Edupredict: Explainable AI-Based Student Performance Prediction

Dr. S Gnanapriya
Associate Professor, Department of Computer Applications, Nehru College of Management, Coimbatore, Tamil Nadu, India
Email:ncmdrsgnanapriya@nehrucolleges.com

Adithya kiran P S
Student, II MCA, Nehru College of Management, Coimbatore, Tamil Nadu, India
Email:kiranadithyaps@gmail.com

Abstract
Identifying students who may experience academic difficulty at an early stage is an important task for educational institutions. Traditional review often depends on periodic examinations and manual interpretation of marks, attendance and student-related information. This paper presents EduPredict, an explainable artificial intelligence-based student performance prediction system designed to estimate a student's academic outcome as Pass or Fail and provide information that helps educators understand the prediction. The project uses a student-score dataset containing subject marks and selected contextual attributes, including gender, part-time employment, extracurricular participation, career aspiration, absences and weekly self-study hours. Direct identifiers such as student ID, first name, last name and email are excluded from model inputs. Categorical attributes are converted into numerical representations, and a binary target is generated using the project rule: a student is labelled Pass when the score is at least 40 in six or more of the seven subjects; otherwise, the label is Fail. A Random Forest classifier with 100 decision trees is trained using an 80:20 train–test split. The trained model and feature-column information are saved for use by a Streamlit application. SHAP is used to explain feature contributions for individual predictions. The application supports single-student prediction, batch CSV prediction, visual summaries, and storage of user and prediction records through SQLite. The current training script creates a test partition but does not calculate or print evaluation metrics; therefore, no numerical accuracy result is claimed in this paper. The system is intended as a decision-support prototype that can help educators review students who may need assistance, rather than as a replacement for professional judgement.

Index Terms—Student performance prediction, explainable artificial intelligence, Random Forest, SHAP, machine learning, educational data mining, Streamlit, academic analytics.

I. INTRODUCTION

Educational institutions collect different types of information about students, including examination marks, attendance, study habits and participation in academic or extracurricular activities. These details can provide useful signals about a student's learning progress. However, reviewing a large number of student records manually can be time-consuming, and students who are experiencing difficulties may not always be identified early.

Machine learning can analyse patterns in historical student data and estimate likely academic outcomes. A classification model can learn relationships between input attributes and an outcome label, such as Pass or Fail. Such predictions may help teachers decide which students require closer review, additional learning support or counselling. A prediction is not a certainty; it is an estimate based on the data used to train the model.

A further challenge is explainability. A prediction that only displays Pass or Fail may not provide enough information for a teacher to understand the factors influencing the result. Explainable AI methods such as SHAP estimate how individual input features contribute to a model output. This makes the prediction easier to inspect and discuss, while not establishing that a feature causes the outcome.

EduPredict combines a Random Forest classifier, SHAP-based explanations and an interactive Streamlit interface. The application accepts individual student details or a CSV file containing multiple student records. It presents the predicted class and visual information, while SQLite stores application users and prediction history.

The main contributions of the project are: (1) a data-processing workflow that removes direct identifiers and prepares student attributes for classification; (2) a Random Forest model that predicts a binary academic outcome; (3) feature-level explanations using SHAP; and (4) a dashboard that supports individual and batch prediction with persistent application records.

The remainder of this paper is organised as follows. Section II discusses related concepts and prior research directions. Section III describes the dataset and methodology. Section IV explains the system architecture and implementation. Section V presents the evaluation status and limitations. Section VI concludes the paper and identifies future improvements.

II. RELATED WORK

Educational data mining studies the use of data analysis and machine learning to understand educational processes and student learning behaviour. Student performance prediction is one of its common applications. Researchers have explored classification techniques using academic records, attendance, demographic information and behavioural indicators to estimate student outcomes.

Traditional statistical approaches, including logistic regression, can provide a relatively simple way to model binary outcomes. Decision trees represent decisions through a sequence of feature-based rules. Ensemble approaches combine multiple models to improve predictive robustness. Random Forest, introduced by Breiman [1], constructs an ensemble of decision trees using sampled training data and feature subsets, and combines their outputs to produce a classification.

Although ensemble models can capture complex relationships, their internal decision process may be difficult for non-technical users to interpret. Explainable AI addresses this concern by providing information about how model inputs influence a prediction. SHAP, based on Shapley values from cooperative game theory, assigns feature contributions to individual predictions [2]. These contributions explain the behaviour of the trained model for a given input; they should not be interpreted as proof of cause and effect.

The EduPredict project combines predictive modelling with an explanation layer and a user-facing application. Its focus is not only to generate an academic outcome label but also to present supporting information in a form that can be reviewed by teachers or administrators.

III. PROPOSED METHODOLOGY

A. Dataset Description

The project reads student records from a CSV file named student_scores_dataset.csv. The dataset contains seven subject-score attributes—Mathematics, History, Physics, Chemistry, Biology, English and Geography—along with contextual attributes. The contextual fields identified in the project include gender, part-time job, extracurricular activities, career aspiration, absences and weekly self-study hours. The dataset record count and exact value ranges should be added from the final dataset used for the submitted experiment.

Attribute group	Fields used in the project	Purpose
Academic attributes	Math, History, Physics, Chemistry, Biology, English, Geography	Represent subject-wise academic performance
Contextual attributes	Gender, part-time job, extracurricular activities, career aspiration	Represent selected student context
Behavioural attributes	Absences, weekly self-study hours	Represent attendance and study behaviour
Target	Pass / Fail	Classification output generated from subject scores
Excluded identifiers	ID, first name, last name, email	Not used as predictive model features

B. Target Label Generation

The project creates a binary target label from the seven subject scores. A student is labelled Pass when the student obtains a score of at least 40 in six or more subjects. If this condition is not satisfied, the student is labelled Fail. This rule converts the original subject-wise marks into a classification target that can be learned by the model. The rule is a project-defined labelling condition and should not be treated as a universal academic promotion policy.

C. Data Preprocessing

Before model training, the CSV data is loaded into a Pandas DataFrame. Direct identifying columns such as ID, first name, last name and email are removed so that they are not used as model predictors. Categorical columns are converted into

numeric values using LabelEncoder. The fitted encoders are saved so that the same mappings can be applied when new student details are submitted through the application. The feature-column order is also saved to help ensure that prediction inputs match the model's training schema.

Consistent preprocessing is important. If a categorical value is encoded differently during prediction, or if the input columns are supplied in a different order, the model may receive data that does not match its training representation. EduPredict therefore stores model-related preprocessing information for reuse in the application.

D. Random Forest Classification

The project uses RandomForestClassifier from the Scikit-learn library. Random Forest is an ensemble learning algorithm that trains multiple decision trees and combines their predictions. In this implementation, the classifier is configured with 100 trees and random_state=42. The final class is selected by combining the outputs of the trees. This approach can model non-linear relationships among academic and contextual attributes without requiring the user to specify a fixed mathematical relationship between each input and the outcome.

The dataset is divided into two parts using an 80:20 train–test split. Eighty percent of the records are used to train the classifier, while the remaining twenty percent are reserved as a test partition. The training process is performed separately from the live Streamlit interface. The trained classifier is saved as model.pkl, and the expected model columns are stored in model_columns.pkl for use during inference.

E. Explainability Using SHAP

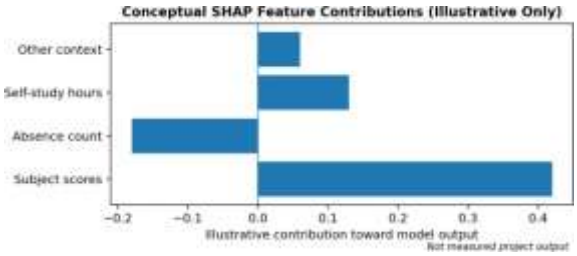


Fig. 3. Illustrative SHAP contribution plot; values are conceptual, not experimental results.

SHAP (SHapley Additive exPlanations) is used to interpret the model output. It estimates the contribution of each input feature to a particular prediction relative to a reference or baseline output. For example, subject scores, absence count and weekly self-study hours may contribute differently to the predicted Pass or Fail class for different student records. The application can display SHAP-based visual explanations so that users can inspect which features influenced the model output.

SHAP explanations describe the model's learned behaviour; they do not prove that changing one feature will necessarily change a student's result. Educators should consider the explanation alongside the student's circumstances and other relevant evidence.

F. Evaluation Protocol

A proper evaluation should use the held-out test partition and report classification metrics such as accuracy, precision, recall, F1-score and a confusion matrix. Recall for the Fail class is particularly useful to examine because it indicates how many students labelled as Fail by the test data are correctly identified by the model. Precision helps assess how many students flagged as Fail are actually Fail in the labelled test set.

The available project description confirms that the training script creates an 80:20 split and trains the Random Forest model, but it does not calculate or print test metrics. A 96% accuracy statement appears in the project README, but it has not been independently verified from the training script. Consequently, this paper does not report 96% as a measured experimental result. The final submission should include the actual metrics after running evaluation on the held-out test data.

IV. SYSTEM ARCHITECTURE AND IMPLEMENTATION

A. Overall Architecture

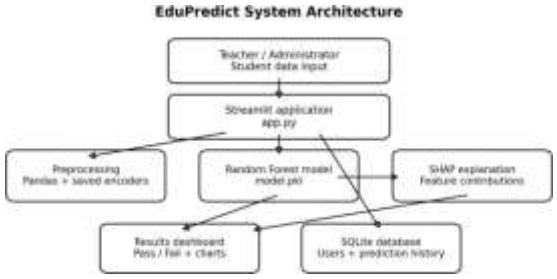


Fig. 1. Overall architecture of the EduPredict application.

EduPredict follows a layered application structure. The Streamlit presentation layer collects user input and displays results. The application logic validates and prepares the input. The preprocessing component applies the saved categorical encoders and arranges the features in the expected order. The machine-learning layer loads the saved Random Forest model and produces the prediction. The explainability layer generates SHAP values, and the output layer displays the result and visual summaries. SQLite provides persistent storage for user accounts and prediction history.

Layer / component	Technology or artifact	Responsibility
Presentation layer	Streamlit; app.py	Navigation, forms, file upload and result display
Data processing	Pandas, NumPy, saved encoders	Cleaning, encoding and feature alignment
Prediction layer	Scikit-learn; model.pkl	Random Forest Pass/Fail classification
Explainability layer	SHAP	Feature contribution explanations
Visualization	Matplotlib	Subject-score and explanation plots
Persistence layer	SQLite; school_system.db	User accounts and prediction history
Model artifacts	model.pkl; model_columns.pkl; encoder files	Reuse trained model and preprocessing schema

B. Application Modules

The Streamlit application provides the following major modules:

- Single-student prediction: allows a user to enter one student's academic and contextual details, then displays a Pass/Fail prediction, probability information, subject-score visualization and SHAP explanation.
- Batch prediction: accepts a CSV file containing multiple student records, processes the records and provides a downloadable output file containing predictions.
- Admin panel: provides options to view and filter saved prediction records, download filtered data, manage user accounts and clear non-admin user records.
- User and database management: stores account information and prediction history in SQLite. Database utility files support table initialization and schema migration.

V. Database Design

The application uses SQLite through a database file named school_system.db. The database layer is separated from the main Streamlit interface. The Users table stores account-related information, including role and subject information. The Predictions table stores prediction records so that they can be reviewed and filtered later. Database initialization and migration utilities create the required tables and update the schema when the application structure changes.

VI. Technology Stack

Technology	Role in EduPredict
Python	Main programming language
Streamlit	Interactive web application and dashboard
Pandas / NumPy	Data loading, tabular processing and numerical operations
Scikit-learn	Label encoding and Random Forest classification
SHAP	Explainable AI and feature attribution
Matplotlib	Charts and visual explanations
SQLite	Persistent user and prediction records
Joblib / Pickle	Saving and loading trained model and related artifacts

VII. Prediction Workflow



Fig. 2. Data preparation, model training and prediction workflow.

1. The user opens the Streamlit application and selects single prediction or batch prediction.
2. The application collects individual student details or reads the uploaded CSV file.
3. The input data is checked and transformed using the saved encoders and expected feature-column order.
4. The saved Random Forest model predicts the Pass or Fail class.
5. SHAP computes feature contributions for the prediction where the explanation view is requested.
6. The application displays the prediction, relevant charts and explanation to the user.
7. Prediction information can be stored in SQLite for later review; batch results can be downloaded as a CSV file.

VIII. RESULTS, DISCUSSION AND LIMITATIONS

A. Implemented Functionality

The project provides an end-to-end prototype that connects data preparation, model training, saved model inference, explanation and a web-based interface. The current application design supports individual and batch prediction, visual output, SHAP explanations and SQLite-backed records. These functions demonstrate how a machine-learning model can be integrated into an interactive academic analytics workflow.

B. Experimental Results Status

The implementation uses a Random Forest classifier with 100 trees and an 80:20 train–test split. However, the training script described in the project materials does not calculate or print accuracy, precision, recall, F1-score or a confusion matrix. Therefore, a verified quantitative performance comparison cannot be presented from the available implementation details. The model should be evaluated on the held-out test set before any numerical performance claim is included.

Evaluation item	Status in the available implementation
Train–test split	80% training and 20% testing
Classifier	RandomForestClassifier, 100 trees, random_state=42
Accuracy / precision / recall / F1	Not calculated or printed by the described training script
Confusion matrix	Should be generated during formal evaluation
README accuracy claim	96% stated in README; not independently verified

C. Limitations

- The project materials do not provide a verified set of test metrics, so predictive performance cannot yet be quantified reliably.
- The prediction target is generated using a fixed rule based on seven subject scores; other institutional grading and pass criteria may differ.
- The quality and representativeness of predictions depend on the dataset size, balance, accuracy and coverage of student records.
- SHAP explains the model output but does not establish causal relationships between student behaviour and academic performance.
- The current registration workflow allows a user to select an admin role. A production deployment should restrict administrator privileges to authorized personnel.
- Uploaded CSV files should be validated for required columns, data types, missing values and reasonable value ranges before processing.
- Student records are sensitive educational information. A real deployment requires appropriate consent, access control, secure storage and institutional privacy review.

D. Responsible Use

EduPredict should be used as a decision-support tool. A Fail prediction should prompt a supportive review of the student's learning needs rather than an automatic penalty or restriction. Teachers and administrators should consider contextual factors, verify the underlying data and make final decisions through human judgement. The model should be monitored for errors and unequal performance across student groups before it is used in consequential settings.

IX. CONCLUSION AND FUTURE WORK

This paper presented EduPredict, an explainable AI-based student performance prediction prototype that estimates Pass or Fail outcomes from academic and selected contextual attributes. The system uses a Random Forest classifier configured with 100 trees, an 80:20 train–test split, saved preprocessing artifacts and SHAP-based feature explanations. A Streamlit interface supports single-student and batch prediction, while SQLite stores user accounts and prediction history.

The project demonstrates an integrated workflow for educational data processing, machine-learning inference and explainable output. However, the current training script does not report test-set metrics, and the README accuracy claim requires independent verification. Quantitative evaluation is therefore an essential next step.

Future work includes calculating accuracy, precision, recall, F1-score and a confusion matrix; applying cross-validation; checking class imbalance; validating uploaded data; strengthening account and administrator security; and conducting fairness and privacy reviews. Further work may also explore model comparison and additional explanation views after the current implementation has been evaluated. With these improvements, EduPredict can become a more dependable tool for helping educators identify students who may benefit from timely academic support.

REFERENCES

[1] L. Breiman, "Random Forests," Machine Learning, vol. 45, pp. 5–32, 2001.

[2] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in Advances in Neural Information Processing Systems, vol. 30, 2017.

[3] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[4] W. McKinney, "Data Structures for Statistical Computing in Python," in Proceedings of the 9th Python in Science Conference, pp. 56–61, 2010.

[5] Streamlit, "Streamlit Documentation." Available: https://docs.streamlit.io/. Accessed: Nov. 19, 2026.

[6] SHAP, "SHAP Documentation." Available: https://shap.readthedocs.io/. Accessed: Nov. 19, 2026.

[7] Scikit-learn, "RandomForestClassifier Documentation." Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. Accessed: Nov. 19, 2026.

8] C. Romero and S. Ventura, "Educational data mining: A review of the state of the art," IEEE Trans. Syst., Man, Cybern. C, vol. 40, no. 6, pp. 601–618, Nov. 2010.

[9] R. S. Baker and K. Yacef, "The state of educational data mining in 2009: A review and future visions," J. Educ. Data Mining, vol. 1, no. 1, pp. 3–17, 2009.

[10] A. Peña-Ayala, "Educational data mining: A survey and a data mining-based analysis of recent works," Expert Syst. Appl., vol. 41, no. 4, pp. 1432–1462, 2014.

[11] A. M. Shahiri, W. Husain, and N. A. Rashid, "A review on predicting student's performance using data mining techniques," Procedia Comput. Sci., vol. 72, pp. 414–422, 2015.

[12] P. Cortez and A. Silva, "Using data mining to predict secondary school student performance," in Proc. 5th Future Business Technology Conf. (FUBUTEC), Porto, Portugal, 2008, pp. 5–12.

[13] C. Romero, M.-I. López, J.-M. Luna, and S. Ventura, "Predicting students' final performance from participation in on-line discussion forums," Comput. Educ., vol. 68, pp. 458–472, 2013.

[14] A. Hellas et al., "Predicting academic performance: A systematic literature review," in Proc. ITiCSE 2018 Companion, Larnaca, Cyprus, 2018, pp. 175–199.

[15] E. A. Alyahyan and D. Düştegör, "Predicting academic success in higher education: Literature review and best practices," Int. J. Educ. Technol. High. Educ., vol. 17, art. 3, 2020.

[16] G. Siemens and R. S. J. d. Baker, "Learning analytics and educational data mining: Towards communication and collaboration," in Proc. 2nd Int. Conf. Learning Analytics and Knowledge (LAK), Vancouver, Canada, 2012, pp. 252–254.

[17] J. Kuzilek, M. Hlosta, and Z. Zdrahal, "Open University Learning Analytics dataset," Sci. Data, vol. 4, art. 170171, 2017.

Explainable AI and SHAP (Section III-E, Related Work)

[18] S. M. Lundberg et al., "From local explanations to global understanding with explainable AI for trees," Nature Mach. Intell., vol. 2, no. 1, pp. 56–67, 2020.

[19] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?': Explaining the predictions of any classifier," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 1135–1144.

[20] A. B. Arrieta et al., "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," Inf. Fusion, vol. 58, pp. 82–115, 2020.

[21] R. Guidotti et al., "A survey of methods for explaining black box models," ACM Comput. Surv., vol. 51, no. 5, art. 93, 2018.

[22] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," arXiv:1702.08608, 2017.

[23] C. Molnar, Interpretable Machine Learning, 2nd ed., 2022. [Online]. Available: https://christophm.github.io/interpretable-ml-book/

Machine learning methods (Section III-D, Future Work on model comparison)

[24] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning, 2nd ed. New York, NY, USA: Springer, 2009.

[25] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 785–794.

[26] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," J. Artif. Intell. Res., vol. 16, pp. 321–357, 2002.

Evaluation metrics (Section III-F, Future Work)

[27] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," Inf. Process. Manage., vol. 45, no. 4, pp. 427–437, 2009.

[28] T. Fawcett, "An introduction to ROC analysis," Pattern Recognit. Lett., vol. 27, no. 8, pp. 861–874, 2006.

[29] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in Proc. 14th Int. Joint Conf. Artificial Intelligence (IJCAI), 1995, vol. 2, pp. 1137–1143.

Fairness and responsible use (Section V-C and V-D)

[30] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," ACM Comput. Surv., vol. 54, no. 6, art. 115, 2021.

Software tools (Section IV-D)

[31] C. R. Harris et al., "Array programming with NumPy," Nature, vol. 585, pp. 357–362, 2020.

[32] J. D. Hunter, "Matplotlib: A 2D graphics environment," Comput. Sci. Eng., vol. 9, no. 3, pp. 90–95, 2007.

AUTHOR CHECKLIST BEFORE FINAL SUBMISSION

The following details were not available in the project materials reviewed for this draft. Complete them from the final code, dataset and college submission requirements; do not estimate or invent values.

- Insert the student author name, guide name (if required), and correct institutional email address.
- Confirm the exact dataset source, number of records, column names and any class distribution.
- Run evaluation on the held-out test set and add accuracy, precision, recall, F1-score and confusion matrix.
- Verify the README accuracy statement against a reproducible evaluation run before reporting it.
- Confirm the exact filenames of all saved encoders and model artifacts from the final project folder.
- Check the final application screenshots and database field names against the submitted source code.
- Format the final manuscript according to the journal or college template, including page limits and reference style.

Drafting note: This manuscript follows the structure and technical-paper tone of the supplied IEEE-style house-price paper. Project-specific facts are based on the available EduPredict project presentation and prior project details. Unknown experimental values have intentionally not been fabricated.