Customer Shopping Behaviour Analysis

1. **Project Overview**

   This project analyzes customer shopping behaviour 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 behaviour 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, Purchased Amount, Season, Size, Color)
     - ➢ Shopping Behaviour (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**
   - Data Preparation and cleaning in Python
     - Data Loading: Imported the dataset using pandas

```
import pandas as pd

#View the dataset
df = pd.read_csv(r"C:\Users\Kingboost\Desktop\Interswitch-job-shadowing\Data Analytics\Data_Analytcs_CompleteProjects\customer_shopping_behavior.csv")
df.head() #dsplay top 5 rows
```

	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	Pay Me
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	C
3	4	21	Male	Sandals	Footwear	90	Rhode Island	M	Maroon	Spring	3.5	Yes	Next Day Air	Yes	Yes	49	P
4	5	45	Male	Blouse	Clothing	49	Oregon	M	Turquoise	Spring	2.7	Yes	Free Shipping	Yes	Yes	31	P

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

```
#check for the structure of the dataset
df.info()
```

```
#check for the structure of the dataset
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
```

```
#To check for the summary statistics of all the columns
df.describe(include='all') #for both numerical and categorical columns
```

	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

```
#check for null values
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
```

```
#replace the null values
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
```

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

```
5]: #Review the columns and remove spacing e.g. Change Customer ID to customer_id
df.columns = df.columns.str.lower() #convert the column names to lowercase
df.columns = df.columns.str.replace(' ','_') #replace empty space with underscore('_')
#df.columns
df = df.rename(columns={'purchase_amount_(usd)':'purchase_amount'})
df.columns
```

```
5]: 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:**
  - Created **age_group** column by binning customer ages

```
# create a column age_group to group customers into 4 groups (Young adult, Adult, Middle_Aged, Senior)
labels = ['Young Adult', 'Adult', 'Middle-aged', 'Senior']
df['age_group'] = pd.qcut(df['age'], q=4, labels = labels)
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

  - **Created purchase_frequency_days column from purchase data**

```
]: # Create column purchase_frequency_days
# Create a dictionary to convert days of purchasing into numeric values for easy analysis

frequency_mapping = {
    'Fortnightly': 14,
    'Weekly': 7,
    'Monthly': 30,
    'Quarterly': 90,
    'Bi-Weekly': 14,
    'Annually':365,
    'Every 3 Months': 90
}

# store the value of each day into the column 'purchase_frequency_days' using map() function
df['purchase_frequency_days'] = df['frequency_of_purchases'].map(frequency_mapping)
# display the 1st 10 rows of purchase_frequency_days, frquency_of_purchases
df[['purchase_frequency_days','frequency_of_purchases']].head(10)
```

	purchase_frequency_days	frequency_of_purchases
0	14.0	Fortnightly
1	14.0	Fortnightly
2	7.0	Weekly
3	7.0	Weekly
4	365.0	Annually

- **Data Consistency Check: Verified if discount-applied and promo_code_used were redundant, dropped promo_code_used**

```
# Check if all discount_applied required promo_code
(df['discount_applied'] == df['promo_code_used']).all()
```

```
np.True_
```

```
#drop promo_code_used column
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'],
      dtype='object')
```

- **Database Integration: Connected Python script to SQL Server Database and loaded the cleaned DataFrame into the database for SQL analysis**

```
#Connecting to SQL Server
!pip install pyodbc sqlalchemy
```

```
Requirement already satisfied: pyodbc in c:\users\kingboost\anaconda3\lib\site-packages (5.2.0)
Requirement already satisfied: sqlalchemy in c:\users\kingboost\anaconda3\lib\site-packages (2.0.39)
Requirement already satisfied: greenlet!=0.4.17 in c:\users\kingboost\anaconda3\lib\site-packages (from sqlalchemy) (3.1.1)
Requirement already satisfied: typing-extensions>=4.6.0 in c:\users\kingboost\anaconda3\lib\site-packages (from sqlalchemy) (4.12.2)
```

```
import pyodbc
from sqlalchemy import create_engine
```

```
# Connect to SQL Server
conn_str = (
    "Driver={ODBC Driver 17 for SQL Server};"
    "Server=DESKTOP-N6NBVSJ\SQLEXPRESS;"
    "Database=customer_behaviour;"
    "Trusted_Connection=yes;"
)
engine = create_engine(f"mssql+pyodbc:///?odbc_connect={conn_str}")
```

```
# Load DataFrame into SQL Server
table_name = "customer" # Name of the table in SQL Server
df.to_sql(table_name, con=engine, if_exists='append', index=False)

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

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

4. **Data Analysis using SQL (Business Transactions)**

Structured analysis were performed in SQL Server Database to answer key business questions.

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

	gender	total_revenue
1	Female	150382
2	Male	315780

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

Results | Messages

	customer_id	purchase_amount	discount_applied
1	2	64	Yes
2	3	73	Yes
3	4	90	Yes
4	7	85	Yes
5	9	97	Yes
6	12	68	Yes
7	13	72	Yes
8	16	81	Yes
9	20	90	Yes
10	22	62	Yes
11	24	88	Yes
12	29	94	Yes
13	32	79	Yes
14	33	67	Yes
15	35	91	Yes
16	37	69	Yes
17	40	60	Yes

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

	item_purchased	average_review
1	Gloves	3.86
2	Sandals	3.84
3	Boots	3.82
4	Hat	3.8
5	Skirt	3.78

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

	shipping_type	Average_Purchased_Amounts
1	Standard	58
2	Express	60

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

	subscription_status	total_customer	Average_Spend	total_revenue
1	Yes	2106	59	125290
2	No	5694	59	340872

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

	item_purchased	discount_percentage
1	Hat	50
2	Coat	49
3	Sneakers	49
4	Sweater	48
5	Pants	47

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

	Customer_Segment	Number of Customers
1	New Customer	166
2	Returning Customer	1402
3	Loyal Customer	6232

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

	item_rank	category	item_purchased	total_orders
1	1	Accessories	Jewelry	342
2	2	Accessories	Belt	322
3	3	Accessories	Sunglasses	322
4	1	Clothing	Blouse	342
5	2	Clothing	Pants	342
6	3	Clothing	Shirt	338
7	1	Footwear	Sandals	320
8	2	Footwear	Shoes	300
9	3	Footwear	Sneakers	290
10	1	Outerwear	Jacket	326
11	2	Outerwear	Coat	322

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

	subscription_status	repeat_buyers
1	Yes	1916
2	No	5036

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

	age_group	total_revenue
1	Young Adult	124286
2	Middle-aged	118394
3	Adult	111956
4	Senior	111526

5. Dashboard in Power BI

6. Business Recommendations

✓ Boost Subscription – Promote exclusive benefits for subscribers

- ✓ Customer Loyalty Programs – Reward repeated 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.