EX:No.8

Create an ARIMA model for time series forecasting.

Aim:

Write a program to create an ARIMA model for time series forecasting.

Algorithm:

1. **Import necessary libraries**:
Import numpy, pandas, matplotlib.pyplot, ARIMA from statsmodels.tsa.arima.model, and mean_squared_error from sklearn.metrics.

2. **Load the dataset**:
Read the art market dataset using pandas.read_csv() and parse the 'Date' column as datetime. Set the 'Date' column as the index.

3. **Select the target column**:
Extract the 'Price' column from the DataFrame for time series forecasting.

4. **Split the data**:
Calculate the training size as 80% of the total data.
Split the data into training and testing sets using this calculated size.

5. **Define and fit the ARIMA model**:
Initialize the ARIMA model with training data and set parameters (p=5, d=1, q=0).
Fit the model using .fit().

6. **Forecast future values**:
Forecast the price values for the length of the test set using .forecast().

7.**Visualize the training data**:
Create a line plot for the training data using matplotlib.

8.**Visualize the actual test data**:
Create a line plot for the test (actual) data.

9.**Visualize the forecasted vs actual test data**:
Plot the ARIMA forecast along with the actual test values on the same graph to compare performance.

10.**Evaluate the model**:
Calculate the Root Mean Squared Error (RMSE) between the forecast and actual test values using mean_squared_error().

11.**Print the RMSE value**.

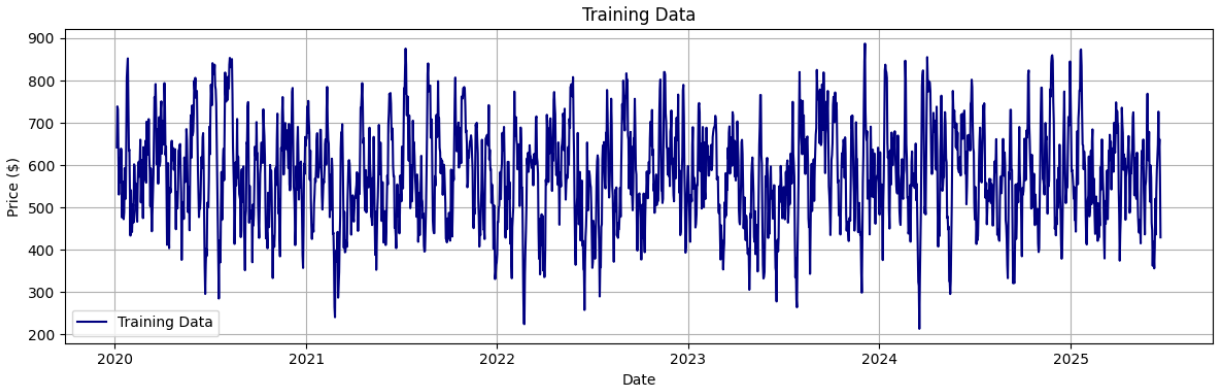
Code:

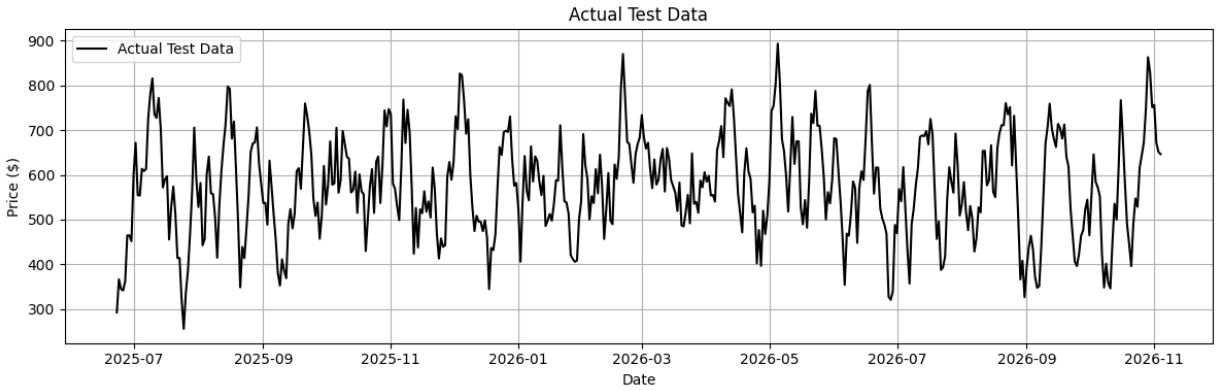
```
import numpy as np

import pandas as pd
```

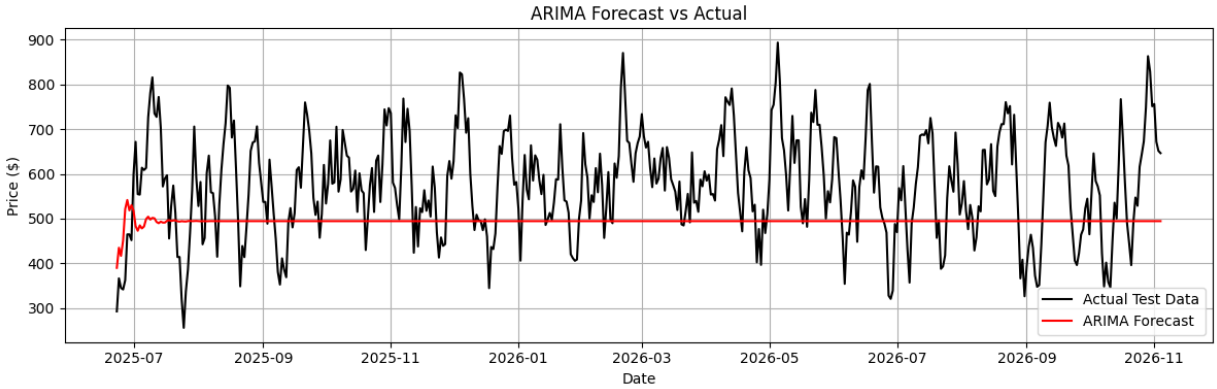
```
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
df = pd.read_csv("artmarket_with_dates.csv")
df['Date'] = pd.to_datetime(df['Date'])
df = df.sort_values('Date')
df.set_index('Date', inplace=True)
ts = df['Price ($)'].resample('D').mean().fillna(method='ffill')
ts_smooth = ts.rolling(window=5).mean().dropna()
split_idx = int(len(ts_smooth) * 0.8)
train = ts_smooth[:split_idx]
test = ts_smooth[split_idx:]
model = ARIMA(train, order=(5, 1, 0))
model_fit = model.fit()
forecast = model_fit.forecast(steps=len(test))
forecast.index = test.index
plt.figure(figsize=(12, 4))
plt.plot(train, label='Training Data', color='navy')
plt.title('Training Data')
plt.xlabel('Date')
plt.ylabel('Price ($)')
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()
plt.figure(figsize=(12, 4))
plt.plot(test, label='Actual Test Data', color='black')
plt.title('Actual Test Data')
plt.xlabel('Date')
plt.ylabel('Price ($)')
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()
```

```
plt.figure(figsize=(12, 4))
plt.plot(test, label='Actual Test Data', color='black')
plt.plot(forecast, label='ARIMA Forecast', color='red')
plt.title('ARIMA Forecast vs Actual')
plt.xlabel('Date')
plt.ylabel('Price ($)')
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()
```

Output:







Result:

Thus, the program to create an ARIMA model for time series forecasting was created successfully.