| **EXP:8**  **17/04/2025** | **Create an ARIMA model for time series forecasting** |
| --- | --- |

**AIM:**

To analyze car sales data over time and develop an ARIMA time series forecasting model that predicts future sales based on historical trends.

**PROCEDURE:**

**1) Import Necessary Libraries**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from statsmodels.tsa.arima.model import ARIMA

from statsmodels.tsa.stattools import adfuller

from pandas.plotting import register\_matplotlib\_converters

register\_matplotlib\_converters()

**2)** **Load and Inspect the Dataset**

**# Load the CSV file**

**df = pd.read\_csv("Car\_sales.csv")**

**# Display first few rows**

**df.head()**

**3) Preprocess the Data**

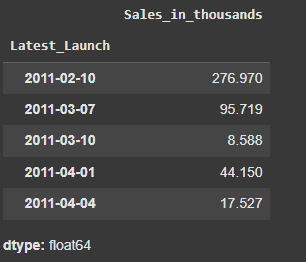
df['Latest\_Launch'] = pd.to\_datetime(df['Latest\_Launch'], errors='coerce')

df = df.dropna(subset=['Latest\_Launch', 'Sales\_in\_thousands'])

ts = df.groupby('Latest\_Launch')['Sales\_in\_thousands'].sum().sort\_index()

ts.head()

**OUTPUT:**

****

**4)** **Visualize the Time Series**

plt.figure(figsize=(12, 5))

plt.plot(ts, marker='o')

plt.title('Car Sales Over Time')

plt.xlabel('Launch Date')

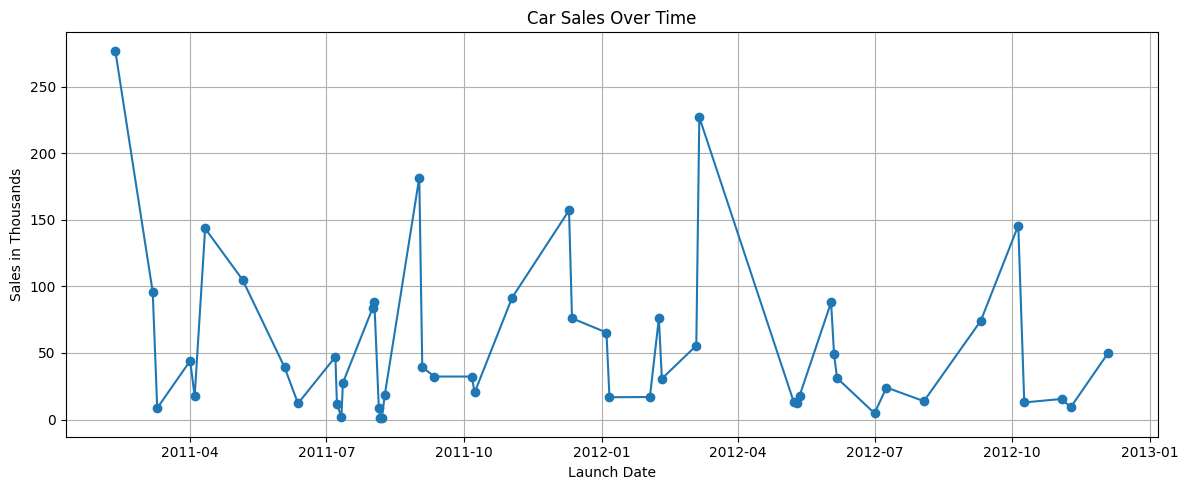
plt.ylabel('Sales in Thousands')

plt.grid(True)

plt.tight\_layout()

plt.show()

**OUTPUT:**

****

**5) Check for Stationarity (ADF Test)**

result = adfuller(ts)

print("ADF Statistic:", result[0])

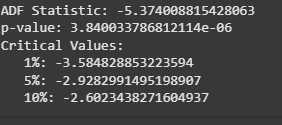
print("p-value:", result[1])

print("Critical Values:")

for key, value in result[4].items():

print(f" {key}: {value}")

**OUTPUT:**

****

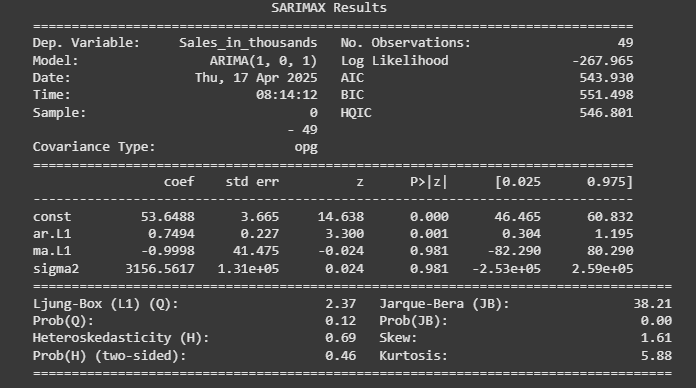
**6) Fit ARIMA Model**

model = ARIMA(ts, order=(1, 0, 1))

model\_fit = model.fit()

print(model\_fit.summary())

**OUTPUT:**

****

### **7) Forecast Future Sales**

# Forecast next 10 time periods

forecast = model\_fit.forecast(steps=10)

# Plot the forecast

plt.figure(figsize=(12, 5))

plt.plot(ts, label='Historical Sales')

plt.plot(forecast.index, forecast, label='Forecasted Sales', color='red', marker='o')

plt.title('ARIMA(1,0,1) Model Forecast')

plt.xlabel('Launch Date')

plt.ylabel('Sales in Thousands')

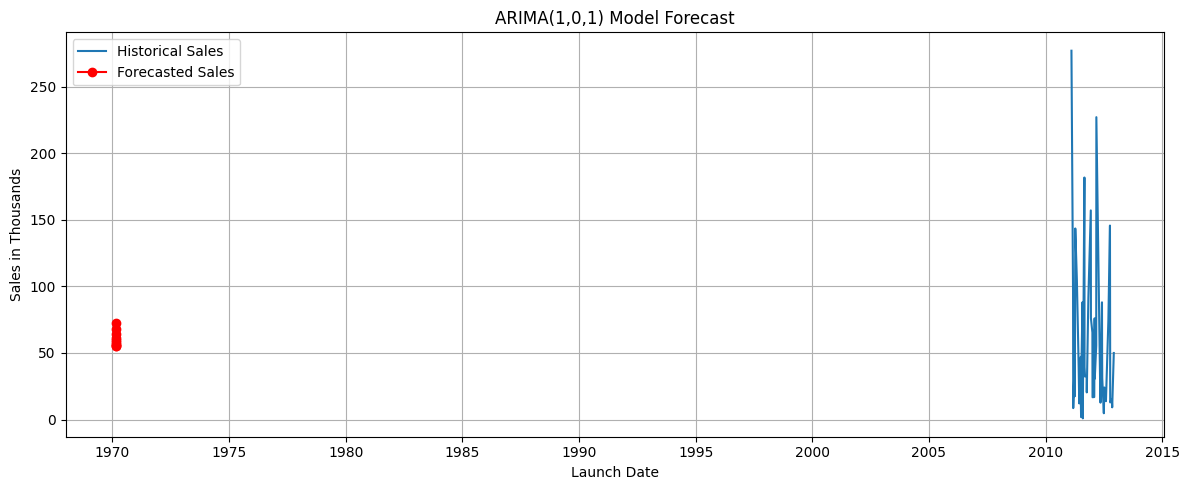
plt.legend()

plt.grid(True)

plt.tight\_layout()

plt.show()

**OUTPUT**

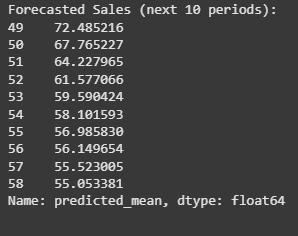
****

**7)Show Forecasted Values**

print("Forecasted Sales (next 10 periods):")

print(forecast)

**OUTPUT:**

****

**RESULT:**

Thus the program has been executed successfully