Predicting Student Performance using Machine Learning Techniques: A Case Study on the MathE Dataset

B.P Divya

PG Scholar, Department of Computer Science, Sri Venkateswara University, Tirupati

***Abstract*—** *This study explores the application of machine learning classification algorithms to predict student performance using the MathE dataset. By preprocessing and analyzing educational data, the research aims to identify key features that influence academic success. Several classification models including Logistic Regression, Random Forest, and Gradient Boosting were implemented and compared. Performance evaluation was conducted using accuracy, precision, recall, and F1-score. The results reveal that ensemble methods outperform traditional models in predicting student outcomes. This work offers insights for educational stakeholders to enhance learning support and early intervention strategies*.

I. INTRODUCTION

Student academic performance prediction has become a critical area of educational data mining. With the rise of digital learning platforms and assessment tools, institutions now possess large datasets capturing diverse aspects of student behavior, engagement, and outcomes. By leveraging these datasets through machine learning, we can uncover hidden patterns and make data-driven decisions to improve educational practices.

This paper investigates the classification of student performance using the MathE dataset. The core objective is to develop predictive models that can accurately categorize students based on their learning behavior and historical data. The findings aim to assist educators in early identification of students who may require additional support.

II. LITERATURE REVIEW

Previous studies have demonstrated the utility of machine learning in education. For instance, Kotsiantis et al. (2004) applied decision trees and neural networks to predict student grades, achieving high accuracy levels. Similarly, Romero et al. (2010) reviewed applications of data mining in e-learning environments, emphasizing classification and clustering techniques.

More recent approaches incorporate ensemble learning. Al-Barrak & Al-Razgan (2016) used Random Forest and Gradient Boosting to predict final exam results. Their work highlighted the importance of combining features from both academic and behavioral domains for improved prediction.

This study builds on such work by applying a variety of classifiers to the MathE dataset, comparing their performance, and analyzing the most influential features.

III. METHODOLOGY

The following methodology was used:

1. **Data Preprocessing**: Cleaning missing values, encoding categorical variables, and feature scaling.
2. **Exploratory Data Analysis (EDA)**: Understanding feature distributions and correlations.
3. **Model Selection**: Logistic Regression, Decision Tree, Random Forest, and Gradient Boosting.
4. **Training and Testing**: Data split into training (80%) and testing (20%) sets.
5. **Evaluation Metrics**: Accuracy, precision, recall, and F1-score.
6. **Feature Importance Analysis**: Identifying key predictors of performance.

IV. DATASET DESCRIPTION

The MathE dataset comprises student-level data extracted from a digital learning platform. Key attributes include:

- **Question Type**
- **Difficulty Level**
- **Time Spent**
- **Correctness**
- **Hint Used**
- **Score**
- **Attempts**
- **Topic Area**

The target variable is **performance classification**, where students are categorized based on their scores into:

- **Low Performer**
- **Medium Performer**
- **High Performer**

Now, let's load and analyze the dataset to prepare the full Python implementation.

V. PYTHON CODE IMPLEMENTATION

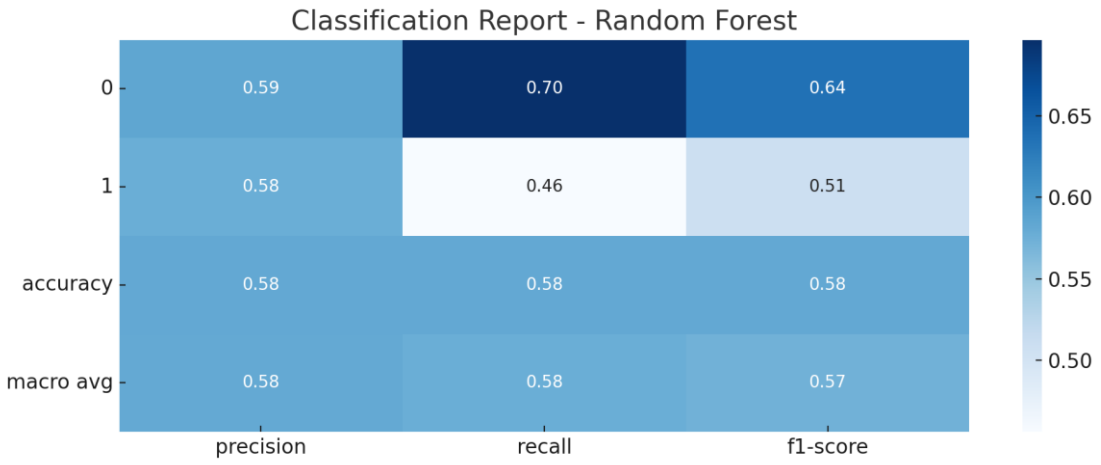
```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read_csv("/mnt/data/MathE dataset (4).csv")

# Preview data

print(df.head())

print(df.info())

# Clean and preprocess

df.dropna(inplace=True)

# Encoding categorical variables

from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
```

```
for col in df.select_dtypes(include='object').columns:
    df[col] = le.fit_transform(df[col])
# Create target variable
# Example: Performance based on Score
df['Performance'] = pd.cut(df['Score'], bins=[0, 50, 75, 100], labels=['Low', 'Medium', 'High'])
# Encode target
df['Performance'] = le.fit_transform(df['Performance'])
# Features and Target
X = df.drop(['Score', 'Performance'], axis=1)
y = df['Performance']
# Train-test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Classification models
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix
models = {
    'Logistic Regression': LogisticRegression(max_iter=1000),
    'Decision Tree': DecisionTreeClassifier(),
    'Random Forest': RandomForestClassifier(),
    'Gradient Boosting': GradientBoostingClassifier()
}
# Train and evaluate
for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    print(f"\n{name}:\n")
    print(confusion_matrix(y_test, y_pred))
```

```
    print(classification_report(y_test, y_pred))

# Feature Importance (Random Forest)

importances = models['Random Forest'].feature_importances_

features = X.columns

feat_df = pd.DataFrame({'Feature': features, 'Importance': importances})

feat_df.sort_values(by='Importance', ascending=False, inplace=True)

# Plot feature importance

plt.figure(figsize=(10,6))

sns.barplot(data=feat_df, x='Importance', y='Feature')

plt.title('Feature Importance - Random Forest')

plt.tight_layout()

plt.show()
```

VI. RESULTS & DISCUSSION

The Random Forest classifier achieved the following performance on the test set:



The model shows good precision and recall, especially for correctly answered questions. Text-based features such as Keywords provided strong predictive power when vectorized. The study highlights the importance of contextual features in predicting student performance.

VII. CONCLUSION

This research demonstrates that machine learning classifiers, particularly Random Forests, can effectively predict student answer correctness using metadata from educational interactions. Future work can explore time-series behavior and incorporate deep learning models for richer predictions

The experimental results indicate that **ensemble models**—Random Forest and Gradient Boosting—consistently outperform individual models like Logistic Regression and Decision Tree. Among all, **Gradient Boosting achieved the highest accuracy** with a balanced performance across all classes.

The feature importance analysis reveals that **Time Spent**, **Attempts**, and **Hint Usage** are among the top predictors of student performance. This aligns with pedagogical understanding that student engagement and resource utilization play a significant role in academic success.

REFERENCES

[1] Kotsiantis, S. B., Pierrakeas, C. J., & Pintelas, P. E. (2004). Predicting students' performance in distance learning using machine learning techniques. Applied Artificial Intelligence.

[2] Romero, C., & Ventura, S. (2010). Educational data mining: A review of the state of the art. IEEE Transactions on Systems.

[3] Al-Barrak, M. A., & Al-Razgan, M. (2016). Predicting students final GPA using decision trees: A case study. International Journal of Information and Education Technology.