EXPERIMENT-1

# AIM:

To implement efficient methods for **cleaning, loading, handling, and preprocessing time**

**series data** to improve its quality for analysis and modeling.

# STEPS TO IMPLEMENT:

import pandas as pd import numpy as np

from sklearn.preprocessing import MinMaxScaler

Step 1: Load the dataset print("Loading dataset...")

data = pd.read\_csv("C:/Users/pavit/Downloads/PRICE\_AND\_DEMAND\_201801\_NSW1.csv")

print(f"Dataset loaded with shape: {data.shape}")

Step 1: Select relevant columns

print("Selecting relevant columns (TOTALDEMAND and RRP)...") data = data[['TOTALDEMAND', 'RRP']]

print(f"Data after column selection:\n{data.head()}")

Step 2: Convert columns to numeric

print("Converting TOTALDEMAND and RRP to numeric...") data['TOTALDEMAND'] = pd.to\_numeric(data['TOTALDEMAND'], errors='coerce') data['RRP'] = pd.to\_numeric(data['RRP'], errors='coerce')

print("Null values in columns after conversion:")

print(data.isnull().sum())

Step 3: Handle missing values

print("Handling missing values using forward fill...") data = data.fillna(method='ffill')

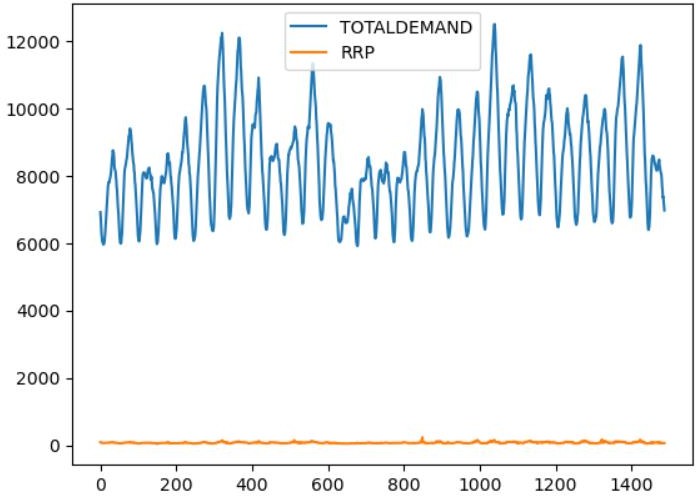
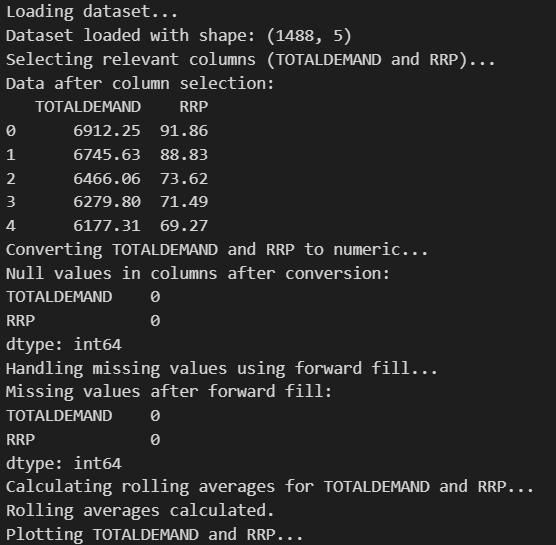
print("Missing values after forward fill:") print(data.isnull().sum())

Step 4: Calculate rolling averages

print("Calculating rolling averages for TOTALDEMAND and RRP...") data['TOTALDEM\_rolling\_avg'] = data['TOTALDEMAND'].rolling(window=7).mean() data['RRP\_rolling\_avg'] = data['RRP'].rolling(window=7).mean()

print("Rolling averages calculated.")

Step 5: Plot TOTALDEMAND and RRP print("Plotting TOTALDEMAND and RRP...") data[['TOTALDEMAND', 'RRP']].plot()



Function to remove outliers

def remove\_outliers(df, column):

print(f"Removing outliers from column: {column}...") Q1 = df[column].quantile(0.25)

Q3 = df[column].quantile(0.75)

IQR = Q3 - Q1

lower\_bound = Q1 - 1.5 \* IQR upper\_bound = Q3 + 1.5 \* IQR

cleaned\_df = df[(df[column] >= lower\_bound) & (df[column] <= upper\_bound)] print(f"Outliers removed from {column}. Data shape: {cleaned\_df.shape}") return cleaned\_df

Reloading dataset... Selecting relevant columns (TOTALDEMAND and RRP)... Dataset reloaded and missing values handled. Removing outliers from column: TOTALDEMAND... Outliers removed from TOTALDEMAND. Data shape: (1488, 2) Removing outliers from column: RRP... Outliers removed from RRP. Data shape: (1455, 2) Removing duplicate rows... Data shape after removing duplicates: (1455, 2) Saving cleaned data to 'cleaned\_price\_demand\_data.csv'... Cleaned data saved. Loading cleaned data and scaling features... Features scaled successfully. Creating lagged features... Lagged features created. Splitting data into training and testing sets... Training data shape: (1164, 8) Testing data shape: (291, 8) Saving training and testing data to CSV files... Training and testing data saved.

Step 6: Split data into training and testing sets print("Splitting data into training and testing sets...") train\_size = int(len(data) \* 0.8)

train\_data, test\_data = data[0:train\_size], data[train\_size:] print(f"Training data shape: {train\_data.shape}") print(f"Testing data shape: {test\_data.shape}")

Step 7: Save preprocessed data to CSV files print("Saving training and testing data to CSV files...")

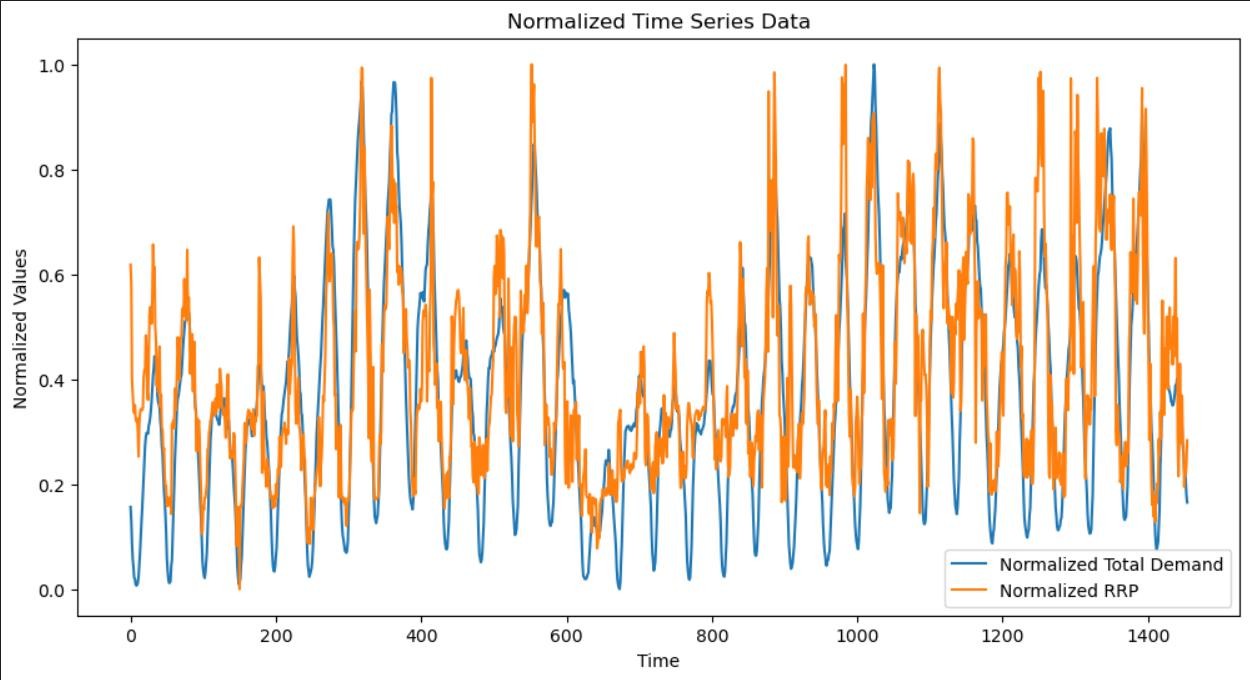
train\_data.to\_csv('train\_price\_demand\_data.csv', index=False) test\_data.to\_csv('test\_price\_demand\_data.csv', index=False) print("Training and testing data saved.")

print("\nData Cleaning and Preprocessing Completed.")

plt.figure(figsize=(12, 6)) plt.plot(data['TOTALDEMAND\_scaled'], label='Normalized Total Demand') plt.plot(data['RRP\_scaled'], label='Normalized RRP') plt.title('Normalized Time Series Data') plt.xlabel('Time')

plt.ylabel('Normalized Values') plt.legend()

plt.show()



VISUALIZATION

# RESULT:

The program to implement programs for time series data cleaning, loading and handling times series data and pre-processing techniques was executed successfully.