Project Documentation

📊 Student Performance Prediction Using Machine Learning

📂 Project Overview

This project aims to predict students' math scores based on various demographic and academic factors using Machine Learning models. It demonstrates the complete ML pipeline from data preprocessing to model evaluation and visualization.

🛠️ Tools & Technologies

- **Python** 🐍 – Programming language
- **Pandas** – Data manipulation and analysis
- **NumPy** – Numerical operations
- **Matplotlib & Seaborn** – Data visualization
- **Scikit-learn** – Machine learning models and evaluation
- **Flask** *(optional)* – For basic deployment

📊 Dataset

- **Source:** Kaggle – Students Performance Dataset
- **Features:**
  - Gender
  - Race/Ethnicity
  - Parental level of education
  - Lunch type
  - Test preparation course
  - Scores: Math, Reading, Writing

📦 Setup & Installation

1. **Install Python (if not installed):**

- Go to python.org.
- Download and install Python (ensure you check the box to **add Python to PATH** during installation).

2. **Set Up the Project Folder:**

- Create a new folder on your Desktop named **student-performance-ml**.

3. **Install Required Libraries:**

1. Open **Terminal** (Command + Space → type **Terminal** → **Enter**).
2. Navigate to the project folder:
3. cd ~/Desktop/student-performance-ml
4. Install the required libraries:
5. pip install pandas numpy matplotlib seaborn scikit-learn

4. **Download the Dataset:**

- Visit Kaggle Dataset Link.
- Download the CSV file and place it inside the **student-performance-ml** folder.

5. **Create the Python File:**

1. Open **TextEdit** → **Format** → **Make Plain Text**.
2. Paste the Python code into the file.
3. Save the file as **student_performance.py** inside the project folder.

6. **Run the Code:**

1. Open **Terminal** and navigate to the project folder:

   cd ~/Desktop/student-performance-ml

1. Run the Python file:

   python student_performance.py

🧮 Project Workflow

1. **Data Preprocessing**

- Load the dataset using Pandas.
- Clean and format data for analysis.
- Rename columns for easy access.
- Encode categorical variables using one-hot encoding.

2. **Model Building**

- Apply **Linear Regression** for basic predictive modeling.
- Apply **Decision Tree Regressor** for more complex relationships.

3. **Model Evaluation**

- Use the following metrics to assess performance:
  - **Mean Squared Error (MSE):** Measures the average squared difference between actual and predicted values.
  - **R^2 Score:** Indicates how well data fits the model.

4. **Hyperparameter Tuning**

- Use **GridSearchCV** to optimize Decision Tree parameters for better performance.

5. **Visualization**

- Use **Matplotlib** and **Seaborn** to create scatter plots and visualize actual vs. predicted scores.

📈 Results & Analysis

- Compare the performance of Linear Regression and Decision Tree.
- Decision Trees often perform better on complex data patterns.

🌐 Optional: Basic Deployment (Flask)

1. **Install Flask:**
2. pip install flask
3. **Create a simple Flask app** to input data and view predictions in your browser.

📚 Learning Outcomes

- Understand the end-to-end Machine Learning pipeline.
- Gain experience in data preprocessing, modeling, evaluation, and visualization.
- Learn basic hyperparameter tuning and model optimization.

💡 Tips for Beginners

- Start with **Linear Regression** to understand basic concepts.
- Gradually explore complex models like **Decision Trees**.
- Use visualization to interpret model results better.

🔗 Useful Links

- Scikit-learn Documentation
- Matplotlib Documentation
- Flask Documentation

✅ **Project Completed!** 🚀

💡 *Explore other algorithms or deploy the project using Flask for a simple web interface.*