Exploring Factors Influencing Student Success in Universities

F, ElShehaby[1], A. Safwat[1], B. Yosry[1], H. Awad[1], M. Alaa[1], O, Rushdy[1], and O, Walid[1]

[1]Department of Computer Engineering, Egypt University of Informatics

May 20, 2024

1 Abstract

This study investigates the factors influencing student success in universities using data mining techniques. The dataset, obtained from a study by Cortez and Silva (2008)[1], encompasses various socio-economic and academic attributes of students, along with their grades in different periods. The research aims to identify significant predictors of student performance and build predictive models to forecast academic outcomes. We applied data cleaning, hypothesis testing, regression, and classification methods to achieve these objectives. Results indicate that variables such as parental education, study time, and family relationships play crucial roles in determining student success. Regression models achieved high R-squared score of (89% train, 87% test) in predicting student grades, while logistic regression accurately classified students into success and failure with an accuracy of 98%. These findings contribute to understanding the complex interplay of factors affecting student achievement and have implications for educational policies and interventions aimed at enhancing academic performance.

2 Introduction

Student success in higher education is a complex phenomenon influenced by a multitude of factors. Understanding these factors allows universities to develop targeted interventions and support systems for students. This study aims to identify the key relationships between various aspects of student life and academic performance using a data-driven approach.

The research utilizes a comprehensive dataset encompassing academic records, social activities, study habits, and even family background information. Through exploratory data analysis (EDA), we delve into the dataset's structure, uncovering key insights. Using visualizations and statistical analyses, we examine features' distributions, outliers, and missing values, laying the foundation for further analysis.

During EDA, we uncover relationships and trends through hypothesis testing and correlation analyses, highlighting factors influencing student achievement. This process helps us prioritize influential features related to academic performance, guiding subsequent analyses and modeling efforts

Moving forward, our modeling efforts encompass various regression and classification algorithms, including linear regression, ridge regression, and logistic regression. By training and evaluating these models on our dataset, we aim to develop robust predictive models that accurately forecast academic outcomes. These models not only provide valuable insights into the factors driving student success but also offer actionable recommendations for educators and policymakers. By leveraging predictive analytics, schools can identify at-risk students early, implement targeted interventions, and foster a supportive learning environment conducive to student success. Through the integration of data-driven approaches into educational practices, our research aims to empower institutions to optimize student outcomes and promote academic excellence.

3 Methodology

3.1 Data Collection

The study utilizes a comprehensive student dataset obtained from the work of Cortez and Silva (2008). This dataset is in CSV format

and encompasses 33 features along with target variables - G1, G2 and final grade (G3). The features encompass a diverse range of student attributes, including demographics (age, sex), socioeconomic background (parents' education, occupation), study habits (weekly study time, absences), and social factors (romantic relationships, alcohol consumption, extra-curricular activities).

3.2 Data Understanding

Initial data exploration involved familiarization with the dataset structure and feature types (categorical and numerical). We identified 33 features alongside the target variable - final grade (G3) - within a dataset of 649 students (df.shape = (649,33)). Reassuringly, the data exhibited no missing values (null) or duplicate entries.

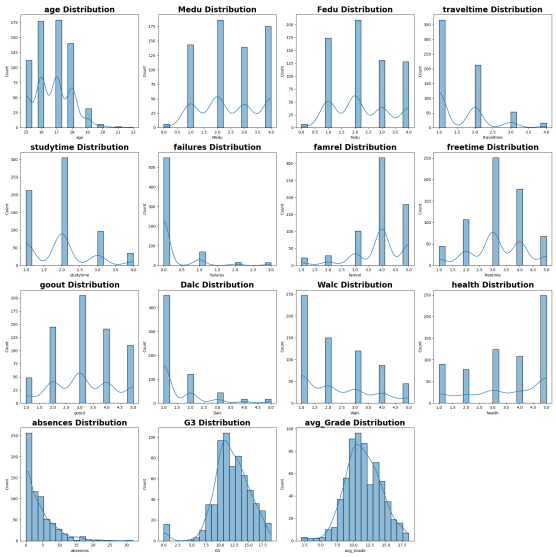
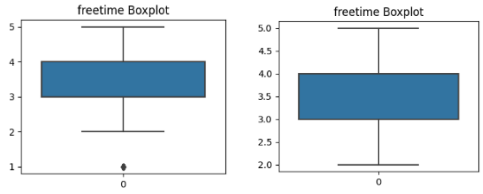


Figure 1: Distribution of Numerical Features in Student Dataset

Next, we examined the distribution of numerical features using techniques like histograms (refer to Figure 1). These visualizations revealed that the majority of students fall within the 15-18 year age range and typically dedicate 2 hours or less to weekly study time. Absences and academic failures are also relatively low. Parental education levels (Medu, Fedu) demonstrate a relatively even spread with a slight preference for mid-level education (refer to Figure 1). Travel times tend to be short, and students generally report positive family relationships and health. While workday alcohol consumption is low, weekend consumption appears moderate for most students. Free time and social activities exhibit a balanced distribution, with a slight inclination towards higher



(a) before (b) after

Figure 2: boxplot of freetime before and after outliers removal

levels of going out. Finally, grades follow a normal distribution, with most students achieving middle-range grades on assessments. These observations suggest a population of students with typical behavior and performance, characterized by balanced lifestyles and moderate academic achievement (refer to Figure 1).

3.3 Outliers Removal

After seeing the feature distibution we checked for outliers like absences, failures, farmel, Dalc, travelyime, studytime, avg_grade and age which we then removed. Detecting and handling outliers is crucial in data analysis. One common method for outlier removal is the Interquartile Range (IQR) technique. Here's how it works:

1. We first calculate the IQR, which represents the spread of the middle 50% of the data. In LaTeX notation, the IQR is expressed as:

$$IQR = Q_3 - Q_1$$

where:
- Q_3 is the third quartile (75th percentile)
- Q_1 is the first quartile (25th percentile)

2. We then define lower and upper bounds for outliers based on the IQR:
* Lower bound: $Q_1 - 1.5 \times IQR$
* Upper bound: $Q_3 + 1.5 \times IQR$

Data points falling outside these bounds are considered outliers.

3. Finally, we remove these outliers from the dataset to potentially improve the reliability of subsequent analyses.

3.4 Exploratory Data Analysis (EDA)

EDA served to uncover underlying trends and relationships within the student dataset. Here, we employed various data visualization techniques to gain insights:

Distribution of Numerical Features: (refer to previous section for details)

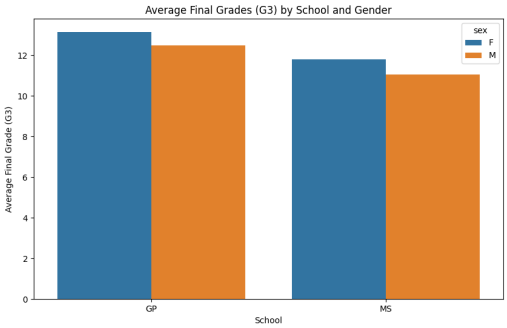


Figure 3: Average Grade for male and female in different schools

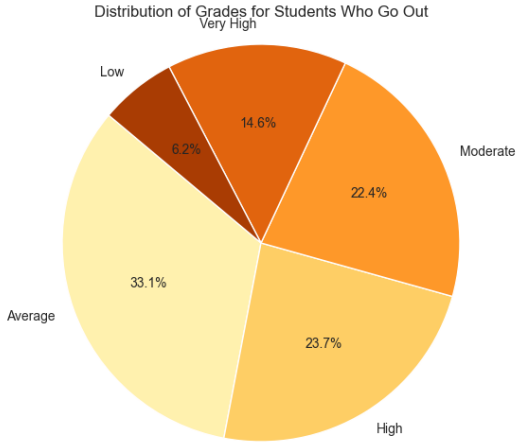


Figure 4: Distribution of Grades for Students Who Go Out

Categorical Feature Analysis: Bar charts or Pie Charts were used to explore categorical features like sex, school, Famrel (family relationship), and weekend activities. This analysis revealed potential relationships between these categories and final grades. For instance, we examined the relationship between sex and final grades using a bar chart (refer to Figure 3). The chart revealed that females generally achieved higher grades compared to males in both schools.

Distribution of Grades by Going Out: The distribution of grades for students categorized by their going out habits was investigated using a pie chart (refer to Figure 4). This visualization revealed that a significant portion (33.1%) of students who go out tend to achieve average grades. Additionally, a notable percentage falls within the high grades (23.7%) category. The remaining students are distributed across moderate (22.4%), very high (6.24%), and low (14.6%) grade categories. This suggests that going out habits may not necessarily have a negative impact on academic performance, as a substantial portion of students who go out achieve average to high grades.

Hypothesis Testing: (refer to next section) Following the exploration of these visual relationships, we have conducted hypothesis tests to assess their statistical significance. This helped determine if the observed trends are likely due to chance or indicate a genuine association between features and final grades. Common tests used in educational data analysis include Pearson's correlation coefficient for linear relationships.

By employing these EDA techniques, we aimed to identify the most promising features that potentially influence student success, paving the way for further analysis through model building.

3.5 Further Analysis and Hypothesis Testing

3.5.1 Correlation Analysis

Multicollinearity occurs when two or more predictor variables in a regression model are highly correlated, meaning that one can be linearly predicted from the others with a substantial degree of accuracy. This can lead to misleading results in regression analysis, as it becomes difficult to isolate the individual effect of each predictor on the dependent variable.

After applying correlation analysis for numerical columns using a heat map (refer to figure 5), we detected multicollinearity between *Medu* and *Fedu*, as well as *Walc* and *Dalc*. To address multicollinearity:

- We dropped *Fedu* due to its high correlation with *Medu*.
- We dropped *Walc* due to its high correlation with *Dalc*.

3.5.2 Hypothesis Test on Binary Categorical Features

Our aim was to identify which binary categorical features significantly affect the final grades.

Steps:

1. Hypothesis Definition:
    - Null Hypothesis (H_0): The two categories in binary columns have the same performance regarding average final grades.
    - Alternative Hypothesis (H_1): The two categories in binary columns have different performance regarding average final grades.

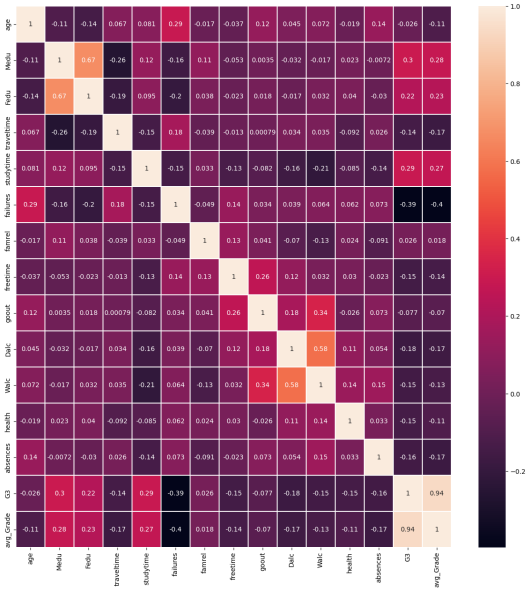


Figure 5: Correlation Analysis for numerical variables

2. Extract categorical columns with only two values.

3. Create a function to perform hypothesis testing on each binary column, comparing the means of the two categories.

4. Use the Z-test for hypothesis testing since the dataset size is greater than 30.

Equations:

$$\mu = \mu_0 - \mu_1$$
$$\bar{x} = \bar{x}_0 - \bar{x}_1$$
$$\sigma = \sqrt{\frac{\sigma_0^2}{n_0} + \frac{\sigma_1^2}{n_1}}$$
$$\text{Z-score} = \frac{\bar{x} - \mu}{\sigma}$$
$$\text{P-value} = 2 \cdot (1 - \Phi(|Z|))$$

If the p-value is less than $\alpha = 0.05$, we reject the null hypothesis.

Results:

- Significant features: *school, sex, address, schoolsup, higher, internet*
- Non-significant features: *famsize, Pstatus, famsup, paid, activities, nursery, romantic*

Evaluation: The significant features had higher correlation coefficients with the target variable (*G3*) than the non-significant features, validating the hypothesis testing method used (refer to figures 6 and 7).

Figure 6: Correlation for rejected categorical variables from our hypothesis

Figure 7: Correlation for non-rejected categorical variables from our hypothesis

3.5.3 Hypothesis Test on Beta Coefficients of Linear Regression Model

Our aim was to improve model performance by ignoring insignificant features, thus avoiding false predictions.

Steps:

1. Hypothesis Definition:
   - Null Hypothesis (H_0): The feature X_m is not significant, $a_m = 0$.
   - Alternative Hypothesis (H_1): The feature X_m is significant, $a_m \neq 0$.
2. Extract the coefficients.
3. Apply the hypothesis test on each coefficient using the following equations:

Equations:

$$\begin{aligned}\text{Test statistic} &= \frac{\text{Value} - \text{Hypothesized value}}{\text{Standard error}} \\ &= \frac{\hat{a}_m - a_m}{SE_{a_m}} \\ &= \frac{\hat{a}_m}{SE_{a_m}}\end{aligned}$$
$$SE_{a_m} = \sqrt{\frac{\sum_{n=1}^{N}(y_n - \hat{y}_n)^2}{N-2}} \Bigg/ \sqrt{\sum_{n=1}^{N}(x_{n,m} - \bar{x}_m)^2}$$

If the p-value is less than $\alpha = 0.05$, we reject the null hypothesis.

Results:

Features with non-significant coefficients were removed, and the model's performance in terms of mean squared error and R^2 score remained unchanged, confirming that the removed features were indeed insignificant.

4 Regression Modeling

Regression modeling is a powerful statistical method used to understand the relationship between independent variables and a continuous

dependent variable. In this context, our target variable of interest is $G3$, representing the final grade of students.

4.1 Pipeline Building

When building a regression model pipeline, we often encounter categorical features that require encoding before model training. Here, we discuss different encoding types:

- **Label Encoder:** Converts categorical labels into numerical representations.
- **Ordinal Encoder:** Converts ordinal categorical features into integer codes.
- **Dummy Encoder/One-Hot Encoder:** Creates binary columns for each category, suitable for non-ordinal categorical variables.

Approach to Pipelining:

1. **Identify Categorical Features:** Begin by identifying which features in your dataset are categorical. Categorical features are those that represent qualitative characteristics rather than quantitative values.

2. **Determine Ordinality:** Determine whether the categorical features are ordinal or nominal. Ordinal features have a clear order or ranking, while nominal features do not have any inherent order.

3. **Choose Encoding Technique:**

- **Ordinal Encoder:** If the categorical feature has a meaningful order or hierarchy, use Ordinal Encoder.
- **Label Encoder:** When the categorical feature is nominal or lacks a clear order, employ Label Encoder.
- **Dummy Encoding (One-Hot Encoding):** For nominal features where no ordinal relationship should be implied, use Dummy Encoding.

4. **Scaling and Imputation:** After encoding categorical features, apply scaling and imputation. Scaling ensures that all features have the same scale, while imputation fills missing values in the dataset with the mean or the median in case of outliers.

Standard Scaler Equation:

$$X_{\text{scaled}} = \frac{X - \mu}{\sigma}$$

Where:

- X is the original feature.
- μ is the mean of the feature.
- σ is the standard deviation of the feature.

5. **Fit Regression Model:** Once the categorical features were encoded using One Hot Encoding, we proceeded to fit our regression model using the encoded features along with the numerical ones. Depending on the nature of our data and the problem at hand, we chose from various regression techniques such as linear and Ridge regression, K-nearest-neighbors, or ensemble methods like XGBoost.

By following this approach, we effectively preprocessed our data and built a regression model pipeline that handled categorical features appropriately.

4.2 Linear Regression

Linear regression is a simple and interpretable model that predicts the target variable by fitting a linear equation to observed data points. The equation for simple linear regression is:

$$Y = \beta_0 + \beta_1 X + \epsilon$$

Where:

- Y is the dependent variable.
- X is the independent variable.
- β_0 is the intercept.
- β_1 is the slope.
- ϵ is the error term.

4.3 Ridge Regression

Ridge regression is a regularized version of linear regression, where the coefficients are estimated by minimizing a penalized residual sum of squares. It adds a penalty term to the least squares loss function, preventing overfitting.

The objective function of Ridge regression can be expressed as:

$$\text{minimize} \left(\sum_{i=1}^{n} (y_i - \hat{y_i})^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right)$$

Where:

- y_i is the observed value.
- $\hat{y_i}$ is the predicted value.
- β_j are the regression coefficients.

- p is the number of predictors.
- λ is the regularization parameter, also known as the penalty term.

The first term in the objective function represents the residual sum of squares, which measures the difference between the observed and predicted values. The second term represents the penalty term, which penalizes large coefficient values by adding their squared magnitudes.

The regularization parameter λ controls the strength of regularization. A larger value of λ results in more shrinkage of the coefficient estimates towards zero, effectively reducing model complexity and preventing overfitting.

By minimizing the combined objective function, Ridge regression finds a balance between fitting the data well and keeping the model coefficients small, thus improving its generalization performance on unseen data.

4.4 K-nearest Neighbors (KNN) Regression

K-nearest neighbors (KNN) regression is a non-parametric method used for regression tasks. Unlike parametric methods such as linear regression, KNN regression does not make any assumptions about the underlying distribution of the data. Instead, it relies on the local neighborhood of data points to make predictions.

4.4.1 Algorithm:

Given a new data point for which we want to make a prediction:

1. **Calculate distances:** Calculate the distance between the new data point and all other data points in the training set. Common distance metrics include Euclidean distance, Manhattan distance, or Minkowski distance.

2. **Find nearest neighbors:** Select the k nearest neighbors to the new data point based on the calculated distances.

3. **Compute prediction:** For regression, the predicted value is the average of the target values of the k nearest neighbors.

4.4.2 Mathematical Formulation:

Let X be the feature matrix of the training set with n samples and p features, and let y be the target vector.

1. **Distance Calculation:** The Euclidean distance between two data points x_i and x_j in the feature space is given by:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{p} (x_{ik} - x_{jk})^2}$$

2. **Prediction:** The predicted value $\hat{y}$ for a new data point x_{new} is computed as the average of the target values of its k nearest neighbors:

$$\hat{y} = \frac{1}{k} \sum_{i \in \text{neighbors}} y_i$$

4.4.3 Hyperparameter Tuning for K-nearest Neighbors (KNN) Regression

Hyperparameter tuning is a crucial step in building machine learning models to optimize their performance. In the case of K-nearest neighbors (KNN) regression, hyperparameters such as the number of neighbors (k), the weight function, and the distance metric significantly impact the model's predictive ability.

4.4.3.1 Parameter Grid: To effectively tune the hyperparameters of a KNN regression model, we define a parameter grid containing different values for each hyperparameter.

For example: param_grid = {
'n_neighbors': [3, 5, 7, 9, 11],
'weights': ['uniform', 'distance'],
'p': [1, 2] # 1 for Manhattan distance, 2 for Euclidean distance }

- **n_neighbors:** This parameter specifies the number of neighbors to consider when making predictions. By exploring different values such as 3, 5, 7, 9, and 11, we can find the optimal value that balances bias and variance in the model.

- **weights:** KNN regression supports two weight functions: 'uniform' and 'distance'. 'uniform' assigns equal weight to all neighbors, while 'distance' assigns weights proportional to the inverse of the distance from the query point. By testing both options, we can determine which weight function yields better performance.

- **p:** This parameter determines the distance metric used for calculating distances between data points. A value of 1 corresponds to the Manhattan distance (sum of absolute differences), while a value of 2 corresponds to the Euclidean distance (square

root of sum of squared differences). By trying both options, we can evaluate the impact of different distance metrics on model performance.

4.4.3.2 Grid Search:

With the parameter grid defined, we employ techniques such as grid search or randomized search to systematically explore different combinations of hyperparameters and identify the combination that maximizes model performance. Grid search exhaustively searches through all possible combinations of hyperparameters, while randomized search samples a fixed number of parameter settings from the specified distributions.

4.4.3.3 Cross-validation:

To ensure the robustness of our model and mitigate overfitting, we utilize cross-validation during hyperparameter tuning. Cross-validation splits the training data into multiple subsets, allowing us to train and validate the model on different combinations of training and validation sets. This helps estimate the model's performance on unseen data and select hyperparameters that generalize well.

By performing hyperparameter tuning with careful consideration of the parameter grid, search strategy, and cross-validation technique, we can build KNN regression models that achieve optimal performance on our dataset.

4.5 XGBoost Regression

XGBoost is a popular ensemble learning technique based on decision trees. It sequentially builds a series of trees, each correcting the errors of the previous one. The final prediction is the sum of predictions from all trees.

When building an XGBoost regression model, several hyperparameters can be tuned to optimize its performance:

- **n_estimators:** The number of trees in the ensemble. Common values include 100, 200, and 300.
- **max_depth:** The maximum depth of each tree. This controls the complexity of the trees and helps prevent overfitting. Common values include 3, 4, and 5.
- **learning_rate:** The learning rate, or step size, used in gradient boosting. It determines the contribution of each tree to the ensemble. Common values include 0.01, 0.1, and 0.2.
- **subsample:** The fraction of samples used to train each tree. It controls the randomness of the training data and helps prevent overfitting. Common values include 0.6, 0.7, 0.8, and 0.9.
- **colsample_bytree:** The fraction of features used to train each tree. It controls the randomness of the features and helps prevent overfitting. Common values include 0.6, 0.7, 0.8, and 0.9.
- **reg_alpha:** L1 regularization term on weights. It encourages sparsity of the weights and helps prevent overfitting. Common values include 0, 0.1, 0.5, and 1.
- **reg_lambda:** L2 regularization term on weights. It helps prevent overfitting by penalizing large weights. Common values include 0, 0.1, 0.5, and 1.

By tuning these hyperparameters appropriately using RandomizedGridSearch which is very similar to GridSearch used in the previous section with KNN Regression, we built an XGBoost regression model that achieved optimal performance on our dataset. By tuning these hyperparameters appropriately using RandomizedGridSearch, which is very similar to GridSearch used in the previous section with KNN Regression, we built an XGBoost regression model that achieved optimal performance on our dataset.

RandomizedGridSearch is a hyperparameter optimization technique that, similar to GridSearch, systematically explores different combinations of hyperparameters. However, it differs in that it does not exhaustively search through all possible combinations, but rather samples a fixed number of parameter settings from the specified distributions. This makes RandomizedGridSearch more efficient and scalable, especially when dealing with a large hyperparameter space.

During the hyperparameter tuning process, we utilized cross-validation to ensure the robustness of our model and mitigate overfitting. Cross-validation splits the training data into multiple subsets, allowing us to train and validate the model on different combinations of training and validation sets. This helped estimate the model's performance on unseen data and select hyperparameters that generalize well.

With careful consideration of the parameter grid, search strategy, and cross-validation technique, we successfully optimized the hyperparameters of the XGBoost regression model, resulting in improved predictive performance on our dataset.

4.6 Evaluation Metrics

Two commonly used evaluation metrics for regression models are Mean Squared Error (MSE) and R^2 (Coefficient of Determination):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y_i})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Where:

- y_i is the observed value.
- $\hat{y_i}$ is the predicted value.
- $\bar{y}$ is the mean of the observed values.
- n is the number of observations.

These metrics quantify the accuracy and goodness of fit of the regression model.

4.7 Classification

Initially, we enhanced our dataset by introducing a new column that assigns grades based on the values in the 'g3' column. For instance, if a student's 'g3' falls between 20 and 17, they are categorized as 'A'.To address categorical data, we utilized the LabelEncoder utility class. This tool facilitates the transformation of categorical labels into numerical representations, assigning them values within the range from 0 to n_classes-1, thereby enabling numerical processing by machine learning algorithms.

Subsequently, we used the following featres ('school2'),('sex2'), ('address2'), school support ('schoolsup2'), interest in higher education ('higher2'), Access to the internet('internet2'), average grade ('avg_Grade'), weekday alcohol consumption ('Dalc'), number of past class failures ('Failures'), weekly study time ('Studytime'), and mother's education level ('Medu'). These features were selected due to their strong correlations with our target variable.

We proceeded by splitting the data into training and testing sets, stored respectively in four variables: X_train, X_test, y_train, and y_test, for further use in our models.

Our initial model choice was logistic regression, which handles categorical values unlike linear regression. Logistic regression employs a sigmoid function to transform the relationship between independent variables and the predicted outcome into probabilities ranging from 0 to 1.

The sigmoid function used for the transformation is represented as:

$$y = \frac{1}{1 + e^{-x}}$$

Moving forward, we employed the Naïve Bayes model, which assumes independence between features given the class label. Despite its simplicity, Naïve Bayes performs well with high-dimensional data.

Lastly, we implemented the k-nearest neighbors (KNN) model. KNN identifies the 'k' data points nearest to the new, unclassified data point using a distance measure (e.g., Euclidean distance) from the training set. Optimal performance in KNN necessitates selecting an appropriate 'k' value and ensuring high-quality training data.

4.7.1 Evaluation Metrics

Accuracy, a fundamental metric for assessing model performance, is calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Where:
- TP: True Positive (the number of correctly predicted positive instances)
- TN: True Negative (the number of correctly predicted negative instances)
- FP: False Positive (the number of incorrectly predicted positive instances)
- FN: False Negative (the number of incorrectly predicted negative instances)

Furthermore, we generated a classification report showcasing F1 score, Precision, and Recall. Precision, representing the proportion of true positive values among all positively predicted values, is calculated as:

$$Precision = \frac{Truepositive}{TruePositive + FalsePositive}$$

F1 score, the harmonic mean of Precision and Recall, is given by:

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

Recall, which evaluates the model's accuracy across various classes, is computed as:

$$Recall = \frac{Truepositive}{TruePositive + FalseNegative}$$

5 Results

5.1 Regression

Based on the comprehensive analysis conducted on our dataset employing various regression models, including Linear Regression, Ridge Regression, KNN Regression, and XGBoost Regression, we have derived insightful conclusions regarding their performance. The tables presented in Table 1 and Table 2 encapsulate the mean Mean Squared Error (MSE), mean R^2 scores, as well as the training and testing R^2 scores for each model. (refer to table 1 and table 2)

From our findings, it's evident that the XGBoost Regression model exhibited the most promising performance, with a mean MSE of 0.90 and a mean R^2 score of 0.88. Furthermore, it demonstrated robustness across both training and testing datasets, showcasing R^2 scores of 0.90 and 0.88, respectively. Following closely behind, the Ridge Regression model displayed competitive results, with a mean MSE of 0.78 and a mean R^2 score of 0.87, highlighting its effectiveness in mitigating overfitting.

While the Linear Regression and KNN Regression models also yielded respectable results, with mean MSE values of 0.93 and 0.92, and mean R^2 scores of 0.88 and 0.87, respectively, they fell slightly short compared to XGBoost and Ridge Regression in terms of overall performance.

In summary, our investigation underscores the efficacy of advanced ensemble techniques like XGBoost Regression in accurately predicting our target variable. These findings can significantly inform future modeling efforts and decision-making processes within our domain, paving the way for more refined and impactful predictive analytics applications.

5.2 Feature Importance:

Feature importance analysis plays a crucial role in understanding the factors driving the predictive power of our regression models. It helps us identify which features contribute the most to the model's predictions and provides insights into the underlying relationships within the data.

One popular technique for assessing feature importance is through SHAP (SHapley Additive exPlanations) values. SHAP values offer a way to explain the output of any machine learning model by attributing the prediction outcome to different features. These values provide a clear understanding of the impact of each feature on individual predictions, offering valuable insights into model behavior.(refer to figure 8)

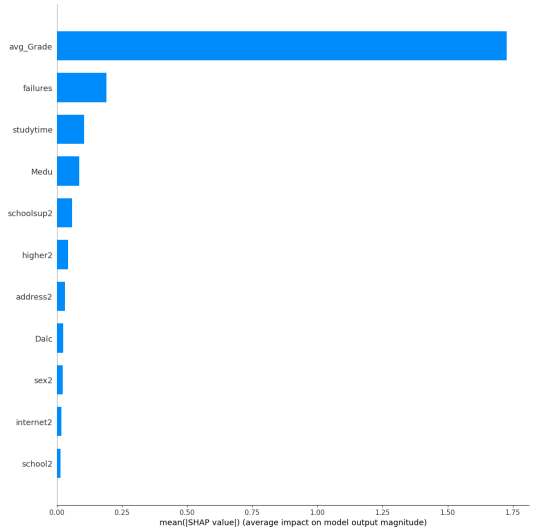


Figure 8: Shap Feature Importance

Upon conducting feature importance analysis using SHAP summary plots, we observed that the 'avg_grade' feature emerged as the most influential predictor in determining the target variable. Following 'avg_grade,' the 'failures' feature exhibited significant importance in shaping the model predictions. However, other features in our dataset were deemed less impactful compared to 'avg_grade' and 'failures,' indicating their relatively lower contribution to the predictive power of the model.

In summary, feature importance analysis using SHAP values provides valuable insights into the key drivers behind our regression models' performance. By identifying the most influential features, such as 'avg_grade' and 'failures,' we gain a deeper understanding of the underlying dynamics within the data, enabling more informed decision-making processes and model refinement efforts.

Table 1: Regression Results

Model	Mean MSE	Mean R^2
Linear	0.93	0.88
Ridge	0.78	0.87
KNN	0.92	0.87
XGBoost	0.90	0.88

Table 2: Regression Performance

Model	Train R^2	Test R^2
Linear	0.88	0.88
Ridge	0.89	0.87
KNN	0.88	0.87
XGBoost	0.90	0.88

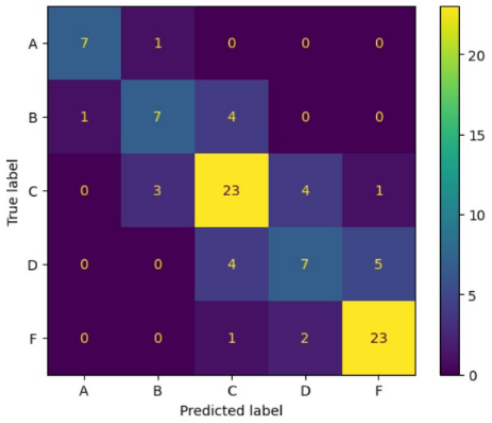


Figure 9: Confusion Matrix for KNN Classifier

5.3 Classification Results:

5.3.1 Multi-class Classification:

In the pursuit of effective classification models, three methodologies were rigorously evaluated: Naive Bayes, Logistic Regression, and K-Nearest Neighbors (KNN).

Naive Bayes, while a classic approach, demonstrated suboptimal performance in this context. Its test accuracy of 38.71% and training accuracy of 41.40% fell short of expectations. The classification report revealed notable inconsistencies, particularly in class 'A' where precision, recall, and F1-score were all negligible.

Moving to Logistic Regression, there was a marked improvement in performance. With a test accuracy of 68.82% and a training accuracy of 75.81%, it displayed a better understanding of the data. However, there remained room for enhancement, especially in achieving higher precision and recall across all classes.

The final contender, KNN, exhibited the most promising results among the trio. Boasting a test accuracy of 72.04% and a training accuracy of 76.34%, it demonstrated robust performance. The classification report underscored its ability to maintain balanced precision, recall, and F1-scores across all classes, indicating a higher level of classification accuracy. (refer to figure 9)

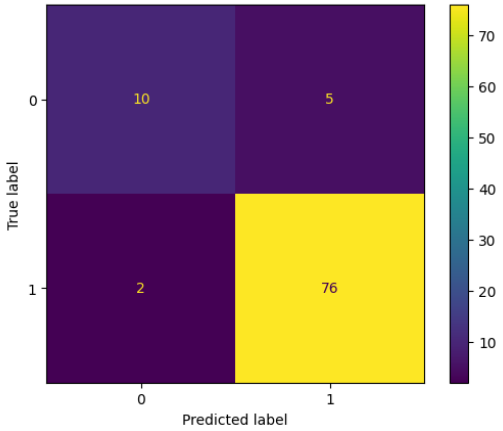


Figure 10: Confusion Matrix for Logistic Binary Classification

5.3.2 Binary classification:

In the realm of binary classification, three distinguished algorithms were put to the test: Naive Bayes, Logistic Regression, and K-Nearest Neighbors (KNN).

Beginning with Naive Bayes, it showcased respectable performance metrics. With a test accuracy and training accuracy both standing at 90.32%, it displayed consistent results. The classification report revealed a balanced precision, recall, and F1-score for class '0', although slightly lower than class '1'. Nonetheless, its overall accuracy stood commendably high at 90%.

Transitioning to Logistic Regression, an uptick in performance was evident. Its test accuracy of 92.47% and training accuracy of 93.82% suggested a more refined understanding of the data. The classification report echoed this sentiment, with improved precision, recall, and F1-score for both classes compared to Naive Bayes. (refer to figure 10)

However, it was K-Nearest Neighbors (KNN) that claimed the crown of superiority in this evaluation. With a test accuracy of 94.62% and a train accuracy of 99.46%, it showcased exceptional proficiency. The classification report underscored its prowess, achieving high precision, recall, and F1-score for both classes. (refer to table 3)

In conclusion, while Naive Bayes and Logistic Regression demonstrated commendable performance, it was KNN that emerged as the clear winner in achieving the highest accuracy and precision in the realm of binary classification.

Table 3: KNN Classification Report

	Precision	Recall	F1-score
0	0.92	0.73	0.81
1	0.95	0.99	0.97
Accuracy			0.95
Macro avg	0.93	0.86	0.89
Weighted avg	0.95	0.95	0.94

6 Conclusion

In this study, we analyzed various factors influencing student success in higher education using a comprehensive dataset. Through exploratory data analysis, hypothesis testing, and correlation analysis, we identified key features that significantly impact academic performance. Our predictive models, including regression and classification algorithms, demonstrated the ability to forecast student outcomes with high accuracy.

7 Recommendations

- **Targeted Interventions:** Universities should focus on significant factors such as study habits, social activities, and family background to design targeted support programs.
- **Early Identification:** Implement predictive analytics to identify at-risk students early and provide necessary interventions.
- **Holistic Support Systems:** Develop comprehensive support systems that address both academic and personal aspects of student life.

By integrating data-driven approaches into educational practices, institutions can enhance student success and foster an environment conducive to academic excellence.

Acknowledgements

We would like to express our sincere gratitude to all those who contributed to the success of this research project. Special thanks go to the individuals who diligently collected and provided the dataset used in our analysis. Your efforts have been invaluable in advancing our understanding of the factors influencing student success in higher education.

A special thanks goes to Dr. Fatma El Shehaby for her guidance and support throughout this research. Your expertise and insights have been crucial in shaping the direction and outcomes of this study.

We also extend our appreciation to our team members for their collaboration and dedication throughout this study. Your insights and hard work have been instrumental in achieving our research objectives.

References

[1] Here's where a full reference would go, should you choose to write up your bibliography in Overleaf.

[2] Here's another. Write these out as you would references anywhere else. The key I've assigned to this reference is 'other_ref' -