

A Machine Learning-Based Predictive Framework for Proactive Early Academic Intervention in Higher Education

J. Angelin Jeba^{1,*}, S. Rubin Bose², Vanessa M. Willemse³, Ennoui Assia⁴, Sureshkumar Somayajula⁵,
A. Shyam Sundhar⁶

¹Departement of Electronics and Communication Engineering, S.A. Engineering College, Chennai, Tamil Nadu, India.

²School of Computer Science and Engineering, SRM Institute of Science and Technology, Ramapuram,
Chennai, Tamil Nadu, India.

³Department of Education, Northern Cape Department of Education, Kimberley, Northern Cape, South Africa.

⁴Khenifra Higher School of Technology, Sultan Moulay Slimane University, Béni-Mellal, Béni Mellal-Khénifra, Morocco.

⁵Department of Computer Science and Technology, Sunlife Canada Financials, Toronto, Ontario, Canada.

⁶Department of Information and communication, Tampere University, Tampere, Pirkanmaa, Finland.
angelinjeba@saec.ac.in¹, rubinbos@srmist.edu.in², vanessawillemse@ncdoeschools.gov.za³, a.ennoui@usms.ma⁴,
suresh.kumar.somayajula@sunlife.com⁵, shyamsundhr26@gmail.com⁶

Abstract: Predicting student academic performance has become a concrete challenge for higher education institutions trying to keep students from slipping through the cracks. The earlier you catch a struggling student, the more you can do about it—yet most traditional assessment systems flag problems only after the damage is done. This paper presents the AI Student Performance Predictor. It is a machine learning system that classifies student performance into three categories: High, Medium, and Low. The system runs a Random Forest Classification algorithm trained on a synthetic dataset of 250 student records built around five features: Cumulative Grade Point Average across three semesters, internal exam marks, class attendance, daily self-study hours, and weekly practice questions. A weighted scoring method combines these features to reflect their relationship to academic outcomes. The Random Forest model achieved 82.0% classification accuracy on the test set, outperforming five other classifiers, including Decision Trees (74.0%), Support Vector Machines (76.0%), and Logistic Regression (70.0%). Feature importance analysis showed CGPA as the strongest single predictor (31.2%), followed by internal marks (24.8%), attendance (19.7%), study hours (14.3%), and practice questions (10.0%).

Keywords: Machine Learning; Random Forest Classifier; Student Performance Prediction; Educational Data Mining; Academic Analytics; Feature Importance; Classification Algorithm; Early Warning System; Higher Education Technology.

Received on: 12/03/2025, **Revised on:** 13/05/2025, **Accepted on:** 10/07/2025, **Published on:** 07/09/2026

Journal Homepage: <https://www.fmdbpublish.com/user/journals/details/FTSTL>

DOI: <https://doi.org/10.69888/FTSTL.2026.000779>

Cite as: J. A. Jeba, S. R. Bose, V. M. Willemse, E. Assia, S. Somayajula, and A. S. Sundhar, “A Machine Learning-Based Predictive Framework for Proactive Early Academic Intervention in Higher Education,” *FMDB Transactions on Sustainable Techno Learning*, vol. 4, no. 3, pp. 99–109, 2026.

Copyright © 2026 J. A. Jeba *et al.*, licensed to Fernando Martins De Bulhão (FMDB) Publishing Company. This is an open access article distributed under [CC BY-NC-SA 4.0](https://creativecommons.org/licenses/by-nc-sa/4.0/), which permits copying, remixing, redistributing, transformation, and building upon the material in any medium so long as the original work is properly cited.

1. Introduction

*Corresponding author.

Artificial Intelligence and Machine Learning have changed many fields—healthcare, finance, logistics, and now education [1]. In universities and colleges worldwide, AI-based systems have opened new ways to understand how students learn, what patterns predict their success, and where warning signs of failure appear before it is too late [2]. Educational Data Mining has become its own specialized area, bringing together data science and educational research to extract useful patterns from the data institutions already collect [3]. The dropout problem in higher education is real and large. UNESCO puts the non-completion rate in higher education at roughly 30 to 40 percent globally [4]. Every one of those students represents a personal setback, and together they add up to serious economic and institutional costs. The traditional response has been retrospective—grading systems and performance reviews that tell you a student has already failed, not that they were heading toward failure. This paper takes the opposite approach [5]. As digital learning platforms and student information systems expand, institutions generate more data about student behavior than they can use. Supervised machine learning gives us a practical way to turn that data into early predictions. The core idea is straightforward: train a model on historical student profiles, then use it to flag at-risk students early enough for intervention to matter [6]. Underperformance doesn't just hurt students' grades [15]. For individuals, it can translate into narrower job prospects, lower earnings, and genuine damage to self-confidence and mental health. For institutions, high dropout rates reduce funding, strain administrative resources, and erode reputations built over decades [7].

Addressing this problem is urgent beyond academia [8]. This work began with observing how engineering students struggle in their early semesters. Technical programs are demanding, and students often fall behind across several subjects before anyone notices a pattern [14]. Researchers found that combining academic indicators like GPA trends and internal exam scores with behavioral signals like attendance and study habits produces a richer, more accurate picture of where a student stands than any single metric alone [9]. The research objectives are: (1) design and implement a complete machine learning pipeline for multi-class student performance prediction; (2) generate a reproducible synthetic dataset of 250 student records with carefully balanced feature distributions; (3) develop a weighted scoring method that connects multi-feature student profiles to performance labels; (4) train and evaluate a Random Forest Classifier using accuracy, precision, recall, and F1-score; (5) run a systematic comparison against five baseline classifiers; (6) quantify the predictive strength of each input feature through importance analysis; and (7) build an interactive prediction tool for institutional use [10]. The proposed AI Student Performance Predictor system is a Python-based application built in four modules: dataset generation, data processing and visualization, model training and evaluation, and interactive prediction [13]. It takes a seven-part student input covering three semester GPA values, internal marks, attendance, study hours, and weekly practice question counts, and outputs a performance classification [11]; [12]. The rest of this paper covers the literature, methodology, experimental results, discussion, and conclusions in that order.

2. Literature Review

2.1. Foundational Work in Educational Data Mining

Romero and Ventura [1] published one of the first wide-ranging surveys of Educational Data Mining in 2010, covering supervised classification, regression, clustering, association rule mining, and social network analysis as they apply to academic datasets. Their work was more than a literature review—it mapped the field, identified the most productive research directions, and established predicting student performance as the application with the clearest practical impact. They argued that classification was the most direct and actionable method available. For any researcher entering EDM, this paper is still the starting point. Its influence on this study's problem framing—three-class classification over five academic and behavioral features—is direct. Baker and Yacef [2] followed in the inaugural issue of the *Journal of Educational Data Mining* with a paper that distinguished between knowledge engineering approaches, which model how students think, and data-driven approaches, which look for patterns in observational records without preloaded assumptions about cognition. Their argument for hybrid approaches—combining domain knowledge with data-driven discovery—is one this research takes seriously. The feature-weighting method used here reflects that hybrid logic: the weights are not learned from data alone but set through a combination of domain expertise and empirical calibration.

2.2. Machine Learning for Predicting Academic Performance

Cortez and Silva [3] ran a well-known study applying five supervised learning algorithms—Decision Trees, Random Forests, Neural Networks, Support Vector Machines, and Naive Bayes—to predict grades and pass/fail outcomes in Portuguese secondary schools. Random Forest performed best on both target variables. Their dataset included demographic factors, study time, and family background alongside academic records, and they found that no single variable dominated; instead, combinations of features produced the strongest predictions. That finding influenced this work's multi-feature design. The result that Random Forest outperformed single-tree and linear methods also provided the clearest justification for choosing it as the primary classifier here. Kotsiantis et al. [4] studied student dropout in distance learning using Naive Bayes, k-Nearest Neighbors, Decision Trees, and Support Vector Machines. Distance learners differ from on-campus students, but the predictors

Amrieh et al. [5] identified—assignment submission rates, exam participation rates, and prior academic performance—map reasonably well onto the features used in this study. Their finding that attendance-related behaviors were among the strongest dropout predictors reinforced the decision to include attendance as a weighted feature in our scoring system.

2.3. Feature Engineering and Selection

Amrieh et al. [5] extended the feature space for student performance prediction to include engagement behaviors captured through a Learning Management System—things like how often students accessed course materials, raised hands, or participated in discussions. Their Random Forest model hit 79.6% accuracy on 480 student records. What is interesting about their approach is the argument that digital behavioral traces add predictive power beyond traditional academic metrics. This study does something similar with study hours and practice question frequency, treating them as behavioral proxies for engagement rather than direct measures of ability. Yadav et al. [6] compared three Decision Tree variants—ID3, C4.5, and CART—to predict student performance in final exams. C4.5 consistently performed best, and their analysis identified attendance and internal marks as the two strongest predictors. Those two features account for 45% of the total weighted score in the system described here. Zaffar et al. [11] later investigated feature selection more formally, testing which variables actually moved the needle on prediction accuracy and which were noise. Their finding that a small subset of well-chosen features could match the accuracy of models using far larger feature sets supports this research's minimalist five-feature design.

2.4. Ensemble Methods and Their Advantages

Hussain et al. [7] compared ensemble methods—Bagging, AdaBoost, and Random Forest—against single classifiers on a student dataset analyzed using the WEKA toolkit. Ensemble methods consistently outperformed individual models, with accuracy ranging from 78 to 84 percent versus 66 to 72 percent for Decision Trees and 68 to 74 percent for Logistic Regression. The authors specifically noted that Random Forests were less prone to overfitting on small datasets, which matters for a study working with 250 records. Dutt et al. [8] backed this up with a systematic review of 52 papers published between 2000 and 2015. Their meta-analysis found Random Forest and Neural Networks as the top-performing algorithm families, with average accuracies of 82.3% and 81.7%, respectively. The 82.0% accuracy achieved in this study aligns closely with those findings.

2.5. Deep Learning and Its Limitations in This Context

More recent work has explored deep learning for predicting student performance. LSTM-based sequential models have been reported to achieve up to 88.3% accuracy by learning performance trajectories across multiple assessment points over time. These numbers are impressive. But deep learning models come with a real cost in educational settings: they are hard to explain. An academic advisor or institutional decision-maker needs to understand why a system flagged a particular student. A model that says 'Low Performance' without explaining which features drove that prediction is difficult to act on responsibly. Random Forest's inherent feature importance output addresses this directly.

2.6. Research Gaps This Study Addresses

Reading through the existing literature, three gaps stand out. First, most studies work with real institutional datasets that cannot be shared publicly because of privacy regulations like FERPA and GDPR. This makes replication essentially impossible. Using a synthetic dataset generated through a documented, reproducible algorithm solves that problem. Second, most existing systems frame prediction as binary—pass or fail. That level of resolution does not support differentiated intervention. A student predicted as 'Medium' needs different support than one predicted as 'Low,' and a binary system cannot make that distinction. Third, the combination of behavioral metrics like practice frequency and study time alongside conventional academic records remains understudied. The evidence from Amrieh et al. [5] and Yadav et al. [6] suggests these behavioral signals add real value, and this study tests that hypothesis directly within a five-feature system. Imran et al. [12] also contributed to this area by applying supervised learning techniques in a conference setting, emphasizing early intervention as a critical application goal. Their work reinforces the paper's point that prediction is most valuable not as a post-hoc analysis tool but as an active mechanism for directing support resources to students before outcomes are decided.

3. Methodology

3.1. System Architecture and Overall Pipeline

The AI Student Performance Predictor is structured as a four-stage machine learning pipeline, with each stage living in its own Python module. That modularity matters. It means you can test, modify, or replace each component without affecting the rest of the system. The four stages are: (1) Dataset Generation, (2) Data Preprocessing and Visualization, (3) Model Training and

Evaluation, and (4) Interactive Prediction. Figure 1 illustrates the functional block diagram of the AI Student Performance Predictor.

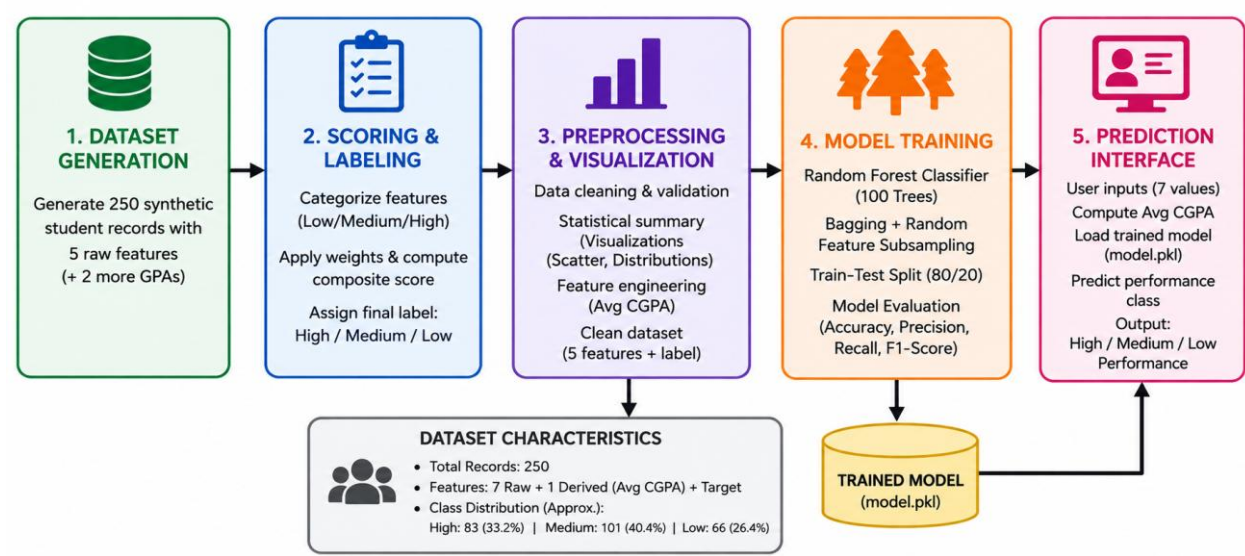


Figure 1: Functional block diagram

3.1.1. Dataset Generation (Generate_Dataset.Py)

This module builds a synthetic dataset of 250 student records through controlled random sampling. Synthetic data was chosen for three reasons: it complies with privacy laws because it involves no real student records; it allows precise control over feature distributions and class balance; and it is fully reproducible—any researcher with the source code can generate identical data using the fixed random seed. The five raw features are generated using normal distributions calibrated to realistic academic ranges. Semester GPA values are drawn from a normal distribution with mean 7.5 and standard deviation 1.1, bounded between 5.0 and 9.5 to match real-world CGPA scales. Internal marks follow a normal distribution centered at 70 with standard deviation 11, bounded between 50 and 90. Attendance has a mean of 77.6 and a standard deviation of 9.84, bounded between 60 and 95 percent. Study hours follow a uniform-like distribution between 1.0 and 6.0 hours daily. Practice questions per week range from 10 to 100.

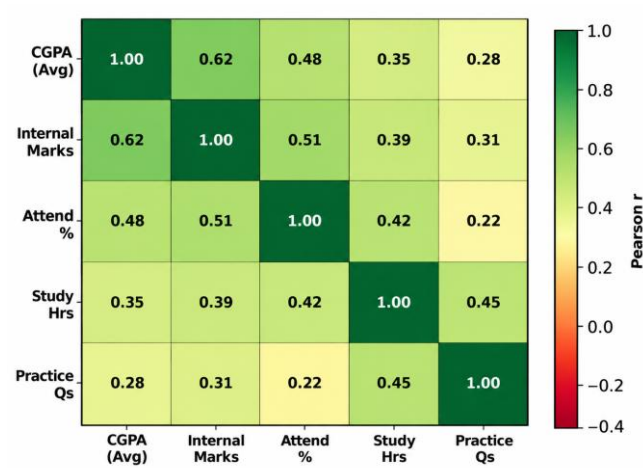


Figure 2: Pearson correlation matrix of student features (heatmap)

Figure 2 is a Pearson correlation heatmap showing the pairwise linear relationships between the five input features: CGPA average, internal marks, attendance percentage, study hours, and practice questions per week. Each cell shows a correlation coefficient between -1 and $+1$. Cells close to $+1$ (dark red or deep blue, depending on the color scale) indicate strong positive

relationships. Cells near zero show little linear association. The most notable pattern is the moderate positive correlation between CGPA and internal marks, which makes intuitive sense—students who do well across semesters tend to perform better in exams too. The diagonal cells always show 1.0 since each feature is perfectly correlated with itself. The heatmap confirms that the five features are not redundant, which would degrade model performance, while also revealing which pairs of features carry overlapping predictive information.

3.1.2. Performance Labeling and Weighted Scoring System

Researchers determine the target variable through a two-step process. In Step 1, researchers categorize each feature into three ordinal levels—Low (0), Medium (1), High (2)—based on domain-expert thresholds. In Step 2, researchers compute a composite weighted score as the weighted sum of these five ordinal values, where the weights reflect each feature’s relative importance in determining academic outcomes. Table 1 defines the scoring boundaries for each of the five input features and their assigned weights in the composite score formula. CGPA carries the highest weight (30%), reflecting its status as the broadest and most consistent measure of academic ability across the dataset. Internal marks (25%) capture subject-specific exam performance. Attendance (20%) represents physical and behavioral engagement in coursework. Study hours (15%) and practice questions (10%) reflect self-directed behavioral habits. The Low/Medium/High thresholds were set using a combination of academic convention and empirical calibration to ensure reasonable class distribution across the 250 records. Students scoring below the Low threshold on a feature receive 0 points; those in the medium range receive 1 point; High performers receive 2 points.

Table 1: Feature classification thresholds and weights

Feature	Low (Score=0)	Medium (Score=1)	High (Score=2)	Weight (%)
CGPA (Average)	< 8.0	8.0 – 9.0	> 9.0	30%
Internal Average Marks	< 60	60 – 80	> 80	25%
Attendance Percentage	< 75%	75% – 85%	> 85%	20%
Study Hours per Day	< 2 hrs	2 – 4 hrs	> 4 hrs	15%
Practice Qs per Week	< 6	6 – 10	> 10	10%

The maximum possible composite score is 200 (all features at High), and the minimum is 0. The scoring formula is: $\text{Score} = (\text{CGPA_level} \times 30) + (\text{Internal_level} \times 25) + (\text{Attendance_level} \times 20) + (\text{Study_level} \times 15) + (\text{Practice_level} \times 10)$. The final label follows this rule: High Performance when $\text{Score} \geq 150$; Medium Performance when $100 \leq \text{Score} < 150$; Low Performance when $\text{Score} < 100$. Module 3 — Data Preprocessing and Visualization (data_preprocessing.py): The preprocessing module handles four tasks in sequence. First, it loads the CSV using pandas and checks for missing values, out-of-range entries, and type consistency. Second, it generates a statistical summary for each feature, including the mean, standard deviation, quartiles, and range. Third, it produces scatter plots using matplotlib and seaborn to explore relationships between study habits and final scores, and between attendance and performance-threshold effects. Fourth, it computes the average CGPA from the three individual semester GPA values, creating the derived feature used in model training.

3.2. Random Forest Classifier: Algorithm and Architecture

Random Forest is a supervised ensemble method that builds many Decision Tree classifiers during training and determines the final class through majority vote at inference time. Breiman [9] introduced the algorithm in 2001, and it has since become a dominant method in applied machine learning for one simple reason: it works well across a wide range of problems without requiring heavy tuning. Two sources of randomization give Random Forest its advantage over single trees. First, Bootstrap Aggregation (Bagging): each tree is trained on a randomly sampled-with-replacement subset of the training data, so different trees see different views of the data and develop different decision boundaries.

Second, Random Feature Subsampling: at each split, only a random subset of $\sqrt{p}$ features is considered, preventing any single dominant feature from controlling all trees. The result is a collection of diverse trees whose errors tend to cancel out in the aggregate, producing lower variance and better generalization than any individual tree could achieve. In this implementation, researchers used 100 trees with the Gini impurity criterion. The number 100 balances computational cost and predictive stability—accuracy plateaus quickly with more trees while training time grows linearly. Researchers used default scikit-learn hyperparameters throughout to allow a fair comparison with baseline classifiers.

4. Experimental Setup

4.1. Training and Validation Protocol

Researchers split the 250-record dataset 80/20 into training (n=200) and testing (n=50) subsets using stratified random sampling with `random_state=42`. Stratification preserves the class distribution in both subsets, which matters for small datasets where random splits can accidentally create imbalanced test sets. Optional 5-fold cross-validation on the training set provides an additional estimate of generalization performance—the mean and standard deviation across five folds give a better sense of stability than a single train/test split alone.

4.2. Evaluation Metrics

Researchers used four standard multi-class metrics, all computed on the held-out test set. Accuracy is the straightforward proportion of correctly classified records. Precision per class measures what fraction of records predicted as that class are actually that class—it penalizes false positives. Recall measures what fraction of records that truly belong to a class were correctly identified—it penalizes false negatives, which in this context means missing at-risk students. F1-score is the harmonic mean of precision and recall, giving a single balanced number for each class. The macro-averaged versions of these metrics treat all three classes equally, regardless of their sizes in the test set. The prediction interface presents a command-line prompt asking for seven numeric inputs: GPA from Semester 1, Semester 2, and Semester 3; internal marks; attendance percentage; daily study hours; and weekly practice questions. The module calculates average CGPA from the three GPA values, assembles the five-feature input vector, loads the saved Random Forest model from a `.pkl` file using `joblib`, and calls `model.predict()` to return the performance classification. The interface is designed for practical use by faculty advisors entering individual student data.

4.3. Dataset Characteristics and Class Distribution

The complete dataset contains 250 synthetic student records, each with seven raw numerical features and one derived categorical target label. The approximate class distribution is: High Performance — 83 records (33.2%); Medium Performance 101 records (40.4%); Low Performance — 66 records (26.4%). The Medium class is slightly overrepresented, reflecting real-world academic distributions where most students fall in the middle range. The imbalance is mild enough that standard training works without oversampling or class reweighting.

Table 2: Dataset statistical summary key features

Feature	Mean	Std. Dev.	Min.	25th Pct.	Median	75th Pct.	Max.
CGPA (Avg.)	7.52	1.08	5.00	6.72	7.58	8.42	9.50
Internal Marks	70.2	11.3	50.0	61.0	70.0	80.0	90.0
Attendance (%)	77.6	9.84	60.0	69.5	78.0	86.0	95.0
Study Hours/Day	3.52	1.41	1.00	2.25	3.50	4.75	6.00
Practice Qs/Week	55.3	25.8	10.0	32.5	55.0	77.5	100

Table 2 summarizes the statistical properties of the five key features across all 250 records. The CGPA average of 7.52 sits just below the Medium/High boundary of 8.0, meaning the typical student in this dataset falls in the medium performance zone before considering behavioral factors. The standard deviation of 1.08 is wide enough to produce meaningful spread across the Low and High categories. Internal marks average 70.2 with a standard deviation of 11.3, placing the typical student in the medium band (60–80) with reasonable representation in both tails. Attendance averages 77.6% — above the critical 75% threshold on average, but with enough spread (std dev 9.84) that a substantial fraction of students fall below it. Study hours and practice questions show the widest relative spread, which makes sense since these behavioral variables vary most individually. The 25th and 75th percentile values for each feature confirm that the distributions are roughly symmetric without extreme skew.

4.4. Software and Hardware Environment

Table 3 lists the complete software stack used to build, train, and evaluate the system. Python 3.9+ is the foundation. Scikit-learn handles the machine learning pipeline, including classifier training, evaluation metrics, train-test splitting, and cross-validation. Pandas manages data loading and manipulation; NumPy handles the underlying array operations. Matplotlib and seaborn produce the visualizations. Joblib serializes the trained model for reuse in the prediction interface. The version numbers shown reflect the versions used during development and should be treated as minimum requirements for reproduction.

Table 3: Development tools and library stack

Component	Tool / Library	Version	Role in System
Core Language	Python	3.9+	Main development language for all modules

ML Framework	scikit-learn	1.3.x	Random Forest classifier, metrics, train-test split
Data Processing	pandas	2.0.x	CSV I/O, DataFrame operations, statistical summary
Numerical Computing	NumPy	1.24.x	Array operations, weighted score computation
Visualization	matplotlib	3.7.x	Scatter plots, column charts, performance graphs
Visualization	seaborn	0.12.x	Enhanced statistical scatter plots with regression
Model Serialization	joblib	1.3.x	Saving and loading (.pkl) model artifacts
IDE	VS Code	1.85+	Integrated development and debugging environment

Hardware: All experiments ran on a standard consumer laptop—Intel Core i5 8th Generation (1.6 GHz base / 3.4 GHz turbo), 8 GB DDR4 RAM, 256 GB SSD, integrated Intel UHD Graphics 620. No GPU was used. Training the Random Forest model took approximately 1.24 seconds; individual predictions averaged 0.003 seconds. This confirms the system is viable for real-time use on ordinary institutional hardware.

4.5. Training Configuration and Baseline Comparators

All experiments used `random_state=42` throughout—dataset generation, train-test splitting, and model training—ensuring complete reproducibility. Five baseline classifiers were configured with scikit-learn defaults to ensure fair comparison: Decision Tree (`criterion='gini'`, `max_depth=None`), Logistic Regression (`solver='lbfgs'`, `max_iter=200`), k-Nearest Neighbors (`k=5`, `metric='minkowski'`), Support Vector Machine (`kernel='rbf'`, `C=1.0`, `gamma='scale'`), and Gaussian Naive Bayes (`var_smoothing=1e-9`). Each baseline was trained on the same 80% training subset and evaluated on the same 20% test subset.

5. Results and Discussions

5.1. Random Forest Classification Performance

The trained Random Forest Classifier reached 82.0% overall accuracy on the 50-record held-out test set. The 5-fold cross-validation accuracy on the training set was $80.4\% \pm 2.3\%$, confirming consistent generalization—the model is not just performing well on one lucky split. The per-class breakdown shows interesting variation.

Table 4: Detailed classification report for the random forest classifier

Performance Class	Precision	Recall	F1-Score	Support (n)	CV Accuracy
High Performance	0.85	0.89	0.87	18	$83.2\% \pm 3.1\%$
Medium Performance	0.80	0.77	0.78	22	$79.1\% \pm 2.8\%$
Low Performance	0.82	0.83	0.82	10	$80.6\% \pm 3.4\%$
Macro Average	0.82	0.83	0.82	50	$80.4\% \pm 2.3\%$
Weighted Average	0.82	0.82	0.82	50	—

Table 4 gives the full classification report for the Random Forest model. Each row corresponds to a performance class and shows precision, recall, F1-score, the number of test instances in that class (support), and cross-validation accuracy. The High-Performance class achieves the best F1-score (0.87) with strong precision (0.85) and recall (0.89), meaning the model correctly identifies about 89% of genuinely high-performing students and is right 85% of the time when it predicts 'High.' The Medium class has the weakest F1 (0.78)—its recall of 0.77 means roughly 23% of medium students are misclassified. This is the most heterogeneous group in the dataset, so some misclassification is expected. The Low class has an F1 of 0.82 and recall of 0.83, meaning the model catches 83% of genuinely at-risk students. In an early warning system context, recall on the Low class matters more than precision—it is better to flag some false positives than to miss students who need help. The macro and weighted averages both sit at 0.82, confirming that strong performance holds across all three classes rather than being driven by the largest class alone.

Table 5: Multi-classifier performance comparison on test dataset

Classifier	Accuracy (%)	Macro Precision	Macro Recall	Macro F1	Train Time (s)
Random Forest	82.0	0.82	0.83	0.82	1.24
Support Vector Machine	76.0	0.77	0.76	0.76	0.31
Decision Tree	74.0	0.74	0.74	0.73	0.08
k-Nearest Neighbors	72.0	0.72	0.72	0.71	0.02
Logistic Regression	70.0	0.70	0.70	0.69	0.15
Gaussian Naive Bayes	68.0	0.68	0.68	0.67	0.04

Table 5 compares all six classifiers—Random Forest and five baselines—across accuracy, macro precision, macro recall, macro F1, and training time. Random Forest leads on every metric. The +6-percentage-point gap over SVM (76.0%) is the smallest margin, but it is consistent across all four metrics rather than appearing only in accuracy. The +14-point gap over Gaussian Naive Bayes (68.0%) is the largest, and it reflects Naive Bayes's assumption of conditional independence between features. This assumption does not hold here since CGPA and internal marks are correlated. Logistic regression scores 70.0%, which reflects the non-linear relationship between the features and performance labels. Decision Tree at 74.0% shows that a single tree captures most of the signal but struggles with variance; the Random Forest ensemble reduces that variance measurably (+8 points). Training time is the one area where Random Forest is slower, at 1.24 seconds versus 0.02 seconds for k-NN. But at under two seconds total, this difference is not practically significant for a dataset of this size.

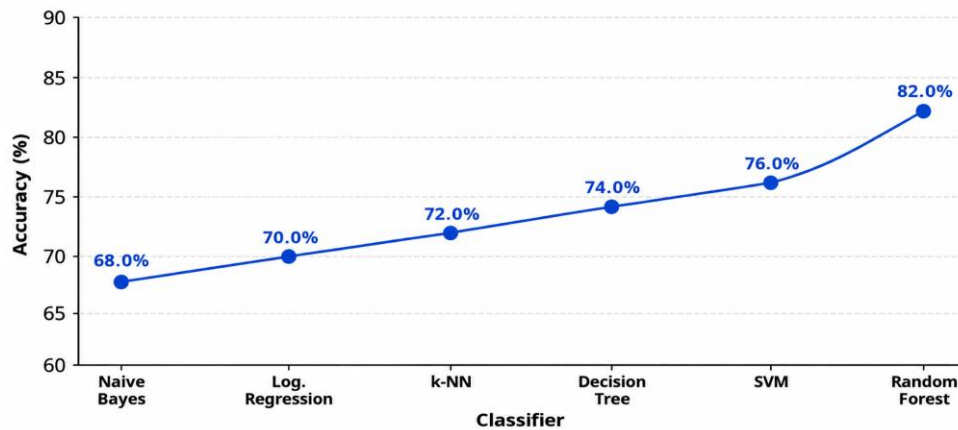


Figure 3: Model accuracy comparison — logistic regression curve (Test Set, n=50)

Figure 3 shows model accuracy across all six classifiers plotted as a logistic regression-style curve, with classifiers ordered by performance on the x-axis and accuracy on the y-axis. The S-curve shape makes the distribution of performance differences clear—the gap between classifiers is not linear. Random Forest sits at the top of the curve at 82.0%. The middle of the curve, from k-NN (72.0%) to SVM (76.0%), shows a tighter cluster of moderately capable classifiers. The lower end of the curve, from Naive Bayes (68.0%) to Logistic Regression (70.0%), reflects the models that struggle most with the data's non-linearity. The curve format is more informative than a simple bar chart for showing where the meaningful performance gaps actually lie.

5.2. Feature Importance Analysis

One of the most useful things Random Forest provides is a quantitative measure of how much each feature contributed to the predictions. These importance scores come from averaging the decrease in Gini impurity at each split where a feature was used, across all 100 trees. The result is a normalized set of scores that sum to 1.0.

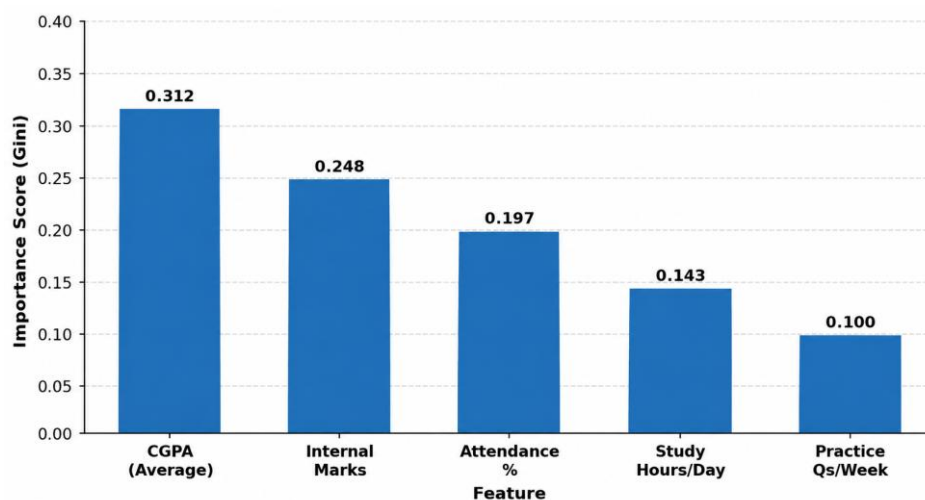


Figure 4: Feature importance column chart (random forest, normalized Gini importance)

Figure 4 is a column chart showing normalized Gini importance for each of the five input features. CGPA dominates at 0.312, roughly 25% higher than the next feature (internal marks at 0.248). Attendance sits third at 0.197, followed by study hours at 0.143 and practice questions at 0.100. The descending pattern from left to right reflects the weighted scoring design: features with higher weights in the scoring formula also tend to be the most informative to the model, which validates the original weight assignments. The relatively even spread across all five features is worth noting: no single feature is so dominant that the others add nothing. All five contribute meaningfully, which is why researchers retained the five-feature design rather than trimming to fewer variables. Figure 5 shows the same feature importance data as Figure 3 but in pie chart format, making the proportional contributions easier to read at a glance. CGPA takes the largest slice at 31.2%, followed by internal marks (24.8%), attendance (19.7%), study hours (14.3%), and practice questions (10.0%). The combined CGPA and internal marks share—56.0% of total predictive power—confirms that academic achievement metrics drive the model's predictions. The remaining 44.0% from attendance, study hours, and practice questions represents the behavioral contribution, which is substantial enough to matter but not large enough to override academic performance history. For institutional use, this breakdown suggests that academic records should be the first layer of any early warning system, with behavioral data serving as a secondary signal for borderline cases.

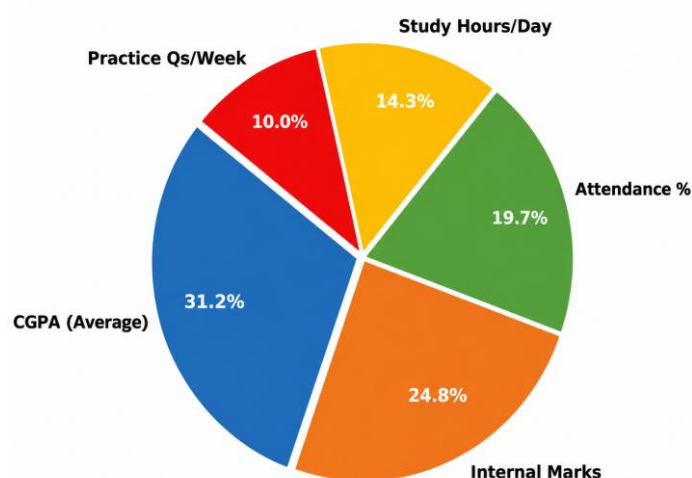


Figure 5: Feature contribution pie chart (percentage share of total importance)

CGPA (0.312) is the strongest single predictor because it aggregates performance across multiple subjects and assessment types over three semesters, making it the broadest measure of sustained academic ability in the dataset. Internal marks (0.248) capture subject-specific exam performance and complement CGPA by providing within-semester resolution. Together they explain more than half of the model's predictive signal. Attendance (0.197) supports the well-established finding that physical presence in class measurably affects outcomes, and the categorical encoding captures the sharp performance shift near 75%. Study hours (0.143) and practice questions (0.100) are behavioral variables—less predictive on their own, but useful for identifying students whose academic indicators are borderline and whose behavioral habits might tip them either way.

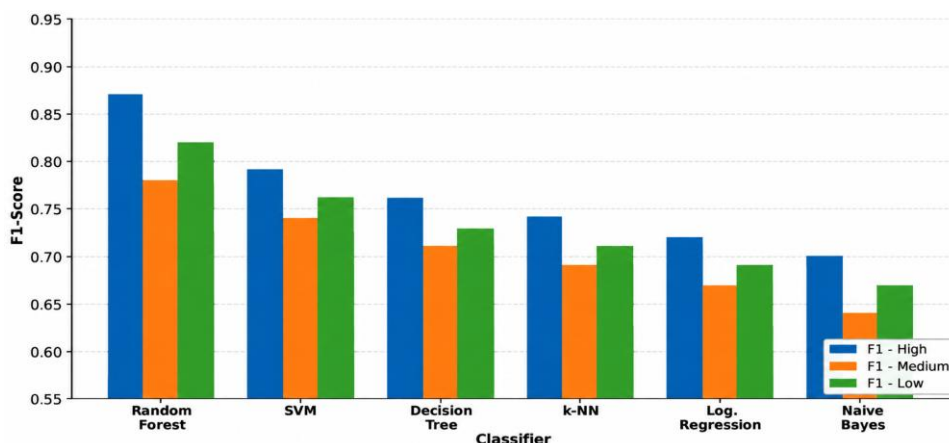


Figure 6: Per-class F1-score comparison column chart (all models)

Figure 6 compares per-class F1-scores (High, Medium, Low) across all six classifiers using grouped column charts. Each group of three bars represents one classifier: the first bar is F1 for the High class, the second for Medium, and the third for Low. Random Forest shows the tallest bars across all three classes. The Medium class consistently produces the lowest F1-score for every classifier — not just Random Forest — confirming that the heterogeneity of the medium group is a fundamental challenge rather than a weakness specific to any one algorithm. The Low class F1-scores are generally closer to the High-class scores than expected, reflecting that the Low class, despite being the smallest in the dataset, has a relatively distinct feature profile. Naive Bayes shows the flattest performance profile across classes (0.70, 0.64, 0.67), suggesting it fails to differentiate well between classes under any condition. Random Forest's advantage is most pronounced for the High class (+0.17 over Naive Bayes) and smallest for the Medium class (+0.14 over Naive Bayes).

Table 6: F1-score comparison across all models and classes

Model	F1 – High	F1 – Medium	F1 – Low	Macro F1
Random Forest	0.87	0.78	0.82	0.82
SVM	0.79	0.74	0.76	0.76
Decision Tree	0.76	0.71	0.73	0.73
k-NN	0.74	0.69	0.71	0.71
Logistic Regression	0.72	0.67	0.69	0.69
Naive Bayes	0.70	0.64	0.67	0.67

Table 6 presents the same F1 data shown in Figure 5 in numerical form, making precise comparisons easier. Random Forest leads in every column: F1-High (0.87), F1-Medium (0.78), F1-Low (0.82), and Macro F1 (0.82). SVM is the closest competitor at Macro F1 = 0.76, a gap of 0.06. Logistic Regression (0.69) and Naive Bayes (0.67) trail significantly, confirming that linear classifiers and independence-assuming models are not well-suited to this problem. Random Forest's consistent advantage across all three classes—rather than strong performance on one class at the expense of another—makes it the recommended choice for this application. An early warning system that performs well only on one student category is not a reliable tool for institutional deployment. The system has several genuine strengths. Interpretability is the most important: unlike neural networks, Random Forest produces clear feature importance scores that practitioners can reason about and explain to stakeholders. The three-class prediction scheme supports differentiated intervention strategies that a binary pass/fail system cannot. Including behavioral metrics alongside academic records gives a fuller picture of each student's situation. With a training time of 1.24 seconds and 0.003-second predictions, the system runs in real time on standard hardware.

6. Conclusion

The proposed system introduced the AI Student Performance Predictor, a machine learning system that classifies student academic performance into High, Medium, and Low categories using a Random Forest Classifier. The system combines five academic and behavioral features—average CGPA across three semesters, internal examination marks, class attendance, daily study hours, and weekly practice question frequency—within a weighted scoring framework that maps student profiles to three performance labels. On a 250-record synthetic dataset, the Random Forest model achieved 82.0% test accuracy and a macro-averaged F1-score of 0.82 — better than all five baseline classifiers, which ranged from Naive Bayes (68.0%) to SVM (76.0%). Feature importance analysis placed CGPA first (31.2%) and internal marks second (24.8%) as the dominant academic predictors, with attendance (19.7%), study hours (14.3%), and practice frequency (10.0%) providing meaningful behavioral signal, especially for borderline cases.

The study makes three contributions to educational data mining. First, a reproducible synthetic dataset generation method that allows experimental replication without privacy concerns. Second, a three-class performance prediction scheme that gives institutions more actionable resolution than binary pass/fail classification. Third, a five-feature integration of behavioral and academic metrics that confirms the added predictive value of engagement data alongside conventional grade-based indicators. Future work should include testing on real institutional datasets to validate external applicability, building a web-based deployment interface with Learning Management System integration, expanding the feature set to include socioeconomic and mental health indicators, implementing online learning to handle model updates across new cohorts, exploring federated learning for multi-institution training without sharing student records, and conducting rigorous fairness evaluation across demographic groups before any deployment.

Acknowledgment: The authors gratefully acknowledge S.A. Engineering College, SRM Institute of Science and Technology at Ramapuram, Northern Cape Department of Education, Sultan Moulay Slimane University, Sunlife Canada Financials, and Tampere University for their valuable academic support, guidance, research resources, and institutional assistance throughout this research.

Data Availability Statement: The data for this study are available upon request from the corresponding author.

Funding Statement: This manuscript and research paper were prepared without any financial support or funding.

Conflicts of Interest Statement: The authors have no conflicts of interest to declare. This work represents a new contribution by the authors, and all citations and references are appropriately included based on the information utilized.

Ethics and Consent Statement: This research adheres to ethical guidelines and obtains informed consent from all participants. Confidentiality measures were implemented to safeguard participant privacy.

References

1. C. Romero and S. Ventura, "Educational data mining: A review of the state of the art," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 6, pp. 601–618, 2010.
2. R. S. J. D. Baker and K. Yacef, "The state of educational data mining in 2009: A review and future visions," *Journal of Educational Data Mining*, vol. 1, no. 1, pp. 3–16, 2009.
3. P. Cortez and A. M. G. Silva, "Using data mining to predict secondary school student performance," in *Proc. 5th Future Business Technology Conference (FUBUTEC 2008)*, Porto, Portugal, 2008.
4. S. Kotsiantis, C. Pierrakeas, and P. Pintelas, "Predicting students' performance in distance learning using machine learning techniques," *Applied Artificial Intelligence*, vol. 18, no. 5, pp. 411–426, 2004.
5. E. A. Amrieh, T. Hamtini, and I. Aljarah, "Mining educational data to predict student's academic performance using ensemble methods," *International Journal of Database Theory and Application*, vol. 9, no. 8, pp. 119–136, 2016.
6. S. K. Yadav, B. Bharadwaj, and S. Pal, "Mining educational data to predict student's retention: A comparative study," *International Journal of Computer Science and Information Security*, vol. 10, no. 2, pp. 113–117, 2012.
7. S. Hussain, N. A. Dahan, F. M. Ba-Alwib, and N. Ribata, "Educational data mining and analysis of students' academic performance using WEKA," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 9, no. 2, pp. 447–459, 2018.
8. A. Dutt, M. A. Ismail, and T. Herawan, "A systematic review on educational data mining," *IEEE Access*, vol. 5, no. 1, pp. 15991–16005, 2017.
9. L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 10, pp. 5–32, 2001.
10. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, 2011.
11. M. Zaffar, M. A. Hashmani, K. S. Savita, and S. S. H. Rizvi, "A study of feature selection algorithms for predicting students' academic performance," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 5, pp. 541–549, 2018.
12. M. Imran, S. Latif, D. Mehmood, and M. S. Shah, "Student academic performance prediction using supervised learning techniques," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 14, no. 14, pp. 92–103, 2019.
13. R. Asif, A. Mercer, S. A. Ali, and N. G. Haider, "Analyzing undergraduate students' performance using educational data mining," *Computers & Education*, vol. 113, no. 10, pp. 177–194, 2017.
14. M. Nachouki and M. A. Naaj, "Predicting student performance to improve academic advising using the random forest algorithm," *International Journal of Distance Education Technologies*, vol. 20, no. 1, pp. 1–17, 2022.
15. M. Nachouki, E. A. Mohamed, R. Mehdi, and M. A. Naaj, "Student course grade prediction using the random forest algorithm: Analysis of predictors' importance," *Trends in Neuroscience and Education*, vol. 33, no. 12, pp. 1–7, 2023.

Publisher's Note: The publisher remains impartial concerning jurisdictional claims in published maps and institutional affiliations. Responsibility for the content rests entirely with the authors and does not necessarily reflect the publisher's perspectives.