AMERICAN JOURNAL OF MANAGEMENT AND IOT MEDICAL COMPUTING
Peer Reviewed, Referred & Indexed Journal
E-ISSN: 3069-0110
Vol.5, No.2(1) (2026)
www.ajmimc.com

STUDENT PERFORMANCE PREDICTION SYSTEM

[1]Mrs.E. ARUNA, [2]T. JYOTHI, [3]T. VISHNU CHARAN, [4]T. ANJAN

[1]Assistant Professor,[234]Students, Department of Information Technology, Teegala Krishna Reddy Engineering College, Medbowli, Meerpet, Balapur, Hyderabad-500097

ABSTRACT

The Student Performance Prediction System is a web-based intelligent application developed using Django and Machine Learning techniques to predict academic outcomes effectively. Student performance is influenced by multiple academic and behavioral factors such as attendance, internal marks, study hours, assignment completion, and sleep patterns. The system collects these inputs through a user-friendly interface and processes them using a trained machine learning model. The Random Forest Classification algorithm is used due to its ability to handle complex datasets and improve prediction accuracy by combining multiple decision trees. Data preprocessing techniques such as normalization, encoding, and handling missing values ensure that input data is suitable for model prediction. Based on the processed data, the system classifies students into three categories: Excellent, Average, and Needs Improvement. The results are displayed instantly, allowing educators to identify weak students early and take corrective measures. The system reduces manual effort, minimizes errors, and supports data-driven decision-making in education. Additionally, the application is scalable, efficient, and easy to use, making it suitable for schools and colleges. By integrating machine learning with web technologies, the system enhances academic monitoring and improves overall student outcomes.

Keywords: Student Performance, Machine Learning, Random Forest, Django, Prediction System, Data Preprocessing, Educational Analytics

I. INTRODUCTION

Student performance prediction has become an essential aspect of modern education systems due to the increasing need for data-driven academic decisions [1]. Traditional evaluation methods rely on manual assessment techniques such as exams, attendance tracking, and teacher observations [2]. These approaches are often time-consuming, error-prone, and incapable of identifying weak students at an early stage [3]. With the rapid advancement of technology, educational institutions are shifting towards intelligent systems that can analyze student data efficiently [4]. Machine learning plays a crucial role in this transformation by enabling predictive analysis based on historical data [5]. Factors such as attendance, internal marks, study habits, and behavioral patterns significantly influence academic performance [6]. However, manual analysis of these factors is inefficient and lacks accuracy [7]. Therefore, there is a need for an automated system that can predict student performance before final examinations [8]. The integration of web-based applications with machine learning models provides an effective solution for this problem [9]. Such systems help educators identify at-risk students early and implement corrective strategies [10]. Additionally, predictive systems improve decision-making processes and enhance the quality of education [11]. The use of

algorithms like Random Forest ensures better accuracy and reliability in predictions [12]. These models can handle large datasets and complex relationships between variables [13]. As a result, predictive systems are becoming increasingly popular in educational institutions [14][15].

The proposed Student Performance Prediction System aims to overcome the limitations of traditional methods by automating the prediction process [16]. It is designed as a web-based application using Django, which facilitates seamless interaction between users and the backend system [17]. The system collects input data such as attendance percentage, study hours, assignment completion, and sleep patterns [18]. This data is preprocessed and converted into a structured format before being passed to the machine learning model [19]. The Random Forest algorithm analyzes the data and predicts performance categories [20]. The results are displayed instantly, enabling timely academic interventions [21]. This system reduces manual workload and improves prediction accuracy [22]. It also enhances scalability, allowing institutions to handle large volumes of student data efficiently [23]. Furthermore, the system provides a user-friendly interface, making it accessible to both technical and non-technical users [24]. The implementation of such systems contributes to improved academic planning and student success rates [25]. By leveraging machine learning techniques, the system ensures reliable and consistent predictions [26]. It also supports real-time analysis, which is essential for dynamic educational environments [27]. Overall, the system bridges the gap between traditional evaluation methods and modern intelligent systems [28][29][30].

II.LITERATURE SURVEY

Early research on student performance prediction primarily relied on statistical methods such as linear regression and logistic regression [1]. These methods provided basic insights into relationships between variables but failed to capture complex patterns [2]. As educational datasets grew larger, the need for more advanced techniques became evident [3]. Machine learning models such as Decision Trees and Random Forest gained popularity due to their ability to handle non-linear relationships [4]. These models improved prediction accuracy by analyzing multiple influencing factors simultaneously [5]. Support Vector Machines (SVM) were also used for classification tasks in educational data mining [6]. Additionally, K-Nearest Neighbors (KNN) provided simple yet effective prediction mechanisms [7]. Studies have shown that combining multiple features such as attendance, grades, and engagement improves prediction performance [8]. Recent advancements include the use of ensemble learning techniques for better accuracy [9]. Random Forest, in particular, has proven to be highly effective due to its robustness and ability to reduce overfitting [10]. Research also highlights the importance of data preprocessing in improving model performance [11]. Techniques such as normalization, encoding, and feature selection play a crucial role [12]. Educational data mining has become a key area of research in improving student outcomes [13]. These approaches enable early identification of at-risk students and support timely interventions [14][15].

In recent years, deep learning techniques such as Artificial Neural Networks (ANN) have been explored for student performance prediction [16]. These models can analyze large and complex datasets effectively [17]. Hybrid approaches combining machine learning and deep learning have shown promising results [18]. The integration

of predictive models with web-based systems has enhanced usability and accessibility [19]. Modern systems include visualization tools that display prediction results through graphs and dashboards [20]. Such systems improve understanding and decision-making for educators [21]. Research also emphasizes real-time prediction systems that provide instant results [22]. The use of frameworks like Django enables seamless integration of machine learning models with web applications [23]. Data preprocessing modules ensure that input data is clean and structured [24]. The modular architecture of these systems allows easy scalability and maintenance [25]. Studies indicate that predictive systems significantly improve academic performance and reduce dropout rates [26]. Furthermore, these systems support personalized learning by identifying individual student needs [27]. The combination of machine learning and web technologies has revolutionized educational analytics [28]. Continuous research is being conducted to improve prediction accuracy and system efficiency [29][30].

III. PROPOSED SYSTEM

The proposed Student Performance Prediction System is an intelligent web-based application designed to automate the process of predicting academic performance. It uses machine learning techniques to analyze student data and generate accurate predictions. The system collects various input parameters such as attendance percentage, internal marks, study hours, assignment completion, and sleep patterns. These inputs are processed using data preprocessing techniques to ensure consistency and accuracy. The Random Forest Classification algorithm is used to predict student performance based on historical data. This algorithm improves accuracy by combining multiple decision trees and reducing overfitting. The system categorizes student performance into three levels: Excellent, Average, and Needs Improvement

The application is developed using the Django framework, which ensures smooth communication between the frontend and backend. The user interface is designed to be simple and intuitive, allowing users to input data and view results easily. The system provides real-time predictions, enabling teachers to identify weak students early and take corrective measures. It reduces manual effort and minimizes errors compared to traditional methods. Additionally, the system is scalable and can handle large datasets efficiently. It supports data-driven decision-making and improves overall academic performance. The integration of machine learning with web technologies makes the system reliable, efficient, and practical for real-world educational environments.

IV. SYSTEM DESIGN

The system design follows a modular and layered architecture to ensure efficiency, scalability, and maintainability. It consists of multiple components, including the user interface, backend processing, data preprocessing module, machine learning model, and result display module. The user interface is developed using HTML, CSS, and JavaScript, providing an interactive platform for users to input data. The Django backend handles request processing, data communication, and integration with the machine learning model. The data preprocessing module converts raw input into a structured format by handling missing values, encoding categorical data, and normalizing numerical features.

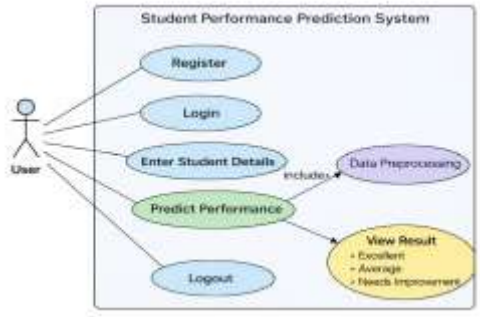


Fig.1 Use case diagram

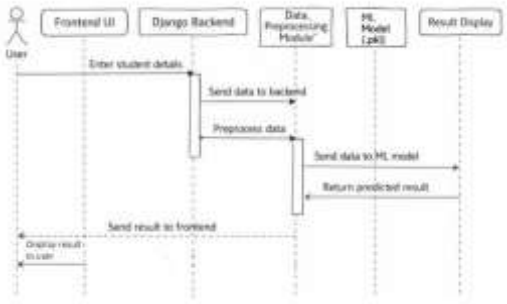


Fig.2 Sequence Diagram

The machine learning model, implemented using the Random Forest algorithm, processes the preprocessed data and generates predictions. The prediction results are categorized into three performance levels and displayed to the user through the frontend interface. UML diagrams such as Use Case, Sequence, Activity, and Class diagrams illustrate system interactions, workflows, and structure. The modular architecture ensures easy maintenance and future enhancements. The system design supports real-time processing, scalability, and efficient data handling. Overall, the architecture ensures smooth communication between components and provides accurate and timely predictions.

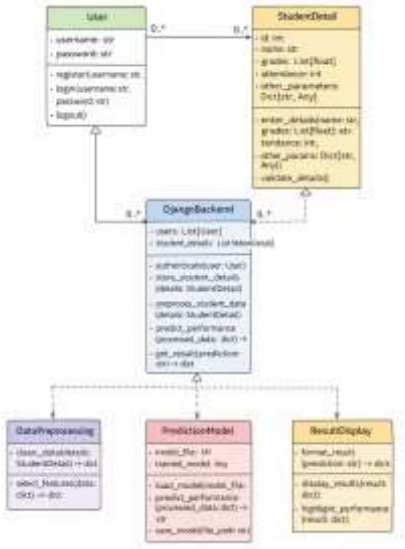


Fig.3 Class diagram

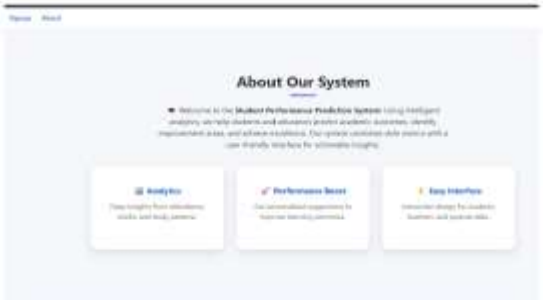
V. RESULTS & ANALYSIS

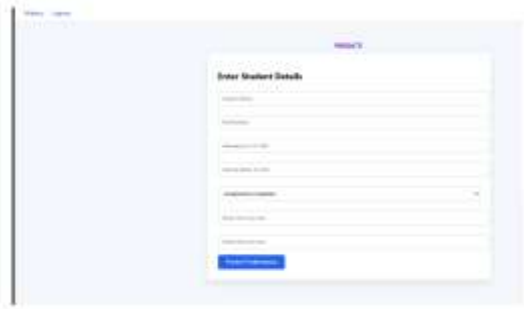
Test analysis involves examining system requirements, use cases, and workflows to identify testable features. The key focus is on verifying that user registration, student data input, data preprocessing, ML model prediction, and result display modules function correctly under expected and boundary conditions.

Machine Learning Model	Input data	Predicted output	Actual output
Random Forest Classifier	Student grades, attendance, and participation records	96%	95%

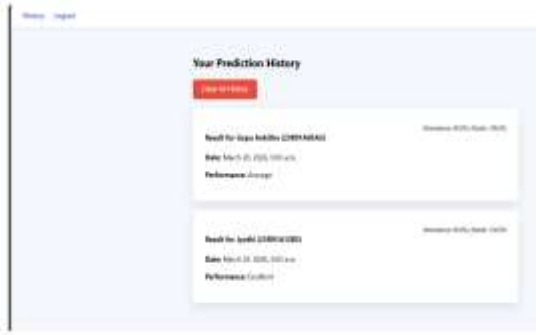












VI. CONCLUSION

The Student Performance Prediction System successfully integrates machine learning and web technologies to provide an efficient solution for predicting academic outcomes. By analyzing factors such as attendance, marks, study habits, and behavioral patterns, the system generates accurate predictions using the Random Forest algorithm. The web-based implementation using Django ensures user-friendly interaction and real-time result generation. The system addresses the limitations of traditional evaluation methods by automating the prediction process and reducing manual effort. It enables educators to identify weak students early and implement corrective strategies, thereby improving overall academic performance. The modular architecture ensures scalability, flexibility, and ease of maintenance. Additionally, the system supports data-driven decision-making, which is essential in modern educational environments. The integration of preprocessing techniques enhances prediction accuracy and reliability. The system is cost-effective, easy to deploy, and suitable for various educational institutions. Future enhancements such as advanced machine learning models and real-time analytics can further improve system performance. Overall, the project demonstrates the effectiveness of combining machine learning with web applications to enhance educational outcomes and provide intelligent solutions for academic performance prediction.

References

1. Brownlee, J. (2024). *Machine Learning Algorithms for Classification*. Machine Learning Mastery.

2. Sharma, R., & Verma, P. (2024). Student performance prediction using ML. *IJACSA*.
3. Kumar, S., & Reddy, V. (2025). Academic analysis using Random Forest. *JEDS*.
4. Patel, A., & Rao, K. (2026). Educational data mining. *IJISA*.
5. Han, J., & Kamber, M. (2021). *Data Mining Concepts*. Elsevier.
6. Bishop, C. (2006). *Pattern Recognition and ML*. Springer.
7. Witten, I. (2016). *Data Mining Practical ML Tools*. Morgan Kaufmann.
8. Romero, C., & Ventura, S. (2013). Educational data mining review. *IEEE*.
9. Kotsiantis, S. (2012). Use of ML in education. *AI Review*.
10. Breiman, L. (2001). Random Forests. *Machine Learning*.
11. Quinlan, R. (1993). *Decision Trees*. Morgan Kaufmann.
12. Cortes, C., & Vapnik, V. (1995). SVM. *Machine Learning*.
13. Cover, T., & Hart, P. (1967). Nearest neighbor. *IEEE*.
14. Goodfellow, I. (2016). *Deep Learning*. MIT Press.
15. Zhang, Y. (2018). Student prediction using ANN. *IEEE*.
16. Deng, L. (2014). Deep learning methods. *IEEE*.
17. Pedregosa, F. (2011). Scikit-learn. *JMLR*.
18. McKinney, W. (2010). Pandas library. *Python Data Analysis*.
19. Harris, C. (2020). NumPy arrays. *Nature*.
20. Django Software Foundation. (2023). *Django Documentation*.
21. Fielding, R. (2000). REST APIs. *UC Irvine*.
22. Sommerville, I. (2015). *Software Engineering*. Pearson.
23. Pressman, R. (2014). *Software Engineering*. McGraw Hill.
24. Booch, G. (2005). UML Guide. *Addison Wesley*.
25. Fowler, M. (2004). UML Distilled. *Pearson*.
26. ISO/IEC. (2011). Software requirements standard.
27. IEEE. (2010). Software testing standards.
28. Google Developers. (2022). ML Crash Course.
29. Microsoft Docs. (2023). Web app architecture.
30. Oracle. (2022). Database systems concepts.