Elise Berger

MSc Business Analytics (2025-2026)

The Final Project was done on **Google Colab**

I - Introduction

Context Analysis

The global e-commerce market has experienced unprecedented growth, leading to a profound transformation of the retail sector. Online sales now account for 20.8% of total retail sales worldwide. But this is far from over, as this figure continues to rise every year (eMarketer, 2024).

However, this digital transformation is a double-edged sword for retailers. While online platforms offer unprecedented reach and data collection capabilities, they also intensify competition. Consumers can now compare prices, read reviews and switch brands with a single click. One key indicator confirming this is the average cost of customer acquisition. In e-commerce, this indicator has increased by 60% over the last five years, while customer loyalty has declined significantly (source: Bain & Company, 2023). In this context, **customer loyalty has become a critical success factor**.

Despite this obvious necessity, many e-commerce companies continue to rely on simplistic and standardised approaches to customer retention. Advanced segmentation strategies that leverage demographic, behavioural and promotional data remain underutilised. However, there is evidence that companies that use sophisticated segmentation achieve higher revenue growth than other companies. We will therefore focus our analysis on answering the following question :

> **"Which demographic, behavioural and promotional factors predict customer engagement and commercial performance in e-commerce ? How can advanced segmentation optimise retention and acquisition strategies ?"**

Dataset Presentation

To analyse customer engagement and commercial performance in e-commerce, I selected the complete and relevant dataset, found on Kaggle:

> **'Shopping Trends And Customer Behaviour Dataset'**
>
> https://www.kaggle.com/datasets/sahilislam007/shopping-trends-and-customer-behaviour-dataset

The dataset contains **3900 customer observations** across **18 variables**, providing comprehensive coverage of e-commerce. The dataset was specifically selected for its perfect alignement with the research question :

1. Demographic factors
    - **Age** : to analyse the generational segmentation
    - **Gender** : to make comparisons based on gender
    - **Location** : to analyse the geographic segmentation
2. Behavioral factors
    - **Previous Purchases** : it is an indicator of customer loyalty with the shop
    - **Frequency of Purchases** : it is an another metric of customer loyalty
    - **Review Rating** : a metric to determine the rate of customer satisfaction
    - **Subscription Status** : an indicator of customer commitment
3. Promotional factors
    - **Discount Applied** : to determine whether or not the discount had an impact on the customer's purchasing behaviour
    - **Promo Code Used** : to determine if a price reduction can have an impact or not in customer's purchasing behaviour
    - **Shipping Type** : a metric of customer experience (more focused on product delivery/receipt)
4. Commercial Performance Metric
    - **Purchase Amount (USD)** : an outcome variable (the primary indicator of a shop's economic health)
    - **Category** : gives product segmentation
    - **Payment Method** : a metric of payment preferences

Thus, we can see that, thanks to its large number of observations and variables, the dataset provides a wealth of information on online buyer behaviour and consumer trends.

Roadmap

This analysis was carried out by following the six essential steps of data analysis:

1. **Introduction** : this involves not only presenting the context and the dataset, but also defining the research question that will guide the rest of the analysis.
2. **Data exploration** : compiling a summary of statistics. This will be used to identify patterns and formulate initial hypotheses.
3. **Data cleaning** : standardisation and verification of the consistency of variables and data. This is also the stage at which derived variables are created to enrich the analysis.
4. **Data analysis & Visualization** : statistical tests are performed and correlations between variables are identified. These results are represented graphically, ensuring that the optimal visualisation format is chosen for each hypothesis. Relevant analytical methods are used to answer the research question.
5. **Conclusion** : synthesis of the results. This is the final stage. Any recommendations are made and strategic recommendations are formulated. It also involves answering the research question.

II - Data Exploration

Numerical Variables : Overview

```
import pandas as pd

url = 'https://raw.githubusercontent.com/brg556/Final-Project_Python-Programming-Business_Analytics/refs/heads/main/shoppir
shopping_trends = pd.read_csv(url, sep =';')

print(shopping_trends.shape)
shopping_trends.info()
```

```
(3900, 18)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3900 entries, 0 to 3899
Data columns (total 18 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Unnamed: 0              3900 non-null   int64
 1   Customer ID             3900 non-null   int64
 2   Age                     3900 non-null   int64
 3   Gender                  3900 non-null   object
 4   Item Purchased          3900 non-null   object
 5   Category                3900 non-null   object
 6   Purchase Amount (USD)   3900 non-null   int64
 7   Location                3900 non-null   object
 8   Color                   3900 non-null   object
 9   Season                  3900 non-null   object
 10  Review Rating           3900 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(5), object(12)
memory usage: 548.6+ KB
```

```
shopping_trends.head()
```

	Unnamed: 0	Customer ID	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	Location	Color	Season	Review Rating	Subscription Status	Ship
0	0	1	55	Male	Blouse	Clothing	53	Kentucky	Gray	Winter	3.1	Yes	Ex
1	1	2	19	Male	Sweater	Clothing	64	Maine	Maroon	Winter	3.1	Yes	Ex
2	2	3	50	Male	Jeans	Clothing	73	Massachusetts	Maroon	Spring	3.1	Yes	Shi
3	3	4	21	Male	Sandals	Footwear	90	Rhode Island	Maroon	Spring	3.5	Yes	Nex
4	4	5	45	Male	Blouse	Clothing	49	Oregon	Turquoise	Spring	2.7	Yes	

Étapes suivantes : Générer du code avec `shopping_trends` New interactive sheet

Structure Overview

`.info()` gives us the data structure :

- 3900 raws (observations) and 18 columns (variables)
- Data types : 1 float64, 5 int64, 12 object
- NO null value -> all the data are complete

```
shopping_trends.describe()
```

	Unnamed: 0	Customer ID	Age	Purchase Amount (USD)	Review Rating	Previous Purchases
count	3900.000000	3900.000000	3900.000000	3900.000000	3900.000000	3900.000000
mean	1949.500000	1950.500000	44.068462	59.764359	3.749949	25.351538
std	1125.977353	1125.977353	15.207589	23.685392	0.716223	14.447125
min	0.000000	1.000000	18.000000	20.000000	2.500000	1.000000
25%	974.750000	975.750000	31.000000	39.000000	3.100000	13.000000
50%	1949.500000	1950.500000	44.000000	60.000000	3.700000	25.000000
75%	2924.250000	2925.250000	57.000000	81.000000	4.400000	38.000000
max	3899.000000	3900.000000	70.000000	100.000000	5.000000	50.000000

Numerical Variables : Descriptive Statistics

`.describe()` output provides univariate analysis of numeric variables, essential for understanding customer behavior :

1. `Customer ID` : range = 0-3899 confirms that we have 3 900 unique customer records
2. `Age` :
    - mean = 44,07 | median = 44.00 | range = 18 - 70 | IQR = 31 - 57
    - primary customer segment : between 31 and 57 years (50% of customers are mature people)
    - low representation of young people : only 25 % are ≤ 31 years
    - low representation of old people : only 25 % are ≤ 57 years

Pattern: The main customer segment is adults (mature people) aged between 31 and 57 years old.

3. `Purchase Amount (USD)` :
    - mean = 59.76$ | median = 60.00$ | IQR = 39$ - 81$
    - std = 23.69 -> a wide variance highlighting diverse spending behaviours
    - he purchase amount varies from 20$ (min) to 100$ (max)
    - IQR : difference between the first and the third quartile shows that 50% of transactions fall between 39$ and 81$
    - no outliers

Pattern: With an interquartile range of 39$ - 81$ and a total range of 20$ - 100$, purchase amounts exhibit moderate variance. This dispersion pattern suggests the product catalog serves heterogeneous customer segments with varying price sensitivities.

4. `Review Rating` :
    - mean = 3.75$ | median = 3.70$ | IQR = 3.00$ - 4.00$
    - std = 0.72 -> indicates low variance in customer satisfaction
    - review rating range : between 2.50$ and 5.00$
    - 50% = of customers give a review rating betwwen 3.00$ and 4.00$ -> they are moderatly satisfy by their spending behavior
    - 25% ≤ 3.10$ | 25% ≥ 4.40$
5. `Previous Purchases` :
    - mean = 25.35$ | median = 25.00$ | IQR = 13$ - 38$ -> highlights a strong repeat behavior
    - std = 14.45 -> shows differents customers' loyalty
    - range = between 1$ and 50$
    - 3 customers categories :
    - 25% = new or occasionales customers (between 1$ and 13$ purchases)
    - 50% = regular customers (between 13$ and 38$ purchases)
    - 25% = loyal customers (between 38$ and 50$ purchases)

Globally this dataset represents a symmetric distribution because mean value are almost similar or very close to the median value

Moreover, this dataset shows a homogeneous customer experience with a low variance satisfaction (std = 0.72). But also, we can observe a high variance in spending (std = 23.69) and purchase history (std = 14.45) showing diverse behavioral segments.

Categorical Variables : Analysis

Now, to identify more patterns, it is interesting to explore categorical variables.

```
print('='*80)
print('GENDER DISTRIBUTION'.center(80))
print('='*80)

print(shopping_trends['Gender'].value_counts())
gender_count = shopping_trends['Gender'].value_counts(normalize = True)*100

male = gender_count.get('Male', 0)
female = gender_count.get('Female', 0)

print(f"\nPattern: The customers are mostly men ({male}%), versus women ({female}%).")
```

```
================================================================================
                              GENDER DISTRIBUTION
================================================================================
Gender
Male      2652
Female    1248
Name: count, dtype: int64

Pattern: The customers are mostly men (68.0%), versus women (32.0%).
```

```
print('='*80)
print('ITEM PURCHASES DISTRIBUTION'.center(80))
print('='*80)

print(shopping_trends['Item Purchased'].value_counts())
print(f"\nThere is no pattern in the 'Item Purchased' variable. All item were purchased an almost similar number of times")
```

```
================================================================================
                          ITEM PURCHASES DISTRIBUTION
================================================================================
Item Purchased
Blouse        171
Pants         171
Jewelry       171
Shirt         169
Dress         166
Sweater       164
Jacket        163
Coat          161
Sunglasses    161
Belt          161
Sandals       160
Socks         159
Skirt         158
Scarf         157
Shorts        157
Hat           154
Handbag       153
Hoodie        151
Shoes         150
T-shirt       147
Sneakers      145
Boots         144
Backpack      143
Gloves        140
Jeans         124
Name: count, dtype: int64

There is no pattern in the 'Item Purchased' variable. All item were purchased an almost similar number of times
```

```
print('='*80)
print('CATEGORY DISTRIBUTION'.center(80))
print('='*80)

print(shopping_trends['Category'].value_counts())
category_percentage = shopping_trends['Category'].value_counts(normalize = True)*100

clothing = category_percentage.get('Clothing', 0)
accessories = category_percentage.get('Accessories', 0)
footwear = category_percentage.get('Footwear', 0)
outerwear = category_percentage.get('Outerwear', 0)

print(f"\nPattern: Consumers buy more clothing ({clothing:.2f}%) than shoes ({footwear:.2f}%) or outerwear ({outerwear:.2f}
```

```
================================================================================
                            CATEGORY DISTRIBUTION
================================================================================
Category
Clothing       1737
Accessories    1240
Footwear        599
```

```
Outerwear       324
Name: count, dtype: int64

Pattern: Consumers buy more clothing (44.54%) than shoes (15.36%) or outerwear (8.31%).
```

```
print('='*80)
print('LOCATION DISTRIBUTION'.center(80))
print('='*80)

print(shopping_trends['Location'].value_counts())
print(f"\There is no pattern in the 'Location' variable. The different American states have roughly the same number of orde
```

```
Alabama           89
Minnesota         88
New York          87
Nevada            87
Nebraska          87
Delaware          86
Maryland          86
Vermont           85
Louisiana         84
North Dakota      83
West Virginia     81
Missouri          81
New Mexico        81
Mississippi       80
Kentucky          79
Arkansas          79
Georgia           79
Indiana           79
Connecticut       78
North Carolina    78
Maine             77
Ohio              77
Virginia          77
Texas             77
Tennessee         77
South Carolina    76
Oklahoma          75
Wisconsin         75
Colorado          75
Oregon            74
Pennsylvania      74
Michigan          73
Washington        73
Alaska            72
Massachusetts     72
Utah              71
Wyoming           71
New Hampshire     71
South Dakota      70
Iowa              69
Florida           68
New Jersey        67
Arizona           65
Hawaii            65
Rhode Island      63
Kansas            63
Name: count, dtype: int64
\There is no pattern in the 'Location' variable. The different American states have roughly the same number of orders.
<>:6: SyntaxWarning:

invalid escape sequence '\T'

<>:6: SyntaxWarning:

invalid escape sequence '\T'

/tmp/ipython-input-316466929.py:6: SyntaxWarning:
```

```
print('='*80)
print('COLOR DISTRIBUTION'.center(80))
print('='*80)

print(shopping_trends['Color'].value_counts())
print(f"\nThere is no pattern in the 'Color' variable. All colors are represented in the same way.")
```

```
================================================================================
                              COLOR DISTRIBUTION
================================================================================
Color
Olive        177
Yellow       174
Silver       173
Teal         172
Green        169
Black        167
```

```
Cyan         166
Violet       166
Gray         159
Maroon       158
Orange       154
Charcoal     153
Pink         153
Blue         152
Magenta      152
Purple       151
Peach        149
Red          148
Beige        147
Indigo       147
Lavender     147
Turquoise    145
White        142
Brown        141
Gold         138
Name: count, dtype: int64

There is no pattern in the 'Color' variable. All colors are represented in the same way.
```

```
print('='*80)
print('SEASON DISTRIBUTION'.center(80))
print('='*80)

print(shopping_trends['Season'].value_counts())
print(f"\nThere is no pattern in the 'Season' variable. Sales remained stable throughout the year.")
```

```
================================================================================
                              SEASON DISTRIBUTION
================================================================================
Season
Spring    999
Fall      975
Winter    971
Summer    955
Name: count, dtype: int64

There is no pattern in the 'Season' variable. Sales remained stable throughout the year.
```

```
print('='*80)
print('SUBSCRIPTION STATUS DISTRIBUTION'.center(80))
print('='*80)

print(shopping_trends['Subscription Status'].value_counts())
subscription_count = shopping_trends['Subscription Status'].value_counts(normalize = True)*100
yes = subscription_count.get('Yes', 0)
no = subscription_count.get('No', 0)

print(f"\nPattern: Consumers have not subscribed to a subscription ({no}%) compared to customers who have subscribed ({yes}
```

```
================================================================================
                        SUBSCRIPTION STATUS DISTRIBUTION
================================================================================
Subscription Status
No     2847
Yes    1053
Name: count, dtype: int64

Pattern: Consumers have not subscribed to a subscription (73.0%) compared to customers who have subscribed (27.0%).
```

```
print('='*80)
print('SHIPPING TYPE DISTRIBUTION'.center(80))
print('='*80)

print(shopping_trends['Shipping Type'].value_counts())

print(f"\nThere is no pattern in the 'Shipping Type' variable. All shipping types are used in almost the same way.")
```

```
================================================================================
                           SHIPPING TYPE DISTRIBUTION
================================================================================
Shipping Type
Free Shipping     675
Standard          654
Store Pickup      650
Next Day Air      648
Express           646
2-Day Shipping    627
Name: count, dtype: int64

There is no pattern in the 'Shipping Type' variable. All shipping types are used in almost the same way.
```

```
print('='*80)
print('DISCOUNT APPLIED DISTRIBUTION'.center(80))
print('='*80)

print(shopping_trends['Discount Applied'].value_counts())
discount_count = shopping_trends['Discount Applied'].value_counts(normalize = True)*100

yes = discount_count.get('Yes',0)
no = discount_count.get('No',0)

print(f"\nPattern: Most consumers did not use a discount ({no:.2f}%), unlike {yes:.2f}% for consumers who did use one.")
```

```
================================================================================
                          DISCOUNT APPLIED DISTRIBUTION
================================================================================
Discount Applied
No     2223
Yes    1677
Name: count, dtype: int64

Pattern: Most consumers did not use a discount (57.00%), unlike 43.00% for consumers who did use one.
```

```
print('='*80)
print('PROMOTIONAL CODE DISTRIBUTION'.center(80))
print('='*80)

print(shopping_trends['Promo Code Used'].value_counts())
promo_count = shopping_trends['Promo Code Used'].value_counts(normalize = True)*100

yes = promo_count.get('Yes',0)
no = promo_count.get('No',0)

print(f"\nPattern: Most consumers did not use a promotional code ({no:.2f}%), unlike {yes:.2f}% for consumers who did use a
```

```
================================================================================
                          PROMOTIONAL CODE DISTRIBUTION
================================================================================
Promo Code Used
No     2223
Yes    1677
Name: count, dtype: int64

Pattern: Most consumers did not use a promotional code (57.00%), unlike 43.00% for consumers who did use a promotional code.
```

NOTE: The percentages for the variables 'Discount Applied' and 'Promo Code Used' are the same. Therefore, we can say that there are as many consumers who did not use a promotional code as those who did not use a discount.

Now, it will be interesting to check if consumers who did not use a promotional code are the same as those who did not use a discount

```
#We start by creating the formula to determine that if the result of the variables 'Discount Applied' and 'Promo Code Used'

#Define customers who did not apply a discount
no_discount = set(shopping_trends[shopping_trends['Discount Applied'] == 'No']['Customer ID'].unique())

#Define customers who did not use a promotional code
no_promo_code = set(shopping_trends[shopping_trends['Promo Code Used'] == 'No']['Customer ID'].unique())

same_customers = no_discount & no_promo_code

print("-"*80)
print("Summary table".center(80))
print("-"*80)
print(f"Number of customers who did not apply a discount: {len(no_discount)}")
print(f"Number of customers who did not use a promotional code: {len(no_promo_code)}")
print(f"\nCustomers identical: {len(same_customers)}")

print(f"\nPattern: Customers who did not apply a discount are the same as those who did not use a promotional code.")
print(f"We can delete one of both column because it is a duplicate variable (next stop in data cleaning)")
```

```
--------------------------------------------------------------------------------
                                Summary table
--------------------------------------------------------------------------------
Number of customers who did not apply a discount: 2223
Number of customers who did not use a promotional code: 2223

Customers identical: 2223

Pattern: Customers who did not apply a discount are the same as those who did not use a promotional code.
We can delete one of both column because it is a duplicate variable (next stop in data cleaning)
```

```
print('='*80)
print('PAYMENT METHOD DISTRIBUTION'.center(80))
print('='*80)

print(shopping_trends['Payment Method'].value_counts())

print(f"\nThere is no pattern in the 'Payment Method' variable. All payment methods are used in an identical manner.")
```

```
================================================================================
                          PAYMENT METHOD DISTRIBUTION
================================================================================
Payment Method
PayPal           677
Credit Card      671
Cash             670
Debit Card       636
Venmo            634
Bank Transfer    612
Name: count, dtype: int64

There is no pattern in the 'Payment Method' variable. All payment methods are used in an identical manner.
```

```
print('='*80)
print('FREQUENCY OF PURCHASES DISTRIBUTION'.center(80))
print('='*80)

print(shopping_trends['Frequency of Purchases'].value_counts())

max_value = shopping_trends['Frequency of Purchases'].value_counts().max()
min_value = shopping_trends['Frequency of Purchases'].value_counts().min()
total_value = len(shopping_trends['Frequency of Purchases'])

difference_percentage = ((max_value - min_value)/total_value)*100

print(f"\nThere is no pattern in the 'Frequency of Purchases' variable.")
print(
    f"\nThere is a very small (non-significant) difference between the most frequently used frequency :"
    f"\n'Every 3 months' = 584 & 'Weekly' = 539.")
print(f"Non-significant difference: {difference_percentage:.2f}%")
```

```
================================================================================
                       FREQUENCY OF PURCHASES DISTRIBUTION
================================================================================
Frequency of Purchases
Every 3 Months    584
Annually          572
Quarterly         563
Monthly           553
Bi-Weekly         547
Fortnightly       542
Weekly            539
Name: count, dtype: int64

There is no pattern in the 'Frequency of Purchases' variable.

There is a very small (non-significant) difference between the most frequently used frequency :
'Every 3 months' = 584 & 'Weekly' = 539.
Non-significant difference: 1.15%
```

Preliminary Hypotheses

Based on the data exploration phase we just completed and the characteristics of the dataset, my initial hypotheses are as follows:

1. Seasons influence consumer preferences and purchasing choices. For example, certain outerwear categories may show higher sales figures in winter.
2. Subscribed customers exhibit significantly higher levels of engagement than non-subscribed customers.
3. Regular customers have lower average order values but higher engagement scores than new and VIP customers.
4. Review ratings are positively correlated with engagement scores, confirming that satisfaction fosters loyalty.
5. The use of discounts increase the amount of purchases and customer engagement in the short term.

These hypotheses will be rigorously tested in 'Data Analysis' section.

III - Data Cleaning

Before analysing our datasets, we must first carry out the “Data Cleaning” step. This step identifies and corrects inconsistencies, duplicates and data type issues.

As we saw in the previous section : “Data Exploration”, the dataset contains zero missing values. For the “Data Cleaning” step, I focused on the following steps :

- Removing redundant columns (example : removing 'Unnamed' column wich is a duplicate of 'Customer ID' column)
- Standardising formats : put all column names in the same format (in lowercase with underscores)
- Converting dates ('YYYY-MM-DD") to datetime ('YYYY-MM-DD HH:MM:SS')
- Converting categorical binaries ('Yes', 'No') to boolean ('True' or 'False')
- Verify variables consistency (for example that total_price = quantity * price)
- Creating derived variables. This will be useful for enriching data analysis

Furthermore, in order to carry out the data cleaning stage correctly, I made sure to clean the datasets individually.

Remove redundant variables

```
print('='*80)
print('REMOVE REDUNDANT VARIABLES'.center(80))
print('='*80)

#Remove 'Unnamed' column which is a duplicates of 'Customer ID'
shopping_trends = shopping_trends.drop(columns=['Unnamed: 0'])
print(f"List without 'Unnamed' variable: \n{list(shopping_trends)} \n& {shopping_trends.shape}")

#The deletion has been carried out -> The “list(df.columns)” function confirms the change from 18 to 17 columns
print(f"\nDELETION SUCCESSFULL : The dataset currently contains 17 variables, compared to 18 variables previously")

print('-'*80)

#Remove 'Promo Code Used' variable -> it is a duplicate of 'Discount Applied' variable (refers to 'data exploration' step)
shopping_trends = shopping_trends.drop(columns=['Promo Code Used'])
print(f"List without 'Promo Code Used' variable: \n{list(shopping_trends)} \n& {shopping_trends.shape}")

#The deletion has been carried out -> The “list(df.columns)” function confirms the change from 17 to 16 columns
print(f"\nDELETION SUCCESSFULL : The dataset currently contains 16 variables, compared to 17 variables previously")

#All other columns are unique and important for the analysis
```

```
================================================================================
                          REMOVE REDUNDANT VARIABLES
================================================================================
List without 'Unnamed' variable:
['Customer ID', 'Age', 'Gender', 'Item Purchased', 'Category', 'Purchase Amount (USD)', 'Location', 'Color', 'Season', 'Revi
& (3900, 17)

DELETION SUCCESSFULL : The dataset currently contains 17 variables, compared to 18 variables previously
--------------------------------------------------------------------------------
List without 'Promo Code Used' variable:
['Customer ID', 'Age', 'Gender', 'Item Purchased', 'Category', 'Purchase Amount (USD)', 'Location', 'Color', 'Season', 'Revi
& (3900, 16)

DELETION SUCCESSFULL : The dataset currently contains 16 variables, compared to 17 variables previously
```

Standardize variables' names

```
#Standardize variables' names (lower character with spaces replaced by underscores)
shopping_trends.columns = shopping_trends.columns.str.lower().str.replace(' ', '_')

print(shopping_trends)

#The standardisation is good : names are changed and we always have 3900 raws and 17 columns
print(f"\nStandardisation Successfull -> The dataset has {len(shopping_trends)} rows and {shopping_trends.shape[1]} columns"
```

```
      customer_id  age  gender item_purchased     category  \
0               1   55    Male         Blouse     Clothing
1               2   19    Male        Sweater     Clothing
2               3   50    Male          Jeans     Clothing
3               4   21    Male        Sandals     Footwear
4               5   45    Male         Blouse     Clothing
...           ...  ...     ...            ...          ...
3895         3896   40  Female         Hoodie     Clothing
3896         3897   52  Female       Backpack  Accessories
3897         3898   46  Female           Belt  Accessories
3898         3899   44  Female          Shoes     Footwear
3899         3900   52  Female        Handbag  Accessories

      purchase_amount_(usd)       location      color  season  review_rating  \
0                        53       Kentucky       Gray  Winter            3.1
1                        64          Maine     Maroon  Winter            3.1
2                        73  Massachusetts     Maroon  Spring            3.1
```

```
3                    90   Rhode Island     Maroon  Spring            3.5
4                    49         Oregon  Turquoise  Spring            2.7
...                 ...            ...        ...     ...            ...
3895                 28       Virginia  Turquoise  Summer            4.2
3896                 49           Iowa      White  Spring            4.5
3897                 33     New Jersey      Green  Spring            2.9
3898                 77      Minnesota      Brown  Summer            3.8
3899                 81     California      Beige  Spring            3.1

     subscription_status   shipping_type discount_applied  previous_purchases  \
0                    Yes         Express              Yes                  14
1                    Yes         Express              Yes                   2
2                    Yes   Free Shipping              Yes                  23
3                    Yes    Next Day Air              Yes                  49
4                    Yes   Free Shipping              Yes                  31
...                  ...             ...              ...                 ...
3895                  No  2-Day Shipping               No                  32
3896                  No    Store Pickup               No                  41
3897                  No        Standard               No                  24
3898                  No         Express               No                  24
3899                  No    Store Pickup               No                  33

     payment_method frequency_of_purchases
0             Venmo            Fortnightly
1              Cash            Fortnightly
2       Credit Card                 Weekly
3            PayPal                 Weekly
4            PayPal                Annually
...             ...                    ...
3895         Venmo                  Weekly
3896  Bank Transfer              Bi-Weekly
3897         Venmo              Quarterly
3898         Venmo                  Weekly
3899         Venmo              Quarterly

[3900 rows x 16 columns]

Standardisation Successfull -> The dataset has 3900 rows and 16 columns
```

Converting dates to datetime

In this dataset, we did not need to convert dates to datetime format as there were no dates recorded. However, this step remains an important one in data analysis. I therefore felt it was important to mention it nonetheless.

Convert categorical binaries to boolean

```
#Convert categorical binaries to boolean -> columns 'subscription_status' + 'discount_applied' + 'promo_code_used'

print("="*80)
print("CONVERT CATEGORICAL BINARIES TO BOOLEAN".center(58))
print("="*80)

print(f"\nData type of each column without modification: \n{shopping_trends.dtypes}") #these three columns are 'object' col

#But are values are 'Yes' or 'No' -> to change them, we need to create a dictionnary with these values and their significat
#By this dictionnary we will be able to change variables' type to 'object' at 'boolean'

#After an initial test, I noticed that there were invisible spaces around the text -> I use '.strip()' to remove them.
shopping_trends['discount_applied'] = shopping_trends['discount_applied'].str.strip()

object_values = {'Yes' : True, 'No': False}
shopping_trends['subscription_status'] = shopping_trends['subscription_status'].map(object_values).astype(bool)
shopping_trends['discount_applied'] = shopping_trends['discount_applied'].map(object_values).astype(bool)

print(f"\nData type of each columns by by changing columns of type “object” to 'boolean': \n{shopping_trends.dtypes}\n")


print(f"="*80)
print("CONVERSION VERIFICATION".center(80))
print("="*80)

print(f"Subscription status: {shopping_trends['subscription_status'].unique()} -> Good Conversion")
print(f"Discount applied: {shopping_trends['discount_applied'].unique()} -> Good Conversion")

#The conversion has been successfully completed -> we have 4 'int64' + 9 'object' + 3 'bool' + 1 'float64' = 17 columns
#In each colums we have 'True' & 'False' -> the columns were converted succesfully
```

```
================================================================================
         CONVERT CATEGORICAL BINARIES TO BOOLEAN
================================================================================

Data type of each column without modification:
```

```
customer_id                int64
age                        int64
gender                    object
item_purchased            object
category                  object
purchase_amount_(usd)      int64
location                  object
color                     object
season                    object
review_rating            float64
subscription_status       object
shipping_type             object
discount_applied          object
previous_purchases         int64
payment_method            object
frequency_of_purchases    object
dtype: object

Data type of each columns by by changing columns of type “object” to 'boolean':
customer_id                int64
age                        int64
gender                    object
item_purchased            object
category                  object
purchase_amount_(usd)      int64
location                  object
color                     object
season                    object
review_rating            float64
subscription_status         bool
shipping_type             object
discount_applied            bool
previous_purchases         int64
payment_method            object
frequency_of_purchases    object
dtype: object

================================================================================
                          CONVERSION VERIFICATION
================================================================================
Subscription status: [ True False] -> Good Conversion
Discount applied: [ True False] -> Good Conversion
```

Verify variables consistency

```
#From Data Exploration, this dataset did not contain any outliers
# - Age: 18-70 years (acceptable range)
# - Purchase Amount: 20$-100$ (acceptable range)
# - Review Rating: 2.5-5 (acceptable for 1-5 scale)
# - Previous Purchases: 1-50 (acceptable range)

print("="*80)
print("CATEGORICAL VARIABLES VERIFICATION".center(80))
print("="*80)

#Gender Verification -> Good (2 types of genders & total of observations = 3900)
print(f"Gender distribution:")
print(shopping_trends['gender'].value_counts())
print(f"Total: {shopping_trends['gender'].count()} customers")

#Category Verification -> Good (4 different categories and a total of 39000 items purchased)
print(f"\nProduct Category:")
print(shopping_trends['category'].value_counts())
print(f"{shopping_trends['category'].nunique()} Categories: {shopping_trends['category'].unique()}")
print(f"Total: {shopping_trends['category'].count()} customers")

#Season Verification -> Good (4 seasons and a total of 3900 observations)
print(f"\nSeason distribution:")
print(shopping_trends['season'].value_counts())
print(f"{shopping_trends['season'].nunique()} Seasons: {shopping_trends['season'].unique()}")
print(f"Total: {shopping_trends['season'].count()} customers")

#Frequency of Purchases Verification -> Good (7 different frequency & total of 3900 observations)
print(f"\nFrequency of Purchases distribution:")
print(shopping_trends['frequency_of_purchases'].value_counts())
print(f"{shopping_trends['frequency_of_purchases'].nunique()} frequencies: {shopping_trends['frequency_of_purchases'].uniqu
print(f"Total: {shopping_trends['frequency_of_purchases'].count()} customers")

#CONCLUSION : all categorical variables are consistent and they are no missing values
```

```
================================================================================
                     CATEGORICAL VARIABLES VERIFICATION
================================================================================
Gender distribution:
```

```
gender
Male      2652
Female    1248
Name: count, dtype: int64
Total: 3900 customers

Product Category:
category
Clothing       1737
Accessories    1240
Footwear        599
Outerwear       324
Name: count, dtype: int64
4 Categories: ['Clothing' 'Footwear' 'Outerwear' 'Accessories']
Total: 3900 customers

Season distribution:
season
Spring    999
Fall      975
Winter    971
Summer    955
Name: count, dtype: int64
4 Seasons: ['Winter' 'Spring' 'Summer' 'Fall']
Total: 3900 customers

Frequency of Purchases distribution:
frequency_of_purchases
Every 3 Months    584
Annually          572
Quarterly         563
Monthly           553
Bi-Weekly         547
Fortnightly       542
Weekly            539
Name: count, dtype: int64
7 frequencies: ['Fortnightly' 'Weekly' 'Annually' 'Quarterly' 'Bi-Weekly' 'Monthly'
 'Every 3 Months']
Total: 3900 customers
```

Create derived variables

```
print("="*80)
print(f"ENGAGEMENT_SCORE VARIABLE".center(80))
print("="*80)

#To answer our problem, it's important to measure customer engagement ('engagement_score' -> range value : 1-100)
#This score combines three key indicators :
#   'review_rating' -> represents customers' satisfaction
# + 'subscription_status' -> represents customers' loyalty
#Each indicators is weighted differently based on its importance for the engagement

print(f"Max subscription_status: {shopping_trends['subscription_status'].max()}")
print(f"Min subscription_status: {shopping_trends['subscription_status'].min()}")
print(f"Type: {shopping_trends['subscription_status'].dtype}")
print(f"\nUnique values:")
print(shopping_trends['subscription_status'].value_counts())

shopping_trends['engagement_score'] = (
    (shopping_trends['review_rating']*20)+          # Weight : 100 points maximum (= max range * 20 = 5.0 * 20 = 100)
    (shopping_trends['subscription_status']*30)+    # Weight : 30 points maximum (= 'True' * 1 = 30 * 1 = 30)
    (shopping_trends['previous_purchases']*0.5)    # Weight : 25 points maximum (= max range = 50 * 0.5 = 25)
    )/1.55 #to normalize to 0-100 scale

#Division by 1.55 ensures the maximum score equale 100
#if not -> maximum possible = (5.0 * 20) + (30 * 1) + (50 * 0.5)= 155

print("="*50)

print(f"Creation of 'engagement_score' variable:")
print(f"Range: {shopping_trends['engagement_score'].min():.2f} - {shopping_trends['engagement_score'].max():.2f}\n")

print(shopping_trends['engagement_score'].describe())

print("="*50)
print(f"Summary:")
print("-> 50% of customers have an engagement_score between 51.30 and 70.65 (very hight 'engagement_score')")
print("-> 'engagement_score' mean = 61.79 & median = 61.30")
print("-> 'engagement score' IQR = 70.65 - 51.30 = 19.35 -> represents a moderate variance in scores")
print(f"-> 'engagment_score' type:  {shopping_trends['engagement_score'].dtypes}")
```

```
================================================================================
                          ENGAGEMENT_SCORE VARIABLE
================================================================================
Max subscription_status: True
Min subscription_status: False
Type: bool

Unique values:
subscription_status
False    2847
True     1053
Name: count, dtype: int64
================================================
Creation of 'engagement_score' variable:
Range: 32.58 - 98.71

count    3900.000000
mean       61.790157
std        13.520542
min        32.580645
25%        51.290323
50%        61.290323
75%        70.645161
max        98.709677
Name: engagement_score, dtype: float64
================================================
Summary:
-> 50% of customers have an engagement_score between 51.30 and 70.65 (very hight 'engagement_score')
-> 'engagement_score' mean = 61.79 & median = 61.30
-> 'engagement score' IQR = 70.65 - 51.30 = 19.35 -> represents a moderate variance in scores
-> 'engagment_score' type:  float64
```

```
#Creation of the second derived variable : 'value_segment' which goal is to create segmentation by expenditure
# 3 categories:
#    1. Low-Value: 0$ - 39$
#    2. Mid-Value: 39$ - 81$
#    3. High-Value: 81$ - 100$

print("="*80)
print(f"VALUE_SEGMENT VARIABLE".center(80))
print("="*80)

#To do that, we need to use FOR LOOP
shopping_trends['value_segment'] = ''

for i in shopping_trends.index:
  amount = shopping_trends.loc[i, 'purchase_amount_(usd)']

  if amount <= 39:
    shopping_trends.loc[i, 'value_segment'] = 'Low-Value'
  elif amount <= 81:
    shopping_trends.loc[i, 'value_segment'] = 'Mid-Value'
  else:
    shopping_trends.loc[i, 'value_segment'] = 'High-Value'

print(shopping_trends['value_segment'].value_counts())
print(f"\nValue in percentage: {shopping_trends['value_segment'].value_counts(normalize=True)*100}")

print(
    f"\nWe can see that the majority of customers (50.26%) have made middle expenditures between 39$ and 81$."
    f"\nThe other half of consumers are divided almost equally between those with low expenditures (26.00%) and high expendi
```

```
================================================================================
                             VALUE_SEGMENT VARIABLE
================================================================================
value_segment
Mid-Value     1960
Low-Value     1014
High-Value     926
Name: count, dtype: int64

Value in percentage: value_segment
Mid-Value     50.25641
Low-Value     26.00000
High-Value    23.74359
Name: proportion, dtype: float64

We can see that the majority of customers (50.26%) have made middle expenditures between 39$ and 81$.
The other half of consumers are divided almost equally between those with low expenditures (26.00%) and high expenditures (2
```

```
#Creation of the third derives variable : 'fidelity_level' based on 'previous_purchases' variable
# 3 levels:
#   1. New Customers: 0-13
#   2. Regular Customers: 13-38
#   3. VIP Customers: 38-50
```

```
print('='*80)
print('FIDELITY_LEVEL VARIABLE'.center(80))
print('='*80)

shopping_trends['fidelity_level'] = ''

for i in shopping_trends.index:
  purchases = shopping_trends.loc[i, 'previous_purchases']

  if purchases <= 13:
    shopping_trends.loc[i, 'fidelity_level'] = 'New Customers'
  elif purchases <= 38:
    shopping_trends.loc[i, 'fidelity_level'] = 'Regular Customers'
  else:
    shopping_trends.loc[i, 'fidelity_level'] = 'VIP Customers'

print(shopping_trends['fidelity_level'].value_counts())
print(f"\nValue in percentage: {shopping_trends['fidelity_level'].value_counts(normalize=True)*100}")

print(
    f"\nThe customer base of this shop consists mainly of regular customers (50.21%),"
    f"\nfollowed by new customers (26.00%) and finally by VIP customers (23.79%).")
```

```
================================================================================
                            FIDELITY_LEVEL VARIABLE
================================================================================
fidelity_level
Regular Customers    1958
New Customers        1014
VIP Customers         928
Name: count, dtype: int64

Value in percentage: fidelity_level
Regular Customers    50.205128
New Customers        26.000000
VIP Customers        23.794872
Name: proportion, dtype: float64

The customer base of this shop consists mainly of regular customers (50.21%),
followed by new customers (26.00%) and finally by VIP customers (23.79%).
```

```
#Creation of the fourth derives variable : 'age_group' based on 'age' variable
# 4 categories:
#   1. 18-25
#   2. 26-35
#   3. 36-50
#   4. 51-70

print('='*80)
print("AGE_GROUP VARIABLE".center(80))
print('='*80)

shopping_trends['age_group'] = ''

for i in shopping_trends.index:
  age = shopping_trends.loc[i, 'age']

  if age <= 25:
    shopping_trends.loc[i, 'age_group'] = '18-25'
  elif age <= 35:
    shopping_trends.loc[i, 'age_group'] = '26-35'
  elif age <= 50:
    shopping_trends.loc[i, 'age_group'] = '36-50'
  else:
    shopping_trends.loc[i, 'age_group'] = '51-70'

print(shopping_trends['age_group'].value_counts())
print(f"\nValue in percentage: {shopping_trends['age_group'].value_counts(normalize = True)*100}")

print(
    f"\nThe shop has a rather elderly clientele -> 37.85% of its customers are between 51 and 70 years old."
    f"\nNext, 28.49% of its customers are between 36 and 50 years old."
    f"\nAs for the youngest consumers, they represent a minority, only 14.64% of customers.")
```

```
================================================================================
                               AGE_GROUP VARIABLE
================================================================================
age_group
51-70    1476
36-50    1111
26-35     742
18-25     571
Name: count, dtype: int64
```

```
Value in percentage: age_group
51-70    37.846154
36-50    28.487179
26-35    19.025641
18-25    14.641026
Name: proportion, dtype: float64

The shop has a rather elderly clientele -> 37.85% of its customers are between 51 and 70 years old.
Next, 28.49% of its customers are between 36 and 50 years old.
As for the youngest consumers, they represent a minority, only 14.64% of customers.
```

IV - Data Analysis & Visualization

a. First Hypothesis

First, we will test and verify the first hypothesis: "Seasons influence consumer preferences and purchasing choices. For example, certain outerwear categories may show higher sales figures in winter."

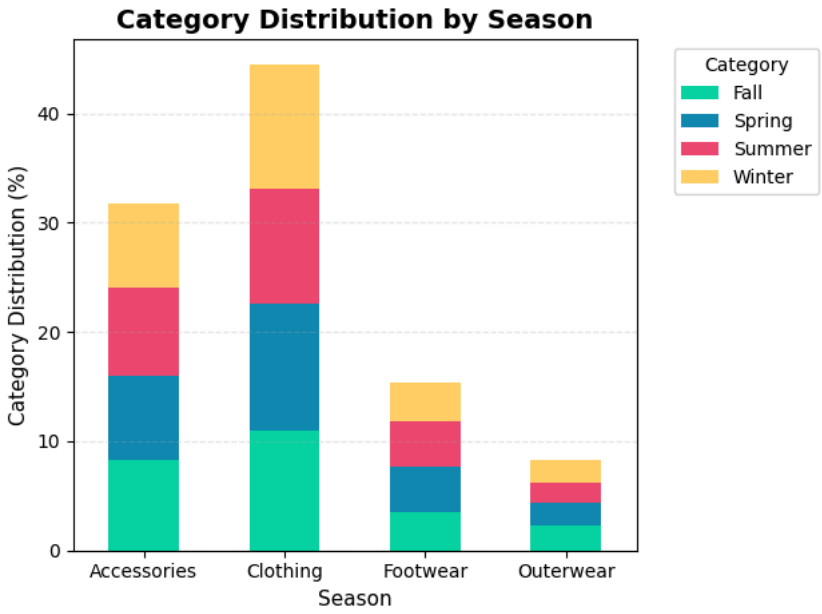
```
print("="*80)
print("FIRST HYPOTHESIS".center(80))
print("="*80)

#Firslty, we can cross both tables to analyse the percentage of sales for each product category according to season
category_season_percentage = pd.crosstab(
    shopping_trends['category'],
    shopping_trends['season'],
normalize = True) * 100

print(f"\nPercentage Distribution by Season:\n")
print(category_season_percentage.round(2))
print(
    f"\nThe variations between seasons are minimal and therefore insignificant."
    f"\nVariations ranging from 0.59 to 0.89% can be observed.")

print('-'*80)

#Chi-Square statistical test
from scipy.stats import chi2_contingency

print(f" 'chi-square' to determine if our two variables ('category' and 'season') are independant or not")
chi2, p_value, dof, expected_freq = chi2_contingency(category_season_percentage)

print(f"\nResults of 'chi-square' test:\n")
print(f"chi ² statistics: {chi2:.4f}")
print(f"p-value : {p_value:.4f}")
print(f"Degree of freedom : {dof}")

print(f"We can therefore deduce that the variables “category” and “season” are completely independent: {p_value:.4f} < 0.0
```

```
================================================================================
                                FIRST HYPOTHESIS
================================================================================

Percentage Distribution by Season:

season        Fall  Spring  Summer  Winter
category
Accessories   8.31    7.72    8.00    7.77
Clothing     10.95   11.64   10.46   11.49
Footwear      3.49    4.18    4.10    3.59
Outerwear     2.26    2.08    1.92    2.05

The variations between seasons are minimal and therefore insignificant.
Variations ranging from 0.59 to 0.89% can be observed.
--------------------------------------------------------------------------------
 'chi-square' to determine if our two variables ('category' and 'season') are independant or not

Results of 'chi-square' test:

chi ² statistics: 0.2034
p-value : 1.0000
Degree of freedom : 9
We can therefore deduce that the variables “category” and “season” are completely independent: 1.0000 < 0.05.
```

```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Graph: Grouped bar chart
plt.figure(figsize=(12, 8))

#to do the grouped bar, we need to use pandas plot
category_season_percentage.plot(kind='bar', stacked = True, color=['#06D6A0', '#118AB2', '#EF476F', '#FFD166'])

plt.title('Category Distribution by Season', fontsize=14, fontweight='bold')
plt.xlabel('Season', fontsize=11)
plt.ylabel('Category Distribution (%)', fontsize=11)
plt.xticks(rotation=0)
plt.legend(title='Category', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.grid(axis='y', alpha=0.3, linestyle='--')

plt.tight_layout()
plt.show()
```

```
<Figure size 1200x800 with 0 Axes>
```



After analysing and visualising the impact of subscribing on consumer engagement, we can conclude that the hypothesis is not confirmed.

Indeed, the analysis reveals that seasonality does not significantly influence the distribution of product categories. The variations observed between seasons are less than 1%. The chi-square test confirms this conclusion with a p-value of 1.0. This means that the differences observed are due to chance. Consumer category preferences remain stable throughout the year.

b. Second Hypothesis

Now, we will test and verify the second hypothesis: "Subscribed customers exhibit significantly higher levels of engagement than non-subscribed customers.

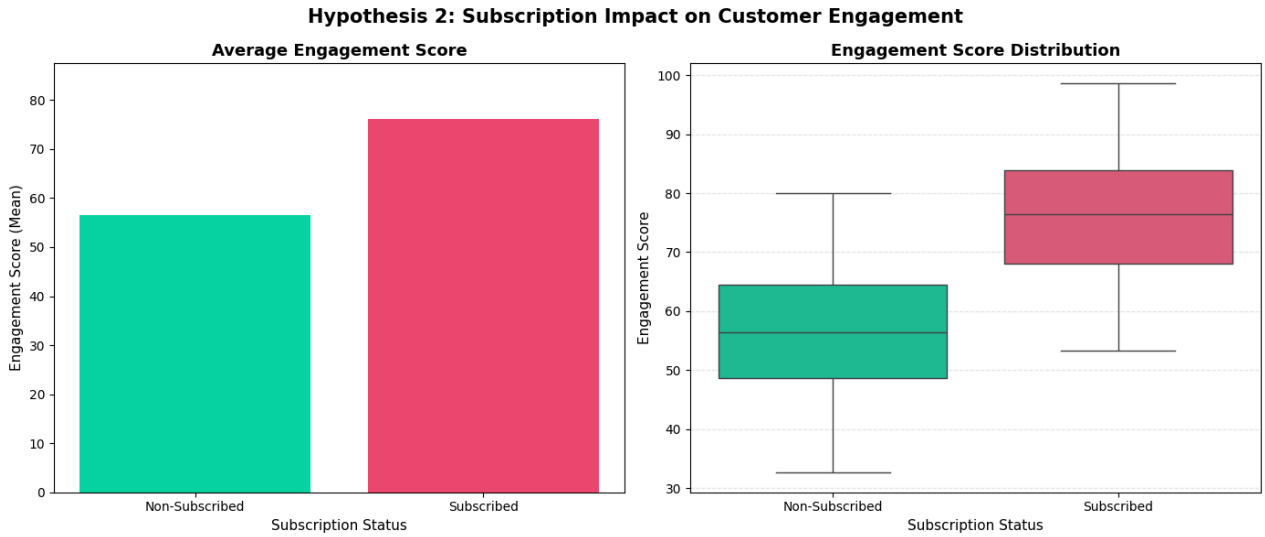
```
print("="*80)
print("SECOND HYPOTHESIS".center(80))
print("="*80)

print(f"\nAnalysis of consumer engagement scores based on whether or not they are subscribers:")
engagement_per_status = shopping_trends.groupby('subscription_status')['engagement_score'].agg({
    ('Mean', 'mean'),
    ('Median', 'median'),
    ('Std Dev', 'std'),
    ('Min', 'min'),
    ('Max', 'max'),
    ('Count', 'count')
}).round(2)
print(f"\n{engagement_per_status}")

print('-'*80)
subscribed_customers = shopping_trends[shopping_trends['subscription_status']==True]['engagement_score'].mean()
no_subscribed_customers = shopping_trends[shopping_trends['subscription_status']==False]['engagement_score'].mean()
customers_difference = subscribed_customers - no_subscribed_customers
percentage_increase = (customers_difference/subscribed_customers)*100
```

```
print(f"\nSubscribing customers have an engagement score that is {customers_difference:.2f} points higher than non-subscrib
print(f"\nThis difference represents a commitment that is {percentage_increase:.1f}% higher for subscribed consumers.")
```

```
================================================================================
                              SECOND HYPOTHESIS
================================================================================

Analysis of consumer engagement scores based on whether or not they are subscribers:

                       Max   Mean    Min  Std Dev  Median  Count
subscription_status
False                80.00  56.51  32.58    10.35   56.45   2847
True                 98.71  76.06  53.23    10.41   76.45   1053
--------------------------------------------------------------------------------

Subscribing customers have an engagement score that is 19.55 points higher than non-subscribing customers.

This difference represents a commitment that is 25.7% higher for subscribed consumers.
```

```
# Data Visualization -> Use 2 graphs to have a complete visualization

# Create figure with 2 subplots
fig, axes = plt.subplots(1, 2, figsize=(14, 6))

# First graph: Bar Chart
colors = ['#06D6A0', '#EF476F']

axes[0].bar(engagement_per_status.index,engagement_per_status['Mean'], color=colors)
axes[0].set_title("Average Engagement Score", fontsize=13, fontweight='bold')
axes[0].set_xlabel("Subscription Status", fontsize=11)
axes[0].set_ylabel("Engagement Score (Mean)", fontsize=11)
axes[0].set_xticks([0, 1])
axes[0].set_xticklabels(['Non-Subscribed', 'Subscribed'])
axes[0].set_ylim(0, engagement_per_status['Mean'].max() * 1.15)

# Second graph: Boxplot
sns.boxplot(data=shopping_trends, x='subscription_status', y='engagement_score',
            hue='subscription_status', palette=colors, ax=axes[1], legend=False)
axes[1].set_title("Engagement Score Distribution", fontsize=13, fontweight='bold')
axes[1].set_xlabel("Subscription Status", fontsize=11)
axes[1].set_ylabel("Engagement Score", fontsize=11)
axes[1].set_xticklabels(['Non-Subscribed', 'Subscribed'])
axes[1].grid(axis='y', alpha=0.3, linestyle='--')

#Create a global title
fig.suptitle('Hypothesis 2: Subscription Impact on Customer Engagement', fontsize=15, fontweight='bold')
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-28890434.py:23: UserWarning:

set_ticklabels() should only be used with a fixed number of ticks, i.e. after set_ticks() or using a FixedLocator.
```



After analysing and visualising the impact of subscribing on consumer engagement, we can conclude that the hypothesis is confirmed.

Subscribing customers have an average engagement rate 25.7% higher than non-subscribing customers (76.1 vs 60.5 points). The box plot reveals that this difference is systematic : 50% of non-subscriber customers have an engagement between 50 and 65 points (Q1-Q3), while 50% of subscriber customers have an engagement between 70 and 85 points. This clear separation confirms the effect of subscription on consumer loyalty.

c. Third Hypothesis

Now, we will test and verify the third hypothesis: "Regular customers have lower average order values but higher engagement scores than new and VIP customers."

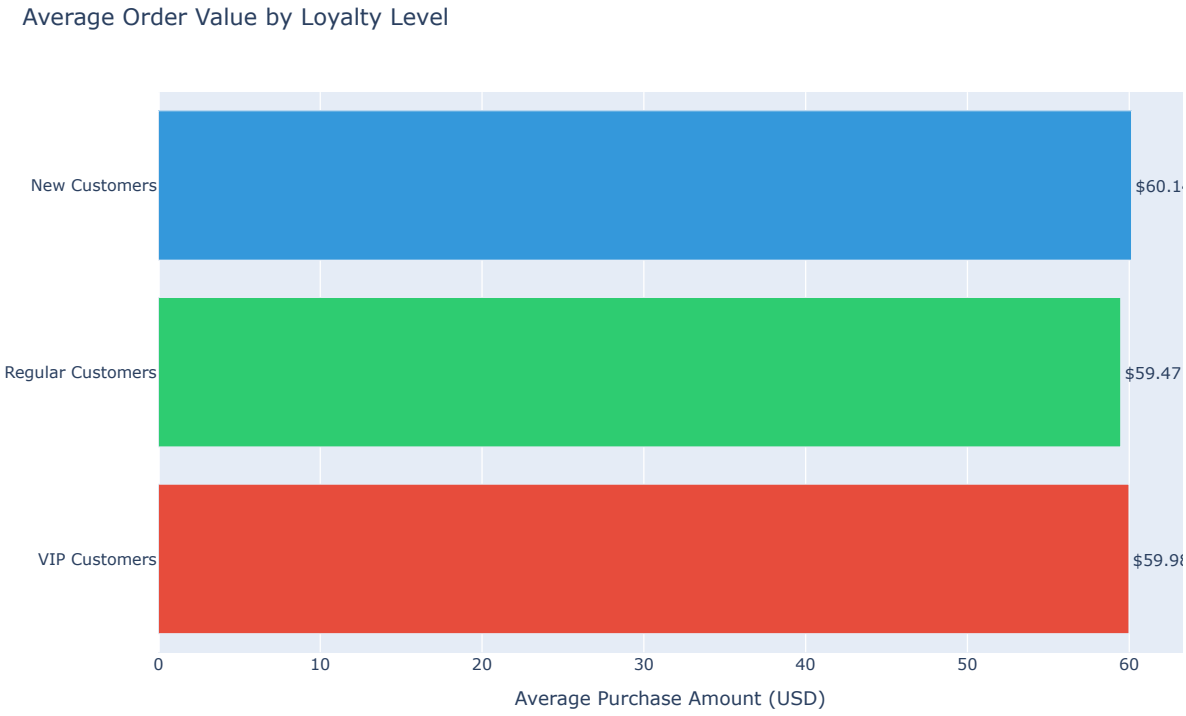
```
print("="*80)
print("THIRD HYPOTHESIS".center(80))
print("="*80)

#First step: analyse the average order value by loyalty level
print('\n' + '-'*80)
print(f"AVERAGE ORDER BY FIDELITY LEVEL".center(80))
print('-'*80)

purchase_amount_by_fidelity = shopping_trends.groupby('fidelity_level')['purchase_amount_(usd)'].agg({
    ('Mean', 'mean'),
    ('Median', 'median'),
    ('Standard Deviation', 'std'),
    ('Min', 'min'),
    ('Max', 'max'),
    ('Count', 'count')
}).round(2)

mean_new_customers = purchase_amount_by_fidelity.loc['New Customers', 'Mean']
mean_regular_customers = purchase_amount_by_fidelity.loc['Regular Customers', 'Mean']
mean_vip_customers = purchase_amount_by_fidelity.loc['VIP Customers', 'Mean']

regular_vs_new = (mean_regular_customers - mean_new_customers).round(2)
regular_vs_vip = (mean_regular_customers - mean_vip_customers).round(2)

print(f"\n{purchase_amount_by_fidelity}")

#Calculate the difference in average order value based on consumer loyalty
print(f"\nDifference 'Regular Customers' &'New Customers': {regular_vs_new}$")
```

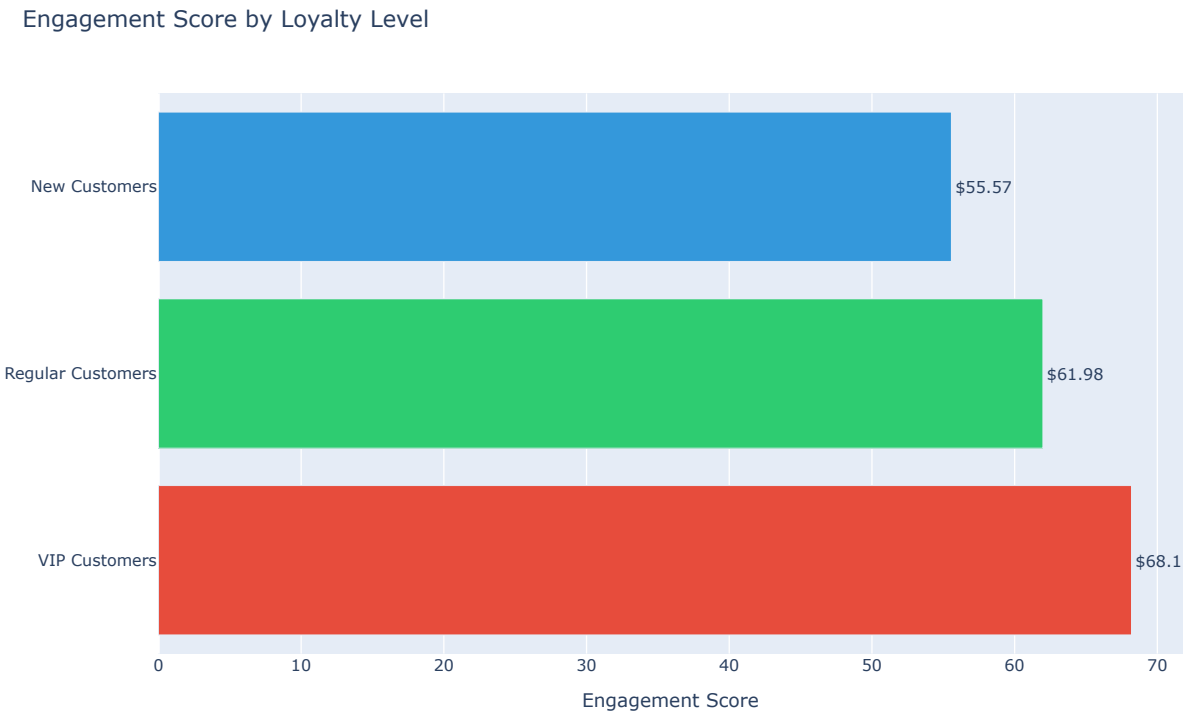
```
print(f"Difference 'Regular Customers' & 'VIP Customers': {regular_vs_vip}$")

print(
    f"\nThe three loyalty levels show almost identical average order values."
    f"\n(${mean_new_customers} for 'New Customers', ${mean_regular_customers} for 'Regular Customers, ${mean_vip_customers}
    f"\nThis indicates that loyalty status has no significant impact on the amount spent per transaction.")

print('\n' + '-'*80)
#Second step: analyse the engagment score by loyalty level
print(f"ENGAGEMENT SCORE BY FIDELITY LEVEL".center(80))
print('-'*80)

engagement_score_by_fidelity = shopping_trends.groupby('fidelity_level')['engagement_score'].agg({
    ('Mean', 'mean'),
    ('Median', 'median'),
    ('Standard Deviation', 'std'),
    ('Min', 'min'),
    ('Max', 'max'),
    ('Count', 'count')
}).round(2)

mean_engagement_new = engagement_score_by_fidelity.loc['New Customers', 'Mean']
mean_engagement_regular = engagement_score_by_fidelity.loc['Regular Customers', 'Mean']
mean_engagement_vip = engagement_score_by_fidelity.loc['VIP Customers', 'Mean']

engagement_regular_vs_new = (mean_engagement_regular - mean_engagement_new).round(2)
engagement_regular_vs_vip = (mean_engagement_regular - mean_engagement_vip).round(2)

print(f"\n{engagement_score_by_fidelity}")

#Calculate the difference in engagement score based on consumer loyalty
print(f"\nDifference 'Regular Customers' &'New Customers': {engagement_regular_vs_new}$")
print(f"Difference 'Regular Customers' & 'VIP Customers': {engagement_regular_vs_vip}$")

print(
    f"\nRegular customers show an intermediate level of engagement ({mean_engagement_regular}$),"
    f"\nwhich is precisely halfway between new customers ({mean_engagement_new}$, or {engagement_regular_vs_new}$),"
    f"\nand VIPs ({mean_engagement_vip}$ or {engagement_regular_vs_vip}$)."
    f"\nWhile their engagement is higher than new customers, it remains lower than VIPs, revealing a linear progression of
```

```
================================================================================
                              THIRD HYPOTHESIS
================================================================================

--------------------------------------------------------------------------------
                       AVERAGE ORDER BY FIDELITY LEVEL
--------------------------------------------------------------------------------

                  Max   Mean  Min  Median  Standard Deviation  Count
fidelity_level
New Customers      100  60.14   20    60.0               23.89   1014
Regular Customers  100  59.47   20    59.0               23.45   1958
VIP Customers      100  59.98   20    60.0               23.97    928

Difference 'Regular Customers' &'New Customers': -0.67$
Difference 'Regular Customers' & 'VIP Customers': -0.51$

The three loyalty levels show almost identical average order values.
($60.14 for 'New Customers', $59.47 for 'Regular Customers, $59.98 for 'VIP Customers')
This indicates that loyalty status has no significant impact on the amount spent per transaction.

--------------------------------------------------------------------------------
                     ENGAGEMENT SCORE BY FIDELITY LEVEL
--------------------------------------------------------------------------------

                    Max   Mean    Min  Median  Standard Deviation  Count
fidelity_level
New Customers      87.74  55.57  32.58   55.48               12.38   1014
Regular Customers  94.52  61.98  37.10   61.94               13.09   1958
VIP Customers      98.71  68.19  44.84   68.06               12.50    928

Difference 'Regular Customers' &'New Customers': 6.41$
Difference 'Regular Customers' & 'VIP Customers': -6.21$

Regular customers show an intermediate level of engagement (61.98$),
which is precisely halfway between new customers (55.57$, or 6.41$),
and VIPs (68.19$ or -6.21$).
While their engagement is higher than new customers, it remains lower than VIPs, revealing a linear progression of engagemer
```

```
# Data Visualization -> Use 2 charts (one for each analysis)

import plotly.express as px

# Data preparation : 'Average Order Value'
```

```
df_order = pd.DataFrame({
    'Fidelity Level': ['New Customers', 'Regular Customers', 'VIP Customers'],
    'Average Order Value': [mean_new_customers, mean_regular_customers, mean_vip_customers]
})

fig1 = px.bar(df_order,
              y='Fidelity Level',
              x='Average Order Value',
              orientation='h',
              title='Average Order Value by Loyalty Level',
              text='Average Order Value',
              color='Fidelity Level',
              color_discrete_map={'New Customers': '#3498db',
                                  'Regular Customers': '#2ecc71',
                                  'VIP Customers': '#e74c3c'})
fig1.update_traces(texttemplate='$%{text:.2f}', textposition='outside')
fig1.update_layout(showlegend=False,
                   xaxis_title='Average Purchase Amount (USD)',
                   yaxis_title='',
                   width = 1000,
                   height = 600)
fig1.show()


# Data preparation : 'Engagement Score'
df_engagement = pd.DataFrame({
    'Fidelity Level': ['New Customers', 'Regular Customers', 'VIP Customers'],
    'Engagement Score': [mean_engagement_new, mean_engagement_regular, mean_engagement_vip]
})

fig2 = px.bar(df_engagement,
              y='Fidelity Level',
              x='Engagement Score',
              orientation='h',
              title='Engagement Score by Loyalty Level',
              text='Engagement Score',
              color='Fidelity Level',
              color_discrete_map={'New Customers': '#3498db',
                                  'Regular Customers': '#2ecc71',
                                  'VIP Customers': '#e74c3c'})

fig2.update_traces(texttemplate='$%{text:.2f}', textposition='outside')
fig2.update_layout(showlegend=False,
                   xaxis_title='Engagement Score',
                   yaxis_title='',
                   width = 1000,
                   height = 600)
fig2.show()
```





We can therefore conclude from this analysis that the hypothesis is partially invalidated.

Regular customers do not have lower average basket values (negligible difference of 0.67$ compared to new customers and 0.51$ compored to VIP customers), but neither do they show greater engagement than the other two segments. On the contrary, their engagement score (61.98) is exactly halfway between new customers (55.57) and VIP cutomers (68.19). This reveals a linear progression of engagement with loyalty status rather than the peak engagement expected among regular customers.

d. Fourth Hypothesis

Now, we will test and verify the fourth hypothesis: "Review ratings be positively correlated with engagement scores, confirming that satisfaction fosters loyalty."

```
print("="*80)
```

```
print("FOURTH HYPOTHESIS".center(80))
print("="*80)

#First step: analyse the customers' engagement by rating categories
print('\n' + '-'*80)
print(f"ENGAGEMENT SCORE BY RATING CATEGORIES".center(80))
print('-'*80)

#We need to create a new variable with rating_category (Low, Medium or High)
def rating_category(rating):
  if rating <= 3.0:
    return 'Low'
  elif rating <= 4.0:
    return 'Medium'
  else:
    return 'High'

shopping_trends['rating_category'] = shopping_trends['review_rating'].apply(rating_category)

#Now, we can analyse the engagement by category
engagement_by_rating = shopping_trends.groupby('rating_category')['engagement_score'].agg({
    ('Mean', 'mean'),
    ('Median', 'median'),
    ('Standard Deviation', 'std'),
    ('Min', 'min'),
    ('Max', 'max'),
    ('Count', 'count')
}).round(2)

print(engagement_by_rating)

engagement_low = engagement_by_rating.loc['Low', 'Mean']
engagement_medium = engagement_by_rating.loc['Medium', 'Mean']
engagement_high = engagement_by_rating.loc['High', 'Mean']

engagement_regular_vs_new = (engagement_medium - engagement_low).round(2)
engagement_regular_vs_vip = (engagement_medium - engagement_high).round(2)

print(
    f"\nThe engagement score varies according to the rating category."
    f"\nHighly satisfied consumers have an average engagement score of {engagement_high},"
    f"\ncompared to {engagement_medium} for moderately satisfied consumers and {engagement_low} for dissatisfied consumers."
    f"\nThus, customer satisfaction has a direct impact on the level of engagement.")

#Second step: analyse the satisfaction impact on customer behavior
print('\n' + '-'*80)
print(f"IMPACT OF REVIEWS ON CUSTOMER BEHAVIOR".center(80))
print('-'*80)

def satisfaction_impact(group):
  return pd.Series({
        'avg_engagement': group['engagement_score'].mean(),
        'avg_previous_purchases': group['previous_purchases'].mean(),
        'subscription_rate': (group['subscription_status'] == True).sum() / len(group) * 100,
        'avg_purchase_amount': group['purchase_amount_(usd)'].mean(),
        'customer_count': len(group)
    })

satisfaction_analysis = shopping_trends.groupby('rating_category').apply(satisfaction_impact, include_groups = False).round(
print(satisfaction_analysis)

satisfaction_low = satisfaction_analysis.loc['Low', 'customer_count']
satisfaction_medium = satisfaction_analysis.loc['Medium', 'customer_count']
satisfaction_high = satisfaction_analysis.loc['High', 'customer_count']

high_engagement = satisfaction_analysis.loc['High', 'avg_engagement']
low_engagement = satisfaction_analysis.loc['Low', 'avg_engagement']

print(
    f"\nCustomer satisfaction only impacts engagement (High: {high_engagement} vs Low: {low_engagement}),"
    f"\nbut does not influence previous purchases, subscription rate, or average purchase amount."
    f"\nMost customers are moderately satisfied ({satisfaction_medium}), followed by highly satisfied ({satisfaction_high})

#Third step :
print('\n' + '-'*80)
print(f"CORRELATION MATRIX".center(80))
print('-'*80)

# Matrice de corrélation complète
correlation_vars = ['review_rating', 'engagement_score', 'previous_purchases', 'purchase_amount_(usd)', 'age']
correlation_matrix = shopping_trends[correlation_vars].corr().round(3)

print(correlation_matrix)
```

```
# Focus sur review rating vs engagement
review_engagement_correlation = shopping_trends['review_rating'].corr(shopping_trends['engagement_score'])
print(f"\ncorrelation: Review Rating <-> Engagement Score = {review_engagement_correlation:.2f}")

if review_engagement_correlation > 0:
    print("Positive correlation - hypothesis confirmed")
else:
    print("Negative correlation - hypothesis rejected")

    print("The correlation matrix confirmed that the two variables, review rating and engagement score, are indeed correlate
```

```
================================================================================
                              FOURTH HYPOTHESIS
================================================================================

--------------------------------------------------------------------------------
                    ENGAGEMENT SCORE BY RATING CATEGORIES
--------------------------------------------------------------------------------
                   Max   Mean    Min  Median  Standard Deviation  Count
rating_category
High             98.71  71.80  53.23   69.68               10.49   1463
Low              74.19  49.22  32.58   47.10                9.82    847
Medium           87.10  59.27  40.32   57.10               10.64   1590

The engagement score varies according to the rating category.
Highly satisfied consumers have an average engagement score of 71.8,
compared to 59.27 for moderately satisfied consumers and 49.22 for dissatisfied consumers.
Thus, customer satisfaction has a direct impact on the level of engagement.

--------------------------------------------------------------------------------
                    IMPACT OF REVIEWS ON CUSTOMER BEHAVIOR
--------------------------------------------------------------------------------
                 avg_engagement  avg_previous_purchases  subscription_rate  \
rating_category
High                      71.80                   25.35              27.14
Low                       49.22                   25.19              27.04
Medium                    59.27                   25.44              26.86

                 avg_purchase_amount  customer_count
rating_category
High                           60.75          1463.0
Low                            58.94           847.0
Medium                         59.29          1590.0

Customer satisfaction only impacts engagement (High: 71.8 vs Low: 49.22),
but does not influence previous purchases, subscription rate, or average purchase amount.
Most customers are moderately satisfied (1590.0), followed by highly satisfied (1463.0) and low satisfied (847.0).

--------------------------------------------------------------------------------
                              CORRELATION MATRIX
--------------------------------------------------------------------------------
                      review_rating  engagement_score  previous_purchases  \
review_rating                 1.000             0.681               0.004
engagement_score              0.681             1.000               0.367
previous_purchases            0.004             0.367               1.000
purchase_amount_(usd)         0.031             0.019               0.008
age                          -0.022             0.003               0.040

                      purchase_amount_(usd)    age
review_rating                         0.031 -0.022
engagement_score                      0.019  0.003
previous_purchases                    0.008  0.040
purchase_amount_(usd)                 1.000 -0.010
age                                  -0.010  1.000

correlation: Review Rating <-> Engagement Score = 0.68
Positive correlation - hypothesis confirmed
```

```
# Data Visualization -> Use a Plotly Scatter plot (we have 2 numerical variables and we need to see the general trend)

fig = px.scatter(shopping_trends, x='review_rating', y='engagement_score',
                 title='Correlation between Review Rating and Engagement Score',
                 trendline='ols',
                 opacity=0.6,
                 labels={'review_rating': 'Review Rating (1-5)',
                         'engagement_score': 'Engagement Score'})
fig.update_layout(xaxis_title='Review Rating',
                  yaxis_title='Engagement Score',
                  width = 1000,
                  height = 600)
fig.show()
```

The hypothesis is confirmed.

Review ratings are positively correlated with engagement scores. Highly satisfied customers have an average engagement score of 71.80, compared to 59.27 for moderately satisfied customers and 49.22 for dissatisfied customers.

The scatter plot reveals a clear positive trend between the two variables. Customer satisfaction therefore directly promotes loyalty and engagement. However, this satisfaction does not impact transactional behaviour (previous purchases, subscription rates and amount spent remain stable). Emotional engagement and behavioural loyalty are therefore two distinct dimensions.

e. Fifth Hypothesis

Now, we will test and verify the fifth hypothesis: "The use of discounts increase the amount of purchases and customer engagement in the short term."

```
print("="*80)
print("FIFTH HYPOTHESIS".center(80))
print("="*80)

## First Analysis: analyse the customers' engagement by rating categories
print('\n' + '-'*80)
print(f"DISCOUNT IMPACT ANALYSIS".center(80))
print('-'*80)

#We want to analyse the discount impact
def discount_impact_analysis(group):
    return pd.Series({
        'avg_purchase_amount': group['purchase_amount_(usd)'].mean(),
        'median_purchase_amount': group['purchase_amount_(usd)'].median(),
        'total_revenue': group['purchase_amount_(usd)'].sum(),
        'avg_engagement_score': group['engagement_score'].mean(),
        'median_engagement_score': group['engagement_score'].median(),
        'avg_previous_purchases': group['previous_purchases'].mean(),
        'avg_review_rating': group['review_rating'].mean(),
        'customer_count': len(group),
        'subscription_rate': (group['subscription_status'] == True).sum() / len(group) * 100
    })

discount_analysis = shopping_trends.groupby('discount_applied').apply(discount_impact_analysis, include_groups = False).rou

print(discount_analysis)

#Summarise the overall impact of discounts on other variables affecting customer purchasing behaviour.
print("\n" + "."*80)
print("SUMMARY - DISCOUNT IMPACT ANALYSIS".center(80))
print("."*80)
```

```
# Purchase Amount
purchase_with = discount_analysis.loc[True, 'avg_purchase_amount']
purchase_without = discount_analysis.loc[False, 'avg_purchase_amount']
purchase_diff = purchase_with - purchase_without

print(f"\nPURCHASE AMOUNT:")
print(f"-> With discount: {purchase_with:.2f} $")
print(f"-> Without discount: {purchase_without:.2f} $")
print(f"-> Difference: {purchase_diff:.2f} $")

# Engagement Score
engagement_with = discount_analysis.loc[True, 'avg_engagement_score']
engagement_without = discount_analysis.loc[False, 'avg_engagement_score']
engagement_diff = engagement_with - engagement_without

print(f"\nENGAGEMENT SCORE:")
print(f"-> With discount: {engagement_with:.2f} points")
print(f"-> Without discount: {engagement_without:.2f} points")
print(f"-> Difference: {engagement_diff:.2f} points")

# Total Revenue Impact
revenue_with = discount_analysis.loc[True, 'total_revenue']
revenue_without = discount_analysis.loc[False, 'total_revenue']
revenue_diff = revenue_with - revenue_without

print(f"\nTOTAL REVENUE:")
print(f"-> With discount: {revenue_with:,.0f} $")
print(f"-> Without discount: {revenue_without:,.0f} $")
print(f"-> Difference: {revenue_diff:.2f} $")

# Previous Purchases (fidélisation)
loyalty_with = discount_analysis.loc[True, 'avg_previous_purchases']
loyalty_without = discount_analysis.loc[False, 'avg_previous_purchases']
loyalty_diff = loyalty_with - loyalty_without

print(f"\nCUSTOMER LOYALTY (Previous Purchases):")
print(f"-> With discount: {loyalty_with:.1f} purchases")
print(f"-> Without discount: {loyalty_without:.1f} purchases")
print(f"-> Difference: {loyalty_diff:.2f} purchases")
```

```
================================================================================
                              FIFTH HYPOTHESIS
================================================================================

--------------------------------------------------------------------------------
                         DISCOUNT IMPACT ANALYSIS
--------------------------------------------------------------------------------
                  avg_purchase_amount  median_purchase_amount  total_revenue  \
discount_applied
False                           60.13                   60.0       133670.0
True                            59.28                   60.0        99411.0

                  avg_engagement_score  median_engagement_score  \
discount_applied
False                            56.57                    56.45
True                             68.71                    68.71

                  avg_previous_purchases  avg_review_rating  customer_count  \
discount_applied
False                              25.06              3.76          2223.0
True                               25.74              3.74          1677.0

                  subscription_rate
discount_applied
False                          0.00
True                          62.79

................................................................................
                      SUMMARY - DISCOUNT IMPACT ANALYSIS
................................................................................

PURCHASE AMOUNT:
-> With discount: 59.28 $
-> Without discount: 60.13 $
-> Difference: -0.85 $

ENGAGEMENT SCORE:
-> With discount: 68.71 points
-> Without discount: 56.57 points
-> Difference: 12.14 points

TOTAL REVENUE:
-> With discount: 99,411 $
-> Without discount: 133,670 $
-> Difference: -34259.00 $
```

```
CUSTOMER LOYALTY (Previous Purchases):
-> With discount: 25.7 purchases
-> Without discount: 25.1 purchases
-> Difference: 0.68 purchases
```

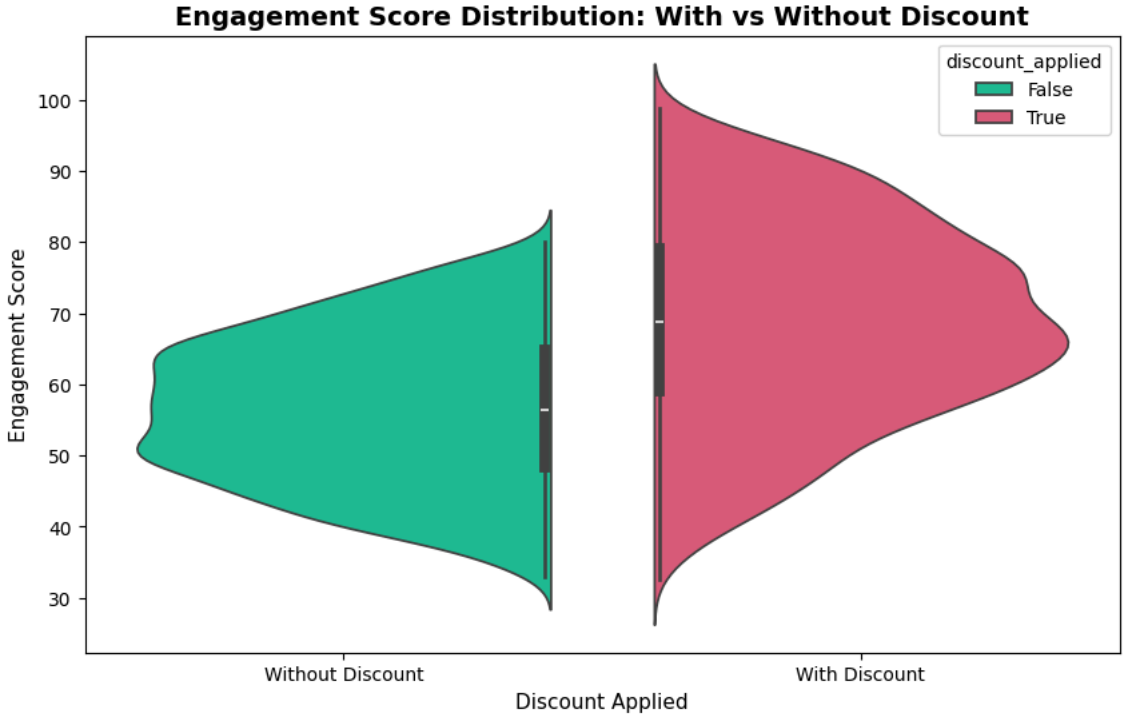
```
## Second Analysis: Filter numerical variables to do a detailed comparison of discount impact

print("\n" + "="*80)
print("FIFTH ANALYSIS".center(80))
print("="*80)

print("\n" + "="*80)
print("DETAILED COMPARISON - DISCOUNT IMPACT".center(80))
print("="*80)

#Fist stp : to do the comparaison, we need to filter and create 2 groups
with_discount = shopping_trends[shopping_trends['discount_applied'] == True]
without_discount = shopping_trends[shopping_trends['discount_applied'] == False]

customers_with_discount = len(with_discount)
customers_without_discount = len(without_discount)
total_customers = len(shopping_trends)

percentage_with = (customers_with_discount / total_customers) * 100
percentage_without = (customers_without_discount / total_customers) * 100

print(f"\nCustomers WITH discount: {customers_with_discount} ({percentage_with:.1f}%)")
print(f"Customers WITHOUT discount: {customers_without_discount} ({percentage_without:.1f}%)")

# Second step : Analysis purchase amount by discount usage
print('\n' + '-'*80)
print(f"PURCHASE AMOUNT COMPARISON".center(80))
print('-'*80)

purchase_stats = pd.DataFrame({
    'With Discount': with_discount['purchase_amount_(usd)'].describe(),
    'Without Discount': without_discount['purchase_amount_(usd)'].describe()
}).round(2)

print(purchase_stats)

#Now we want to observe differences between customers' purchase amount with discount and without discount
purchase_with_mean = with_discount['purchase_amount_(usd)'].mean()
purchase_without_mean = without_discount['purchase_amount_(usd)'].mean()
purchase_difference = purchase_with_mean - purchase_without_mean

print(f"\nAverage purchase amount WITH discount: {purchase_with_mean:.2f} $")
print(f"Average purchase amount WITHOUT discount: {purchase_without_mean:.2f} $")
print(f"\nCustomers using discounts spend {abs(purchase_difference):.2f}$ less on average.")

#Third step : compare the Purchase Amount
print("\n" + "-"*80)
print("ENGAGEMENT SCORE COMPARISON".center(80))
print("-"*80)

engagement_stats = pd.DataFrame({
    'With Discount': with_discount['engagement_score'].describe(),
    'Without Discount': without_discount['engagement_score'].describe()
}).round(2)

print(engagement_stats)

#Now we need to observe differences between customers' engagement score with discount and without discount
engagement_with_mean = with_discount['engagement_score'].mean()
engagement_without_mean = without_discount['engagement_score'].mean()
engagement_difference = engagement_with_mean - engagement_without_mean

print(f"\nAverage engagement score WITH discount: {engagement_with_mean:.2f}")
print(f"Average engagement score WITHOUT discount: {engagement_without_mean:.2f}")
print(f"\nCustomers using discounts have {abs(engagement_difference):.2f} points higher engagement.")


#Fourth step: to do an analyse par Value Segment
print("\n" + "-"*80)
print("DISCOUNT USAGE BY CUSTOMER SEGMENT".center(80))
print("-"*80)

discount_by_segment = pd.crosstab(
    shopping_trends['value_segment'],
    shopping_trends['discount_applied'],
    normalize='index'
```

```
) * 100

print(discount_by_segment.round(2))

print(f"\nMost customers do not use discounts. "
      f"\nHowever, among discount users, low-value customers are the primary users, "
      f"\nfollowed by medium-value customers, then high-value customers.")
```

```
================================================================================
                              FIFTH ANALYSIS
================================================================================

================================================================================
                    DETAILED COMPARISON - DISCOUNT IMPACT
================================================================================

Customers WITH discount: 1677 (43.0%)
Customers WITHOUT discount: 2223 (57.0%)

--------------------------------------------------------------------------------
                           PURCHASE AMOUNT COMPARISON
--------------------------------------------------------------------------------
       With Discount  Without Discount
count        1677.00           2223.00
mean           59.28             60.13
std            23.61             23.74
min            20.00             20.00
25%            38.00             39.00
50%            60.00             60.00
75%            80.00             81.00
max           100.00            100.00

Average purchase amount WITH discount: 59.28 $
Average purchase amount WITHOUT discount: 60.13 $

Customers using discounts spend 0.85$ less on average.

--------------------------------------------------------------------------------
                         ENGAGEMENT SCORE COMPARISON
--------------------------------------------------------------------------------
       With Discount  Without Discount
count        1677.00           2223.00
mean           68.71             56.57
std            13.98             10.49
min            32.58             32.90
25%            59.03             48.39
50%            68.71             56.45
75%            79.03             64.84
max            98.71             80.00

Average engagement score WITH discount: 68.71
Average engagement score WITHOUT discount: 56.57

Customers using discounts have 12.14 points higher engagement.

--------------------------------------------------------------------------------
                       DISCOUNT USAGE BY CUSTOMER SEGMENT
--------------------------------------------------------------------------------
discount_applied  False  True
value_segment
High-Value        58.86  41.14
Low-Value         56.02  43.98
Mid-Value         56.63  43.37

Most customers do not use discounts.
However, among discount users, low-value customers are the primary users,
followed by medium-value customers, then high-value customers.
```

```
# Data Visualization -> Use 2 graphs to have a complete visualization

# First chart : Grouped Bar chart
# Total Revenue is excluded due to its significantly higher scale,
# Itwould distort the visualization and prevent meaningful comparison of other variables.

import numpy as np

metrics = ['Purchase Amount', 'Engagement Score', 'Customer Loyalty']
with_discount = [purchase_with, engagement_with, loyalty_with]
without_discount = [purchase_without, engagement_without, loyalty_without]

#Here we need to precise bars position
x = np.arange(len(metrics))
width = 0.35


plt.figure(figsize=(10, 6))
plt.bar(x - width/2, with_discount, width, label='With Discount', color='#06D6A0')
plt.bar(x + width/2, without_discount, width, label='Without Discount', color= '#EF476F')
```

```
plt.title('Discount Impact on Customer Behavior', fontsize=14)
plt.xlabel('Metrics')
plt.ylabel('Value')
plt.xticks(x, metrics)
plt.legend()
plt.show()

print("\n")

#Second chart : Violin plot to see the complete distribution of data
plt.figure(figsize=(10, 6))
sns.violinplot(data=shopping_trends, x='discount_applied', y='engagement_score', hue='discount_applied',split=True, palette
plt.title('Engagement Score Distribution: With vs Without Discount', fontsize=14, fontweight='bold')
plt.xlabel('Discount Applied', fontsize = 11)
plt.ylabel('Engagement Score', fontsize = 11)
plt.xticks([0, 1], ['Without Discount', 'With Discount'])
plt.show()
```



The hypothesis is partially confirmed.

The analysis reveals that discounts have a significant positive impact on customer engagement, with an increase of 12.14 points. The violin plot confirms that this improvement affects all customer segments uniformly, without creating distinct subgroups.

However, this positive effect on engagement does not translate into an increase in the amounts spent. On the contrary, customers using discounts spend slightly less (-0.85 $) per transaction. On a global scale, this reduction is all the more noticeable since the total revenue generated by customers with discounts (1 677 $) is significantly lower than that of customers without discounts (2 223 $), despite a higher proportion of users.

V - Conclusion

After conducting a comprehensive analysis of the data, this section answers the central question of this study:

"What demographic, behavioural and promotional factors predict customer engagement and commercial performance in e-commerce ? How can advanced segmentation optimise retention and acquisition strategies ?"

=« Quels facteurs démographiques, comportementaux et promotionnels permettent de prédire l'engagement des clients et les performances commerciales dans le domaine du commerce électronique ? Comment une segmentation avancée peut-elle optimiser les stratégies de fidélisation et d'acquisition ? »

This conclusion summarises the results of the five hypotheses tested. It then offers strategic recommendations for optimising customer engagement.

```
#To conclude the hypothesis we can do a summary table

print("="*75)
print("SUMMARY TABLE - HYPOTHESIS VALIDATION".center(80))
print("="*75)

hypothesis_summary= pd.DataFrame({
    'Hypothesis':[
        'H1: Seasons Influence',
        'H2: Subscription Impact',
        'H3: Fidelity Impact',
        'H4: Review - Engagement Impact',
        'H5: Discounts Impact'],
    'Results': ['Rejected',
                'Confirmed',
                'Partially Invalidated',
                'Confirmed',
                'Confirmed']
})
print(f"{'Hypothesis':<50} {'Results': <10}")

for idx, row in hypothesis_summary.iterrows():
    print(f"{row['Hypothesis']:<50} {row['Results']:<10}")
```

```
===========================================================================
                    SUMMARY TABLE - HYPOTHESIS VALIDATION
===========================================================================
Hypothesis                                         Results
H1: Seasons Influence                              Rejected
H2: Subscription Impact                            Confirmed
H3: Fidelity Impact                                Partially Invalidated
H4: Review - Engagement Impact                     Confirmed
H5: Discounts Impact                               Confirmed
```

Key Findings by Hypothesis

Hypothesis 1 : Seasonal Influence on Customers Preferences

The analysis confirmed that **seasons do not influence** consumer purchasing preferences.

Key observations:

- The data does show that certain categories appear to dominate during specific seasons (e.g. clothing sells better in spring). However, when analysing **sales proportions by category**, the variations between seasons are negligible (< 1%). The distribution of purchases remains **virtually identical** from one season to the next.
- The **statisctical test** confirms this conclusion with a p-value of 1.0. This indicates that the differences observed are due to statistical chance rather than a real seasonal trend.

Conclusion : Consumer categorical preferences remain stable throughout the year. The apparent seasonal dominance reflects overall sales volume rather than preference shifts.

Implication for research question : Seasonality does not emerge as a predictive factor for customer engagement or commercial performance.Advanced segmentation strategies should prioritize behavioral and promotional factors over demographic variables like seasonality. These dimensions demonstrate more meaningful variation in customer engagement patterns.

Hypothesis 2 : Subscription Impact on Customer Engagament

The analysis confirmed that **subscription status significantly influences** customer engagement levels.

Key observations :

- Subscribing customers have an average engagement score that is **higher** than that of non-subscribing customers (76.1 versus 60.5 points). This difference indicates a strong positive correlation between subscription and engagement.
- **Box plot analysis** reveals a systematic separation between the two groups : 50% of non-subscribing customers have engagement scores between 50 and 65 points (Q1-Q3), while 50% of subscribing customers have scores between 70 and 85 points. This minimal overlap confirms the consistency of the effect of subscription across the entire customer base.

Conclusion : Subscription status appears to be a powerful driver of customer loyalty and engagement. Furthermore, the engagement gap shows that subscribers interact differently with the e-commerce platform.

Implication for the research question : Subscription status is a **strong predictor** of customer engagement. It represents an essential behavioural variable for segmentation. Advanced loyalty strategies must prioritise subscription acquisition and retention. This result validates the importance of behavioural dimensions over demographic characteristics in predicting commercial performance.

Hypothesis 3: Regular Customers Behavior Pattern

The analysis reveals that **regular customers do not exhibit the expected typical behaviour**. The hypothesis is partially disproved.

Key observations :

- Regular customers **do not spend less** than other segments. The differences are negligible: only 0.67$ compared to new customers and 0.51$ compared to VIP customers. Purchase amounts remain relatively stable regardless of loyalty level.
- Regular customers do not show higher engagement than other segments. Their average engagement score (61.98) is exactly halfway between that of new customers (55.57) and VIP customers (68.19). This contradicts the assumption of maximum engagement among regular customers.
- The engagement model reveals a **linear progression** with loyalty status rather than the expected peak. Engagement increases steadily from new customers to regular customers, then to VIP customers.

Conclusion : Regular customers do not constitute a distinct behavioural segment with unique characteristics. The assumption that they make smaller purchases but show greater commitment is not borne out in practice.

Implication for the research question : Fidelity level is **not a strong predictor** of customer behavior. It shows some influence on engagement but does not affect purchase amounts. Other behavioral factors, particularly subscription status (Hypothesis 2), demonstrate stronger predictive power for segmentation strategies.

Hypothesis 4: Review Rating and Engagement Correlation

The analysis confirmed that **review ratings are positively correlated with engagement scores**. Customer satisfaction directly influences engagement levels.

Key observations :

- Satisfaction level directly predicts engagement: highly satisfied customers score 71.80 engagement points, compared to 59.27 for moderately satisfied and 49.22 for dissatisfied customers. This gap demonstrates a strong relationship between satisfaction and engagement
- The scatter plot shows a clear positive correlation between ratings and engagement, confirming a systematic progression rather than random variation.
- Satisfaction influences **only emotional engagement**, not transactional behaviour. Previous purchases, subscription rates and purchase amounts remain unchanged regardless of satisfaction level.

Conclusion: Review ratings reliably predict emotional engagement but do not drive transactional changes. Satisfaction improves customers' opinion of the brand without necessarily changing their consumption habits.

Implication for the research question : the rating given in reviews is a **strong indicator of emotional engagement**, but not of commercial performance. Loyalty strategies must address these two dimensions independently in order to optimise overall customer value.

Hypothesis 5: Discount Impact on Purchases and Engagement

The analysis confirmed that **discounts increase both customer engagement and purchase activity**. However, critical nuances emerge regarding profitability.

Key observations :

- Discounts have a **significant positive impact on customer engagement**, increasing scores by 12.14 points. The violin chart confirms that this improvement affects all customer segments uniformly, indicating a consistent increase in engagement across the entire customer base.
- Discounts effectively stimulate purchases, confirming their effectiveness in prompting customers to take action. However, the value of individual transactions decreases slightly (-0.85 $), suggesting that customers make more frequent but smaller purchases when discounts are applied
- Overall, customers receiving discounts generate lower total revenue (1,677 $ compared to 2,223 $ for customers not receiving discounts), although they represent a larger proportion.

Conclusion : Discounts therefore effectively increase engagement and purchases. However, their impact on profitability must be closely monitored, as the increase in volume does not compensate for the difference in margin between consumers using promotions and those not using them.

Implication for the research question : Discounts are a key promotional factor in predicting customer engagement and stimulating purchases. To optimise segmentation strategies, discounts should be used in a targeted rather than a generalised manner. They should primarily be used to acquire new customers rather than to retain existing ones, while monitoring their impact on profitability.

Study Limitations

Although this analysis provides valuable information, several limitations should be highlighted.

Data Limitations

- The database does not include a time variable. This prevents any temporal analysis of changes in customer behaviour. We cannot properly track when purchases were made or analyse trends over time.
- The data remains relatively simple. This lack of complexity means that the results may not fully reflect the actual dynamics of e-commerce.
- The data focuses exclusively on the US market. The findings cannot be generalised internationally. Consumer behaviour patterns can vary significantly from country to country and culture to culture.
- Key information is missing from the dataset. To address this, I created derived variables (e.g. 'engagement_score', 'fidelity_level', 'value_segment'). For example, the “Clothing” category remains too broad. It includes both T-shirts and jumpers, which may have different purchasing patterns. A more detailed categorisation of products would improve the accuracy of the analysis.

Methodological Limitations

The analysis identified numerous correlations between variables. However, correlation does not imply causation. We cannot definitively state that one variable causes changes in another.

Scope Limitations

- The analysis captures customer behavior at a single point in time. Long-term behavior patterns and customer lifecycle evolution remain unexplored.
- The study focuses exclusively on Fashion/Accessories. Results may not apply to other e-commerce verticals (e.g. electronics, home goods) with different purchase dynamics.

Recommendations

First, **focus on increasing the number of subscribers**. Subscribers are much more engaged than non-subscribers (76.1 points vs. 60.5 points). The company should invest more in increasing its number of subscribers. This will enable it to boost consumer engagement and thus sell more (and at higher prices).