EX:No.1 221501060

25/01/25

**Program to implement time series data for import library, load data,Preprocessing and visualising**

**Aim:**

Write a program to implement time series data for import library, load data, Preprocessing and visualising.

**Algorithm:**

1. **Load the Data**:
   * Read the CSV file containing the weather data.
   * Parse the date column as a datetime index.
2. **Clean the Data**:
   * Handle missing values by performing forward and backward filling.
   * Drop any remaining NaN values.
3. **Normalize the Data**:
   * Apply **Min-Max Scaling** to normalize each column's values between 0 and 1.
4. **Add Time-Based Features**:
   * Extract additional features from the datetime index: day, month and year
5. **Visualize the Data**:
   * Plot the time series for a specific column (e.g., temperature T) over time.
6. **Execute the Program**:
   * Sequentially call the functions to load, clean, normalize, add features, and visualize the data.

**Code:**

import pandas as pd

from scipy.stats import zscore

from sklearn.preprocessing import MinMaxScaler

import matplotlib.pyplot as plt

def load\_energy\_data(file\_path):

df = pd.read\_csv(file\_path, sep=';', parse\_dates=['Date'], dayfirst=True,

index\_col='Date', low\_memory=False)

return df

def handle\_missing\_data(df):

df.replace('?', pd.NA, inplace=True)

df = df.apply(pd.to\_numeric, errors='coerce')

df.fillna(method='ffill', inplace=True)

return df

def resample\_data(df, freq='D'):

df\_resampled = df.resample(freq).sum()

return df\_resampled

def remove\_outliers\_zscore(df, threshold=3):

z\_scores = zscore(df)

df\_no\_outliers = df[(z\_scores < threshold).all(axis=1)]

return df\_no\_outliers

def add\_time\_features(df):

df['Year'] = df.index.year

df['Month'] = df.index.month

df['Day'] = df.index.day

df['Weekday'] = df.index.weekday

return df

def scale\_data(df):

scaler = MinMaxScaler()

scaled\_data = scaler.fit\_transform(df[['Global\_active\_power']])

return pd.DataFrame(scaled\_data, columns=['Global\_active\_power'], index=df.index)

def visualize\_data(df):

plt.figure(figsize=(10, 6))

plt.plot(df.index, df['Global\_active\_power'], label='Scaled Global Active Power')

plt.title('Scaled Energy Consumption Over Time')

plt.xlabel('Date')

plt.ylabel('Global Active Power (Scaled)')

plt.legend()

plt.grid(True)

plt.show()

def preprocess\_energy\_data(file\_path):

data = load\_energy\_data(file\_path)

data = handle\_missing\_data(data)

data = resample\_data(data, freq='D')

data = remove\_outliers\_zscore(data[['Global\_active\_power']])

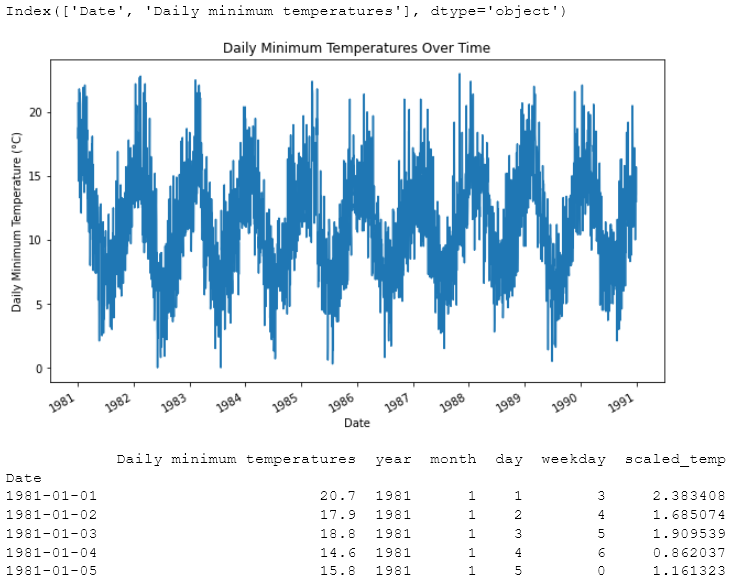
data = add\_time\_features(data)

scaled\_data = scale\_data(data)

visualize\_data(scaled\_data)

return scaled\_data

**Output:**



**Result:**

Thus, the program using the time series data implementation has been done successfully.