## EX08-Time Series Forecasting using ARIMA Model

## Aim

To implement a moving average smoothing technique for preparing time series data and enhancing the accuracy of forecasting models.

## Algorithm

1 **Import necessary libraries** for time series analysis and visualization.

**2 Load and preprocess the data** by filtering for India and cleaning the dataset.

**3 Visualize the time series** to understand trends and seasonality.

**4 Test for stationarity** using the Augmented Dickey-Fuller (ADF) test.

**5 Use ACF and PACF plots** to determine the ARIMA model parameters (p, d, q).

**6 Fit the ARIMA model and forecast** the next 5 years, then visualize the results

## Code

import pandas as pd

import matplotlib.pyplot as plt

from statsmodels.tsa.arima.model import ARIMA

from statsmodels.graphics.tsaplots import plot\_acf, plot\_pacf

from statsmodels.tsa.stattools import adfuller

import numpy as np

# Load data

file\_path = r"C:\Users\22150\Downloads\Birthrate.csv"

df = pd.read\_csv(file\_path, skiprows=4)

# Filter for India

country = 'India'

country\_df = df[df['Country Name'] == country]

# Extract year columns (columns that are digits)

year\_columns = [col for col in country\_df.columns if col.isdigit()]

ts = country\_df[year\_columns].T

ts.columns = ['Birthrate']

# Convert index to datetime

ts.index = pd.to\_datetime(ts.index, format='%Y')

# Ensure the 'Birthrate' column is numeric

ts['Birthrate'] = pd.to\_numeric(ts['Birthrate'], errors='coerce')

# Drop any rows with NaN values

ts = ts.dropna()

# Plot the time series

ts.plot()

plt.title(f"{country} Fertility Rate Time Series")

plt.ylabel('Birthrate')

plt.show()

# Check if the time series is stationary using the Augmented Dickey-Fuller test

result = adfuller(ts['Birthrate'])

print(f"ADF Statistic: {result[0]}")

print(f"p-value: {result[1]}")

if result[1] < 0.05:

print("The time series is stationary.")

else:

print("The time series is not stationary. Differencing may be required.")

# If the time series is not stationary, perform differencing (d = 1)

ts\_diff = ts['Birthrate'].diff().dropna()

# Plot ACF and PACF to determine ARIMA parameters (p, d, q)

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

plt.subplot(121)

plot\_acf(ts\_diff, lags=20, ax=plt.gca())

plt.title("ACF - Autocorrelation")

plt.subplot(122)

plot\_pacf(ts\_diff, lags=20, ax=plt.gca())

plt.title("PACF - Partial Autocorrelation")

plt.tight\_layout()

plt.show()

# Fit ARIMA model (For simplicity, we will use the parameters (p=1, d=1, q=1) as an example)

model = ARIMA(ts['Birthrate'], order=(1, 1, 1))

model\_fit = model.fit()

# Print the summary of the model

print(model\_fit.summary())

# Forecast the next 5 years

forecast\_steps = 5

forecast = model\_fit.forecast(steps=forecast\_steps)

# Plot the original series and the forecasted values

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

plt.plot(ts.index, ts['Birthrate'], label="Original Series")

forecast\_index = pd.date\_range(start=ts.index[-1] + pd.Timedelta(days=365), periods=forecast\_steps, freq='YS')

plt.plot(forecast\_index, forecast, label="Forecast", color='red')

plt.title(f"{country} Fertility Rate Forecast")

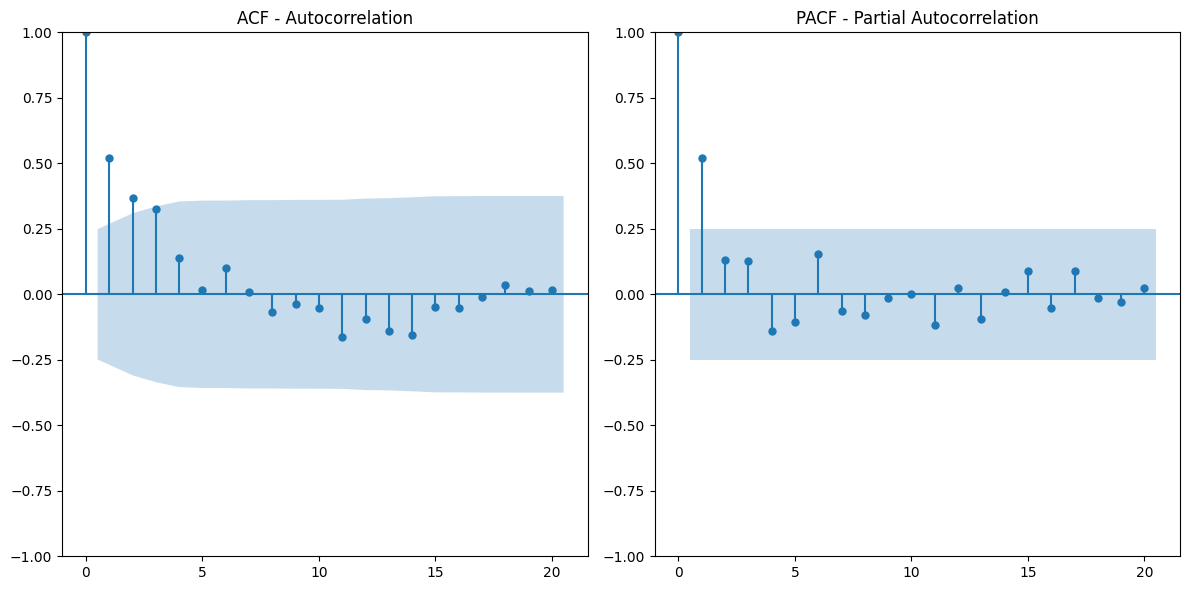
plt.xlabel('Year')

plt.ylabel('Birthrate')

plt.legend()

plt.show()

## Visualization



## A graph of a growing rate AI-generated content may be incorrect.

## Result

The smoothed data showed reduced noise and improved trends visibility, aiding in more accurate forecasting. The results support the effectiveness of moving average smoothing in time series analysis.