Project: Customer sales behaviour analysis

Overview of project:

Python part

1) **Data loading**
2) **Exploratory Data Analysis (EDA)**
3) **Feature engineering**
4) **Export postgre sql**
5) **Conclusion**

SQL part

1)Using Query tool to answer the questions by writing query

PowerBI part

2) Creating meaningful insights generatable efficient Dashboard

Step1: Data loading

```
[12]: !pip install pandas
      import pandas as pd
      df=pd.read_csv("customer_shopping_behavior.csv")
      df.head()
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pandas in c:\users\win\appdata\roaming\python\python313\site-packages (2.3.3)
Requirement already satisfied: numpy>=1.26.0 in c:\users\win\appdata\roaming\python\python313\site-packages (from pandas) (2.3.5)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\win\appdata\roaming\python\python313\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\win\appdata\roaming\python\python313\site-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in c:\users\win\appdata\roaming\python\python313\site-packages (from pandas) (2025.2)
Requirement already satisfied: six>=1.5 in c:\users\win\appdata\roaming\python\python313\site-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
```

[12]:

	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 M
0	1	55	Male	Blouse	Clothing	53	Kentucky	L	Gray	Winter	3.1	Yes	Express	Yes	Yes	14	V
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	
3	4	21	Male	Sandals	Footwear	90	Rhode Island	M	Maroon	Spring	3.5	Yes	Next Day Air	Yes	Yes	49	
4	5	45	Male	Blouse	Clothing	49	Oregon	M	Turquoise	Spring	2.7	Yes	Free Shipping	Yes	Yes	31	

Step2: EDA

[14]:
```
df.shape
```

[14]:
```
(3900, 18)
```

[15]:
```
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
```

Detailed description

```
df.describe()
```

	Customer ID	Age	Purchase Amount (USD)	Review Rating	Previous Purchases
count	3900.000000	3900.000000	3900.000000	3863.000000	3900.000000
mean	1950.500000	44.068462	59.764359	3.750065	25.351538
std	1125.977353	15.207589	23.685392	0.716983	14.447125
min	1.000000	18.000000	20.000000	2.500000	1.000000
25%	975.750000	31.000000	39.000000	3.100000	13.000000
50%	1950.500000	44.000000	60.000000	3.800000	25.000000
75%	2925.250000	57.000000	81.000000	4.400000	38.000000
max	3900.000000	70.000000	100.000000	5.000000	50.000000

Checking 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
```

Filling 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
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
```

Shaping columns to lower case and space into _ for further easy query performing in sql

```
# shaping columns names to lowercase and spaces to _
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')
```

Step3: Feature Engineering

```
# creating new column(feature engineering)
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
7	27	Young Adult
8	26	Young Adult
9	57	Middle Aged

[25]:
```
# creating new column purchase_frequency_days

frequency_mapping={
    'fortinightly':14,
    'weekly':7,
    'monthly':30,
    'quaterly':90,
    'annually':365,
    'bi-weekly':14,
    'Every 3 months':90
}

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

[25]:

	purchase_frequency_days	frequency_of_purchases
0	NaN	Fortnightly
1	NaN	Fortnightly
2	NaN	Weekly
3	NaN	Weekly
4	NaN	Annually
5	NaN	Weekly
6	NaN	Quarterly
7	NaN	Weekly
8	NaN	Annually

Step4: Export To PostgreSql

[32]:
```
pip install psycopg2-binary sqlalchemy
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: psycopg2-binary in c:\users\win\appdata\roaming\python\python313\site-packages (2.9.11)
Requirement already satisfied: sqlalchemy in c:\users\win\appdata\roaming\python\python313\site-packages (2.0.44)
Requirement already satisfied: greenlet>=1 in c:\users\win\appdata\roaming\python\python313\site-packages (from sqlalchemy) (3.2.4)
Requirement already satisfied: typing-extensions>=4.6.0 in c:\programdata\miniconda3\lib\site-packages (from sqlalchemy) (4.15.0)
Note: you may need to restart the kernel to use updated packages.
```

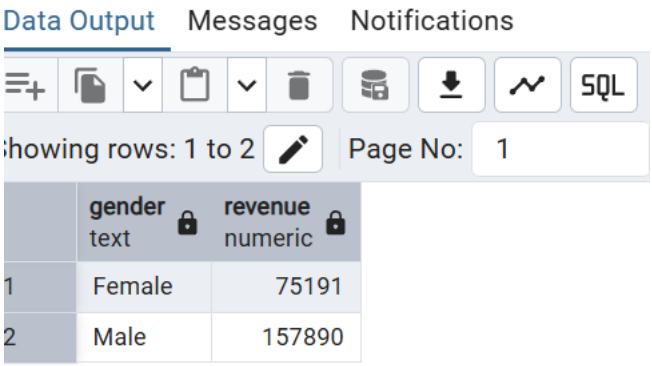
[37]:
```
from sqlalchemy import create_engine
# step1:connect to postgresql
username="postgres"
password="5052586"
host="localhost"
port="5432"
database="sales_behaviour"

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

# load data into postgresql
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}")
```

```
Data successfully loaded into table customer in database sales_behaviour
```

Postgres SQL QUERY

-- Q1 what is the total revenue generated by male vs female customers?

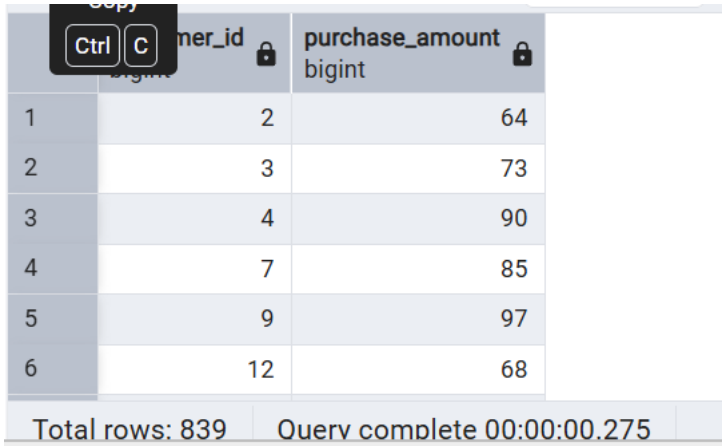
```
select gender, sum(purchase_amount) as revenue
from customer
group by gender
```



	gender text	revenue numeric
1	Female	75191
2	Male	157890

-- Q2 which customer used the discounts but still they spent more than the average purchase amount

```
select customer_id,purchase_amount
from customer
where discount_applied='Yes' and purchase_amount >= (select avg(purchase_amount) from customer)
```



	mer_id	purchase_amount bigint
1	2	64
2	3	73
3	4	90
4	7	85
5	9	97
6	12	68

Total rows: 839 Query complete 00:00:00.275

```sql
--Q3 which are the top 5 products with the hihest review rating?

select item_purchased,round(avg(review_rating::Numeric),2) as "avg product ra
from customer
group by item_purchased
order by avg(review_rating) desc
limit 5;
```

Data Output | Messages | Notifications

Showing rows: 1 to 5 | Page No: 1 of 1

	item_purchased text	avg product rating numeric
1	Gloves	3.86
2	Sandals	3.84
3	Boots	3.82
4	Hat	3.80
5	Skirt	3.78

```sql
-- Q4 compare the average purchase shipping between the standard and express

select shipping_type,
round(avg(purchase_amount),2)
from customer
where shipping_type in('Standard','Express')
group by shipping_type
```

Data Output | Messages | Notifications

Showing rows: 1 to 2 | Page No: 1 of 1

	shipping_type text	round numeric
1	Standard	58.46
2	Express	60.48

```sql
-- Q5 do subscribers customer spend more ? compare average spend on total rev
select subscription_status,
count(customer_id) as total_customer,
round(avg(purchase_amount),2) as average_spent,
sum(purchase_amount) as total_revenue
from customer
group by subscription_status
order by total_revenue,average_spent;
```

Data Output | Messages | Notifications

Showing rows: 1 to 2 | Page No: 1 of 1

	subscription_status text	total_customer bigint	average_spent numeric	total_revenue numeric
	Yes	1053	59.49	62645
	No	2847	59.87	170436

```sql
-- Q6 which 5 products have highest percentage of purchases with discount ap
select item_purchased,
round(100*sum(case when discount_applied='Yes' then 1 else 0 end)/count(*),2)
from customer
group by item_purchased
order by discount_rate desc
limit 5;
```

Data Output | Messages | Notifications

Showing rows: 1 to 5 | Page No: 1 of 1

	item_purchased text	discount_rate numeric
1	Hat	50.00
2	Sneakers	49.00
3	Coat	49.00
4	Sweater	48.00
5	Pants	47.00

```sql
-- Q7 segments the customer into new,returning,loyal based on their previous_purchase
-- also show the count of each segements
with customer_type as (
select customer_id , previous_purchases,
Case
    when previous_purchases=1 then 'New'
            when previous_purchases between 2 and 10 then 'Returning'
```

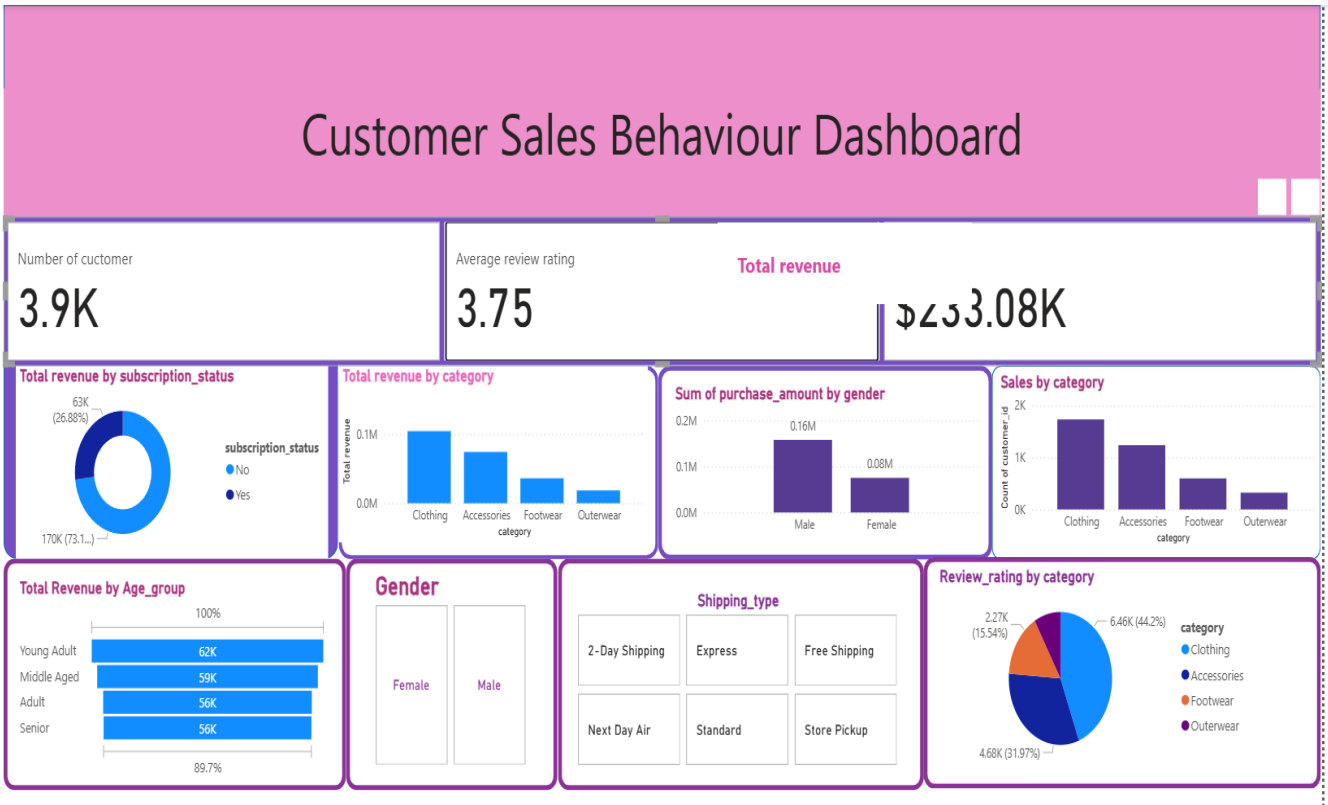
```
            else 'Loyal'
            end as customer_segment
from customer
)

select customer_segment,count(*) as no_of_customers
from customer_type
group  by customer_segment;
```

Data Output | Messages | Notifications

Showing rows: 1 to 3 | Page No: 1

	customer_segment text	no_of_customers bigint
1	Loyal	3116
2	New	83
3	Returning	701

PowerBI Dashboard



Conclusion :

1. **I successfully loaded and explored the customer sales dataset**, ensuring the data was clean, organized, and ready for deeper analysis.
2. **Through EDA, we identified missing values, corrected column formats, and improved data consistency**, which made further processing and SQL querying much easier.
3. **Feature engineering added meaningful new attributes**, helping us understand customer behaviour more effectively.
4. **The cleaned and processed dataset was exported to PostgreSQL**, allowing structured storage and efficient SQL-based analysis.
5. **Using SQL queries, we uncovered key insights** such as revenue differences between genders, customers who spent more even after using discounts, and segmenting customers into New, Returning, and Loyal categories.
6. **These insights were visualized in Power BI**, where an interactive dashboard was created to clearly highlight trends, patterns, and business opportunities.
7. **Overall, this project demonstrates the complete journey from raw data to meaningful insights**, helping businesses understand customer behaviour and make smarter, data-driven decisions.