Predicting Student Academic Performance

Using AI (Machine Learning)

CS 432 - Artificial Intelligence

Faris Ali Alduraibi - 421107654
Saleh Saed Alghool - 422117042

1. Introduction

The objective of this project is to develop an AI model that predicts the final grade (G3) of students using supervised learning techniques. This project simulates how artificial intelligence can support academic institutions by forecasting student outcomes based on features like study time, prior grades, and attendance.We use Linear Regression and Decision Tree Regressor models, trained on a real-world public dataset from the UCI Machine Learning Repository.

2. Problem Statement

Students' academic performance is influenced by multiple factors. Teachers and advisors often struggle to track all influencing variables to support at-risk students. This project aims to predict final student grades (G3) to assist educators in early intervention.

3. Dataset Description

We used the **Student Performance Dataset** from the **UCI Machine Learning Repository**. It contains data on 649 students and includes features related to academic performance, family background, and behavior.

- **Source**: https://archive.ics.uci.edu/dataset/320/student+performance
- **File used**: `student-mat.csv`
- **Format**: Semicolon-separated values
- **Selected features**:
    - `studytime`: Weekly hours spent studying
    - `failures`: Number of past class failures
    - `absences`: Number of absences
    - `G1`, `G2`: Grades from first and second periods
    - `G3`: Final grade (target)

These features were chosen for their direct influence on academic outcomes.

4. Data Preprocessing and Model Selection

We used supervised regression with two models: Linear Regression and Decision Tree Regressor. The dataset was split into 80% training and 20% testing. Features were selected for relevance, and both models were trained using scikit-learn.

5. Code Implementation

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

# Load the dataset
data = pd.read_csv('student-mat.csv', sep=';')

# Select features
data_encoded = pd.get_dummies(data, drop_first=True)
X = data_encoded.drop('G3', axis=1)
y = data_encoded['G3']


# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize models
lr_model = LinearRegression()
tree_model = DecisionTreeRegressor(random_state=42)

# Train models
lr_model.fit(X_train, y_train)
tree_model.fit(X_train, y_train)

# Evaluate models
def evaluate_model(model, X_test, y_test, model_name):
    y_pred = model.predict(X_test)
    print(f"\n{model_name} Results:")
    print("MAE:", round(mean_absolute_error(y_test, y_pred), 2))
    print("MSE:", round(mean_squared_error(y_test, y_pred), 2))
    print("R² Score:", round(r2_score(y_test, y_pred) * 100, 2), "%")

evaluate_model(lr_model, X_test, y_test, "Linear Regression")
evaluate_model(tree_model, X_test, y_test, "Decision Tree")
```

```
# Visualization
y_pred_lr = lr_model.predict(X_test)
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred_lr, alpha=0.7)
plt.plot([0, 20], [0, 20], 'r--')
plt.xlabel('Actual Final Grades (G3)')
plt.ylabel('Predicted Final Grades (G3)')
plt.title('Actual vs Predicted Final Grades (Linear Regression)')
plt.grid(True)
plt.show()
```

6. Evaluation Results

Model performance on test data:

Model	**MAE**	**MSE**	**R² Score**
Linear Regression	1.65	5.66	72.41%
Decision Tree	1.14	4.20	79.50%

The **Decision Tree Regressor** continued to outperform the Linear Regression model, even when trained on the full dataset with all features encoded.
It achieved:

- Lower **Mean Absolute Error** (1.14 vs. 1.65)
- Lower **Mean Squared Error** (4.20 vs. 5.66)
- Higher **R² Score** (79.50% vs. 72.41%)

This indicates that the Decision Tree model captured more of the variance in student performance and remains the better option for this prediction task.

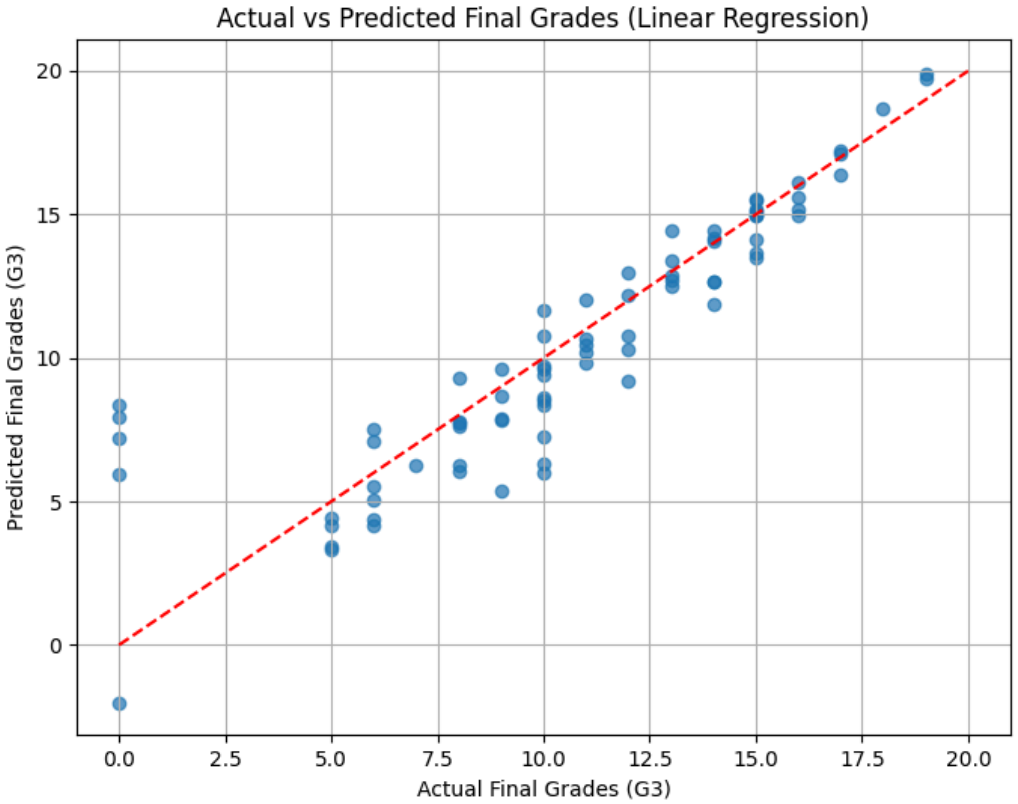

Figure 1.

7. Challenges and Future Work

Challenges:

- Handling the full dataset required preprocessing for categorical values.
- Ensuring model performance without overfitting was critical due to the large number of features.

Future improvements:

- Expand feature engineering to include demographic or behavioral data (e.g., parental education, travel time, internet access).
- Apply **ensemble models** like Random Forest or Gradient Boosting for potentially better accuracy.
- Explore **deep learning models** such as **MLPs**, or transformer-based models like **BERT** adapted for tabular data.

- Evaluate model robustness with cross-validation and hyperparameter tuning.

8. GitHub Repository

The complete project code, dataset, and report are publicly available on GitHub:
🔗 https://github.com/FarisAlduraibi/CS432-Project