**Implement programs to check stationary of a time series data for the given dataset**

**EX.No:3**

**DATE:**

**AIM:**

To check if a time series is stationary using statistical and visual methods for effective forecasting.

**ALGORITHM:**

1. Load and preprocess data (datetime conversion, missing value handling).
2. Perform ADF test on 'AC\_POWER'.
3. Visualize data distribution and relationships using histograms, box plots, and scatter plots.
4. Visualize trends and patterns using bar charts, heatmaps, and time series plots.
5. Interpret results and draw conclusions about electricity production.

**CODE:**

import pandas as pd

import matplotlib.pyplot as plt

from statsmodels.tsa.stattools import adfuller

df = pd.read\_csv('Plant\_1\_Generation\_Data.csv')

df['DATE\_TIME'] = pd.to\_datetime(df['DATE\_TIME'], dayfirst=True)

df = df.set\_index('DATE\_TIME')

ts\_data = df['AC\_POWER']

numerical\_features = ['DC\_POWER', 'AC\_POWER', 'DAILY\_YIELD', 'TOTAL\_YIELD']

ts\_data = ts\_data.ffill()

df[numerical\_features] = df[numerical\_features].fillna(method='ffill')

adf\_result = adfuller(ts\_data)

df[numerical\_features].hist(figsize=(12, 8))

plt.suptitle('Histograms of Numerical Features', fontsize=16)

plt.show()

df[numerical\_features].plot(kind='box', figsize=(10, 6))

plt.title('Box Plots of Numerical Features', fontsize=14)

plt.show()

plt.figure(figsize=(8, 6))

plt.scatter(df['DC\_POWER'], df['AC\_POWER'])

plt.title('Scatter Plot: DC Power vs. AC Power', fontsize=14)

plt.xlabel('DC Power')

plt.ylabel('AC Power')

plt.show()

daily\_yield\_by\_date = df['DAILY\_YIELD'].resample('D').sum()

daily\_yield\_by\_date.plot(kind='bar', figsize=(12, 6))

plt.title('Daily Yield Over Time', fontsize=14)

plt.xlabel('Date')

plt.ylabel('Daily Yield')

plt.show()

corr\_matrix = df[numerical\_features].corr()

plt.figure(figsize=(8, 6))

plt.imshow(corr\_matrix, cmap='viridis', interpolation='nearest')

plt.colorbar()

plt.title('Correlation Matrix', fontsize=14)

plt.xticks(range(len(numerical\_features)), numerical\_features, rotation=45)

plt.yticks(range(len(numerical\_features)), numerical\_features)

plt.show()

plt.figure(figsize=(12, 6))

plt.plot(ts\_data)

plt.title('AC Power Time Series', fontsize=14)

plt.xlabel('Date Time')

plt.ylabel('AC Power')

plt.grid(True)

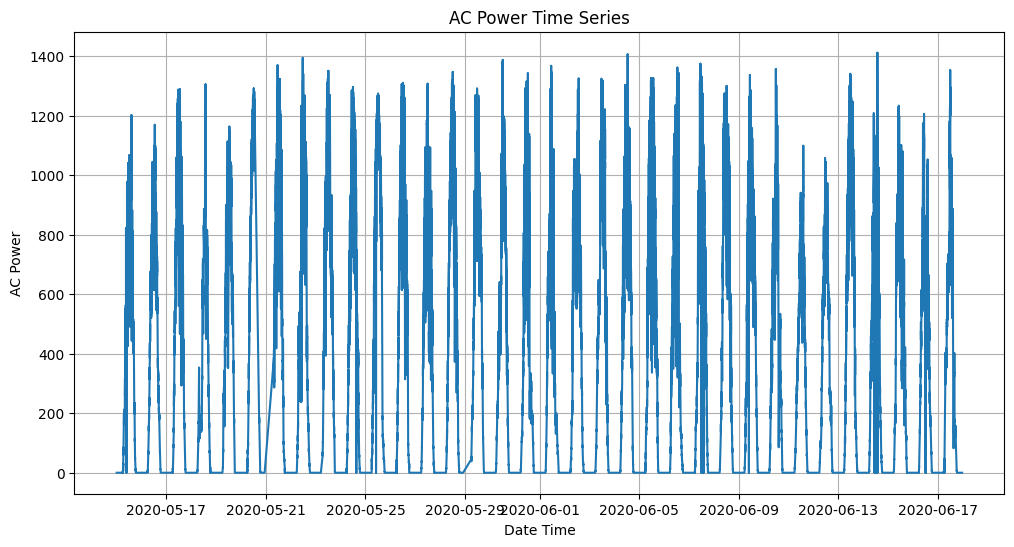
plt.show()

print("ADF Statistic:", adf\_result[0])

print("p-value:", adf\_result[1])

print("Critical Values:", adf\_result[4])

**OUTPUT:**



**RESULT:**

Thus the program has been completed and verified successfully.