A PROJECT REPORT

ON

"STUDENT PERFORMANCE PREDICTION"

Submitted in partial fulfillment of

BACHELOR OF TECHNOLOGY DEGREE

Session: 2024-2025

DEPARTMENT OF CSE (AIML)

KIET Group of Institutions,

Ghaziabad

Submitted To:

MR. ABHISHEK SHUKLA

Submitted By:

PRASHANT RAJPUT (202401100400141)

1. Introduction

Predicting student performance is an essential task in educational data mining. It enables educators to identify at-risk students early and implement targeted interventions to improve academic outcomes. In this project, we aim to predict whether a student will **pass or fail** based on various factors such as attendance, study habits, and past academic scores. We use machine learning techniques with a focus on classification.

□ 2. Methodology

2.1 Dataset

The dataset contains anonymized student records, including features such as:

- Academic scores
- Attendance
- Study hours
- Participation in extracurricular activities
- Lifestyle and habits

The target variable is whether a student **passes (1)** or **fails (0)** based on their `GradeClass`.

2.2 Preprocessing Steps

- Removed non-predictive features (`StudentID`)
- Created binary target `Result` (1 = pass, 0 = fail)
- Dropped less informative features (like `Music`, `Sports`, etc.)
- Applied feature scaling using `StandardScaler`
- Balanced the dataset using `SMOTE` (Synthetic Minority Oversampling Technique)

2.3 Model

We used an optimized **XGBoost Classifier** due to its high performance on classification tasks.

2.4 Evaluation

The model is evaluated using:

- Accuracy
- Precision
- Recall
- Confusion Matrix

3. Code

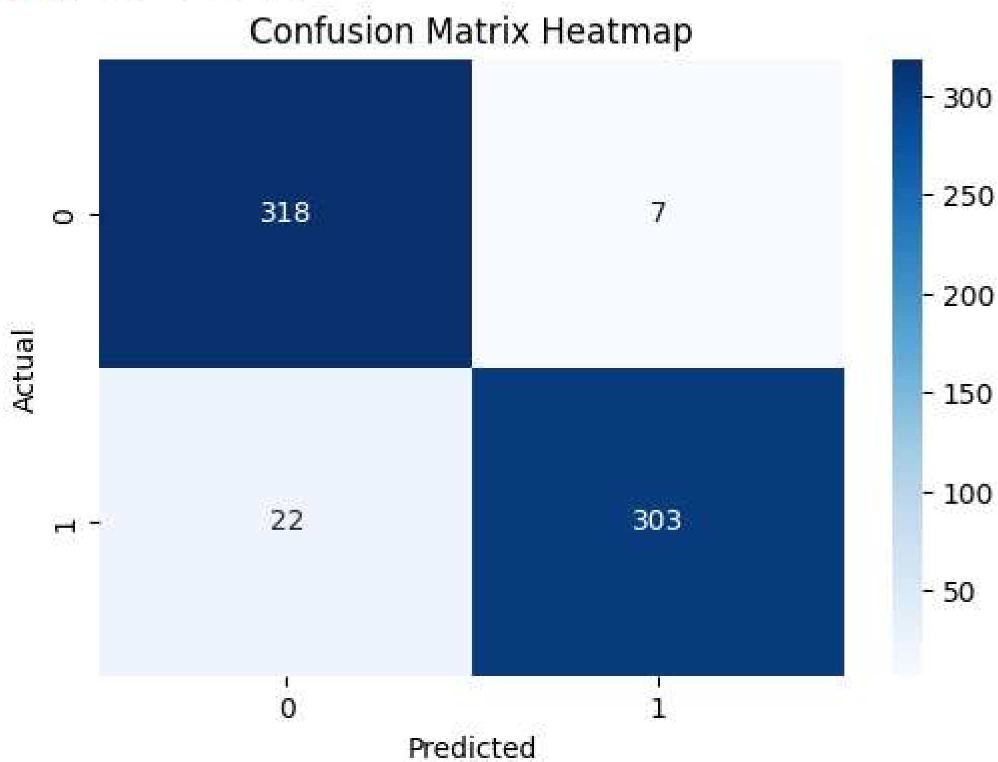
```python
# Install required packages
!pip install xgboost imbalanced-learn seaborn scikit-learn pandas
matplotlib --quiet

# Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, precision_score, recall_score,
confusion_matrix
from xgboost import XGBClassifier
from imblearn.over_sampling import SMOTE

# Upload CSV file
from google.colab import files
uploaded = files.upload()

# Load the dataset
file_name = list(uploaded.keys())[0]
df = pd.read_csv(file_name)

# Preprocessing
df = df.drop(columns=['StudentID'])  # Drop ID column

# Define target: 1 = Pass (GradeClass <= 2), 0 = Fail
df['Result'] = df['GradeClass'].apply(lambda x: 1 if x <= 2 else 0)
df = df.drop(columns=['GradeClass'])

# Optional: Drop low-importance features
drop_cols = ['Music', 'Volunteering', 'Sports']  # You can adjust this
df = df.drop(columns=drop_cols)

# Features and label
X = df.drop(columns=['Result'])
y = df['Result']

# Feature Scaling
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

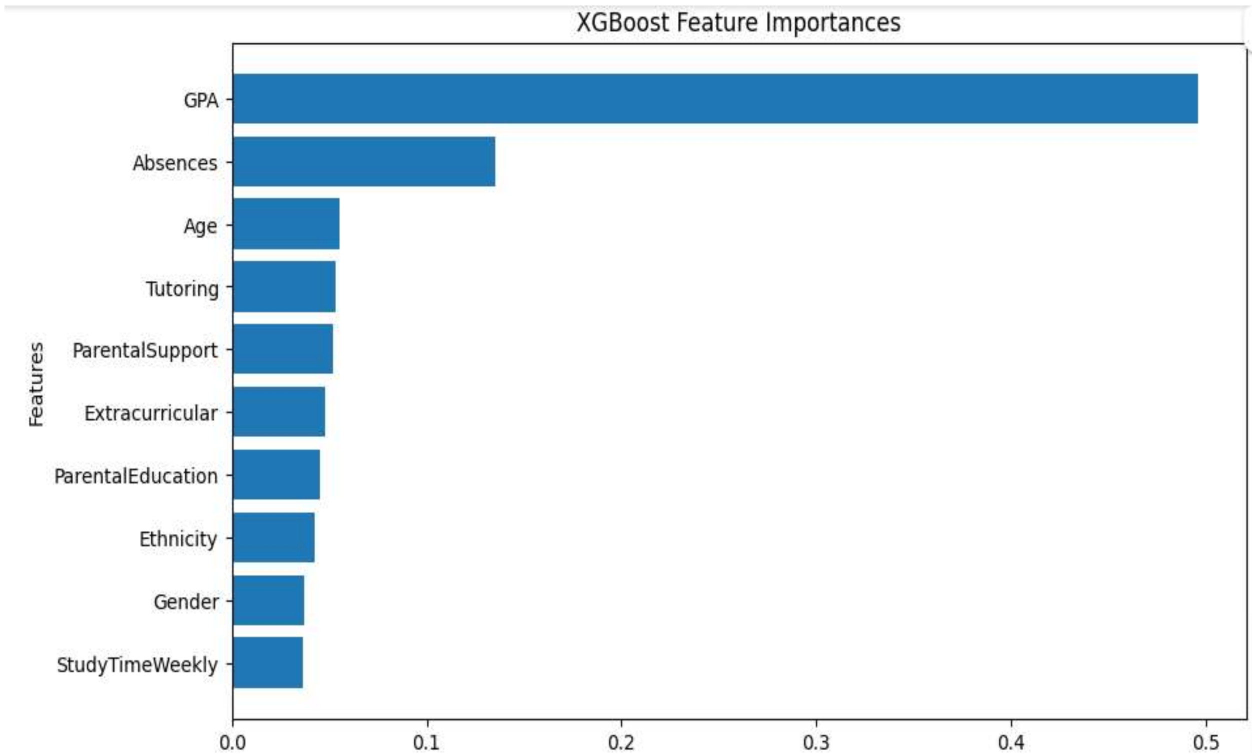
```
# Balance using SMOTE
smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X_scaled, y)

# Split dataset
X_train, X_test, y_train, y_test = train_test_split(
    X_resampled, y_resampled, test_size=0.2, random_state=42,
stratify=y_resampled
)

# Optimized XGBoost model
model = XGBClassifier(
    n_estimators=300,
    max_depth=6,
    learning_rate=0.05,
    subsample=0.9,
    colsample_bytree=0.9,
    min_child_weight=3,
    reg_alpha=0.1,
    reg_lambda=1,
    use_label_encoder=False,
    eval_metric='logloss',
    random_state=42
)

# Train the model
model.fit(X_train, y_train)

# Predictions
y_pred = model.predict(X_test)

# Evaluation metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)

print("\n□ Final Model Evaluation:")
print(f"✅Accuracy : {accuracy:.4f}")
print(f"□ Precision: {precision:.4f}")
print(f"□ Recall    : {recall:.4f}")

# Confusion matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 4))
```

```python
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix Heatmap")
plt.show()

# Feature importance
importances = model.feature_importances_
feature_names = X.columns
sorted_idx = np.argsort(importances)

plt.figure(figsize=(10, 6))
plt.barh(range(len(importances)), importances[sorted_idx], align='center')
plt.yticks(range(len(importances)), feature_names[sorted_idx])
plt.title("XGBoost Feature Importances")
plt.xlabel("Importance")
plt.ylabel("Features")
plt.show()
```

4. Output / Results

```
📊 Final Model Evaluation:
✅ Accuracy : 0.9554
🎯 Precision: 0.9774
📢 Recall   : 0.9323
```





5. References

- [XGBoost Documentation]
- [imbalanced-learn (SMOTE)]
- Scikit-learn: https://scikit-learn.org
- Dataset: Provided as `Student Performance Prediction.csv`