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. Dataset Summary

- Rows: 3,900

- Columns: 18

- Key Features: - Customer demographics (Age, Gender, Location, Subscription Status)

 - Purchase details (Item Purchased, Category, Purchase Amount, Season, Size, Color)

- Shopping behavior (Discount Applied, Promo Code Used, Previous Purchases, Frequency of Purchases, Review Rating, Shipping Type)

- Missing Data: 37 values in Review Rating column

3. Exploratory Data Analysis using Python

We began with data preparation and cleaning in Python:

Loading Dataset: Imported the csv dataset using Python and df.head() is used to preview the dataset

```
import pandas as pd
df = pd.read_csv(r"E:\Data Analytics Project\customer_shopping_behavior.csv")
```

```
df.head()
```

	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	Paym Met
0	1	55	Male	Blouse	Clothing	53	Kentucky	L	Gray	Winter	3.1	Yes	Express	Yes	Yes	14	Ve
1	2	19	Male	Sweater	Clothing	64	Maine	L	Maroon	Winter	3.1	Yes	Express	Yes	Yes	2	(
2	3	50	Male	Jeans	Clothing	73	Massachusetts	S	Maroon	Spring	3.1	Yes	Free Shipping	Yes	Yes	23	Cr (
3	4	21	Male	Sandals	Footwear	90	Rhode Island	M	Maroon	Spring	3.5	Yes	Next Day Air	Yes	Yes	49	Pa
4	5	45	Male	Blouse	Clothing	49	Oregon	M	Turquoise	Spring	2.7	Yes	Free Shipping	Yes	Yes	31	Pa

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

```python
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
```

```python
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	
count	3900.000000	3900.000000	3900	3900	3900	3900.000000	3900	3900	3900	3900	3863.000000	3900	3900	3900	3900	3
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	

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

```python
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
```

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

```python
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
```

Column Standardization: Renamed columns to snake case for better readability and documentation.

```python
df.columns = [
    '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'
]
```

```python
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')
```

Feature Engineering:

1. Created age_group column by binning customer ages.
2. Created purchase_frequency_days column from purchase data.

```python
#create a colum age_group
labels = ['Young Adult', 'Adult','Middle-aged','Senior']
df['age_group'] = pd.qcut(df['age'], q=4,labels=labels)
```

```python
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

```python
# 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)
```

```python
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

Data Consistency Check: Verified if discount_applied and promo_code_used were redundant; dropped promo_code_used.

```python
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

Database Integration: Connected Python script to PostgreSQL and loaded the cleaned DataFrame into the database for SQL analysis.

```python
from sqlalchemy import create_engine

# Step 1: Connect to PostgreSQL
# Replace placeholders with your actual details
username = "postgres"          # default user
password = "******"          # the password you set during installation
host = "localhost"             # if running locally
port = "5432"                  # default PostgreSQL port
database = "customer_behavior"  # the database you created in pgAdmin

# Create the connection engine
engine = create_engine(f"postgresql+psycopg2://{username}:{password}@{host}:{port}/{database}")

# Step 2: Load DataFrame into PostgreSQL
table_name = "customer"        # choose any table name

# df refers to your existing Pandas DataFrame
df.to_sql(table_name, engine, if_exists="replace", index=False)

print(f"Data successfully loaded into table '{table_name}' in database '{database}'.")
```

```
Data successfully loaded into table 'customer' in database 'customer_behavior'.
```

4.Data Analysis using SQL (Business Transactions)

We performed structured analysis in PostgreSQL to answer key business questions:

1. **Revenue by Gender –** Compared total revenue generated by male vs. female customers.

	gender text	revenue numeric
1	Female	75191
2	Male	157890

2. **High-Spending Discount Users –** Identified customers who used discounts but still spent above the average purchase amount.

	customer_id bigint	purchase_amount bigint
1	2	64
2	3	73
3	4	90
4	7	85
5	9	97
6	12	68
7	13	72
8	16	81
9	20	90
10	22	62
11	24	88
12	29	94
13	32	79

3. **Top 5 Products by Rating –** Found products with the highest average review ratings.

	item_purchased text	Average Product Rating numeric
1	Gloves	3.86
2	Sandals	3.84
3	Boots	3.82
4	Hat	3.80
5	Skirt	3.78

4. **Shipping Type Comparison –** Compared average purchase amounts between Standard and Express shipping.

	shipping_type text	purchase_amount numeric
1	Standard	58.46
2	Express	60.48

5. **Subscribers vs. Non-Subscribers –** Compared average spend and total revenue across subscription status.

	subscription_status text	total_customers bigint	avg_spend numeric	total_revenue numeric
1	No	2847	59.87	170436.00
2	Yes	1053	59.49	62645.00

6. **Discount-Dependent Products –** Identified 5 products with the highest percentage of discounted purchases.

	item_purchased text	discount_rate numeric
1	Hat	50.00
2	Sneakers	49.66
3	Coat	49.07
4	Sweater	48.17
5	Pants	47.37

7. **Customer Segmentation –** Classified customers into New, Returning, and Loyal segments based on purchase history.

	customer_segment text	Number of Customer bigint
1	Loyal	3116
2	New	83
3	Returning	701

8. **Top 3 Products per Category –** Listed the most purchased products within each category.

	item_rank bigint	category text	item_purchased text	total_orders bigint
1	1	Accessori...	Jewelry	171
2	2	Accessori...	Sunglasses	161
3	3	Accessori...	Belt	161
4	1	Clothing	Blouse	171
5	2	Clothing	Pants	171
6	3	Clothing	Shirt	169
7	1	Footwear	Sandals	160
8	2	Footwear	Shoes	150
9	3	Footwear	Sneakers	145
10	1	Outerwear	Jacket	163
11	2	Outerwear	Coat	161

9. **Repeat Buyers & Subscriptions –** Checked whether customers with >5 purchases are more likely to subscribe.

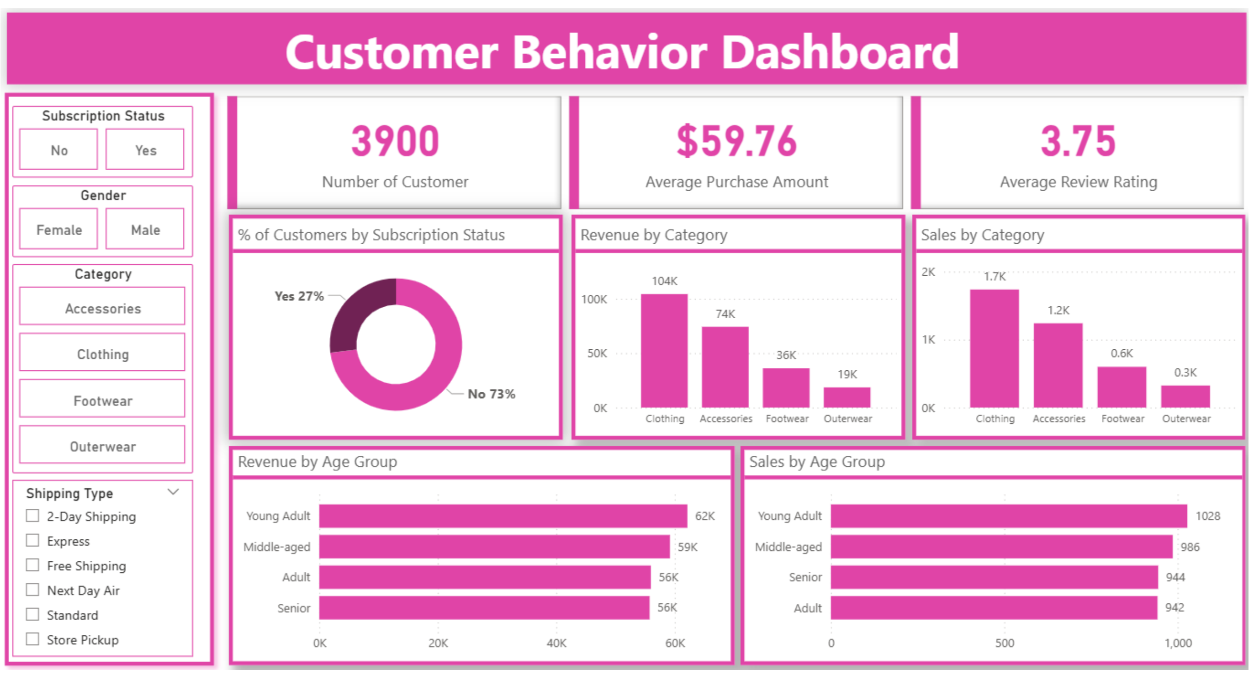
	subscription_status text	repeat_buyers bigint
1	No	2518
2	Yes	958

10. **Revenue by Age Group –** Calculated total revenue contribution of each age group.

	age_group text	total_revenue numeric
1	Young Adult	62143
2	Middle-aged	59197
3	Adult	55978
4	Senior	55763

5. Dashboard in Power BI

Finally, we built an interactive dashboard in Power BI to present insights visually



6. 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.