

Customer Shopping Behavior Analysis – Project Report

1. Project Overview

This project analyzes 3,900 customer transactions across multiple product categories to uncover spending patterns, product preferences, subscription behavior, and customer segmentation. The insights aim to guide strategic business decisions and improve revenue, loyalty, and marketing effectiveness.

2. Dataset Summary

- **Rows:** 3,900
- **Columns:** 18
- **Key Features:** Customer demographics (Age, Gender, Location, Subscription Status), purchase details (Item, Category, Amount, Season, Size, Color), shopping behavior (Discount Applied, Promo Code, Previous Purchases, Frequency, Review Rating, Shipping Type)
- **Missing Data:** 37 values in Review Rating column

3. Methodology

Data Preparation & Python Analysis

- Imported data with **Pandas** and explored structure using `.info()` and `.describe()`.

```
: 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
```

```
: df.describe(include = 'all')
```

	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
count	3900.000000	3900.000000	3900	3900	3900	3900.000000	3900	3900	3900	3900	3863.000000	3900	3900	3900	3900	3900.000000	3900	3900
unique	NaN	NaN	2	25	4	NaN	50	4	25	4	NaN	2	6	2	2	NaN	6	7
top	NaN	NaN	Male	Blouse	Clothing	NaN	Montana	M	Olive	Spring	NaN	No	Free Shipping	No	No	NaN	PayPal	Every 3 Months
freq	NaN	NaN	2652	171	1737	NaN	96	1755	177	999	NaN	2847	675	2223	2223	NaN	677	584
mean	1950.500000	44.068462	NaN	NaN	NaN	59.764359	NaN	NaN	NaN	NaN	3.750065	NaN	NaN	NaN	NaN	25.351538	NaN	NaN
std	1125.977353	15.207589	NaN	NaN	NaN	23.685392	NaN	NaN	NaN	NaN	0.716983	NaN	NaN	NaN	NaN	14.447125	NaN	NaN
min	1.000000	18.000000	NaN	NaN	NaN	20.000000	NaN	NaN	NaN	NaN	2.500000	NaN	NaN	NaN	NaN	1.000000	NaN	NaN
25%	975.750000	31.000000	NaN	NaN	NaN	39.000000	NaN	NaN	NaN	NaN	3.100000	NaN	NaN	NaN	NaN	13.000000	NaN	NaN
50%	1950.500000	44.000000	NaN	NaN	NaN	60.000000	NaN	NaN	NaN	NaN	3.800000	NaN	NaN	NaN	NaN	25.000000	NaN	NaN
75%	2925.250000	57.000000	NaN	NaN	NaN	81.000000	NaN	NaN	NaN	NaN	4.400000	NaN	NaN	NaN	NaN	38.000000	NaN	NaN
max	3900.000000	70.000000	NaN	NaN	NaN	100.000000	NaN	NaN	NaN	NaN	5.000000	NaN	NaN	NaN	NaN	50.000000	NaN	NaN

- Imputed missing Review Ratings using **median by product category**.

```
: df.isnull().sum()
```

```
: 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
```

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

```
: df.isnull().sum()
```

```
: 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       0
   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
```

- Standardized column names and engineered features: `age_group` and `purchase_frequency_days`.

```
# for better readability and analysis change to snake case
df.columns = df.columns.str.lower()
df.columns = df.columns.str.replace(' ', '_')
df = df.rename(columns = {'purchase_amount_(usd)' : 'purchase_amount'})
```

```
df.columns
```

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

```
# create a column age_group

labels = ['Young Adult', 'Adult', 'Middle-Aged', 'Senior']
df['age_group'] = pd.qcut(df['age'], q = 4, labels = labels) #splits ages into four equal sized groups and assign labels defined
```

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

	age	age_group
0	55	Middle-Aged
1	19	Young Adult
2	50	Middle-Aged
3	21	Young Adult
4	45	Middle-Aged
5	46	Middle-Aged
6	63	Senior
7	27	Young Adult
8	26	Young Adult
9	57	Middle-Aged

```
# create column purchase_frequency_days

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) # replace the text to corresponding no of in columns

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

	purchase_frequency_days	frequency_of_purchases
0	14	Fortnightly
1	14	Fortnightly
2	7	Weekly
3	7	Weekly
4	365	Annually
5	7	Weekly
6	90	Quarterly
7	7	Weekly
8	365	Annually
9	90	Quarterly

- Dropped redundant columns (**promo_code_used**).

```
: df[['discount_applied', 'promo_code_used']].head(10)
```

```
:      discount_applied  promo_code_used
```

0	Yes	Yes
1	Yes	Yes
2	Yes	Yes
3	Yes	Yes
4	Yes	Yes
5	Yes	Yes
6	Yes	Yes
7	Yes	Yes
8	Yes	Yes
9	Yes	Yes

```
: (df['discount_applied'] == df['promo_code_used']).all() #is any one of the column redundant or not
```

```
: np.True_
```

```
: df = df.drop('promo_code_used', axis = 1)
```

```
: df.columns
```

```
: 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')
```

SQL Analysis (MYSQL)

- Revenue by gender, age group, and subscription status.

```

1 • SELECT * FROM customer limit 20;
2
3 # 1. What is the total revenue generated by male vs. female customers?
4
5 • SELECT gender, sum(purchase_amount) as revenue
6 FROM customer
7 GROUP BY gender
8
9 # 2. Which customer used a discount but still spent more than the average purchase amount?

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

gender	revenue
Male	157890
Female	75191

- Identified high-spending discount users.

```

9 # 2. Which customer used a discount but still spent more than the average purchase amount?
10
11 ✖ SELECT customer_id, purchase_amount
12 FROM customer
13 where discount_applied = 'yes' and purchase_amount >= (SELECT AVG(purchase_amount) FROM customer)
14
15 # 3. Which are the top 5 products with the highest average review rating?

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	purchase_amount
2	64
3	73
4	90
7	85
9	97
12	68
13	72
16	81
20	90
22	62
24	88
29	94
32	79
33	67

- Top 5 products by average rating and top 3 per category.

```

15      # 3. Which are the top 5 products with the highest average review rating?
16
17      SELECT item_purchased, avg(review_rating) as Average_ProductRating
18      FROM customer
19      group by item_purchased
20      order by avg(review_rating) DESC limit 5;
21
22      # 4. Compare the average purchase amounts between standard and express ship

```

Result Grid		
	Filter Rows:	Export: Wrap Cell Content: Fetch rows:
	item_purchased	Average_ProductRating
▶	Gloves	3.8627737226277383
	Sandals	3.8446540880503144
	Boots	3.818881118881119
	Hat	3.801307189542483
	Skirt	3.7853503184713366

- Customer segmentation into **New, Returning, Loyal**.

```

43      # 7. Segment customers into New, Returning, and Loyal based on their total number of previous purchases, and show the count of each segment
44
45      with customer_type as(
46      select customer_id,
47      previous_purchases,
48      case
49      when previous_purchases = 1 then 'New'
50      when previous_purchases between 2 and 10 then 'Returning'
51      else 'Loyal'
52      end as customer_segment
53      from customer
54      )
55
56      select customer_segment, count(*) as 'Number of Customers'
57      from customer_type
58      group by customer_segment
59
60      # 8. What are the top 3 most purchased products within each category?

```

Result Grid		
	Filter Rows:	Export: Wrap Cell Content:
	customer_segment	Number of Customers
▶	Loyal	3116
	Returning	701
	New	83

- Shipping type comparisons and repeat buyer analysis.

```
22 # 4. Compare the average purchase amounts between standard and express shipping?
23
24 • select shipping_type, avg(purchase_amount)
25     from customer
26     where shipping_type in ('Standard','Express')
27     group by shipping_type
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
shipping_type	avg(purchase_amount)			
Express	60.4752			
Standard	58.4602			

```
75 # 9. Are customers who are repeat buyers (more than 5 previous purchases) also likely to subscribe?
76
77 • select subscription_status, count(customer_id) as repeat_buyers
78     from customer
79     where previous_purchases > 5
80     group by subscription_status
81
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
subscription_status	repeat_buyers			
Yes	958			
No	2518			

Visualization

- Built interactive dashboards in **Power BI** to highlight revenue trends, customer segments, and top-performing products.



4. Key Insights

- Male and female customers contribute similarly to revenue, with certain age groups driving higher sales.
- High-spending customers often leverage discounts, indicating effective promotions.
- Subscribers spend more consistently than non-subscribers.
- Repeat buyers are more likely to subscribe, suggesting loyalty potential.
- Top-rated products align closely with high-revenue items, indicating successful offerings.

5. Recommendations

- Boost Subscriptions:** Offer exclusive benefits to subscribers.
- Customer Loyalty Programs:** Reward repeat buyers to increase retention.
- Optimize Discount Strategy:** Balance promotions with profit margins.

4. **Product Positioning:** Highlight top-rated and high-revenue products.
5. **Targeted Marketing:** Focus campaigns on high-value age groups and frequent buyers.

6. Tools & Technologies

- **Python:** Pandas, NumPy
- **SQL:** PostgreSQL
- **Visualization:** Power BI