# INF 395 Final Report

*Students Performance Prediction*


***Madi Tegisbek***

***ID: 220103250***


05.05.2024

3rd grade Information Systems

## INTRODUCTION

In the modern educational landscape, understanding and predicting student performance has become a crucial aspect of improving learning outcomes and ensuring academic success. Educational institutions are increasingly leveraging data-driven approaches to identify students at risk, provide timely interventions, and optimize teaching strategies.

This project focuses on predicting student performance using machine learning techniques based on various academic, demographic, and behavioral factors. By analyzing data such as study habits, parental involvement, extracurricular activities, and other personal attributes, we aim to classify students into performance categories (A–F) using supervised learning algorithms.

The primary goal of this project is not only to achieve accurate predictions but also to gain insights into the key factors that influence academic achievement. Such insights can help educators and policymakers develop more personalized and effective support systems for students.

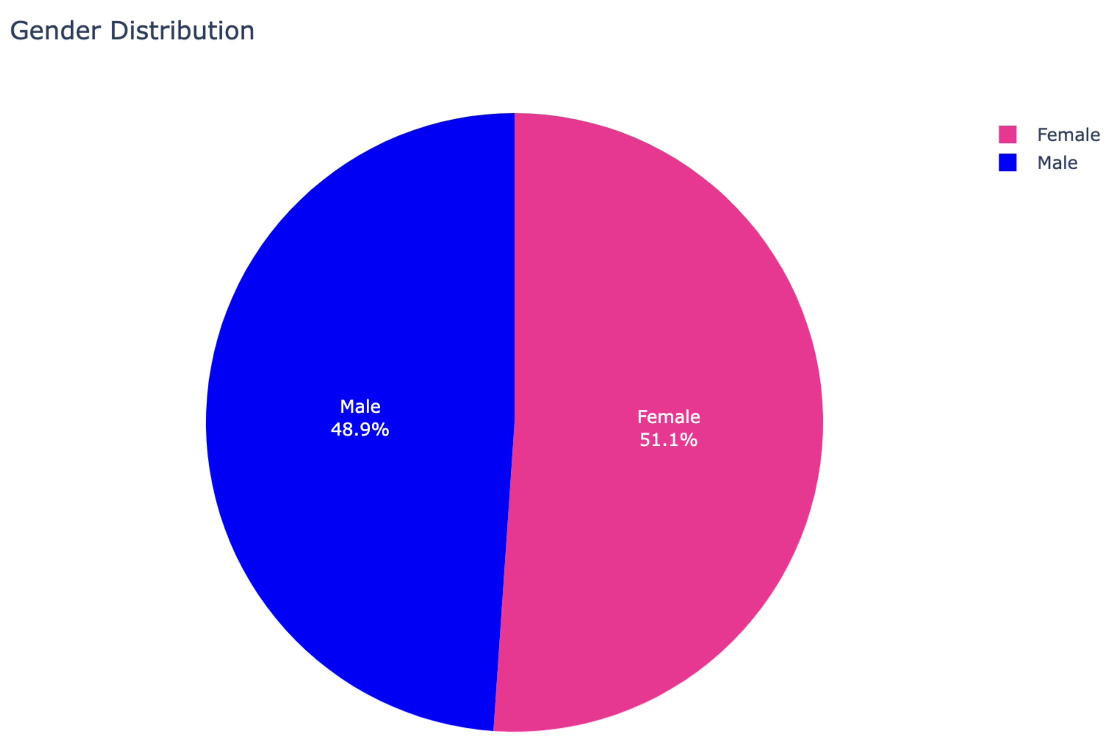
## CONTENT

The dataset used in this project contains **2,392 clean and ready-to-use student records**, covering a wide range of features:

- **Demographics**: Age, gender, ethnicity, parental education
- **Study habits**: Weekly study time, absences, tutoring
- **Parental involvement**: Level of support
- **Extracurricular activities**: Participation in sports, music, volunteering
- **Academic performance**: GPA (2.0–4.0 scale), with classification into Grade A–F (`GradeClass`)

Each student is uniquely identified by a `StudentID`. The target variable for prediction is `GradeClass`, derived from the GPA.

## EDA - EXPLORATORY DATA ANALYSIS



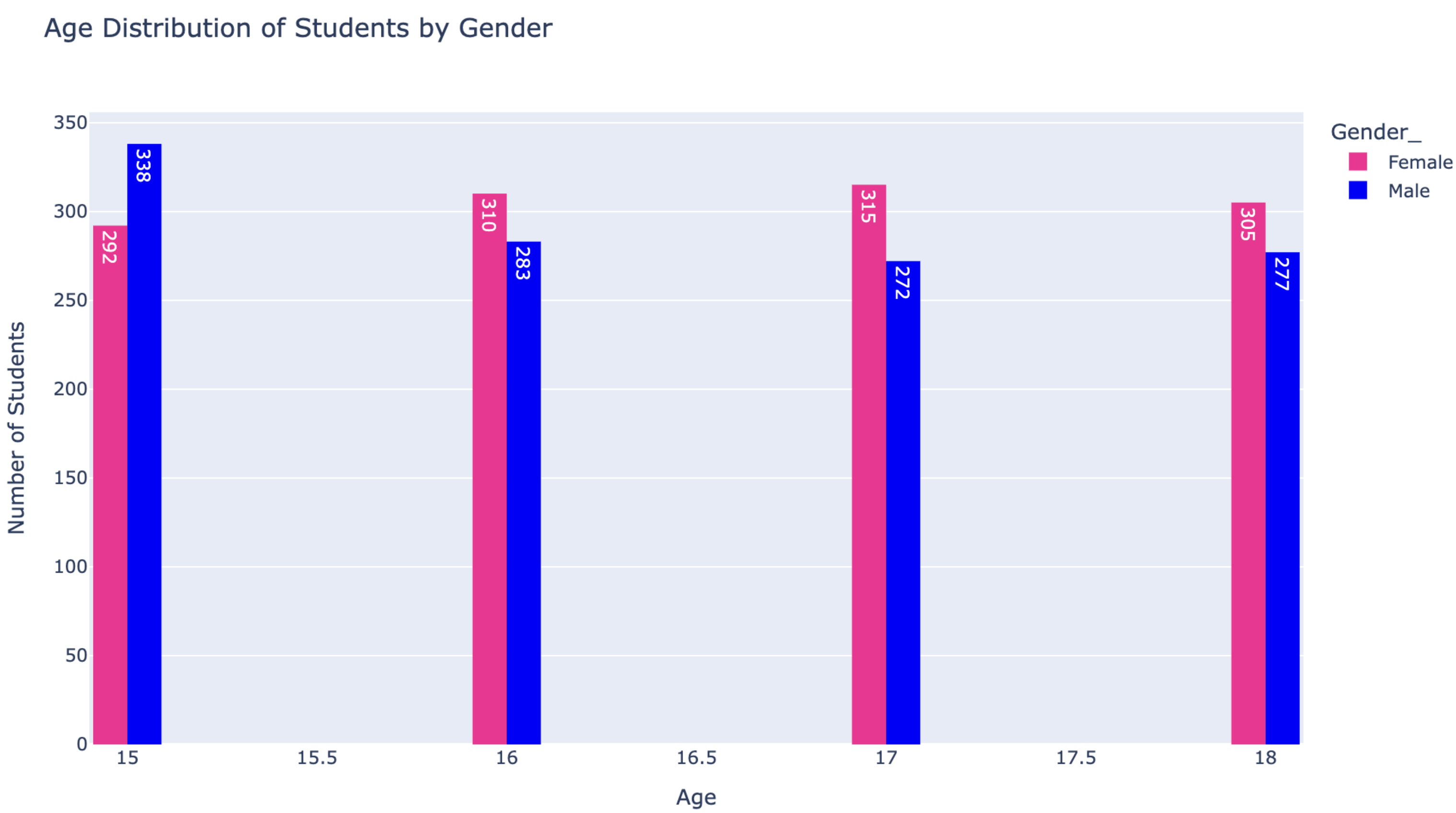
MALE: 1170

FEMALE: 1222

| Gender | count |
|---|---|
| Male | 1170 |
| Female | 1222 |

---

### Age Distribution Of Students By Gender



Insights: The Student's ages are distributed most equally , but in 15' ages there are male students dominating and other ages female students dominating , Female students are more than Male Students at 2.2%.

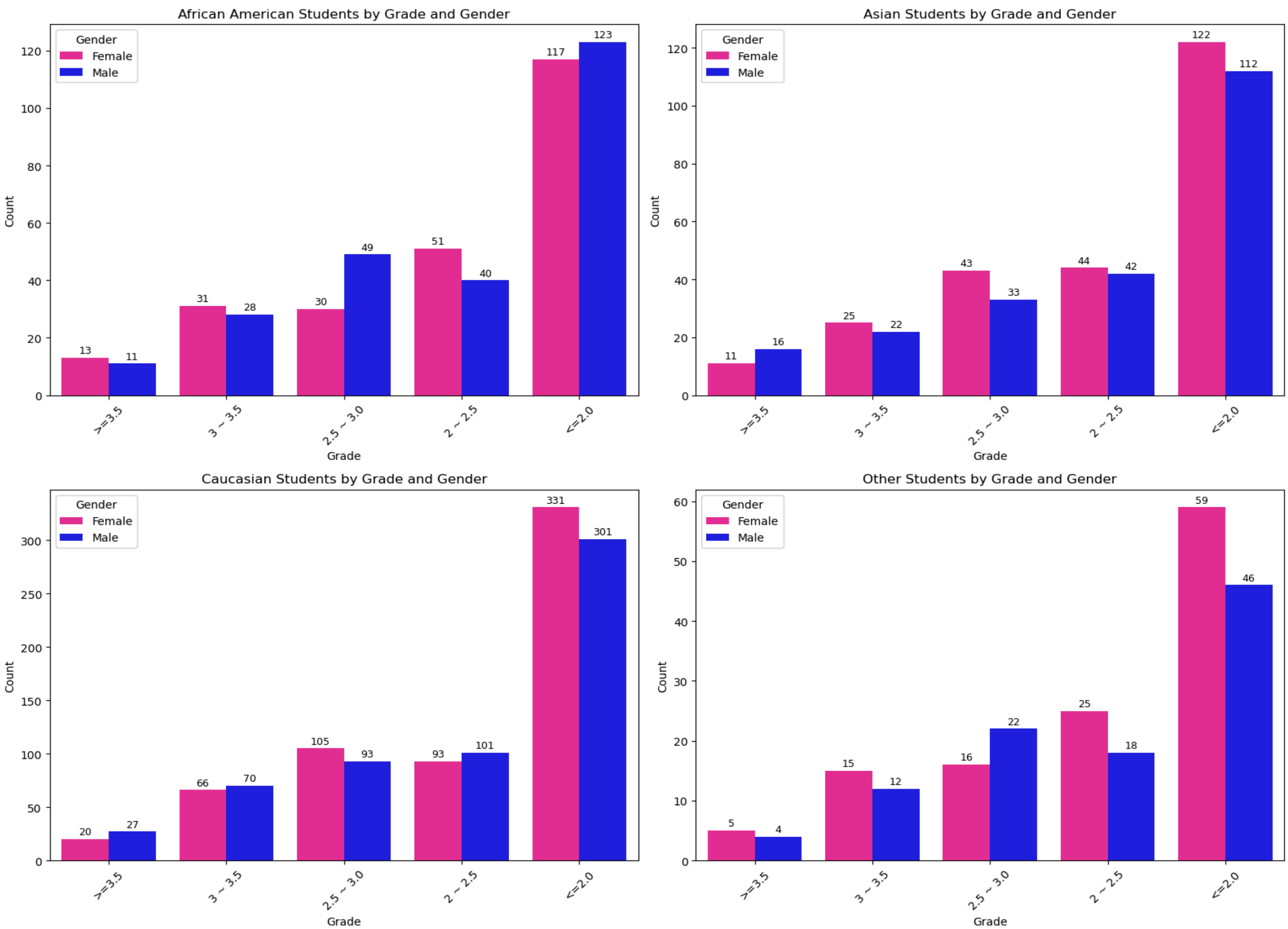
---

**Ethnicity and Their AVG_GPA & COUNT & AGE**

| | Ethnicity | Age | Count | Avg_GPA |
|---|---|---|---|---|
| 0 | African American | 15 | 135 | 1.96 |
| 1 | African American | 16 | 122 | 1.94 |
| 2 | African American | 17 | 128 | 1.90 |
| 3 | African American | 18 | 108 | 2.00 |
| 4 | Asian | 15 | 125 | 1.73 |
| 5 | Asian | 16 | 106 | 1.91 |
| 6 | Asian | 17 | 122 | 1.95 |
| 7 | Asian | 18 | 117 | 2.11 |
| 8 | Caucasian | 15 | 305 | 1.92 |
| 9 | Caucasian | 16 | 304 | 1.90 |
| 10 | Caucasian | 17 | 288 | 1.90 |
| 11 | Caucasian | 18 | 310 | 1.78 |
| 12 | Other | 15 | 65 | 1.99 |
| 13 | Other | 16 | 61 | 1.87 |
| 14 | Other | 17 | 49 | 2.09 |
| 15 | Other | 18 | 47 | 1.84 |

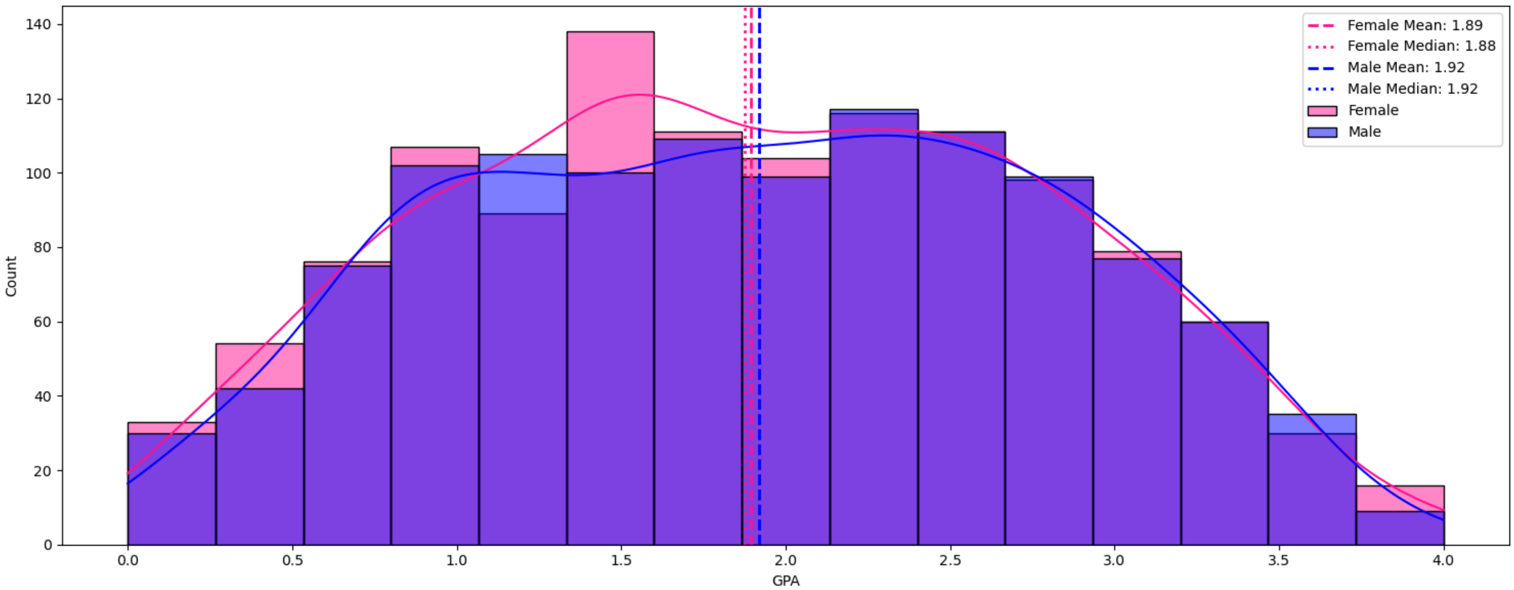
**Insights**: The Smartest Students from Asian, and in second place are the students from Other, And There are a lot of students from Caucasian: 300+ ,  and the less one is from other, and Students from African American and From Asia they are most equal to each other.

## Ethnicity and GPA Distribution By Gender




**Insights**: In this plot , we can notice that There a lot of students with <= 2.0 GPA. And In Caucasian There a lot of  632 Students, Male: 301, Female : 331. And we can see the Trend that the less GPA the higher the number of Students in There.

## GPA Distribution By Gender

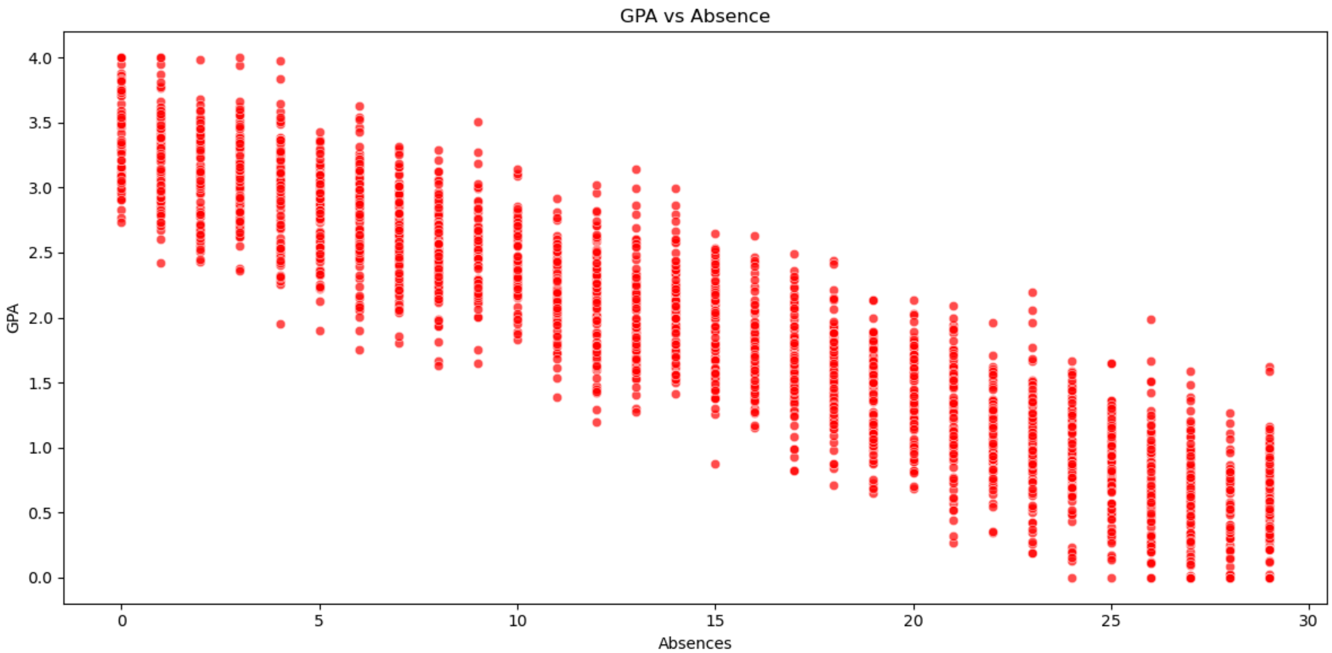



**Insights**:

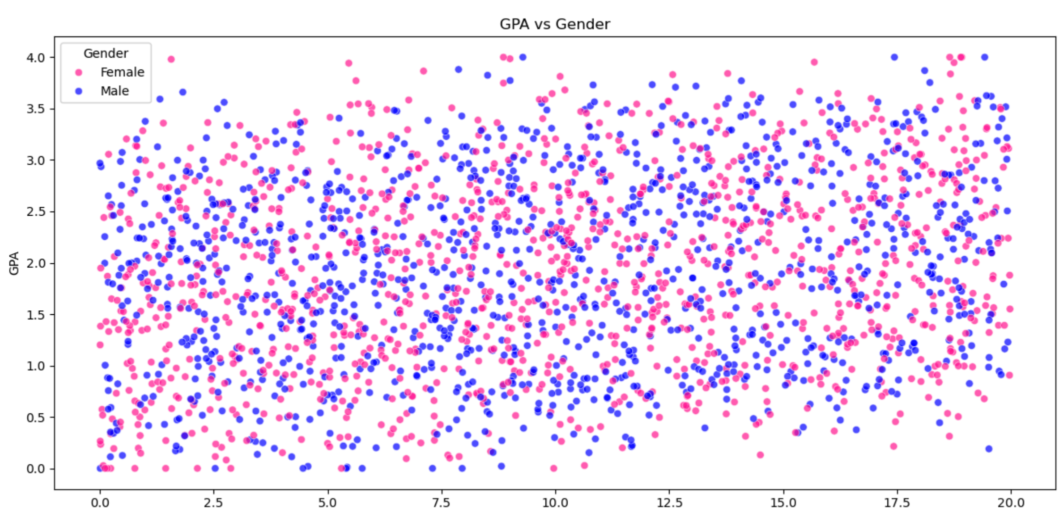
- **Mean GPA**: Males (1.92), Females (1.89)
- **Median GPA**: Males (1.92), Females (1.88)
- Males have a slightly higher GPA on average, but the difference is minimal.
- The overlap in KDE curves suggests similar academic performance patterns between genders.

In summary, GPA distributions are similar across genders, with only slight variations.

## GPA vs Absence & GPA vs Gender



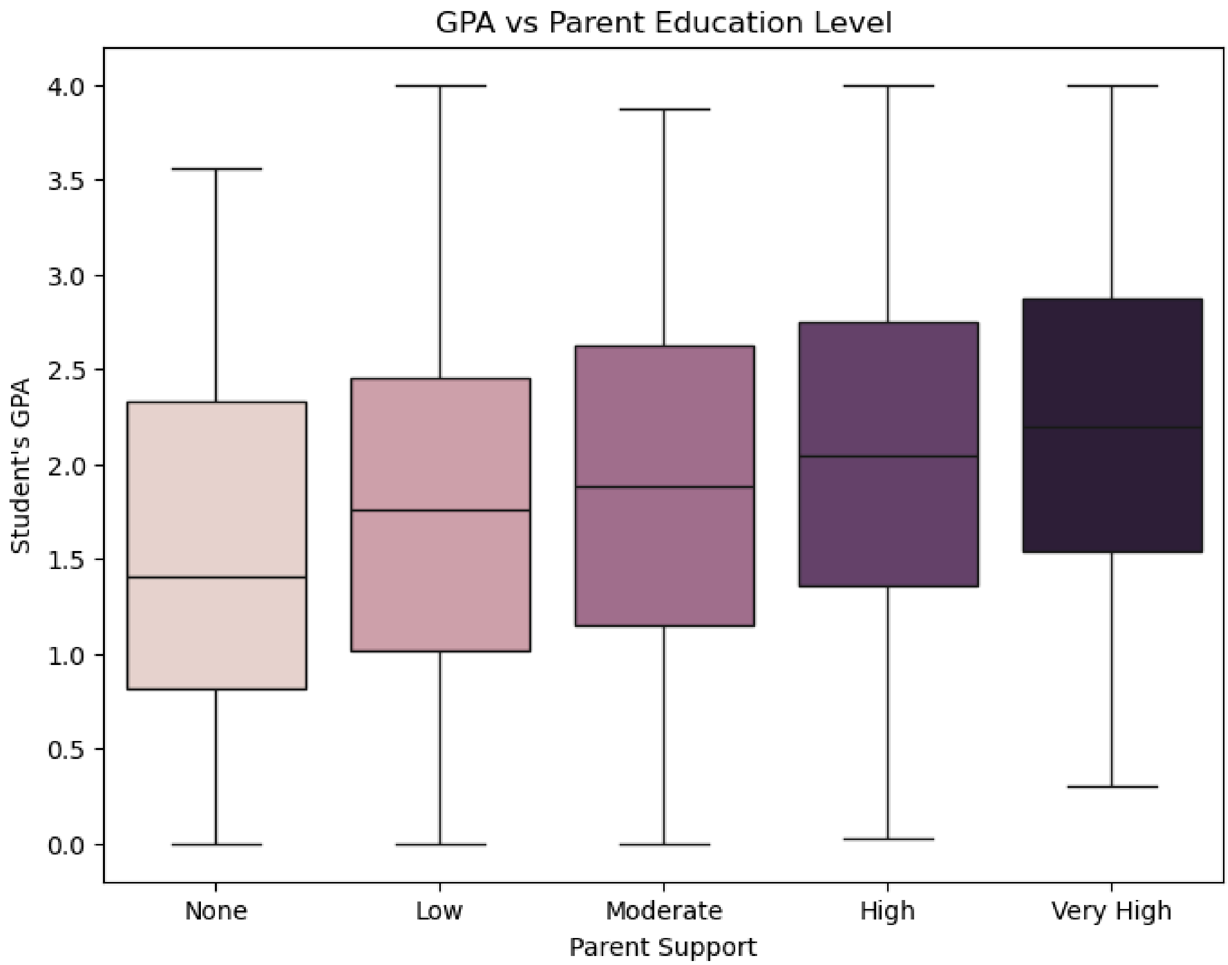

**Insights**: Here we can see the strong Correlation between GPA and Absence, so the less than Absence the Greater GPA score of Students. We can use this information when we will Create the Machine Learning model, such as : Random - Forest, KNN, SVM etc.




**Insights**: There is no Correlation between GPA and Gender.

---

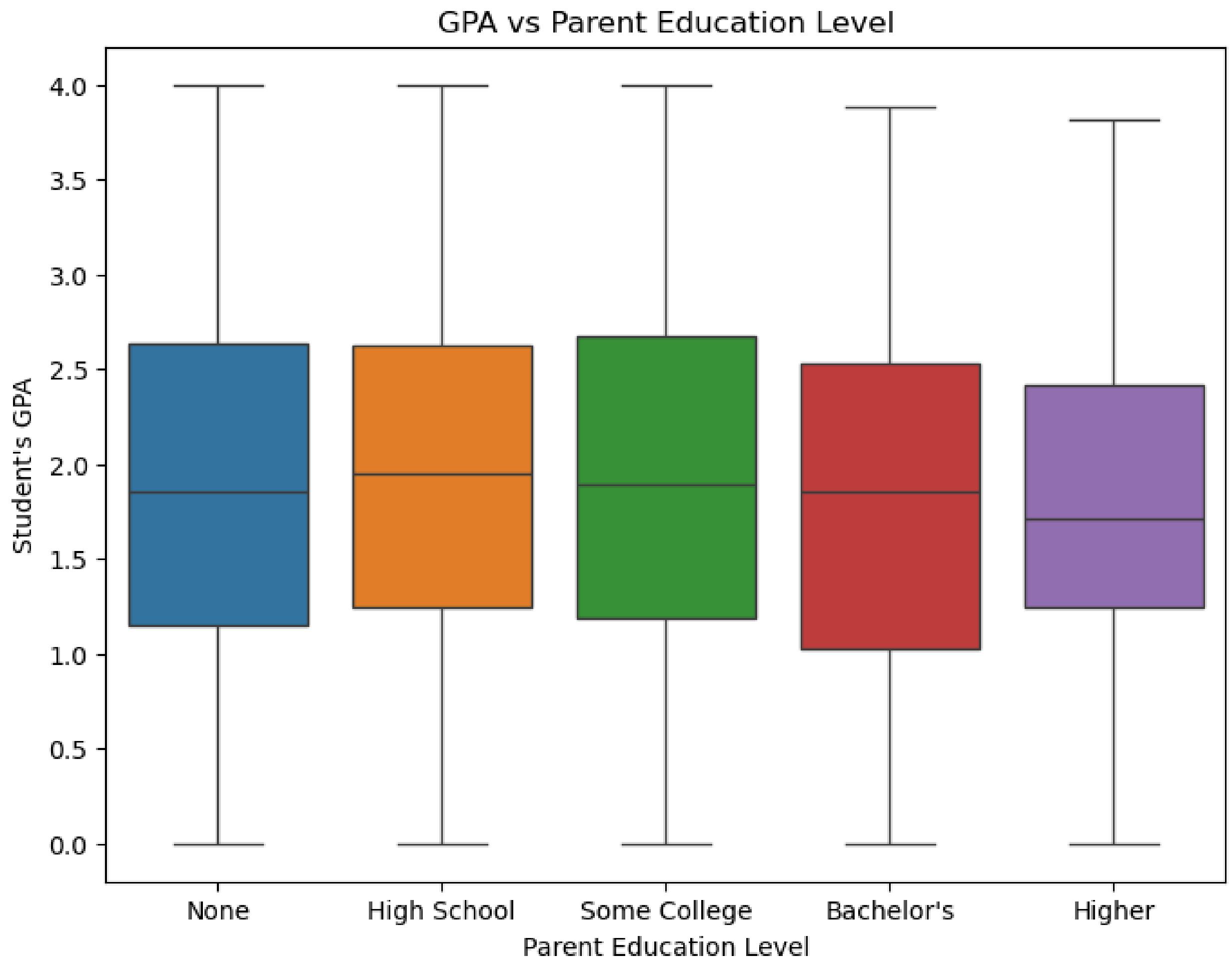
**GPA vs Parent Education Level**




**Insights**: The less Parent Support the less the student's GPA score.

- **Median** of:
    - Very High support : between 2.0 and 2.5
    - High support : less great than 2.0
    - Moderate Support: Between 1.5 and 2.0
    - Low Support: Between 1.5 and 2.0
    - None Support: Between 1.0 and 1.5

**GPA vs Parent Education Level**

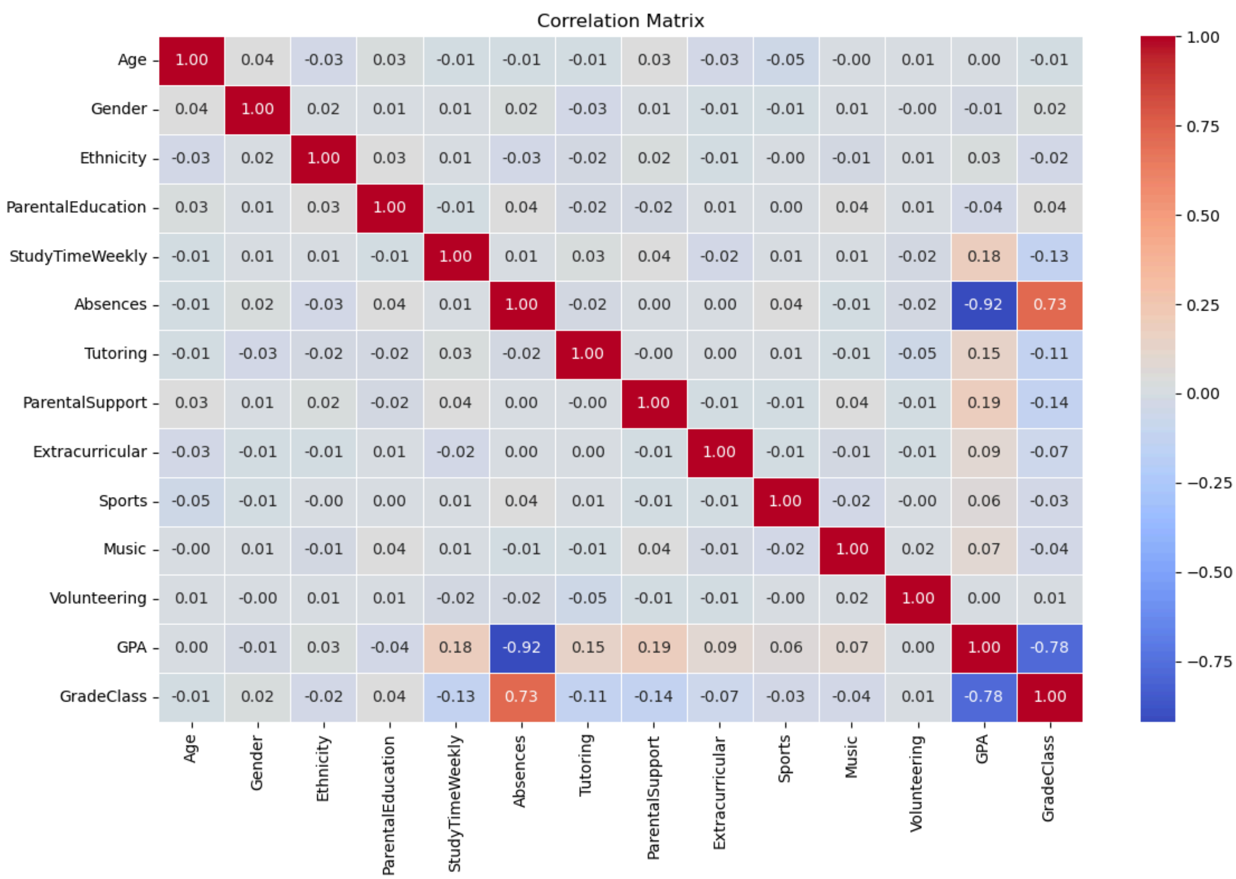



**Insight**: Median GPA is fairly consistent across all education levels, ranging around 1.7–2.0.

- Students whose parents have *no education*, *high school*, or *some college* tend to have slightly higher median GPAs compared to those with *Bachelor's* or *higher* degrees.
- GPA distributions are wide across all groups, with many outliers and similar overall spread.

  **Conclusion**: There's no strong correlation between parent education level and student GPA in this dataset.

## Correlation Matrix




Before the Start the Build ML Model , we should define which features have the most correlation and which features have less correlations.

**Insights**: the most strong correlation between Absence and GradeClass, and the most negative correlation are between GPA and absence, and GPA and Grade Class.

# BUILDING THE ML MODEL

## MODEL EVALUATION

**To predict student academic performance, I experimented with three machine learning models: K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Random Forest. Their performance varied significantly throughout different stages of development.**

## 📌 Initial Results:

1. **KNN**:
    - Accuracy: **62.0%**
    - Struggled with minority classes, especially Grade A and B.
2. **SVM**:
    - Accuracy: **77.9%**
    - Better generalization across most classes.
3. **Random Forest**::
    - Accuracy: **91.0%**
    - Achieved high precision and recall, especially for high-performing (Grade A) and low-performing (Grade F) students.

**⚙️ After Hyperparameter Optimization**

- **KNN** (optimized): **70.4%** accuracy
- **SVM** (optimized): **82.9%** accuracy
- **Random Forest** (optimized): **91.4%** accuracy
    - Feature selection and tuning slightly improved performance for all models.

🧪 **After Adding Synthetic Data (Data Balancing)**

To address class imbalance, synthetic data were generated and added. The results significantly improved across all models:

- **KNN**:
    - Accuracy: **83.0%**
    - Major improvement in detecting underrepresented classes.
- **SVM**:
    - Accuracy: **86.0%**
    - Balanced performance across all classes.
- **Random Forest**:
    - Accuracy: **93.0%**

✅ **Final Optimized Results (With Synthetic Data)**

- **KNN:** 92.7%
- **SVM (optimized with `C=10`, `gamma=0.1`, `kernel='rbf'`):** 93.6%
- **Random Forest:** 93.2%

**Both Random Forest and Optimized SVM reached over 93% accuracy, demonstrating excellent classification performance and robustness on the balanced dataset. These models are well-suited for academic performance prediction tasks involving complex and imbalanced data.**

## Performance Insights

1. ✅ **Consistency**:
   - The models demonstrated stable and consistent performance across different experiments. Especially after data balancing and hyperparameter tuning, **F1-scores** for all major classes consistently hovered around **90%**, indicating reliable predictive capabilities.
2. 🌍 **Generalization**:
   - The models, particularly **Random Forest** and **SVM**, maintained strong generalization to unseen data. Their performance remained high not only on the training set but also on the test set, which reflects good model robustness and low risk of overfitting.
3. 📉 **Validation Trends**

   - Before optimization and data balancing, some models (e.g., **KNN**) struggled with underrepresented classes, leading to uneven F1-scores. However, after incorporating synthetic data, all models showed **more balanced performance across all grade categories**, with improved precision and recall metrics, especially for Grades A, B, and C.

## Challenges Observed

1. ⚖️ **Unbalanced Dataset**
   - Initially, the dataset was **heavily imbalanced**, with a significantly higher number of students falling into lower performance categories (especially Grade F). This caused the models to **bias toward the majority class**, making it difficult to accurately predict students with higher grades (e.g., Grade A or B).
2. 🧠 **Initial Overfitting**
   - In early training stages (before balancing), models like **KNN** and **SVM** showed signs of **overfitting**, especially to the dominant class. This led to **poor performance on minority classes**, resulting in low precision and recall for high-performing students.
3. 📉 **Validation Performance Variance**
   - There was noticeable **fluctuation in validation accuracy and F1-scores across different classes**, indicating instability and the need for data balancing and fine-tuning to achieve fair representation for all grade categories.

4. 🔁 **Precision vs. Recall Trade-off**
    - Some models achieved high recall for underperforming students, correctly identifying those who may need academic support. However, this sometimes came at the cost of lower precision, producing more false positives — which could lead to unnecessary interventions for students performing adequately.

# 📌 Initial Results(Accuracy And Confusion Matrix):

- KNN:

```
=== KNN Classification ===
Accuracy: 0.6200417536534447

Classification Report:
              precision    recall  f1-score   support

         0.0       0.33      0.14      0.19        22
         1.0       0.28      0.22      0.25        49
         2.0       0.38      0.46      0.41        85
         3.0       0.42      0.36      0.39        86
         4.0       0.84      0.90      0.87       237

    accuracy                           0.62       479
   macro avg       0.45      0.42      0.42       479
weighted avg       0.60      0.62      0.61       479
```

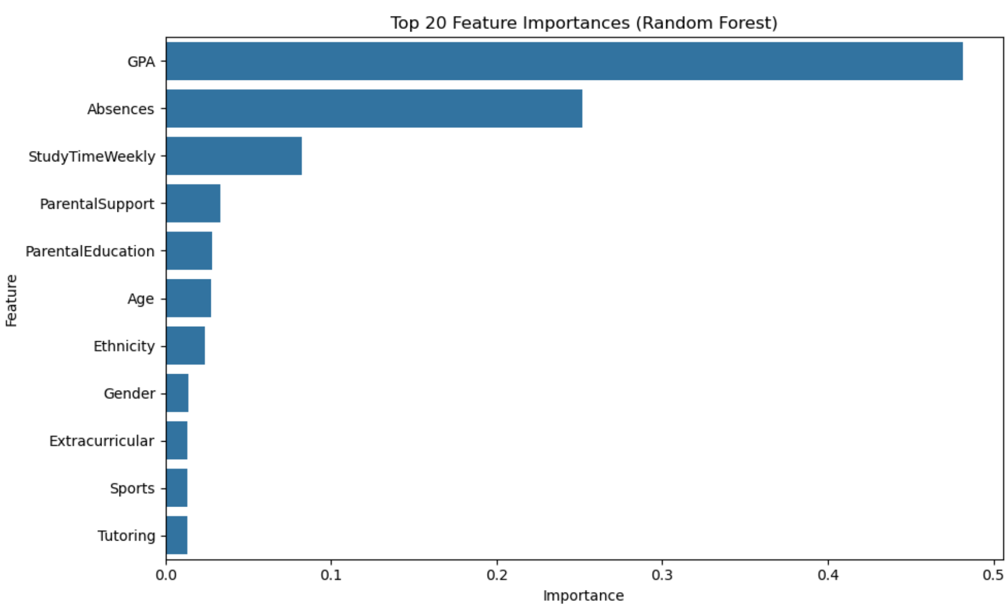
- SVM

```
=== SVM Classification ===
Accuracy: 0.778705636743215

Classification Report:
              precision    recall  f1-score   support

         0.0       0.75      0.14      0.23        22
         1.0       0.58      0.67      0.62        49
         2.0       0.70      0.67      0.68        85
         3.0       0.67      0.63      0.65        86
         4.0       0.89      0.95      0.92       237

    accuracy                           0.78       479
   macro avg       0.72      0.61      0.62       479
weighted avg       0.78      0.78      0.77       479
```

- Random Forest

```
=== SVM Classification ===
Accuracy: 0.778705636743215

Classification Report:
              precision    recall  f1-score   support

         0.0       0.75      0.14      0.23        22
         1.0       0.58      0.67      0.62        49
         2.0       0.70      0.67      0.68        85
         3.0       0.67      0.63      0.65        86
         4.0       0.89      0.95      0.92       237

    accuracy                           0.78       479
   macro avg       0.72      0.61      0.62       479
weighted avg       0.78      0.78      0.77       479
```

### ⚙️ After Hyperparameter Optimization




I had to collect the only Features that were more than 0.01 , to improve my model.

- **KNN**:
    - For KNN , I’ve used a GridSearchCV

```
=== Optimized KNN Results ===
Accuracy: 0.7035490605427975

Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      0.14      0.24        22
         1.0       0.53      0.47      0.50        49
         2.0       0.55      0.56      0.55        85
         3.0       0.53      0.35      0.42        86
         4.0       0.81      0.98      0.89       237

    accuracy                           0.70       479
   macro avg       0.68      0.50      0.52       479
weighted avg       0.69      0.70      0.68       479
```

- **SVM**:
    - For SVM , I’ve used a Also GridSearchCV

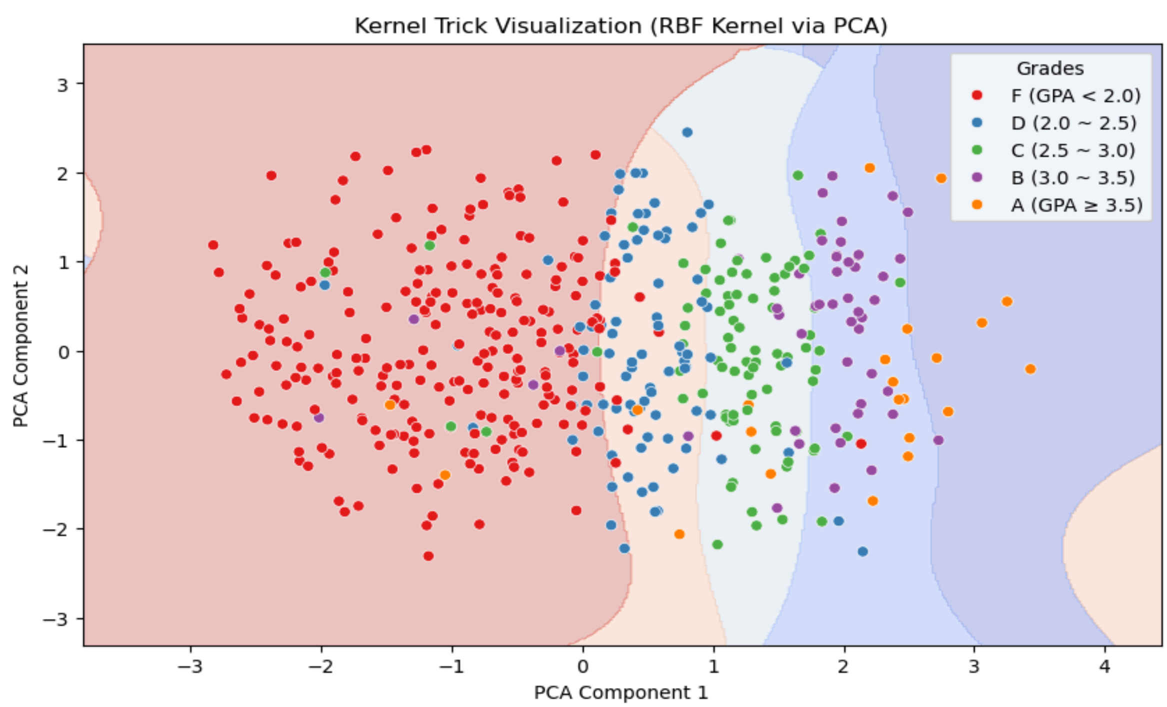
```
=== Optimized SVM Results ===
Accuracy: 0.8288100208768268

Classification Report:
              precision    recall  f1-score   support

         0.0       0.00      0.00      0.00        22
         1.0       0.58      0.80      0.67        49
         2.0       0.79      0.73      0.76        85
         3.0       0.79      0.78      0.78        86
         4.0       0.92      0.97      0.94       237

    accuracy                           0.83       479
   macro avg       0.62      0.65      0.63       479
weighted avg       0.80      0.83      0.81       479
```

- **Random-Forest**:
    - For Random-Forest I've Used a Random Forest Classifier

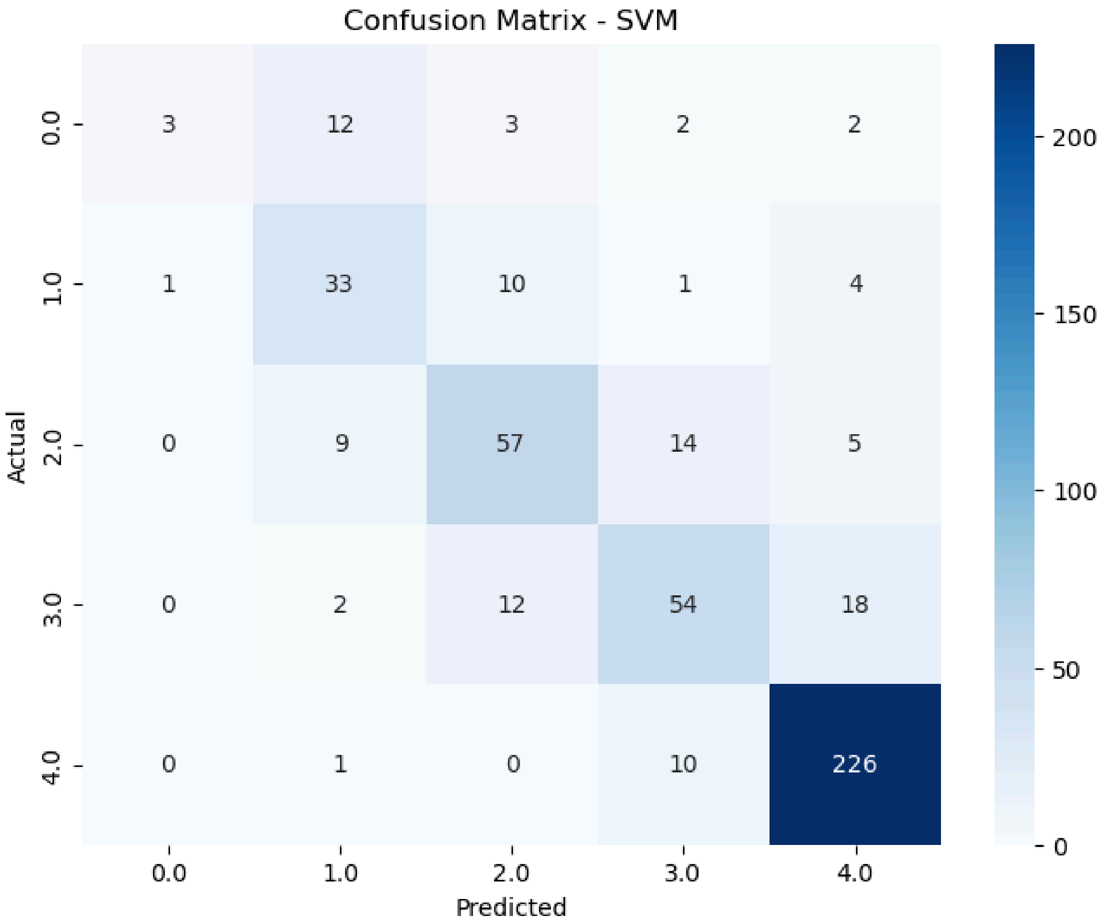
```
=== Random Forest (Reduced Features) ===
Accuracy: 0.9144050104384134

Classification Report:
              precision    recall  f1-score   support

         0.0       0.85      0.50      0.63        22
         1.0       0.83      0.88      0.85        49
         2.0       0.94      0.86      0.90        85
         3.0       0.89      0.90      0.89        86
         4.0       0.94      0.99      0.96       237

    accuracy                           0.91       479
   macro avg       0.89      0.82      0.85       479
weighted avg       0.91      0.91      0.91       479
```

**Visualisation:**

- KNN:



- **SVM:**

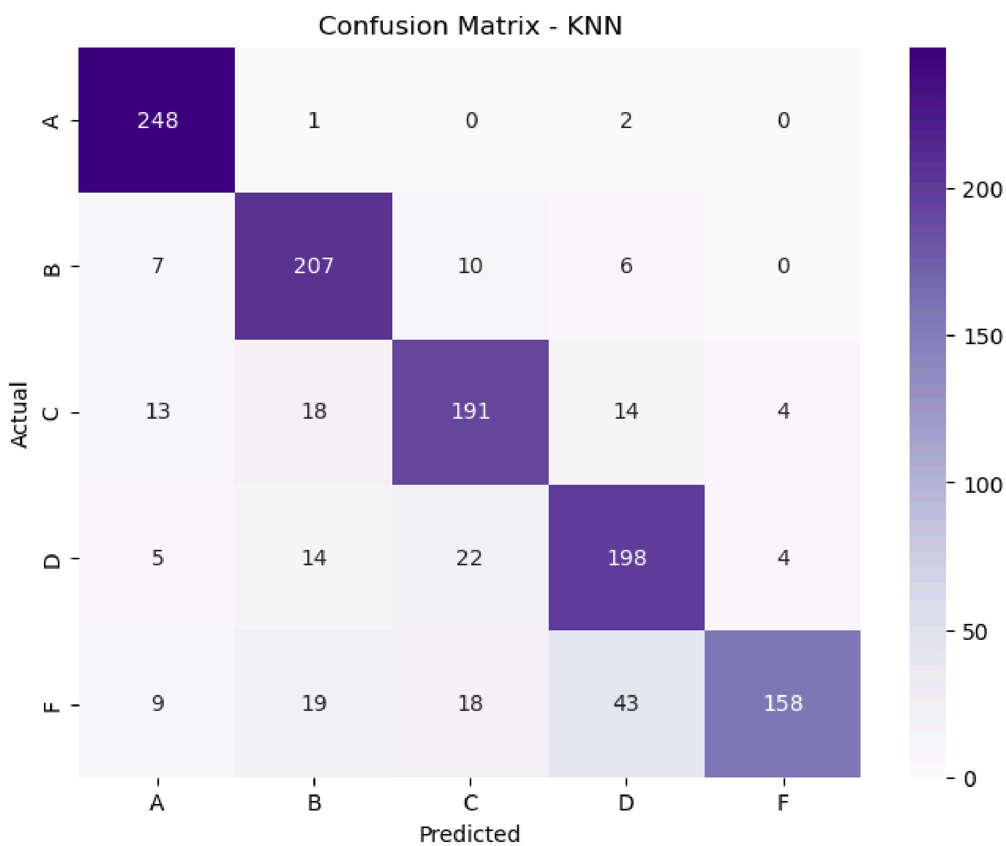



So Here We can notice that our Data set is not Balanced and then I decided to ADD THE SYNTHETIC DATA to Balancing the Data Set.

### 🧪 After Adding Synthetic Data (Data Balancing)

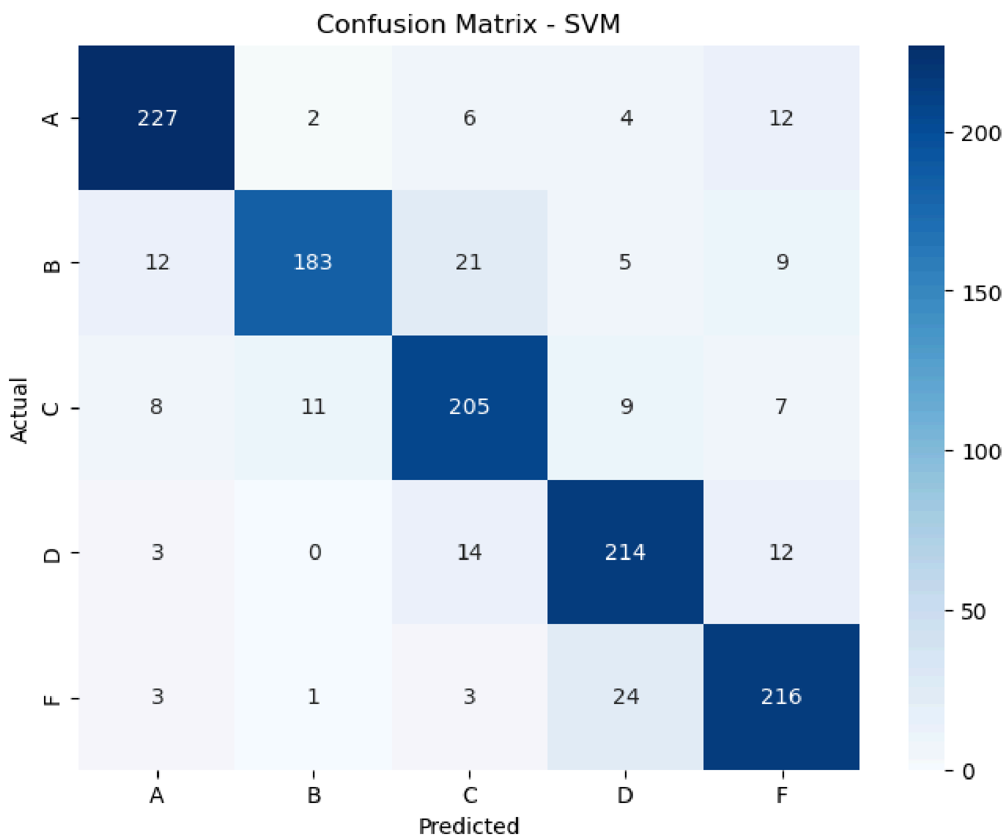
- KNN:

```
=== KNN Classification AFTER BALANCING===
Accuracy: 0.83%
KNN: Classification Report:
              precision    recall  f1-score   support

         0.0       0.88      0.99      0.93       251
         1.0       0.80      0.90      0.85       230
         2.0       0.79      0.80      0.79       240
         3.0       0.75      0.81      0.78       243
         4.0       0.95      0.64      0.77       247
```



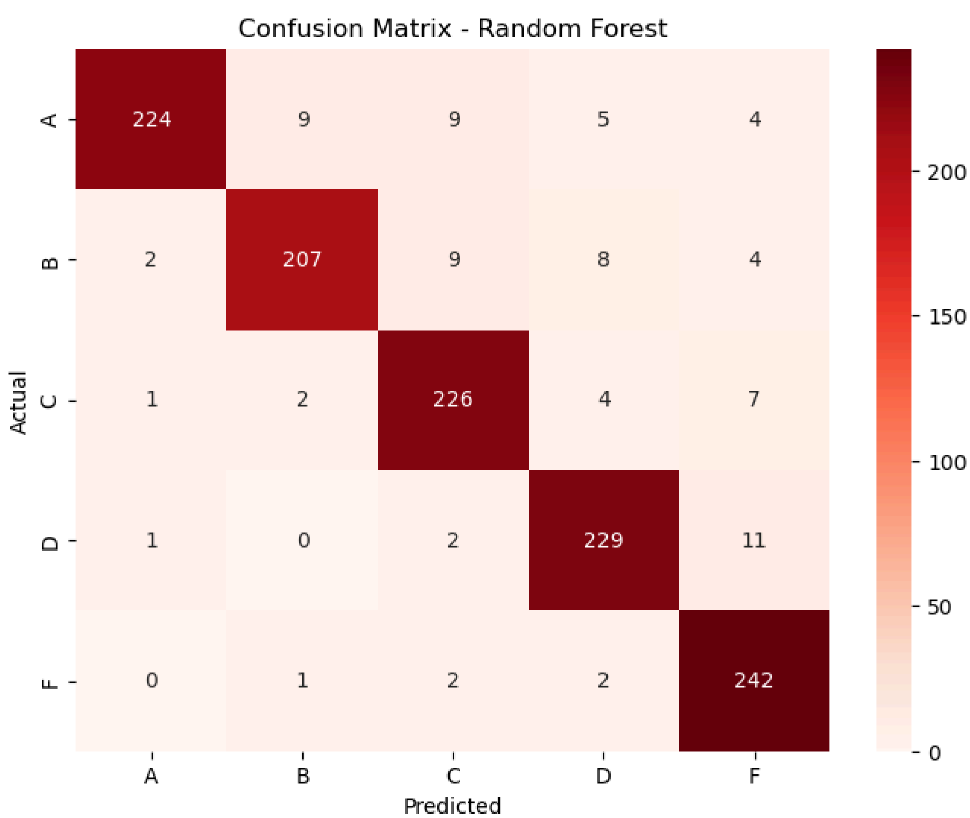
- **SVM:**

```
=== SVM Classification AFTER BALANCING===
SVM Accuracy: 0.86%
SVM Classification Report:
               precision    recall  f1-score   support

         0.0       0.90      0.90      0.90       251
         1.0       0.93      0.80      0.86       230
         2.0       0.82      0.85      0.84       240
         3.0       0.84      0.88      0.86       243
         4.0       0.84      0.87      0.86       247

    accuracy                           0.86      1211
   macro avg       0.87      0.86      0.86      1211
weighted avg       0.87      0.86      0.86      1211
```



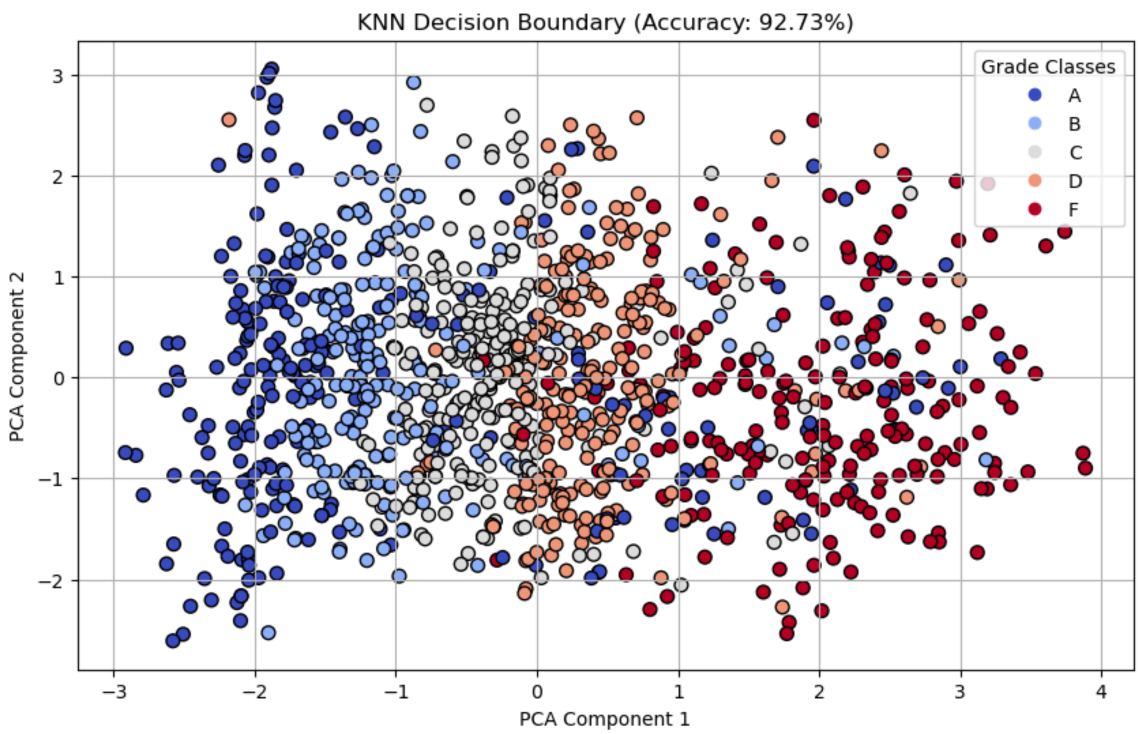
## Random Forest

```
=== Random Forest AFTER BALANCING ===
Accuracy: 0.93%
Random Forest Classification Report:
              precision    recall  f1-score   support

         0.0       0.98      0.90      0.94       251
         1.0       0.95      0.90      0.92       230
         2.0       0.91      0.94      0.93       240
         3.0       0.92      0.95      0.93       243
         4.0       0.91      0.98      0.94       247

    accuracy                           0.93      1211
   macro avg       0.93      0.93      0.93      1211
weighted avg       0.93      0.93      0.93      1211
```

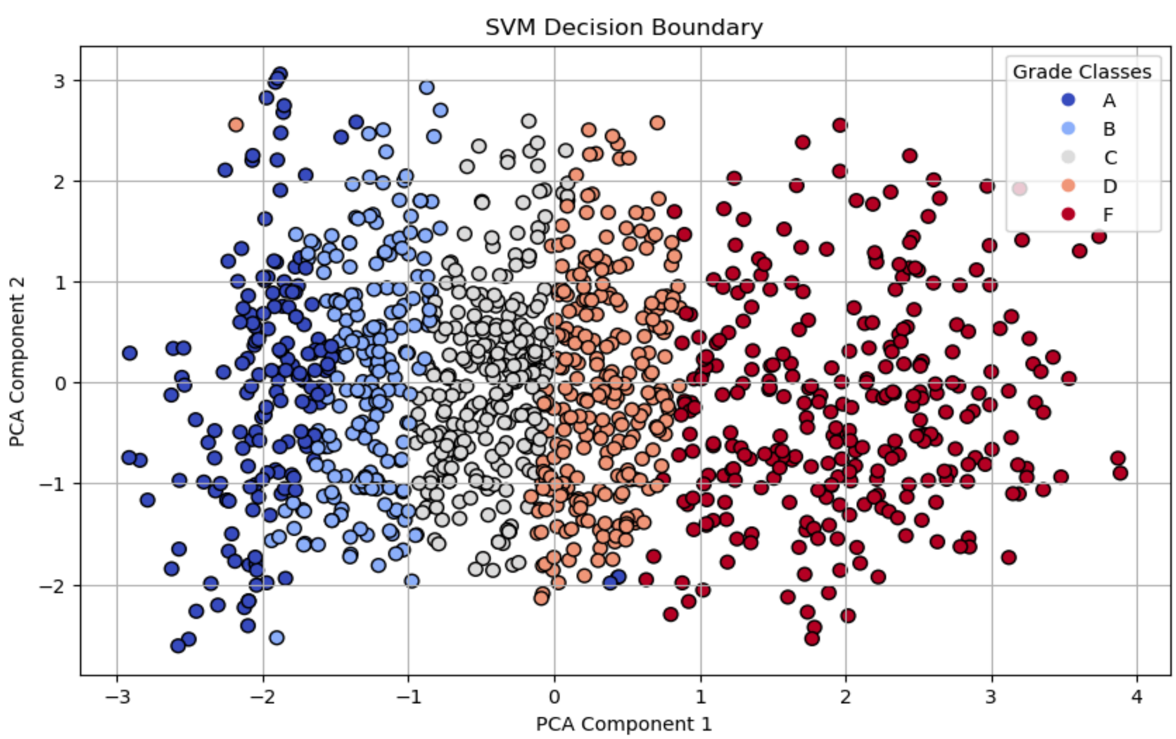


## 🧪 Optimisation After Adding Synthetic Data (Data Balancing)

- **KNN:**

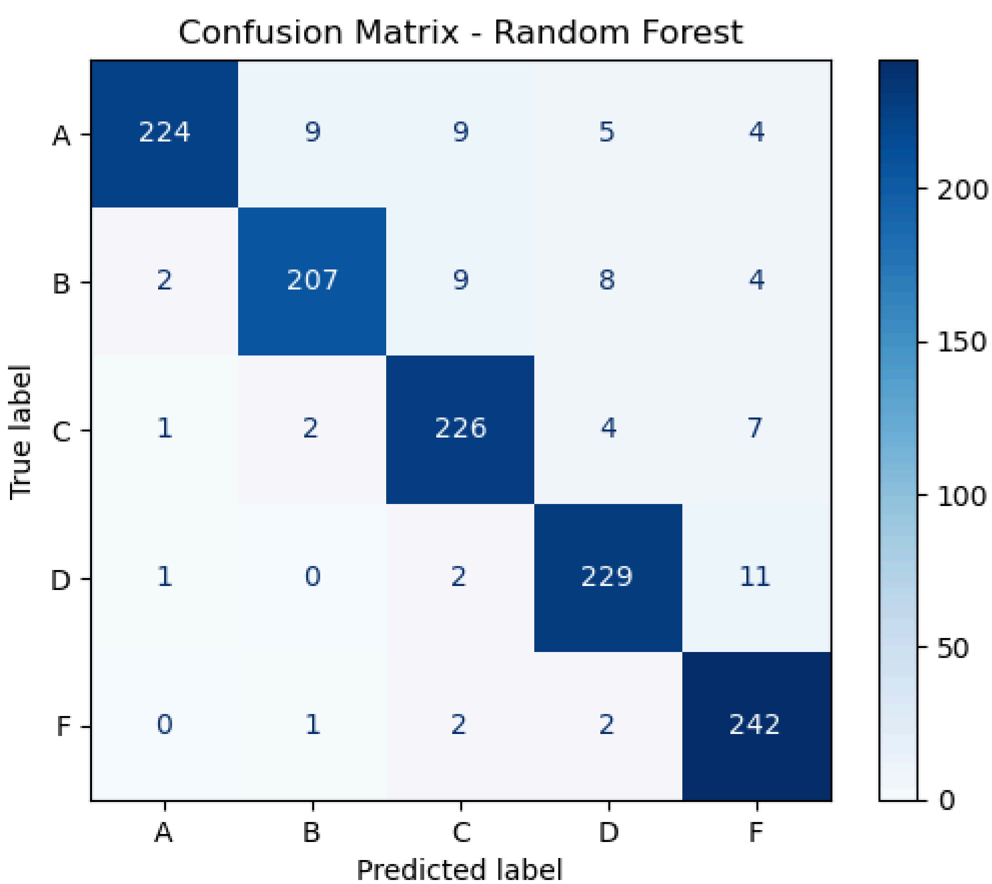



- **SVM with 75.06% Of Accuracy**



- **Random Forest**

```
Test accuracy: 93.15%
Classification report:
              precision    recall  f1-score   support

         0.0       0.98      0.89      0.94       251
         1.0       0.95      0.90      0.92       230
         2.0       0.91      0.94      0.93       240
         3.0       0.92      0.94      0.93       243
         4.0       0.90      0.98      0.94       247

    accuracy                           0.93      1211
   macro avg       0.93      0.93      0.93      1211
weighted avg       0.93      0.93      0.93      1211
```



# EXPLANATION OF MODEL

## 🧰 Libraries and Tools Used

**scikit-learn (sklearn)**

The main library used for training, evaluating, and optimizing machine learning models. It provided:

- **Model training and classification** using:
    - `KNeighborsClassifier` for KNN
    - `SVC` for SVM
    - `RandomForestClassifier` for Random Forest
- **Model evaluation** through:
    - `accuracy_score`, `classification_report`, and `confusion_matrix` from `sklearn.metrics`
- **Model optimization** with tools like `GridSearchCV` from `sklearn.model_selection` to find the best hyperparameters for each model
- **Data preprocessing and splitting** using `train_test_split`

**Pandas**

Used for data manipulation and analysis, including:

- Reading and processing CSV files
- Exploring and transforming features

- Handling missing values and encoding categorical variables

**Matplotlib & Seaborn**

Used for data visualization and exploratory data analysis:

- **Matplotlib** (`pyplot`) was used to plot learning curves, accuracy trends, and performance comparisons

- **Seaborn** was used to visualize the **confusion matrix**, class distributions, and correlation heatmaps for feature relationships

**Plotly**

Provided interactive visualizations for more detailed performance analysis and presentation, such as dynamic plots of model performance metrics across different settings or hyperparameters.

---

## CONCLUSION

This project focused on predicting student academic performance using traditional machine learning models such as SVM, KNN, and Random Forest. Data preprocessing, feature analysis, and class balancing were carefully handled to improve model reliability. With the help of evaluation metrics like accuracy, confusion matrix, and F1-score, the models demonstrated solid performance. Visualizations using libraries like Matplotlib, Seaborn, and Plotly provided deeper insights into feature relationships and performance trends. Overall, the project offers a practical framework for identifying students at academic risk and supporting data-driven educational interventions.

## REFERENCES

1. Link to the Git repository
2. Data Set From Kaggle