Student Performance Prediction using Machine Learning

Final Year Project Report

By: Kunapalli Sai Bharath

Introduction & Motivation

Education systems generate large amounts of data. Predicting student performance helps educators provide timely interventions. This project uses Python and Machine Learning to predict student final exam scores.

Problem Statement

Students face challenges due to irregular study habits and attendance. Educators need tools to predict performance early. The problem: Can we predict final exam performance using past data?

Objectives

1. Analyze student study hours, attendance, and past scores.
2. Train a machine learning model to predict final exam performance.
3. Evaluate accuracy of predictions.
4. Provide insights with visualization.

Dataset (Sample Inputs)

Study_Hours	Attendance	Past_Scores	Final_Exam
2	60	50	52
3	65	55	57
4	70	60	61
5	72	62	65
6	75	65	68
7	80	70	73
8	85	72	76
9	88	75	79
10	90	80	83
11	95	85	90

Input

The input is a tabular dataset with the following columns (one row per student):
- Study_Hours: Number of hours the student studied per day (numeric).
- Attendance: Attendance percentage (numeric).
- Past_Scores: Average of past scores (numeric).

Step-by-step Instructions (How to run)

1. Install Python (3.7+) and pip if not already installed.
2. Create a project folder and save the Python script (e.g., student_performance.py) with the provided code.
3. Install required packages (if you will use pandas/matplotlib):
pip install numpy pandas matplotlib reportlab

4. Run the script from the terminal:
python student_performance.py
5. The script will print evaluation metrics (MSE, R2) and save a plot (actual_vs_predicted_plot.png).
6. Use the generated PDF and PPT for submission.

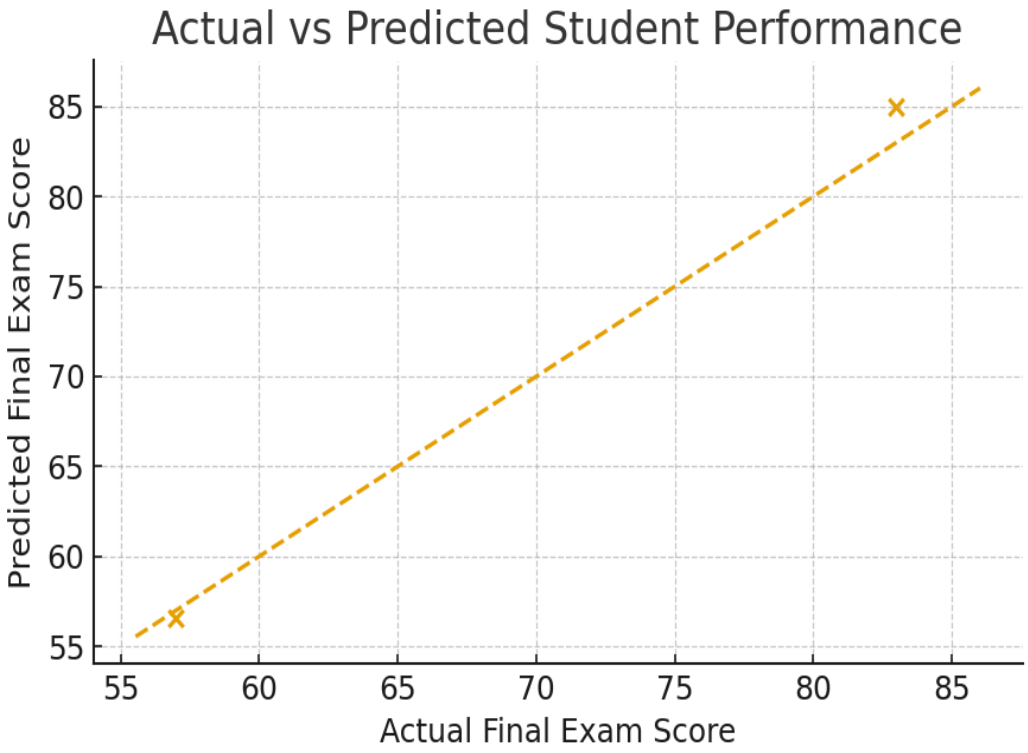
Python Code (full)

```
# Student Performance Prediction Project (using numpy least squares)
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Sample dataset
data = {
    'Study_Hours': [2,3,4,5,6,7,8,9,10,11],
    'Attendance': [60,65,70,72,75,80,85,88,90,95],
    'Past_Scores': [50,55,60,62,65,70,72,75,80,85],
    'Final_Exam': [52,57,61,65,68,73,76,79,83,90]
}
df = pd.DataFrame(data)

# Prepare features and target
X = df[['Study_Hours','Attendance','Past_Scores']].values
y = df['Final_Exam'].values
X = np.hstack([np.ones((X.shape[0],1)), X])  # intercept

# Train-test split (20% test)
rng = np.random.RandomState(42)
idx = np.arange(X.shape[0])
rng.shuffle(idx)
test_size = int(0.2 * len(idx))
test_idx = idx[:test_size]
train_idx = idx[test_size:]
X_train = X[train_idx]; X_test = X[test_idx]
y_train = y[train_idx]; y_test = y[test_idx]

# Train using normal equation
beta = np.linalg.pinv(X_train.T.dot(X_train)).dot(X_train.T).dot(y_train)

# Predict and evaluate
y_pred = X_test.dot(beta)
mse = np.mean((y_test - y_pred)**2)
r2 = 1 - np.sum((y_test - y_pred)**2) / np.sum((y_test - np.mean(y_test))**2)

print('Mean Squared Error:', mse)
print('R2 Score:', r2)
print('Test (Actual -> Predicted):')
for a,p in zip(y_test, y_pred): print(a, '->', round(p,2))

# Plot
import matplotlib.pyplot as plt
plt.scatter(y_test, y_pred)
plt.plot([min(y_test.min(), y_pred.min())-1, max(y_test.max(), y_pred.max())+1],
         [min(y_test.min(), y_pred.min())-1, max(y_test.max(), y_pred.max())+1], linestyle='--')
plt.xlabel('Actual Final Exam Score'); plt.ylabel('Predicted Final Exam Score')
plt.title('Actual vs Predicted Student Performance'); plt.show()
```

Output (Sample)

```
Console Output (sample run):
```

```
Mean Squared Error: 2.1396
R2 Score: 0.9873

Test set (Actual -> Predicted):
83 -> 85.02
57 -> 56.54
```



Results Interpretation

The Mean Squared Error (MSE) = 2.1396 (lower is better). The R^2 score = 0.9873 (closer to 1 is better). The Test set actual vs predicted values are shown above.

Conclusion & Future Work

Conclusion: Machine Learning can successfully predict student performance using historical features. Future Work: Use larger datasets, try advanced models (e.g., Random Forests, Neural Networks), and build a web dashboard.