

COVER PAGE:

DATA SCIENCE TOOLBOX: PYTHON PROGRAMMINGPROJECT

(Project Semester January-April 2025)

(Data Analysis and Visualization of Real-World Dataset using Python)

Submitted by

SMP KIRAN SANDEEP

12320836

Data Science – K23FA

INT375

Under the Guidance of

Sandeep Kaur-(23614)

Discipline of CSE/IT

Lovely School of Computer Science

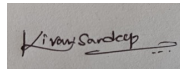
Lovely Professional University, Phagwara

DECLARATION

I, SMP Kiran Sandeep, student of DATA SCIENCE TOOLBOX: PYTHON PROGRAMMINGPROJECT under CSE/IT Discipline at, Lovely Professional University, Punjab, hereby declare that all the information furnished in this project report is based on my own intensive work and is genuine.

Date: 16/04/25

Signature:

A small rectangular box containing a handwritten signature in black ink. The signature appears to read 'Kiran Sandeep' with a stylized flourish at the end.

Registration No: 12320836

Objective 1: Sales by Payment Mode

Goal: Load and clean the dataset to prepare it for analysis.

```
jupyter INT375 Project Last Checkpoint: 26 minutes ago Trusted
File Edit View Run Kernel Settings Help
+ - - - - - Code
JupyterLab Python 3 (ipykernel)

[31]: import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

[32]: df = pd.read_csv('/Users/kiransandeep/Downloads/python project/dataset.csv')
df['Date'] = pd.to_datetime(df['Date'], errors='coerce', dayfirst=True)
df['Month_Year'] = df['Date'].dt.to_period('M')
for col in ['Net_sales', 'Taxes']:
    df[col] = df[col].str.replace("?", "").str.replace(", ", "").astype(float)

[33]: # 1. Pie Chart: Sales by Payment Mode
plt.figure(figsize=(6, 6))
df.groupby('Payment_Mode')['Total_sales'].sum().plot.pie(autopct='%1.1f%%', startangle=140)
plt.title("Total Sales by Payment Mode")
plt.ylabel("")
plt.tight_layout()
plt.show()
```

```
import pandas as pd
```

```
import matplotlib.pyplot as plt
```

```
import seaborn as sns
```

```
import numpy as np
```

```
df = pd.read_csv('/Users/kiransandeep/Downloads/python project/dataset.csv')
```

```
df['Date'] = pd.to_datetime(df['Date'], errors='coerce', dayfirst=True)
```

```
df['Month_Year'] = df['Date'].dt.to_period('M')
```

```
for col in ['Net_sales', 'Taxes']:
```

```
    df[col] = df[col].str.replace("?", "").str.replace(", ", "").astype(float)
```

```
# 1. Pie Chart: Sales by Payment Mode
```

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

```
df.groupby('Payment_Mode')['Total_sales'].sum().plot.pie(autopct='%1.1f%%', startangle=140)
```

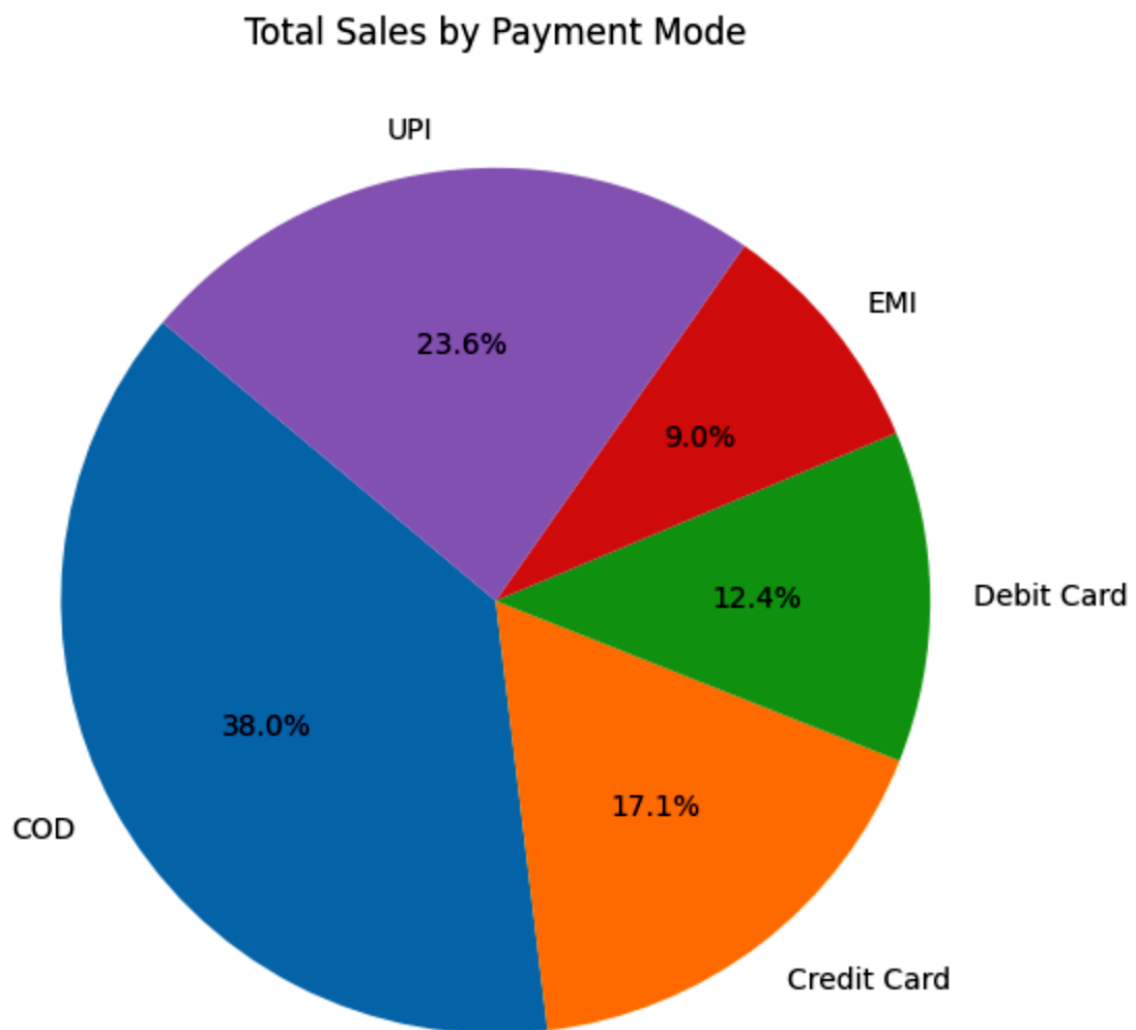
```
plt.title("Total Sales by Payment Mode")
```

```
plt.ylabel("")
```

```
plt.tight_layout()
```

```
plt.show()
```

Output:



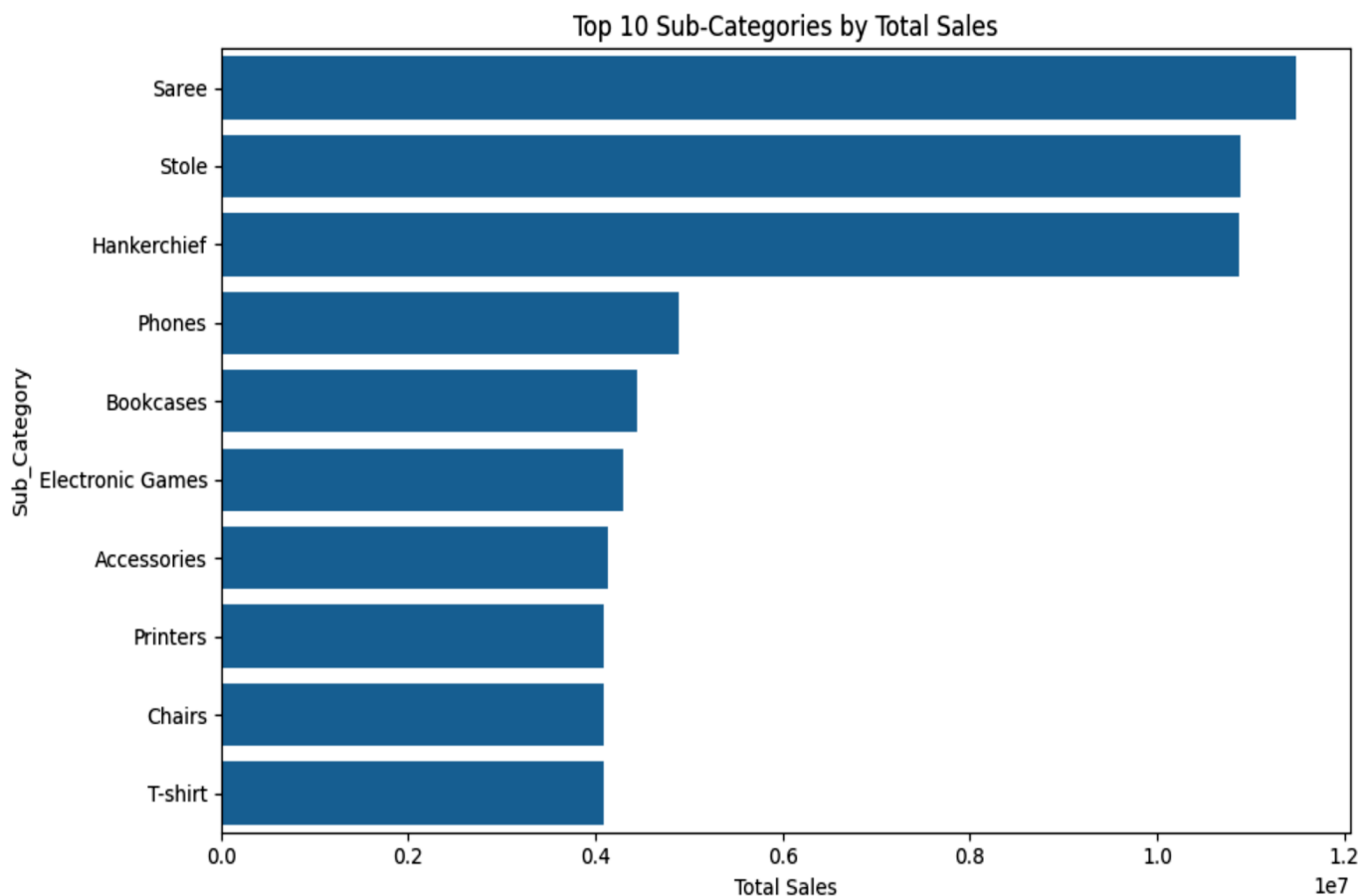
Objective 2: Top 10 Sub-Categories by Total Sales

```
# 2. Horizontal Bar: Top 10 Sub-Categories by Total Sales
top_subcats = df.groupby('Sub_Category')['Total_sales'].sum().sort_values(ascending=False).head(10)
plt.figure(figsize=(10, 6))
sns.barplot(x=top_subcats.values, y=top_subcats.index)
plt.title("Top 10 Sub-Categories by Total Sales")
plt.xlabel("Total Sales")
plt.tight_layout()
plt.show()
```

2. Horizontal Bar: Top 10 Sub-Categories by Total Sales

```
top_subcats =  
df.groupby('Sub_Category')['Total_sales'].sum().sort_values(ascending=False).  
head(10)  
plt.figure(figsize=(10, 6))  
sns.barplot(x=top_subcats.values, y=top_subcats.index)  
plt.title("Top 10 Sub-Categories by Total Sales")  
plt.xlabel("Total Sales")  
plt.tight_layout()  
plt.show()
```

Output:



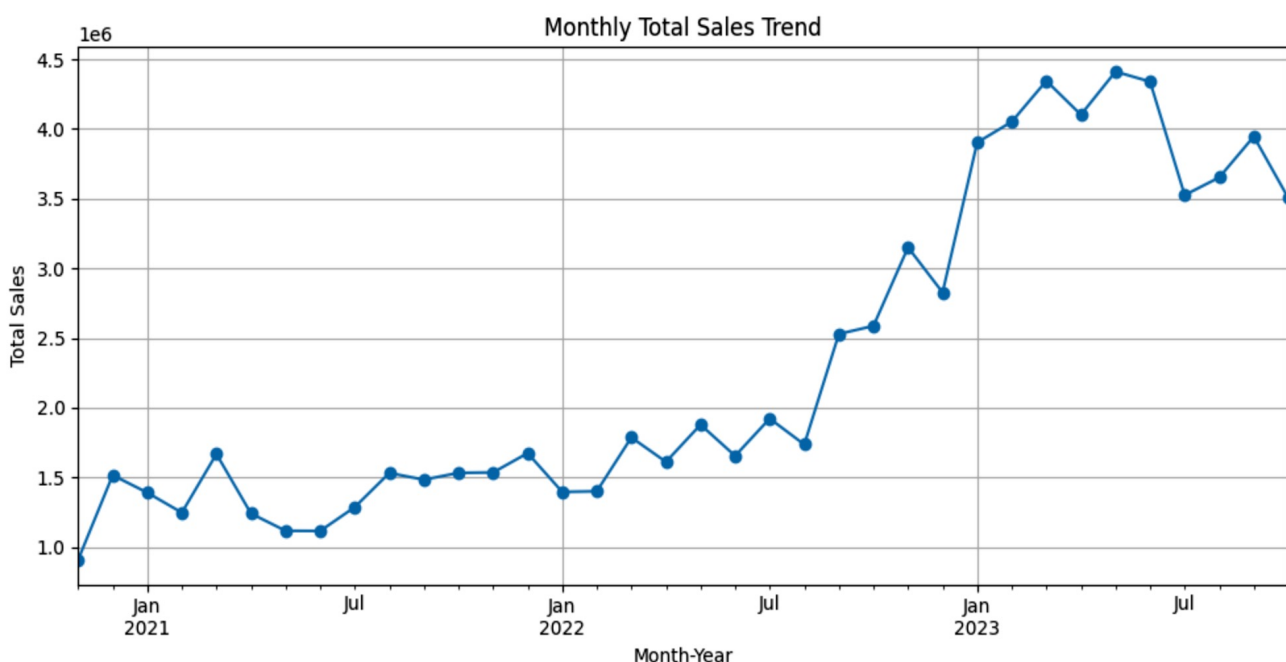
Objective 3: Monthly Sales Trend

```
# 3. Line Plot: Monthly Sales Trend
monthly_sales = df.groupby('Month_Year')['Total_sales'].sum()
plt.figure(figsize=(10, 5))
monthly_sales.plot(marker='o')
plt.title("Monthly Total Sales Trend")
plt.xlabel("Month-Year")
plt.ylabel("Total Sales")
plt.grid(True)
plt.tight_layout()
plt.show()
```

3. Line Plot: Monthly Sales Trend

```
monthly_sales = df.groupby('Month_Year')['Total_sales'].sum()
plt.figure(figsize=(10, 5))
monthly_sales.plot(marker='o')
plt.title("Monthly Total Sales Trend")
plt.xlabel("Month-Year")
plt.ylabel("Total Sales")
plt.grid(True)
plt.tight_layout()
plt.show()
```

Output:



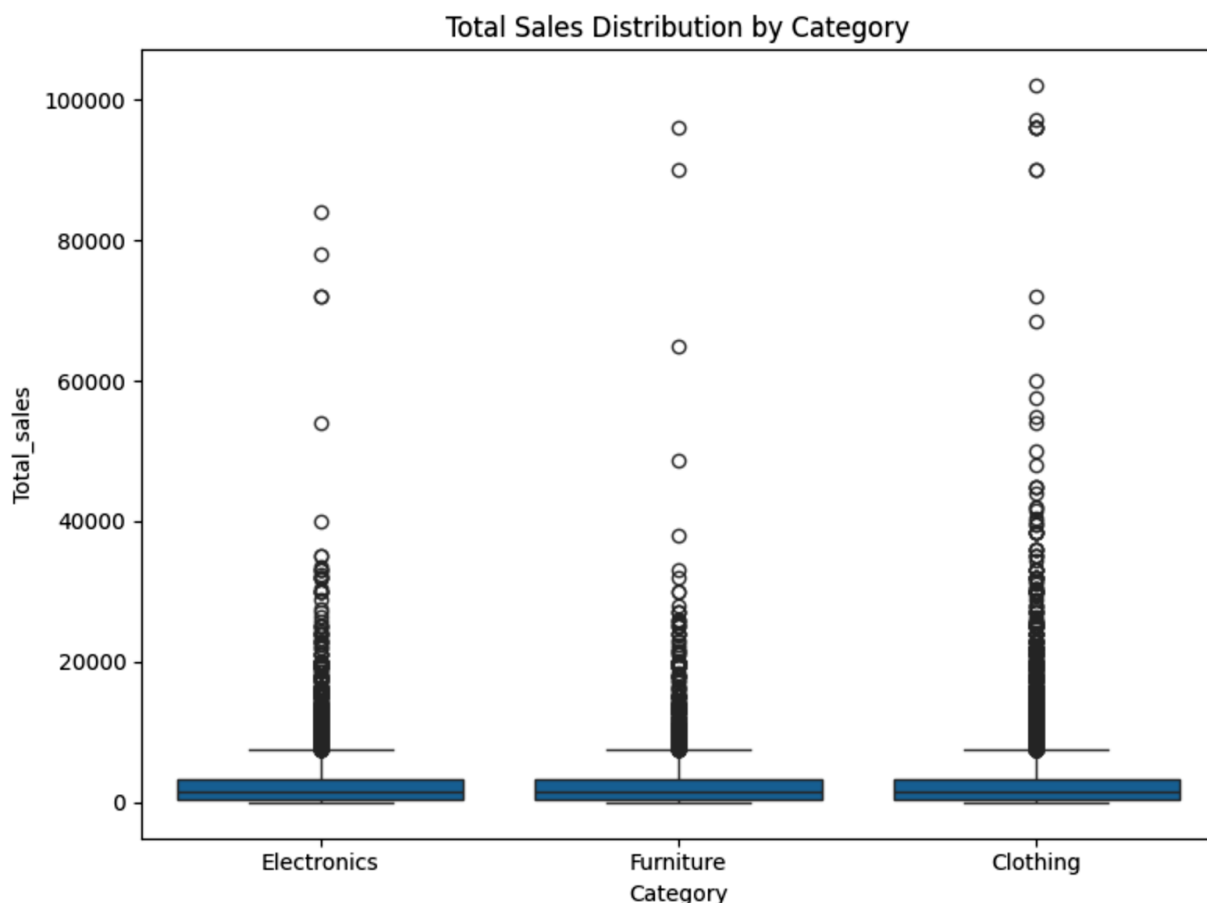
Objective 4: Total Sales per Category

```
# 4. Box Plot: Total Sales per Category
plt.figure(figsize=(8, 6))
sns.boxplot(data=df, x='Category', y='Total_sales')
plt.title("Total Sales Distribution by Category")
plt.tight_layout()
plt.show()
```

4. Box Plot: Total Sales per Category

```
plt.figure(figsize=(8, 6))
sns.boxplot(data=df, x='Category', y='Total_sales')
plt.title("Total Sales Distribution by Category")
plt.tight_layout()
plt.show()
```

Output:



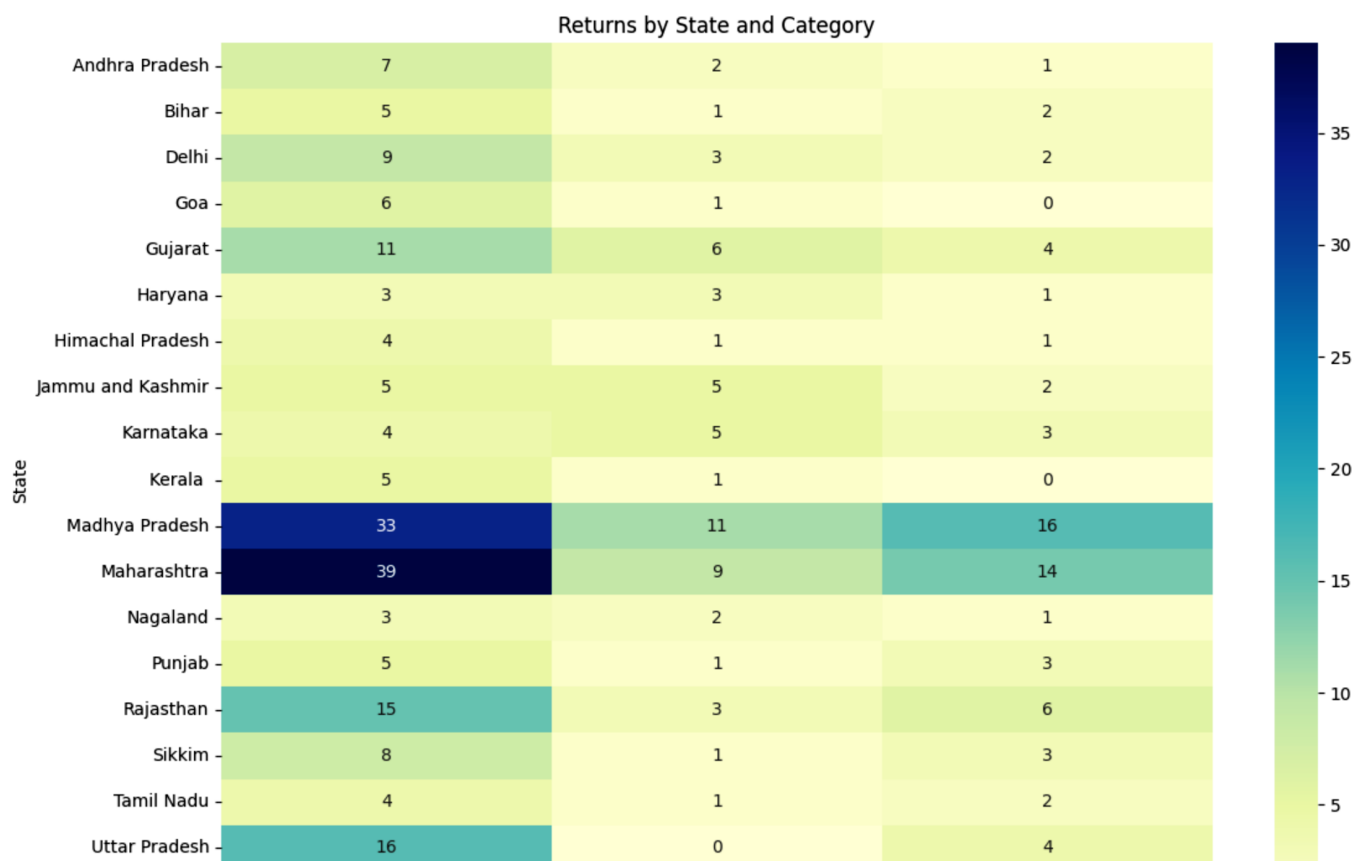
Objective 5: Returns by State and Category

```
# 5. Heatmap: Returns by State and Category
heatmap_data = pd.pivot_table(df, values='Returns', index='State', columns='Category', aggfunc='sum', fill_value=0)
plt.figure(figsize=(12, 8))
sns.heatmap(heatmap_data, annot=True, fmt="d", cmap="YlGnBu")
plt.title("Returns by State and Category")
plt.tight_layout()
plt.show()
```

5. Heatmap: Returns by State and Category

```
heatmap_data = pd.pivot_table(df, values='Returns', index='State',
columns='Category', aggfunc='sum', fill_value=0)
plt.figure(figsize=(12, 8))
sns.heatmap(heatmap_data, annot=True, fmt="d", cmap="YlGnBu")
plt.title("Returns by State and Category")
plt.tight_layout()
plt.show()
```

Output:



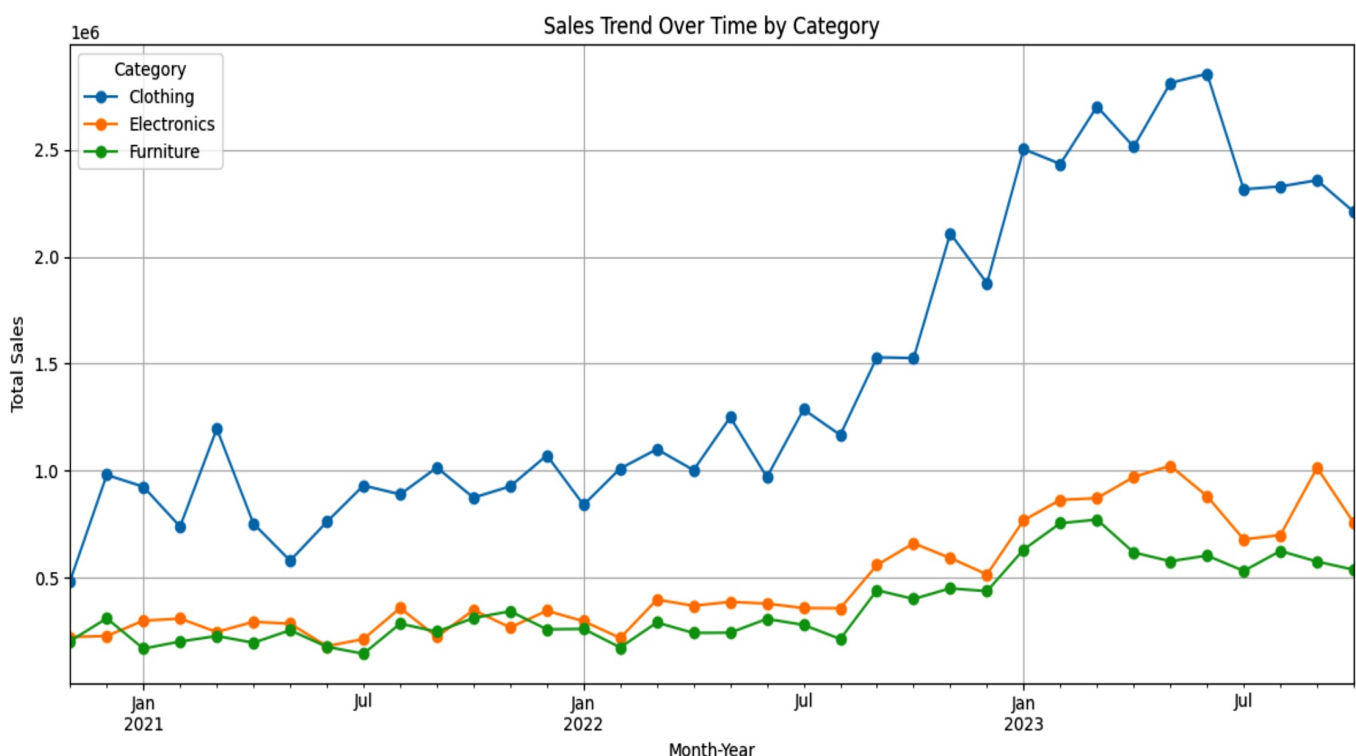
Objective 6: Sales Trend by Category Over Time

```
# 6. Multi-Line Chart : Sales Trend by Category Over Time
category_time_sales = df.groupby(['Month_Year', 'Category'])['Total_sales'].sum().unstack()
category_time_sales.plot(figsize=(12, 6), marker='o')
plt.title("Sales Trend Over Time by Category")
plt.xlabel("Month-Year")
plt.ylabel("Total Sales")
plt.grid(True)
plt.tight_layout()
plt.show()
```

6. Multi-Line Chart : Sales Trend by Category Over Time

```
category_time_sales = df.groupby(['Month_Year',
'Category'])['Total_sales'].sum().unstack()
category_time_sales.plot(figsize=(12, 6), marker='o')
plt.title("Sales Trend Over Time by Category")
plt.xlabel("Month-Year")
plt.ylabel("Total Sales")
plt.grid(True)
plt.tight_layout()
plt.show()
```

Output:



Conclusion :

By analyzing the “E-commerce Order Line-Item Transactions (Nov 2020 – Oct 2023)” dataset and visualizing key metrics, you’ll uncover how the business has grown and shifted over a three-year period. You may observe:

Seasonal & Growth Trends: Clear peaks around major sale events (Diwali, year-end festivals) and an overall upward trajectory in net sales.

Category Dynamics: Electronics and Furniture driving the bulk of revenue, while fast-fashion segments show higher return rates.

Payment & Fulfilment Insights: COD remains popular but digital payments steadily gain share; expedited shipping correlates with higher average order values.

Regional Variations: Some states (e.g., Maharashtra, Delhi) consistently outperform in both volume and average order size, suggesting opportunities for localized marketing.

Discount & Return Impacts: Heavy discounting lifts volume but compresses margins; return hotspots highlight products or categories needing better quality controls or clearer sizing information.

Future Scope :

1. Predictive Modeling:

Build time-series forecasts (ARIMA, Prophet, LSTM) to predict short-term and long-term sales, enabling proactive inventory planning.

Develop classification models to predict which orders are most at risk of return.

2. Customer Segmentation & Lifetime Value (CLV):

Cluster customers based on purchase behavior (RFM analysis) and tailor personalized marketing campaigns.

Calculate CLV per segment to guide acquisition vs. retention spend.

3. Price Elasticity & Promotion Analysis:

Quantify how discount depth affects purchase volume and profitability by category.

A/B test different promotion structures and measure uplift.

4. Supply-Chain & Delivery Optimization:

Incorporate logistics data (warehouse locations, transit times) to model and minimize delivery costs and delays.

Simulate different fulfillment strategies (centralized vs. distributed) for cost/risk trade-offs.

5. Real-Time Dashboard & Alerts:

Upgrade the static analysis to a real-time streaming dashboard (using Kafka + Spark + Plotly Dash) to monitor KPI deviations (e.g., sudden spike in returns).

Set automated alerts for anomalous patterns (e.g., fraud detection signals, stockouts).

6. Cross-Channel Attribution & Marketing ROI:

Integrate marketing touchpoint data (ads, emails, social media) to model the customer journey and attribute conversions.

Optimize marketing spend by channel using multi-touch attribution models.

Reference :

https://www.data.gov.in/resource/year-wise-total-number-digital-payment-transactions-undertaken-2018-19-2021-22?utm_source=chatgpt.com#:~:text=Feedback-,Download,-Preview