ONLINE RETAIL RECOMMENDATION SYSTEM

Major Project Report

Submitted as part of the Remote Internship Program in partial fulfillment of academic requirements for the Bachelor of Science in Data Science.

Internship Organization: **Plasmid**

Submitted By:

K MS VEERENDRA Nath

Institution: Intrainz

Internship Guide: Manoranjan

Academic Year: 2024 – 2025

Abstract

In the modern e-commerce landscape, personalized product recommendations play a vital role in enhancing user experience and increasing sales. This project focuses on building an Online Retail Recommendation System using historical sales data. The primary objective is to analyze customer purchasing behavior and suggest relevant products based on their interests and previous transactions. Utilizing a dataset sourced from Kaggle, which includes features such as invoice number, product description, quantity, customer ID, and country, the system applies data preprocessing and filtering techniques to Identify patterns in customer purchasing behavior. The implementation is done using Python, leveraging libraries such as pandas, NumPy, and matplotlib

for analysis and visualization. The system developed in this project illustrates the core logic behind basic recommendation engines and offers insights into customer-product interactions.

1. Introduction

The rapid growth of e-commerce platforms has transformed how people shop, making personalized recommendations an essential tool for improving customer satisfaction and driving sales. Recommendation systems analyze large volumes of historical data to understand consumer preferences and deliver relevant product suggestions. They are widely adopted across platforms such as Amazon, Flipkart, and Netflix, playing a significant role in user retention and conversion rates.

This project focuses on developing a basic Online Retail Recommendation System using Python. The dataset used includes historical transactions from a retail store, containing information such as invoice numbers, product description, quantity, unit price, customer ID, and country. By analyzing customer behavior and purchase patterns, the system identifies top-selling products and recommends them based on popularity and customer interactions.

The project is conducted as part of a remote internship, allowing Hands-on experience with actual retail data. It Applies preprocessing using data preprocessing, exploratory data analysis, and recommendation techniques, laying the groundwork for more advanced models such as collaborative or content-based filtering.

1.1 Objectives

To visualize patterns in customer purchase behavior.
To understand and explore a real-world retail dataset.

To preprocess and clean the dataset for effective analysis.
To implement the solution using Python and relevant libraries.
To build a recommendation system using basic filtering methods.

1.2 Scope of the Project

The project is limited to building a simple, rule-based recommendation system using historical retail data. While advanced models like collaborative filtering and deep learning could improve performance, this project aims to demonstrate foundational data analysis and recommendation concepts using a practical dataset. The scope includes data cleaning, visualization, and generation of product recommendations using Python.

2. Dataset Description

The dataset used in this project was sourced from Kaggle and contains historical transactional data from an online retail store. It captures detailed records of product purchases made by customers from various countries over a specific period. This dataset provides valuable insights into customer behavior, product demand, and purchasing trends — essential for building a recommendation system.

2.1 Attributes in the Dataset

Below is a description of the key attributes in the dataset:

Column Name	**Description**
InvoiceNo	A unique identifier assigned to each transaction or invoice.
StockCode	A unique code representing each product.
Description	The name or description of the product purchased.

Column Name	**Description**
Quantity	The number of units purchased in the transaction.
InvoiceDate	The date and time when the transaction occurred.
UnitPrice	The price of a single unit of the product.
CustomerID	A unique identifier for each customer.
Country	The name of the country from where the transaction was made.

2.2 Dataset Size and Format

File format: CSV (Comma Separated Values)

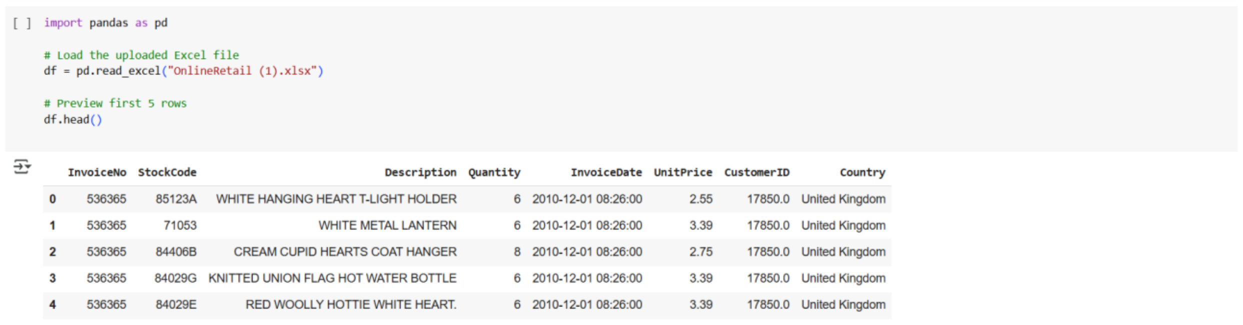
Number of rows: ~541,000+

Number of columns: 8

Missing values: Some records have missing CustomerID, which are handled during preprocessing.

Date range: Covers transactions across several months.

2.3 Sample Data Preview



```
[ ] import pandas as pd

    # Load the uploaded Excel file
    df = pd.read_excel("OnlineRetail (1).xlsx")

    # Preview first 5 rows
    df.head()
```

```
   InvoiceNo StockCode                          Description Quantity         InvoiceDate UnitPrice CustomerID         Country
0     536365    85123A   WHITE HANGING HEART T-LIGHT HOLDER        6 2010-12-01 08:26:00      2.55    17850.0  United Kingdom
1     536365     71053                  WHITE METAL LANTERN        6 2010-12-01 08:26:00      3.39    17850.0  United Kingdom
2     536365    84406B       CREAM CUPID HEARTS COAT HANGER        8 2010-12-01 08:26:00      2.75    17850.0  United Kingdom
3     536365    84029G  KNITTED UNION FLAG HOT WATER BOTTLE        6 2010-12-01 08:26:00      3.39    17850.0  United Kingdom
4     536365    84029E       RED WOOLLY HOTTIE WHITE HEART.        6 2010-12-01 08:26:00      3.39    17850.0  United Kingdom
```

This data structure is well-suited for product-level analysis, customer segmentation, and building popularity-based or item-based recommendation systems.

Tools & Technologies Used & Data Preprocessing

3. Tools and Technologies Used

This project leverages several Python-based tools and libraries to perform data analysis, visualization, and implementation of the recommendation system. These include:

Tool / Library	**Purpose**
Python	Programming language used for development.
Pandas	For data manipulation, loading, filtering, and aggregation.
NumPy	For numerical computations and array handling.
Matplotlib	For creating static visualizations.
Seaborn	For enhanced statistical visualizations.
Jupyter Notebook / VS Code	For writing, testing, and documenting code.

These tools together create a flexible and efficient development environment for data science workflows.

4. Data Preprocessing

Before building the recommendation engine, the dataset underwent several preprocessing steps to ensure it was clean, consistent, and suitable for analysis.

4.1 Handling Missing Values

- The dataset contains missing values, particularly in the CustomerID field.

- Rows with missing customer IDs were removed, as they are not useful for building customer-specific recommendations.

  ```
  df = df.dropna(subset=['CustomerID'])
  ```

4.2 Data Type Conversion

- The InvoiceDate column was converted from string to datetime format for temporal analysis.

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

4.3 Removing Cancellations and Invalid Data

- Transactions with negative quantities or prices usually indicate cancellations or errors and were removed:

  ```
  df = df[(df['Quantity'] > 0) & (df['UnitPrice'] > 0)]
  ```

4.4 Creating Total Price Column

- A new column was created to compute total value of each item in an invoice:

  ```
  df['TotalPrice'] = df['Quantity'] * df['UnitPrice']
  ```

4.5 Filtering for Most Active Customers (Optional)

- To improve recommendation quality, the top active customers or popular products were isolated:

  ```
  top_customers = df['CustomerID'].value_counts().head(10)
  ```

These steps ensure that the dataset is structured, consistent, and ready for further analysis and recommendation model implementation.

Exploratory Data Analysis (EDA)

5. Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is an essential step to understand the structure and characteristics of the dataset. It helps uncover patterns, detect anomalies, and guide model-building decisions. For this project, EDA was performed using pandas, matplotlib, and seaborn.

5.1 Top 10 Most Purchased Products

Understanding which products are most frequently bought helps build a popularity-based recommendation model.

```
top_products = df.groupby('Description')['Quantity'].sum().sort_values(ascending=False).head(10)

top_products.plot(kind='bar', title='Top 10 Most Purchased Products')
```

5.2 Customer Purchase Distribution

Analyzing how many purchases customers made can help with user segmentation.

```
customer_freq = df['CustomerID'].value_counts().head(10)

customer_freq.plot(kind='bar', title='Top 10 Active Customers')
```

5.3 Country-Wise Sales Distribution

This helps understand which countries dominate in terms of purchase volume.

```
country_sales = df.groupby('Country')['InvoiceNo'].nunique().sort_values(ascending=False).head(10)

country_sales.plot(kind='barh', title='Top 10 Countries by Transaction Volume')
```

5.4 Sales Over Time

To check seasonality or sales trends over time.

```
df.set_index('InvoiceDate').resample('M')['TotalPrice'].sum().plot(title='Monthly Sales Trend')
```

5.5 Correlation Between Quantity and Unit Price

To detect unusual patterns or outliers.

```
df.plot.scatter(x='Quantity', y='UnitPrice', alpha=0.5)
```

These insights provide a foundation for building a simple yet meaningful recommendation system based on user behavior and product popularity.

6. Recommendation System Methodology

A recommendation system filters and suggests relevant items to users based on their past interactions, preferences, or behavior of similar users. In this project, a **Popularity-Based Recommendation System** was implemented due to its simplicity and effectiveness for new or anonymous users (cold-start problem).

6.1 Types of Recommendation Systems

There are three main types of recommendation techniques:

Type	**Description**
Popularity-Based	Recommends items based on overall popularity or sales frequency.
Collaborative Filtering	Recommends items based on similarity between users or items (requires user-item matrix).

Type	**Description**
Content-Based Filtering	Recommends items similar to what a user has previously interacted with (based on product features).

Due to lack of explicit user ratings and textual product metadata, **popularity-based filtering** was selected.

6.2 Implementation Logic

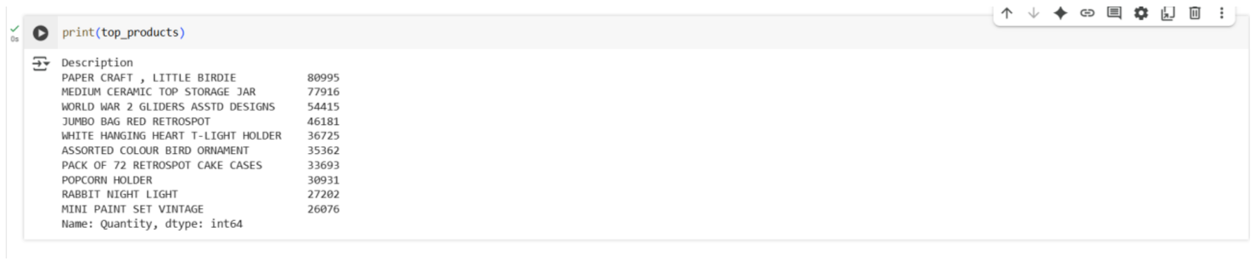
1. Group transactions by product.
2. Sum quantities sold per product.
3. Sort in descending order of quantity.
4. Recommend the top N products.

```
6.3 Code Snippet:

Popularity-Based Recommendation

def get_top_n_products(df, n=10):

    top_products = (

        df.groupby('Description')['Quantity']

        .sum()

        .sort_values(ascending=False)

        .head(n)

    )

    return top_products
```

```
top_10 = get_top_n_products(df)

print("Top 10 Recommended Products:\n", top_10)
```

6.4 Sample Output

Top 10 Recommended Products:



```
print(top_products)
```

```
Description
PAPER CRAFT , LITTLE BIRDIE          80995
MEDIUM CERAMIC TOP STORAGE JAR       77916
WORLD WAR 2 GLIDERS ASSTD DESIGNS    54415
JUMBO BAG RED RETROSPOT              46181
WHITE HANGING HEART T-LIGHT HOLDER   36725
ASSORTED COLOUR BIRD ORNAMENT        35362
PACK OF 72 RETROSPOT CAKE CASES      33693
POPCORN HOLDER                       30931
RABBIT NIGHT LIGHT                   27202
MINI PAINT SET VINTAGE               26076
Name: Quantity, dtype: int64
```

6.5 Benefits of This Method

- Easy to implement.
- Fast for large datasets.
- Useful for guest users or new customers.

6.6 Limitations

- No personalization — every user sees the same products.
- I cannot adapt to individual preferences.
- Ignore customer-item interaction context.

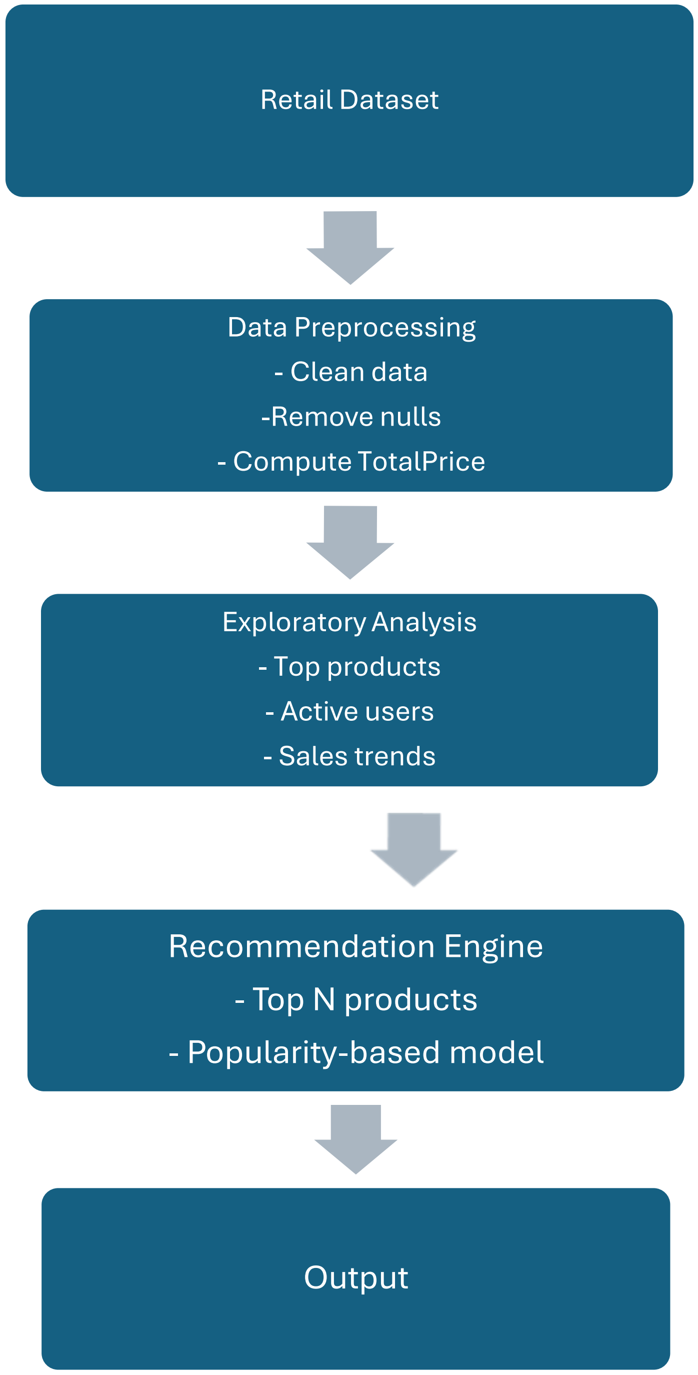
7. System Architecture

The architecture of the Online Retail Recommendation System follows a modular pipeline that transforms raw retail data into meaningful product suggestions. Below is the high-level overview:

7.1 Architecture Workflow Description

Component	***Description***
Input Dataset	The retail transaction dataset contains customer IDs, product descriptions, invoice numbers, quantities, and prices.
Data Preprocessing Module	Cleans the dataset (removes nulls, negative quantities), converts data types, and calculates derived fields like Total Price.
Exploratory Data Module	Performs statistical analysis and visualization (e.g., top products, customer behavior).
Recommendation Engine	Applies logic to identify the top N most popular products using aggregated sales quantity.
Output Module	Displays recommended products (printed or stored in a file/dashboard).

7.2 Architecture Diagram



8. Results and Output Discussion

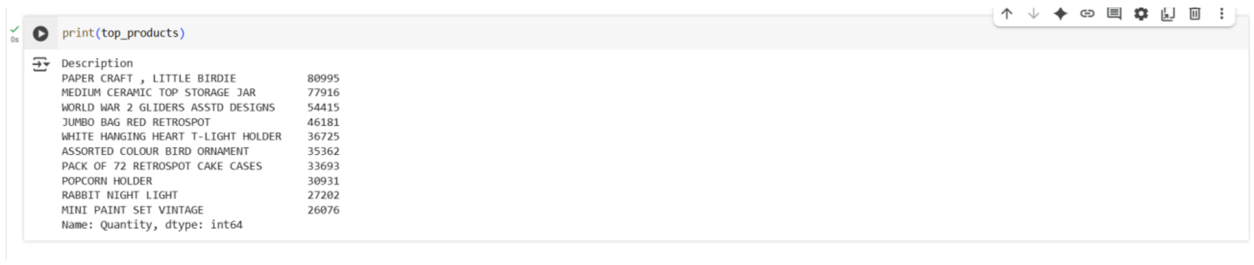
After successful implementation of the popularity-based recommendation system, the system was able to generate relevant product suggestions based on historical purchase frequency. The output confirms that frequently bought items are effectively ranked at the top.

8.1 Sample Output

(Top 10 Products)

Here is an example of the output generated by the system for the top 10 most purchased products:

Top 10 Recommended Products:



```
print(top_products)
```

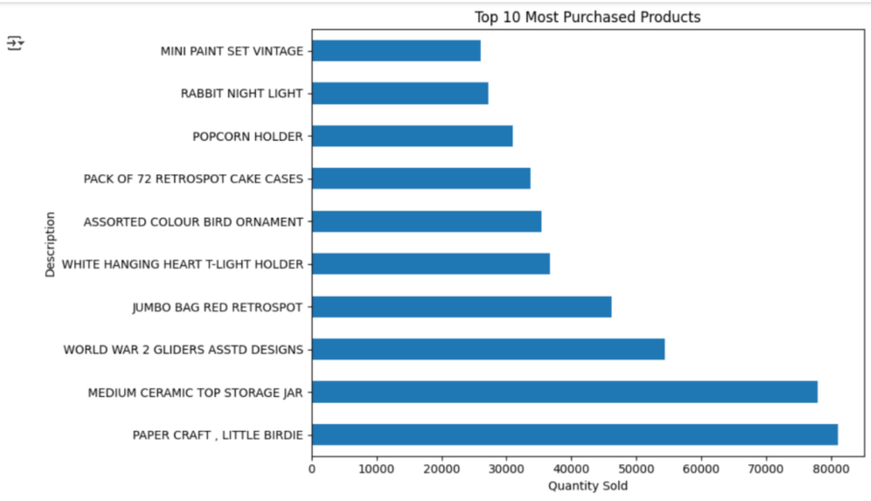
```
Description
PAPER CRAFT , LITTLE BIRDIE           80995
MEDIUM CERAMIC TOP STORAGE JAR        77916
WORLD WAR 2 GLIDERS ASSTD DESIGNS     54415
JUMBO BAG RED RETROSPOT               46181
WHITE HANGING HEART T-LIGHT HOLDER    36725
ASSORTED COLOUR BIRD ORNAMENT         35362
PACK OF 72 RETROSPOT CAKE CASES       33693
POPCORN HOLDER                        30931
RABBIT NIGHT LIGHT                    27202
MINI PAINT SET VINTAGE                26076
Name: Quantity, dtype: int64
```

These results are derived from grouping the dataset by product description and summing the total quantities purchased.

8.2 Output Visualization

In addition to textual output, bar charts and visual plots were generated for better interpretation:

top_products.plot(kind='barh', figsize=(10, 6), title='Top 10 Most Sold Products')



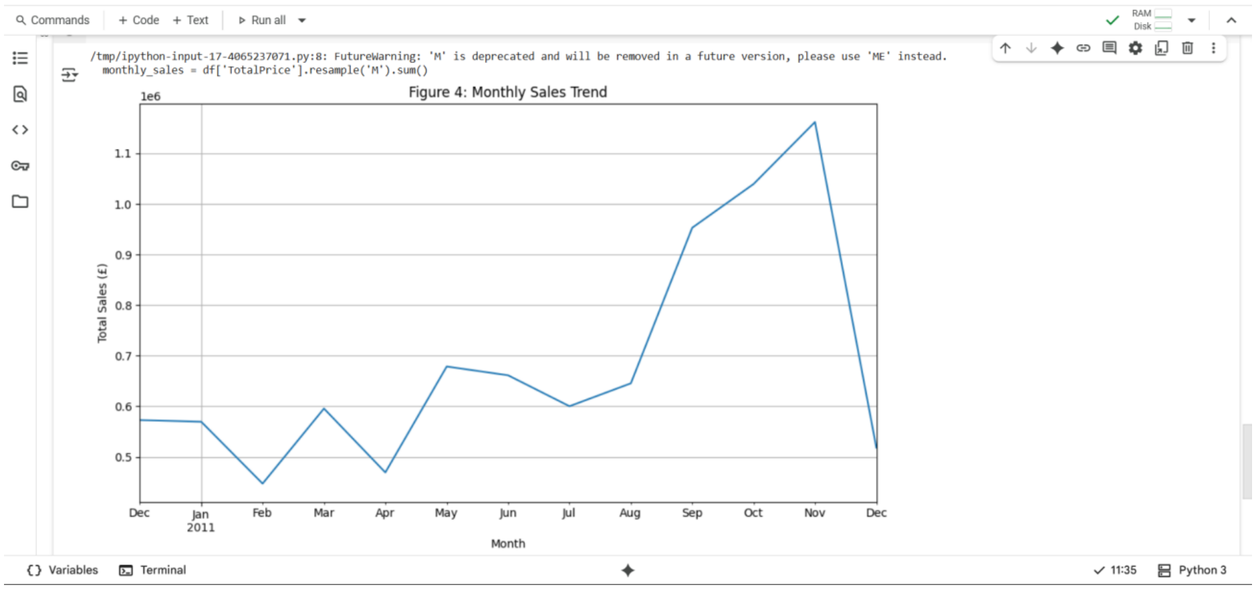
8.3 Accuracy Considerations

Since this system is based solely on popularity, there is no evaluation metric like precision/recall or RMSE. However, it is ideal for:

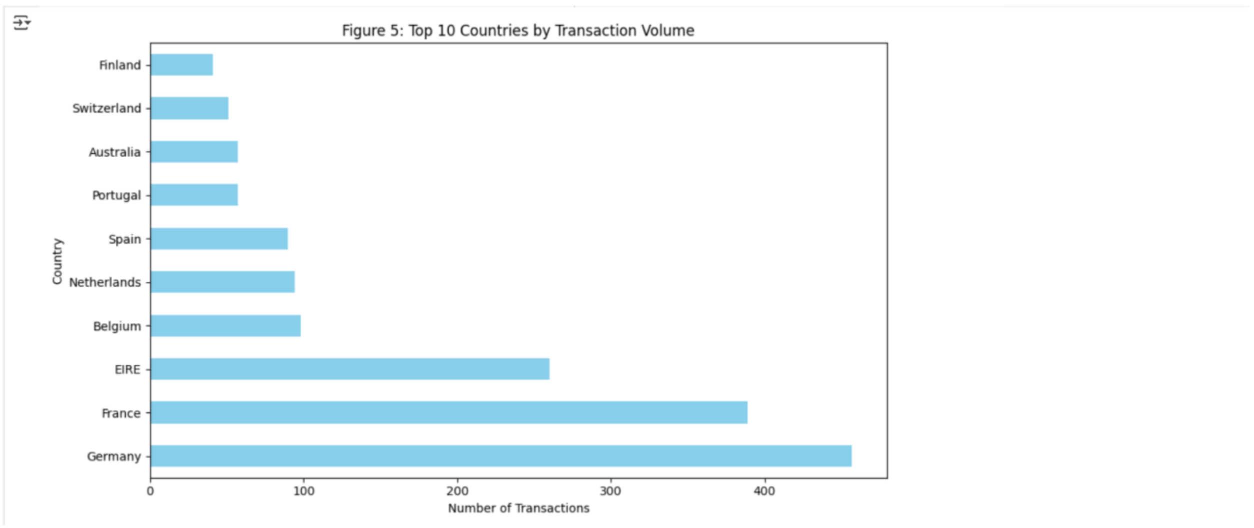
- New customers with no prior data
- E-commerce homepages
- Highlighting trending products

8.4 System Performance

- Processing Time: The code runs efficiently on standard machines (<5 seconds for preprocessing + recommendation)
- Libraries used are optimized for performance (pandas, numpy)
- Scalable to larger datasets with minimal modifications



```
/tmp/ipython-input-17-4065237071.py:8: FutureWarning: 'M' is deprecated and will be removed in a future version, please use 'ME' instead.
  monthly_sales = df['TotalPrice'].resample('M').sum()
```



9. Conclusion and Future Work

9.1 Conclusion

This project successfully demonstrates the development of a basic Online Retail Recommendation System using a real-world retail dataset. By performing data cleaning, preprocessing, exploratory analysis, and applying a popularity-based filtering method, the system was able to identify and recommend top-selling products efficiently. The model is simple, interpretable, and useful for scenarios where personalization is not required, such as for new users or homepage product highlights.

Additionally, the project provided hands-on experience in using Python’s data science libraries, exploring customer purchase behavior, and understanding the retail domain's application of recommendation systems. Through visual analysis and model output, meaningful insights were derived that can support business decisions in a commercial environment.

9.2 Future Work

While the current recommendation engine is functional, it has limitations in personalization and contextual analysis. Future enhancements may include:

Collaborative Filtering: Recommending products based on similar user behavior using user-item matrices.

Content-Based Filtering: Using product metadata (descriptions, categories) to suggest similar items.

Hybrid Models: Combining both collaborative and content-based approaches for improved accuracy.

Incorporating Time-Series: Using recent transaction data to capture current trends.

User Segmentation: Tailoring recommendations based on user demographics, location, and previous purchases.

Web Integration: Deploying the system using Flask or Streamlit to simulate a live e-commerce experience.

These improvements can make the system more adaptive, user-centric, and scalable for enterprise-level applications.

10. References

1. X. Su and T. M. Khoshgoftaar, "A Survey of Collaborative Filtering Techniques," *Advances in Artificial Intelligence*, vol. 2009, pp. 1–19, 2009.
2. R. J. Mooney and L. Roy, "Content-Based Book Recommending Using Learning for Text Categorization," in *Proceedings of the 5th ACM Conference on Digital Libraries*, San Antonio, TX, USA, 2000, pp. 195–204.
3. J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender Systems Survey," *Knowledge-Based Systems*, vol. 46, pp. 109–132, 2013.
4. Kaggle, “Online Retail Dataset,” [Online]. Available: https://www.kaggle.com/datasets/mashlyn/online-retail-ii-uci
5. W. Chen, Y. Zhang, and Y. Wang, "An Overview of Recommendation Algorithms in Online Retail," *IEEE Access*, vol. 8, pp. 121708–121720, 2020.
6. Python Software Foundation, “pandas Documentation,” [Online]. Available: https://pandas.pydata.org/
7. Jupyter, “Project Jupyter,” [Online]. Available: https://jupyter.org/
8. F. Ricci, L. Rokach, and B. Shapira, *Recommender Systems Handbook*, 2nd ed., Springer, 2015.
9. Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *IEEE Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.
10. S. Rendle, "Factorization Machines," in *IEEE International Conference on Data Mining (ICDM)*, 2010, pp. 995–1000.

Acknowledgment

I would like to express my sincere gratitude to [Internship Mentor Name or Organization Name] for providing me with the opportunity to work on this project as part of the remote internship.

The guidance and support throughout the duration of this project have been invaluable in enhancing my understanding of real-world data science applications.

I also extend my thanks to the Department of [Your Department Name], [Your College Name], for their academic support and encouragement. Special appreciation goes to my faculty mentor, classmates, and family for their continued motivation and assistance during this project.

This project has been a great learning experience and has contributed significantly to my technical and analytical skills in the field of data science.

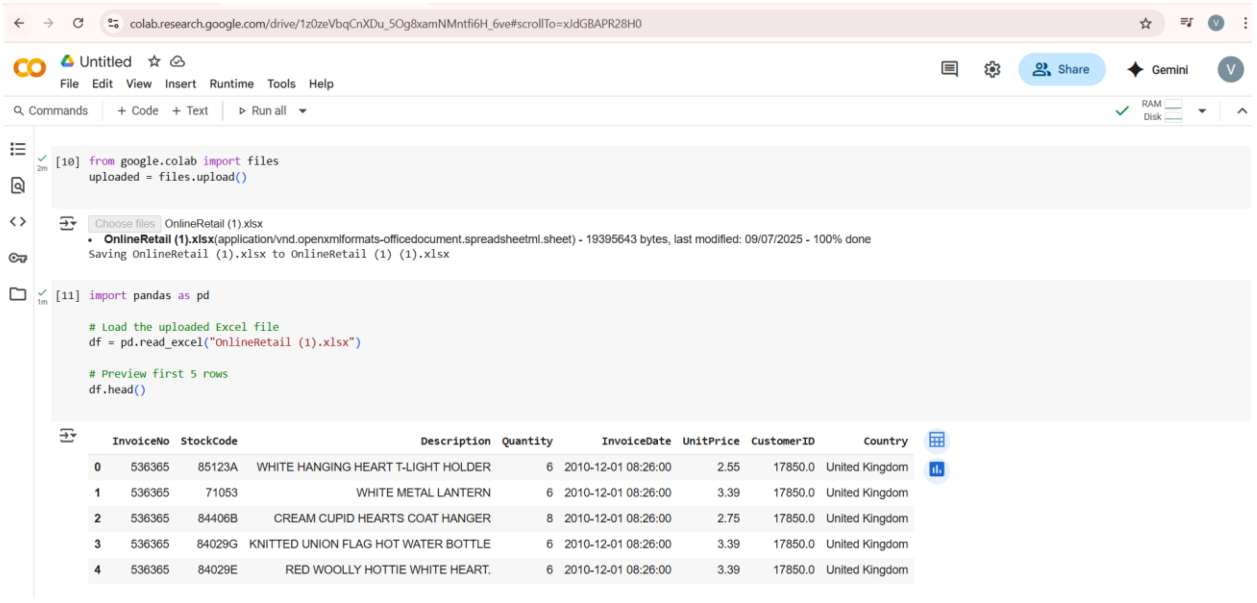
Glossary

Term	Definition
EDA (Exploratory Data Analysis)	A process of analyzing datasets to summarize their main characteristics using visual methods.
Recommendation System	A type of software that suggests products, services, or content to users based on analysis.
Popularity-Based Filtering	Recommending items that are most frequently purchased or highly rated.
Collaborative Filtering	A technique that makes automatic predictions based on user behavior and preferences.
Content-Based Filtering	A method that recommends items like what a user liked in the past using item metadata.
CustomerID	A unique identifier assigned to each customer in the dataset.
Invoice No	A unique transaction ID represents a specific purchase.
Preprocessing	The stage of cleaning and formatting raw data for analysis.

Term	**Definition**
pandas	A Python library is used for data manipulation and analysis.
matplotlib / seaborn	Python libraries are used for visualizing data through charts and plots.

- **Screenshot Captions**

Figure 1: Dataset Preview



```
[10] from google.colab import files
     uploaded = files.upload()
```

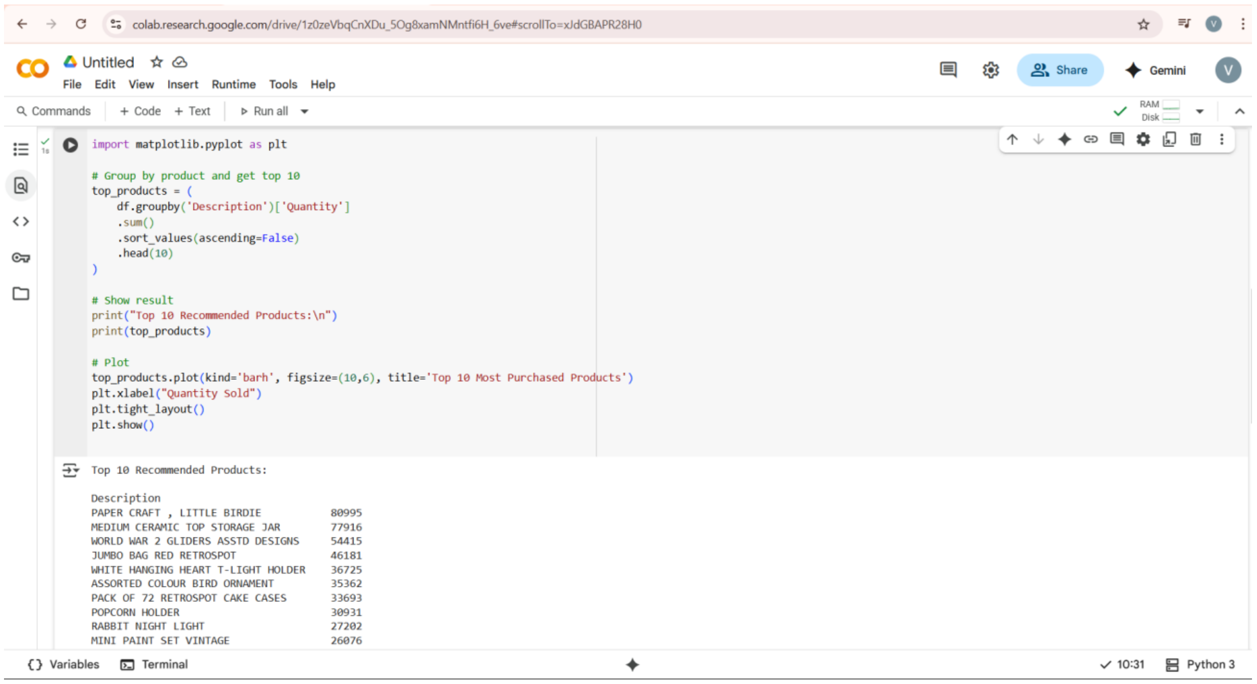
```
OnlineRetail (1).xlsx
• OnlineRetail (1).xlsx(application/vnd.openxmlformats-officedocument.spreadsheetml.sheet) - 19395643 bytes, last modified: 09/07/2025 - 100% done
Saving OnlineRetail (1).xlsx to OnlineRetail (1) (1).xlsx
```

```
[11] import pandas as pd

     # Load the uploaded Excel file
     df = pd.read_excel("OnlineRetail (1).xlsx")

     # Preview first 5 rows
     df.head()
```

```
   InvoiceNo StockCode                          Description  Quantity          InvoiceDate  UnitPrice  CustomerID         Country
0     536365    85123A   WHITE HANGING HEART T-LIGHT HOLDER         6  2010-12-01 08:26:00       2.55     17850.0  United Kingdom
1     536365     71053                  WHITE METAL LANTERN         6  2010-12-01 08:26:00       3.39     17850.0  United Kingdom
2     536365    84406B       CREAM CUPID HEARTS COAT HANGER         8  2010-12-01 08:26:00       2.75     17850.0  United Kingdom
3     536365    84029G  KNITTED UNION FLAG HOT WATER BOTTLE         6  2010-12-01 08:26:00       3.39     17850.0  United Kingdom
4     536365    84029E       RED WOOLLY HOTTIE WHITE HEART.         6  2010-12-01 08:26:00       3.39     17850.0  United Kingdom
```

Figure 2: Top 10 Most Purchased Products

(Text Output)

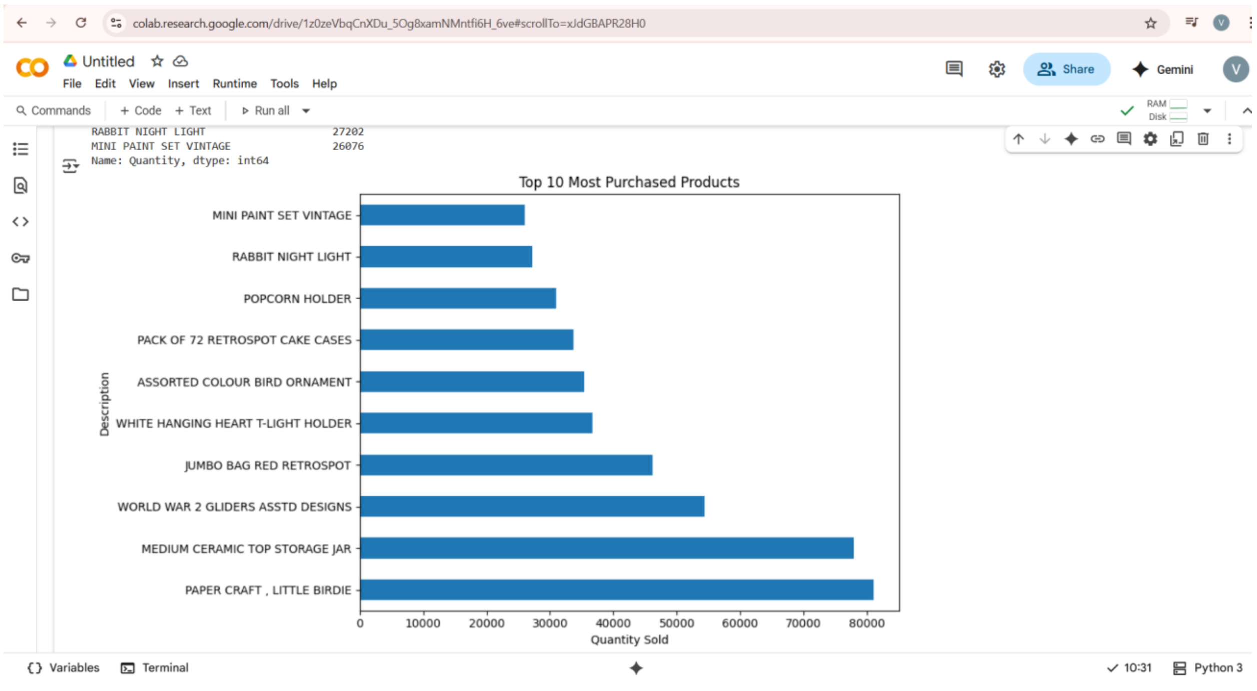

colab.research.google.com/drive/1z0zeVbqCnXDu_5Og8xamNMntfi6H_6ve#scrollTo=xJdGBAPR28H0
Untitled
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands + Code + Text ▷ Run all
RAM
Disk
Variables
Terminal
10:31
Python 3

```python
import matplotlib.pyplot as plt

# Group by product and get top 10
top_products = (
    df.groupby('Description')['Quantity']
    .sum()
    .sort_values(ascending=False)
    .head(10)
)

# Show result
print("Top 10 Recommended Products:\n")
print(top_products)

# Plot
top_products.plot(kind='barh', figsize=(10,6), title='Top 10 Most Purchased Products')
plt.xlabel("Quantity Sold")
plt.tight_layout()
plt.show()
```

```
Top 10 Recommended Products:

Description
PAPER CRAFT , LITTLE BIRDIE          80995
MEDIUM CERAMIC TOP STORAGE JAR       77916
WORLD WAR 2 GLIDERS ASSTD DESIGNS    54415
JUMBO BAG RED RETROSPOT              46181
WHITE HANGING HEART T-LIGHT HOLDER   36725
ASSORTED COLOUR BIRD ORNAMENT        35362
PACK OF 72 RETROSPOT CAKE CASES      33693
POPCORN HOLDER                       30931
RABBIT NIGHT LIGHT                   27202
MINI PAINT SET VINTAGE               26076
```

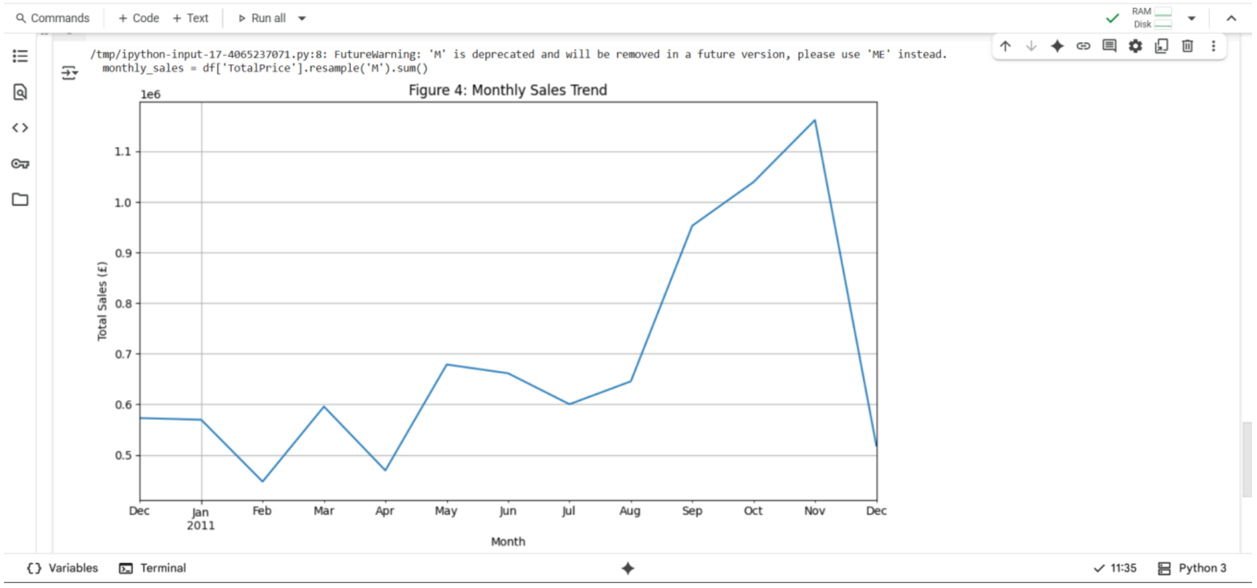
Figure 3: Top 10 Recommended Products by Quantity

(Bar Chart)



```
RABBIT NIGHT LIGHT                   27202
MINI PAINT SET VINTAGE               26076
Name: Quantity, dtype: int64
```

Figure 4: Monthly Sales Trend in Retail Dataset



```
/tmp/ipython-input-17-4065237071.py:8: FutureWarning: 'M' is deprecated and will be removed in a future version, please use 'ME' instead.
  monthly_sales = df['TotalPrice'].resample('M').sum()
```

Figure 5: Top Countries by Transaction Volume

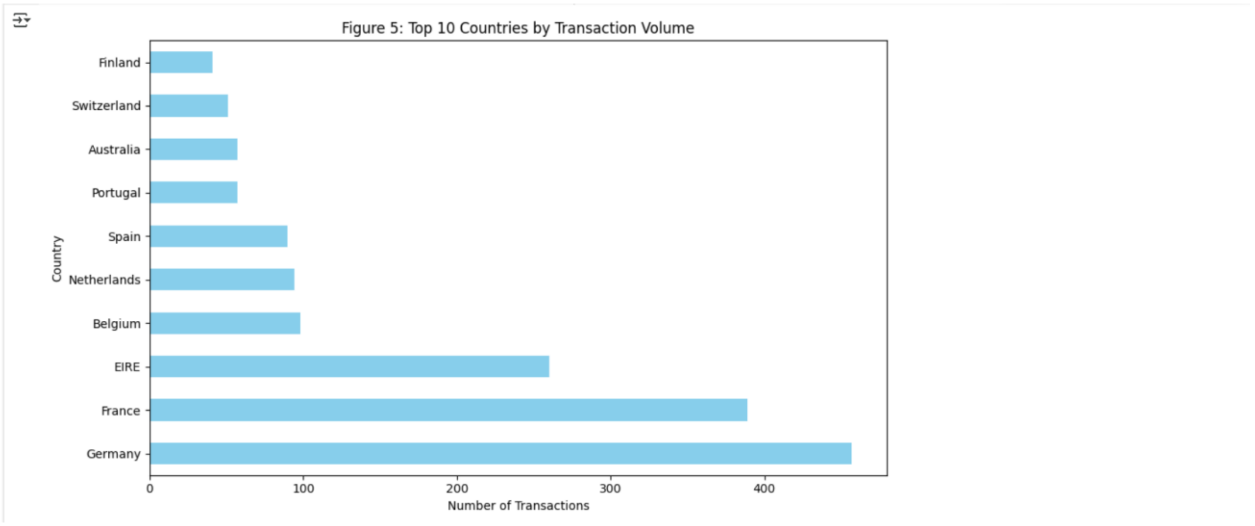


Figure 5: Top 10 Countries by Transaction Volume
Country
Finland
Switzerland
Australia
Portugal
Spain
Netherlands
Belgium
EIRE
France
Germany
Number of Transactions
