Students Performance in Exams Dataset

Assignment: Predicting Students' Scores Based on Performance Factors

Objective:
Predict a student's **math score** based on other factors like gender, test preparation, parental education, and lunch type using regression techniques. Alternatively, students can also predict if the student will pass or fail a threshold (classification task).

Assignment Breakdown:

Step 1: Load the Data

1. **Task**: Download the dataset from Kaggle and load it using Pandas.
2. **Expected Output**:
    - A loaded dataset (`students_data`) overview using `.info()`, `.head()`, and `.describe()`.
    - Inspect columns such as `gender`, `race/ethnicity`, `parental level of education`, `lunch`, and the three score columns: `math score`, `reading score`, and `writing score`.

Step 2: Data Cleaning

1. **Task**:
    - Handle missing values (if any).
    - Drop irrelevant columns that may not contribute to predicting the target (`math score`).
    - Encode categorical features (like `gender`, `parental level of education`) using one-hot encoding or label encoding.
2. **Expected Output**:
    - A clean dataset ready for analysis with all categorical features converted to numerical form.

Step 3: Exploratory Data Analysis (EDA)

1. **Task**:
    - Visualize the distribution of `math score` and other performance scores.
    - Check correlations between the target variable (`math score`) and other features using scatter plots or correlation matrices.
    - Visualize relationships between categorical features (like `gender`, `test preparation`) and the scores.
2. **Expected Output**:
    - At least 4 visualizations:

- Histogram of `math score`.
- Bar plots for the relationship between `parental level of education` and `math score`.
- Correlation heatmap for scores and other features.

Step 4: Feature Engineering

1. **Task**:
   - Create any necessary new features. For example, you could combine `reading score` and `writing score` into a `total score` column.
   - Decide if you want to predict the exact score (regression) or a binary outcome (pass/fail).
2. **Expected Output**:
   - A new feature like `total score`.
   - Binary column for classification (pass/fail) if chosen.

Step 5: Splitting the Data

1. **Task**:
   - Split the dataset into training and testing sets (80% training, 20% testing) using `train_test_split`.
2. **Expected Output**:
   - Split data ready for modeling.

Step 6: Applying a Machine Learning Model

1. **Task**:
   - For **regression**, apply **Linear Regression** or **Decision Tree Regressor** to predict `math score`.
   - For **classification** (pass/fail), apply **Logistic Regression** or **Random Forest**.
2. **Expected Output**:
   - Model predictions on the test set.
   - Summary of the model training process.

Step 7: Model Evaluation

1. **Task**:
   - For **regression**, evaluate using metrics like **R-squared**, **Mean Absolute Error (MAE)**, and **Root Mean Squared Error (RMSE)**.
   - For **classification**, evaluate with accuracy, precision, recall, and F1-score. Use confusion matrix and ROC curve.
2. **Expected Output**:
   - For regression: `R-squared`, MAE, RMSE.
   - For classification: accuracy, confusion matrix, ROC curve.