Customer Shopping Behavior Analysis

1. Project Overview

This project analyzes customer shopping behavior using transactional data from 3,900 purchases across various product categories. The goal is to uncover insights into spending patterns, customer segments, product preferences, and subscription behavior to guide strategic business decisions.

2. Problem Statement

A leading retail company seeks to better understand customer shopping behavior to improve sales, customer satisfaction, and long-term loyalty. Changes in purchasing patterns across demographics, product categories, and sales channels have raised the need for data-driven analysis. Factors such as discounts, customer reviews, seasonal trends, and payment preferences significantly influence consumer decisions and repeat purchases. This project aims to analyze consumer shopping data to identify key trends and behavioral patterns. The insights obtained will support improved customer engagement, optimized marketing strategies, and informed product decision-making.

Based on this analysis, the central research question addressed in this project is: *How can the company leverage consumer shopping data to identify purchasing trends, enhance customer engagement, and optimize marketing and product strategies to drive long-term business growth?*

3. Dataset Summary

I. Rows: 3,900 - Columns: 18
II. Key Features: - Customer demographics (Age, Gender, Location, Subscription Status)
III. Purchase details (Item Purchased, Category, Purchase Amount, Season, Size, Color)
IV. Shopping behavior (Discount Applied, Promo Code Used, Previous Purchases, Frequency of Purchases, Review Rating, Shipping Type)
V. Missing Data: 37 values in Review Rating column

4. Exploratory Data Analysis using Python

We began with data preparation and cleaning in Python:

● Data Loading: Imported the dataset using pandas.

● Initial Exploration: Used df.info() to check structure and .describe() for summary statistics

```
[16]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3900 entries, 0 to 3899
Data columns (total 18 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Customer ID             3900 non-null   int64  
 1   Age                     3900 non-null   int64  
 2   Gender                  3900 non-null   object 
 3   Item Purchased          3900 non-null   object 
 4   Category                3900 non-null   object 
 5   Purchase Amount (USD)   3900 non-null   int64  
 6   Location                3900 non-null   object 
 7   Size                    3900 non-null   object 
 8   Color                   3900 non-null   object 
 9   Season                  3900 non-null   object 
 10  Review Rating           3863 non-null   float64
 11  Subscription Status     3900 non-null   object 
 12  Shipping Type           3900 non-null   object 
 13  Discount Applied        3900 non-null   object 
 14  Promo Code Used         3900 non-null   object 
 15  Previous Purchases      3900 non-null   int64  
 16  Payment Method          3900 non-null   object 
 17  Frequency of Purchases  3900 non-null   object 
dtypes: float64(1), int64(4), object(13)
memory usage: 548.6+ KB
```

Dataset Information

```
[6]: df.describe(include="all")
```

[6]:

	Customer ID	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	Location	Size	Color	Season	Review Rating	Subscription Status	Shipping Type	Discount Applied	Promo Code Used
count	3900.000000	3900.000000	3900	3900	3900	3900.000000	3900	3900	3900	3900	3863.000000	3900	3900	3900	3900
unique	NaN	NaN	2	25	4	NaN	50	4	25	4	NaN	2	6	2	2
top	NaN	NaN	Male	Blouse	Clothing	NaN	Montana	M	Olive	Spring	NaN	No	Free Shipping	No	No
freq	NaN	NaN	2652	171	1737	NaN	96	1755	177	999	NaN	2847	675	2223	2223
mean	1950.500000	44.068462	NaN	NaN	NaN	59.764359	NaN	NaN	NaN	NaN	3.750065	NaN	NaN	NaN	NaN
std	1125.977353	15.207589	NaN	NaN	NaN	23.685392	NaN	NaN	NaN	NaN	0.716983	NaN	NaN	NaN	NaN
min	1.000000	18.000000	NaN	NaN	NaN	20.000000	NaN	NaN	NaN	NaN	2.500000	NaN	NaN	NaN	NaN
25%	975.750000	31.000000	NaN	NaN	NaN	39.000000	NaN	NaN	NaN	NaN	3.100000	NaN	NaN	NaN	NaN
50%	1950.500000	44.000000	NaN	NaN	NaN	60.000000	NaN	NaN	NaN	NaN	3.800000	NaN	NaN	NaN	NaN
75%	2925.250000	57.000000	NaN	NaN	NaN	81.000000	NaN	NaN	NaN	NaN	4.400000	NaN	NaN	NaN	NaN
max	3900.000000	70.000000	NaN	NaN	NaN	100.000000	NaN	NaN	NaN	NaN	5.000000	NaN	NaN	NaN	NaN

Dataset Description

- Missing Data Handling:

Checked for null values and imputed missing values in the Review Rating column using the median rating of each product category.

```
[18]: df.isnull().sum()
```

```
[18]: Customer ID                 0
      Age                         0
      Gender                      0
      Item Purchased              0
      Category                    0
      Purchase Amount (USD)       0
      Location                    0
      Size                        0
      Color                       0
      Season                      0
      Review Rating              37
      Subscription Status         0
      Shipping Type               0
      Discount Applied            0
      Promo Code Used             0
      Previous Purchases          0
      Payment Method              0
      Frequency of Purchases      0
      dtype: int64
```

```
[19]: df['Review Rating'] = df.groupby("Category")["Review Rating"].transform(lambda x : x.fillna(x.median()))
```

- Column Standardization:

Renamed columns to snake case for better readability and documentation.

```
[21]: df.columns = df.columns.str.lower()
```

```
[22]: df.columns
```

```
[22]: Index(['customer id', 'age', 'gender', 'item purchased', 'category',
             'purchase amount (usd)', 'location', 'size', 'color', 'season',
             'review rating', 'subscription status', 'shipping type',
             'discount applied', 'promo code used', 'previous purchases',
             'payment method', 'frequency of purchases'],
            dtype='object')
```

```
[23]: df.columns = df.columns.str.replace(' ','_')
```

```
[24]: df.columns
```

```
[24]: Index(['customer_id', 'age', 'gender', 'item_purchased', 'category',
             'purchase_amount_(usd)', 'location', 'size', 'color', 'season',
             'review_rating', 'subscription_status', 'shipping_type',
             'discount_applied', 'promo_code_used', 'previous_purchases',
             'payment_method', 'frequency_of_purchases'],
            dtype='object')
```

```
[25]: df = df.rename(columns = { 'purchase_amount_(usd)': 'purchase_amount'})
```

● Feature Engineering:

○ Created age_group column by binning customer ages.

[27]:
```
#adding new columns
labels = ['Young Adult','Adult','Middle-aged','Senior']
df['age_group'] = pd.qcut(df['age'],q=4,labels=labels)
```

[40]:
```
df.columns
```

[40]:
```
Index(['customer_id', 'age', 'gender', 'item_purchased', 'category',
       'purchase_amount', 'location', 'size', 'color', 'season',
       'review_rating', 'subscription_status', 'shipping_type',
       'discount_applied', 'previous_purchases', 'payment_method',
       'frequency_of_purchases', 'age_group', 'purchase_frequency_days'],
      dtype='object')
```

[41]:
```
df[['age','age_group']].head(10)
```

[41]:

	age	age_group
0	55	Middle-aged
1	19	Young Adult
2	50	Middle-aged
3	21	Young Adult

○ Created purchase_frequency_days column from purchase data.

•[55]:
```
#create one more column  called frequency_of_days -> becuase in our dataset we have
# one column called frequency_of_purchases
# which contains text so to create chart we require numbers so thats why based
# on this existing coulmn we are creating new column which contains numbers
frequency_mapping = {
    'Fortnightly':14,
    'Weekly':7,
    'Monthly':30,
    'Quarterly':90,
    'Bi-Weekly':14,
    'Annually':365,
    'Every 3 Months':90
}

df['purchase_frequency_days'] = df['frequency_of_purchases'].map(frequency_mapping)
```

[57]:
```
df[['purchase_frequency_days','frequency_of_purchases']].head(10)
```

[57]:

	purchase_frequency_days	frequency_of_purchases
0	14	Fortnightly
1	14	Fortnightly
2	7	Weekly

- Data Consistency Check:

  Verified if discount_applied and promo_code_used were redundant; dropped promo_code_used.

```
[45]: (df['discount_applied'] == df['promo_code_used']).all()
```

```
[37]: #Dropping a column
      df = df.drop('promo_code_used',axis=1)
```

- Database Integration:

  Connected Python script to MYSQL and loaded the cleaned DataFrame into the database for SQL analysis.

```
[59]: from sqlalchemy import create_engine
      import pandas as pd

      # Step 1: MySQL credentials
      username = "root"
      password = "root"
      host = "localhost"
      port = "3306"
      database = "customer_behaviour"

      # Step 2: Create MySQL engine
      engine = create_engine(
          f"mysql+pymysql://{username}:{password}@{host}:{port}/{database}"
      )

      # Step 3: Load DataFrame into MySQL
      table_name = "customer"

      df.to_sql(
          table_name,
          engine,
          if_exists="replace",
          index=False
      )

      print(f"Data Successfully Loaded into table '{table_name}' in database '{database}'."

      # Step 4: Verify data
      pd.read_sql("SELECT * FROM customer LIMIT 5;", engine)

      Data Successfully Loaded into table 'customer' in database 'customer_behaviour'.
```

5. Data Analysis using SQL (Business Transactions)

- What is the total revenue generated by male vs. female customers?

gender	revenue
Male	157890
Female	75191

- Which customers used a discount but still spent more than the average purchase amount?

customer_id	purchase_amount
2	64
3	73
4	90
7	85
9	97
12	68
13	72
16	81
20	90
22	62

- Which are the top 5 products with the highest average review rating?

item_purchased	Average_Rating
Gloves	3.86
Sandals	3.84
Boots	3.82
Hat	3.8
Skirt	3.78

- Compare the average Purchase Amounts between Standard and Express Shipping.

shipping_type	Average Amount
Express	60.48
Standard	58.46

- Do subscribed customers spend more? Compare average spend and total revenue between subscribers and non-subscribers.

subscription_status	total_Customer	Average_Revenue	Total_Revenue
Yes	1053	59.49	62645
No	2847	59.87	170436

- Which 5 products have the highest percentage of purchases with discounts applied?

item_purchased	Discount_percentage
Hat	50.00
Sneakers	49.66
Coat	49.07
Sweater	48.17
Pants	47.37

- Segment customers into New, Returning, and Loyal based on their total number of previous purchases, and show the count of each segment.

customer_segment	total_customer
LOYAL	3116
RETURNING	701
NEW	83

- What are the top 3 most purchased products within each category?

item_rank	category	item_purchased	total_orders
1	Accessories	Jewelry	171
2	Accessories	Sunglasses	161
3	Accessories	Belt	161
1	Clothing	Blouse	171
2	Clothing	Pants	171
3	Clothing	Shirt	169
1	Footwear	Sandals	160
2	Footwear	Shoes	150
3	Footwear	Sneakers	145
1	Outerwear	Jacket	163
2	Outerwear	Coat	161

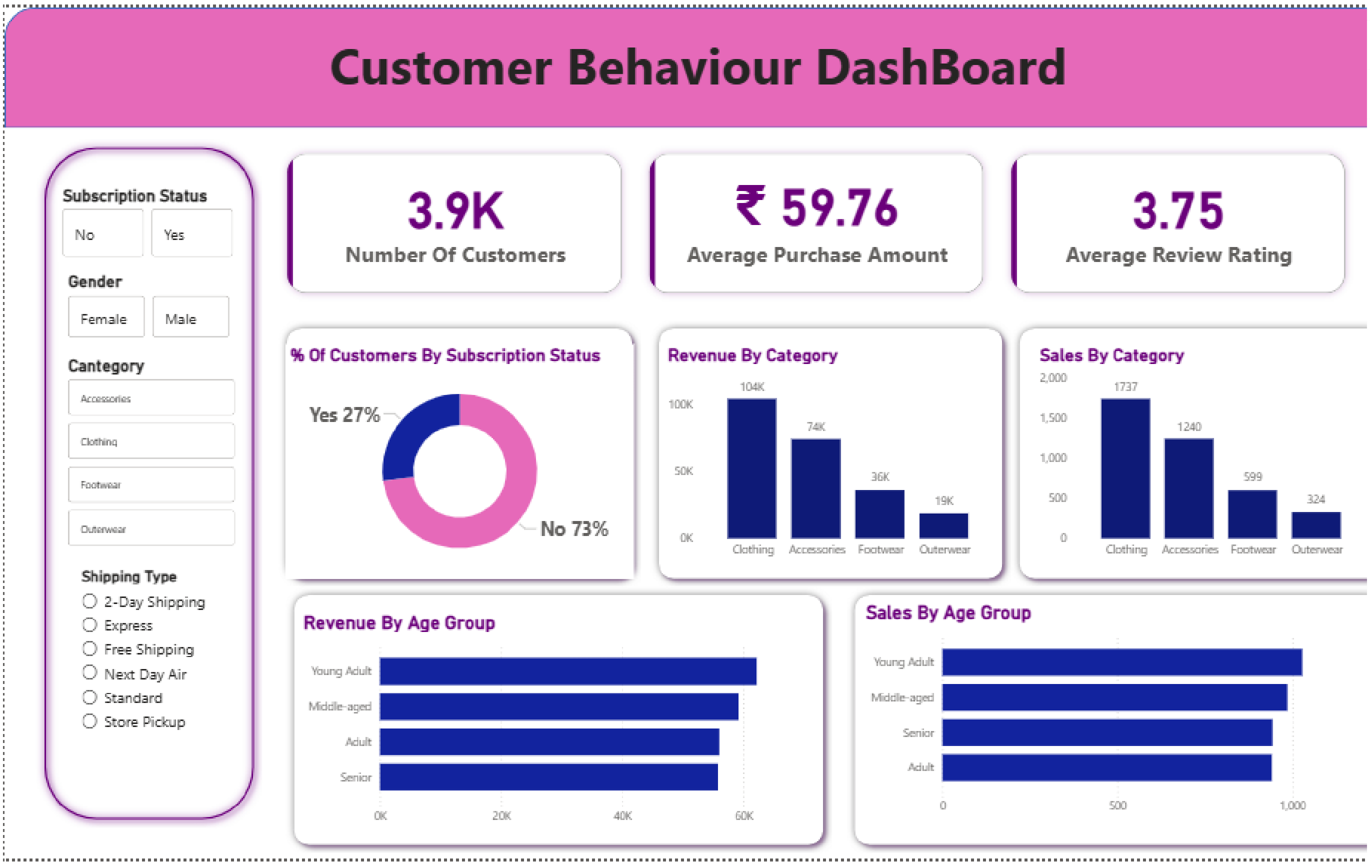
- Are customers who are repeat buyers (more than 5 previous purchases) also likely to subscribe?

subscription_status	Total_Customer
No	2518
Yes	958

- What is the revenue contribution of each age group?

age_group	total_revenue
Young Adult	62143
Middle-aged	59197
Adult	55978
Senior	55763

6. Dashboard in Power BI



7. Business Recommendations

- Boost Subscriptions –

  Promote exclusive benefits for subscribers.

- Customer Loyalty Programs –

  Reward repeat buyers to move them into the “Loyal” segment.

- Review Discount Policy –

  Balance sales boosts with margin control.

- Product Positioning –

  Highlight top-rated and best-selling products in campaigns.

- Targeted Marketing –

  Focus efforts on high-revenue age groups and express-shipping users.