www.kiet.edu
Delhi-NCR, Ghaziabad

KIET
GROUP OF INSTITUTIONS
Connecting Life with Learning

A

Assessment Report

on

“Student Performance Prediction”

Submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

Name of discipline

By

Vidisha Goel (202401100300278)

Under the supervision of

Abhishek Shukla Sir

KIET Group of Institutions, Ghaziabad

Affiliated to

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

(Formerly UPTU)

May, 2025

INTRODUCTION

In the field of education, understanding student performance is crucial for timely intervention and support. Predicting whether a student is likely to pass or fail can help educators, parents, and institutions take appropriate measures to guide students in the right direction.

This project focuses on using machine learning techniques to classify students based on their academic performance and analyze patterns among them. We used the **Student Performance Dataset** from the UCI Machine Learning Repository, which includes features such as attendance, study habits, previous grades, and other relevant academic indicators.

The primary goal of this project is twofold:

1. **Classification** – Predict whether a student will pass or fail using historical academic and behavioral data.
2. **Clustering** – Segment students into groups based on similar characteristics, enabling institutions to design more personalized learning or support strategies.

This study not only demonstrates the predictive capabilities of machine learning in education but also emphasizes how data-driven insights can support better academic planning and decision-making.

METHODOLOGY

To build this solution, we followed a structured and practical approach to handle both classification and clustering problems effectively.

We began by importing the dataset and manually uploading it into Google Colab for preprocessing. The dataset initially contained various features related to student academic habits and background. We cleaned the data, encoded categorical variables into numerical format using **Label Encoding**, and normalized the features using **StandardScaler** for uniformity.

For the classification part, we defined a new target variable named `Pass`, derived from the `GradeClass` column. Students with GradeClass 1 or 2 were considered to have "passed," while those with GradeClass 3 or 4 were considered to have "failed." Using this binary target, we trained a **Random Forest Classifier**—a robust, ensemble-based algorithm known for its accuracy and resistance to overfitting.

The model was evaluated using metrics such as accuracy, precision, and recall. We also generated a **confusion matrix** and visualized it using a heatmap to clearly see the distribution of correct and incorrect predictions.

In the second part of the project, we performed **clustering** to segment the students into meaningful groups. We used the **KMeans clustering algorithm**, which grouped students based on their academic behavior and performance features. To visualize these clusters effectively, we applied **Principal Component Analysis (PCA)** to reduce the high-dimensional data into two dimensions. The clusters were then plotted with proper labels and colors to interpret them easily.

This two-step methodology allowed us to both **predict outcomes** and **understand the hidden patterns** in student data, giving us both actionable results and analytical insight.

CODE

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

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score,
precision_score, recall_score

# Allow manual upload
from google.colab import files
uploaded = files.upload()

# Load dataset
df = pd.read_csv(next(iter(uploaded)))

# Preview dataset
df.head()


# Let's see the structure of the data
print(df.info())

# Drop irrelevant columns if any (adjust as per your dataset)
# Example: df = df.drop(['StudentID'], axis=1)

# Encode categorical features if any
label_encoders = {}
for col in df.select_dtypes(include='object').columns:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    label_encoders[col] = le

# Define features and target
df['Pass'] = df['GradeClass'].apply(lambda x: 1 if x <= 2 else 0)
# Assuming the target column is 'Pass' or something similar; change as needed
```

```
X = df.drop(['StudentID', 'GradeClass', 'Pass'], axis=1)
y = df['Pass']

# Normalize features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Split into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# Train a classifier
clf = RandomForestClassifier(random_state=42)
clf.fit(X_train, y_train)

# Predict
y_pred = clf.predict(X_test)

# Confusion matrix and evaluation metrics
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", cm)

# Heatmap
# Define class labels
labels = ['Fail', 'Pass']

# Plot confusion matrix with labels
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=labels, yticklabels=labels)
plt.title('Confusion Matrix Heatmap')
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.show()


# Metrics
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Precision:", precision_score(y_test, y_pred, average='binary'))  # Change average if multiclass
print("Recall:", recall_score(y_test, y_pred, average='binary'))

# Full classification report
```

```
print("\nClassification Report:\n", classification_report(y_test, y_pred))

from sklearn.cluster import KMeans
from sklearn.decomposition import PCA

# Use relevant features for clustering
features = df.drop(['StudentID', 'GradeClass', 'Pass'], axis=1)

# Normalize features
scaler = StandardScaler()
features_scaled = scaler.fit_transform(features)

# Apply KMeans clustering
kmeans = KMeans(n_clusters=3, random_state=42)
cluster_labels = kmeans.fit_predict(features_scaled)

# Add cluster labels to original dataframe
df['Cluster'] = cluster_labels

# Reduce dimensions for visualization
pca = PCA(n_components=2)
reduced_features = pca.fit_transform(features_scaled)

# Plot clusters
plt.figure(figsize=(8,6))
sns.scatterplot(x=reduced_features[:,0], y=reduced_features[:,1], hue=cluster_labels,
palette='Set2')
plt.title("Student Clusters (via PCA)")
plt.xlabel("PCA Component 1")
plt.ylabel("PCA Component 2")
plt.legend(title='Cluster')
plt.show()
```

OUTPUT / RESULT

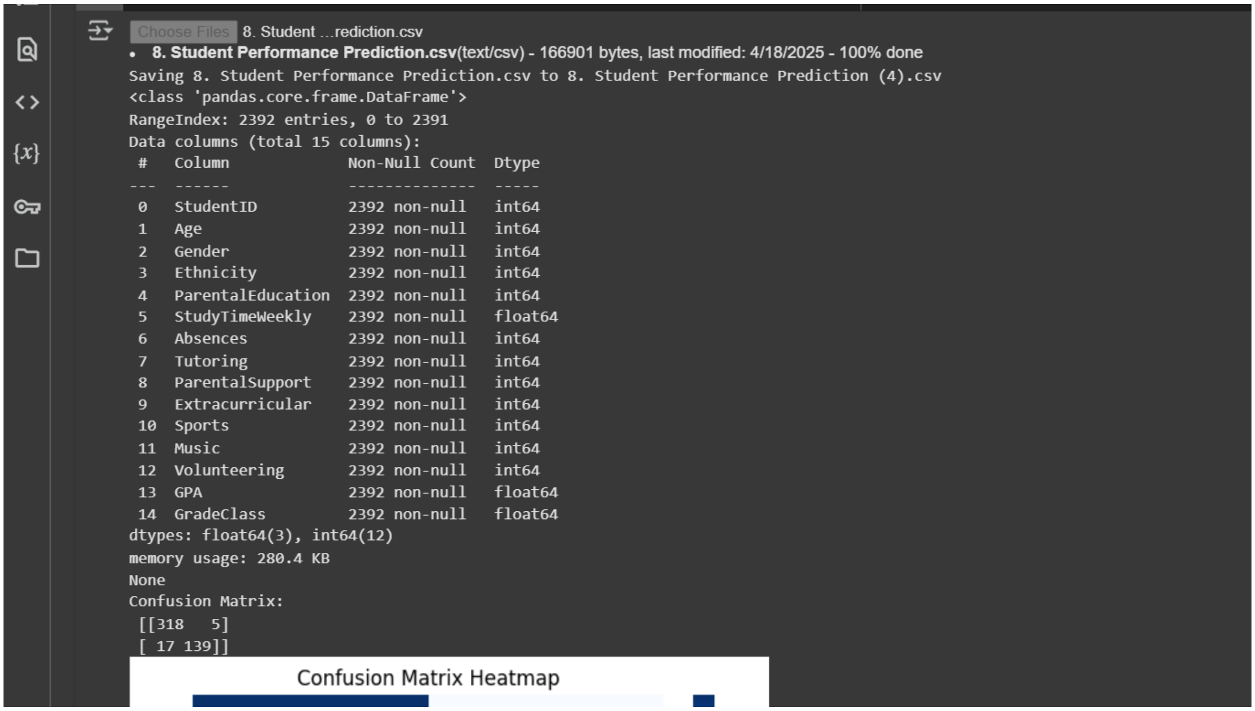
The final output consists of both classification and clustering results, presented through printed metrics and visual plots.

For classification, the model achieved measurable scores across various metrics:

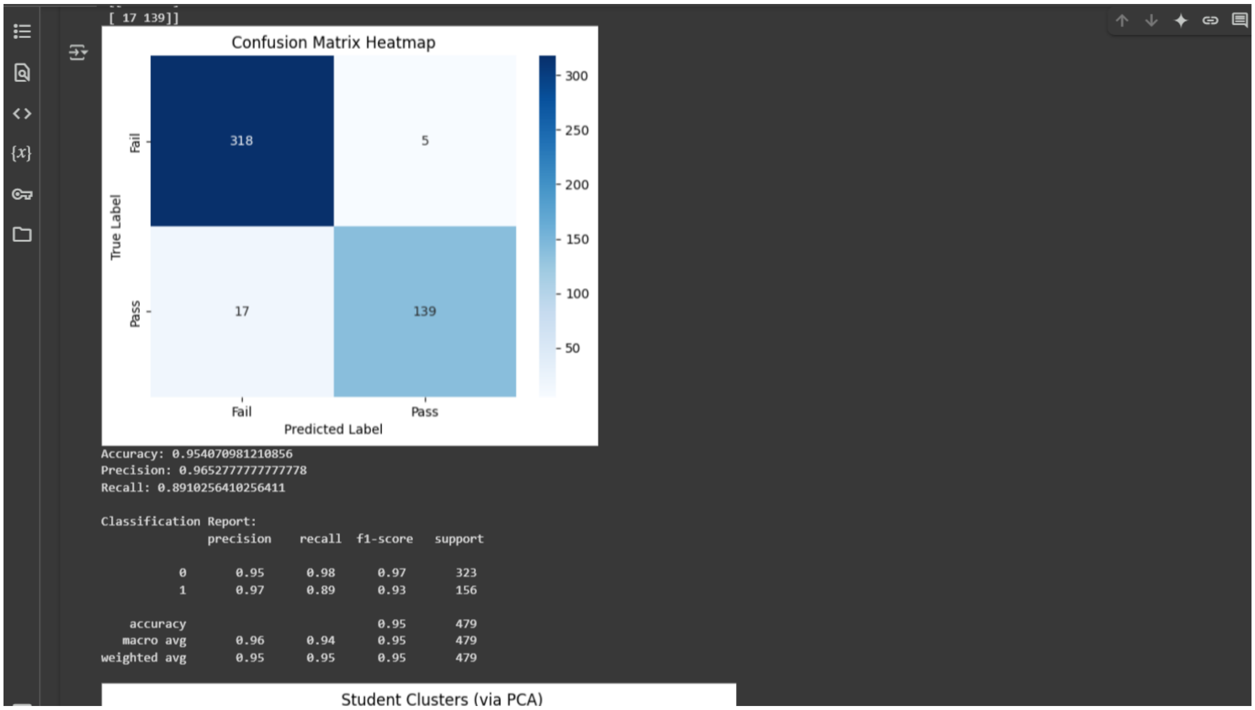
- **Accuracy** – indicating the overall correctness of the model.
- **Precision** – showing how many predicted "pass" cases were actually correct.
- **Recall** – reflecting the model’s ability to capture actual pass cases.

The **confusion matrix heatmap** clearly visualizes the performance of our classifier by comparing actual versus predicted labels. High diagonal values in the heatmap indicate strong model performance.

In the clustering section, students were divided into three distinct clusters. Each cluster groups students with similar traits, such as study habits, past grades, and attendance patterns. The **scatter plot**



```
8. Student ...rediction.csv
• 8. Student Performance Prediction.csv(text/csv) - 166901 bytes, last modified: 4/18/2025 - 100% done
Saving 8. Student Performance Prediction.csv to 8. Student Performance Prediction (4).csv
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2392 entries, 0 to 2391
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   StudentID          2392 non-null   int64  
 1   Age                2392 non-null   int64  
 2   Gender             2392 non-null   int64  
 3   Ethnicity          2392 non-null   int64  
 4   ParentalEducation  2392 non-null   int64  
 5   StudyTimeWeekly    2392 non-null   float64
 6   Absences           2392 non-null   int64  
 7   Tutoring           2392 non-null   int64  
 8   ParentalSupport    2392 non-null   int64  
 9   Extracurricular    2392 non-null   int64  
 10  Sports             2392 non-null   int64  
 11  Music              2392 non-null   int64  
 12  Volunteering       2392 non-null   int64  
 13  GPA                2392 non-null   float64
 14  GradeClass         2392 non-null   float64
dtypes: float64(3), int64(12)
memory usage: 280.4 KB
None
Confusion Matrix:
 [[318   5]
 [ 17 139]]
```



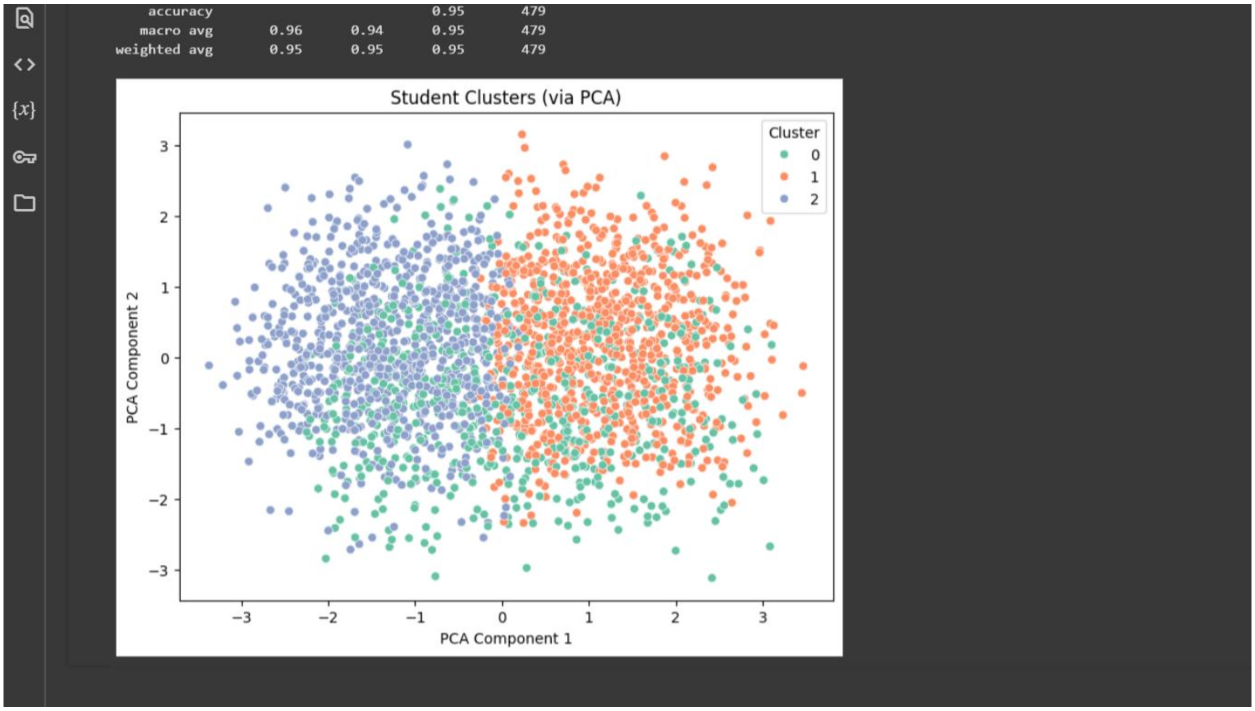
```
 [ 17 139]]
```

```
Accuracy: 0.954070981210856
Precision: 0.9652777777777778
Recall: 0.8910256410256411

Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.98      0.97       323
           1       0.97      0.89      0.93       156

    accuracy                           0.95       479
   macro avg       0.96      0.94      0.95       479
weighted avg       0.95      0.95      0.95       479
```



```
    accuracy                           0.95       479
   macro avg       0.96      0.94      0.95       479
weighted avg       0.95      0.95      0.95       479
```