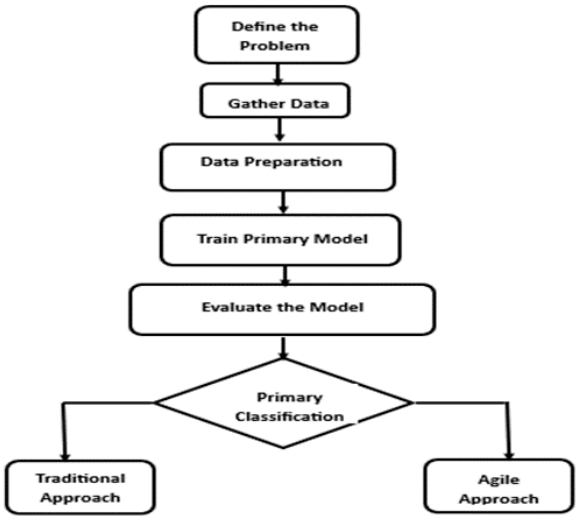

Fig 4. Flow diagram of recommendation system building

The study of a "real world" event using scientific principles to iteratively validate and improve a model or hypothesis is known as machine learning (ML) [17]. Following are the Machine learning algorithms used for developing software methodology recommender model.

Logistic Regression: When analyzing outcomes with only two possible outcomes, one common method used is logistic regression. Understanding choices and results in diverse circumstances is made easier by it [17]. Regression modeling accomplishes two key tasks: first, it predicts the result based on updated knowledge about the contributing factors. Second, by illustrating the relative contributions of each aspect to the final result, it helps in our understanding of the issue under

consideration [18]. Thus, while accounting for other factors, we can determine which factors are truly important and how they impact the outcome. Numerous independent variables $X1, X\ldots, Xk$ can be examined in relation to the dependent variable Y using the logistic regression model. The variable Y is dichotomous, meaning it can only have two values. In logistic regression, the outcomes are denoted by the numbers 1 and 0. If all goes well, we employ 1. We use 0 if it's something we don't want. We can determine which factors are critical, which ones are not, and how each one affects the others by using the analysis [19].

Kneighbors Classifier: Under the umbrella of supervised learning, the K-NN algorithm is a straightforward machine learning technique. It's similar to having a teacher help you when you learn under supervision [19]. You provide the machine with a large number of labeled examples, such as images of dogs and cats, and indicate to it which is which. Based on its prior learning, it can then determine if a new picture it sees is of a dog or a cat.

Let us consider the case of K-NN, in which you have two groups, A and B, and you are assigned a new point, X_1. You wish to ascertain which group X_1 is a member of. This is determined by the K-NN method, which looks at the closest data points to X_1 and determines which group they belong to. Determining where X_1 fits best is similar to asking your neighbors (the closest data points) which category they belong to.

The following is a description of how the K-NN algorithm functions.

Step-1: chooses the neighbor with the K-number first.

Step-2: The Euclidean distance needs to be computed for every K neighbors.

Step-3: Determine the K closest neighbors using the Euclidean distance calculation.

Step-4: Determine how many data points there are in each category for each of these K neighbors.

Step-5: The category with the greatest number of neighbors will receive new data points.

Step-6: The K-NN classification model will be constructed in this way.

Gaussian NB : The Naive Bayes algorithm is a simple learning technique that combines a strong assumption of feature independence within each class with Bayes' rule. Although Naive Bayes routinely attains competitive accuracy in classification tasks, its dependence on the feature independence assumption may cause problems in practical applications [20]. Because of its numerous advantages and high computational efficiency, Naive Bayes is an extensively utilized algorithm in practice. With an object x, it assists us in determining the posterior probability value, P(y|x), for each class y. For this assessment, Naive Bayes offers a helpful tool through sample data analysis. Once completed, these assessments can be applied to categorization tasks or any other type of decision-making.

DecisionTree Classifier: Classification systems, which manage massive amounts of data by grouping it into distinct classifications, are frequently used in data mining. These algorithms aid in the classification of freshly obtained data, the classification of data based on training sets and labels, and the inference of categorical class names. The decision tree algorithm, the subject of this study, is a well-known classification algorithm. Decision trees make the algorithm's structure easier to understand by providing a visual representation of it. Decision trees are strong techniques that are applied in many domains, including image processing and machine learning. They function by effectively combining a number of fundamental checks and comparing numerical properties to test threshold levels. Decision trees are easier to comprehend than neural networks since they rely on conceptual rules instead of numerical weights. Each tree is made up of nodes and branches, and they are frequently employed in data mining tasks, including grouping and categorization. Every node in a decision tree represents a feature that needs to be classified, and every subset suggests a potential value for that feature. Decision trees are extensively used because they are accurate and easy to analyze for a variety of data types. These qualities enable them to find use in a variety of fields [21].

RandomForest Classifier: In machine learning and data science, the Random Forest classifier is a well-liked ensemble classification method that is used in a variety of application domains. It makes use of an approach called "parallel ensembling," in which several decision tree classifiers are trained concurrently on various dataset subsets. Usually, the final prediction is decided by averaging the results of these different classifiers or by a majority voting mechanism [22]. Random Forest improves prediction accuracy and control while addressing overfitting issues in an efficient manner. As a result, multiple decision tree-based Random Forest learning models generally perform better than single decision tree-based models [22]. Through the combination of random feature selection and bootstrap aggregation (bagging), it generates a sequence of decision trees with regulated variability. This method is flexible, able to handle both continuous and categorical data, and may be applied to both regression and classification applications [23]. Random Forest is a powerful ensemble learning algorithm in machine learning. Table 2 shows a list of questions asked to get the data values for the traditional and agile software development approaches.

Table 2. Traditional & Agile methodology data used for the design of Recommender model

Sr. No.	Questions	Traditional (Waterfall)Software Development Approach	Agile Software Development Approach
1	All the requirements are clearly defined at the start of project only.	TRUE	FALSE
2	Not easily adaptable to any changes after the start of the project	TRUE	FALSE
3	User is involved only during starting phase of the project.	TRUE	FALSE
4	Extra Skilled professionals are not required in team	TRUE	FALSE
5	No need of tester from the start of project. Tester is required only during testing phase.	TRUE	FALSE
6	Works only on small size projects.	TRUE	FALSE
7	Time taken for development of product is more	TRUE	FALSE
8	cost required is more.	TRUE	FALSE
9	Risk analysis and management is done at moderate level.	TRUE	FALSE
10	Documentation is very important or created at each phase pf project.	TRUE	FALSE
11	Preferable for improvement of an old systems	FALSE	TRUE
12	reusable components are Developed	FALSE	TRUE
13	Flexibility in the process	FALSE	TRUE
14	Good security provided	FALSE	TRUE
15	Deployment time is less	FALSE	TRUE

3 Results and Discussion

3.1 Existing System

Faiz Akram's systematic literature review[4] demonstrates the requirements for elicitation method that has been integrated with recommendation systems that use collaborative filtering suggestion systems to forecast the requirements of the stakeholders based on their preferences for functional and non-functional requirements. This recommendation system aids in the resolution of issues such as: How might recommendation systems help identify stakeholders throughout the requirements elicitation process? How can the selection of requirements elicitation approaches be automated?

Recommendation algorithms have been built as a result of Juri Di's 2021 CROSSMINER[5] expertise to assist software developers at different phases of the process. The systems provide developers with a range of artifacts, such as third-party libraries, usage guidelines, and required API function calls for the adopted APIs. In order to provide these recommendations, technical choices have been made to address a variety of issues, such as the lack of baselines, the shortage of data, the use of performance indicators, and evaluation techniques.

Liang Wei[24] developed four open-source recommendation systems in 2021 and described their implementations, benchmarking and comparing each one. Fuzzy heuristics, such as which tools have been used previously and which business domains are pertinent to the current demands, can be configured in a recommendation system. A technical lead or project manager can utilize these fuzzy heuristics to identify the best software tools for handling tasks and defect tracking, communication channels, source code management, and documentation, among other things. They categorized them using the features of the software solutions they offered, and then they made recommendations for a rule-based recommendation system based on that information.

3.2 Proposed System

As shown in Table 3, the current situation lacks a structure for recommending software development methodologies. The proposed approach is used to create a recommendation system that IT industry software developers will use to choose the best development strategy for creating higher-quality software products that will meet all of the needs of the client.

The experiments were carried out on the four classes of datasets (0, 1, 2, and 3), with 80% training and 20% testing data. Class 0 denotes the waterfall approach; class 1 denotes the agile approach; class 2 denotes both approaches; and class 3 denotes

Table 3. Research Gaps in existing recommendation sytems & novelty in proposed approach

Existing System	Gaps	Proposed System
A recommendation system to help identify stakeholders in the requirements elicitation process and to recommend ways to automate the selection of requirements elicitation techniques. Algorithms for recommendation: to support software developers throughout the process, the systems provide developers with a range of artifacts, such as third-party libraries, usage guidelines, and required API function calls for the adopted APIs. Recommendation systems that can create fuzzy heuristics, like which tools have been used in prior projects and which business domains are relevant to the demands of the moment, will be utilized by the project manager.	Existing recommendation systems are available to identify elicitation techniques, different phases of software development, and tools to be used for the development. There is no recommender system that can help recommend software development methodologies to be selected.	A recommender system for software development methodologies that assists IT developers in choosing the most suitable development technique for a software project or product depending on the type of project to be produced and other data criteria.

any one of the traditional or agile development approaches. The following table shows the performance of all four classes using Logistic Regression, Kneighbors Classifier, Gaussian NB, DecisionTree Classifier, and Random Forest Classifier.

The counterplot of all four classes with 8000 plus database on each class is as shown in Figure 5.

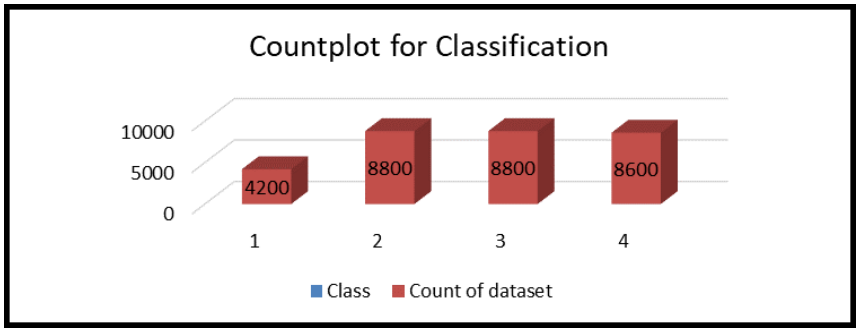

Fig 5. Countplot of dataset for classification

The confusion matrix of all the algorithms is shown in Figure 6.

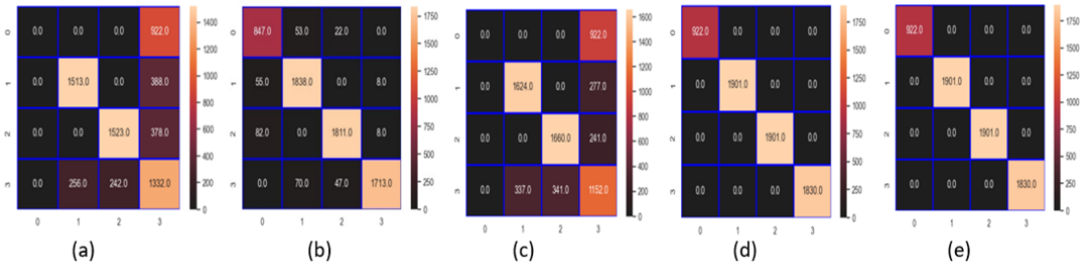

Fig 6. Confusion Matrix of (a) Logical regression (b) Kneighbors Classifier (c) Gaussian NB (d) DecisionTree Classifier (e) Random Forest Classifier

The comparative analysis of all five algorithms w.r.t. Recall, Precision, F1-score & support is shown in the Figure 7.

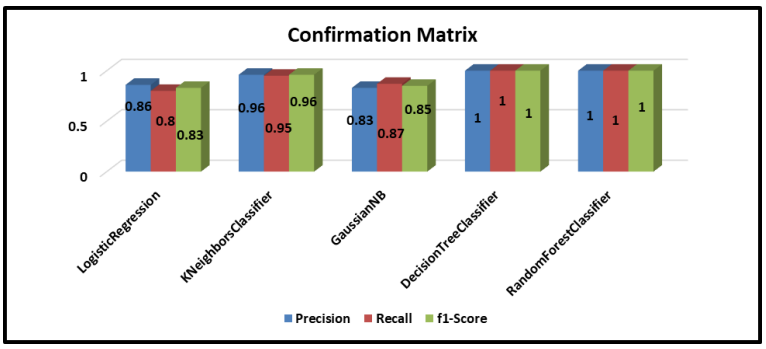

Fig 7. Comparative analysis of ML algorithms

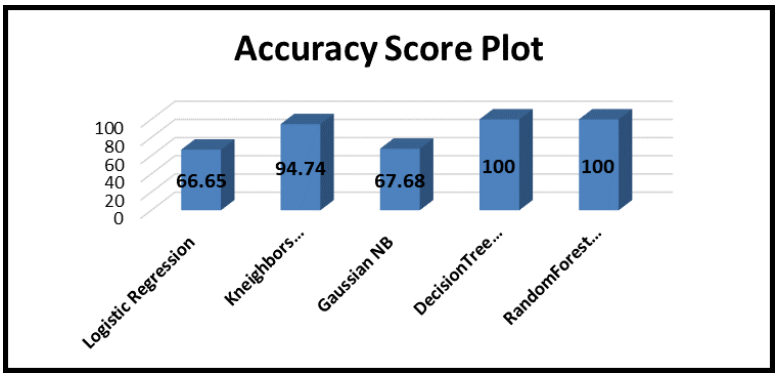

Fig 8. Accuracy Score Plot of all the Machine Learning Algorithm

As seen in Figures 7 and 8, it has been observed that pre-processing and data cleaning using various dataset classifications have been found to have a significant impact on the accuracy of machine learning models. The methods used for the DecisionTree and Random Forest classifiers show 100% accuracy, while the Kneighbors classifier yields 95% accuracy and the Kneighours classification system, logistic regression, yields 67% accuracy.

3.3 Comparative Discussion of Proposed System and existing System

Table 4 shows a comparative analysis of existing systems with the proposed system shown in this research article.

Table 4. Comparative analysis of Existing Systems & Proposed System

Existing System	Proposed System
The recommender system proposed by Faiz Akram et al. [4] in 2024 has developed a recommender system that helps in the selection of requirements gathering techniques and is used only during the first phase of the software development process, i.e., during communication activity. It will not be applicable for the rest of the software development process.	Proposed sytem is applicable throughout the all phases of software developeemnt process i.e. Communication, Planning, Modelling , Construction & deployment of the software system/ product.
The existing system proposed by Rocco JD et al. in 2021 [5] focuses on the CrossMiner project, which has developed various recommendation systems to recommend tools to be used, requirements for supporting libraries, and suggested APIs to be used during various phases of the software development process through different recommendation systems.	Rather than creating multiple recommendation systems, the proposed system created a single recommendation system that applies to the entire software development process and assists IT developers in choosing the best development methodology to produce a higher-quality product that will meet all of the needs of the client.

Continued on next page

Table 4 continued

In 2022, Michael B. et al. [7]. offered a system for project management practice recommendations that assesses if an algorithm can choose the right practices for a given situation. Additionally, they instituted important procedures designed to promote increased adaptability and agility in the development setting.	The proposed system incorporates a developed recommender model to suggest software development approaches, which are used by IT industries to directly choose development approaches for their projects. The model considers requirements, user involvement, development teams, project types, and project risks. Accuracy of the results produced is 100% using machine learning algorithms.
In 2021, Liang Wei [24] created four open-source recommendation systems, detailed how they were implemented, and compared and benchmarked each one. A recommendation system can be designed with fuzzy heuristics, like which business domains are relevant to the current demands and which tools have been used in the past. They make recommendations about which tool is most suited for current project work, at what phase it can be utilized, and how it should be documented based on past experience and past usage of tools, techniques, and documentation strategies.	

3.4 Benefits of Proposed System over Existing Systems

The current state recommendation systems concentrate on particular stages or parts of software development, including project management procedures or requirement gathering strategies. The suggested method, on the other hand, provides a thorough solution that may be used at any stage of development. The suggested solution streamlines the process by providing a single recommendation system that takes into account all pertinent factors and recommends the best development style, whether traditional or agile, whereas present systems offer a variety of tools and libraries for developers.

4 Conclusion

This study and the proposed system have made significant contributions to the field of software development technique recommendations. IT developers would find the developed recommendation system helpful in choosing the appropriate software development methodology. The built recommender model will suggest the appropriate approach, either traditional (Waterfall) methodology or agile methodology, for software product development based on requirements, user participation, development team, and project type. Out of five machine learning methods, the Random Forest model and Decision Tree provide 100% accuracy in the results of the recommender model. The proposed system's novelty makes this research strong because no other recommender model has been developed to aid in the selection of a software development approach. Instead, recommender systems have been developed for requirement elicitation techniques and development phases, not for the purpose of selecting a development approach.

Software development projects with clearly specified requirements at the beginning of the project are best suited to the traditional waterfall approach. Projects with strict deadlines and future changes don't prefer traditional methodology. This is frequently used for smaller-scale projects where the manufacturing of reusable components is not required and risk management and analysis are not the main focus. It's crucial to remember that, in comparison to alternative approaches, the traditional strategy may come at a higher cost. An agile software development approach is used for large-scale projects where all the requirements are not stated clearly at the start of the project and there may be various changes possible during the development process because of different situations. This approach is applicable for quick delivery of working versions of the product's sprints and focuses on the reusability of the components of the product. The development team needs to be experts, and the time and cost of development are less with this approach. Large-scale projects with unclear or changing needs are better suited for the agile software development methodology, which promotes flexibility and adaptation all along during development. In addition, compared to conventional methods, agile approaches frequently yield shorter development times and lower development costs.

References

1) Mishra A, Alzoubi YI. Structured software development versus agile software development: a comparative analysis. *International Journal of System Assurance Engineering and Management*. 2023;14(4):1504–1522. Available from: https://dx.doi.org/10.1007/s13198-023-01958-5.
2) Edison H, Wang X, Conboy K. Comparing Methods for Large-Scale Agile Software Development: A Systematic Literature Review. *IEEE Transactions on Software Engineering*. 2022;48(8):2709–2731. Available from: https://dx.doi.org/10.1109/tse.2021.3069039.
3) Behrens A, Ofori M, Noteboom C, Bishop D. A systematic literature review: how agile is agile project management? *Issues In Information Systems*. 2021;22(3):278–295. Available from: https://iacis.org/iis/2021/3_iis_2021_298-316.pdf.

4) Akram F, Ahmad T, Sadiq M. Recommendation systems-based software requirements elicitation process—a systematic literature review. *Journal of Engineering and Applied Science*. 2024;71(1):1–21. Available from: https://dx.doi.org/10.1186/s44147-024-00363-4.
5) Rocco JD, Ruscio DD, Sipio CD, Nguyen PT, Rubei R. Development of recommendation systems for software engineering: the CROSSMINER experience. *Empirical Software Engineering*. 2021;26(4):1–40. Available from: https://dx.doi.org/10.1007/s10664-021-09963-7.
6) Herawati S, Negara YDP, Febriansyah HF, Fatah DA. Application of the Waterfall Method on a Web-Based Job Training Management Information System at Trunojoyo University Madura. In: International Conference on Science and Technology (ICST 2021);vol. 328 of E3S Web of Conferences. EDP Sciences. 2021;p. 1–6. Available from: https://dx.doi.org/10.1051/e3sconf/202132804026.
7) Bianchi MJ, Conforto EC, Rebentisch E, Amaral DC, Rezende SO, de Pádua R. Recommendation of Project Management Practices: A Contribution to Hybrid Models. *IEEE Transactions on Engineering Management*. 2022;69(6):3558 –3571. Available from: https://doi.org/10.1109/TEM.2021.3101179.
8) Barata SFPG, Ferreira FAF, Carayannis EG, Ferreira JJM. Determinants of E-Commerce, Artificial Intelligence, and Agile Methods in Small- and Medium-Sized Enterprises. *IEEE Transactions on Engineering Management*. 2023;71:6903 –6917. Available from: https://doi.org/10.1109/TEM.2023.3269601.
9) Sankhe P, and MD. A Review and Survey of Software Development Methodologies. *International Journal of Creative Reasearch Thoughts*. 2023;11(5):e726–e733. Available from: https://ijcrt.org/papers/IJCRT2305598.pdf.
10) Sankhe P, Mathur S, Rehman TB, Dixit M. Review of an Agile Software Development Methodology with SCRUM & Extreme Programming. In: 2022 IEEE International Conference on Current Development in Engineering and Technology (CCET). IEEE. 2023;p. 5415–5420. Available from: https://doi.org/10.1109/CCET56606.2022.10080640.
11) Bianchi M, Marzi G, Dabic M. Guest Editorial: Agile Beyond Software—In Search of Flexibility in a Wide Range of Innovation Projects and Industries. *IEEE Transactions on Engineering Management*. 2022;69(6):3454–3458. Available from: https://dx.doi.org/10.1109/tem.2022.3206408.
12) Kuhrmann M, Tell P, Hebig R, Klunder J, Munch J, Linssen O, et al. What Makes Agile Software Development Agile? *IEEE Transactions on Software Engineering*. 2022;48(9):3523–3539. Available from: https://dx.doi.org/10.1109/tse.2021.3099532.
13) Kasauli R, Knauss E, Horkoff J, Liebel G, de Oliveira Neto FG. Requirements engineering challenges and practices in large-scale agile system development. *Journal of Systems and Software*. 2021;172:1–26. Available from: https://dx.doi.org/10.1016/j.jss.2020.110851.
14) Ghimire D, Charters S. The Impact of Agile Development Practices on Project Outcomes. *Software*. 2022;1(3):265–275. Available from: https://dx.doi.org/10.3390/software1030012.
15) Gurung G, Shah R, Jaiswal DP. Software Development Life Cycle Models-A Comparative Study. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*. 2020;6(4):30–37. Available from: https://dx.doi.org/10.32628/cseit206410.
16) Lunesu MI, Tonelli R, Marchesi L, Marchesi M. Assessing the Risk of Software Development in Agile Methodologies Using Simulation. *IEEE Access*. 2021;9:134240 –134258. Available from: https://doi.org/10.1109/ACCESS.2021.3115941.
17) Hanslo R, Tanner M. Machine Learning models to predict Agile Methodology adoption. In: Proceedings of the 2020 Federated Conference on Computer Science and Information Systems. IEEE. 2020;p. 697–704. Available from: https://doi.org/10.15439/2020F214.
18) Jain H, Khunteta A, Srivastava S. Churn Prediction in Telecommunication using Logistic Regression and Logit Boost. *Procedia Computer Science*. 2020;167:101–112. Available from: https://dx.doi.org/10.1016/j.procs.2020.03.187.
19) Suyal M, Goyal P. A Review on Analysis of K-Nearest Neighbor Classification Machine Learning Algorithms based on Supervised Learning. *International Journal of Engineering Trends and Technology*. 2022;70(7):43–48. Available from: https://dx.doi.org/10.14445/22315381/ijett-v70i7p205.
20) Gadekallu TR, Khare N, Bhattacharya S, Singh S, Maddikunta PKR, Ra IH, et al. Early Detection of Diabetic Retinopathy Using PCA-Firefly Based Deep Learning Model. *Electronics*. 2020;9(2):1–16. Available from: https://dx.doi.org/10.3390/electronics9020274.
21) Mrva J, Neupauer S, Hudec L, Sevcech J, Kapec P. Decision Support in Medical Data Using 3D Decision Tree Visualisation. In: 2019 E-Health and Bioengineering Conference (EHB). IEEE. 2020;p. 1–4. Available from: https://doi.org/10.1109/EHB47216.2019.8969926.
22) Sarker IH, Kayes ASM, Watters P. Effectiveness analysis of machine learning classification models for predicting personalized context-aware smartphone usage. *Journal of Big Data*. 2019;6(1):1–28. Available from: https://dx.doi.org/10.1186/s40537-019-0219-y.
23) Jalal N, Mehmood A, Choi GS, Ashraf I. A novel improved random forest for text classification using feature ranking and optimal number of trees. *Journal of King Saud University - Computer and Information Sciences*. 2022;34(6, Part A):2733–2742. Available from: https://dx.doi.org/10.1016/j.jksuci.2022.03.012.
24) Wei L, Capretz LF. Recommender Systems for Software Project Managers. In: Proceedings of the 25th International Conference on Evaluation and Assessment in Software Engineering. ACM. 2021;p. 412–417. Available from: https://doi.org/10.1145/3463274.3463951.