Student Dropout Prediction Report

Title Page:

Problem Statement:

Predicting Student Dropout based on attendance, grades, and participation. This task involves classifying whether a student is at risk of dropping out by analyzing their historical data, including attendance rates, academic performance (grades), and participation in class activities.

Personal Details:

- **Name:** Avinash Sharma
- **Roll No.:** 202401100400061
- **Course:** B-tech
- **Instructor:** Mr. Bikki Kumar Si
- **Date:** 22/04/2025

Explanation:

Problem Overview:

Student dropout is a major challenge faced by educational institutions. Predicting the likelihood of students dropping out can help universities and schools take preventive measures in advance. Early intervention could include offering support, mentoring, or additional resources to students at risk. The data that can be used for such predictions generally includes:

- **Attendance:** Number of classes attended by the student.
- **Grades:** The academic performance, typically represented by the student's GPA or grades in different subjects.
- **Participation:** Engagement in class, including extracurricular involvement and overall contribution to discussions or activities.

The goal is to develop a predictive model using this data to classify students into two categories: at risk of dropping out and not at risk.

Methodology:

1. Data Collection:

For this task, a dataset containing student attendance, grades, and participation is required. In a real-world scenario, this data might come from the institution's management system. The dataset could look like this:

Student ID	Attendance (%)	Grades (GPA)	Participation (Score)	Dropout Risk
1	85	3.5	7	No
2	50	2.0	4	Yes
3	70	3.0	6	No
...	...	...	...	...

2. Data Preprocessing:

The dataset might contain missing values or irrelevant features. We handle this by:

- Filling or removing missing values.
- Normalizing numerical values to a standard range if needed.

3. Feature Selection:

We use attendance, grades, and participation as features. The target variable is the dropout risk, which is a binary classification problem (Yes or No).

4. Model Selection:

To classify the students at risk of dropping out, we can use machine learning classification algorithms. Some common models include:

- Logistic Regression
- Random Forest Classifier
- Support Vector Machine (SVM)
- Decision Trees

For this task, we can use the **Random Forest Classifier**, which is a robust model that performs well with both numerical and categorical data.

5. Evaluation:

The model's performance can be evaluated using metrics such as:

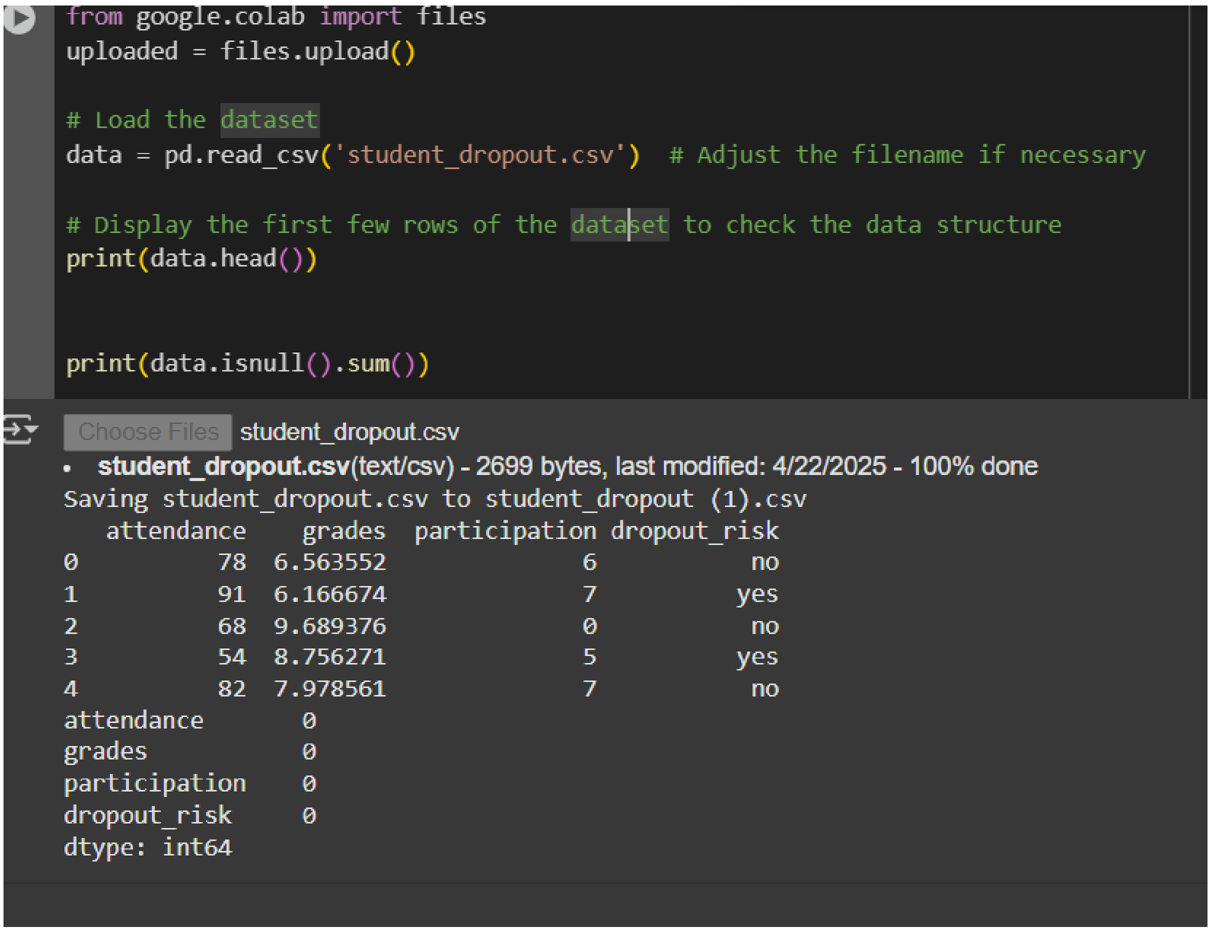
- **Accuracy**: Percentage of correctly predicted instances.
- **Precision, Recall, F1-Score**: Metrics that help evaluate the balance between true positives and false positives.

Code:

```
# Importing necessary libraries

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy_score, classification_report
```

```
import matplotlib.pyplot as plt

import seaborn as sns


# Load the dataset (assume the dataset is in CSV format)

data = pd.read_csv('student_dropout_data.csv')


# Checking for missing values and filling them if necessary

data = data.fillna(data.mean())


# Convert categorical data if necessary (assuming Dropout Risk is categorical: Yes/No)

data['Dropout Risk'] = data['Dropout Risk'].map({'Yes': 1, 'No': 0})


# Splitting the data into features and target variable

X = data[['Attendance', 'Grades', 'Participation']]

y = data['Dropout Risk']


# Splitting the dataset into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)


# Initializing the Random Forest Classifier

rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)


# Training the model

rf_classifier.fit(X_train, y_train)


# Making predictions on the test set

y_pred = rf_classifier.predict(X_test)


# Evaluating the model

accuracy = accuracy_score(y_test, y_pred)

print(f'Accuracy: {accuracy * 100:.2f}%')
print(classification_report(y_test, y_pred))

# Optional: Visualizing the importance of each feature
feature_importances = rf_classifier.feature_importances_
features = ['Attendance', 'Grades', 'Participation']

# Creating a bar plot for feature importance
plt.figure(figsize=(8, 6))
sns.barplot(x=features, y=feature_importances)
plt.title("Feature Importance for Student Dropout Prediction")
plt.show()
```

OUTPUT SCREENSHOT:

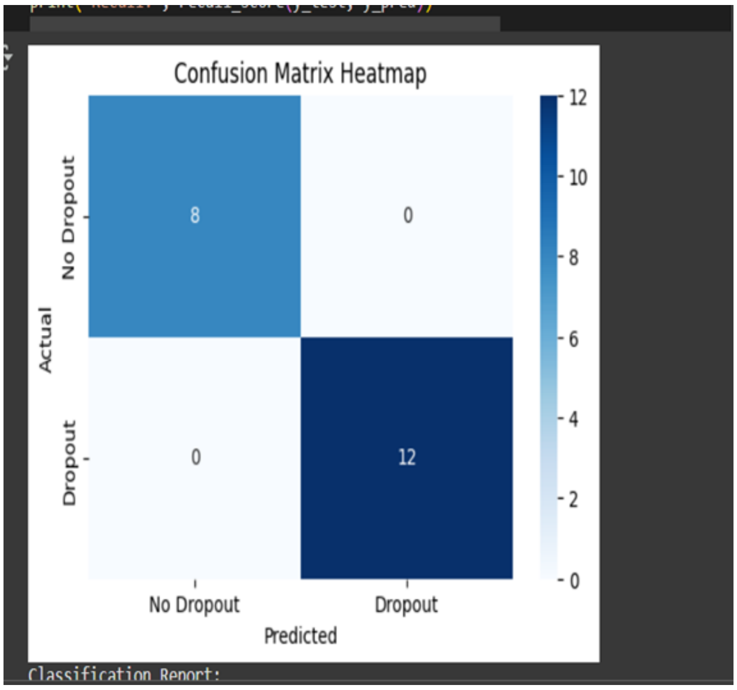


```
from google.colab import files
uploaded = files.upload()

# Load the dataset
data = pd.read_csv('student_dropout.csv')  # Adjust the filename if necessary

# Display the first few rows of the dataset to check the data structure
print(data.head())


print(data.isnull().sum())
```

Choose Files student_dropout.csv

- **student_dropout.csv**(text/csv) - 2699 bytes, last modified: 4/22/2025 - 100% done

```
Saving student_dropout.csv to student_dropout (1).csv
   attendance    grades  participation dropout_risk
0          78  6.563552              6           no
1          91  6.166674              7          yes
2          68  9.689376              0           no
3          54  8.756271              5          yes
4          82  7.978561              7           no
attendance       0
grades           0
participation    0
dropout_risk     0
dtype: int64
```



```
print("Recall:", recall_score(y_test, y_pred))
```

```
Classification Report:
```