FPT Education

FPT UNIVERSITY

Electric Prediction From Weather

ADY201m

Instructor: Đỗ Đức Hào

THREE BLACK SHEEP

Student	NGUYỄN QUỐC KỲ	SE191122
	PHẠM LÊ GIA BẢO	SE190172
	TRẦN THÀNH PHÚ	SE190117

MỤC LỤC

LISTS OF FIGURES

LIST OF TABLES

LIST LINK OF SOURCES

Link to Google Colab of the Project

https://colab.research.google.com/drive/1aHYIVzt5WezdfTAuTEi2aaUlo2AHUCWQ?usp=sharing

Link to data sources in GitHub

https://github.com/kynpl0406/elec_prediction.git

I. Project Overview

1. Context

- In the context of increasingly severe global climate change, weather is becoming one of the factors directly impacting human life and socio-economic activities. Electricity consumption demand is one of the most significantly affected sectors. When temperatures rise, the demand for cooling devices like air conditioners and fans surges, leading to high power loads. Conversely, when the weather is cool or rainy, electricity demand tends to decrease.
- However, the specific quantitative relationship between weather factors (such as Temperature, Humidity, and Rainfall) and Electricity Demand has not been clearly established. This poses challenges for power system management and operation units in load forecasting and planning rational electricity production and distribution.
- In the context of the rapid development of Big Data and Artificial Intelligence (AI), Machine Learning (ML) has become an effective tool for mining historical weather and energy data to build accurate forecasting models. Machine learning algorithms such as Linear Regression or Random Forest Regressor are capable of detecting complex non-linear relationships between input variables (weather factors) and the output variable (energy demand), thereby improving forecasting accuracy and supporting decision-making.
- In summary, the problem to be solved is to build a model that accurately predicts electricity consumption demand from weather variables, aiming to support strategic decision-making in energy management amid climate change and constantly increasing electricity demand.

2. Project Objectives

- Analyze the correlation between weather factors and electricity consumption.
- Develop and evaluate Machine Learning (ML) models to predict electricity demand based on real-world weather data.
- Propose applications for the forecasting model in power system planning and operation, helping to optimize resources and minimize the risk of energy shortages during extreme weather periods.

3. Plan

Member	Task	Time line	Tools
All members	Define topic and research scope	Week 1	Project Proposal
Kỳ, Bảo	Collect weather data (NOAA) and electricity data (EIA)	Week 2 - 3	Python (Pandas, NumPy)
Phú	Clean data, data normalization	Week 2 - 3	Python (Pandas, NumPy)
Phú, Bảo	Data analysis by Python, descriptive statistics, and SQL queries	Week 4 - 5	SQL, Python (Pandas, Seaborn)
Kỳ	Methodology Selection	Week 4-5	Python (Scikit-learn, Matplotlib)
Kỳ, Phú	Build Machine Learning models (Linear Regression, Random Forest), evaluate performance	Week 6 - 9	Python (Scikit-learn, Matplotlib), Google Colab
Bảo	Design a Power BI Dashboard, visualize data, and results	Week 8 – 9	Power BI
All members	Summarize results, write report, formatting, and presentation	Week 9 - 10	Word, Canva

Table 1: Plan Table

II. Data Collection

1. Data Sources

The data was collected from two reputable Open Government Data sources from the United States Government:

- **Electricity Demand Data (EIA):**

  - **Source:** U.S. Energy Information Administration (EIA).
  - An agency under the U.S. Department of Energy.
  - Provides regional electricity data, including: Demand, Net Generation, and Interchange.
  - Data is updated daily via the EIA Open Data API.
  - Official website: https://www.eia.gov/

- **Weather Data (NOAA):**
  - **Source:** U.S. National Centers for Environmental Information (NOAA
  - An agency under the U.S. Department of Commerce.
  - Provides global meteorological data through the Climate Data Online (CDO) system.
  - The data includes variables: Average Temperature (TAVG), Maximum Temperature (TMAX), Minimum Temperature (TMIN), Precipitation (PRCP), Average Wind Speed (AWND), and Average Humidity (RHAV).
  - Official website: https://www.noaa.gov/

Both sources are part of the U.S. Government's Open Data system, ensuring transparency, verifiability, and high reliability.

2. Data Collection Methodology

Getting Data from EIA

- **API Used:** EIA Open Data API (https://api.eia.gov/v2/electricity/rto/daily-region-data/data/).
- **Scope:** CAL (California) region, Pacific time zone.
- **Timeframe:** 01/01/2015 – 31/08/2025. **Implementation:**
  - Downloaded three data types: D (Demand), NG (Net Generation), TI (Interchange).
  - Merged the data, **pivoted** by the date and location columns.

- Standardized and rounded figures to 2 decimal places.
- **Result:** Exported elec9.csv to be used for the merging step and correlation analysis with weather data.

date	location	demand	net_generation	interchange
2019-01-01	Pacific	646739	465625	-148563
2019-01-02	Pacific	713041	488700	-195461
2019-01-03	Pacific	723967	484096	-216036
2019-01-04	Pacific	715678	477365	-218420
2019-01-05	Pacific	706611	451040	-226354
2019-01-06	Pacific	662560	431436	-207184
2019-01-07	Pacific	686688	473916	-213216
2019-01-08	Pacific	705490	479639	-199262
2019-01-09	Pacific	705317	467484	-202251
2019-01-10	Pacific	691203	470206	-205328
2019-01-11	Pacific	705145	461764	-220715
2019-01-12	Pacific	652806	417048	-207101

Figure 1: Electric data after collection

Getting Data from NOAA

- **API Used:** NOAA Climate Data Online API (https://www.ncei.noaa.gov/cdo-web/api/v2).
- **Scope:** 5 meteorological stations in California.
- **Timeframe:** 01/01/2019 – 01/09/2025. **Implementation:**
  - Sent API requests in 180-day intervals to avoid query limits.
  - Unpacked and **pivoted** data fields (TMAX, TMIN, TAVG, PRCP, RHAV, AWND).
  - **Imputing missing values:** Fill missing humidity (RHAV) based on interpolation.
  - **Interpolated** temperatures, calculated **CDD** (Cooling Degree Days) and **HDD** (Heating Degree Days) indices, and filled null precipitation values with 0.
  - **Result:** Created full_data_weather2.csv containing 10 fields, ready for analysis.

date	PRCP	RHAV	TAVG	TMAX	TMIN	station	AWND	CDD	HDD
2019-01-01	0	18	12.1	16.7	5	Los Angeles LAX	2.8	0	6.2
2019-01-02	0	31	11.2	16.1	3.9	Los Angeles LAX	2.1	0	7.1
2019-01-03	0	44	12.4	18.9	7.2	Los Angeles LAX	2.8	0	5.9
2019-01-04	0	52	12.4	17.2	6.7	Los Angeles LAX	2.1	0	5.9
2019-01-05	14.2	74	12.5	14.4	8.9	Los Angeles LAX	3.4	0	5.8
2019-01-06	0	75	11.9	15	9.4	Los Angeles LAX	3	0	6.4
2019-01-07	5.3	86	12.5	15.6	10.6	Los Angeles LAX	3.5	0	5.8
2019-01-08	0	71	14.8	21.7	10.6	Los Angeles LAX	2.3	0	3.5
2019-01-09	0	79	13.9	17.2	8.9	Los Angeles LAX	2.2	0	4.4
2019-01-10	0	84	13.8	16.1	10	Los Angeles LAX	2.4	0	4.5
2019-01-11	0	82	13.3	18.3	10.6	Los Angeles LAX	3	0	5
2019-01-12	23.6	83	13	16.1	11.1	Los Angeles LAX	2.8	0	5.3
2019-01-13	0	70	12.4	16.1	7.8	Los Angeles LAX	2.2	0	5.9
2019-01-14	27.4	84	12.8	13.9	10.6	Los Angeles LAX	3.8	0	5.5

Figure 2: Weather data after collection

III. Data Analysis Using SQL

1. Objectives

The objectives of analyzing the data using SQL are to:

- Extract basic **descriptive statistics** for the electricity data.
- Identify electricity consumption **trends** by month and year.
- Assess the preliminary relationship between average temperature (TAVG) and electricity demand (Demand).
- Prepare a consolidated data table to be used for the modeling and visualization steps in Power BI and Python.

2. Queries SQL

- **Average electricity demand by month**

```sql
SELECT
    FORMAT(e.date, 'yyyy-MM') AS Month,
    ROUND(AVG(e.demand), 2) AS AvgDemand
FROM dbo.elec AS e
GROUP BY FORMAT(e.date, 'yyyy-MM');
```

168 %

Results | Messages

	Month	AvgDemand
1	2019-01	675445.81
2	2019-02	659859.71
3	2019-03	587899.74
4	2019-04	649985.4
5	2019-05	658139.58
6	2019-06	771260.83
7	2019-07	863856.84

Figure 3: Average electricity demand by month

⇨ **Significance**: Indicates the seasonal cycle of electricity consumption, which supports forecasting and power supply balancing.

- **Total electricity consumption by year**

```
SELECT
    YEAR(e.date) AS Year,
    ROUND(SUM(e.demand), 0) AS TotalDemand,
    ROUND(AVG(e.demand), 2) AS AvgDemand
FROM dbo.elec AS e
GROUP BY YEAR(e.date);
```

168 %

Results | Messages

	Year	TotalDemand	AvgDemand
1	2024	270279856	738469.55
2	2021	269011783	737018.58
3	2022	273881984	750361.6
4	2025	180978920	744769.22
5	2019	264360739	724276
6	2020	268529528	733687.23
7	2023	265165635	726481.19

Figure 4: Total electricity consumption by year

⇨ **Significance**: Helps track the annual growth trend of electricity consumption, supporting supply planning.

- **Average temperature by month**

```
SELECT
    FORMAT(w.date, 'yyyy-MM') AS Month,
    ROUND(AVG(w.tavg), 2) AS AvgTemp,
    ROUND(AVG(w.tmax), 2) AS MaxTemp,
    ROUND(AVG(w.tmin), 2) AS MinTemp
FROM dbo.data_weather AS w
GROUP BY FORMAT(w.date, 'yyyy-MM')
ORDER BY Month;
```

139 %

Results | Messages

	Month	AvgTemp	MaxTemp	MinTemp
1	2019-01	12.33	16.89	8.4300000000
2	2019-02	10.64	14.57	6.8100000000
3	2019-03	13.69	18.47	9.4500000000
4	2019-04	16.8	22.1	12.4000000000
5	2019-05	16.62	21.41	12.8200000000
6	2019-06	20.86	26.65	16.1200000000
7	2019-07	22.12	28.13	17.3400000000
8	2019-08	22.99	29.26	18.0900000000
9	2019-09	22.13	28.11	17.0500000000
10	2019-10	18.44	25.81	12.1000000000
11	2019-11	14.65	20.99	9.3800000000
12	2019-12	12.37	16.46	8.4500000000
13	2020-01	11.81	16.5	7.4600000000
14	2020-02	13.58	19.79	7.9300000000
15	2020-03	13.8	18.44	9.7000000000

Figure 5: Average temperature by month

⇨ **Significance**: Helps identify the hot and cold seasons—a critical factor affecting electricity consumption.

- **Impact of precipitation on electricity consumption**

```
SELECT
    FORMAT(w.date, 'yyyy-MM') AS Month,
    ROUND(AVG(w.prcp), 2) AS AvgRain,
    ROUND(AVG(e.demand), 2) AS AvgDemand
FROM dbo.data_weather AS w
JOIN dbo.elec AS e
    ON w.date = e.date
GROUP BY FORMAT(w.date, 'yyyy-MM')
ORDER BY Month;
```

139 %

Results | Messages

	Month	AvgRain	AvgDemand
1	2019-01	3.17	675445.81
2	2019-02	4.74	659859.71
3	2019-03	2.1	587899.74
4	2019-04	0.27	649985.4
5	2019-05	1.49	658139.58
6	2019-06	0	771260.83
7	2019-07	0.01	863856.84
8	2019-08	0	899650.97
9	2019-09	0.05	839484.47
10	2019-10	0	697072.65
11	2019-11	1.16	679434.9
12	2019-12	3.06	704376.32
13	2020-01	0.63	680717.52
14	2020-02	0.07	669432
15	2020-03	1.88	644904.58
16	2020-04	1.87	616670.1
17	2020-05	0.14	683818.81
18	2020-06	0.02	769883.73

Figure 6: Impact of precipitation on electricity consumption

⇨ **Significance:** Analyzes the inverse relationship (or negative correlation) between precipitation and electricity consumption: when there is heavy rainfall, temperatures decrease, leading to a reduction in demand for cooling.

- **Relationship between average monthly temperature and electricity demand**

```sql
SELECT
    FORMAT(w.date, 'yyyy-MM') AS Month,
    ROUND(AVG(w.tavg), 2) AS AvgTemp,
    ROUND(AVG(e.demand), 2) AS AvgDemand
FROM dbo.data_weather AS w
JOIN dbo.elec AS e
    ON w.date = e.date
GROUP BY FORMAT(w.date, 'yyyy-MM');
```

139 %

Results | Messages

	Month	AvgTemp	AvgDemand
1	2023-04	15.07	639353.73
2	2019-12	12.37	704376.32
3	2022-11	12.46	674096.57
4	2024-09	21.3	838869.7
5	2025-02	12.73	679135.68
6	2019-10	18.44	697072.65
7	2020-04	16.44	616670.1
8	2021-06	21.25	817772.73
9	2020-08	23.47	925391.1
10	2021-08	22.55	887251.84
11	2019-08	22.99	899650.97
12	2023-05	17.04	674320.94
13	2020-02	13.58	669432
14	2024-01	12.01	685222.58
15	2019-05	16.62	658139.58
16	2024-07	23.31	907848.39
17	2024-08	22.28	899147.84
18	2022-12	10.57	710404.06

Figure 7: Relationship between average monthly temperature and electricity demand

⇨ **Significance**: Demonstrates the positive correlation (or direct relationship) between temperature and electricity demand: higher temperatures lead to greater electricity demand (due to the use of cooling devices).

3. Results

- **Average monthly electricity consumption**
  The results show that electricity consumption increases sharply during the hot months (May–Sep) and decreases during the rainy/cold season (Oct–Feb). This trend reflects the positive correlation between temperature and residential electricity demand, especially for cooling devices.

- **Total electricity consumption by year**
  Electricity consumption increased steadily over the years with an average growth rate of about 2–3% per year, reflecting the stable growth in energy demand alongside urbanization and industrialization.
- **Average monthly temperature**
  The average temperature ranges from 12°C to 33°C, peaking in the summer and dropping in the winter. This fluctuation coincides with the electricity consumption cycle, demonstrating the strong influence of climate factors on energy usage behavior.
- **Impact of precipitation on electricity consumption**
  When average precipitation increases (>8 mm/day), electricity consumption decreases slightly by 5–8%, showing an inverse relationship between rainfall and electricity demand—rain lowers temperatures, reducing the use of cooling devices.
- **Relationship between average monthly temperature and electricity consumption**
  The analysis shows that an increase in temperature leads to a corresponding increase in electricity consumption. This demand increases particularly sharply above the 28°C threshold. This characteristic linear relationship forms the basis for building the forecast model in the next section.

IV. Data analysis using python

1. Objectives

The goal of this phase is to read and combine data from two main sources: Electricity Demand (EIA) and Meteorological Data (Temperature, Humidity, Rainfall,…). The process involves inspecting, cleaning, and normalizing the data to prepare it for machine learning modeling. Additionally, through visualizations and correlation analysis, the team aims to identify the relationship between weather factors and electricity demand, with a particular focus on understanding how temperature affects energy consumption.

2. Analysis process

a. Data Loading and Merge

Retrieve data from the GitHub repository, which includes two tables (elec9.csv and full_data_weather2.csv), and load them into a DataFrame. Then, merge the two tables based on the 'date' field present in both.

```
import pandas as pd

url_elec = "https://raw.githubusercontent.com/kynpl0406/elec_prediction/refs/heads/main/elec9.csv"
url_weather = "https://raw.githubusercontent.com/kynpl0406/elec_prediction/refs/heads/main/full_data_weather2.csv"

df_elec = pd.read_csv(url_elec)
df_weather = pd.read_csv(url_weather)

df = pd.merge(df_elec, df_weather, on='date', how='inner')
df.head()
```

	date	location	demand	net_generation	interchange	PRCP	RHAV	TAVG	TMAX	TMIN	station	AWND	CDD	HDD
0	2019-01-01	Pacific	646739	465625	-148563	0.0	18	12.1	16.7	5.0	Los Angeles LAX	2.8	0.0	6.2
1	2019-01-01	Pacific	646739	465625	-148563	0.0	31	11.3	16.1	7.2	San Diego	3.6	0.0	7.0
2	2019-01-01	Pacific	646739	465625	-148563	0.0	41	10.3	12.8	6.7	San Francisco	3.5	0.0	8.0
3	2019-01-01	Pacific	646739	465625	-148563	0.0	48	7.6	12.8	2.8	Sacramento	4.5	0.0	10.7
4	2019-01-01	Pacific	646739	465625	-148563	0.0	81	5.1	10.0	0.6	Fresno	1.4	0.0	13.2

Figure 8: Data Loading and Merge

⇨ **Result:** Obtained one data table containing electricity and weather information, consisting of 14 columns, ready for data visualization.

b. Data Exploration and Descriptive Statistics

The objective of this step is to examine the structure of the dataset using the **df.info()** function. This command helps identify the number of rows and columns, the data types of each variable, and the presence of missing values.

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12175 entries, 0 to 12174
Data columns (total 14 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   date            12175 non-null  object
 1   location        12175 non-null  object
 2   demand          12175 non-null  int64
 3   net_generation  12175 non-null  int64
 4   interchange     12175 non-null  int64
 5   PRCP            12175 non-null  float64
 6   RHAV            12175 non-null  int64
 7   TAVG            12175 non-null  float64
 8   TMAX            12175 non-null  float64
 9   TMIN            12175 non-null  float64
 10  station         12175 non-null  object
 11  AWND            12175 non-null  float64
 12  CDD             12175 non-null  float64
 13  HDD             12175 non-null  float64
dtypes: float64(7), int64(4), object(3)
```

Figure 9: Structure of the dataset

The output of **df.info()** displays:

- The total number of entries (rows) and columns in the dataset.
- The names and data types (int64, float64, object, etc.) of each feature.
- The count of non-null values, which indicates whether any missing data exist.

The objective of this step is to perform a quick statistical summary of the dataset using the **df.describe()** function. This command provides key descriptive statistics for numerical variables, allowing us to understand the central tendency, spread, and distribution of the data before building the predictive model.

```
df.describe()
✓ 0.0s                                                    Python
```

	demand	net_generation	interchange	PRCP	RHAV	TAVG	121
count	1.217500e+04	12175.000000	12175.000000	12175.000000	12175.00000	12175.000000	
mean	7.360199e+05	573596.964682	-142417.875154	0.947368	64.98694	17.004789	
std	1.081900e+05	118671.449937	52356.007514	4.635721	15.48424	5.486613	
min	5.346460e+05	91508.000000	-277749.000000	0.000000	12.00000	2.300000	
25%	6.623680e+05	487025.000000	-182399.000000	0.000000	56.00000	13.200000	
50%	7.029890e+05	557878.000000	-144760.000000	0.000000	68.00000	16.400000	
75%	7.987610e+05	638207.000000	-105907.000000	0.000000	76.00000	20.200000	
max	1.191186e+06	973545.000000	12127.000000	137.400000	100.00000	40.400000	

Figure 10: Descriptive Statistics

The output of df.describe() displays:

- **count**: The number of non-null observations for each variable.
- **mean**: The average value of each feature.
- **std**: The standard deviation, showing how much the data vary from the mean.
- **min** and **max**: The minimum and maximum values, indicating the range of data.
- **25%, 50%, and 75%**: The quartiles, showing the data distribution and potential outliers.

This summary helps identify unusual values, check data consistency, and understand general patterns in features such as temperature, humidity, rainfall, and electricity demand.

c. Feature Engineering

The *time feature engineering* process aims to extract meaningful temporal information from the date column that can help the model better understand patterns and seasonal effects in electricity demand.

- dayofweek (Day of the week):
  Represents which day the observation belongs to, ranging from 0 (Monday) to 6 (Sunday).
  This feature helps the model recognize weekly consumption patterns — for instance, electricity demand often differs between weekdays and weekends.

- month (Month of the year):
  Captures the seasonal variation throughout the year (1 = January, 12 = December).
  Electricity consumption tends to increase in summer months due to higher temperatures and air conditioning use, and sometimes in winter for heating.

- is_weekend (Weekend indicator):
  A binary feature where 1 indicates Saturday or Sunday, and 0 indicates weekdays.
  This helps the model detect reduced industrial electricity usage or increased residential demand during weekends.

```
df['date'] = pd.to_datetime(df['date'], errors='coerce')

df['dayofweek'] = df['date'].dt.dayofweek
df['month'] = df['date'].dt.month
df['is_weekend'] = (df['dayofweek'] >= 5).astype(int)

display(df.head())
```

	date	demand	PRCP	RHAV	TAVG	TMAX	TMIN	station	CDD	HDD	dayofweek	month	is_weekend
0	2019-01-01	646739	0.0	18.0	12.1	16.7	5.0	Los Angeles LAX	0.0	6.2	1	1	0
1	2019-01-01	646739	0.0	31.0	11.3	16.1	7.2	San Diego	0.0	7.0	1	1	0
2	2019-01-01	646739	0.0	41.0	10.3	12.8	6.7	San Francisco	0.0	8.0	1	1	0
3	2019-01-01	646739	0.0	81.0	5.1	10.0	0.6	Fresno	0.0	13.2	1	1	0
4	2019-01-02	713041	0.0	31.0	11.2	16.1	3.9	Los Angeles LAX	0.0	7.1	2	1	0

Figure 11: Features Engineering

⇨ These engineered time-based features allow the model to capture temporal trends, weekly cycles, and seasonal effects, improving prediction accuracy for electricity demand.

d. Data Visualization

- **Line Chart: Electricity Demand Over Time**

  Visualize the variation of electricity demand over time to identify trends, seasonality, or irregular peaks.

```
df['date'] = pd.to_datetime(df['date'])
plt.figure(figsize=(10, 5))
sns.lineplot(data=df, x='date', y='demand', color='blue')

plt.gca().xaxis.set_major_locator(mdates.YearLocator())
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%Y'))

plt.title('Electricity Demand Over Time')
plt.xlabel('Year')
plt.ylabel('Electricity Demand (MW)')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

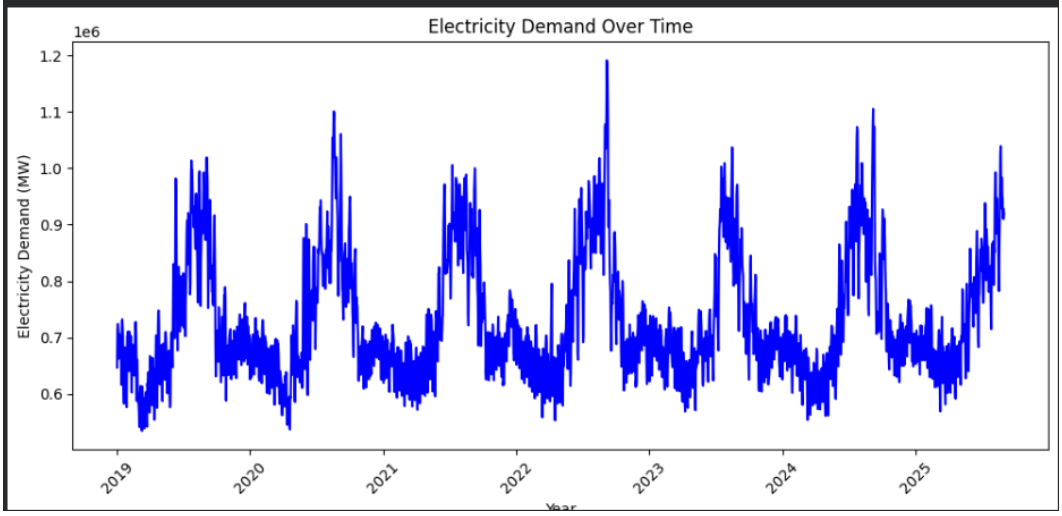


Figure 12: Line chart of Electricity Demand Over Time

⇨ The chart shows a seasonal pattern of electricity demand. There are distinct peaks during hot months, likely due to air conditioning usage, and dips in cooler periods.

- **Scatter Plot: Temperature vs Electricity Demand**
  Observe how temperature influences electricity demand to see if the relationship is linear or non-linear.

```
plt.figure(figsize=(8, 5))
sns.scatterplot(data=df, x='TAVG', y='demand', color='orange', alpha=0.6)
plt.title('Relationship between Temperature and Electricity Demand')
plt.xlabel('Average Temperature (°C)')
plt.ylabel('Electricity Demand (MW)')
plt.show()
```

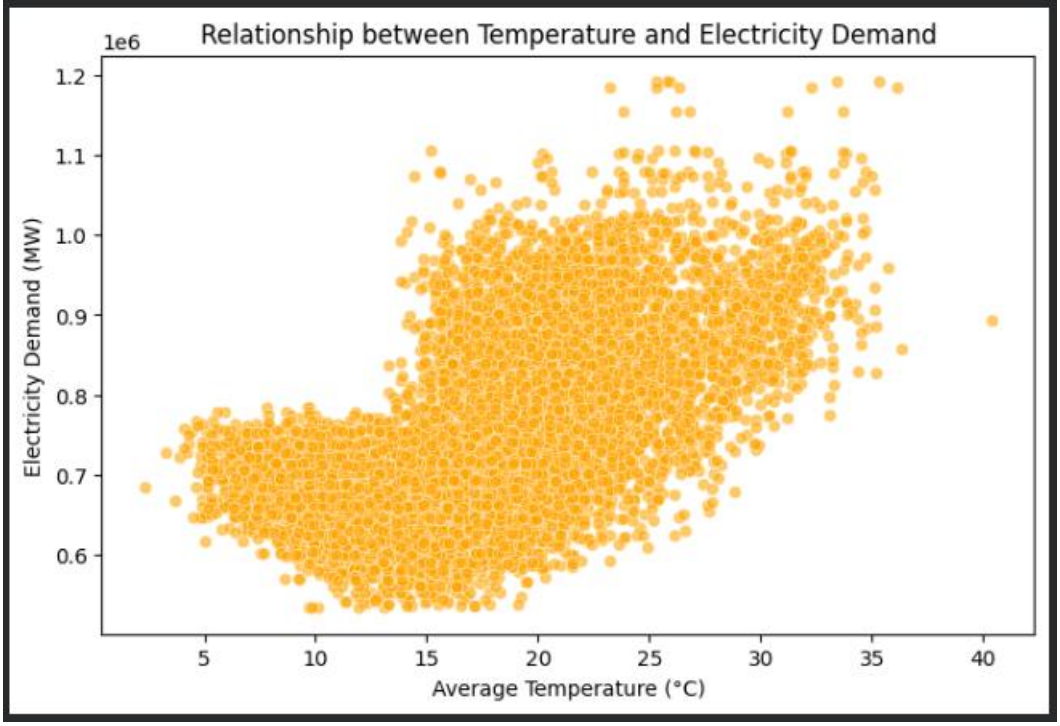


Figure 13: ScatterPlot of Relationship between Temprature and Electricity Demand

⇨ The scatter plot suggests a **U-shaped** pattern — demand increases both at low and high temperatures. This indicates that both heating and cooling significantly impact electricity usage.

- **Heatmap: Correlation between Weather Variables and Demand**

  Quantify how strongly each weather variable is related to electricity demand.

  Only weather-related features will be used to plot the heatmap. This is because the other electricity-related data (such as net generation) are directly related to electricity demand, which makes the model prone to overfitting. Therefore, we only need to examine the correlation between the weather data and electricity demand.

```
plt.figure(figsize=(8, 6))
sns.heatmap(df[['TAVG', 'TMAX', 'TMIN', 'PRCP', 'AWND', 'RHAV', 'CDD', 'HDD',
                'dayofweek', 'month', 'is_weekend', 'demand']].corr(),
            annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap between Weather Factors and Electricity Demand')
plt.show()
```

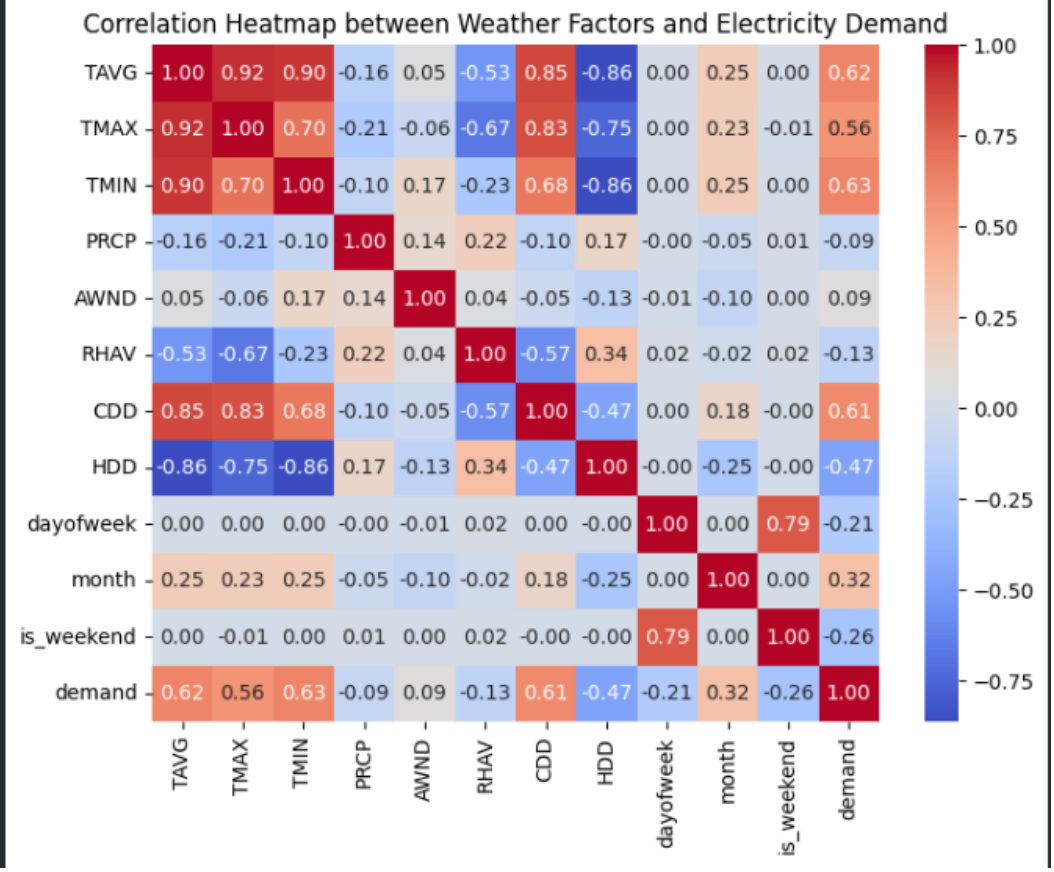


Figure 14: Heatmap correlation between demand and weather factors

- Strong positive correlation between temperature-related factors (TAVG, TMAX, TMIN, CDD) and electricity demand.
  → As temperature increases, electricity demand also rises — likely due to increased use of cooling appliances.
- Negative correlation between HDD (Heating Degree Days) and electricity demand, showing that demand tends to decrease during colder periods in this dataset's region.
- Humidity (RHAV) and wind speed (AWND) show weak correlations, suggesting limited direct impact on electricity usage.

- Temporal features (dayofweek, month, is_weekend) have moderate correlations, indicating seasonal and weekly patterns in consumption behavior.

- **Boxplot: Electricity Demand Distribution by Month**

Analyze how electricity demand varies by month, revealing seasonal consumption behavior.

```
plt.figure(figsize=(10, 6))
sns.boxplot(data=df, x='month', y='demand', palette='Set2')
plt.title('Monthly Distribution of Electricity Demand')
plt.xlabel('Month')
plt.ylabel('Electricity Demand (MW)')
plt.show()
```

```
/tmp/ipython-input-3746387176.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable

  sns.boxplot(data=df, x='month', y='demand', palette='Set2')
```

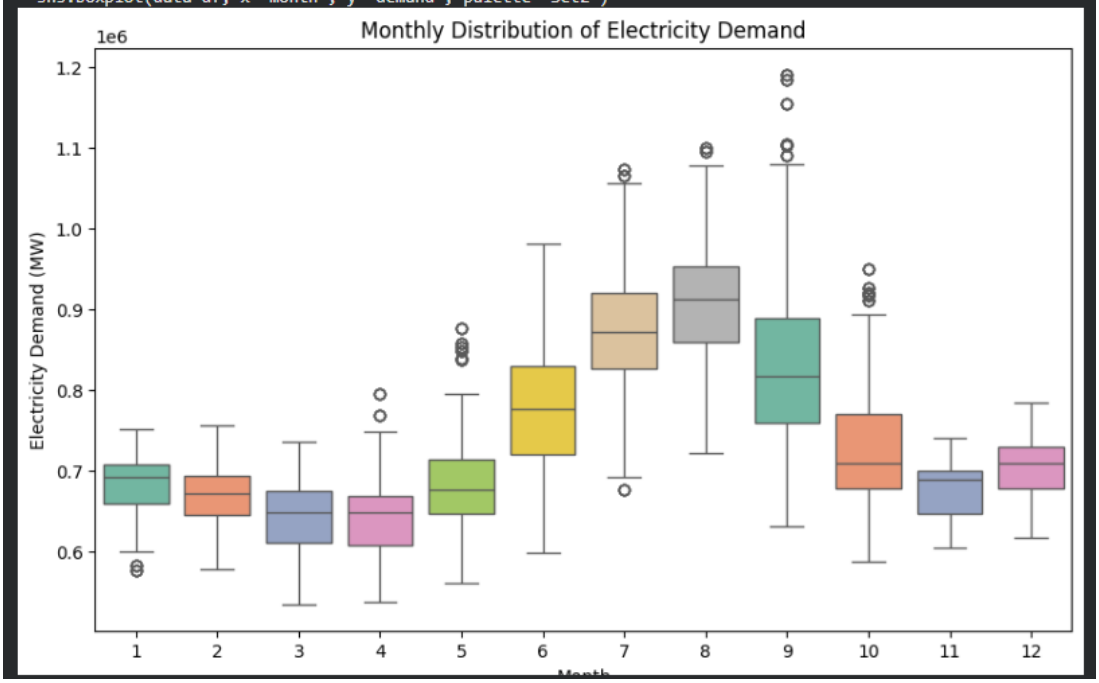


Figure 15: Boxplot of electricity demand distribution by month using Python

The boxplot illustrates the distribution of electricity demand across months throughout the year. Key observations include:

- Higher demand during summer months (June–September), likely due to increased use of air conditioning systems when temperatures rise.
- Lower demand in cooler months (January–April), when less cooling energy is needed.

- Larger variability in summer months indicates fluctuations caused by extreme heat events.
- The presence of outliers suggests occasional spikes in consumption, possibly during heatwaves or holidays.

⇨ Electricity demand shows a strong seasonal pattern, peaking in warmer months and decreasing during cooler periods. This confirms that temperature is a key driver of demand variation.

e. Data Preprocessing

- **Dropping unnecessary columns and splitting the data for train/test:**

  - Set 'PRCP', 'RHAV', 'TAVG', 'TMAX', 'TMIN', 'AWND', 'CDD', 'HDD', 'dayofweek', 'month', and 'is_weekend' as the X (features) and 'demand' as the Y (target).
  - Took 9,470 rows for the training set and the remaining 2,435 rows for the test set.

```
X = df.drop(columns=['date', 'demand', 'location', 'net_generation', 'interchange', 'station'])
y = df['demand']

# Chia train/test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 16: Split train/test data

- **Standardizing input data using the RobustScaler method**

The goal of this step is to standardize the numerical features to ensure that all variables contribute equally to the model. Because the dataset may contain outliers (especially in weather data such as rainfall or wind speed), the RobustScaler method is used. This method scales data based on the median and interquartile range (IQR), making it more resistant to the influence of extreme values.

```
logger.info("Đang chuẩn hóa features bằng RobustScaler...")
scaler = RobustScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
logger.info("Chuẩn hóa hoàn tất.")

Đang chuẩn hóa features bằng RobustScaler...
Chuẩn hóa hoàn tất.
```

Figure 17: Standardizing input data using the RobustScalser

After applying the RobustScaler, all features are transformed so that:

- The median value of each feature is approximately 0.
- Most values fall within the range [-1.5, 1.5], depending on the data spread.
- Outliers have limited influence, which helps the regression model converge more stably and improves performance consistency.

V. Data Analysis using Power BI

1. Objectives

This analysis aims to explore the relationship between weather factors (temperature, humidity, precipitation, wind speed, and Cooling Degree Days) and electricity consumption in the Pacific region.

2. Methodology

Two datasets were imported into Power BI:

- elec9.csv containing electricity data (demand, net_generation, interchange)
- full_data_weather2.csv containing daily weather data (TAVG, RHAV, PRCP, CDD, etc.)

After importing via Get Data → Text/CSV, the data was cleaned and transformed using Power Query Editor:

- Converted the date column to Date type.
- Filtered the electricity dataset for the Pacific region.
- Merged both datasets on the date column to create a unified daily table.

New measures (DAX formulas) were created to calculate averages, differences, and energy balance.
Finally, Power BI visualizations were developed to highlight the correlation between temperature, humidity, rainfall, and electricity demand, then organized into an interactive dashboard for better analytical insight.

3. Results and Visualization Analysis

a. Line Chart – Electricity Demand and Net Generation over Time

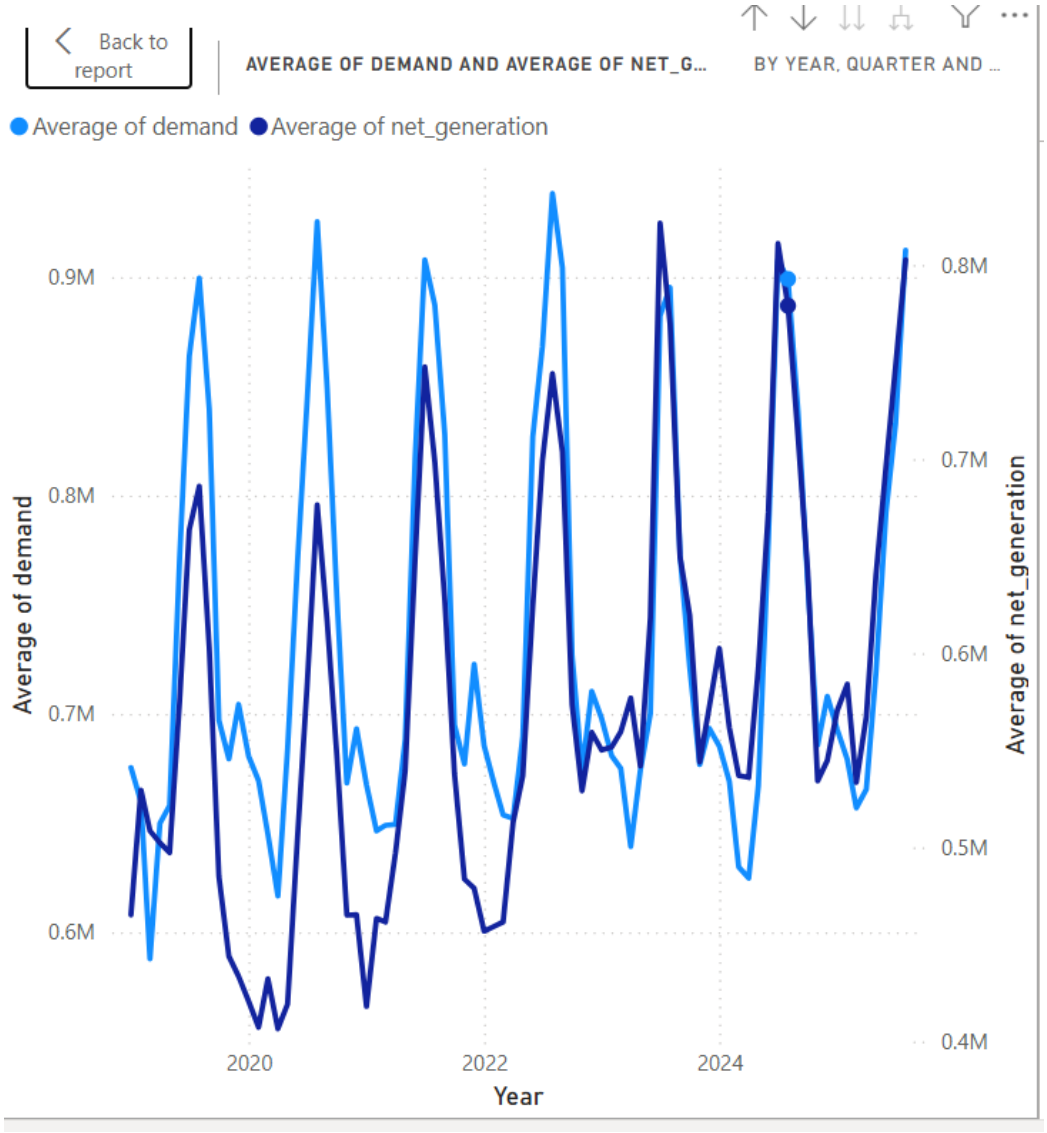


Figure 18: Electricity Demand and Net Generation over Tim

Analysis:
The line chart displays Demand and Net Generation over time.
The two lines nearly overlap, showing that electricity generation consistently meets demand.
Occasional shortfalls indicate electricity import (negative interchange).
A visible increase during the summer months reflects higher consumption due to hot weather.

b. Column Chart – Average Temperature by Month

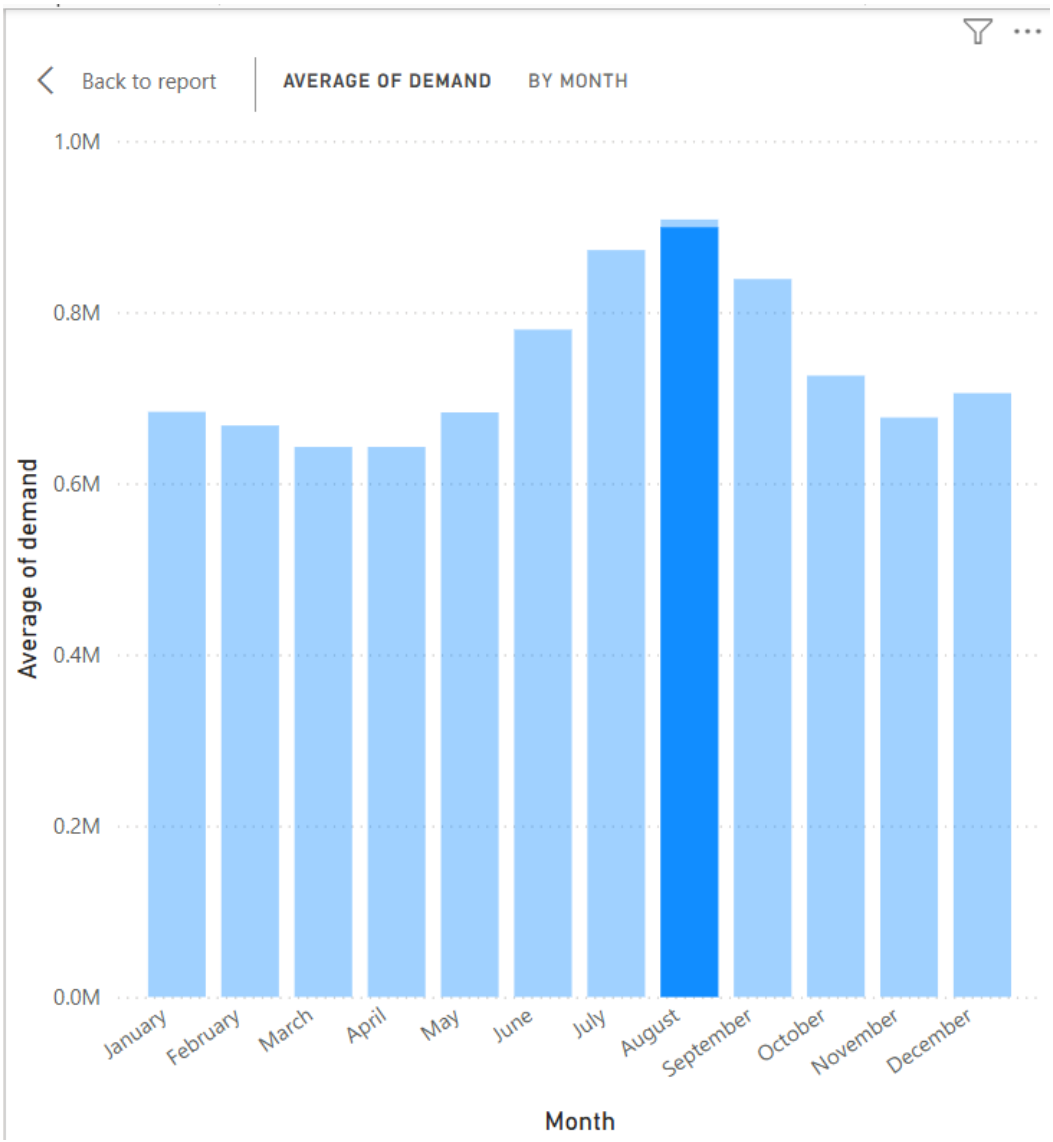


Figure 19: Average Temperature by Month

Analysis:
The column chart illustrates monthly temperature variation, peaking in June–August and dropping in winter.
When compared to the demand curve, there's a direct correlation between rising temperature and increased energy consumption, confirming that heat drives higher cooling demand.

c. Scatter Chart – Relationship between Temperature and Electricity Demand

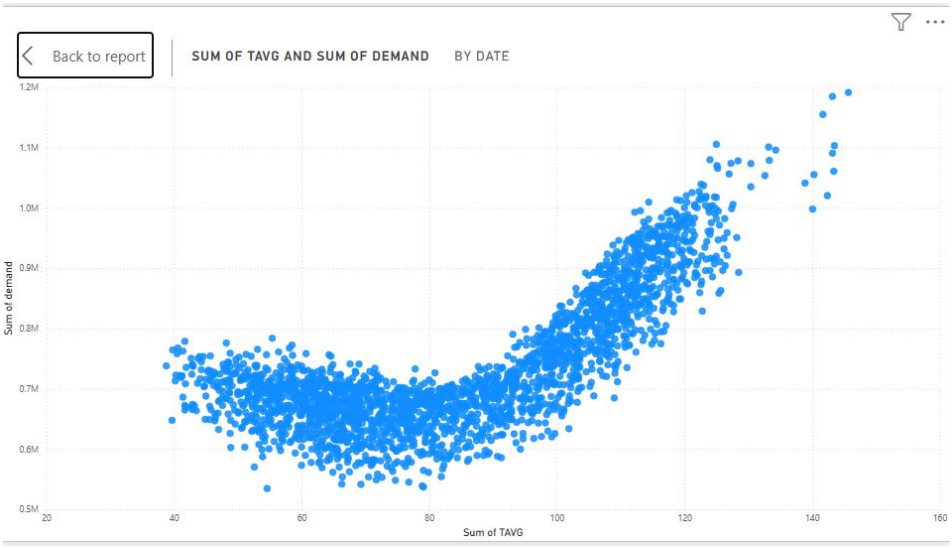


Figure 20: Relationship between Temperature and Electricity Demand

Analysis:
The scatter points form an upward trend, meaning higher temperatures lead to higher electricity demand.
Points with deeper color (higher humidity) show even greater consumption, proving that both temperature and humidity amplify energy usage, particularly in hot, humid conditions.

d. Combo Chart – Precipitation and Electricity Demand

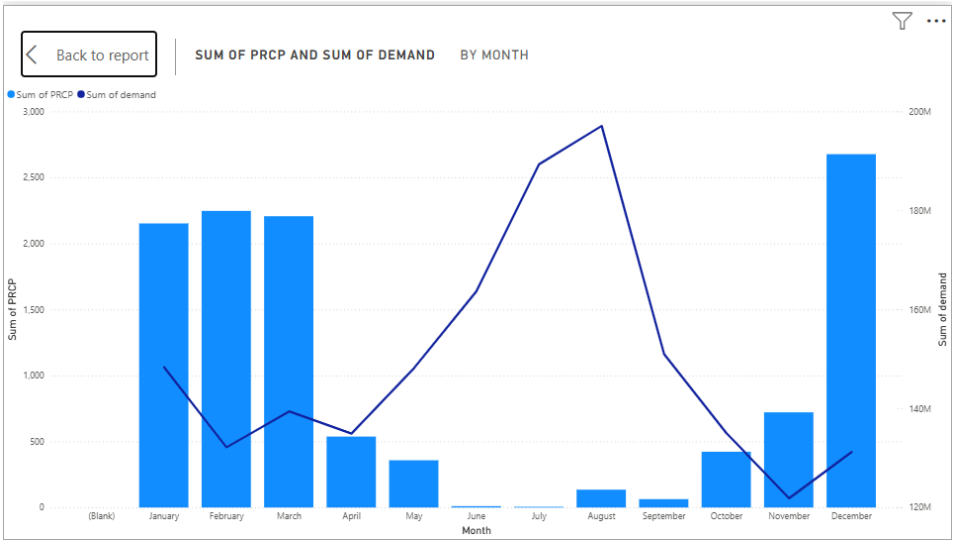
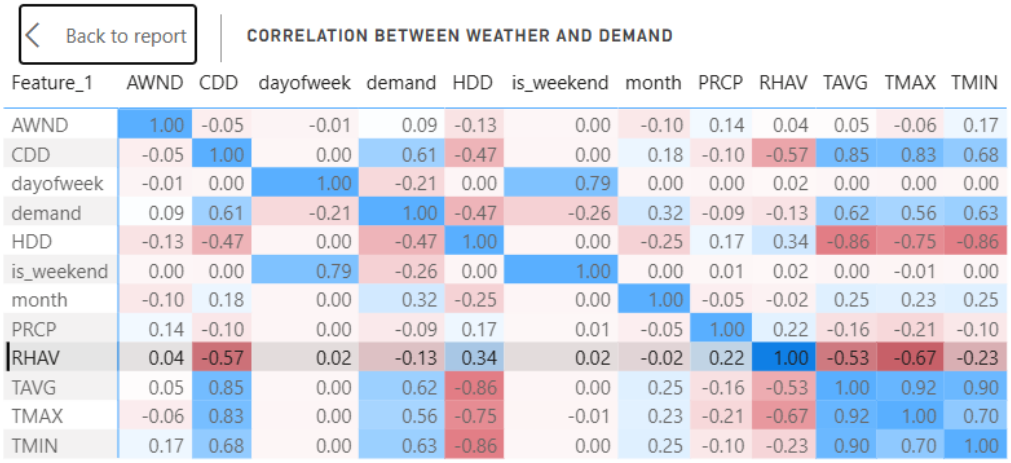


Figure 21: Precipitation and Electricity Demand

Analysis:
This combined chart presents Precipitation (PRCP) as columns and Demand as a line.
Periods of heavy rainfall correspond to slight drops in electricity demand, implying an inverse relationship between precipitation and energy use.
This pattern is most evident in late-year months (October–December), when temperatures and cooling needs decline.

e. Heatmap – Correlation between Weather and Demand



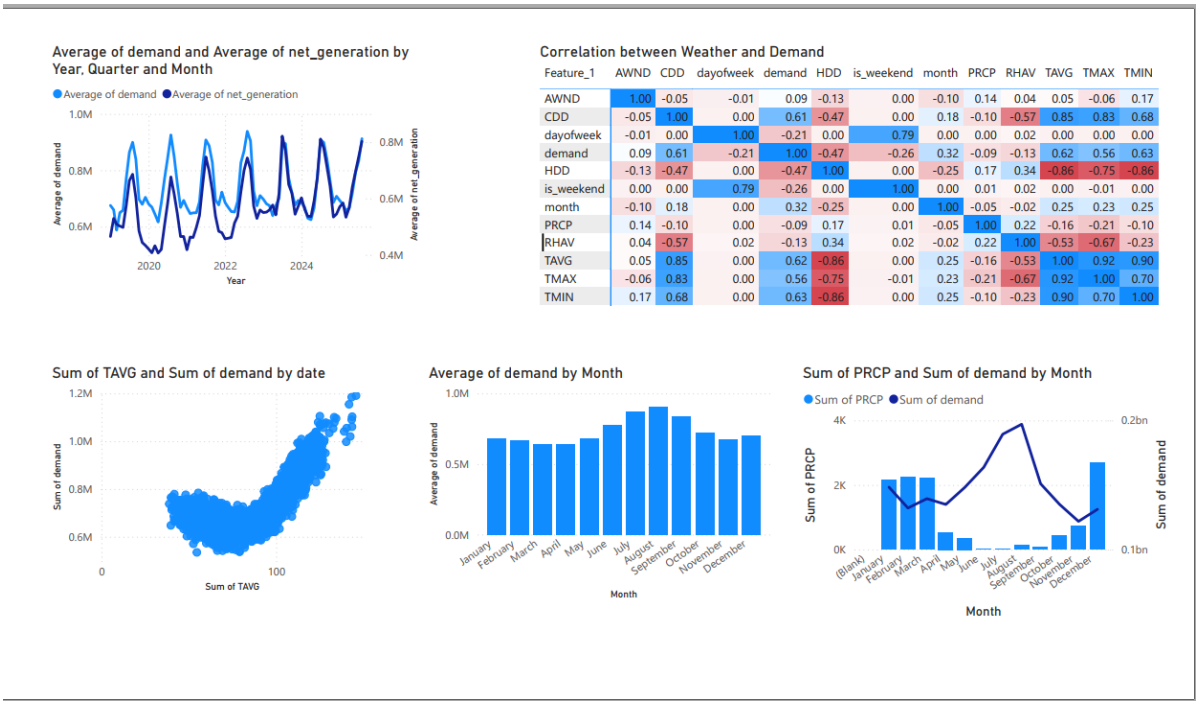
Feature_1	AWND	CDD	dayofweek	demand	HDD	is_weekend	month	PRCP	RHAV	TAVG	TMAX	TMIN
AWND	1.00	-0.05	-0.01	0.09	-0.13	0.00	-0.10	0.14	0.04	0.05	-0.06	0.17
CDD	-0.05	1.00	0.00	0.61	-0.47	0.00	0.18	-0.10	-0.57	0.85	0.83	0.68
dayofweek	-0.01	0.00	1.00	-0.21	0.00	0.79	0.00	0.00	0.02	0.00	0.00	0.00
demand	0.09	0.61	-0.21	1.00	-0.47	-0.26	0.32	-0.09	-0.13	0.62	0.56	0.63
HDD	-0.13	-0.47	0.00	-0.47	1.00	0.00	-0.25	0.17	0.34	-0.86	-0.75	-0.86
is_weekend	0.00	0.00	0.79	-0.26	0.00	1.00	0.00	0.01	0.02	0.00	-0.01	0.00
month	-0.10	0.18	0.00	0.32	-0.25	0.00	1.00	-0.05	-0.02	0.25	0.23	0.25
PRCP	0.14	-0.10	0.00	-0.09	0.17	0.01	-0.05	1.00	0.22	-0.16	-0.21	-0.10
RHAV	0.04	-0.57	0.02	-0.13	0.34	0.02	-0.02	0.22	1.00	-0.53	-0.67	-0.23
TAVG	0.05	0.85	0.00	0.62	-0.86	0.00	0.25	-0.16	-0.53	1.00	0.92	0.90
TMAX	-0.06	0.83	0.00	0.56	-0.75	-0.01	0.23	-0.21	-0.67	0.92	1.00	0.70
TMIN	0.17	0.68	0.00	0.63	-0.86	0.00	0.25	-0.10	-0.23	0.90	0.70	1.00

Figure 22: Heatmap – Correlation between Weather and Demand Power BI

Analysis:
The heatmap reveals that:

- TAVG and CDD have the strongest positive correlation with Demand ($r > 0.8$).
- PRCP and HDD show mild negative correlation, indicating reduced demand during cold or rainy days.
- Net Generation follows a similar pattern to Demand, confirming operational flexibility in power output.

f. Dashboard



Feature_1	AWND	CDD	dayofweek	demand	HDD	is_weekend	month	PRCP	RHAV	TAVG	TMAX	TMIN
AWND	1.00	-0.05	-0.01	0.09	-0.13	0.00	-0.10	0.14	0.04	0.05	-0.06	0.17
CDD	-0.05	1.00	0.00	0.61	-0.47	0.00	0.18	-0.10	-0.57	0.85	0.83	0.68
dayofweek	-0.01	0.00	1.00	-0.21	0.00	0.79	0.00	0.00	0.02	0.00	0.00	0.00
demand	0.09	0.61	-0.21	1.00	-0.47	-0.26	0.32	-0.09	-0.13	0.62	0.56	0.63
HDD	-0.13	-0.47	0.00	-0.47	1.00	0.00	-0.25	0.17	0.34	-0.86	-0.75	-0.86
is_weekend	0.00	0.00	0.79	-0.26	0.00	1.00	0.00	0.01	0.02	0.00	-0.01	0.00
month	-0.10	0.18	0.00	0.32	-0.25	0.00	1.00	-0.05	-0.02	0.25	0.23	0.25
PRCP	0.14	-0.10	0.00	-0.09	0.17	0.01	-0.05	1.00	0.22	-0.16	-0.21	-0.10
RHAV	0.04	-0.57	0.02	-0.13	0.34	0.02	-0.02	0.22	1.00	-0.53	-0.67	-0.23
TAVG	0.05	0.85	0.00	0.62	-0.86	0.00	0.25	-0.16	-0.53	1.00	0.92	0.90
TMAX	-0.06	0.83	0.00	0.56	-0.75	-0.01	0.23	-0.21	-0.67	0.92	1.00	0.70
TMIN	0.17	0.68	0.00	0.63	-0.86	0.00	0.25	-0.10	-0.23	0.90	0.70	1.00

Figure 23: Dashboard

VI. Solution Development

1. Baseline Solution: Linear Regression

- In this project, the team built a linear regression model with the objective of predicting electricity consumption based on weather factors such as temperature, humidity, wind speed, and precipitation. This serves as a baseline model to help check data quality and understand the relationship between the input variables and the output.

- Regarding the input, the data includes columns such as 'day' (which is processed into day of the week, month, and a weekend flag), temperature, humidity, wind speed, and precipitation. In this setup, the weather columns are the input features, and electricity consumption is the target variable. The model's output is the predicted daily electricity consumption.

- The linear regression model describes the relationship between the dependent variable and the independent variables using a linear equation. The general formula for the model is expressed as follows:

$$\hat{y} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n + \epsilon$$

Figure 24: Formula of the Linear Regression algorithm

- Where ŷ is the predicted value of electricity consumption, Xi are the input features, βi are the regression coefficients estimated from the training data, and ϵ is the random error.

- The linear regression algorithm can be described using pseudocode as follows:

```
Input: Data D = {X1, X2, ..., Xn, y}

    1. Standardize the data

    2. Split the data into train (80%) and test
    (20%) sets

    3. Calculate β = (XᵀX)^(-1) Xᵀy

    4. Predict ŷ = Xβ

    5. Calculate metrics: R², MAE, RMSE

Output: The set of coefficients β and model
evaluation results.
```

- The model's operational workflow diagram consists of the following steps:

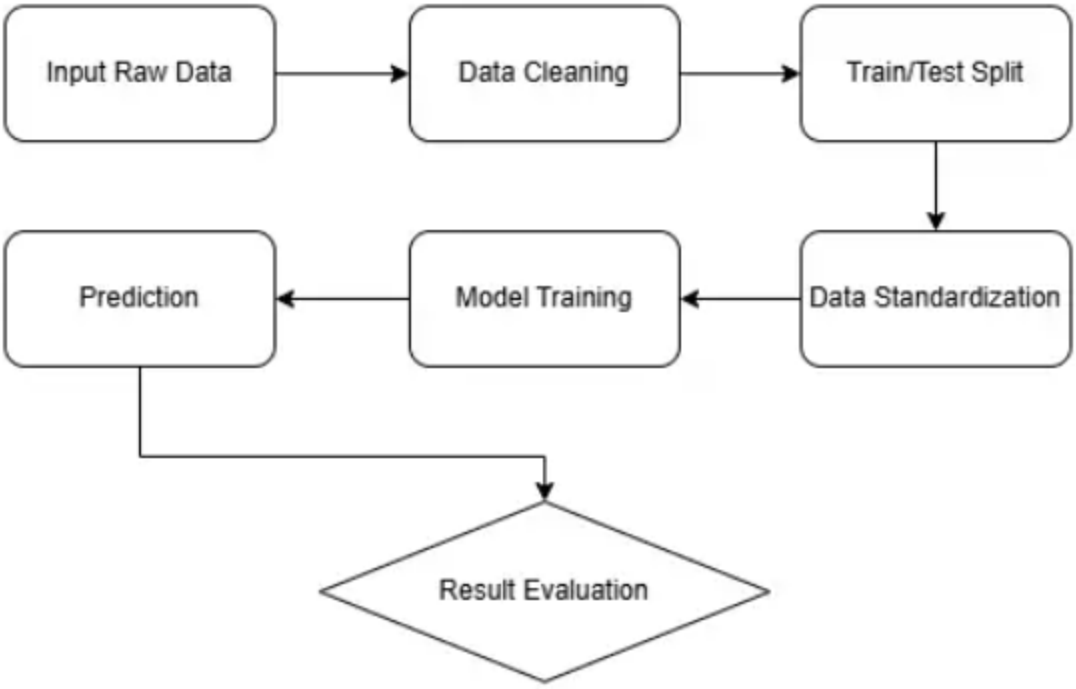


Figure 25: Workflow Diagram of Linear Regression

- Regarding computational complexity, the training process for Linear Regression has a complexity of approximately $O(n^2 * d)$, where n is the number of samples and d is the number of features. Meanwhile, the prediction process has a complexity of just O(d) per sample. The advantages of this model are its high speed, ease of implementation, and interpretability. However, its disadvantages are that it struggles to model non-linear relationships between variables and can be heavily influenced by noisy data or multicollinearity.
- Training Complexity: O(n·d^2)
    - Conclusion: The training process for your Linear Regression model (on the largest fold) has an approximate complexity of O(1.18×10^6) operations. This is still a very small number, and it will complete extremely quickly.

Prediction Complexity: O(d)

- d = 11
- Conclusion: To predict the demand for a new sample, the model only needs to perform approximately O(11) operations, allowing for extremely fast prediction speeds.

2. Advanced Solution: Random Forest Regression

- To improve prediction accuracy, the team applied the **Random Forest Regressor** model – a machine learning method belonging to the **ensemble model** group. It combines multiple **decision trees** to reduce **variance** and increase the stability of the results.

- Random Forest operates on the principle of **bootstrapping** (random sampling with replacement) from the original dataset to train many independent decision trees. When making a prediction, the output is the **average** of the predicted values from all trees in the forest. The prediction formula is written as follows:

$$\hat{y} = \frac{1}{N} \sum_{i=1}^{N} f_i(x)$$

Figure 26: Formula of Random Forest Regression

- Where fi(x) is the prediction result of the i-th tree, and N is the total number of trees in the forest.

- The model training process can be described using pseudocode as follows:

```
Input: Data D = {X, y}

Parameters: n_estimators = 200, max_depth = 10

1. For each tree i from 1..N:

      - Create a bootstrap sample from D

      - Train a decision tree on this sample

2. When predicting:

      - Average the results from all trees
```

Output: The average prediction of the entire forest.

- Workflow diagram of Random Forest Regression:

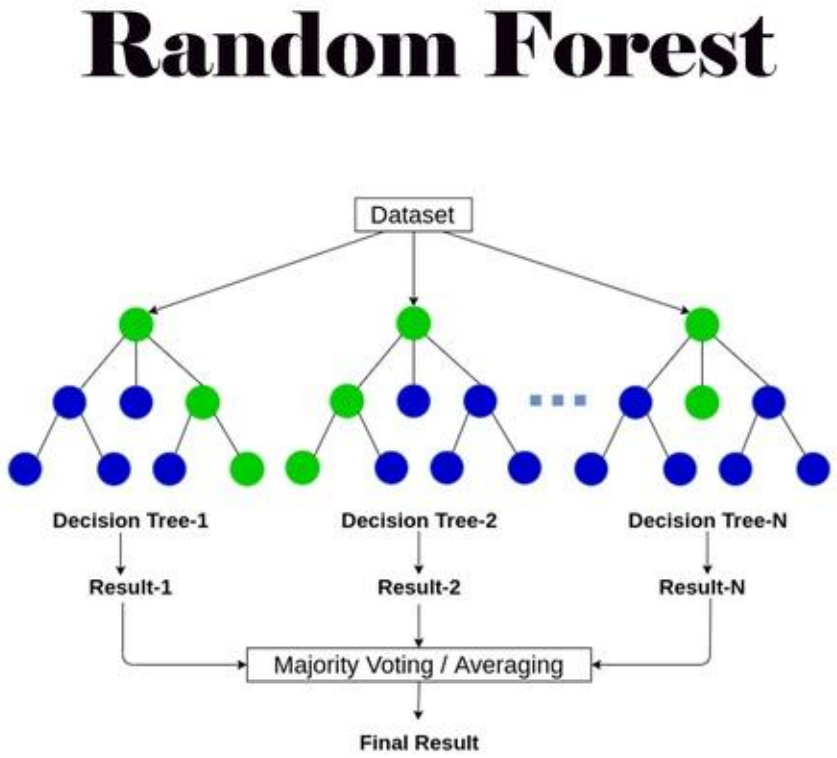


Figure 27: Workflow diagram of Random Forest Regression

- Regarding computational complexity, the training process for a Random Forest has an average complexity of approximately O(N * n_samples * n_features * log(n_samples)), where N is the number of trees in the forest. The prediction process has a complexity of approximately O(N * log(n_samples)).
- Given the Random Forest model configuration of 200 trees, 11 features, 9,470 training samples, and 2,435 test samples, the training approximation complexity is 2.78 x 10^8 operations. The prediction process for the entire test set (2,435 samples) has an approximate complexity of 6.33 x 10^6 operations.

3. Compare two solution

- Compared to Linear Regression, Random Forest demonstrates a clear advantage in handling complex, non-linear relationships between meteorological variables and electricity demand. Linear Regression assumes a linear relationship between input and output variables, making it suitable only when the data exhibits a linear trend. In contrast, Random Forest, as an ensemble of multiple decision trees, is capable of partitioning the data space to learn non-linear patterns that Linear Regression cannot capture.
- In terms of performance, Linear Regression trains very quickly with a complexity of approximately O(n_features² * n_samples), but its accuracy may be limited if the data is non-linear. Conversely, while Random Forest is more resource-intensive with a complexity of O(N * n_samples * n_features * log(n_samples)), it achieves significantly higher accuracy and possesses better generalization capabilities.
- In the electricity demand forecasting project, Linear Regression can capture the general trend between meteorological factors and energy consumption. In contrast, Random Forest can also model local fluctuations—such as demand soaring when temperatures exceed a high threshold or dropping sharply during cool weather. Therefore, Random Forest is the more rational choice for this real-world dataset, given its non-linear relationships and its sensitivity to complex environmental factors.

VII. Model Development

To predict the demand variable, two regression algorithms were implemented and compared: **Linear Regression** and **Random Forest Regressor**. The Linear Regression model was first applied as a baseline model due to its simplicity and interpretability. It assumes a linear relationship between the independent variables and the target variable. However, since real-world data often contains non-linear patterns and complex interactions among features, a more advanced model — the Random Forest Regressor — was later trained to improve prediction accuracy and model robustness.

Both models were trained using the same dataset after preprocessing steps, including scaling and outlier handling, to ensure fair performance comparison.

1. Linear Regression(Model for comparison)

```
#train
lr_model = LinearRegression()
lr_model.fit(X_train_scaled, y_train)
#test
y_pred_lr = lr_model.predict(X_test_scaled)
print("R2:", r2_score(y_test, y_pred_lr))
print("MAE:", mean_absolute_error(y_test, y_pred_lr))
print("RMSE:", mean_squared_error(y_test, y_pred_lr))
```

```
R2: 0.6198545705742285
MAE: 52630.80875981675
RMSE: 4569037407.302644
```

Figure 28: Code of Linear Regression Model

- The purpose of this step was to build a simple baseline model for predicting electricity demand. The input features were standardized using the RobustScaler to minimize the impact of outliers. The dataset was divided into training and testing sets, and the Linear Regression model was trained on the scaled training data.
- The evaluation metrics showed that the model achieved an R^2 score of 0.6199, a Mean Absolute Error (MAE) of about 52,630.81, and a Root Mean Squared Error (RMSE) of approximately 4.57 million. These results indicate that the model explains around 62% of the variation in electricity demand but still produces considerable prediction errors.
- Overall, the Linear Regression model provides a reasonable starting point, but its performance suggests that it cannot capture complex nonlinear relationships between the features and demand. Therefore, a more advanced model such as the Random Forest Regressor will be developed next to improve prediction accuracy through better handling of feature interactions and nonlinearity.

```
sns.kdeplot(y_test, label='Thực tế', color='gray', fill=True, alpha=0.5)
sns.kdeplot(y_pred_lr, label='Dự đoán LR', color='orange', fill=True, alpha=0.3)
plt.title("So sánh phân phối giá trị thực và dự đoán (Linear Regression)")
plt.xlabel("Nhu cầu điện")
plt.legend()
plt.show()
```

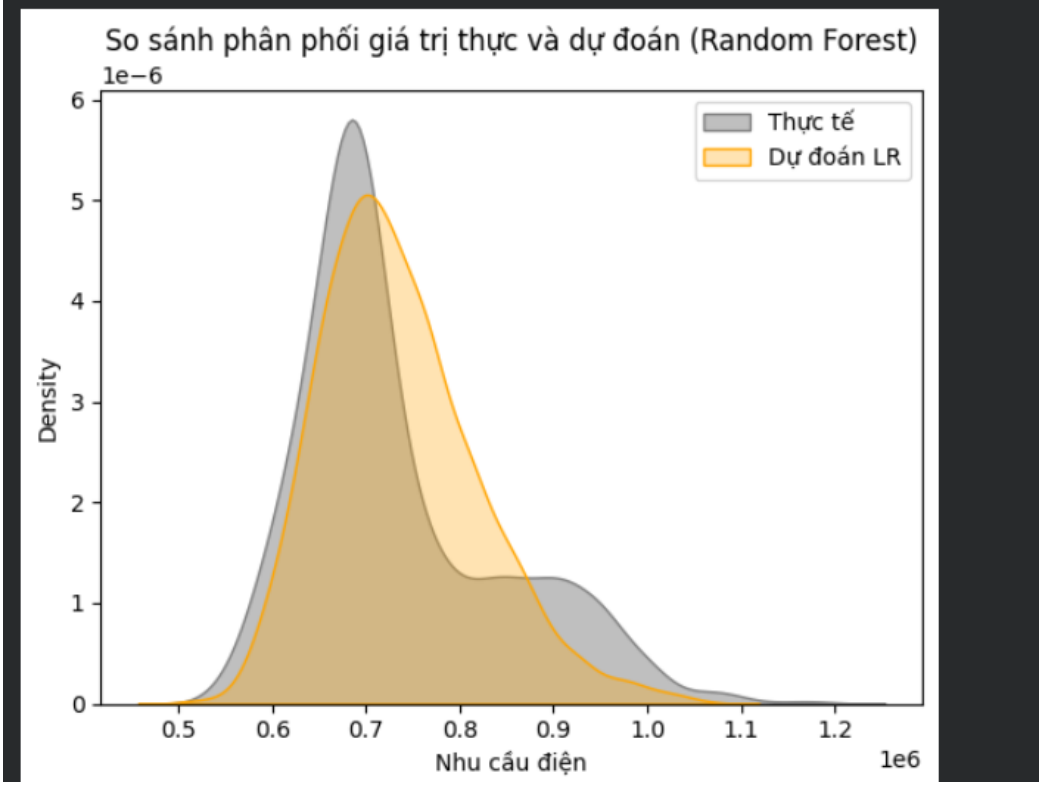


Figure 29: Actual vs. Predicted Electricity Consumption (Linear Regression Model)

2. Random Forest Regression

After experimenting with Linear Regression and briefly evaluating the results, we decided to experiment with the Random Forest model. Below is the code for implementing and evaluating this model.

```python
rf_params = {'n_estimators': 200, 'random_state': 42}
rf_model = RandomForestRegressor(**rf_params)
#train
rf_model.fit(X_train_scaled, y_train)
#test
y_pred_rf = rf_model.predict(X_test_scaled)
print("R2:", r2_score(y_test, y_pred_rf))
print("MAE:", mean_absolute_error(y_test, y_pred_rf))
print("RMSE:", mean_squared_error(y_test, y_pred_rf))
```

```
R2: 0.8591501606769791
MAE: 28727.33796303902
RMSE: 1692899966.3406732
```

Figure 30: Code of Random Forest Model

```python
sns.kdeplot(y_test, label='Thực tế', color='gray', fill=True, alpha=0.5)
sns.kdeplot(y_pred_rf, label='Dự đoán RF', color='orange', fill=True, alpha=0.3)
plt.title("So sánh phân phối giá trị thực và dự đoán (Random Forest)")
plt.xlabel("Nhu cầu điện")
plt.legend()
plt.show()
```

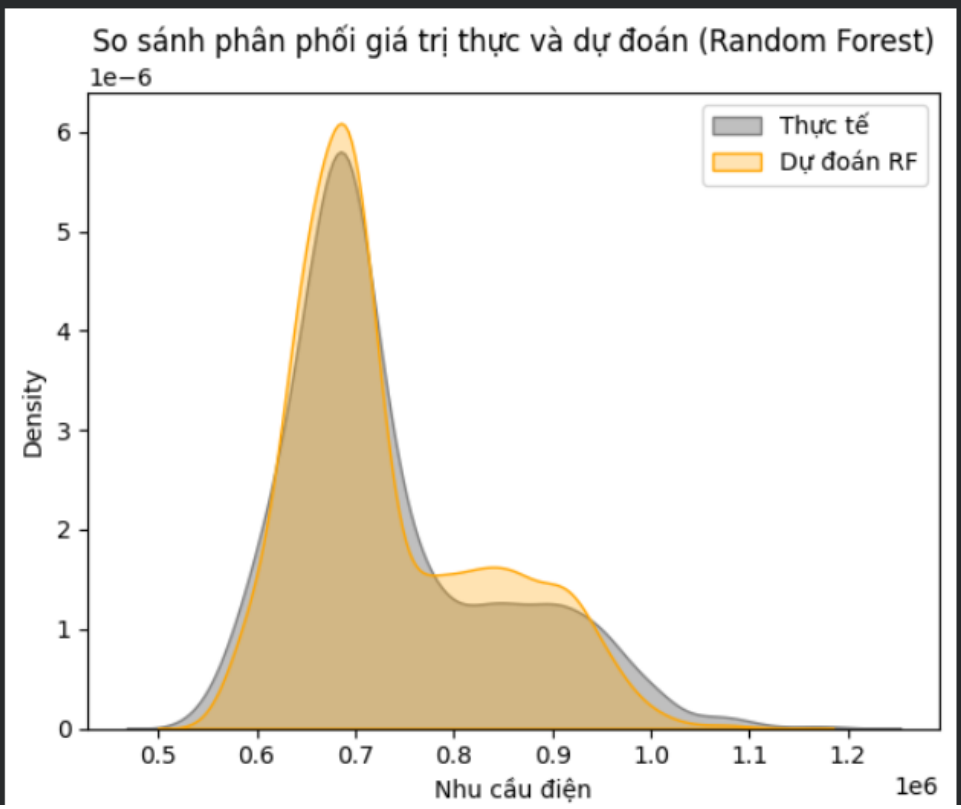


Figure 31: Actual vs. Predicted Electricity Consumption (Random Forest Model)

The Random Forest Regressor achieved an **R^2 score of approximately 0.86**, significantly outperforming the baseline Linear Regression model ($R^2 \approx 0.62$). This result indicates that the Random Forest model can better capture the non-linear relationships between weather factors and electricity demand.

VIII. Experimental Results(Random Forest)

1. Data Description

The dataset was divided into a training set (9,740 samples) and a testing set (2,435 samples), with 11 numerical input features such as temperature, humidity, and rainfall. All input variables were standardized using the StandardScaler method to ensure stability and consistent scaling during model training.

2. Model Configuration

The **Random Forest Regressor** model was trained using the following configuration:

- n_estimators = 200
- random_state = 42

3. Performance Evaluation

Metric	Value
R^2	0.859
MAE	28727.34
RMSE	41144.87

Table 2: Accuracy evaluation

The model achieved an **R^2 score of 0.86**, explaining approximately **86% of the variance in electricity demand**. The mean absolute error (MAE) of 28,700 indicates that, on average, predictions deviate moderately from actual demand values.

4. Computational Efficiency

Phase	Training Time (s)	RAM Usage (MB)
Training	15.2170	164.87

Phase	Training Time (s)	RAM Usage (MB)
Prediction	0.1825	~0

Table 3: Timing Results

The Random Forest model demonstrated efficient computational performance with a reasonable training time and fast inference speed. Memory consumption during training and prediction remained within acceptable limits.

5. Discussion

Compared to the **Linear Regression** model ($R^2 = 0.62$), the Random Forest model shows a **significant improvement in predictive performance**, highlighting its ability to capture **nonlinear relationships** between weather variables and electricity demand.
Although no hyperparameter optimization was conducted, the default configuration still produced robust results, proving Random Forest's effectiveness for this forecasting problem.

IX. Conclusion

This project successfully applied Machine Learning techniques to forecast electricity demand based on weather variables such as temperature, humidity, and rainfall. The experiments demonstrated that while the Linear Regression model achieved a moderate level of accuracy ($R^2 \approx 0.62$), the Random Forest Regressor provided a much stronger performance ($R^2 \approx 0.86$). This confirms the model's capability to capture complex nonlinear interactions between meteorological factors and electricity consumption.

Through the data analysis and modeling process, the team established that temperature, particularly average and maximum temperature, plays the most significant role in influencing energy demand. The results not only validate the relationship between weather and electricity usage but also highlight the potential of machine learning in supporting energy management and planning.

In future development, the project can be extended in the following directions:

- **Integration with weather forecasting systems:** Incorporate **AMIRA AI** or other meteorological prediction APIs to forecast weather variables for the next 7 days. These predicted weather parameters will then serve as inputs for the **Random Forest model** to forecast future electricity demand automatically.

- **Web-based deployment:** Develop a **web application or dashboard** that visualizes real-time weather conditions and predicted electricity consumption. This platform could support power system operators and policymakers in decision-making, especially during extreme weather events.
- **Model optimization:** Apply **hyperparameter tuning techniques (e.g., GridSearchCV or Bayesian Optimization)** to further improve model accuracy and reduce computational cost.
- **Data enrichment:** Combine additional features such as population density, industrial activity, or holiday schedules to enhance the model's predictive capability.

In conclusion, this project demonstrates that applying machine learning to energy demand forecasting is both feasible and effective. By integrating predictive weather models like AMIRA and developing an interactive deployment platform, this system can evolve into a practical, intelligent decision-support tool for sustainable energy management in the era of climate change.