PROJECT TITLE
Students Data Analysis
A project submitted for the Data Analysis 1 course (DS2313)

COURSE INSTRUCTOR
Ghadeer Kurdi

SUBMITTED BY:
Lina Fawzi Wali (445001168)
Ghadah Abdullah Al Zahrani (445007065)
Nourah Al masoudi (445000188)
Ruba Saad Almuqati (445000127)

DEPARTMENT OF DATA SCIENCE
COLLEGE OF COMPUTING
UMM AL-QURA UNIVERSITY
2025

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

This project aimed to analyze student academic performance using various machine learning techniques to extract meaningful insights, predict outcomes, and detect abnormal patterns. The dataset used, *Students_Grading_Dataset*, consists of 5,000 student records with 23 features including demographic, academic, and behavioral data such as quiz scores, assignments, attendance, stress level, and sleep hours. The project followed five core phases: data preprocessing to remove missing values and duplicates; regression using Linear and Polynomial Regression to model the relationship between study hours and sleep patterns; classification using Decision Tree, Random Forest, Logistic Regression, SVM, and Gradient Boosting to predict student grades; clustering using K-Means and Hierarchical methods to group students based on academic attributes; and anomaly detection using Z-Score, IQR, Isolation Forest, One-Class SVM, and LOF. Key results showed that regression models performed poorly, with an R^2 score of -0.0075, indicating no meaningful linear relationship. Classification accuracy peaked at 41.5% using Gradient Boosting. K-Means clustering achieved a silhouette score of 0.18, and Isolation Forest detected 129 anomalies. Notable insights included the dominance of “Grade A” in predictions, the weak impact of individual predictors in regression, and the overlap between low-performing students and flagged anomalies. Challenges involved class imbalance, weak feature correlations, and overfitting risks. The findings highlight the importance of applying multiple analytical approaches to better understand student behavior and support educational decision-making.

INTRODUCTION

Understanding student performance is a critical aspect of improving educational systems and supporting academic success. With the growing availability of educational data, data science techniques offer powerful tools to analyze patterns, predict outcomes, and uncover hidden insights. This project aims to apply a range of machine learning and statistical models to explore and evaluate student performance based on academic and behavioral attributes. Using a dataset of 5,000 student records containing 23 features, we investigate various aspects such as study habits, grades, participation, stress levels, and sleep patterns.

The primary objective of this project is to utilize data-driven methods to identify the most influential factors affecting student outcomes, classify academic performance levels, detect anomalies, and discover underlying group structures among students. By implementing supervised learning models (regression and classification), unsupervised techniques (clustering), and anomaly detection methods, this project seeks to demonstrate the practical application of machine learning in the education domain. The findings can offer valuable insights for educators and decision-makers aiming to enhance student support strategies and early intervention.

METHODOLGY

This section outlines the methodology followed to conduct the analysis on the Students_Grading_Dataset. The project was divided into five main phases: data preprocessing, regression, classification, clustering, and anomaly detection.

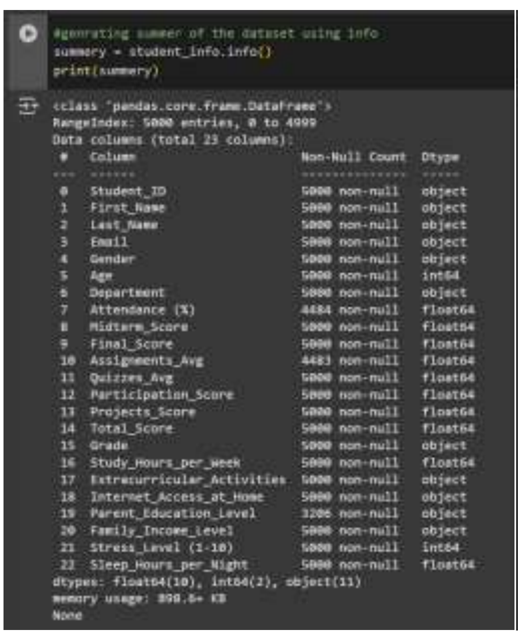
3.1 Data Preprocessing

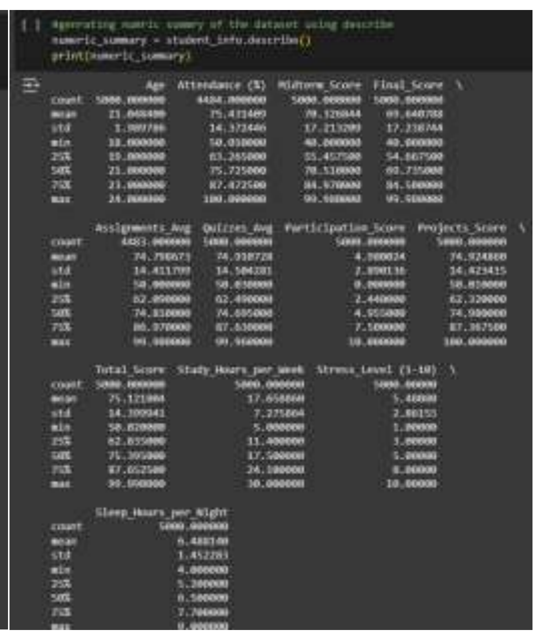
The dataset used in this project is titled Students_Grading_Dataset and was obtained from Kaggle, an open-source data science platform.

https://www.kaggle.com/datasets/mahmoudelhemaly/students-grading-dataset

It contains 5,000 student records with 23 columns representing demographic, academic, and behavioral attributes such as Gender, Age, Department, Attendance (%), Assignments_Avg, Quizzes_Avg, Total_Score, and Stress_Level (1-10).

We began by examining the structure of the dataset using summary methods to understand the data types and overall completeness.





To identify missing values in the dataset, we used a method that showed the number of null entries per column.

```
#number of missing values in each column
missing_values_per_columns = student_info.isnull().sum()
print(missing_values_per_columns)

Student_ID                      0
First_Name                      0
Last_Name                       0
Email                           0
Gender                          0
Age                             0
Department                      0
Attendance (%)                516
Midterm_Score                   0
Final_Score                     0
Assignments_Avg               517
Quizzes_Avg                     0
Participation_Score             0
Projects_Score                  0
Total_Score                     0
Grade                           0
Study_Hours_per_Week            0
Extracurricular_Activities      0
Internet_Access_at_Home         0
Parent_Education_Level       1794
Family_Income_Level             0
Stress_Level (1-10)             0
Sleep_Hours_per_Night           0
dtype: int64
```

We also calculated the number of rows containing any missing value.

```
#number of missing values in each row
missing_values_per_rows = student_info.isnull().sum(axis=1)
print(missing_values_per_rows)

0       0
1       2
2       0
3       0
4       0
       ..
4995    1
4996    2
4997    0
4998    0
4999    0
Length: 5000, dtype: int64
```

After reviewing the extent of the missing data, we decided to drop all rows that contained missing values.

We re-checked the dataset to confirm that there were no remaining missing values. All null counts became zero.

```python
#checking missing values after cleaning
missing_values = student_info.isnull().sum()
print(missing_values)
```

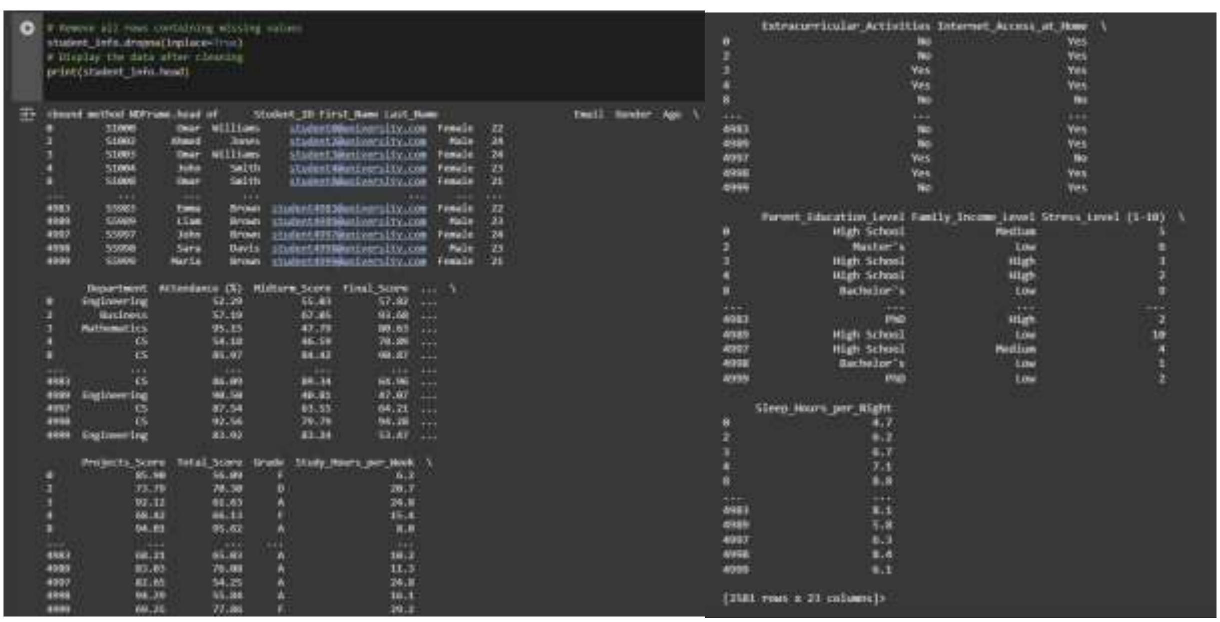
```
Student_ID                   0
First_Name                   0
Last_Name                    0
Email                        0
Gender                       0
Age                          0
Department                   0
Attendance (%)               0
Midterm_Score                0
Final_Score                  0
Assignments_Avg              0
Quizzes_Avg                  0
Participation_Score          0
Projects_Score               0
Total_Score                  0
Grade                        0
Study_Hours_per_Week         0
Extracurricular_Activities   0
Internet_Access_at_Home      0
Parent_Education_Level       0
Family_Income_Level          0
Stress_Level (1-10)          0
Sleep_Hours_per_Night        0
dtype: int64
```

We then checked for duplicate records, and none were found.

```python
#number of duplicates
duplicates_count = student_info.duplicated().sum()
print(duplicates_count)
```

```
0
```

Finally, we previewed the cleaned data using the .head() function.



3.2 Regression Analysis

In this phase, we aimed to predict Sleep_Hours_per_Night using two regression models: Linear Regression and Polynomial Regression.

We tested both models using two sets of independent variables:

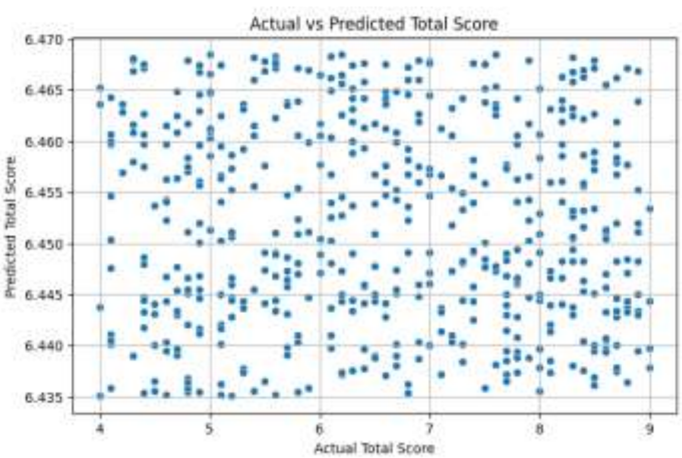
1. Independent Variable: Study_Hours_per_Week

```
Model Evaluation:
Root Mean Squared Error (RMSE): 1.45
Adjusted R-squared (Adj. R²): -0.0095
R-squared (R²): -0.0075
```

2. Independent Variables: Study_Hours_per_Week + Stress_Level

```
Model Evaluation:
Root Mean Squared Error (RMSE): 1.45
Adjusted R-squared (Adj. R²): -0.0122
R-squared (R²): -0.0082
```

Since the first combination gave better performance, we selected it as the final regression model, and created a scatter plot to compare predicted vs. actual values.



We also applied Polynomial Regression using the same independent variable to try a non-linear approach, but the results were still poor.

```
Polynomial Regression Model:
Intercept (b): 6.33
Coefficient of 1: 0.00
Coefficient of Study_Hours_per_Week: 0.01
Coefficient of Study_Hours_per_Week^2: -0.00
Polynomial Regression Evaluation:
Root Mean Squared Error (RMSE): 1.45
Adjusted R-squared (R² adjusted): -0.01
```

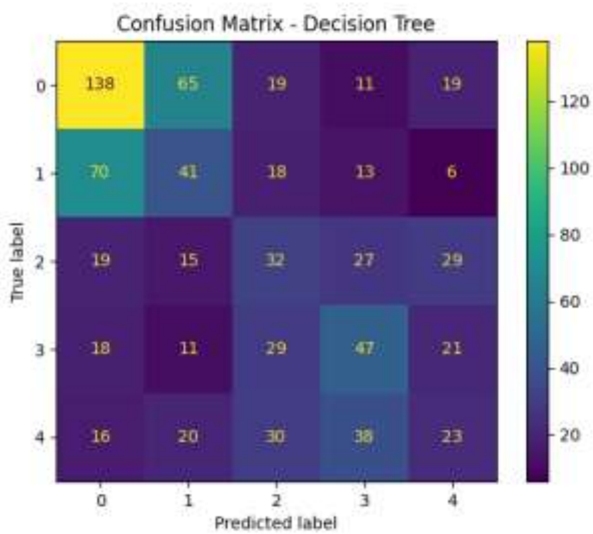
3.3 Classification Models

In this phase, the goal was to classify students into grade categories based on their academic and behavioral features. The data was encoded, scaled, and split (70% training, 30% testing).

We applied:

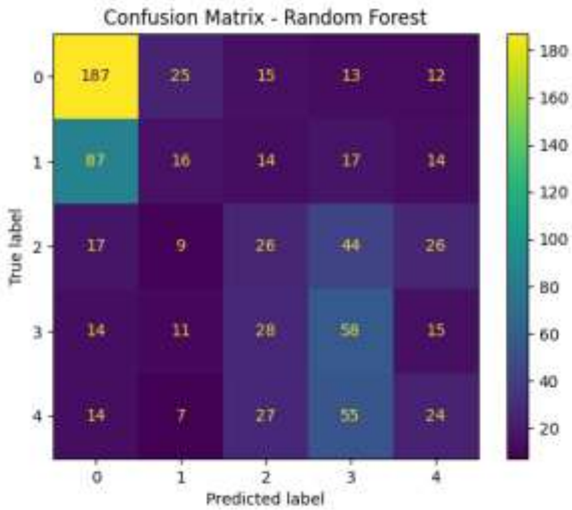
1. Decision Tree

```
--- Decision Tree ---
Accuracy: 0.36258064516129035
Classification Report:
              precision    recall  f1-score   support

           A       0.53      0.55      0.54       252
           B       0.27      0.28      0.27       148
           C       0.25      0.26      0.26       122
           D       0.35      0.37      0.36       126
           F       0.23      0.18      0.20       127

    accuracy                           0.36       775
   macro avg       0.33      0.33      0.33       775
weighted avg       0.36      0.36      0.36       775
```



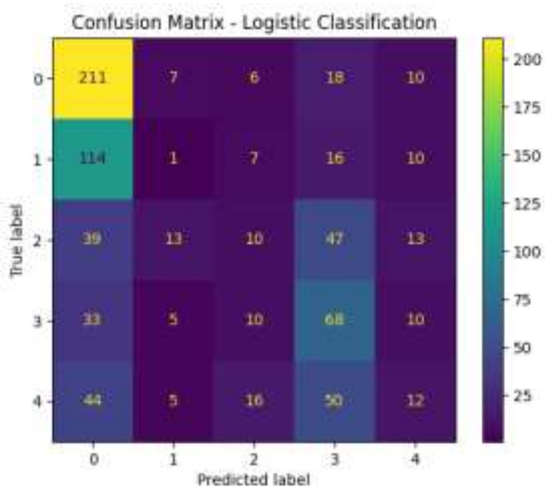
2. Random Forest

```
--- Random Forest ---
Accuracy: 0.4012903225806452
Classification Report:
              precision    recall  f1-score   support

           A       0.59      0.74      0.65       252
           B       0.24      0.11      0.15       148
           C       0.24      0.21      0.22       122
           D       0.31      0.46      0.37       126
           F       0.26      0.19      0.22       127

    accuracy                           0.40       775
   macro avg       0.33      0.34      0.32       775
weighted avg       0.37      0.40      0.37       775
```



3. Logistic Regression

```
--- Logistic Classification ---
Accuracy: 0.3896774193548387
Classification Report:
              precision    recall  f1-score   support

           A       0.48      0.84      0.61       252
           B       0.03      0.01      0.01       148
           C       0.20      0.08      0.12       122
           D       0.34      0.54      0.42       126
           F       0.22      0.09      0.13       127

    accuracy                           0.39       775
   macro avg       0.25      0.31      0.26       775
weighted avg       0.29      0.39      0.31       775
```



Each model was evaluated using accuracy, precision, recall, and F1-score, with confusion matrices for visual analysis.

As an additional step, we tested advanced models:

SVM

```
--- SVM ---
Accuracy: 0.39483870967741935
Classification Report:
              precision    recall  f1-score   support

           A       0.49      0.79      0.60       252
           B       0.23      0.07      0.10       148
           C       0.33      0.18      0.23       122
           D       0.32      0.52      0.40       126
           F       0.16      0.06      0.09       127

    accuracy                           0.39       775
   macro avg       0.31      0.33      0.29       775
weighted avg       0.33      0.39      0.33       775
```

Gradient Boosting

```
--- Gradient Boosting ---
Accuracy: 0.4154838709677419
Classification Report:
              precision    recall  f1-score   support

           A       0.62      0.70      0.66       252
           B       0.22      0.18      0.19       148
           C       0.31      0.30      0.31       122
           D       0.34      0.42      0.37       126
           F       0.31      0.24      0.27       127

    accuracy                           0.42       775
   macro avg       0.36      0.37      0.36       775
weighted avg       0.40      0.42      0.40       775
```

Gradient Boosting achieved the highest accuracy at 41.5%, though performance was generally limited due to class imbalance.

We also tried a different combination of independent variables, but the results were worse.

```
--- Decision Tree ---
Accuracy: 0.23870967741935484
Classification Report:
              precision    recall  f1-score   support

           A       0.32      0.38      0.35       252
           B       0.19      0.24      0.21       148
           C       0.21      0.20      0.20       122
           D       0.19      0.16      0.17       126
           F       0.13      0.08      0.10       127

    accuracy                           0.24       775
   macro avg       0.21      0.21      0.21       775
weighted avg       0.23      0.24      0.23       775
```

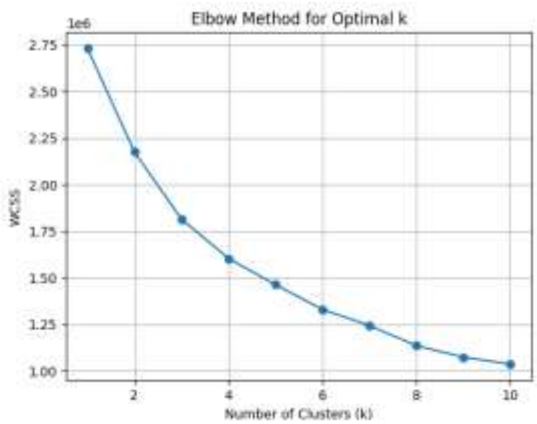
```
--- Random Forest ---
Accuracy: 0.22451612903225807
Classification Report:
              precision    recall  f1-score   support

           A       0.34      0.34      0.34       252
           B       0.20      0.22      0.21       148
           C       0.19      0.17      0.18       122
           D       0.14      0.13      0.14       126
           F       0.13      0.13      0.13       127

    accuracy                           0.22       775
   macro avg       0.20      0.20      0.20       775
weighted avg       0.22      0.22      0.22       775
```

```
--- Logistic Classification ---
Accuracy: 0.3225806451612903
Classification Report:
              precision    recall  f1-score   support

           A       0.33      0.98      0.49       252
           B       0.25      0.01      0.01       148
           C       0.00      0.00      0.00       122
           D       0.18      0.02      0.03       126
           F       0.00      0.00      0.00       127

    accuracy                           0.32       775
   macro avg       0.15      0.20      0.11       775
weighted avg       0.18      0.32      0.17       775
```

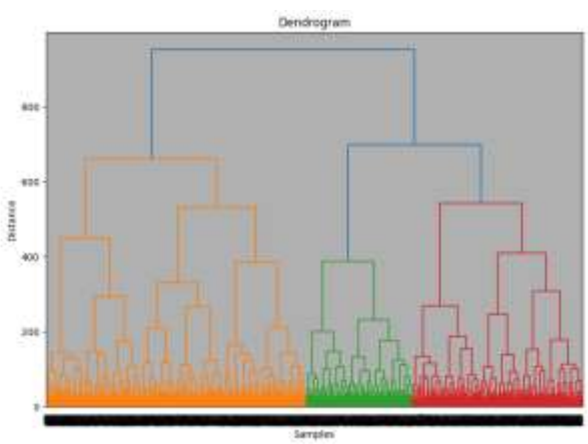
3.4 Clustering

Clustering was performed to explore natural groupings in the student data based on numeric features such as Attendance (%), Study_Hours_per_Week, Final_Score, and Stress_Level.

We began with the K-Means clustering method. Before applying the model, we used the Elbow Method to determine the optimal number of clusters. The elbow point was observed at k = 3, which indicated the best choice for clustering.

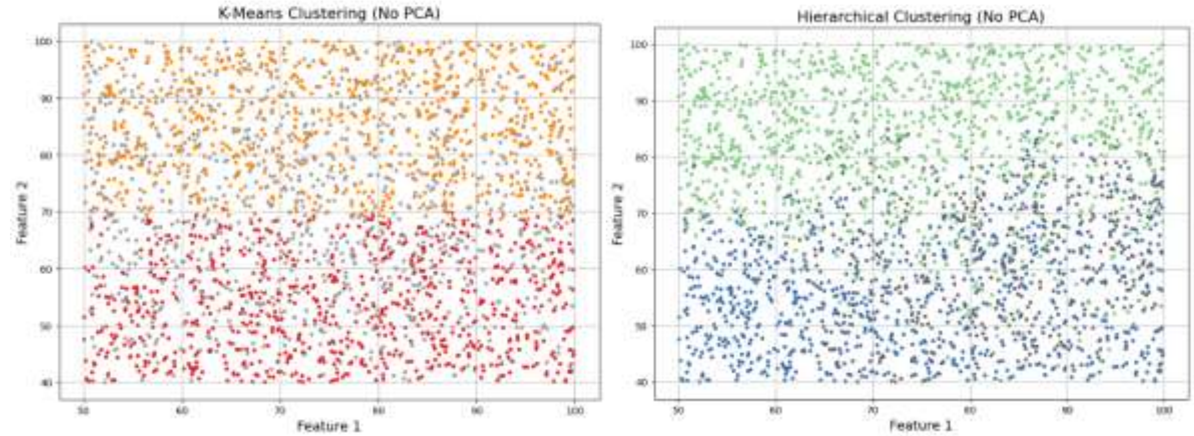


We also used Hierarchical Clustering to explore the relationships between data points. A dendrogram was generated to visualize the merging process and support the choice of using three clusters.

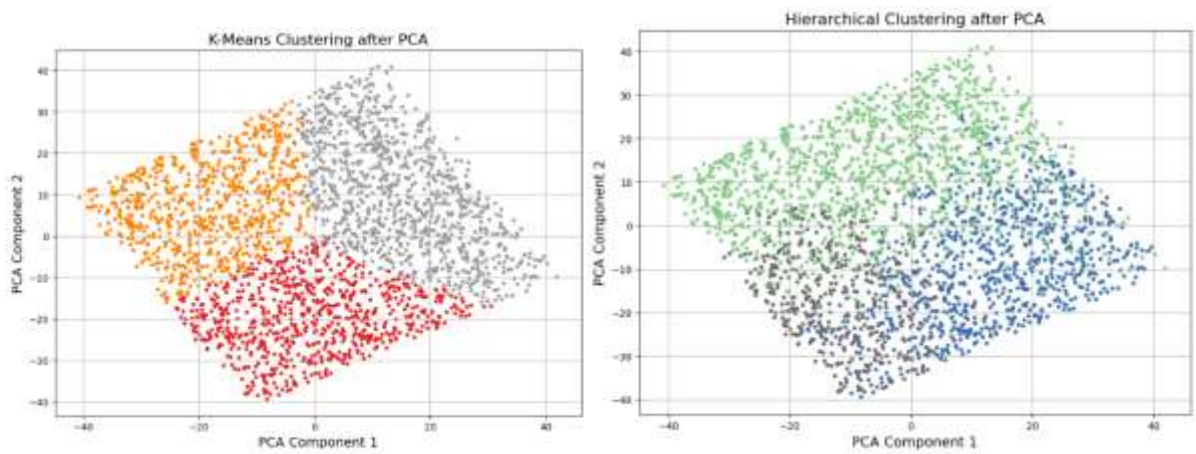


After determining the optimal number of clusters (k = 3), we applied both K-Means and Hierarchical Clustering using this number of clusters.

We visualized the clustering results in 2D space based on the original features. However, the cluster separation was not clear, and many points were overlapping.



To improve the clarity of the visualization, we applied Principal Component Analysis (PCA) to reduce dimensionality and better display the cluster boundaries.



Finally, we evaluated clustering performance using the Silhouette Score to measure how well-separated the clusters were.

```
from sklearn.metrics import silhouette_score

# Calculate silhouette scores for both models
kmeans_silhouette = silhouette_score(X, kmeans_labels)
agglo_silhouette = silhouette_score(X, agglo_labels)

# Print silhouette scores
print(f"K-Means Silhouette Score: {kmeans_silhouette}")
print(f"Hierarchical Clustering Silhouette Score: {agglo_silhouette}")

K-Means Silhouette Score: 0.18712496332095566
Hierarchical Clustering Silhouette Score: 0.119988225921519
```

3.5 Anomaly Detection

Anomaly detection aimed to identify students with unusual or extreme patterns that differ significantly from the majority. We used the original raw dataset before preprocessing, because the cleaned version no longer showed meaningful anomalies.

We used two main anomaly detection approaches:

1. Statistical Methods

This approach depends on mathematical rules to identify values that deviate significantly from the norm.

Z-Score

```
Z-score anomalies: 0
```

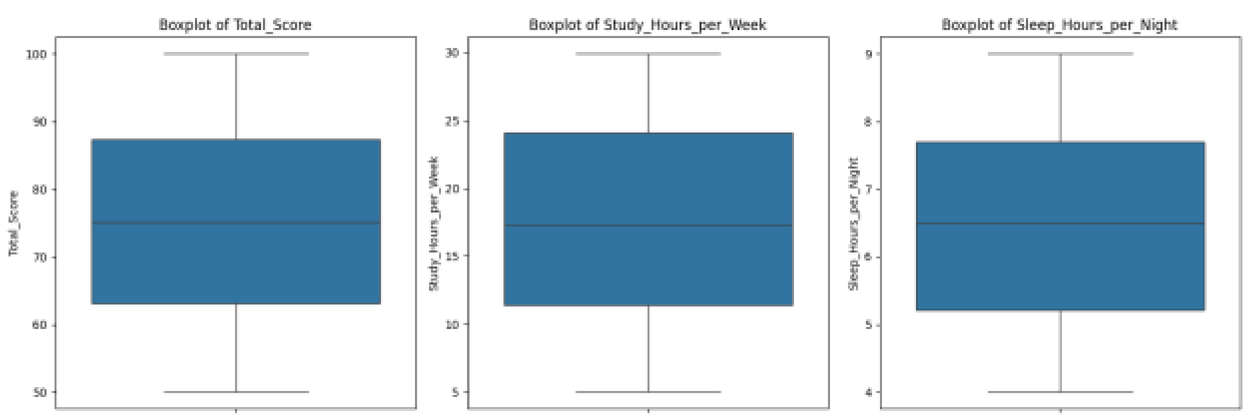
Interquartile Range (IQR)

```
IQR anomalies: 0
```

We also used boxplots to visually detect potential anomalies.

Three separate plots were generated for the following features:

Total_Score, Study_Hours_per_Week, Sleep_Hours_per_Night



However, the boxplots did not show any clear outliers in these features, as there were no data points visibly separated from the whiskers. This confirmed the result obtained from the statistical methods.

2. Unsupervised Machine Learning Models

This approach uses machine learning algorithms to detect outliers based on data distribution, without needing labeled examples.

Models used:

Isolation Forest

```
Isolation Forest anomalies: 129
```

One-Class SVM

```
One-Class SVM anomalies: 682
```

LOF

```
LOF anomalies: 129
```

From the results, we noticed that the models detected different numbers of anomalies:

Isolation Forest and LOF: 129 anomalies

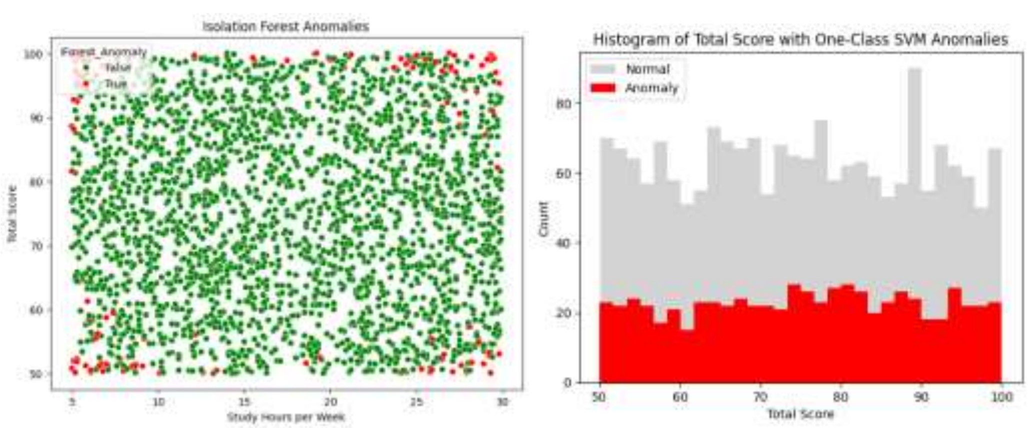
One-Class SVM: 682 anomalies

The difference in detection reflects how each model defines outliers and handles data variance.

We added two visualizations:

A scatter plot showing the anomalies detected by the Isolation Forest model,

and a histogram for the Total_Score feature using the One-Class SVM.



RESULTS AND DISCUSSION

1. Method Performance (metrics & visuals)

- Regression: Linear Regression gave very weak results ($R^2 = -0.0075$, Adjusted $R^2 = -0.0095$). Polynomial Regression didn't improve the performance. The model could not predict sleep hours.
- Classification: The best accuracy was 41.5% with Gradient Boosting. Other models like Decision Tree and Random Forest performed worse. Confusion matrices showed frequent misclassifications.
- Clustering: K-Means and Hierarchical were applied with k = 3. PCA improved the visualization, but Silhouette Scores were low (0.18, 0.12), showing weak cluster separation.
- Anomaly Detection: Statistical methods (Z-Score, IQR) found no major outliers. ML models gave different results:
  - Isolation Forest & LOF: 129 anomalies
  - One-Class SVM: 682 anomalies
    Visuals included scatter plots and histograms.

---

2. Main Results with Visuals

All results were supported by clear visuals:

- Regression: scatter plots
- Classification: confusion matrices
- Clustering: elbow curve, PCA plots, silhouette scores
- Anomaly detection: boxplots, scatter plot, histogram

---

3. Training vs. Test / Cross-Validation

We used a 70/30 train-test split for all models.
No cross-validation was used.
Some models (like Gradient Boosting) performed better on training than testing, suggesting slight overfitting.

---

4. Meaning of the Results (Context)

- Study hours and stress level could not predict sleep.
- Features didn't separate grade categories well → poor classification.
- Clustering failed to find meaningful groups.
- Anomaly results changed based on the model used.

---

5. Limitations and Suggestions

Limitations:

- Important features (like mental health, motivation) are missing
- Class imbalance in classification
- No cross-validation
- Weak structure in data

Suggestions:

- Add more diverse and meaningful features
- Use cross-validation
- Apply class balancing techniques
- Try advanced feature selection

CONCLUSION

This project aimed to analyze student data to uncover patterns related to sleep, grades, and academic behaviors using various machine learning techniques. The analysis included data preprocessing, regression, classification, clustering, and anomaly detection.
Across the models, results were generally weak. Linear and polynomial regression models failed to predict sleep hours. Classification models such as Gradient Boosting achieved the highest accuracy (41.5%) but still struggled due to feature overlap and class imbalance. Clustering showed weak group separation, and anomaly detection results varied significantly depending on the algorithm used.
These outcomes highlight that the available features were not sufficient to explain or predict key behaviors. Major challenges included limited data variability, missing behavioral features, and lack of class balance.
For future work, improvements could include testing more advanced algorithms, using cross-validation, collecting additional features (e.g., mental health, motivation), and applying feature engineering or hyperparameter tuning.
Despite the limitations, this project demonstrated how machine learning can be applied to educational data, and provided insights into what features may be valuable for understanding student outcomes. The results can guide future efforts in using data-driven methods to support academic performance analysis.

REFERENCES

1. Elhemaly, M. (n.d.). *Students Grading Dataset*. Retrieved from https://www.kaggle.com/datasets/mahmoudelhemaly/students-grading-dataset
2. Chen, T., & Guestrin, C. (2016). *XGBoost: A scalable tree boosting system*. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794.
3. McKinney, W. (2010). *Data Structures for Statistical Computing in Python*. *Proceedings of the 9th Python in Science Conference*, 51–56.
4. Oliphant, T. E. (2006). *A Guide to NumPy*. USA: Trelgol Publishing.
5. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). *Scikit-learn: Machine learning in Python*. *Journal of Machine Learning Research*, 12, 2825–2830.
6. Waskom, M. (2021). *Seaborn: Statistical data visualization*. *Journal of Open Source Software*, 6(60), 3021.

THE EXPERIENCES AND SKILLS ACQUIRED BY THE TEAM MEMBERS

Student Name	Experiences	Skills
Lina Fawzi wali	Participated in all phases of the project including data preprocessing, model development (regression, classification, clustering, and anomaly detection), and results interpretation. Worked collaboratively with the team on coding, testing, and documenting the findings. Each member was equally involved and contributed throughout the project.	Teamwork, communication, Python programming, data cleaning, model evaluation (R^2, accuracy, silhouette score), using machine learning libraries (Scikit-learn, XGBoost), and creating visualizations with Matplotlib and Seaborn
Ghadah Abdullah Al Zahrani		
Nourah Al masoudi		
Ruba Saad Almuqati		