An Assessment Report

on

“Student Performance Prediction”

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY

DEGREE

SESSION 2024-25

By

Shivanshi Dhyani(202401100400178)

Under the supervision of

“Mr. Abhishek Shukla”

KIET Group of Institutions, Ghaziabad

Affiliated to

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

(Formerly UPTU)

May, 2025

Introduction

In this project, the goal is to predict the performance of students based on various features such as demographic data, study habits, and other academic information. By predicting the grade class of students, educators can gain insights into which students may need additional support and intervention. The model used for this prediction is a **Random Forest classifier**, a popular machine learning algorithm that works well for classification tasks.

To evaluate the model, a **confusion matrix heatmap** is generated, which allows us to visualize how well the model performs in predicting students' grade classes. Along with this, performance metrics like **accuracy**, **precision**, and **recall** are calculated to give us a deeper understanding of how effective the model is in predicting the correct grade class.

Methodology

The methodology for this project involves several key steps:

1. **Data Preprocessing:**
    a. The dataset is loaded into the system, and columns that don't provide meaningful information are removed.
    b. Features (input variables) and the target (the variable we want to predict) are separated.
2. **Train-Test Split:**
    a. The data is split into two parts: one part is used to train the model (80%), and the other part is reserved for testing the model (20%).
3. **Model Training:**
    a. A **Random Forest classifier** is trained using the training data. Random Forest is an ensemble learning method that builds multiple decision trees and combines them to improve classification accuracy.
4. **Prediction & Evaluation:**
    a. Once the model is trained, it is used to predict the grade class of students in the test data.

b. The model's performance is evaluated using **accuracy**, **precision**, and **recall**, which help us understand how often the model is correct and how well it handles different types of errors.

5. **Confusion Matrix Heatmap:**
   a. A **confusion matrix heatmap** is created to visually represent the number of correct and incorrect predictions, categorized by the actual and predicted grade classes.

Code

```
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.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score

# Load the dataset
df = pd.read_csv("8. Student Performance Prediction.csv")  # Ensure this file is in the same folder

# Create binary 'PassFail' target: 1 = Pass (GradeClass <= 2), 0 = Fail
df["PassFail"] = df["GradeClass"].apply(lambda x: 1 if x <= 2 else 0)

# Drop unnecessary columns
df_model = df.drop(columns=["StudentID", "GPA", "GradeClass"])

# Features and target
X = df_model.drop("PassFail", axis=1)
y = df_model["PassFail"]

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

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

# Predict
y_pred = clf.predict(X_test)
```

```
# Evaluate
conf_matrix = confusion_matrix(y_test, y_pred)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, zero_division=0)
recall = recall_score(y_test, y_pred, zero_division=0)

# Show heatmap
plt.figure(figsize=(6, 5))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap="YlGnBu",
            xticklabels=["Fail", "Pass"], yticklabels=["Fail", "Pass"])
plt.title("Confusion Matrix Heatmap (Pass/Fail Prediction)")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.tight_layout()
plt.show()

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

Result/Output