Model Report: Student Performance Factors

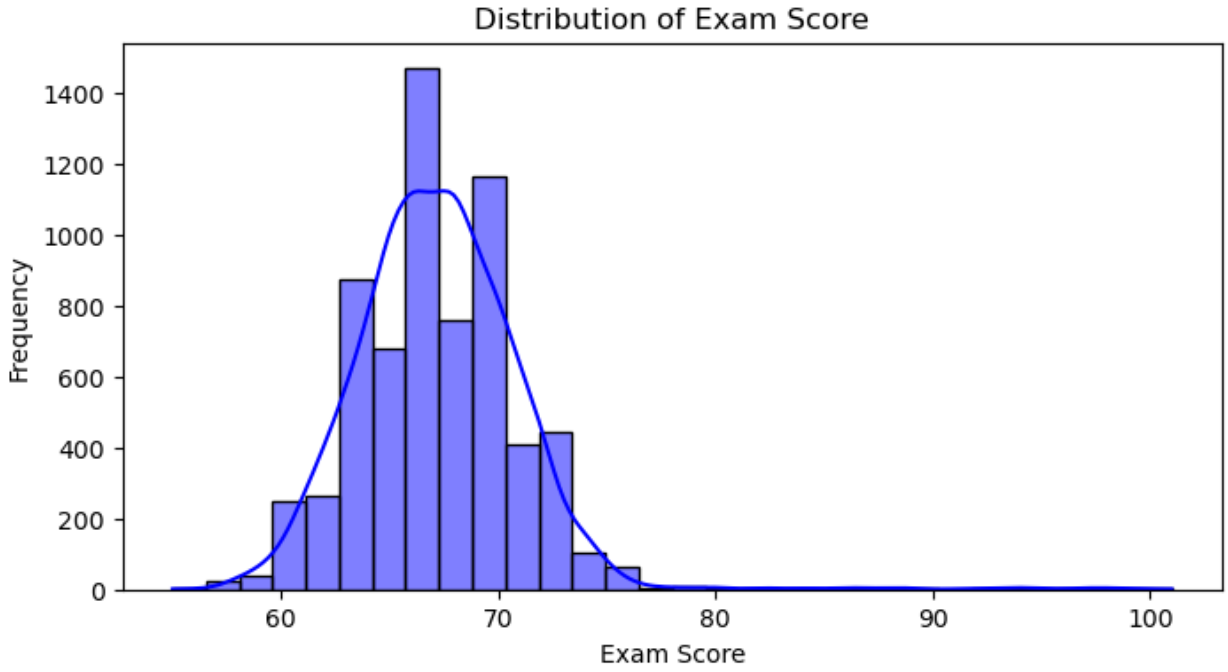
1. Introduction

This project aims to analyze and predict factors influencing student academic performance using various regression models. The goal is to identify key determinants of student success and evaluate model performance to make accurate predictions for future use in educational settings.

2. Dataset Overview

- **Dataset name**: StudentPerformanceFactors.csv
- **Source**: Kaggle
- **Type**: Tabular dataset with both numerical and categorical features
- **Objective**: Predict student performance based on socio-demographic and school-related variables.

Distribution of Exam Scores **between 55 and 101**. The graph above illustrates the frequency distribution of students' exam scores, showing a roughly normal distribution centered around 68–70.



3. Data Preprocessing

Key preprocessing steps included:

- Handled missing values appropriately before modeling.

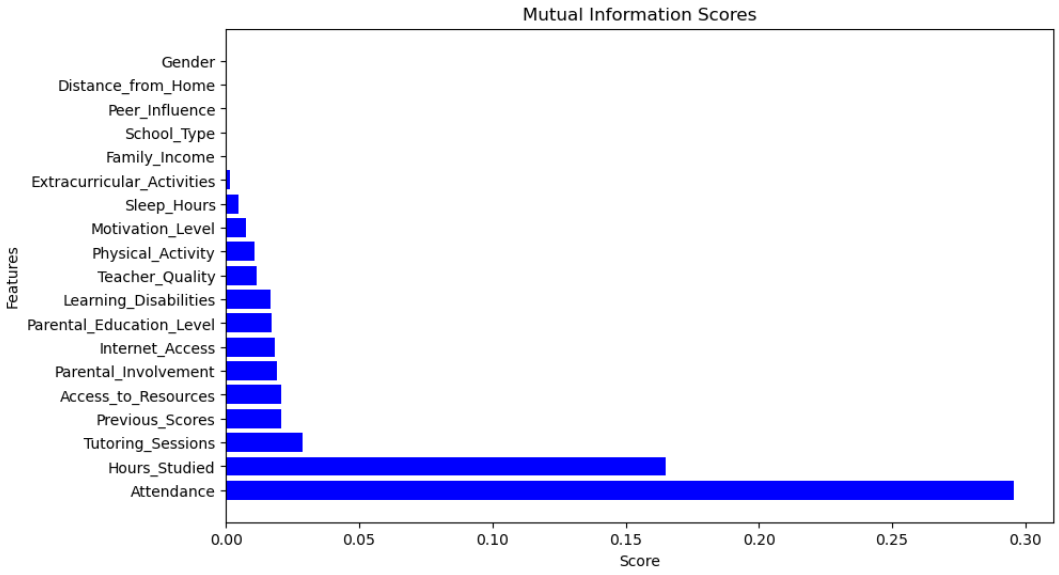
- Encoded categorical variables using **LabelEncoder** and **OrdinalEncoder**. Since the categorical column was ordinal, **Ordinal Encoding** was preferred.
- Standardized numerical features using **StandardScaler** to ensure consistent scaling.

4. **Feature Selection Approach**

In this project, since the target variable is continuous, the task is a **regression problem**. Therefore, I employed **two feature evaluation methods** to ensure robustness in selecting the most relevant predictors:

1. **Mutual Information Regression (MI Regression)**

This method measures the dependency between each feature and the target variable, capturing both linear and non-linear relationships. It is particularly useful for regression tasks.



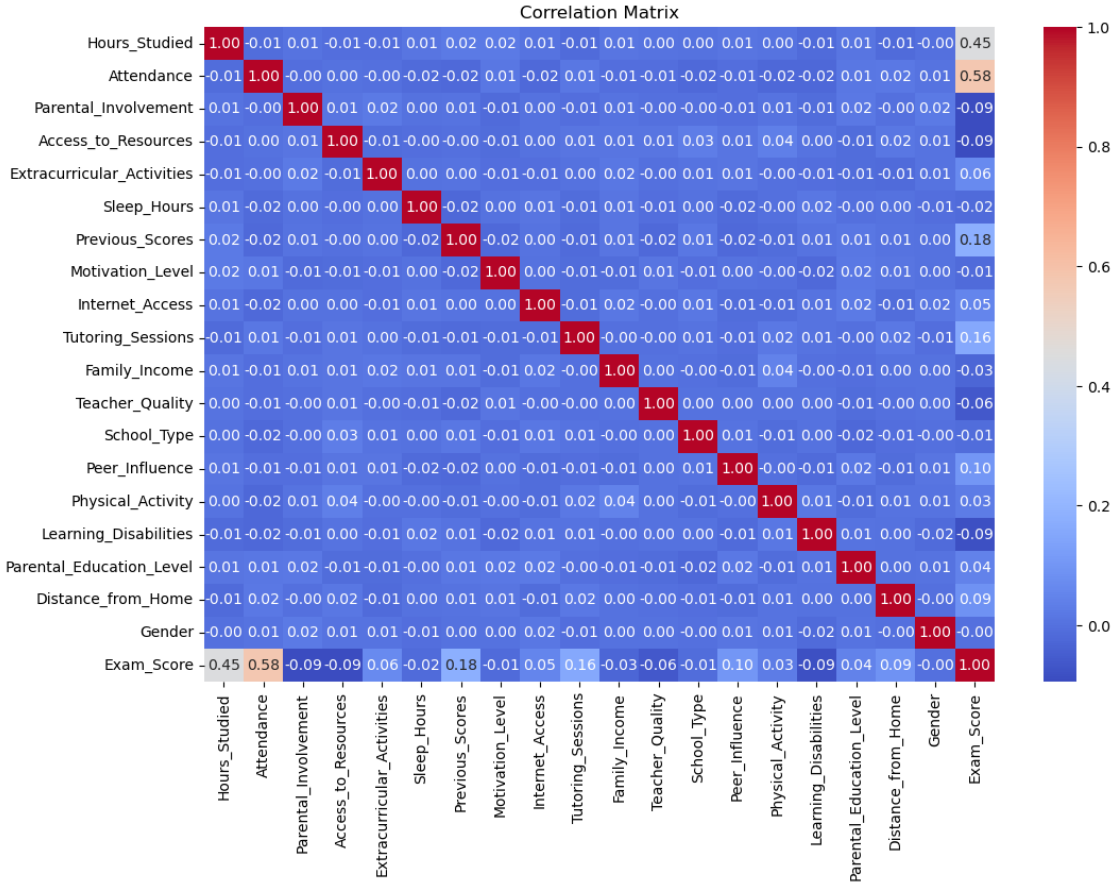
This bar chart displays the mutual information scores of all features with respect to the target variable (*Exam Score*).

It quantifies how much information each feature contributes to predicting the exam score, regardless of whether the relationship is linear or non-linear.

- The most informative features were **Attendance** and **Hours Studied**.
- Features like **Gender**, **Distance from Home**, and **Peer Influence** had negligible information gain and were considered less relevant for prediction.

2. **Correlation Matrix (Pearson's Correlation)**

While Pearson correlation only captures **linear relationships**, it still provides useful insights when features and target are numeric. It was used to cross-verify the strength and direction of linear dependencies between features and the target.



This heatmap shows the Pearson correlation coefficients between all pairs of features and the target variable. It helps to identify linear relationships, collinearity between predictors, and potential multicollinearity issues.

Key observations:

- The strongest positive correlation with Exam Score was seen for Attendance (0.58) and Hours Studied (0.45).
- Most other features showed very weak correlations (close to 0), indicating low linear dependency.

Note: If the task were **classification**, correlation analysis would not be suitable for categorical targets. In such cases, **mutual information** alone is more appropriate for feature selection.

By combining both methods, I ensured that the selected features are relevant not only statistically but also contextually, improving the overall model performance and interpretability.

Feature Engineering

To improve model performance and enhance predictive power, I created new features by combining existing ones that are conceptually related. The goal was to capture interaction effects between variables that might not be apparent individually.
Each of these new features was designed to represent a meaningful combination:

- **internet_sleep_combo**: Quality of study environment considering internet availability and rest.
- **study_time_quality**: Productivity of study time, factoring in teacher quality.
- **family_income_quality**: Overall educational support based on income and parents' education.

Mutual Information Comparison

After creating new features, I re-ran the **mutual_info_regression** analysis to check if these engineered features added any predictive value.

Before Feature Engineering (Top Features):

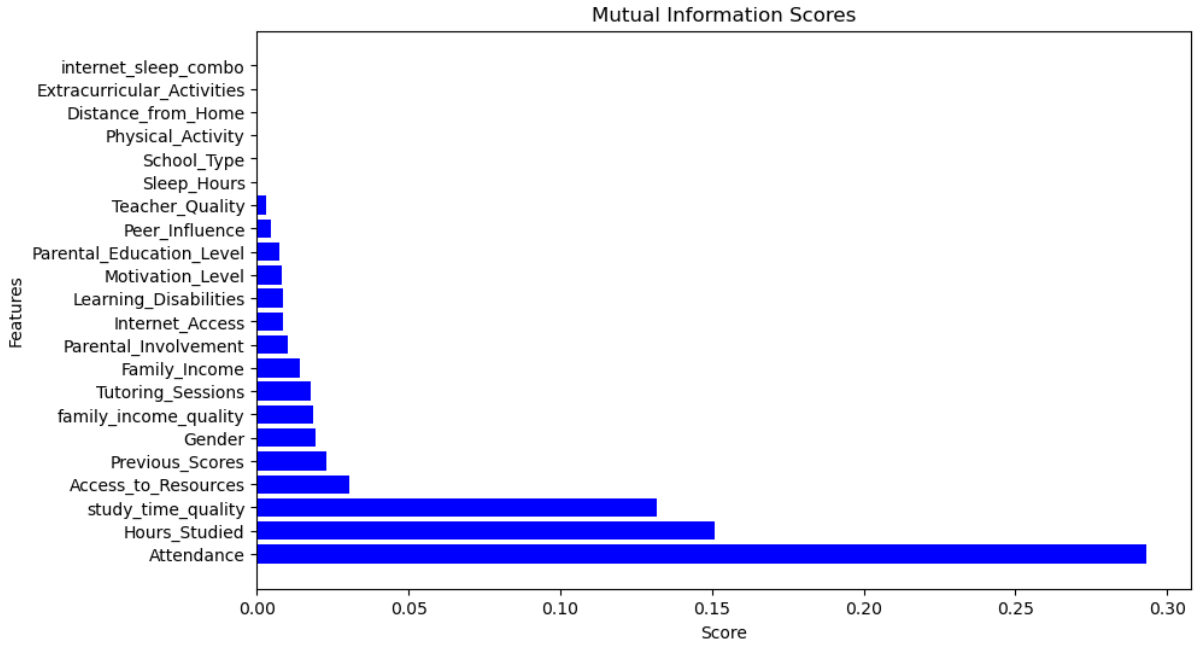
- **Attendance** – 0.2959
- **Hours_Studied** – 0.1652
- **Tutoring_Sessions** – 0.0288
- **Previous_Scores** – 0.0206

After Feature Engineering (Top Features including new ones):

- **Attendance** – 0.2933
- **Hours_Studied** – 0.1510
- **study_time_quality** *(new)* – 0.1318
- **Access_to_Resources** – 0.0303
- **family_income_quality** *(new)* – 0.0183
- **internet_sleep_combo** *(new)* – 0.0000

Interpretation

- The newly created feature **study_time_quality** added substantial predictive value.
- **family_income_quality** contributed modestly.
- **internet_sleep_combo** did **not** improve model performance and had no mutual information gain.



Insight on Feature Contribution Beyond Mutual Information

Although some features had a **mutual information score of zero**, I decided to test the model's performance both with and without them.

Interestingly, when I excluded these zero-scoring features and retrained the model, the **R^2 score decreased**. This revealed a key insight:

Even if individual features do not carry predictive power on their own, they may still contribute indirectly through interactions with other variables.

Therefore, I concluded that **mutual information scores alone are not sufficient** for definitive feature elimination. It is important to also evaluate the collective effect of features within the model's architecture.

5. Model Development

The following machine learning regression models were applied:

- **Linear Regression**
- **Random Forest Regressor**
- **Gradient Boosting Regressor**
- **XGBoost Regressor**

Evaluation Metrics

Models were evaluated using:

- **R^2 Score**
- **Mean Absolute Error (MAE)**

- **Mean Squared Error (MSE)**
- **Root Mean Squared Error (RMSE)**

In this project, I evaluated the impact of removing features with very low or zero mutual information (MI) scores on model performance. The experiment was conducted in two stages:

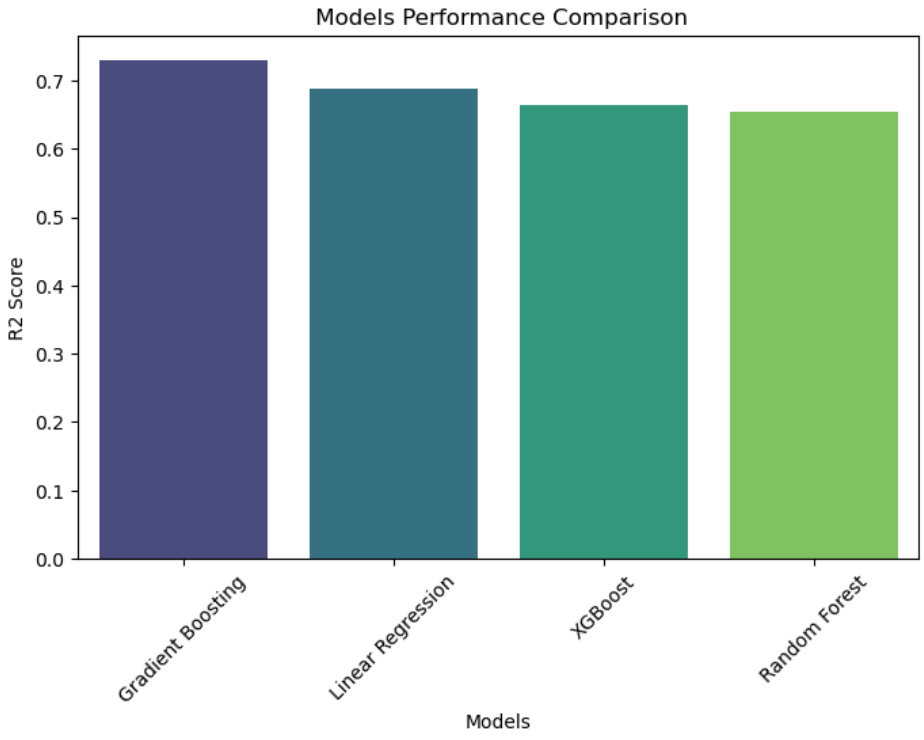
Run 1: All Features Included

In the first run, all available features — including those with MI scores close to zero — were retained. The rationale was to explore the possibility that some features might contribute to model accuracy indirectly, through interaction effects.

Model Evaluation Summary

Model	MSE	MAE	R^2 Score
Gradient Boosting	3.826021	0.821372	0.729324
Linear Regression	4.401070	1.015380	0.688642
XGBoost	4.752240	0.981416	0.663798
Random Forest	4.901650	1.129640	0.653228

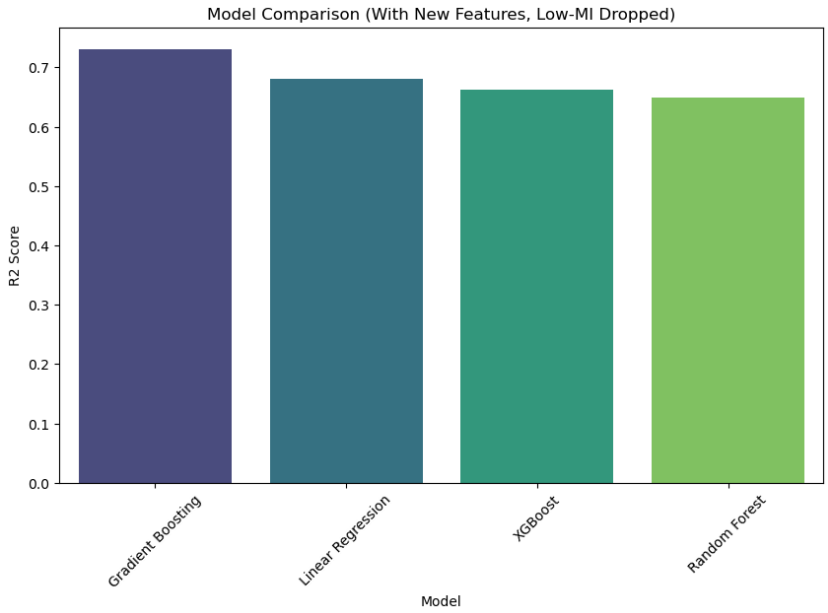
The table below shows the model performance using all features, including those with very low or zero mutual information. Gradient Boosting achieved the best R^2 score, followed closely by Linear Regression and XGBoost.



Run 2: Low-MI Features Removed

In the second run, I removed all features that had mutual information scores equal to or near zero. The assumption was that excluding these features might simplify the model and potentially improve performance.

After removing the low-MI features, some models experienced a decrease in R^2 performance. Surprisingly, Gradient Boosting and XGBoost models maintained stable performance. Linear Regression declined slightly, while Random Forest slightly improved.

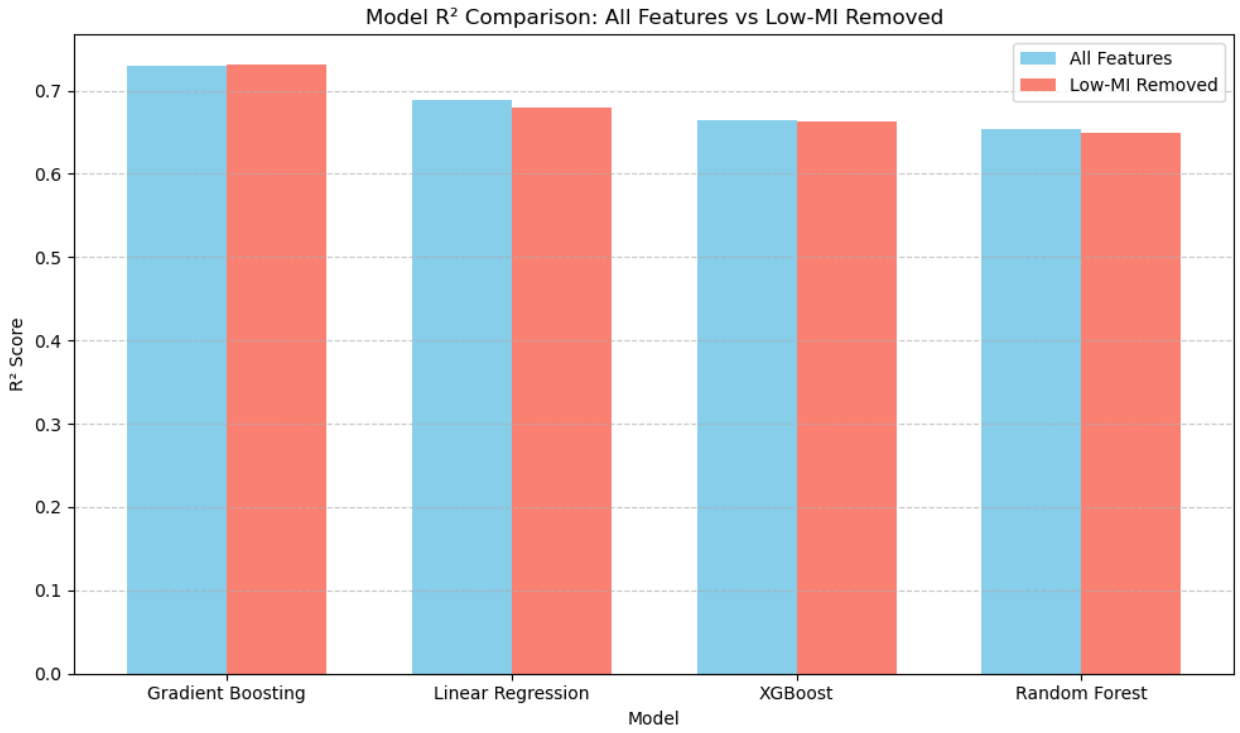


Conclusion

Gradient Boosting consistently achieved the highest R^2 score across both runs.

Low-MI features may not have individual predictive power but can enhance model performance via interactions.

Model robustness depends on architecture: ensemble methods handled feature removal better than simpler models.



Hence, it's critical to **validate feature selection decisions through actual model performance**, rather than relying solely on MI scores.

Among all models, **XGBoost Regressor** achieved the highest R^2 score, indicating its superior predictive power in this context.

8. Conclusion

- Gradient Boosting consistently achieved the highest R^2 score across both runs.
- Low-MI features may not have individual predictive power but can enhance model performance via interactions.
- Model robustness depends on architecture: ensemble methods handled feature removal better than simpler models.

9.Additional Exploration: Classification Task To further investigate the predictive capability of the dataset, a binary classification task was conducted. The target variable Exam_Result was derived from the original Exam_Score, with a threshold of **70** —

students scoring **70 or above** were labeled as **1 (Passed)**, and those below 70 as **0 (Failed)**.

Three classification models were tested:

- Logistic Regression
- Random Forest Classifier
- XGBoost Classifier

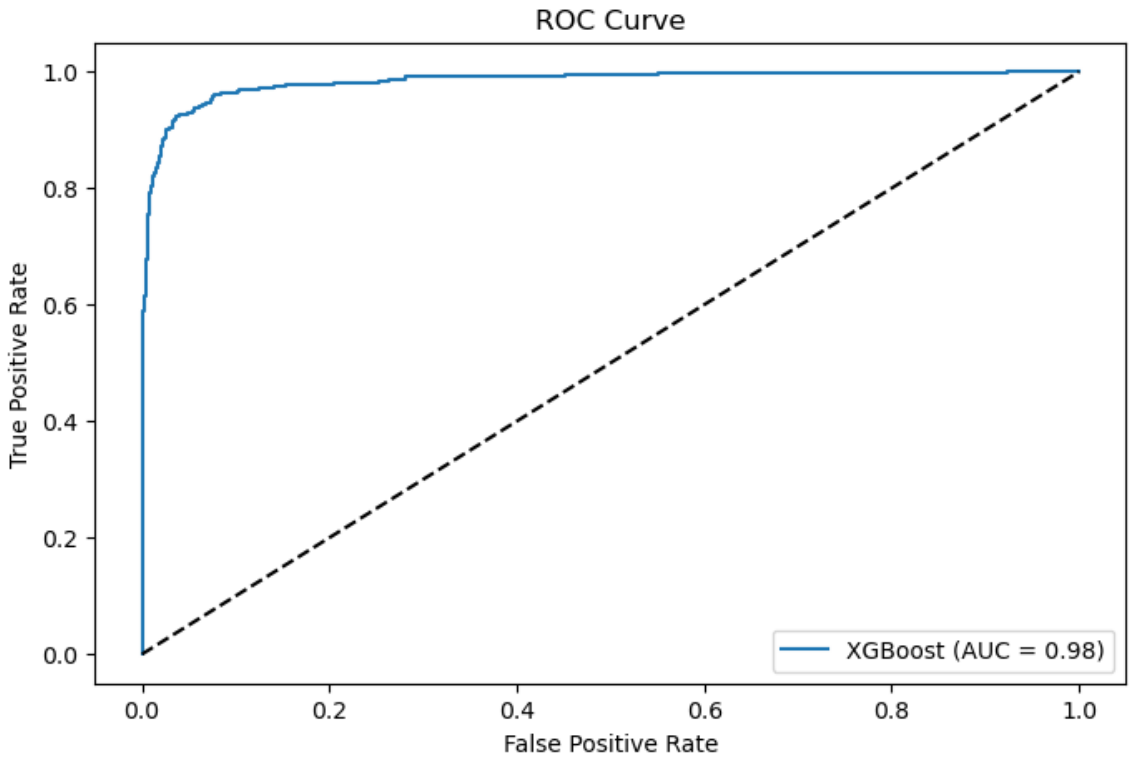
Each model was evaluated using **Accuracy** and **AUC (Area Under the ROC Curve)**.

Summary of Results:

- **XGBoost** showed the best performance with an AUC score of **0.989** and accuracy of **0.956**.
- **Random Forest** followed closely with accuracy of **0.980** and AUC of **0.960**.
- **Logistic Regression** achieved 91.4% accuracy and 0.961 AUC.

Model Evaluation Summary (Low-MI Dropped)

Model	Accuracy	AUC
XGBoost	0.946293	0.982626
Random Forest	0.919062	0.970630
Logistic Regression	0.920575	0.969612



These results indicate that the dataset contains strong signals that help distinguish between students who are likely to pass or fail. XGBoost, as an ensemble model, proved especially effective for classification.

Insight: While regression helps to predict exact scores, classification provides clearer decision-making (pass/fail). In practice, both approaches are valuable depending on the use case — regression for analytics, classification for intervention.