DATA ANALYST

Intenship Task 16

DESCRIPTION

This task focuses on performing time series forecasting using the Superstore dataset. The objective is to analyze historical sales data, identify trends and patterns, apply a forecasting model, evaluate prediction accuracy using error metrics, and generate future sales estimates to support strategic business planning and decision-making.

PREPARED BY

Reema Safrin M
(13-02-2026)

MY WORK

In this project, I used the cleaned Superstore dataset to perform time series forecasting on monthly sales data. I converted the Order_Date column into datetime format and aggregated total sales by month to create a proper time series structure. I visualized trends and checked seasonality using rolling averages. The dataset was split into training and testing sets to validate the model. I applied the Exponential Smoothing forecasting technique to predict future sales and evaluated model performance using MAE and MAPE metrics. The analysis revealed a steady growth trend with fluctuations, providing useful insights for inventory and financial planning

DATASET USED

superstore dataset

FINAL DATASET

forecast_output

```python
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.holtwinters import ExponentialSmoothing
from sklearn.metrics import mean_absolute_error, mean_absolute_percentage_error
```

```python
df = pd.read_csv("sample_superstore_clean(superstore dataset).csv")
df.head()
```

	Row_ID	Order_ID	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment	C
0	1	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	
1	2	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	
2	3	CA-2016-138688	6/12/2016	6/16/2016	Second Class	DV-13045	Darrin Van Huff	Corporate	
3	4	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	
4	5	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	

5 rows × 21 columns

```python
df.columns
```

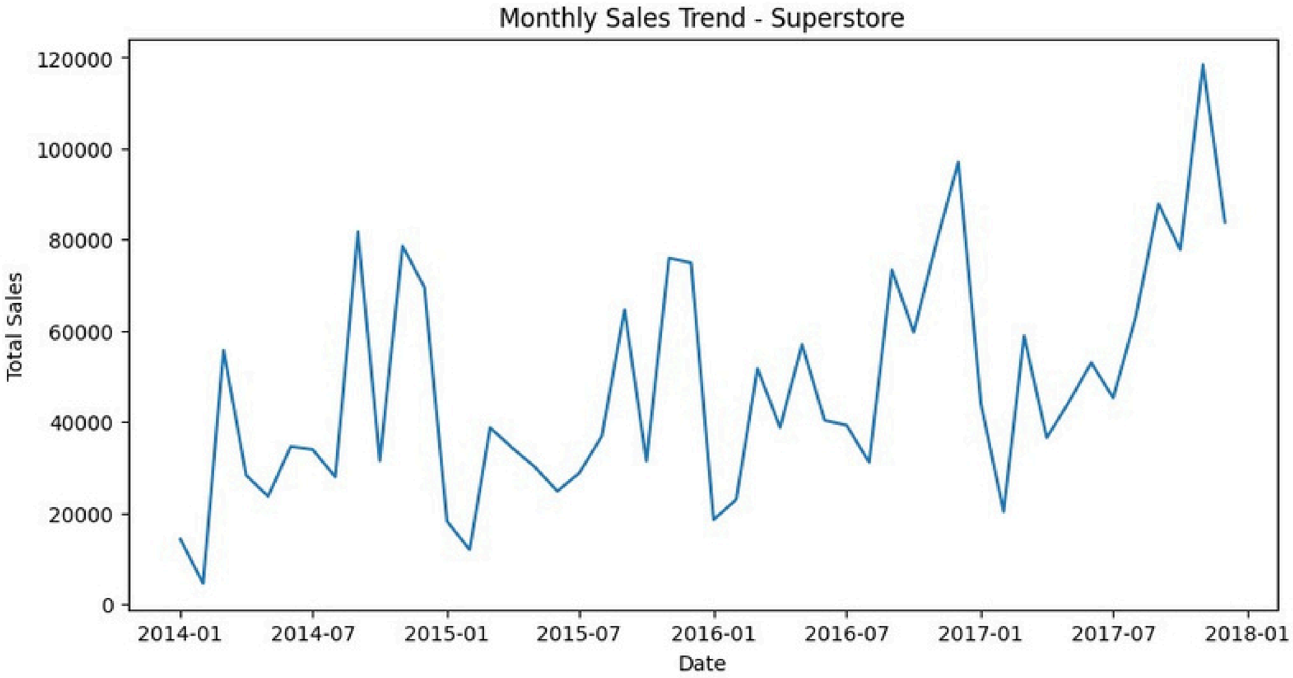
```
Index(['Row_ID', 'Order_ID', 'Order_Date', 'Ship_Date', 'Ship_Mode',
       'Customer_ID', 'Customer_Name', 'Segment', 'Country', 'City', 'State',
       'Postal_Code', 'Region', 'Product_ID', 'Category', 'Sub_Category',
       'Product_Name', 'Sales', 'Quantity', 'Discount', 'Profit'],
      dtype='object')
```

```python
df['Order_Date'] = pd.to_datetime(df['Order_Date'])
```

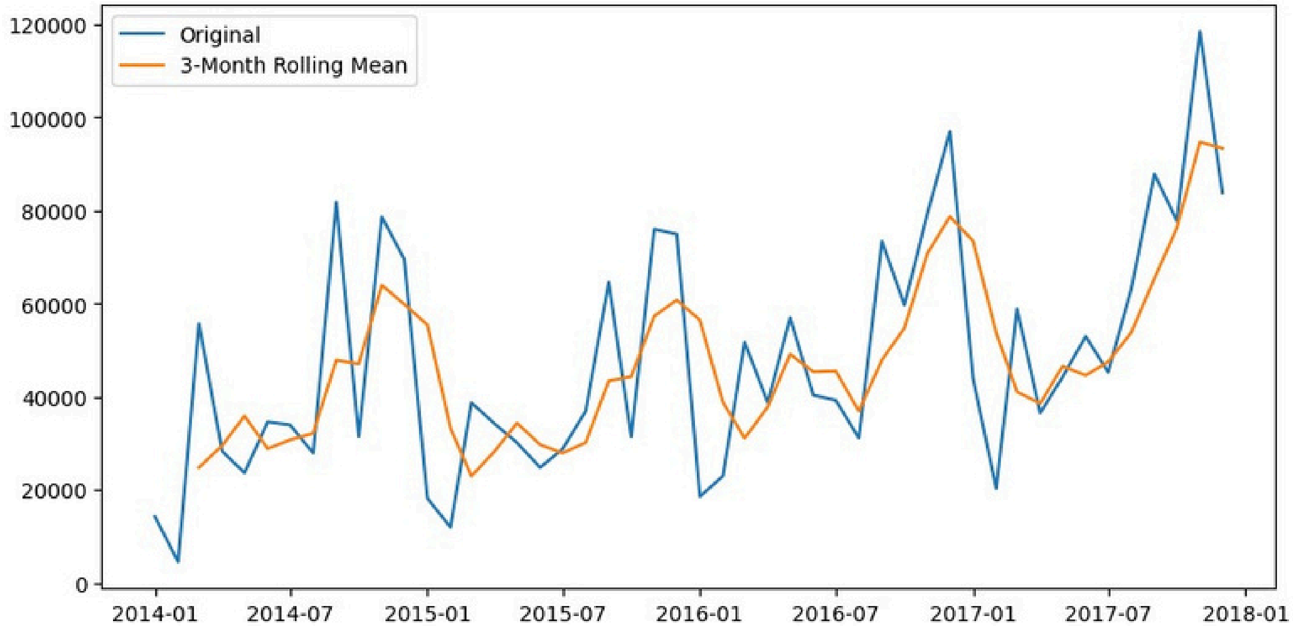
```python
monthly_sales = df.groupby(df['Order_Date'].dt.to_period('M'))['Sales'].sum()
monthly_sales = monthly_sales.to_timestamp()
```

```python
plt.figure(figsize=(10,5))
plt.plot(monthly_sales)
```

```
plt.title("Monthly Sales Trend - Superstore")
plt.xlabel("Date")
plt.ylabel("Total Sales")
plt.show()
```



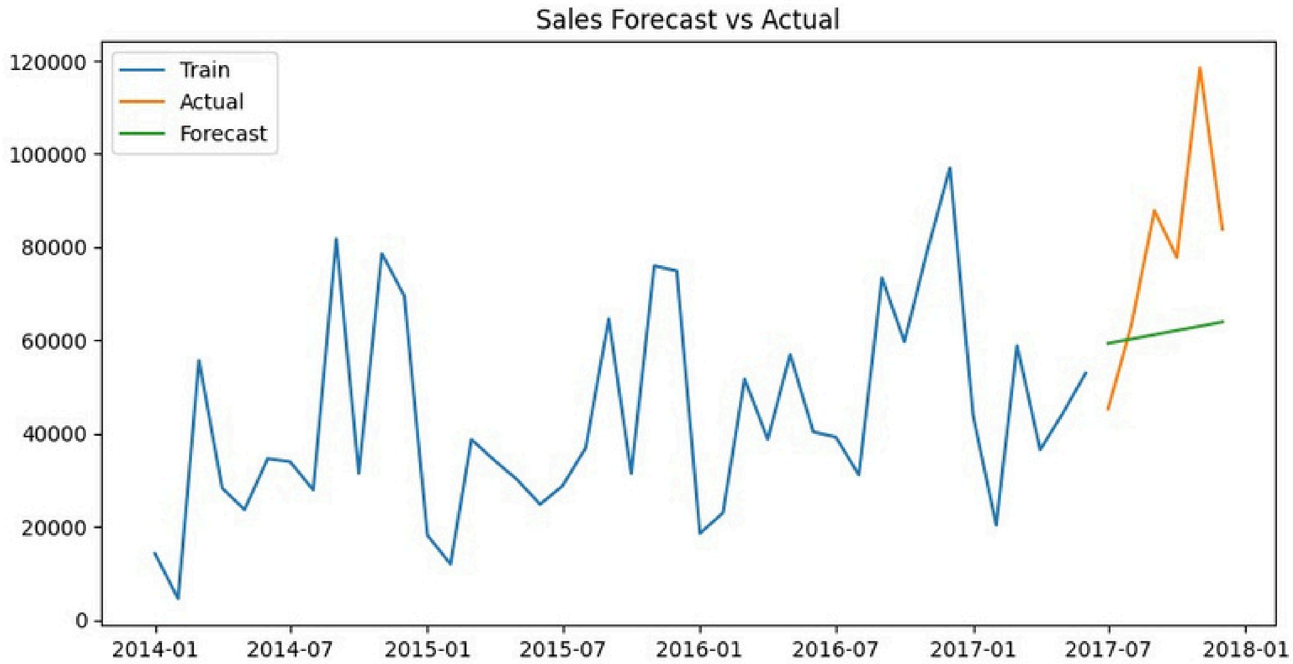

```
rolling_mean = monthly_sales.rolling(window=3).mean()

plt.figure(figsize=(10,5))
plt.plot(monthly_sales, label='Original')
plt.plot(rolling_mean, label='3-Month Rolling Mean')
plt.legend()
plt.show()
```



```
train = monthly_sales[:-6]
test = monthly_sales[-6:]
```

```
model = ExponentialSmoothing(train, trend='add', seasonal=None)
fit = model.fit()
forecast = fit.forecast(6)
```

```
plt.figure(figsize=(10,5))
plt.plot(train,         label='Train')
plt.plot(test,         label='Actual')
plt.plot(forecast,    label='Forecast')
plt.legend()          plt.title("Sales
Forecast vs Actual") plt.show()
```




```
mae = mean_absolute_error(test, forecast)
mape = mean_absolute_percentage_error(test, forecast)

print("MAE:", mae)
print("MAPE:", mape)
```

```
MAE: 22429.722861274047
MAPE: 0.2610726940707732
```

```
forecast_df = pd.DataFrame({
    "Date": forecast.index,
    "Forecasted Sales": forecast.values
})

forecast_df.to_csv("forecast_output.csv", index=False)
```