README: Electricity Price Prediction and Smoothening for AEMO

Project Title: Electricity Price Prediction and Smoothening for AEMO

Author: Tanmay Somani (n11485094)

Description

This project focuses on forecasting electricity prices in the National Electricity Market (NEM) with an emphasis on mitigating price volatility. It combines machine learning techniques, such as Random Forests and RNN-LSTM models, with data smoothing techniques to improve the accuracy and stability of price predictions.

Key Objectives

1. Analyzing which region exhibits both high price and demand spikes.
2. Predict electricity prices for the NSW region.
3. Detect and classify price spikes.
4. Apply Gaussian smoothing to minimize volatility and enhance decision-making insights.
5. Evaluate the effectiveness of models for both classification and regression tasks.

Project Structure

1. **Executive Summary**:
   - Discusses the goals and high-level findings of the project.
   - Machine learning models utilized: Random Forest Classifier and RNN-LSTM hybrid.
   - Role of Gaussian smoothing for reducing volatility.

2. **Methodology**:
    - Data collection from AEMO, focusing on NSW for the year 2022.
    - Feature engineering and importance ranking using Random Forests.
    - Sequential modeling using LSTM layers for time-series forecasting.
3. **Results**:
    - Random Forest achieved high accuracy in spike classification but struggled with temporal dependencies.
    - RNN-LSTM showed strong temporal modeling capabilities, achieving a test accuracy of ~90.65%.
    - Gaussian smoothing effectively reduced noise and false positives.
4. **Code Implementation**:
    - Python scripts utilizing libraries such as TensorFlow, Scikit-learn, and Matplotlib.
    - Feature engineering and visualizations for demand and price analysis.

Data Used

- **Sources**: AEMO datasets
    - Historical electricity prices.
    - Actual and forecast demand values.
    - Rooftop solar generation data.
    - Hourly temperature records for capital cities.
- **Regions Covered**: Adelaide, Brisbane, Sydney, Canberra, Melbourne, and Hobart.
- **Focus Region**: NSW.

Key Features

- **OPERATIONAL_DEMAND**: Real-time electricity demand.
- **Lagged Demand/Price**: Capturing temporal dependencies.
- **Day_of_Week and Hour_of_Day**: Categorical features for cyclical consumption trends.

- **Temperature**: Key external factor impacting demand.

Code Overview

- **Preprocessing**:
  - Merging and cleaning datasets.
  - Feature scaling using MinMaxScaler.
  - Handling missing data using forward fill.
- **Random Forest Model**:
  - Used for price spike classification.
  - Feature importance analysis.
- **RNN-LSTM Model**:
  - Hybrid architecture with LSTM and RNN layers for sequential data.
  - Binary classification and regression tasks for price forecasting.
- **Visualization**:
  - Time-series plots for demand and prices.
  - Actual vs smoothed price trends.
  - Feature importance bar charts.

How to Run

1. **Setup**:
   - Install required Python libraries: TensorFlow, Scikit-learn, Matplotlib, Seaborn, and NumPy.
   - Place data files (`nsw_demand_actual.csv`, `nsw_prices.csv`, etc.) in the working directory.
2. **Execute Code**:
   - Preprocess data and train models using the provided Python scripts.
   - Generate visualizations for demand, prices, and model outputs.
3. **Results**:

- Examine outputs such as feature importance rankings, smoothed price trends, and model performance metrics.

Results Summary

1. **Metrics**:
    - **Random Forest**: Spike classification accuracy ~82%.
    - **RNN-LSTM**: Forecasting accuracy ~90.65%, with a Mean Absolute Error of ~4.15.
2. **Visualization**:
    - "Actual vs Smoothed Prices" highlighted the effectiveness of Gaussian smoothing.
    - Feature importance and volatility trends provide actionable insights.

Future Work

1. Incorporate additional data sources, such as renewable generation and economic indicators.
2. Extend models to other NEM regions.
3. Optimize LSTM for real-time applications.
4. Analyze seasonal price trends more thoroughly.
