**Assesment Report**

on

# Student Performance Prediction

submitted as partial fulfillment for the award of

# BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

## CSE (AIML)-B

By

Krrish Kumar (202401100400111)

**Under the supervision of**

"Abhishek Shukla"

## KIET Group of Institutions, Ghaziabad

Affiliated to

**Dr. A.P.J. Abdul Kalam Technical University, Lucknow**

(Formerly UPTU)

**May, 2025**

# INTRODUCTION

Academic performance is a multifaceted outcome influenced by a variety of factors, including attendance patterns, prior academic achievements, and study habits. Understanding and predicting student performance is of significant interest to educators, institutions, and students themselves. Early identification of students who may be at risk of underperforming can facilitate timely interventions and support mechanisms, ultimately aiming to improve educational outcomes.

Traditional methods of assessing student potential often rely on past performance metrics and qualitative teacher assessments. However, the increasing availability of educational data, encompassing attendance records, historical grades, engagement in learning activities, and even self-reported study behaviors, presents an opportunity to leverage data-driven approaches for more nuanced and potentially predictive insights.

# METHEDOLOGY

To achieve the objective of predicting student GPA, this study will employ a supervised machine learning approach, specifically utilizing regression techniques.

The study will utilize a dataset containing information on student characteristics and their corresponding GPA.

**Data Loading:** The dataset will be loaded into a pandas DataFrame for efficient manipulation and analysis using Python.

**Handling Missing Values:** Any missing values within the dataset will be addressed using appropriate imputation techniques, such as replacing them with the mean or median of the respective feature, to ensure the integrity of the modeling process.

**Feature Scaling:** To prevent features with larger numerical ranges from dominating the learning process of the regression model, numerical features will be scaled using the StandardScaler from scikit-learn. This will transform the features to have zero mean and unit variance.

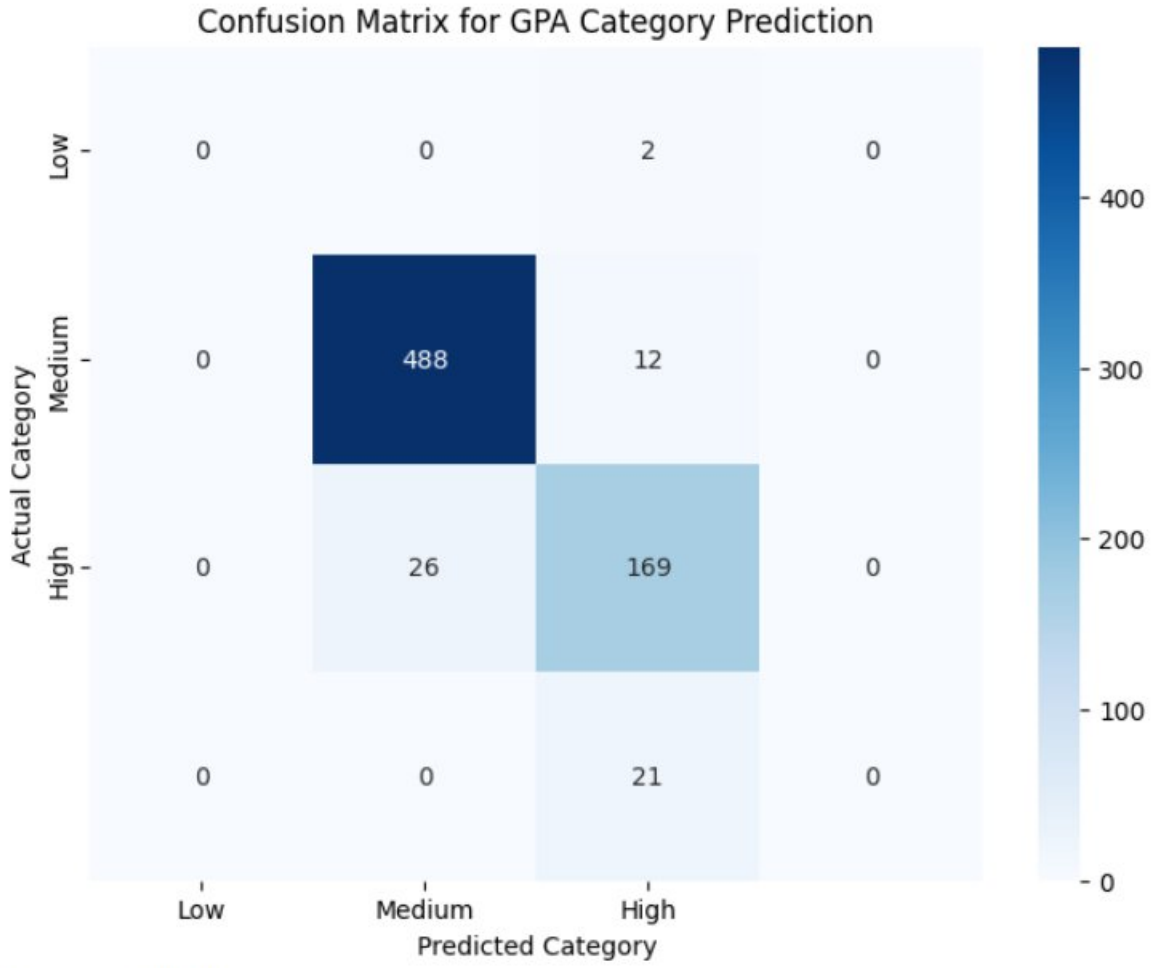
# CODE

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression  # Using a classifier now
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from google.colab import files
import numpy as np
# Upload the CSV file
uploaded = files.upload()
file_name = list(uploaded.keys())[0]
# Load data
try:
    data = pd.read_csv(file_name)
except FileNotFoundError:
    print(f"Error: '{file_name}' not found after upload.")
    exit()
except Exception as e:
    print(f"Error reading the CSV file: {e}")
    exit()
# Feature and target columns
feature_cols = ['Absences', 'ParentalEducation', 'Tutoring', 'Age', 'StudyTimeWeekly',
'ParentalSupport', 'Extracurricular', 'Sports', 'Music']
```

```
target_col = 'GPA'

# Define GPA categories

bins = [0, 2.5, 3.5, 4.0]

labels = ['Low', 'Medium', 'High']

data['GPA_Category'] = pd.cut(data[target_col], bins=bins, labels=labels, right=False)

data['GPA_Category'] = data['GPA_Category'].astype('category').cat.codes  # Convert to numerical labels


# Separate features and the new categorical target

X = data[feature_cols]

y_categorical = data['GPA_Category']


# Handle missing values

X = X.fillna(X.mean())


# Split data

X_train, X_test, y_train_cat, y_test_cat = train_test_split(X, y_categorical, test_size=0.3, random_state=42)


# Feature Scaling

scaler = StandardScaler()

X_train_scaled = scaler.fit_transform(X_train)

X_test_scaled = scaler.transform(X_test)


# Train a Logistic Regression model (for classification)

model_clf = LogisticRegression(random_state=42, multi_class='ovr')  # Use a classifier

model_clf.fit(X_train_scaled, y_train_cat)
```

```
# Make predictions on the test set
y_pred_cat = model_clf.predict(X_test_scaled)

# Calculate the confusion matrix
cm = confusion_matrix(y_test_cat, y_pred_cat)

# Generate a heatmap of the confusion matrix
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=labels, yticklabels=labels)
plt.xlabel('Predicted Category')
plt.ylabel('Actual Category')
plt.title('Confusion Matrix for GPA Category Prediction')
plt.show()

# Calculate evaluation metrics for classification
accuracy = accuracy_score(y_test_cat, y_pred_cat)
precision = precision_score(y_test_cat, y_pred_cat, average='weighted')
recall = recall_score(y_test_cat, y_pred_cat, average='weighted')

print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
```

# RESULT




```
Accuracy: 0.9150
Precision: 0.8861
Recall: 0.9150
```

# CREDIT

**Data set from:-**

Student Performance: Student Performance Dataset from UCI

**Platform used:-**

Google Colab

**Image from:-**

Google Colab