import numpy as np import pandas as pd

import matplotlib.pyplot as plt

import statsmodels.tsa.stattools as ts

from statsmodels.tsa.arima.model import ARIMA

df = pd.read\_csv("Advertising.csv")

train\_size = int(len(df) \* 0.8)

train, test = df[:train\_size], df[train\_size:]

model = ARIMA(train['Sales'], order=(5,1,0)) model\_fit = model.fit()

print(model\_fit.summary())

SARIMAX Results

==============================================================================

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Dep. Variable: | | Sales | No. Observations: | | 160 | |
| Model: | | ARIMA(5, 1, 0) | Log Likelihood | | -493.502 | |
| Date: | | Thu, 20 Feb 2025 | AIC | | 999.005 | |
| Time: | | 14:02:10 | BIC | | 1017.418 | |
| Sample: | | 0 | HQIC | | 1006.482 | |
| Covariance Type: | | - 160  opg |  | |  | |
| ==============================================================================  coef std err z P>|z| [0.025 0.975] | | | | | | |
| ar.L1 | -0.8784 | 0.088 | -9.948 | 0.000 | -1.051 | -0.705 |
| ar.L2 | -0.8096 | 0.102 | -7.971 | 0.000 | -1.009 | -0.611 |
| ar.L3 | -0.5749 | 0.106 | -5.448 | 0.000 | -0.782 | -0.368 |
| ar.L4 | -0.4542 | 0.094 | -4.823 | 0.000 | -0.639 | -0.270 |
| ar.L5 | -0.2954 | 0.080 | -3.703 | 0.000 | -0.452 | -0.139 |
| sigma2 | 28.8057 | 3.933 | 7.324 | 0.000 | 21.097 | 36.514 |
| ===================================================================================  Ljung-Box (L1) (Q): 0.06 Jarque-Bera (JB): 3.58 | | | | | | |
| Prob(Q): | | | 0.80 | Prob(JB): | | 0.17 |
| Heteroskedasticity (H): | | | 1.22 | Skew: | | 0.26 |
| Prob(H) (two-sided): | | | 0.47 | Kurtosis: | | 2.48 |

===================================================================================

Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).

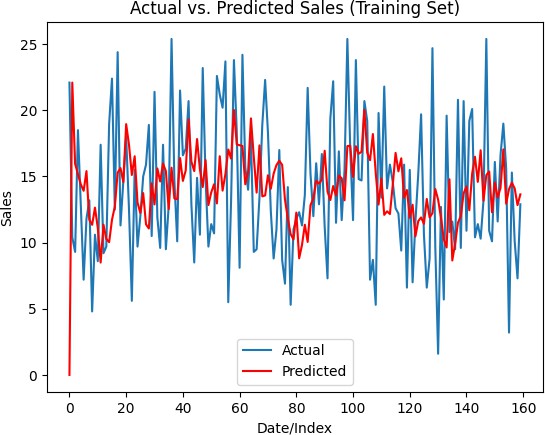
plt.plot(train.index, train['Sales'], label='Actual')

plt.plot(train.index, model\_fit.fittedvalues, color='red', label='Predicted') plt.title('Actual vs. Predicted Sales (Training Set)')

plt.xlabel('Date/Index') plt.ylabel('Sales')

plt.legend() plt.show()





C C

plt plot(test index test['Sales'] label='Actual')

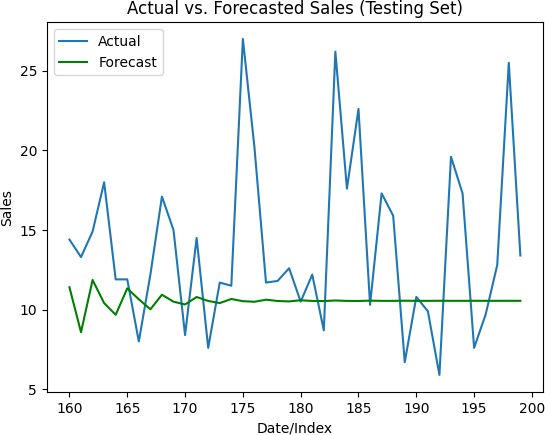
forecast = model\_fit.forecast(steps=len(test))

plt.plot(test.index, forecast, color='green', label='Forecast') plt.title('Actual vs. Forecasted Sales (Testing Set)')

plt.xlabel('Date/Index') plt.ylabel('Sales')

plt.legend() plt.show()





C C

mse = ((forecast - test['Sales']) \*\* 2).mean() rmse = mse \*\* 0.5

print("Mean Squared Error:-", mse)

print("Root Mean Squared Error:-", rmse)

 Mean Squared Error:- 37.60876105920678

Root Mean Squared Error:- 6.132598230701794