WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Student Performance Prediction Using Machine Learning

Veena K [1], Lakshme Arthee C[2], Lakkireddy Vinay[2], O D Vijay Raju Goud[2]

[1]Assistant Professor, Department of Computer Science and Engineering, R L Jalappa Institute of Technology, Doddaballapura, Bangaluru, Karnataka, India.

[2] Student, Department of Computer Science and Engineering, R L Jalappa Institute of Technology, Doddaballapura, Bangaluru, Karnataka, India.

ABSTRACT—

Data science and machine learning, over the years have proven very well-organized and significant in many sectors including education. Machine learning is an aspect of artificial intelligence in which a computing system can able to learn from data and make conclusions. The recent development in education sector provides assessment tools to predict the student performance by exploring education data using machine learning and data mining techniques. Student performance assessment is an important measurement metrics in education which affects the university accreditation. Student performance improvement plan must be implemented in those universities, by counselling the low performer students. It helps both students and teachers to overcome the problems experienced by the student during studies and teaching techniques of teachers. In this review paper, different student performance prediction literature related to find out low performer student. The survey results indicated that different machine learning techniques are used to overcome the problems related to predicting student at risk and assessment of student performance. Machine learning techniques plays an important role in progress and prediction of student performance, thus improving student performance prediction system.

Keywords— Student Performance Prediction (SPP) · Artificial Intelligence (AI) · Machine Learning (ML)

I. INTRODUCTION

Predicting student academic performance is a critical area of research within educational data mining (EDM). Educational institutions face significant challenges, including managing large student bodies, optimizing resource allocation, reducing dropout rates, and providing high quality, personalized education in an increasingly competitive environment. Early identification of students at risk of underperforming allows institutions to implement timely interventions, such as targeted academic support, counseling, or tailored learning plans, thereby improving student retention and success rates. Furthermore, understanding the factors influencing academic success helps educators refine teaching methodologies and curriculum design. Accurate performance prediction not only benefits individual students by enhancing their learning journey but also contributes positively to institutional reputation and societal outcomes through a better-educated populace.

II. LITERATURE SURVEY

In an era characterized by rapid technological advancements, the field of education is undergoing a transformative shift propelled by the integration of Artificial Intelligence (AI) and Machine Learning (ML) technologies. This paradigm shift offers unprecedented opportunities to revolutionize traditional educational practices, particularly in the realm of student performance analysis.

Cortez and Silva (2008) implemented more advanced models, including Support Vector Machines (SVM), Random Forests, and Neural Networks. These techniques demonstrated higher accuracy and were capable of modelling nonlinear relationships, although they required more computational resources and model tuning.

Bunkar et al. (2016) focused on Logistic Regression and SVM for binary classification of student outcomes. These models were efficient and effective for simple classification tasks but struggled with imbalanced datasets and multi-class prediction scenarios.

Amrieh et al. (2016) used ensemble methods like Bagging and Boosting to improve model performance. Ensemble models generally increased prediction accuracy by combining the strengths of multiple base learners but were more prone to overfitting and required careful parameter tuning.

Hussain et al. (2019) explored Deep Learning models such as Artificial Neural Networks (ANN) and Long Short-Term Memory networks (LSTM) to capture complex patterns in student data. While deep learning achieved state-of-the-art results in terms of prediction accuracy, it also demanded large training datasets and longer training time, and often suffered from a lack of model interpretability.

Ramaswami and Bhaskaran (2010) integrated feature selection methods with Decision Trees to improve model efficiency. Their approach reduced dimensionality and improved processing speed, though it sometimes excluded variables that could have been important for prediction.

Tomasevic et al. (2020) proposed hybrid models that combined Deep Learning with traditional ML algorithms to balance accuracy and explainability. While this approach showed promise, it introduced additional complexity and increased computational cost.

Lastly, Dutt et al. (2021) employed Explainable AI (XAI) techniques to make prediction models more transparent to educators and stakeholders. These methods improved the interpretability of complex models, though occasionally at the expense of prediction accuracy.

Overall, the literature indicates a progression from simple interpretable models to highly accurate yet complex algorithms, with recent emphasis on hybrid and explainable systems to bridge the gap between usability and performance

III. PROPOSED SYSTEM

In contrast to the existing system, the proposed system for student performance analysis leverages Artificial Intelligence (AI) and Machine Learning (ML) techniques to overcome the limitations of traditional methods and enhance predictive accuracy. The proposed system incorporates the following key components:

Advantages:

- **Enhanced Predictive Accuracy:** By harnessing the power of AI and ML, the proposed system improves predictive accuracy and identifies at-risk students more effectively than traditional methods, enabling early interventions and targeted support mechanisms.
- **Data-driven Decision Making:** The proposed system empowers educators and policymakers with actionable insights derived from advanced analytics and predictive modeling, facilitating evidence-based decision-making and resource allocation.
- **Scalability and Efficiency:** Leveraging cloud computing and scalable infrastructure, the proposed system ensures efficient data processing, analysis, and deployment of AI and ML applications, enabling timely interventions and proactive measures to support students' learning needs.
- **Holistic Approach to Student Success:** By integrating disparate data sources and fostering collaboration among stakeholders, the proposed system promotes a holistic approach to student success, addressing socio-economic, academic, and environmental factors influencing educational outcomes.

IV. SYSTEM ARCHITECTURE

he system architecture is structured as a streamlined pipeline that ensures smooth data ingestion, secure storage, and real-time accessibility. The proposed model in this study has four components which are data preprocessing, hyperparameter tuning, recommender model, and model evaluation. How ever, these main components incorporate other elements. The general architecture of the model is presented in Figure 1. First, dataset collection involves collecting the data from the Wollo University learning management system called A+. Next, we utilized three stages of data preprocessing. The data preprocessing consists of data cleaning, categorization, and reduction to make the dataset ready to train the data mining algorithms. Ten, we utilized feature extraction to determine the most informative features. After this, we used hyperparameter tuning for the enhancement of the algorithm. Hyperparameter tuning is used for the automatic enhancement of the hyperparameters of a model. Hyper parameters are all the parameters of a model that are not updated during the learning and are used to configure the algorithm to lower the cost function of the learning rate for the gradient descent algorithm. We apply this to the features which are fed into the algorithm. In this study, hyper parameter tuning is used just to enhance the loop of model learning to find the set of hyper-parameters leading to the lowest error on the validation set. Thus, a validation set has to be set apart, and a loss has to be defined. In this study, we clustered to predict a student's final result based on using various prediction models and choosing the best prediction model. The clustering algorithm used in this study is K-Means. Model building involves developing a wide range of models using prediction methods. Finally, evaluating the model: It in volves testing the validity of the model against each other and the goals of the study. Using the model involves making it a part of the decision-making process.

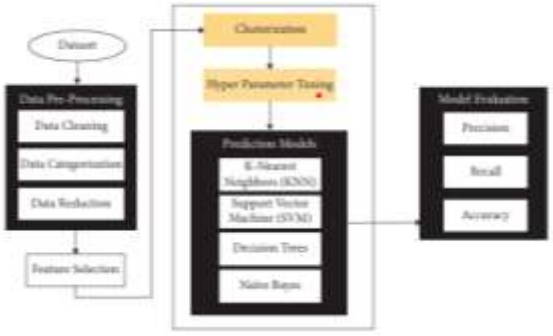

Fig. 1. Methodology

The system follows a layered architecture typical for ML web applications: 1. Data Layer: Manages data storage (new_student_data.csv) and access. 2. Preprocessing Layer: Handles data loading, cleaning (assumed), encoding, scaling (using Pandas, Scikit-learn). 3. Model Training Layer: Includes model definition (RF, GB), hyperparameter tuning (GridSearchCV), cross-validation, training (.fit), evaluation (MSE, R^2), and model saving (using Joblib). 4. Prediction/Application Layer: A Flask web application (run_me1.py) loads the trained model, scaler, and encoder. It receives user input via HTML forms (input.html), preprocesses the input, makes predictions (predict), saves the input and prediction back to the CSV, and displays results (result.html). 5. Presentation Layer: Consists of HTML templates (input.html, login.html, result.html) rendered by Flask to interact with the user.

V. IMPLEMENTATION

1.Data Collection

- Dataset: The primary data source for this project is a Comma-Separated Values (CSV) file named new_student_data.csv. This file is assumed to contain records for individual students.
- Features: The dataset includes the following features used for modelling:
  - Attendance: Student's attendance percentage.
  - Subject2_Marks to Subject9_Marks: Marks obtained in eight different subjects (presumably prerequisites or concurrent subjects).
  - ExtraCurricular_Participation: Level of participation in extracurricular activities (Categorical: Low, Medium, High mapped numerically).
  - Behaviour: Observed student behavior (Categorical: Poor, Average, Good mapped numerically).
  - Feedback_From_Teachers: Teacher feedback sentiment (Categorical: Negative, Neutral, Positive mapped numerically).
  - Assignments_Completion: Percentage of assignments completed. O Subject1_Marks: The target variable representing the marks obtained in the subject to be predicted.
  - StudentID: An identifier assigned when new data is added via the web application.
- Source: The origin of the new_student_data.csv is not explicitly defined in the provided code but is likely institutional data or a specifically compiled survey dataset.

2. preprocessing :Data Processing is a crucial step to prepare the raw data for machine learning models. The following steps were implemented using Python libraries Pandas and Scikit-learn: 1. Handling Categorical Features: The features ExtraCurricular_Participation, Behaviour, and Feedback_From_Teachers are categorical. They were first encoded using Scikit learn's Ordinal Encoder. Subsequently, a custom function mapped the resulting ordinal values (assumed to be 0, 1, 2, etc.) to numerical scores (e.g., 0->25, 1->50, 2->75, 3 >100).

This represents domain knowledge or a specific scaling requirement for these features. 2. Feature Scaling: To ensure that features with larger numerical ranges do not dominate the model training process, all predictor features (independent variables) were scaled using Scikit-learn's StandardScaler. This transforms the data to have a mean of 0 and a standard deviation of 1. The scaler was fitted on the training data and used to transform both training and testing data, as well as new input data during prediction. 3. Data Splitting: The dataset was divided into training and testing sets to evaluate the models' generalization performance. 80% of the data was used for training and 20% for testing, using a fixed random_state (42) for reproducibility (train_test_split(X_scaled,Y,test_size=0.2,random_state=42)). 4. Missing Values/Outliers: The provided code does not explicitly show steps for handling missing values (e.g., imputation) or outlier detection/removal. It is assumed that the dataset was relatively clean or these steps were performed prior to the provided scripts.

VI. RESULTS AND DISCUSSION

Performance Metrics

The performance of the final tuned Random Forest and Gradient Boosting models was evaluated on the unseen test dataset using Mean Squared Error (MSE) and R-squared (R^2).

- Mean Squared Error (MSE): Measures the average squared difference between the actual (Subject1_Marks) and predicted values. Lower values indicate better fit. The formula is: $MSE=n1\sum i=1n(yi-y\hat{}i)2$.
- R-squared (R^2): Represents the proportion of the variance in the dependent variable (Subject1_Marks) that is predictable from the independent variables (features). Values range from 0 to 1, with higher values indicating a better fit (1 being a perfect fit). The formula is: $R2=1-\sum i=1n(yi-y\bar{})2\sum i=1n(yi-y\hat{}i)2$. (Note: The specific MSE and R^2 values obtained from running Model.py should be inserted here. The code prints these values. Example:)
- Random Forest: MSE = [Insert Value], R^2 = [Insert Value]

• Gradient Boosting: MSE = [Insert Value], R^2 = [Insert Value] (Metrics like Accuracy, Precision, Recall, F1-score, and ROC curves are primarily used for classification tasks and are not directly applicable to this regression problem of predicting marks.)

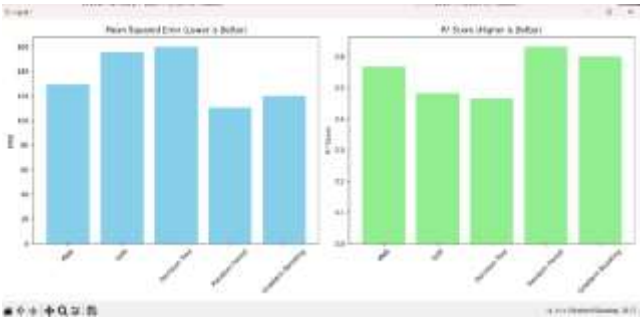

Fig 2: graph representation

VII. CONCLUSION

This paper presents a practical and deployable solution for automating the airport cargo supply chain by integrating Ethereum-based blockchain technology with a real-time cloud backend using Firebase. The system successfully demonstrates how smart contracts can be used for secure, tamper-proof recording of cargo movements, while Firebase ensures fast, accessible, and user-friendly data visualization. The hybrid architecture minimizes manual errors, improves data integrity, and brings greater transparency and traceability to logistics operations.

The implementation validates the effectiveness of combining decentralized and centralized technologies to solve long-standing inefficiencies in airport logistics. With features such as bulk data automation, real-time synchronization, and a responsive frontend, the system offers a foundation for future scalability. Enhancements such as QR code integration, public blockchain deployment, and role-based access control can further evolve the system into a robust, production-grade solution for multi-terminal and multi-airport environments.

Future Work

Several avenues exist for extending and improving this project:

• Feature Engineering: Incorporate additional relevant features, such as detailed LMS engagement data, socio-economic factors, psychological assessments, or more granular behavioral data, if available. Explore interactions between features.

• Dataset Enhancement: Utilize larger and more diverse datasets, potentially spanning multiple academic years or institutions, to improve model generalizability. Address data quality issues like missing values and outliers explicitly.

• Model Exploration: Experiment with other advanced regression algorithms, such as Support Vector Regression (SVR), Neural Networks, or hybrid approaches. Consider ensemble techniques like stacking.

• Interpretability: Implement and analyze feature importance measures (e.g., built-in RF importance, permutation importance) or use model-agnostic techniques like SHAP to better understand why the model makes certain predictions.

• Real-time Integration: Modify the system to handle real-time data streams and provide dynamic predictions as new data becomes available.

• Application Enhancement: Develop the web application further to include user roles (student, teacher, admin), visualizations of predictions and historical trends, and automated alerts or intervention recommendations based on prediction results.

• Longitudinal Analysis: Track student performance over time to predict trajectories rather than single-point outcomes.

REFERENCES

[1] Predicting Student Performance and Enhancing Learning Outcomes: A Data-Driven Approach Using Educational Data Mining Techniques - MDPI \

[2] Enhancing Academic Performance Prediction For At-Risk Students: Comparative Analysis of Machine Learning Algorithms In Early War - Jatit

[3] Predicting Students' Academic Performance Via Machine Learning Algorithms: An Empirical Review and Practical Application - ScholarWorks @ UTRGV

[4] STUDENT PERFORMANCE PREDICTION USING MACHINE LEARNING ALGORITHMS - Journal of Emerging Technologies and Innovative Research

[5] An Intelligent Analytic Framework for Predicting Students Academic Performance Using Multiple Linear Regression and Random Forest - EA Journals

[6] Wang, J., & Yu, Y. (2025). Machine learning approach to student performance prediction of online learning. PLoSONE,20(1),e0299018.https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0299018

[7] lbreiki, B.,Zaki, N & Alashwal, H. (2021). A Systematic Literature Review of Student' Performance Prediction Using Machine Learning Techniques. EducationSciences,11(9),552.https://www.mdpi.com/2227-7102/11/9/552

[8] Ahsan, M. M., Siddique, Z., & Islam, M. R. (2022). Proposed architecture for student performance prediction.ResearchGate.https://www.researchgate.net/figure/Proposed-architecture-for-student-performance prediction_fig2_354325406

[9] Python Software Foundation. Python Language Reference. Available at https://www.python.org

[10] McKinney, W. (2010). Data Structures for Statistical Computing in Python. Proceedings of the 9th Python in Science Conference, 56-61. (Pandas)