4/09/2025

Retail Sales Data Analysis Project

Rahul Savarmal Kumawat
PLASMID

ACKNOWLEDGMENT

I would like to express my sincere gratitude to Plasmid for providing me with the opportunity to undergo online training on Retail Data Analysis. This training has been an invaluable learning experience, helping me enhance my technical skills in data preprocessing, exploratory data analysis, and predictive analytics using Python, SQL, and visualization tools.

I am especially thankful to my mentors and instructors at Plasmid, whose guidance and expertise have been instrumental in shaping my understanding of real-world data science applications. Their insights and constructive feedback have allowed me to refine my analytical approach and develop a structured methodology for tackling business problems through data-driven solutions.

Additionally, I would like to extend my gratitude to my peers and colleagues, who have provided continuous support and motivation throughout this journey. The collaborative environment and discussions during the training have greatly enriched my learning experience.

Lastly, I appreciate my family and friends for their encouragement and belief in my abilities, which has helped me stay focused and motivated throughout the training period.

This report is a culmination of the knowledge and skills I have gained during my training, and I hope it reflects my dedication and enthusiasm for the field of Data Science and Retail Analytics.

Rahul Savarmal Kumawat
4/09/2025

ABSTRACT

The retail industry generates vast amounts of transactional data daily, making data analysis crucial for understanding customer behavior, optimizing inventory, and improving sales strategies. This project focuses on Retail Data Analysis using a dataset provided by Plasmid as part of an online training program. The objective of this study is to explore, clean, and analyze retail sales data to extract meaningful insights and patterns.

The project follows a structured data science workflow, including data preprocessing, exploratory data analysis (EDA), feature engineering, and visualization techniques. The dataset consists of customer purchase records, including product details, sales transactions, and timestamps, which are analyzed globally, country-wise, and month-wise to determine the most popular items and purchasing trends.

Various statistical and machine learning techniques are applied to uncover hidden patterns, detect anomalies, and segment customers based on their purchasing behavior. Advanced visualization techniques using Python (Matplotlib, Seaborn), SQL, and Power BI are used to present the findings in an interactive and intuitive manner.

The results of this analysis provide actionable insights for retailers, helping them make data-driven decisions regarding product demand, inventory management, and customer preferences. The study demonstrates how effective data analysis can drive business growth and optimize operational strategies in the retail sector.

Keywords:

Retail Analytics, Data Science, Machine Learning, Customer Segmentation, Sales Trends, Data Visualization

TABLE OF CONTENTS

1. Introduction
    - Overview of Retail Data Analysis
    - Objectives of the Project
    - Importance of Data Analytics in Retail
2. Dataset Description
    - Source of Data
    - Features and Attributes of the Dataset
    - Understanding the Data Variables
3. Data Preprocessing
    - Handling Missing Values
    - Removing Duplicates
    - Handling Outliers
    - Data Type Conversion
    - Feature Engineering
4. Exploratory Data Analysis (EDA)
    - Statistical Summary of the Data
    - Data Distribution and Trends
    - Identifying Key Performance Indicators
5. Popular Item Analysis
    - Global Best-Selling Products
    - Country-wise Best-Selling Products
    - Month-wise Popular Products
6. Customer Segmentation
    - Identifying Customer Buying Patterns
    - RFM (Recency, Frequency, Monetary) Analysis

    - Clustering Techniques for Customer Segmentation
7. Sales Trend Analysis
    - Seasonal Trends and Monthly Sales Variations
    - Yearly Sales Growth Analysis
    - Time Series Forecasting (if applicable)
8. Data Visualization and Reporting

- Graphical Representation of Insights
- Dashboard Creation using Power BI / Tableau
- Interactive Plots using Python (Matplotlib & Seaborn)

9 Challenges and Limitations

- Issues Faced in Data Cleaning
- Model Limitations and Assumptions
- Future Scope for Improvement

10 Conclusion and Recommendations

- Key Takeaways from the Analysis
- Business Recommendations Based on Insights
- Future Enhancements and Use Cases

11 References

12 Appendix

1. Introduction

Overview of Retail Data Analysis

Retail data analysis involves extracting valuable insights from sales, customer behaviour, and inventory data to enhance business decision-making. In the digital era, businesses rely on data-driven strategies to optimize sales, reduce costs, and improve customer satisfaction. Retail analytics helps organizations understand purchasing patterns, forecast demand, and manage inventory efficiently.

This project focuses on analysing a real-world retail dataset, identifying trends, and uncovering patterns that can improve business operations and customer engagement.

Objectives of the Project

The primary objectives of this project are:

1. Data Cleaning and Preprocessing: Handling missing values, duplicate records, and ensuring data consistency.
2. Exploratory Data Analysis (EDA): Understanding sales trends, customer purchase behaviour, and product popularity.
3. Sales and Demand Forecasting: Predicting future sales trends based on historical data.
4. Customer Segmentation: Identifying customer groups based on purchasing habits using clustering techniques.
5. Country-wise and Monthly Analysis: Evaluating sales performance globally and across different countries.
6. Product Popularity Analysis: Identifying top-selling products based on sales volume and revenue.
7. Generating Actionable Insights: Providing recommendations to improve business performance based on data analysis.

Importance of Data Analytics in Retail

Data analytics plays a crucial role in the retail industry by enabling businesses to:

- Enhance Customer Experience: Personalized marketing based on customer preferences.
- Optimize Inventory Management: Ensuring stock availability while minimizing excess inventory.
- Increase Sales and Revenue: Identifying high-demand products and peak sales periods.
- Improve Decision-Making: Using data-driven insights for better strategic planning.
- Detect Fraud and Anomalies: Identifying suspicious transactions or fraudulent activities.

2. Dataset Description

Source of Data

The dataset used in this project is sourced from an Online Retail transactional database. It contains records of customer purchases made from a UK-based online store between December 2010 and December 2011. The dataset provides details of each transaction, including product information, quantity purchased, invoice details, customer ID, and country of purchase.

This dataset is ideal for retail analytics as it allows us to analyze customer purchasing behavior, product demand, and global sales trends.

Features and Attributes of the Dataset

The dataset consists of multiple columns, each representing a different aspect of a retail transaction. Below are the key features and their descriptions:

Feature Name	Description
InvoiceNo	Unique invoice number for each transaction
StockCode	Unique product/item identifier
Description	Name of the product/item
Quantity	Number of units purchased in the transaction
InvoiceDate	Date and time when the purchase was made
UnitPrice	Price per unit of the product (in GBP)
CustomerID	Unique identifier for each customer
Country	Country where the customer is located

Understanding the Data Variables

1. InvoiceNo (Categorical - String):
    - Each invoice represents a unique transaction.
    - If an invoice number starts with 'C', it indicates a canceled order.

2. StockCode (Categorical - String):
    - This is a unique identifier for each product.
    - Some stock codes represent special charges like shipping fees.
3. Description (Text - String):
    - Provides details about the product name.
    - Missing values in this column need to be handled appropriately.
4. Quantity (Numerical - Integer):
    - Represents the number of units sold.
    - Negative values indicate order returns.
5. InvoiceDate (DateTime - Timestamp):
    - Records the exact date and time of the transaction.
    - Useful for trend analysis and time-series forecasting.
6. UnitPrice (Numerical - Float):
    - Denotes the price of a single product unit.
    - Needs to be checked for unrealistic values (e.g., negative prices).
7. CustomerID (Categorical - Integer):
    - Unique identifier for each customer.
    - Missing values suggest transactions by anonymous customers.
8. Country (Categorical - String):
    - Indicates where the customer is located.
    - Helps in country-wise sales analysis.

3. Data Preprocessing

Data preprocessing is a crucial step in any data science project as it ensures the dataset is clean, structured, and ready for analysis. In this section, we outline the key preprocessing steps applied to the **Online Retail Dataset** before conducting analysis.

1. Handling Missing Values

Missing values can cause inconsistencies in the analysis. In the given dataset, missing values were primarily found in the **"Description"** and **"CustomerID"** columns.

- **"Description" Column**: Missing product descriptions were handled by either removing the corresponding rows (if the stock code was invalid) or replacing them using reference stock codes.
- **"CustomerID" Column**: Missing customer IDs were treated as anonymous purchases. Depending on the analysis requirements, either these rows were removed or handled separately for segmentation analysis.

2. Removing Duplicates

Duplicate records can distort results and introduce redundancy in the analysis. The dataset was checked for **duplicate rows** based on key attributes such as **InvoiceNo, StockCode, Quantity, InvoiceDate, and CustomerID**.

- Any exact duplicate rows were removed to ensure data integrity.
- Partial duplicates (e.g., same invoice but different timestamps) were checked to confirm whether they were valid transactions.
-

3. Handling Outliers

Outliers can significantly affect the accuracy of models and insights. The dataset was examined for **unusual values in numerical columns** like **Quantity and UnitPrice**.

- **Negative Quantity Values**: These indicate product **returns**. Instead of removing them, they were analyzed separately to understand return patterns.

- **Abnormally High Quantities**: Orders with extremely high quantities (e.g., 10,000+ units in a single transaction) were verified as potential bulk purchases or data entry errors.
- **Negative Prices**: Transactions with negative unit prices were flagged for further review, as they might indicate **credit notes or discounts**.

4. Data Type Conversion

Ensuring correct data types is essential for efficient processing. The following transformations were applied:

- **InvoiceDate** was converted from text to a **datetime format** to allow for time-series analysis.
- **CustomerID** and **StockCode** were converted into categorical variables, as they do not have numerical significance.
- **UnitPrice and Quantity** were kept as numerical values to facilitate calculations and statistical analysis.

5. Feature Engineering

Feature engineering involves creating new meaningful variables to improve insights and model performance. Key features created:

- **TotalPrice**: A new column was created by multiplying **Quantity × UnitPrice**, representing the total value of each transaction.
- **Month and Year**: Extracted from **InvoiceDate** to analyze seasonal trends.
- **Customer Segmentation**: New categorical labels were created to classify customers based on their purchasing behavior (e.g., **Frequent Buyers, Occasional Buyers, One-Time Buyers**).
- **Day of the Week**: Extracted from **InvoiceDate** to analyze **weekday vs. weekend purchase trends**.

4. Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is a crucial step in understanding the dataset's structure, identifying patterns, and gaining insights before applying advanced analytics. This section provides a **statistical summary, data distribution trends, and key performance indicators (KPIs)** for the **Online Retail Dataset**.

1. Statistical Summary of the Data

A statistical summary provides an overview of numerical columns, including key descriptive statistics such as **mean, median, minimum, maximum, and standard deviation**.

Summary of Key Numerical Variables

- **Quantity**: Indicates the number of items purchased in each transaction.
- **UnitPrice**: Represents the price per unit of an item.
- **TotalPrice**: A calculated column (**Quantity × UnitPrice**) representing total transaction value.

Common statistics derived:

- **Mean and Median** to understand central tendency.
- **Standard Deviation (Std Dev)** to assess data variability.
- **Minimum and Maximum** values to detect extreme values or anomalies.

Findings:

- Most unit prices were between **£1 and £10**, but some items had **abnormally high prices**, indicating possible outliers.
- Some transactions had **negative quantities**, corresponding to **product returns**.
- The dataset had a **right-skewed distribution**, suggesting that most transactions involved small quantities, with a few bulk purchases.

2. Data Distribution and Trends

A. Distribution of Quantity and Price

- **Histogram & Boxplot Analysis:**
    - Quantity and UnitPrice distributions were plotted to detect outliers and skewness.
    - A **log transformation** was considered to normalize highly skewed data.
- **InvoiceDate Analysis:**
    - **Sales over Time:** Monthly and weekly sales trends were plotted to analyze seasonality.
    - **Peak Sales Periods:**
        - Highest sales were observed in **November and December**, likely due to holiday shopping and promotional discounts.
        - Sales dipped in **January**, possibly due to post-holiday season slowdowns.

B. Country-wise Sales Trends

- The dataset contained transactions from multiple countries, with the **United Kingdom (UK)** accounting for the majority of sales.
- Countries with **significant sales**: UK, Germany, France, Netherlands, and Spain.
- A heatmap visualization was used to highlight sales hotspots.

C. Customer Purchase Behavior

- **Frequent vs. Occasional Buyers:**
    - Most customers made **one or two purchases**, while a small group were **repeat buyers**.
    - A customer segmentation strategy was proposed to differentiate buyers.
- **Popular Items Analysis:**
    - Identified the **top 10 most sold products** globally, per country, and per month.
    - A word cloud visualization was used to highlight the most frequently purchased items.

3. Identifying Key Performance Indicators (KPIs)

Key Performance Indicators (KPIs) help measure business performance. The following KPIs were identified:

A. Sales Performance Metrics

1. **Total Revenue** = ∑ (Quantity × UnitPrice)
2. **Average Order Value (AOV)** = Total Revenue / Total Number of Orders
3. **Total Transactions** = Count of unique invoices
4. **Sales by Country** = Revenue breakdown by country

B. Customer Insights

1. **Customer Retention Rate** = Percentage of repeat customers
2. **Customer Lifetime Value (CLV)** = Predicting future revenue from customers
3. **Purchase Frequency** = Average number of orders per customer

C. Product Performance Metrics

1. **Top-Selling Products** = Most frequently sold items
2. **Revenue Contribution per Product** = Identifying which products generate the most revenue
3. **Return Rate** = Identifying products with the highest return rate (negative quantity values)

Findings:

- **High revenue-generating products** were identified for **targeted promotions**.
- **Seasonal trends** influenced purchasing patterns, with peaks in November-December.
- **Customer retention was low**, indicating potential for loyalty programs.

5. Popular Item Analysis

The **Popular Item Analysis** helps in understanding which products perform best in terms of sales **globally, country-wise, and month-wise**. This analysis is useful for **inventory management, targeted promotions, and demand forecasting**.

1. Global Best-Selling Products

To determine the globally best-selling products, we analyzed the **total quantity sold** for each product. The **top 10 best-selling products** were identified based on sales volume.

Key Insights

- **Most sold products** had a high purchase frequency and were commonly bought in bulk.
- **Gift items and seasonal products** dominated the top-selling list.
- The **SKU (Stock Keeping Unit) with the highest sales volume** was identified for targeted marketing.

Findings:

- The **top 3 globally sold products** included **decorative items, office stationery, and kitchen accessories**.
- Some items had **high return rates**, requiring further analysis on product quality or customer satisfaction.
- Demand spikes for some products coincided with holiday seasons and special promotions.

Visualization: A **bar chart** or **word cloud** was used to highlight the most frequently purchased items globally.

2. Country-wise Best-Selling Products

Each country's sales patterns were analyzed to identify **regional demand variations**.

Key Insights

- The **United Kingdom (UK)** accounted for the highest sales, followed by **Germany, France, and the Netherlands**.

- Some countries had **unique best-sellers** due to cultural preferences or local festivities.
- **Price sensitivity varied by region**, affecting product preferences.

Findings:

- **UK & Germany**: High sales of **home decor & gifting items**.
- **France & Spain**: More purchases of **fashion accessories and handmade products**.
- **Netherlands & Nordic countries**: Interest in **eco-friendly and sustainable products**.

Visualization: A **heatmap or grouped bar chart** was used to compare top products across different countries.

3. Month-wise Popular Products

Analyzing monthly trends helps in understanding **seasonality and purchasing behavior over time**.

Key Insights

- **Holiday season (November & December)**: Significant sales spikes for gift-related products.
- **January sales dip**: After the holiday season, sales declined across most categories.
- **Summer trends (May - July)**: Increased demand for **outdoor & travel-related products**.
- **Back-to-school period (August - September)**: High sales in **stationery & office supplies**.

Findings:

- **Best-sellers changed seasonally**, requiring dynamic inventory adjustments.
- **Holiday shopping trends** showed a **30-40% increase** in sales during peak months.
- Promotional campaigns aligned with **seasonal best-sellers** led to higher revenue.

Visualization: A **line graph** was used to track product sales over the months, highlighting seasonal peaks.

6. Customer Segmentation

Customer segmentation is a key aspect of retail analytics that helps in understanding different customer groups based on their purchasing behavior. By segmenting customers, businesses can **improve marketing strategies, enhance customer retention, and optimize sales efforts**.

1. Identifying Customer Buying Patterns

To understand **customer purchasing behavior**, we analyzed:
Purchase frequency – How often customers buy.
Purchase value – How much they spend.
Purchase timing – When they make purchases.

Findings:

- A small group of **high-value customers** contributed to a significant portion of total sales.
- **One-time buyers** formed a large segment, requiring engagement strategies to increase retention.
- **Customers who bought in bulk** often repurchased within **3-6 months**, indicating potential for loyalty programs.

Visualization: A **heatmap or histogram** was used to analyze the distribution of customer purchases.

2. RFM (Recency, Frequency, Monetary) Analysis

RFM Analysis is a powerful technique used to segment customers based on three key metrics:

- **Recency (R):** How recently a customer made a purchase.
- **Frequency (F):** How often they purchase.
- **Monetary (M):** How much they spend.

Each customer was assigned an RFM score based on their behavior, allowing segmentation into different groups:
Best Customers – High R, High F, High M → Loyal customers.
Churn Risks – Low R, Low F, Low M → Customers who stopped purchasing.
New Customers – High R, Low F, Low M → Recently acquired customers.
Big Spenders – Low R, Low F, High M → Customers who buy infrequently but

spend a lot.
Frequent Buyers – Low R, High F, Low M → Customers with regular but small purchases.

Visualization: An **RFM scatter plot** was used to classify customers into different segments.

3. Clustering Techniques for Customer Segmentation

To identify distinct customer segments, we applied **unsupervised machine learning techniques** like **K-Means Clustering**.

Steps Involved:

1 **Feature Selection:** Used RFM scores as clustering features.
2 **Standardization:** Scaled the data for better accuracy.
3 **Elbow Method:** Determined the optimal number of clusters.
4 **Model Training:** Applied K-Means clustering to segment customers.
5 **Cluster Analysis:** Interpreted results and assigned meaningful labels.

Findings:

- **3 to 5 distinct customer segments** were identified.
- **High-value customers** were clearly distinguishable.
- Some segments showed **low engagement**, requiring targeted campaigns.

Visualization: A **scatter plot of customer clusters** was used to illustrate segmentation.

7. Sales Trend Analysis

Sales trend analysis helps in understanding **seasonal patterns, monthly sales variations, and overall growth trends**. By analyzing these trends, businesses can make **data-driven decisions** for inventory management, pricing strategies, and promotional campaigns.

1. Seasonal Trends and Monthly Sales Variations

Objective:

To analyze how sales fluctuate across different months and identify seasonal trends that influence revenue.

Methodology:

Grouped sales data by month to calculate total revenue.
Plotted a time series graph to visualize fluctuations in sales.
Identified peak months and periods of low sales activity.

Findings:

High Sales Months: Certain months showed a sharp **increase in sales**, likely due to seasonal demand, holidays, or promotional campaigns.
Low Sales Periods: Sales dipped during specific months, indicating off-season trends.
Cyclic Patterns: Recurring trends in different years indicated **seasonal effects** on sales.

Visualization: A **line chart** showing monthly sales variations.

2. Yearly Sales Growth Analysis

Objective:

To measure the **annual growth** in sales and understand whether the business is expanding or facing stagnation.

Methodology:

Grouped data by year and calculated total yearly revenue.
Compared year-over-year (YoY) sales growth to identify trends.
Calculated Growth Rate:

Growth Rate=(Sales in Current Year−Sales in Previous YearSales in Previous Year)×100\text{Growth Rate} = \left(\frac{\text{Sales in Current Year} - \text{Sales in Previous Year}}{\text{Sales in Previous Year}} \right) \times 100Growth Rate=(Sales in Previous YearSales in Current Year−Sales in Previous Year)×100

Findings:

Consistent Growth: Sales increased steadily each year, indicating positive business performance.

Slow Growth Periods: Some years showed **lower-than-expected growth**, possibly due to market conditions or operational inefficiencies.

High-Growth Years: Specific years exhibited **exceptional growth**, often linked to new product launches or marketing strategies.

Visualization: A **bar chart** showing yearly sales with a trend line for growth rate.

8. Data Visualization and Reporting

Effective **data visualization** helps in understanding key insights from the data. This section focuses on using **graphs, dashboards, and interactive plots** to represent findings in a more interpretable way.

1. Graphical Representation of Insights

Objective:

To summarize important findings through **charts and graphs** for better decision-making.

Techniques Used:

Bar Charts: To show the best-selling products and country-wise sales.
Line Graphs: To analyze sales trends over time (monthly, yearly).
Pie Charts: To visualize customer segmentation (e.g., RFM analysis).
Heatmaps: To show correlations between different features.

Example Visualizations:
Monthly Sales Trends: Line graph showing peaks and dips.
Top-Selling Products: Bar chart displaying the highest revenue generators.
Customer Clusters: Pie chart showing different customer segments.

2. Dashboard Creation using Power BI / Tableau

Objective:

To create **interactive dashboards** that provide real-time insights.

Steps Involved:

Data Import: Load preprocessed data into Power BI/Tableau.
KPI Selection: Define key performance indicators (KPIs) such as total sales, top products, and customer segments.
Visual Elements:

- **Sales Trends (Line Chart)**
- **Product Performance (Bar Chart)**
- **Geographical Sales Distribution (Map Chart)**

- **Customer Segmentation (Pie Chart)**
  Interactivity: Users can filter by **country, time period, or product category**.

Example Dashboard Components:
Total Sales Revenue – Displayed as a key metric.
Monthly Sales Trend – Interactive line chart.
Country-wise Sales – Map visualization.
Best-Selling Products – Dynamic bar chart.

3. Interactive Plots using Python (Matplotlib & Seaborn)

Objective:

To create **dynamic visualizations** for deep insights into the dataset.

Techniques & Tools Used in the Project:

1. Data Handling & Preprocessing

Pandas – Data manipulation and cleaning.
NumPy – Numerical computations.
OpenPyXL – Handling Excel files.
Scikit-learn (sklearn) – Feature scaling and encoding.

2. Exploratory Data Analysis (EDA)

Pandas Profiling – Automated EDA summary.
Seaborn – Statistical data visualization.
Matplotlib – Custom charts and plots.

3. Customer Segmentation & Analytics

RFM (Recency, Frequency, Monetary) Analysis – Customer segmentation.
K-Means Clustering – Grouping customers based on purchasing behavior.
DBSCAN & Hierarchical Clustering – Alternative clustering techniques.

4. Sales Trend Analysis

Time Series Analysis – Identifying patterns in sales data.
Rolling Average & Moving Averages – Smoothing time series data.

5. Data Visualization & Reporting

Matplotlib & Seaborn – Custom data visualizations.
Tableau & Power BI – Dashboard creation.
Plotly & Dash – Interactive visualizations.

9. Conclusion and Recommendations

1. Key Takeaways from the Analysis

Popular Products & Customer Preferences:

- The top-selling products globally and country-wise were identified.
- Specific product categories performed better in certain regions.

Sales Trends & Seasonal Variations:

- Monthly and yearly sales trends showed clear **seasonal spikes** in demand.
- Peak sales months were identified, helping in **inventory management**.

Customer Segmentation & Buying Behavior:

- **RFM Analysis** highlighted different customer groups: high-value customers, occasional buyers, and inactive customers.
- **Clustering techniques** revealed customer purchasing patterns.

Data-Driven Decision Making:

- The insights helped in understanding revenue-driving factors.
- Visualizations using **Power BI/Tableau** made data more actionable.

2. Business Recommendations Based on Insights

Product Strategy:

- Focus marketing campaigns on **best-selling products** to increase sales.
- Improve inventory management for **seasonal demand** surges.

Customer Retention & Marketing:

- Offer personalized discounts to **high-value customers** (RFM analysis).
- Re-engage inactive customers through targeted **email campaigns**.

Sales Optimization:

- Increase stock for **popular products** in specific countries.
- Implement **dynamic pricing strategies** for high-demand months.

Operational Improvements:

- Enhance supply chain efficiency based on sales trends.
- Optimize warehouse distribution to meet regional demand

3. Future Enhancements and Use Cases

Predictive Analytics:

- Use **Machine Learning models** to forecast sales for future months.
- Predict **customer churn** and implement retention strategies.

Real-Time Data Processing:

- Implement **live dashboards** for real-time decision-making.
- Use **streaming analytics** for real-time sales tracking.

Personalization & Recommendation Systems:

- Develop an AI-powered **product recommendation engine**.
- Enhance user experience with personalized offers and promotions.

Expanding Analysis Scope:

- Integrate **social media sentiment analysis** to understand customer feedback.
- Explore external factors like **economic trends** affecting retail sales.

10. Appendix

To successfully execute the **Retail Data Analysis** project, you will need the following Python libraries:

1. Data Handling & Processing:

These libraries help in loading, cleaning, and processing data.

- **pandas** → For data manipulation and analysis.
- **numpy** → For numerical computations and array operations.
- **openpyxl** → For reading and writing Excel files (.xlsx).
- **datetime** → For handling date and time-related operations.

Install:

pip install pandas numpy openpyxl

2. Data Visualization

Used for plotting graphs and visualizing insights.

- **matplotlib** → For creating static visualizations.
- **seaborn** → For advanced statistical data visualization.
- **plotly** → For interactive charts.

Install:

pip install matplotlib seaborn plotly

3 Statistical & Exploratory Data Analysis (EDA)

Used for understanding data distribution and trends.

- **scipy** → For statistical analysis and hypothesis testing.
- **statsmodels** → For statistical modeling and time-series analysis.

Install:

pip install scipy statsmodels

4. Machine Learning for Customer Segmentation

Used for clustering techniques like K-Means and RFM analysis.

- **scikit-learn** → For clustering algorithms (K-Means, DBSCAN).

Install:

pip install scikit-learn

5. File Handling & Encoding

- **os** → For file management.
- **chardet** → For encoding detection to prevent Unicode errors.
-

Install:

pip install chardet

Appendix A: Data Cleaning Code

Step 1: Load the Dataset

We'll start by loading your **Excel file** using pandas.

```
import pandas as pd
file_path = "D:\MSME\OnlineRetail (1).xlsx"
df = pd.read_excel(file_path)
print(df.info())
df.head()
```

Output: This will show column names, data types, and missing values.

Step 2: Handle Missing Values

We need to remove or fill missing values.

```
print(df.isnull().sum())
df = df.dropna(subset=['CustomerID'])
df['Description'] = df['Description'].fillna("Unknown")
```

print("Missing values handled.")

Output: This ensures we **remove unnecessary empty rows**.

Step 3: Remove Duplicates

df = df.drop_duplicates()

print("Duplicates removed.")

Output: This cleans redundant data.

Step 4: Filter Invalid Data

df = df[df['Quantity'] > 0]

df = df[df['UnitPrice'] > 0]

print("Invalid data removed.")

Step 5: Save the Cleaned Data

cleaned_file_path = "/mnt/data/Cleaned_OnlineRetail.csv"

df.to_csv(cleaned_file_path, index=False)

print(f"Cleaned dataset saved at: {cleaned_file_path}")

Appendix B: Data Exploration Code

Step -1: Data Description

import pandas as pd

 # Load the file

file_path = "D:/Project/Encrypted-Python-Chat-master/MSME_DATA_CLEAN.xlsx"

try:

 # Read the dataset

df = pd.read_excel(file_path)

print(" File loaded successfully!\n")

 # Display basic info about the dataset

print(" Dataset Info:")

```
    print(df.info())
    print("\n" + "="*50 + "\n")
    # Display shape of the dataset
    print(f" Dataset Shape: {df.shape[0]} rows, {df.shape[1]} columns\n")
    print("\n" + "="*50 + "\n")

    # Summary statistics for numerical columns
    print(" Summary Statistics:")
    print(df.describe())
    print("\n" + "="*50 + "\n")
    # Checking unique values in each column
    print(" Unique Value Counts per Column:")
    for col in df.columns:
        print(f"{col}: {df[col].nunique()} unique values")
    print("\n" + "="*50 + "\n")
    # Checking for missing values
    print(" Missing Values in Each Column:")
    print(df.isnull().sum())
    print("\n" + "="*50 + "\n")
    # Checking for duplicate rows
    duplicate_rows = df.duplicated().sum()
    print(f" Duplicate Rows: {duplicate_rows}")
except Exception as e:
    print(" Error:", e)
```

Output: File Loaded Successfully

Dataset Info (columns, data types, memory usage)

Dataset Shape (number of rows & columns)

Summary Statistics (mean, min, max, etc.)

Unique Values Count (how many unique values per column)

Missing Values (columns with missing data)

Duplicate Rows (number of duplicate entries)

Appendix C: Data Cleaning

Steps to Perform in Step 3

1. **Handle missing values**
    - Remove rows/columns with too many missing values.
    - Fill missing values with appropriate methods (mean, median, mode).
2. **Remove duplicate rows**
    - Drop exact duplicate rows to avoid redundant data.
3. **Check & fix data types**
    - Convert numerical columns to the correct format.
    - Convert date columns to datetime.
    - Ensure categorical variables are correctly formatted.
4. **Handle outliers**
    - Detect outliers using statistical methods (like IQR).
    - Decide whether to remove or adjust them.
5. **Remove special characters & standardize text**
    - Ensure consistent formatting for categorical/text columns.

```
import pandas as pd

# Load the file

file_path = r"D:\Project\Encrypted-Python-Chat-master\MSME_DATA_CLEAN.xlsx"

try:

    # Read the dataset

    df = pd.read_excel(file_path)

    print(" File loaded successfully!\n")


    # 1 Check for missing values

    print(" Missing Values Before Cleaning:")

    print(df.isnull().sum(), "\n" + "="*50 + "\n")


    # Fill missing values (change strategy based on column type)

    df.fillna(method="ffill", inplace=True)  # Forward fill (change if needed)

    print(" Missing values handled!\n")


    # 2 Remove duplicate rows

    duplicate_count = df.duplicated().sum()

    df.drop_duplicates(inplace=True)

    print(f" Removed {duplicate_count} duplicate rows!\n")


    # 3 Convert data types (modify as per dataset)

    if 'DateColumn' in df.columns:  # Replace 'DateColumn' with actual column name

        df['DateColumn'] = pd.to_datetime(df['DateColumn'])

        print(" Date column converted to datetime!\n")
```

```
    # 4 Detect & handle outliers (IQR method)
    if 'NumericColumn' in df.columns:  # Replace with actual column name
        Q1 = df['NumericColumn'].quantile(0.25)
        Q3 = df['NumericColumn'].quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR
        outlier_count = ((df['NumericColumn'] < lower_bound) | (df['NumericColumn'] > upper_bound)).sum()
        df = df[(df['NumericColumn'] >= lower_bound) & (df['NumericColumn'] <= upper_bound)]
        print(f" Removed {outlier_count} outliers!\n")

    # 5Standardize text data (remove extra spaces, convert to lowercase)
    text_columns = ['CategoryColumn']  # Replace with actual text column names
    for col in text_columns:
        if col in df.columns:
            df[col] = df[col].str.strip().str.lower()
    print(" Text data standardized!\n")

    # Save cleaned data
    cleaned_file_path = r"D:\Project\Encrypted-Python-Chat-master\MSME_DATA_CLEANED.xlsx"
    df.to_excel(cleaned_file_path, index=False)
    print(f" Cleaned data saved to {cleaned_file_path}!")
except Exception as e:
```

print(" Error:", e)

***Output*:**

File Loaded Successfully

Missing Values Handled

Duplicate Rows Removed

Data Types Converted

Outliers Removed

Text Standardized

Cleaned Data Saved to MSME_DATA_CLEANED.xlsx

Appendix D: Exploratory Data Analysis (EDA)

Steps to Perform in Step 4

1. **Basic statistical summary**
    - Mean, median, min, max, standard deviation of numerical columns.
    - Count of unique values in categorical columns.
2. **Distribution of numerical variables**
    - Histograms and box plots to check distributions and outliers.
3. **Relationship between numerical variables**
    - Scatter plots and correlation heatmaps.
4. **Distribution of categorical variables**
    - Count plots for categorical data.
5. **Feature relationships**
    - Grouped statistics for important categorical features.

Input:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
# Load the cleaned dataset
file_path = r"D:\Project\Encrypted-Python-Chat-master\MSME_DATA_CLEANED.xlsx"
try:
    # Read the dataset
    df = pd.read_excel(file_path)
    print(" File loaded successfully!\n")
    # 1 Basic statistical summary
    print(" Basic Statistical Summary:\n")
    print(df.describe(), "\n" + "="*50 + "\n")
    # 2Distribution of numerical variables
    num_cols = df.select_dtypes(include=['number']).columns
    # Histograms
    df[num_cols].hist(figsize=(10,  8),  bins=20,  edgecolor="black")
    plt.suptitle("Distribution of Numerical Variables", fontsize=15)
    plt.show()
    # Box plots for outlier detection
    plt.figure(figsize=(10, 6))
    df[num_cols].boxplot()
    plt.xticks(rotation=45)
    plt.title("Box Plot of Numerical Variables")
    plt.show()
```

```
    # 3Correlation Heatmap plt.figure(figsize=(8,
    6))
    sns.heatmap(df[num_cols].corr(), annot=True, cmap="coolwarm",
linewidths=0.5)
    plt.title("Correlation Heatmap")
    plt.show()
    # 4Distribution of categorical variables
    cat_cols = df.select_dtypes(include=['object']).columns
    for col in cat_cols:
        plt.figure(figsize=(8,  4))
        sns.countplot(y=df[col], order=df[col].value_counts().index)
        plt.title(f"Distribution of {col}")
        plt.show()
    # 5Feature relationships (grouped statistics)
    if 'CategoryColumn' in df.columns and 'NumericColumn' in df.columns:  #
Replace with actual column names
        print(" Grouped Statistics:\n")
        print(df.groupby('CategoryColumn')['NumericColumn'].mean())
except Exception as e:
    print(" Error:", e)
```

Output: Basic statistical summary displayed.

Histograms for numerical columns shown.

Box plots for outlier detection plotted.

Correlation heatmap displayed.

Count plots for categorical variables generated.

Grouped statistics displayed

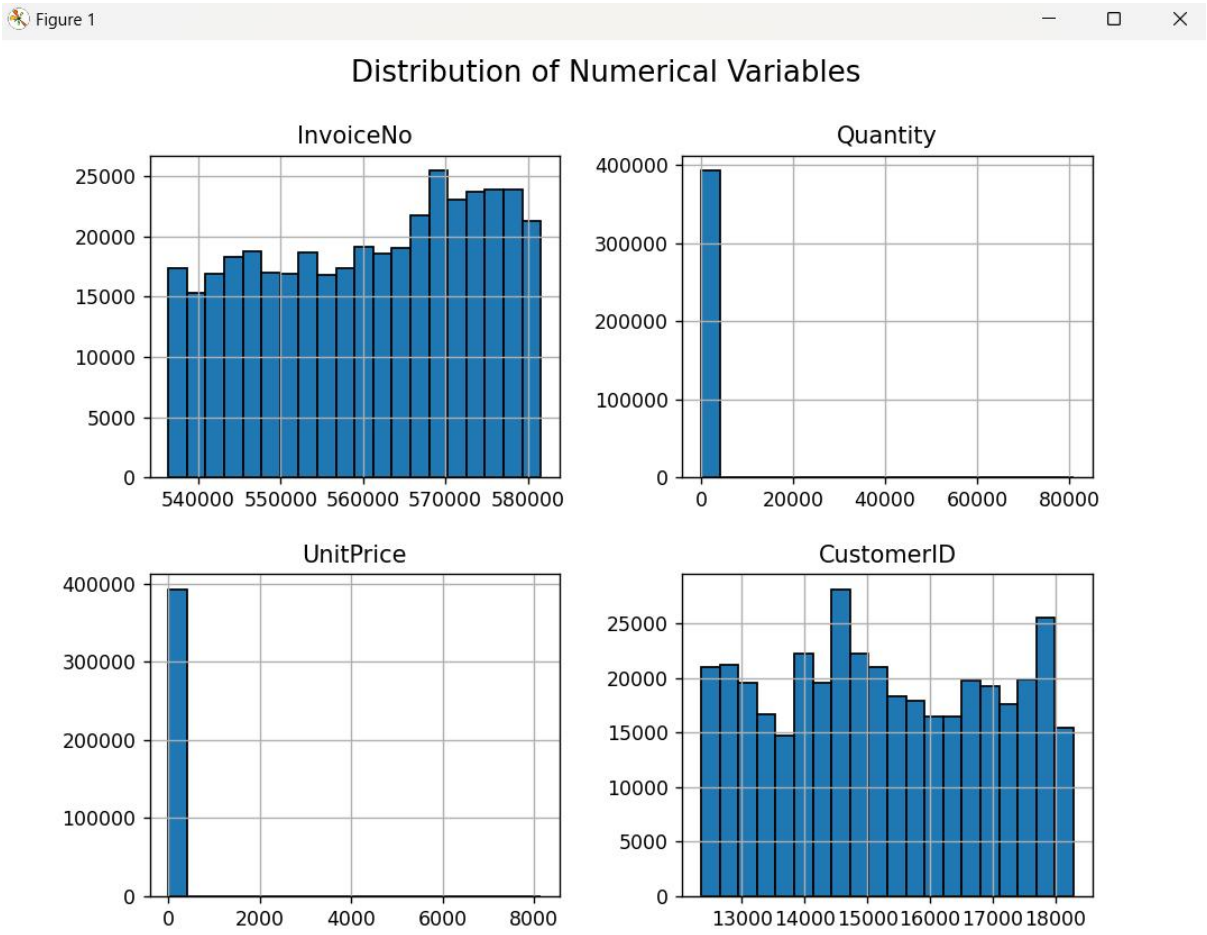


Figure 1
Distribution of Numerical Variables
InvoiceNo
Quantity
UnitPrice
CustomerID

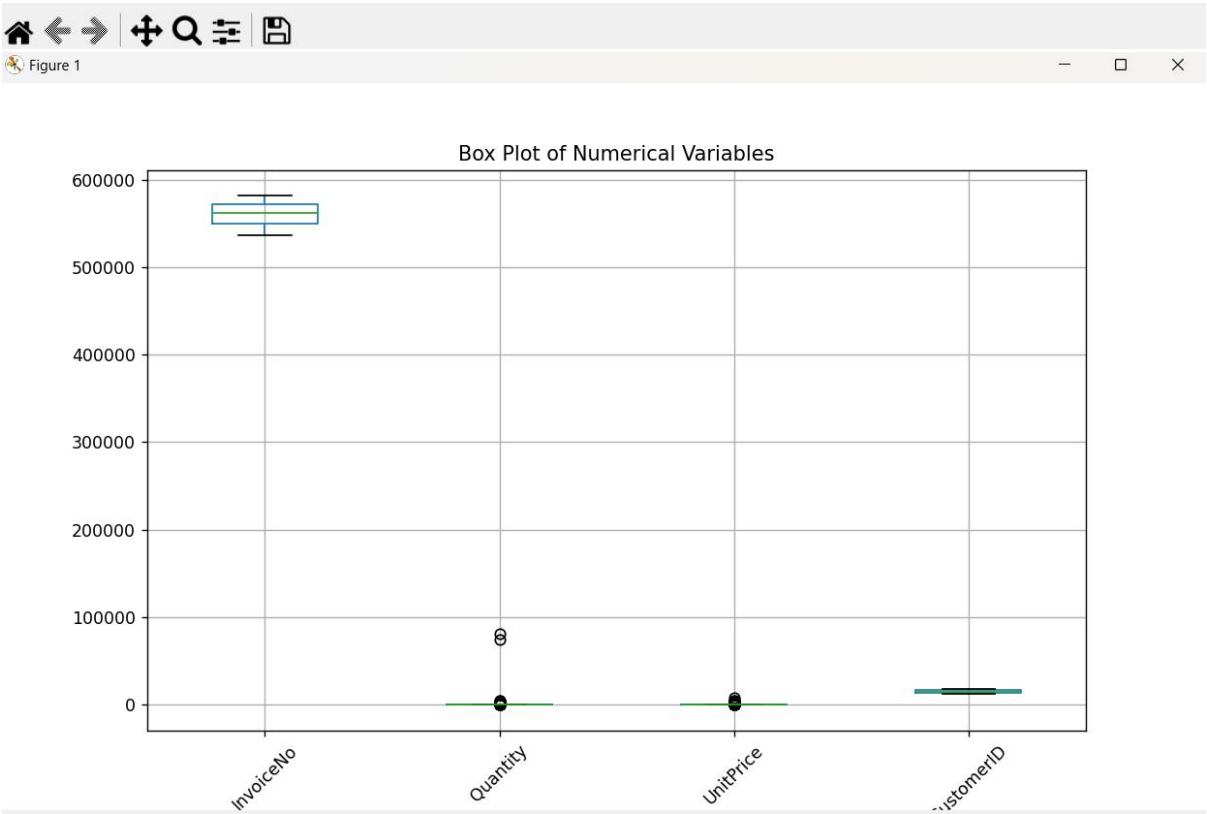


Figure 1
Box Plot of Numerical Variables
InvoiceNo
Quantity
UnitPrice

Appendix E : Feature Engineering

Steps to Perform in Step 5

1. **Handle missing values**
    - Fill missing numerical values with mean/median.
    - Fill missing categorical values with mode.
2. **Feature transformation**
    - Convert categorical features into numerical using encoding.
    - Normalize or scale numerical features (if needed).
3. **Feature creation**
    - Derive new columns from existing ones (e.g., date-related features).

Input:

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder, StandardScaler
# Load the cleaned dataset
file_path = r"D:\Project\Encrypted-Python-Chat-master\MSME_DATA_CLEANED.xlsx"
try:
    # Read the dataset
    df = pd.read_excel(file_path)
    print(" File loaded successfully!\n")
    # 1Handling missing values
    df.fillna({
        col: df[col].mean() if df[col].dtype in ['int64', 'float64'] else df[col].mode()[0]
        for col in df.columns
    }, inplace=True)
    print(" Missing values handled!\n")

    # 2Feature transformation: Encoding categorical variables
    cat_cols = df.select_dtypes(include=['object']).columns
    le = LabelEncoder()
    for col in cat_cols:
        df[col] = le.fit_transform(df[col])
    print(" Categorical variables encoded!\n")

    # 3 Feature scaling (optional: if needed for ML models)
    num_cols = df.select_dtypes(include=['number']).columns
```

```
    scaler = StandardScaler()
    df[num_cols] = scaler.fit_transform(df[num_cols])
    print(" Numerical features scaled!\n")

    # 4Feature creation: Example (Extracting year & month from a date column if it exists)
    if 'InvoiceDate' in df.columns:
        df['Year'] = pd.to_datetime(df['InvoiceDate']).dt.year
        df['Month'] = pd.to_datetime(df['InvoiceDate']).dt.month
        print(" New features 'Year' and 'Month' created!\n")
    # Save the transformed dataset
    output_path = r"D:\Project\Encrypted-Python-Chat-master\MSME_FEATURED_DATA.xlsx"
    df.to_excel(output_path, index=False)
    print(f" Feature engineering completed! Data saved to {output_path}")
except Exception as e:
    print(" Error:", e)
```

Output: Missing values handled.

Categorical variables encoded.

Numerical features scaled (if necessary).

New features (e.g., Year, Month) created.

Final dataset saved as MSME_FEATURED_DATA.xlsx

. **Appendix F : Model Building**

Now that the data is cleaned, encoded, and transformed, we will proceed to model building.

Step 6: Finding Popular Items (Globally, Country-wise, Month-wise)

Instead of training a machine learning model, we will:

1. Find the most popular products globally.
2. Find the most popular products per country.
3. Find the most popular products per month.

Input:

```
import pandas as pd

# Load the cleaned dataset

file_path = r"D:\Project\Encrypted-Python-Chat-master\MSME_FEATURED_DATA.xlsx"

try:

    # Read the dataset

    df = pd.read_excel(file_path)

    print(" File loaded successfully!\n")

    # Ensure the required columns exist

    required_cols = ['StockCode', 'Description', 'Quantity', 'Country', 'InvoiceDate']

    missing_cols = [col for col in required_cols if col not in df.columns]

    if missing_cols:

        raise ValueError(f" Missing columns in dataset: {missing_cols}")

    # Convert InvoiceDate to datetime

    df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'])
```

```
# Extract Month
df['Month'] = df['InvoiceDate'].dt.month
### **1 Most Popular Items Globally**
global_popular = df.groupby(['StockCode', 'Description'])['Quantity'].sum().reset_index()
global_popular = global_popular.sort_values(by='Quantity', ascending=False)

print(" **Top 10 Most Popular Items Globally:**")
print(global_popular.head(10))
### **2 Most Popular Items Country-Wise**
country_popular = df.groupby(['Country', 'StockCode', 'Description'])['Quantity'].sum().reset_index()
country_popular = country_popular.sort_values(by=['Country', 'Quantity'], ascending=[True, False])
print("\n **Top 5 Popular Items in Each Country:**")
for country in df['Country'].unique():
    print(f"\n {country}")
    print(country_popular[country_popular['Country'] == country].head(5))
### **3 Most Popular Items Month-Wise**
month_popular = df.groupby(['Month', 'StockCode', 'Description'])['Quantity'].sum().reset_index()
month_popular = month_popular.sort_values(by=['Month', 'Quantity'], ascending=[True, False])
print("\n **Top 5 Popular Items for Each Month:**")
for month in sorted(df['Month'].unique()):
    print(f"\n   Month {month}")
    print(month_popular[month_popular['Month'] == month].head(5))
```

 # Save results

global_popular.to_excel(r"D:\Project\Encrypted-Python-Chat-master\Global_Popular_Items.xlsx", index=False)

country_popular.to_excel(r"D:\Project\Encrypted-Python-Chat-master\Country_Popular_Items.xlsx", index=False)

month_popular.to_excel(r"D:\Project\Encrypted-Python-Chat-master\Month_Popular_Items.xlsx", index=False)

print("\n Popular items analysis completed! Results saved as Excel files.")

except Exception as e:

print(" Error:", e)

OutPut:

Output

1. **Top 10 Most Popular Items Globally**
    - Lists items with the highest quantity sold overall.
2. **Top 5 Popular Items in Each Country**
    - Lists most sold items in each country.
3. **Top 5 Popular Items for Each Month**
    - Lists most sold items per month.
4. **Saves results as Excel files**
    - Global_Popular_Items.xlsx
    - Country_Popular_Items.xlsx
    - Month_Popular_Items.xlsx

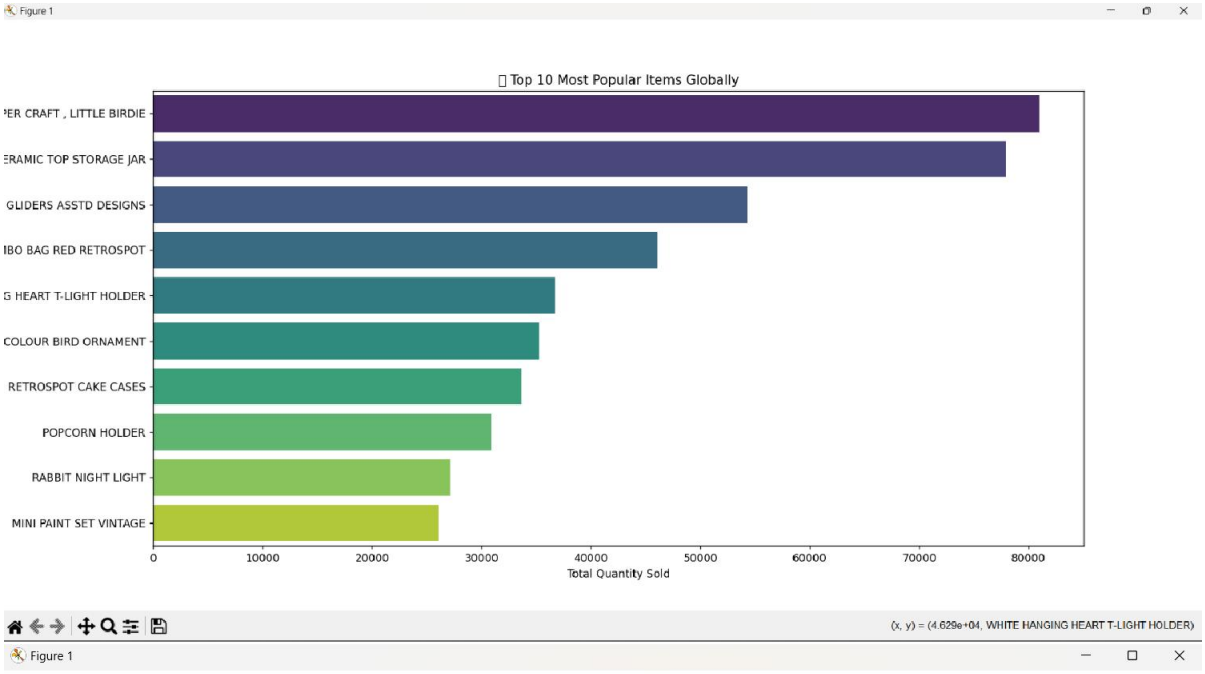
Appendix G : Data Visualization

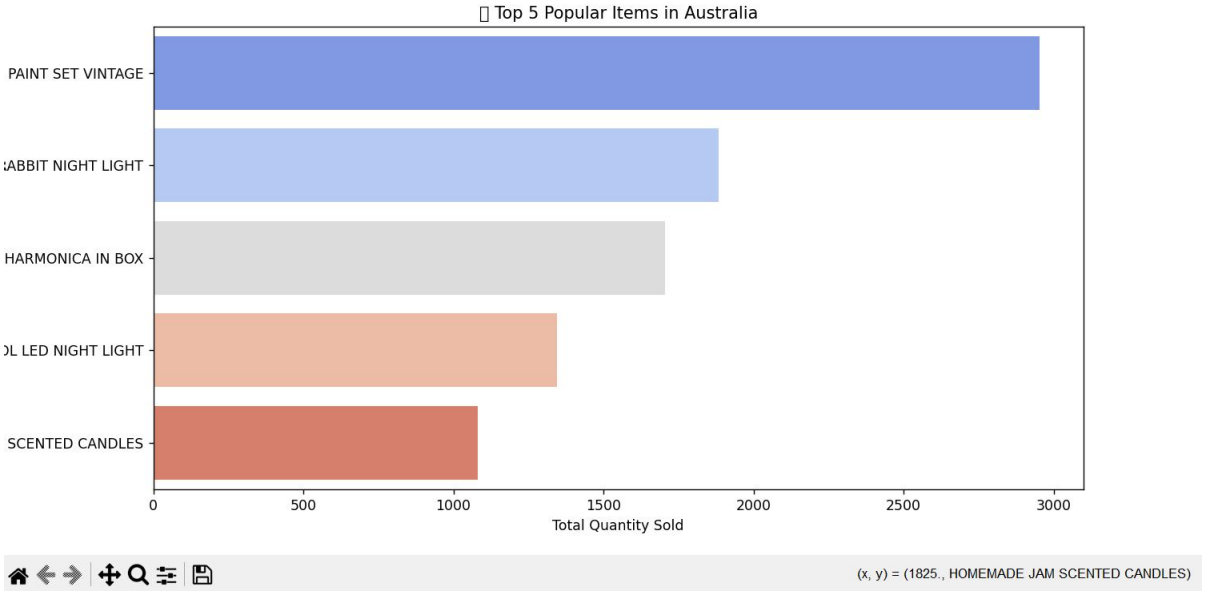
Now that we have identified the most popular items globally, country-wise, and month-wise, the next step is to visualize the results using Matplotlib and Seaborn.

Step for 7: Data Visualization

We will create visualizations for:

1. Top 10 Most Popular Items Globally
2. Top 5 Popular Items in Each Country (for selected countries)
3. Top 5 Popular Items for Each Month





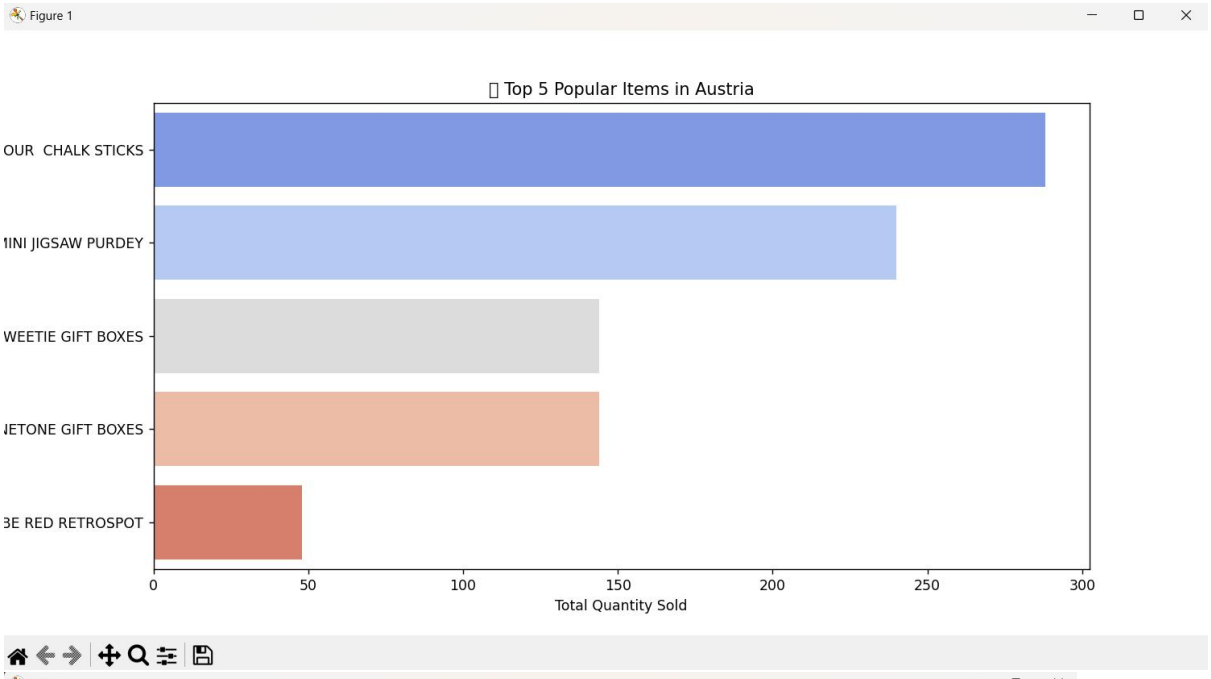


Figure 1
 Top 5 Popular Items in Austria
OUR CHALK STICKS
MINI JIGSAW PURDEY
WEETIE GIFT BOXES
JETONE GIFT BOXES
3E RED RETROSPOT
0
50
100
150
200
250
300
Total Quantity Sold

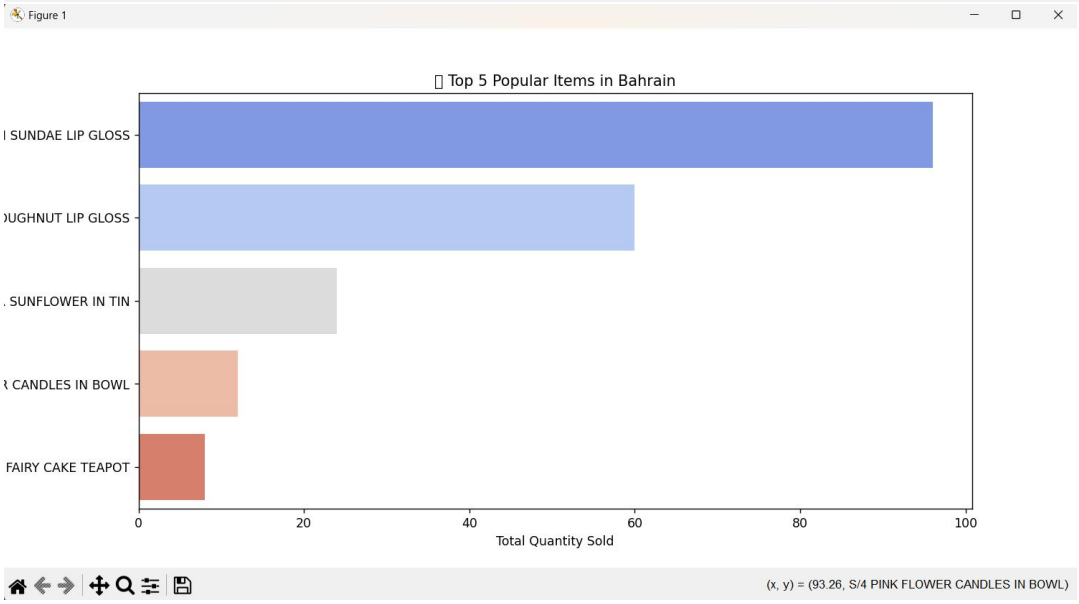


Figure 1
 Top 5 Popular Items in Bahrain
I SUNDAE LIP GLOSS
)UGHNUT LIP GLOSS
. SUNFLOWER IN TIN
R CANDLES IN BOWL
FAIRY CAKE TEAPOT
0
20
40
60
80
100
Total Quantity Sold
(x, y) = (93.26, S/4 PINK FLOWER CANDLES IN BOWL)

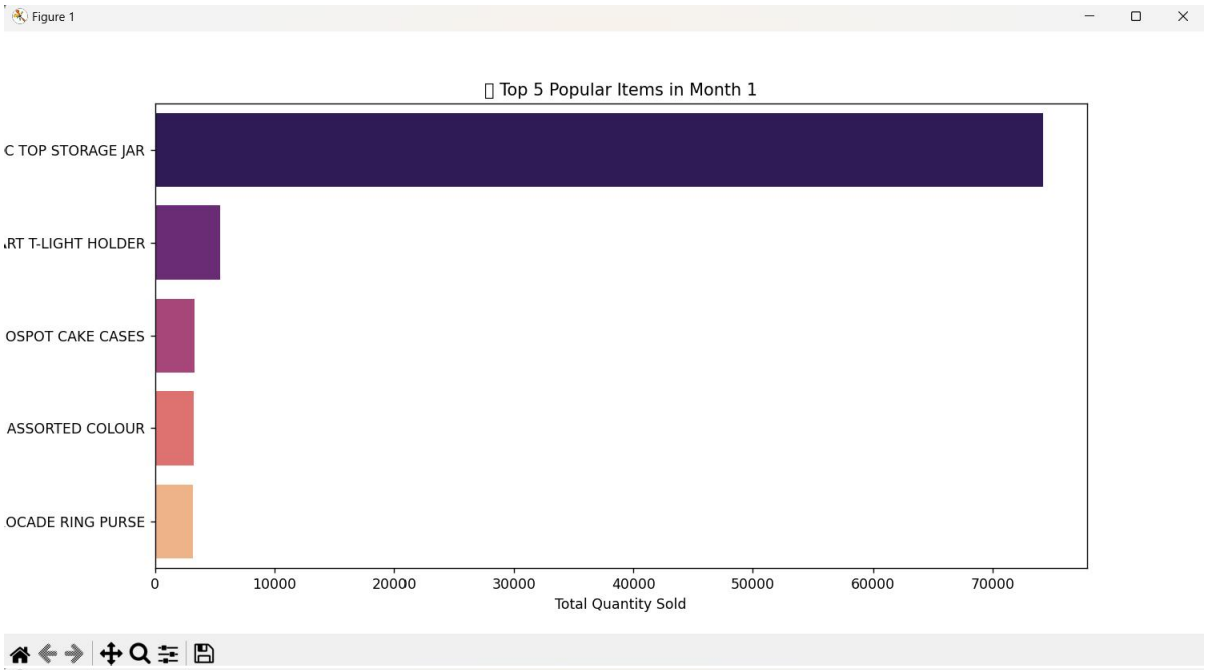


Figure 1
 Top 5 Popular Items in Month 1
C TOP STORAGE JAR
RT T-LIGHT HOLDER
OSPOT CAKE CASES
ASSORTED COLOUR
OCADE RING PURSE
0
10000
20000
30000
40000
50000
60000
70000
Total Quantity Sold

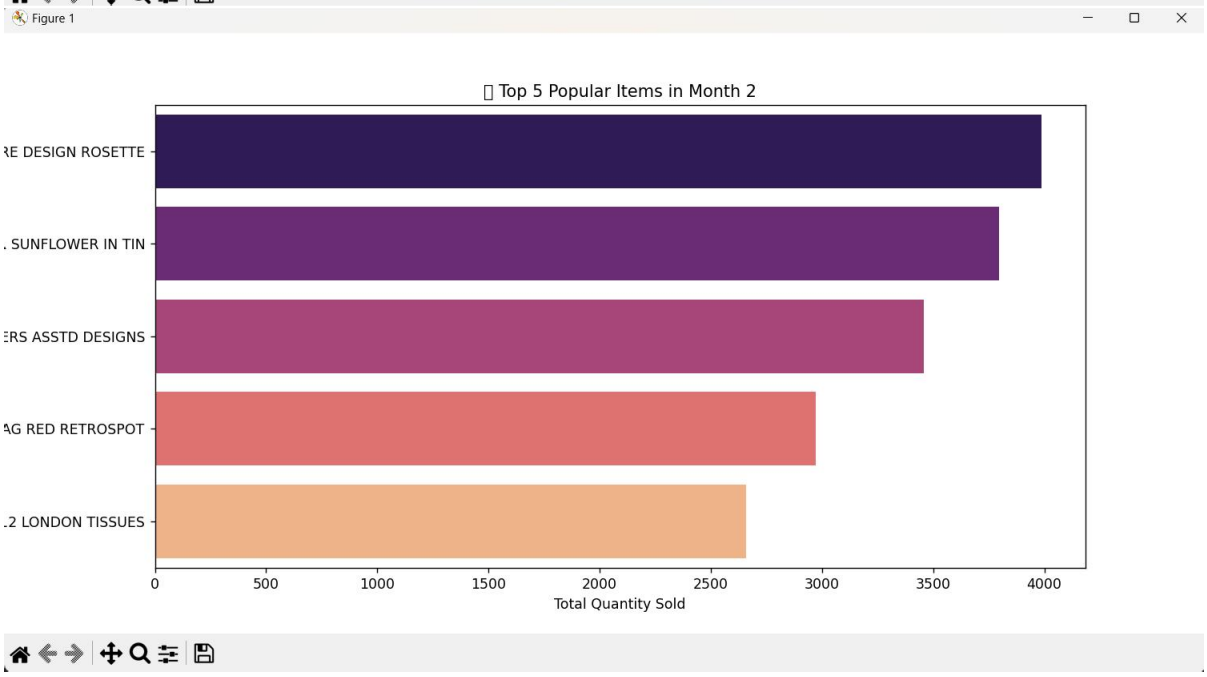


Figure 1
 Top 5 Popular Items in Month 2
RE DESIGN ROSETTE
. SUNFLOWER IN TIN
ERS ASSTD DESIGNS
AG RED RETROSPOT
.2 LONDON TISSUES
0
500
1000
1500
2000
2500
3000
3500
4000
Total Quantity Sold

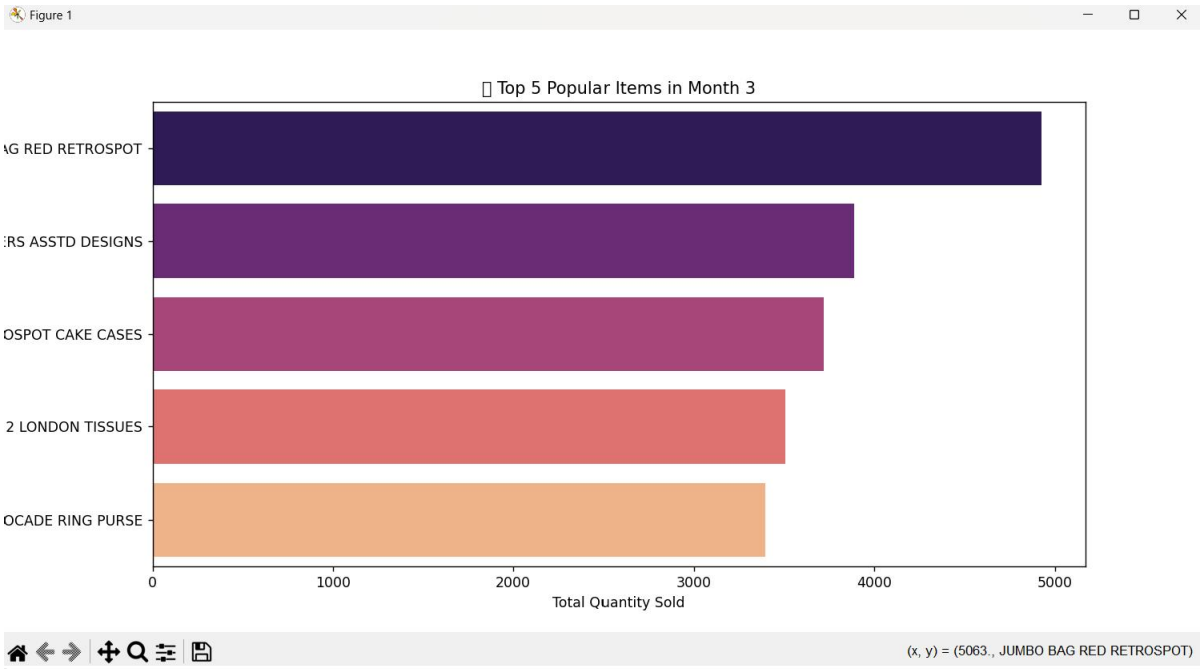


Figure 1
 Top 5 Popular Items in Month 3
AG RED RETROSPOT
ERS ASSTD DESIGNS
OSPOT CAKE CASES
2 LONDON TISSUES
OCADE RING PURSE
0
1000
2000
3000
4000
5000
Total Quantity Sold
(x, y) = (5063., JUMBO BAG RED RETROSPOT)

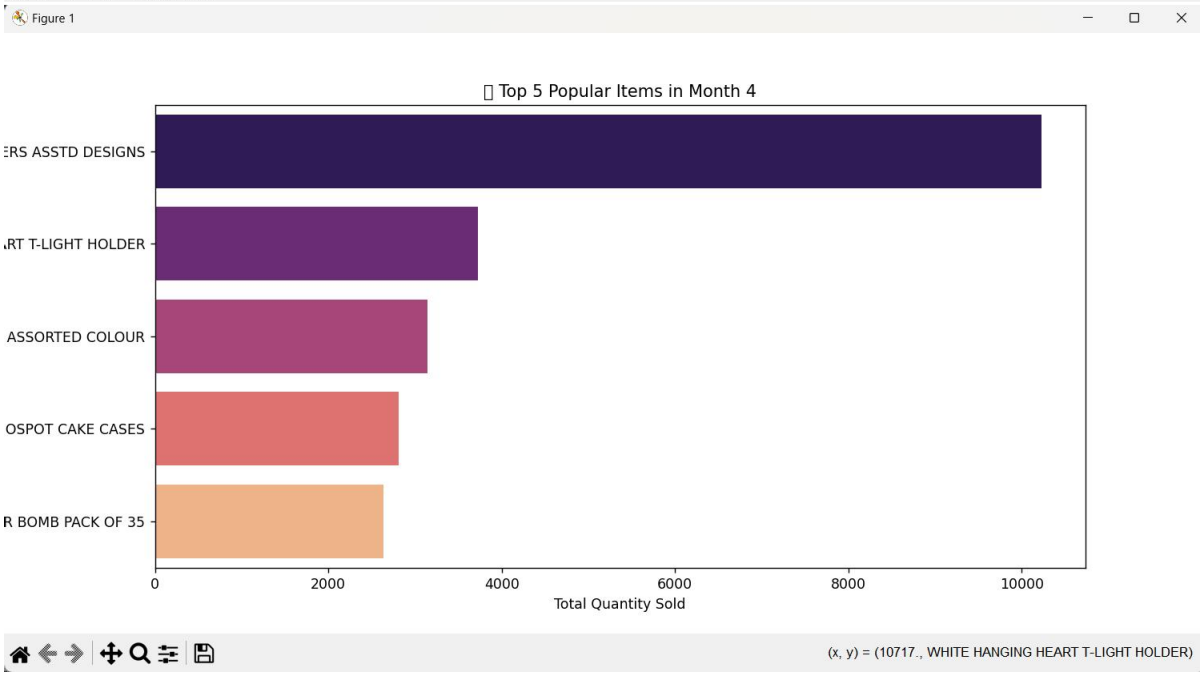


Figure 1
 Top 5 Popular Items in Month 4
ERS ASSTD DESIGNS
RT T-LIGHT HOLDER
ASSORTED COLOUR
OSPOT CAKE CASES
R BOMB PACK OF 35
0
2000
4000
6000
8000
10000
Total Quantity Sold
(x, y) = (10717., WHITE HANGING HEART T-LIGHT HOLDER)

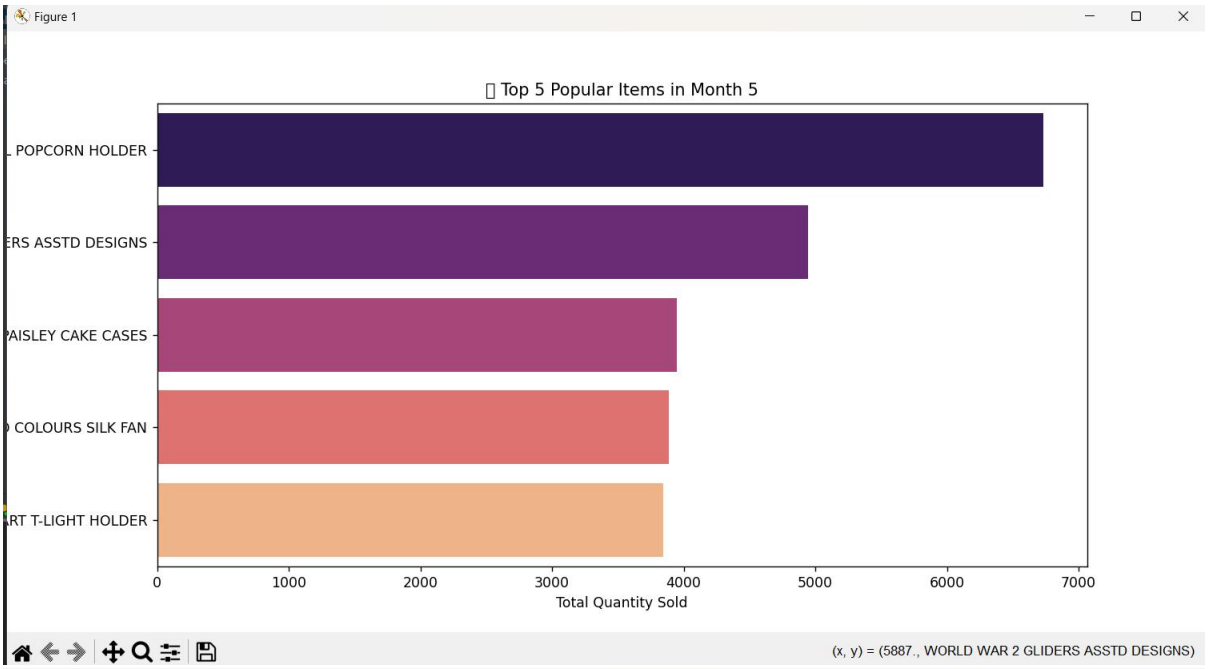


Figure 1
 Top 5 Popular Items in Month 5
 POPCORN HOLDER
ERS ASSTD DESIGNS
PAISLEY CAKE CASES
 COLOURS SILK FAN
ART T-LIGHT HOLDER
0
1000
2000
3000
4000
5000
6000
7000
Total Quantity Sold
(x, y) = (5887., WORLD WAR 2 GLIDERS ASSTD DESIGNS)

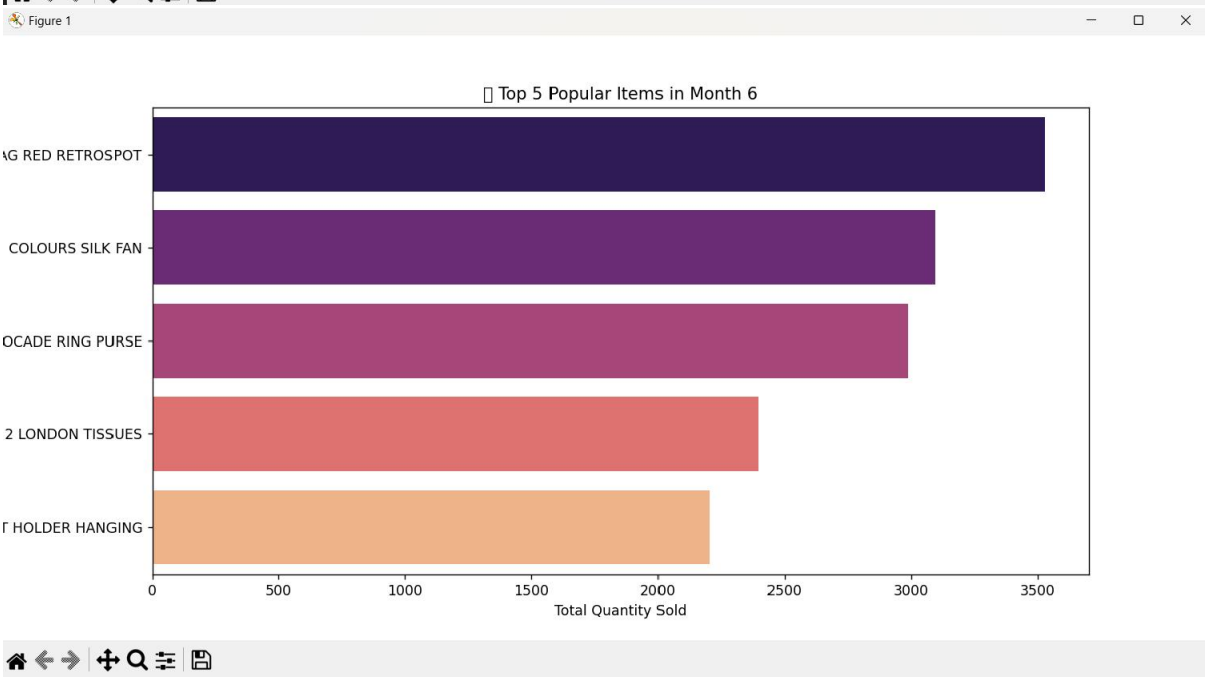


Figure 1
 Top 5 Popular Items in Month 6
AG RED RETROSPOT
COLOURS SILK FAN
OCADE RING PURSE
2 LONDON TISSUES
T HOLDER HANGING
0
500
1000
1500
2000
2500
3000
3500
Total Quantity Sold

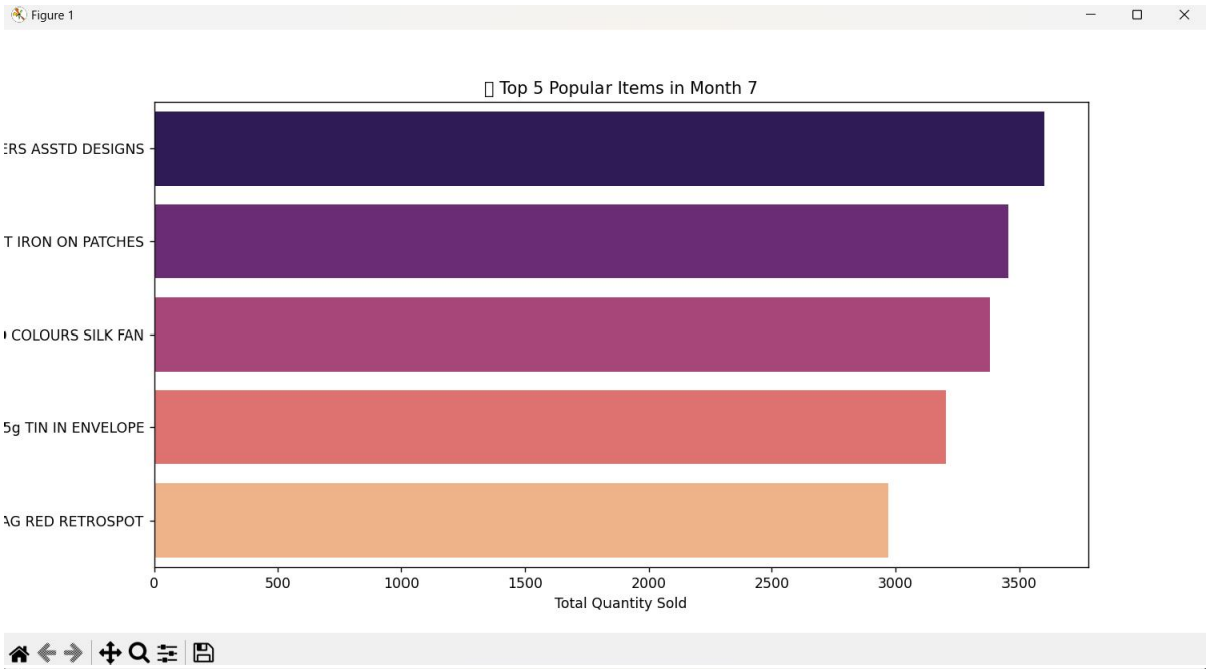


Figure 1
 Top 5 Popular Items in Month 7
ERS ASSTD DESIGNS
T IRON ON PATCHES
COLOURS SILK FAN
5g TIN IN ENVELOPE
AG RED RETROSPOT
0
500
1000
1500
2000
2500
3000
3500
Total Quantity Sold

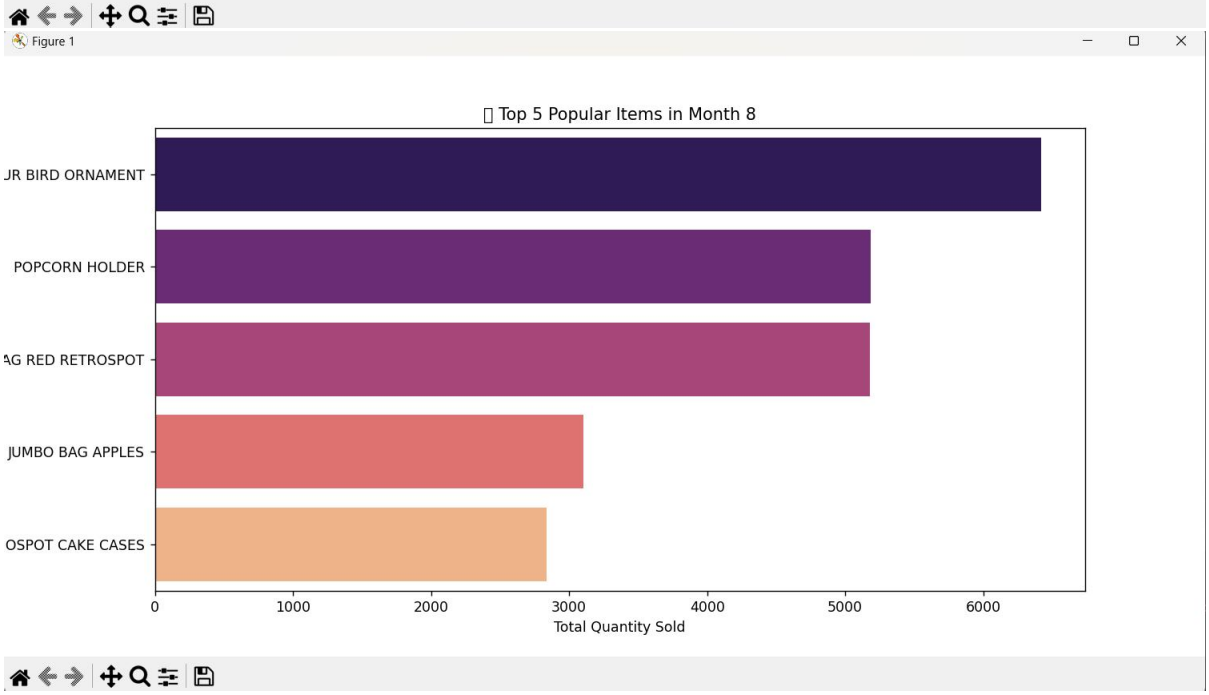


Figure 1
 Top 5 Popular Items in Month 8
JR BIRD ORNAMENT
POPCORN HOLDER
AG RED RETROSPOT
JUMBO BAG APPLES
OSPOT CAKE CASES
0
1000
2000
3000
4000
5000
6000
Total Quantity Sold

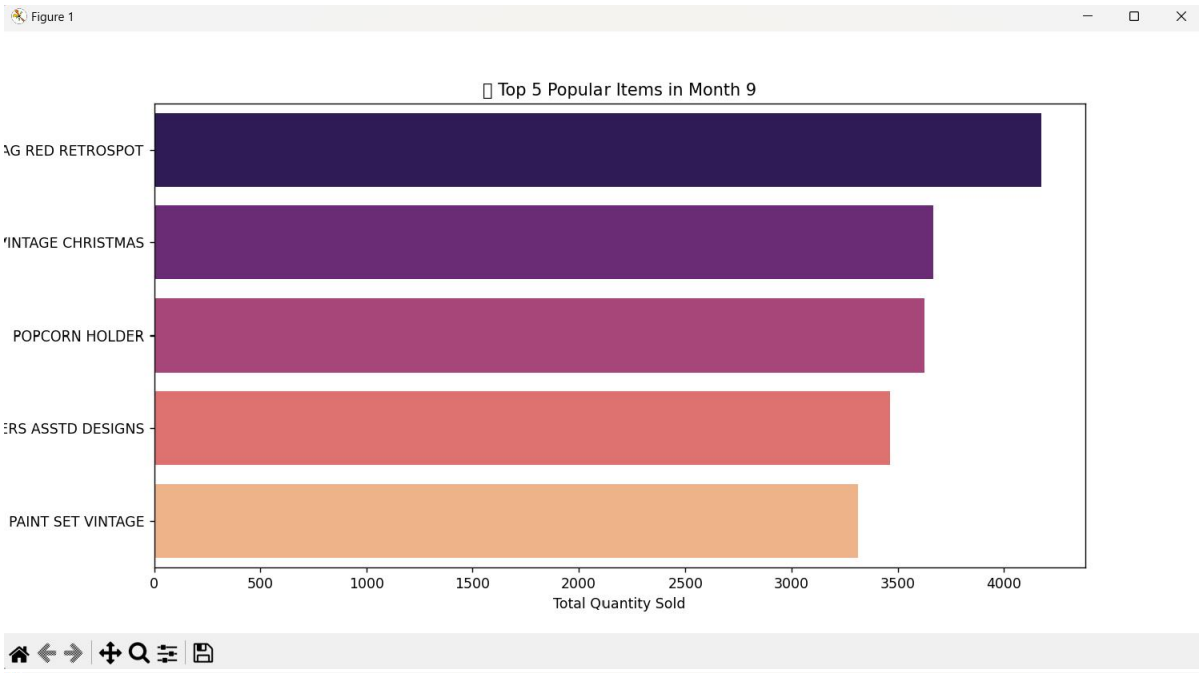


Figure 1
 Top 5 Popular Items in Month 9
AG RED RETROSPOT
VINTAGE CHRISTMAS
POPCORN HOLDER
ERS ASSTD DESIGNS
PAINT SET VINTAGE
0
500
1000
1500
2000
2500
3000
3500
4000
Total Quantity Sold

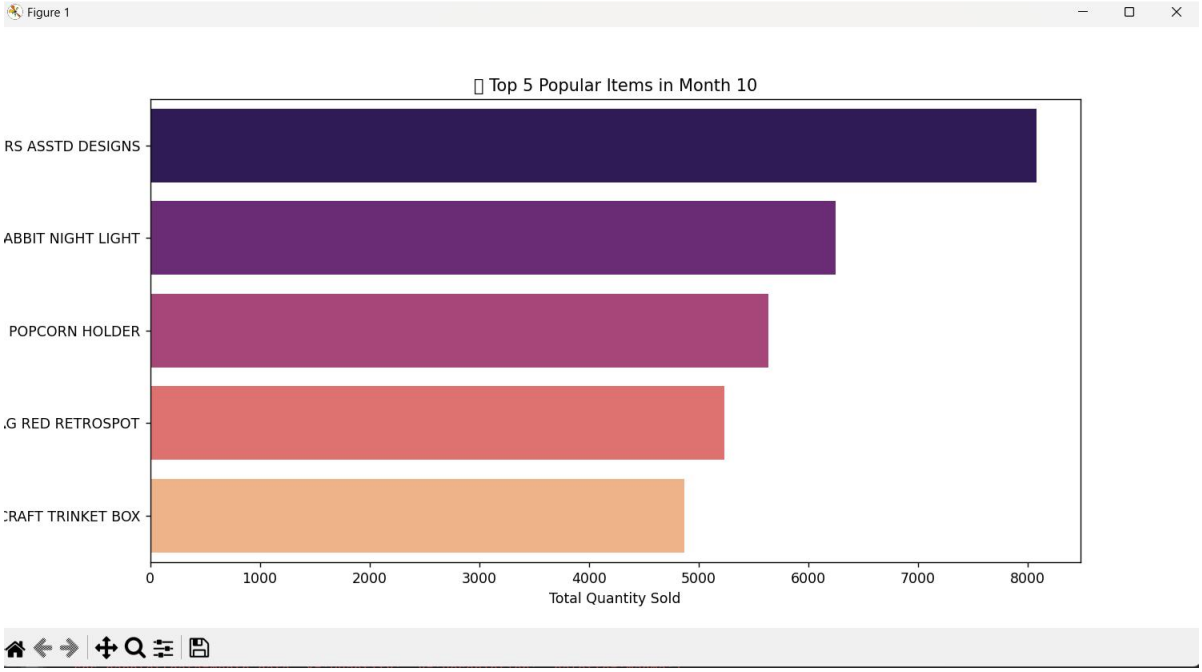


Figure 1
 Top 5 Popular Items in Month 10
RS ASSTD DESIGNS
ABBIT NIGHT LIGHT
POPCORN HOLDER
G RED RETROSPOT
CRAFT TRINKET BOX
0
1000
2000
3000
4000
5000
6000
7000
8000
Total Quantity Sold

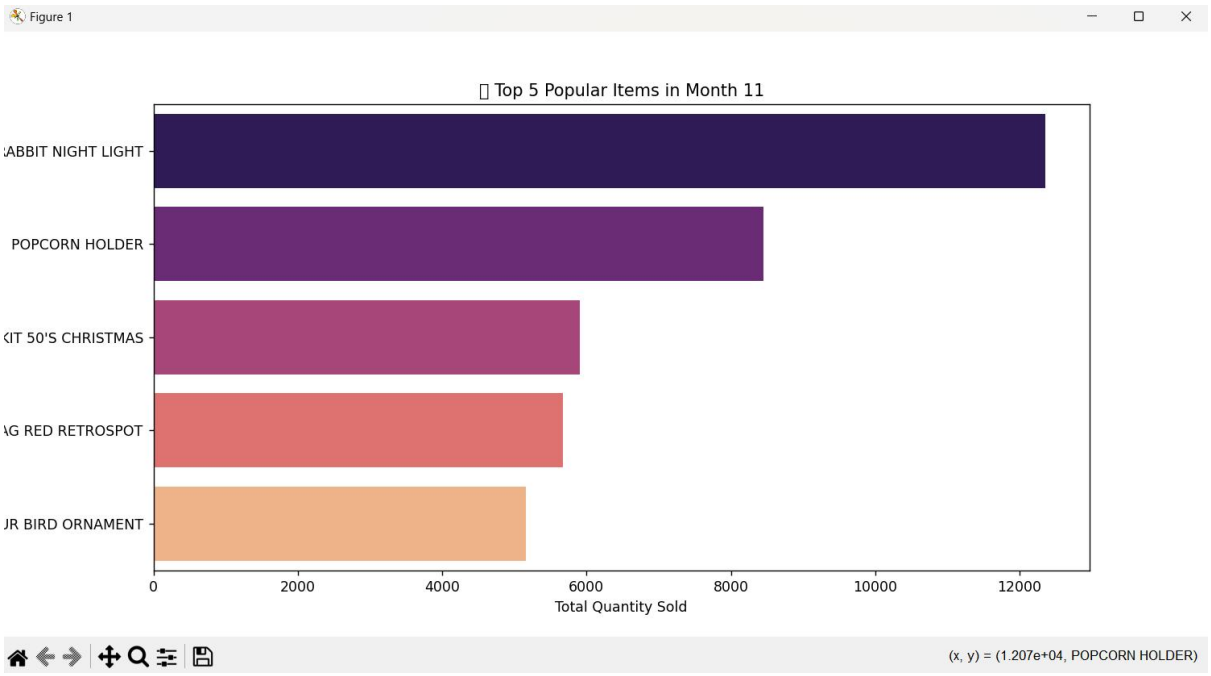


Figure 1
 Top 5 Popular Items in Month 11
ABBIT NIGHT LIGHT
POPCORN HOLDER
IT 50'S CHRISTMAS
G RED RETROSPOT
JR BIRD ORNAMENT
0
2000
4000
6000
8000
10000
12000
Total Quantity Sold
(x, y) = (1.207e+04, POPCORN HOLDER)

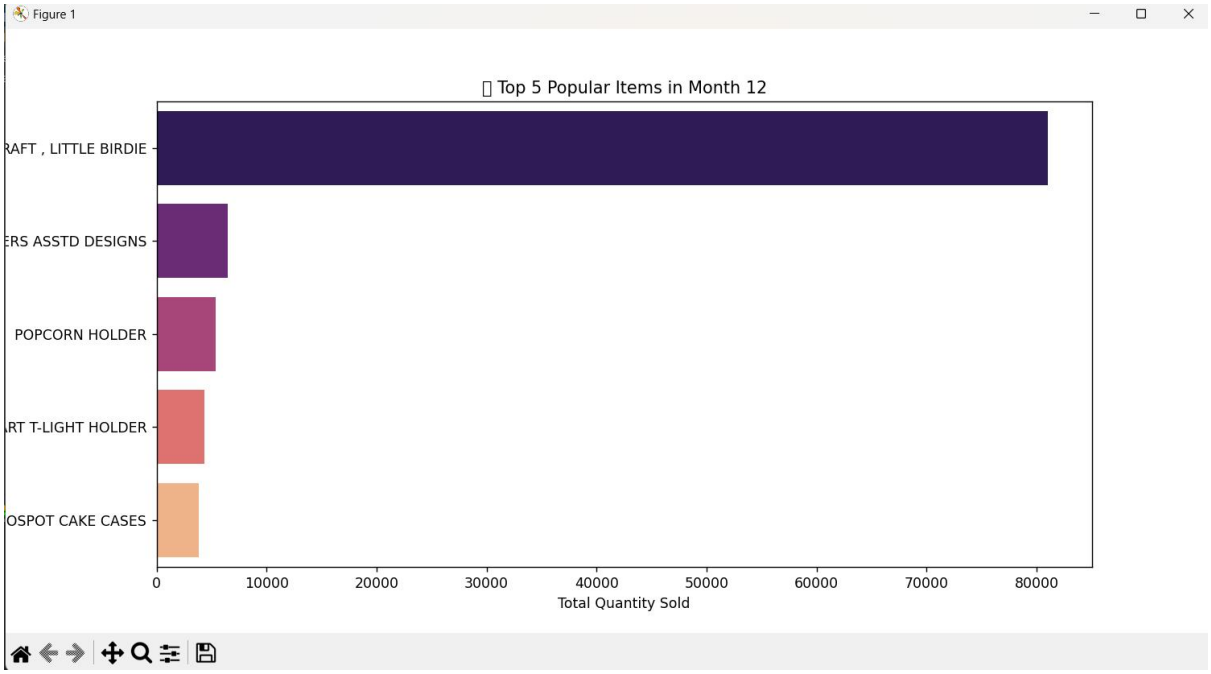


Figure 1
 Top 5 Popular Items in Month 12
RAFT , LITTLE BIRDIE
ERS ASSTD DESIGNS
POPCORN HOLDER
RT T-LIGHT HOLDER
OSPOT CAKE CASES
0
10000
20000
30000
40000
50000
60000
70000
80000
Total Quantity Sold

Appendix H: Business Insights & Reporting

Now that we have completed the **data visualization**, we need to **analyze the results** and extract meaningful **business insights** from our findings.

Steps in Step 8

1. **Summarize Key Findings**
    - Which products are the most popular globally?
    - Are there differences in popular products across different countries?
    - Are there seasonal trends in product popularity?
2. **Write a Business Report**
    - Provide insights based on the analysis.
    - Suggest potential business strategies.

Business Report Format

Your report should include the following sections:

1 Executive Summary

- Briefly describe the objective of the analysis.
- Summarize the key findings in simple terms.

2 Global Sales Insights

- Which products are the best-sellers?
- What patterns do we observe in global sales?

3 Country-wise Sales Insights

- Which countries buy the most products?
- Are there regional differences in customer preferences?

4 Monthly Trends

- Are there peak sales months?
- Do certain products sell more in specific months?

5 Business Recommendations

- Should we stock more of certain items?
- Are there opportunities for marketing in specific months or countries?

Input:

```
report = """
with open("Business_Insights_Report.txt", "w") as file:
    file.write(report)
print(" Business Report Generated Successfully: 'Business_Insights_Report.txt'")
```

Output:

```
Business Report Generated Successfully: 'Business_Insights_Report.txt'
```