IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, August 2026
Impact Factor: 8.2

A Machine Learning-Based Peer Tutoring Web Application for Intelligent Tutor–Student Matching

Fatade Oluwayemisi Boye[1], Ukwute Chukwudumebi Nkemakolem[2], Towolawi Ayomide Fadilulahi[3], Pinwa Joshua Nwibari[4]
[1234]Department of Computer Science, School of Computing,
Babcock University, Ilisan-Remo, Ogun State, Nigeria
[1]fatadeo@babcock.edu.ng

Abstract: *Peer tutoring is a widely recognized strategy for improving academic performance, yet most existing platforms rely on manual or rule-based pairing methods that may not fully account for individual learner needs, tutor expertise, and historical interaction outcomes. This study presents the development of a peer tutoring web application based on machine learning algorithms for automatic tutor–student matching using a Random Forest classifier. The application was developed using a modular software architecture including a React.js front end, Flask back-end API, and a PostgreSQL relational database. The matching algorithm was trained on a synthetic dataset of interactions between tutors and students, which included features such as average student improvement rate, tutor rating, compatibility of subjects, experience level, and session outcomes. The classifier's hyperparameters were tuned using GridSearchCV across multiple parameter combinations. Functional evaluation showed that all critical system functionalities, such as user registration, tutor–student matching, session booking, asynchronous messaging, and rating submission, performed as expected. The machine learning algorithm achieved a classification accuracy of 81%, with balanced precision and recall metrics for both classes, exceeding the accuracy of random assignment. These results suggest that supervised learning can support personalized tutor recommendations within a peer tutoring platform.*

Keywords: Peer tutoring, machine learning, Random Forest, recommendation system, educational technology, web application, tutor-student matching.

I. INTRODUCTION

The peer-learning concept has been widely applied in various fields of distance learning. The concept of peer tutoring as a learning strategy of the more knowledgeable peer teaching the less knowledgeable peer was proven to be a successful practice, helping both parties involved to achieve better academic performance, increase collaboration, and strengthen communication skills (Tan et al., 2020; Herrmann-Werner et al., 2022). The COVID-19 pandemic situation prompted universities and educational facilities worldwide to embrace remote services as substitutes for physical interactions (Krause & Moore, 2022; Sanches-Ferreira et al., 2022). The current practice of peer tutoring has also adopted digital platforms to facilitate the interactions between students; however, the main challenge remains the provision of a matching mechanism to ensure that both the tutor and tutee are compatible (Ma et al., 2020; Lechuga & Doroudi, 2023). Most platforms rely on manual mechanisms, administrative involvement, or use filtering mechanisms based on students' selection and subject or year of study rather than a more elaborate matching model.

Although the described advantages of peer tutoring are substantial, the method has one significant disadvantage: a lack of consideration of compatibility and individual characteristics. In most cases, students are assigned to tutors based on manual input, administrative decisions, or simple rules, such as students' preferences or the year of study (Ma et al., 2020; Lechuga & Doroudi, 2023). At the same time, it is clear that the characteristics of students and tutors are

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, August 2026
Impact Factor: 8.2

multifaceted, and their compatibility goes far beyond students' subjective preferences and a few objective factors, such as subject area or the year of study.

Machine learning (ML) offers a data-driven alternative that can learn patterns from historical interaction data and make increasingly accurate matching recommendations over time. Several studies have applied ML techniques to the peer tutoring context, including collaborative filtering (Ma et al., 2020), clustering algorithms (Lechuga & Doroudi, 2023), and multi-agent systems (Gösling et al., 2024). However, the majority of these contributions remain proof-of-concept implementations, often lacking a fully functional web deployment, a comprehensive feature set, or a holistic evaluation of both the software system and its learning model.

This study addresses these gaps by contributing a fully functional, end-to-end peer tutoring web application that integrates a Random Forest-based recommendation model directly into the user-facing platform. The major contributions of this paper are: (i) a modular, scalable web-based system architecture incorporating machine learning algorithms directly into the tutoring process; (ii) a Random Forest classifier built on a synthetic data set with multiple features to classify and rank tutoring sessions' success; (iii) testing of all system components along with the model performance evaluation based on classification metrics; and (iv) analysis of design choices, limitations and future directions of research.

This paper is organized as follows: Section 2 gives the review of related literature; Section 3 provides the design and methodology of the system; Section 4 provides the results and their evaluation; Section 5 presents conclusions, and Section 6 gives suggestions for future work.

II. RELATED WORK

A. Peer Tutoring: Effectiveness and Challenges

Peer tutoring has been extensively studied as a collaborative learning strategy. Herrmann-Werner et al. (2022) conducted a longitudinal study on online student tutorials in medical education, finding that students who participated in peer-led tutorials reported significantly greater satisfaction and motivation compared to those in teacher-led digital sessions. According to Hardt et al. (2023), individual remote peer tutoring in economics courses at the university level led to an increase in the number of credits obtained by 30% in comparison with the control group. Sanchez-Ferreira et al. (2022) arrived at the same conclusion, as online peer tutoring enhanced the reading fluency and accuracy scores of elementary education students during pandemic-related school closings.

A consistent finding across the literature is that the effectiveness of peer tutoring is contingent on the quality of the pairing. Tan et al. (2020) and Achón et al. (2023) both note that manual pairings that ignore compatibility factors such as learning pace, communication style, or subject depth frequently result in suboptimal outcomes and missed learning opportunities.

B. Intelligent Matching Systems

Several researchers have attempted to automate and optimize tutor–student pairing using computational techniques. Ma et al. (2020) developed the Peer Tutoring Recommender System (PTRS), which applied a Random Forest classifier alongside computer vision-based automated assessment to match vocational students with tutors. The system demonstrated measurable improvements in student performance on computer application tasks. Lechuga and Doroudi (2023) compared three grouping algorithms, including K-means clustering and reciprocal matching on data from the ALEKS adaptive learning platform, finding that multi-dimensional groupings significantly outperformed ability-based assignments alone. Gösling et al. (2024) proposed a multi-agent system using the Contract Net Protocol within virtual learning environments, enabling dynamic tutor–tutee pair formation at scale, though at high computational cost and with limited LMS compatibility.

Achón et al. (2023) proposed a diversity-aware recommender application (SOS Tutoris UC), which included tutors' Big Five personality traits and competencies when generating recommended matches. The system successfully integrated the competence prioritization feature, which was well received by users. However, the application faced challenges related to low demand during examination periods and issues with the use of specific personality filters.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, August 2026
Impact Factor: 8.2

C. Web-Based Learning Platforms

The architectural dimension of peer tutoring systems has received growing attention. Pathak et al. (2025) have proposed an enterprise architecture of a multilayered web-based platform for peer-to-peer academic tutoring services to enhance search optimization and accessibility in higher education. Ningsih et al. (2024) have designed an enterprise architecture of online tutoring academic services based on the TOGAF and Zachman frameworks to meet institutional enterprise requirements. Seah et al. (2023) have employed the Design Thinking methodology to re-engineer the "TutorYuk" peer-to-peer tutoring platform and identify ways of improving its commercialization potential.

D. Identified Research Gaps

Despite the breadth of existing research, the following gaps persist: (i) most intelligent matching systems are not embedded within fully functional, deployable web applications; (ii) evaluation is typically limited to algorithmic performance on small or simulated datasets, without assessing system-level functionality in a holistic manner; and (iii) few systems combine ML-based matching with integrated session management, messaging, and feedback tools in a single platform (Gösling et al., 2024; Pathak et al., 2025; Achón et al., 2023). This study directly addresses all three of these gaps.

III. METHODOLOGY

A. System Overview

The proposed system is a full-stack peer tutoring web application designed to connect students seeking academic assistance with peer tutors in a secure, scalable, and intelligent environment. The system supports two primary user roles: tutee and tutor, as well as an administrative role for platform oversight. The functional components of the application include user sign-up and authentication, profile management, machine learning-based tutor recommendation, booking of sessions, asynchronous communication, and a rating and feedback system for the tutors. The system is designed according to the three-tier architectural design, which consists of presentation, application, and database tiers, with the inclusion of a machine learning inference module in the backend.

B. System Architecture

The frontend of the application was configured using React.js and Tailwind CSS. Components were developed using React's component-based approach, while Tailwind CSS was used to support a consistent and user-friendly interface and make the development process faster. The Backend was developed using Flask, a Python Web framework. This allowed the rapid development of the RESTful APIs necessary for the application. Each endpoint catered to a specific feature, namely: authentication, tutor matching, sessions, and messaging. The Database used was PostgreSQL, which served as the application's primary database. It contained the core tables, namely: Users, Tutors, Sessions, Subjects, Feedback, Progress, and Universities. It also enforced the primary and foreign key constraints to maintain referential integrity of the data in the system.

The processes of authentication and session management are carried out via JWT (JSON Web Tokens), making it possible to implement stateless access control on the basis of a role-based principle for the following roles: students, tutors, administrators. All data transfer is secured through HTTPS/SSL. Machine learning functionality is integrated into the Flask backend and is loaded dynamically, only when needed, to generate recommendations through the recommend-with-filters API endpoint. Machine learning functionality is implemented in the Flask backend and is dynamically loaded when required to provide recommendations through the recommend-with-filters API endpoint. This endpoint gets preferences of learners (subject, experience level, meeting format, university), and returns ordered tutors according to ML-calculated scores.

C. Dataset Description

Due to the absence of public datasets specifically oriented towards tutor–student matching in the context of peer tutoring, there was a need to generate a synthetic dataset for this purpose. The dataset was created based on controlled distributions, taking into account the features mentioned in the peer tutoring literature (Ma et al., 2020; Lechuga &

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, August 2026
Impact Factor: 8.2

Doroudi, 2023), and contains records of both tutors and students. The dataset includes several tutor-related features, such as experience level (beginner, intermediate, or expert), average tutor rating on a scale of 1 to 5, number of sessions completed, subject expertise, and availability schedule. Student-related features include pre-session knowledge score, post-session knowledge score, average improvement rate (post-score − pre-score), subject, and engagement. Session-related features include session duration, communication clarity rating, session quality rating, tutor attentiveness, student engagement, and a binary output label (1 for success / 0 for no success). Before training the model, the dataset was pre-processed using the following techniques: label encoding of categorical features (experience level, subject, and meeting mode), min-max normalization of numerical features, and validation of missing values and class balance in the output labels.

D. Machine Learning Model

A Random Forest classifier was selected as the core prediction model. The algorithm was chosen because of its ability to handle heterogeneous data and capture relationships between multiple features. It also provides good performance while being less sensitive to overfitting than individual decision trees. i.e., when both categorical and numeric features are present, its natural robustness to overfitting by virtue of ensemble averaging multiple decision trees, and its proven efficiency in similar problems in educational technology research (Ma et al., 2020). The model calculates the probability of a successful tutoring session for each tutor–student pair, which serves as the primary criterion for ranking tutor recommendations. Hyperparameter tuning was conducted using GridSearchCV with 5-fold cross-validation. The parameters adjusted included the number of trees (estimators), maximum tree depth, and minimum number of samples required to split a node. The optimal combination of parameters was selected based on the highest F1 score obtained during cross-validation. The F1 score was used as the evaluation metric because it considers both precision and recall in a binary classification problem. The result of the model prediction for each possible tutor is a number from the [0, 1] range – the probability of a successful tutoring session together with the compatibility bonus due to the learning style similarity:

final_match_score = ml_probability + learning_style_compatibility_bonus

TABLE 1: SUMMARY OF THE TECHNOLOGY STACK USED IN SYSTEM DEVELOPMENT

COMPONENT	TECHNOLOGY
Frontend	React.js, HTML5, Tailwind CSS
Backend	Python, Flask, RESTful APIs
Database	PostgreSQL
Authentication	JSON Web Tokens (JWT)
ML Framework	scikit-learn (Random Forest, GridSearchCV)
Version Control	Git / GitHub

IV. RESULT

All eight core system functions passed testing without errors, confirming correct implementation of the intended functionality across all user roles and interaction scenarios.

TABLE 2: FUNCTIONAL TESTING RESULTS FOR CORE SYSTEM MODULES

Feature	Test Case	Expected Result	Actual Result	Status
User Registration	Valid user details submitted	Account created	Account created	Pass
User Login	Correct credentials entered	Login successful	Login successful	Pass
Subject Selection	Subject selected from list	Tutors displayed	Tutors displayed	Pass
Tutor Recommendation	Filters applied, ML model invoked	Ranked tutor list returned	Ranked list returned	Pass
Session Booking	Time slot selected and confirmed	Session saved to database	Session booked	Pass
Messaging	Message sent between users	Message delivered	Message delivered	Pass

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, August 2026
Impact Factor: 8.2

Rating Submission	Tutee submits post-session rating	Rating saved and displayed	Rating saved	Pass
Admin Dashboard	Admin logs in and views platform data	User and session data visible	Data visible	Pass

Machine Learning Model Evaluation

The trained Random Forest model was evaluated on a held-out test set derived from the synthetic dataset. Model performance was assessed using standard binary classification metrics: accuracy, precision, recall, and F1-score, reported for both outcome classes and as weighted averages. The class distribution in the test set was near-balanced (No Success: 50.62%; Success: 49.38%), confirming that reported metrics are not inflated by class imbalance artifacts.

TABLE 3: CLASSIFICATION REPORT FOR THE RANDOM FOREST TUTOR-MATCHING MODEL (OVERALL ACCURACY: 81%)

Class	Precision	Recall	F1-Score	Support
No Success (0)	0.81	0.79	0.80	~50%
Success (1)	0.80	0.82	0.81	~50%
Weighted Average	**0.81**	**0.81**	**0.81**	**100%**

The balanced F1-score of 0.81 across both classes indicates that the model performs consistently in identifying successful matches and avoiding false recommendations. This is a meaningful improvement over a random assignment baseline, which would yield approximately 50% accuracy on a balanced binary outcome. The confusion matrix visualization confirmed well-distributed prediction across classes with a slight directional bias toward predicting session success (recall of 0.82 for class 1), which is an acceptable trade-off in a recommendation context where false negatives failing to suggest a compatible tutor carry a higher practical cost than false positives.

V. DISCUSSION

The results of this study confirm that integrating a Random Forest classifier into a peer tutoring web platform is both technically feasible and practically beneficial. The ML-based recommendation approach demonstrated an overall classification accuracy of 81%, significantly outperforming the random assignment baseline. This finding is consistent with Ma et al. (2020), who reported that ML-based peer tutor recommendation produces higher pairing accuracy and greater learner satisfaction than manual or rule-based assignment procedures.

The feedback loop embedded in the system architecture represents a notable design contribution: tutor ratings collected after each session are persisted and subsequently used as input features in future recommendation computations. This design ensures that the quality of recommendations can improve organically over time as more interaction data is accumulated, addressing a limitation common to static recommendation systems (Achón et al., 2023; Gösling et al., 2024). Platforms that adapt to accumulating performance data are better positioned for long-term effectiveness, particularly as the user base scales.

The integration of rule-based pre-filtering with ML scoring represents a pragmatic architectural decision. Rule-based constraints (e.g., minimum rating thresholds, university affiliation preferences) eliminate clearly incompatible tutors before the ML model is invoked, reducing computational load while ensuring basic compatibility requirements are met. The ML layer then applies nuanced, pattern-based scoring to the remaining candidates, a hybrid approach that balances interpretability with predictive sophistication.

The use of a synthetic dataset is an acknowledged limitation of this study. While the dataset was constructed with controlled distributions informed by the literature, it cannot fully capture the complexity and variability of real student interaction data. Consequently, the reported model metrics should be interpreted as indicative of potential performance under idealized conditions. This limitation is consistent with broader challenges in educational data mining research, where institutional data sharing restrictions frequently necessitate synthetic or anonymized datasets (Ningsih et al.,

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 1, August 2026 Impact Factor: 8.2

2024; Lechuga & Doroudi, 2023). Real-world deployment would require retraining the model on live interaction data to achieve validated performance benchmarks.

The asynchronous messaging and session booking features address practical coordination challenges identified as barriers to effective peer tutoring adoption (Tan et al., 2020; Pathak et al., 2025). By consolidating matching, scheduling, communication, and feedback into a single platform, the system reduces the coordination overhead that typically burdens both tutors and students in decentralized peer tutoring arrangements. This integrated design positions the platform as a more holistic solution than systems that address only the algorithmic matching component in isolation.

VI. CONCLUSION

This study presented the design, development, and evaluation of a machine learning-based peer tutoring web application that addresses the limitations of manual and rule-based student pairing. The system integrates a Random Forest classifier into a modular, full-stack web platform built on Flask, React.js, and PostgreSQL, providing tutees with ranked tutor recommendations based on subject compatibility, tutor performance history, and predicted session success probability.

Functional evaluation confirmed that all eight core system modules operate correctly across all user roles. The ML model achieved an overall accuracy of 81% with balanced precision and recall across outcome classes, a significant improvement over the random assignment baseline. The embedded rating-based feedback loop further positions the system for continuous improvement as interaction data accumulates over time.

Future work should prioritize: (i) real-world deployment within an academic institution with live student data and IRB-approved data collection; (ii) a longitudinal quasi-experimental study comparing learning outcomes under ML-based versus manual matching; (iii) integration with institutional Learning Management Systems (LMS) such as Moodle or Canvas; and (iv) exploration of more advanced algorithms, such as hybrid collaborative-content filtering or reinforcement learning, to further refine recommendation quality. The platform's modular architecture positions it well for these extensions, and its alignment with UN Sustainable Development Goals 4 (Quality Education) and 10 (Reduced Inequalities) underscores its potential as a tool for broadening access to effective academic support.

REFERENCES

1. Achón, L., De Souza, A., Hume, A., Chenu-Abente, R., De Götzen, A., & Cernuzzi, L. (2023). SOS Tutoris UC: A diversity-aware application for tutor recommendation based on competence and personality. CEUR Workshop Proceedings, 3456, 243–247.
2. Dixit, R. K., & Yalagi, P. S. (2020). Enhanced Programming Learning Model (EPLM) through Continuous Collaborative Coding (CCC) practice. Journal of Engineering Education Transformations, 33(Special Issue), 509–513.
3. Gösling, H., Dudek, J., Krause, T., & Thomas, O. (2024). Multi-agent-based peer tutoring in virtual learning environments. Proceedings of the 45th International Conference on Information Systems (ICIS 2024), Paper 1474. https://aisel.aisnet.org/icis2024/learnandiscurricula/learnandiscurricula/12
4. Hardt, D., Naggler, M., & Rincke, J. (2023). Tutoring in (online) higher education: Experimental evidence. Economics of Education Review, 92, 102350. https://doi.org/10.1016/j.econedurev.2022.102350
5. Herrmann-Werner, A., Griewatz, J., Zipfel, S., Erschens, R., Kern, N., & Festl-Wietek, T. (2022). Online student tutorials for effective peer teaching in digital times: A longitudinal quantitative study. BMC Medical Education, 22, Article 756. https://doi.org/10.1186/s12909-022-03842-9
6. Krause, A. J., & Moore, S. Y. (2022). Creating an online peer-to-peer mentoring program: Promoting student relationships, engagement, and satisfaction during the era of COVID-19. College Teaching, 70(3), 296–308.
7. Kuo, Y.-C., Yao, C.-B., & Wu, Z.-Y. (2022). Online peer-tutoring for programming languages based on programming ability and teaching skill. Applied Sciences, 12(17), 8513. https://doi.org/10.3390/app12178513
8. Lechuga, C. G., & Doroudi, S. (2023). Three algorithms for grouping students: A bridge between personalized tutoring system data and classroom pedagogy. International Journal of Artificial Intelligence in Education, 33, 843–884. https://doi.org/10.1007/s40593-022-00309-y

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, August 2026
Impact Factor: 8.2

9. Ma, Z. H., Hwang, W. Y., & Shih, T. K. (2020). Effects of a peer tutor recommender system (PTRS) with machine learning and automated assessment on vocational high school students' computer application operating skills. Journal of Computers in Education, 7(3), 435–462.
10. Ningsih, S., Wedha, B. Y., & Sholihati, I. D. (2024). Online tutoring's technological foundation and future prospects: Enterprise architecture development. Journal of Computer Networks, Architecture and High Performance Computing, 6(1), Article 3433. https://doi.org/10.47709/cnahpc.v6i1.3433
11. Pathak, S., Kant, S., Yadav, S., & Nimbhorkar, S. (2025). Peer Tutor – A web based system for peer academic assistance and collaboration. i-manager's Journal of Educational Technology, 22(3), 41. https://doi.org/10.26634/jet.22.3.22242
12. Sanches-Ferreira, M., Santos, M., Alves, S., & Silveira-Maia, M. (2022). Implementing an online peer tutoring intervention to promote reading skills of elementary students: Effects on fluency and accuracy. Frontiers in Education, 7, 983332. https://doi.org/10.3389/feduc.2022.983332
13. Seah, J., Sidabutar, G., Rafael, C., Tjhang, S., Kristan, Y., & Singgalen, Y. (2023). Elevating "TutorYuk" online peer tutoring services through information system: A design thinking framework approach. Journal of Information Systems and Informatics, 5(3), 956–970. https://doi.org/10.51519/journalisi.v5i3.523
14. Tan, S. C., Looi, C. K., & Cheung, Y. L. (2020). Holistic design of a mobile peer tutoring application (MENTOR). In T. Bastiaens & G. Marks (Eds.), Proceedings of Innovate Learning Summit 2020 (pp. 673–684).
15. Tan, S. C., Looi, C. K., Cheung, Y. L., Chung, S. H., Lim, S. J., & Wong, W. H. (2023). Designing and evaluating a mobile peer tutoring application: A cultural historical activity theory approach. Interactive Learning Environments, 31(8), 4806–4817