**Create an ARIMA model for time series forecasting run in python code**

**EX.No:8**

**DATE:**

**AIM:**

To build an ARIMA model for forecasting future values of a time series based on historical data.

**ALGORITHM:**

1. Load the time series data and set the date as the index.
2. Plot the data and check for stationarity.
3. Select ARIMA parameters (p, d, q).
4. Fit the ARIMA model to the data.
5. Forecast future values and visualize the results.

**CODE:**

import pandas as pd

import numpy as np

from statsmodels.tsa.arima.model import ARIMA

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error

import warnings

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').resample('h').mean(numeric\_only=True)

split = int(len(df) \* 0.8)

train, test = df[:split], df[split:]

def mape(y\_true, y\_pred):

return np.mean(np.abs((y\_true - y\_pred) / (y\_true + 1e-7))) \* 100

warnings.filterwarnings('ignore')

init\_model = ARIMA(train['DC\_POWER'], order=(5, 1, 0)).fit()

init\_pred = init\_model.predict(start=len(train), end=len(df)-1)

print("Initial ARIMA(5,1,0):",

"MAE:", mean\_absolute\_error(test['DC\_POWER'], init\_pred),

"RMSE:", np.sqrt(mean\_squared\_error(test['DC\_POWER'], init\_pred)),

"MAPE:", mape(test['DC\_POWER'], init\_pred))

best\_score, best\_order = float('inf'), None

for p in range(6):

for d in range(2):

for q in range(6):

try:

model = ARIMA(train['DC\_POWER'], order=(p,d,q)).fit()

pred = model.predict(start=len(train), end=len(df)-1)

score = mape(test['DC\_POWER'], pred)

if score < best\_score:

best\_score, best\_order = score, (p,d,q)

except: continue

final\_model = ARIMA(train['DC\_POWER'], order=best\_order).fit()

final\_pred = final\_model.predict(start=len(train), end=len(df)-1)

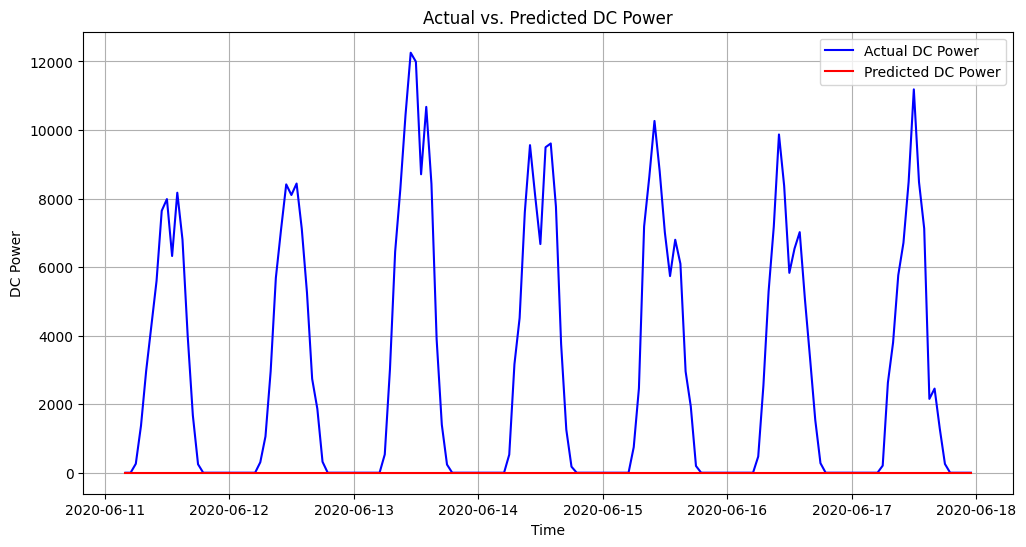
print(f"Best ARIMA{best\_order}:",

"MAE:", mean\_absolute\_error(test['DC\_POWER'], final\_pred),

"RMSE:", np.sqrt(mean\_squared\_error(test['DC\_POWER'], final\_pred)),

"MAPE:", best\_score)

**OUTPUT:**



**RESULT:**

Thus the program has been completed and verified successfully.