Building a SimpleRNN Model for Weather Temperature Prediction

Objective

The goal of this assignment is to design, implement, and evaluate a **Recurrent Neural Network (RNN)** model (using **SimpleRNN**) to forecast the next day's temperature based on past weather data.

Dataset

Use the **Daily Weather Dataset**

- Example: Daily Weather Dataset – Kaggle
- Key Features:
    - `Date`
    - `Temperature` (target variable)
    - `Humidity`
    - `Wind Speed`
    - `Pressure` (optional)

Tasks

Part A: Data Understanding and Preprocessing

1. Load the dataset and explore:
    - Display first 10 rows.
    - Plot temperature trends over time.
    - Check for missing values.

2. Preprocess the data:
    - Handle missing values (impute/remove).
    - Normalize values using MinMaxScaler (so RNN converges faster).
    - Create **input sequences**:
        - Use **past 7–14 days' weather data** (temperature, humidity, wind speed) as input.
        - Target: **next day's temperature**.
    - Split dataset into **train, validation, and test sets**.

Part B: RNN Model Development

3. Build a **SimpleRNN model** using TensorFlow/Keras:
    - Input layer (with shape = sequence length × number of features).
    - SimpleRNN layer (e.g., 32–64 units).
    - Dropout layer (optional, to avoid overfitting).
    - Dense layer with 1 unit (linear activation for regression).
4. Compile the model:
    - Loss: **Mean Squared Error (MSE)**.
    - Optimizer: Adam.
    - Metrics: Mean Absolute Error (MAE).
5. Train the model on training data:
    - Batch size: 32
    - Epochs: 50–100

- Use validation data to monitor performance.
- Plot training vs validation **loss curves**.

Part C: Model Evaluation & Forecasting

6. Evaluate the model on the **test set**:
    - Calculate **RMSE, MAE, and R^2 score**.
    - Plot **predicted vs actual temperatures** for the test period.
7. Forecast the **next 7 days of temperature** using the trained model and visualize predictions vs recent historical data.