CUSTOMER SEGMENTATION USING PYTHON CLUSTERING TECHNIQUES FOR A U.S.-BASED E-COMMERCE PLATFORM

1. Background

A national e-commerce company in the U.S., specializing in lifestyle and fashion products, wanted to move beyond demographic-based marketing and instead segment its user base by actual behavior. They had large volumes of purchase, visit, and browsing data but lacked a clear structure to profile their diverse customer base.

We were engaged to perform data mining using Python, applying clustering algorithms to extract behavior-based customer segments. These segments were then translated into profiles usable by their marketing and product recommendation teams.

2. Objective

- To use unsupervised learning techniques in Python to segment customers based on purchase behavior, recency, frequency, and monetary value
- To generate distinct, interpretable segments with business meaning and marketing utility
- To deliver a report with segment descriptions, visualizations, and suggestions for personalized campaigns
- To provide Python scripts for future model retraining as new customer data becomes available

3. Data Used

Source: Internal customer activity logs (2022–2023)

Dataset Details:

- 82,000 customer records
- Fields included:
    - Customer_ID
    - Last_Purchase_Date
    - Total_Purchase_Value

- Total_Orders
- Average_Order_Value
- Category_Engagement (Fashion, Home, Electronics, etc.)
- Visit_Frequency_Per_Month
- Return_Rate

4. Methodology

4.1 Data Preprocessing

- Removed inactive customers (no purchase in last 12 months)
- Standardized numeric variables using Min-Max scaling
- Encoded categorical features like Category_Engagement using one-hot encoding
- Created RFM (Recency, Frequency, Monetary) features as clustering inputs

4.2 Clustering Techniques

- Applied **K-Means Clustering** with optimal k found via Elbow Method and Silhouette Score
- Ran **Hierarchical Clustering** to validate group stability and interpretability
- Performed **PCA** to reduce dimensionality for visual explanation and segment separation

4.3 Tools and Libraries

- Python 3.10
- Libraries: pandas, numpy, scikit-learn, matplotlib, seaborn, scipy

5. Mining Results

Identified **5 meaningful customer clusters**:

Segment Name	**Key Characteristics**
Premium Loyalists	High spenders, low return rate, regular monthly visits
Discount Hunters	High frequency, low spend, heavily coupon-dependent
Infrequent Buyers	Low engagement, 1–2 purchases/year, low AOV

Category-Specific	Loyal to 1–2 product categories, high order volume in niche segments
At-Risk Dormants	Declining visits, no orders in 6+ months, historically average spenders

- **Premium Loyalists** made up 11% of users but contributed 38% of total revenue
- **Discount Hunters** had the highest return rate (24%) and were more responsive to flash sales
- **At-Risk Dormants** showed high potential for reactivation with personalized win-back offers

6. Strategic Insights

- Suggested targeted **email campaigns** for each segment:
  - "VIP Previews" for Premium Loyalists
  - "Flash Coupon Packs" for Discount Hunters
  - "We Miss You" offers for Dormants with high past spend
- Recommended product recommendation engine use segment IDs to vary homepage offers
- Proposed pausing remarketing for Infrequent Buyers with high return rates
- Built segment labels for export into CRM and email marketing platform

7. Reporting Output

- **Python Jupyter Notebook**:
  - Full codebase from preprocessing to clustering to export
  - Segment profile export with CSV and visual segment distribution
- **PDF Report (18 pages)**:
  - Description of each segment with KPI tables
  - Cluster evaluation charts (Silhouette, PCA)
  - Sample outreach campaign ideas per group
- **Excel Sheet**:

    - Master customer table with Segment ID
    - Pivot summaries by segment for quick marketing queries

8. Business Impact

- Within 6 weeks of implementing segment-based campaigns:
    - **Email CTR improved by 19%**
    - **Conversion rate lifted by 14%** in targeted segments
    - **Overall ROI on marketing improved by 26%** compared to the previous quarter
- Python model is now re-run quarterly with updated data to reflect shifting customer behavior