EXP No: 1

Implement programs for time series data cleaning, loading and handling time series data and pre-processing techniques

Aim:

To preprocess the time series dataset loading, cleaning and visualization

Objective:

The process is to analyze time series data related to climate change indications

Background:

1. **Seasonality detection:** Understanding cyclic behavior over time
2. **Trend detection:** Identifying long term pattern
3. **Forecasting:** Predicting future values of key climate indicators based on historical data
4. **Anomaly detection:** Identifying unusual data points on outlines

Scope:

Global air quality (PM)

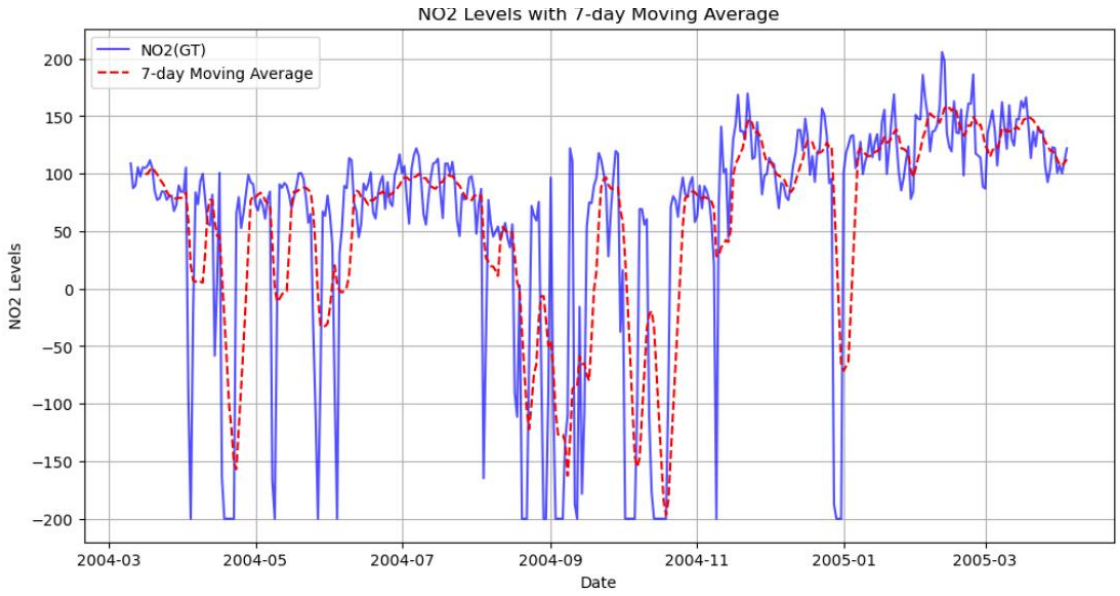
Sensex PM 2.5 pollutant

Code:

```
 import pandas as pd
import matplotlib.pyplot as plt

# Step 1: Load the dataset with the correct delimiter (semicolon)
file_path = r"C:\Users\AI_LAB\Downloads\archive (4)\AirQuality.csv"
df = pd.read_csv(file_path, delimiter=";")

# Step 2: Inspect the column names
print(df.columns)

# Step 3: Clean the 'Time' column by replacing periods with colons
df['Time'] = df['Time'].str.replace('.', ':', regex=False)

# Step 4: Combine 'Date' and 'Time' columns into a single 'datetime' column
df['datetime'] = pd.to_datetime(df['Date'] + ' ' + df['Time'], format='%d/%m/%Y %H:%M:%S')

# Step 5: Drop the original 'Date' and 'Time' columns (optional)
df.drop(['Date', 'Time'], axis=1, inplace=True)
```

```
# Step 6: Set 'datetime' as the index
df.set_index('datetime', inplace=True)

# Step 7: Handle missing data by forward filling
df.fillna(method='ffill', inplace=True)

# Step 8: Resample the data to daily frequency (mean of each day)
df_daily = df.resample('D').mean()

# Step 9: Apply a 7-day moving average to 'NO2(GT)' column
df_daily['NO2_SMA_7'] = df_daily['NO2(GT)'].rolling(window=7).mean()

# Step 10: Visualize the data
plt.figure(figsize=(12, 6))
plt.plot(df_daily.index, df_daily['NO2(GT)'], label='NO2(GT)', color='blue', alpha=0.7)
plt.plot(df_daily.index, df_daily['NO2_SMA_7'], label='7-day Moving Average', color='red',
linestyle='--')
plt.title('NO2 Levels with 7-day Moving Average')
plt.xlabel('Date')
plt.ylabel('NO2 Levels')
plt.legend()
plt.grid(True)
plt.show()

# Step 11: Save the cleaned data to a CSV file
df_daily.to_csv(r"C:\Users\AI_LAB\Downloads\cleaned_air_quality_data.csv")
# Check the cleaned data
print(df_daily.head())
```

Output:



datetime	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	\
2004-03-10	1.966667	1316.500000	86.500000	8.450000	912.333333	
2004-03-11	-6.187500	1244.166667	104.500000	7.979167	851.958333	
2004-03-12	-14.095833	1281.666667	141.500000	12.129167	1008.291667	
2004-03-13	-5.750000	1330.666667	139.250000	10.916667	992.833333	
2004-03-14	-5.966667	1361.125000	116.958333	9.637500	943.916667	

datetime	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S5(O3)	\
2004-03-10	132.000000	1167.333333	108.833333	1545.500000	1096.000000	
2004-03-11	130.041667	1277.250000	87.375000	1522.833333	885.250000	
2004-03-12	142.583333	1101.875000	89.916667	1627.291667	1084.375000	
2004-03-13	168.416667	993.208333	105.583333	1595.791667	1245.916667	
2004-03-14	132.166667	1001.291667	97.458333	1602.375000	1234.208333	

datetime	T	RH	AH	Unnamed: 15	Unnamed: 16	\
2004-03-10	12.033333	54.900000	0.765633	NaN	NaN	
2004-03-11	9.837500	64.075000	0.775767	NaN	NaN	
2004-03-12	11.287500	51.095833	0.663104	NaN	NaN	
2004-03-13	12.866667	51.533333	0.732296	NaN	NaN	
2004-03-14	16.012500	48.850000	0.849671	NaN	NaN	

datetime	NO2_SMA_7
2004-03-10	NaN
2004-03-11	NaN
2004-03-12	NaN
2004-03-13	NaN
2004-03-14	NaN

Result :

Thus the time series programs for data cleaning, pre-processing, handling time series data implemented.