Project 2 Report

Customer Segmentation Using RFM Analysis

IE6400 18545 Foundations Data Analytics Eng

SEC 06 Fall 2024 [BOS-2-TR]

Group Members

Sai Pranav Krovvidi

Sneha Mishra

Mugdha Mishra

1. Introduction

Customer segmentation helps businesses tailor their marketing strategies effectively by grouping customers based on purchasing behavior. This report describes the process and results of applying RFM (Recency, Frequency, Monetary) analysis on the provided E-Commerce dataset.

2. Objectives

- Perform RFM analysis to derive RFM scores for each customer.
- Segment customers into distinct groups using clustering techniques.
- Provide actionable marketing recommendations based on the insights.

3. Methodology

The project workflow includes the following steps:

1. **Data Preprocessing**: Data cleaning, handling missing values, and formatting.
   Cleaned and formatted the dataset, addressing any missing values to ensure accurate analysis.

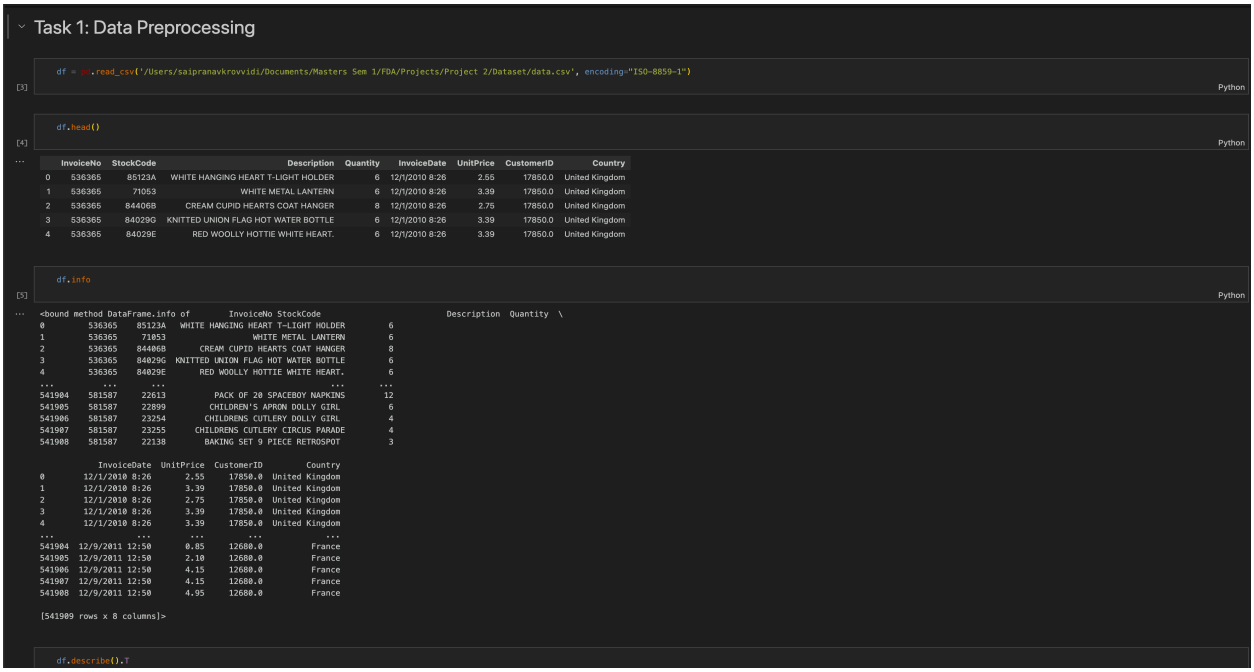
Handling Missing Values Strategy:

CustomerID (24.93% missing values)

The CustomerID column contains nearly a quarter of missing data. This column is essential for clustering customers and creating a recommendation system. Imputing such a large percentage of missing values might introduce significant bias or noise into the analysis.

Description (0.27% missing values)

The Description column has a minor percentage of missing values. However, it has been noticed that there are inconsistencies in the data where the same StockCode does not always have the same Description. This indicates data quality issues and potential errors in the product descriptions.



Task 1: Data Preprocessing

```
df = pd.read_csv('/Users/saipranavkrovvidi/Documents/Masters Sem 1/FDA/Projects/Project 2/Dataset/data.csv', encoding="ISO-8859-1")
```

[3] Python

```
df.head()
```

[4] Python

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850.0	United Kingdom

```
df.info
```

[5] Python

```
<bound method DataFrame.info of        InvoiceNo StockCode                          Description  Quantity  \
0         536365    85123A   WHITE HANGING HEART T-LIGHT HOLDER         6
1         536365     71053                  WHITE METAL LANTERN         6
2         536365    84406B       CREAM CUPID HEARTS COAT HANGER         8
3         536365    84029G  KNITTED UNION FLAG HOT WATER BOTTLE         6
4         536365    84029E       RED WOOLLY HOTTIE WHITE HEART.         6
...          ...       ...                                 ...       ...
541904    581587     22613          PACK OF 20 SPACEBOY NAPKINS        12
541905    581587     22899         CHILDREN'S APRON DOLLY GIRL         6
541906    581587     23254        CHILDRENS CUTLERY DOLLY GIRL         4
541907    581587     23255      CHILDRENS CUTLERY CIRCUS PARADE         4
541908    581587     22138        BAKING SET 9 PIECE RETROSPOT          3

            InvoiceDate  UnitPrice  CustomerID         Country
0        12/1/2010 8:26       2.55     17850.0  United Kingdom
1        12/1/2010 8:26       3.39     17850.0  United Kingdom
2        12/1/2010 8:26       2.75     17850.0  United Kingdom
3        12/1/2010 8:26       3.39     17850.0  United Kingdom
4        12/1/2010 8:26       3.39     17850.0  United Kingdom
...                 ...        ...         ...             ...
541904  12/9/2011 12:50       0.85     12680.0          France
541905  12/9/2011 12:50       2.10     12680.0          France
541906  12/9/2011 12:50       4.15     12680.0          France
541907  12/9/2011 12:50       4.15     12680.0          France
541908  12/9/2011 12:50       4.95     12680.0          France

[541909 rows x 8 columns]>
```

```
df.describe().T
```

```
df.describe().T
```

Python

	count	mean	std	min	25%	50%	75%	max
Quantity	541909.0	9.552250	218.081158	-80995.00	1.00	3.00	10.00	80995.0
UnitPrice	541909.0	4.611114	96.759853	-11062.06	1.25	2.08	4.13	38970.0
CustomerID	406829.0	15287.690570	1713.600303	12346.00	13953.00	15152.00	16791.00	18287.0

```
df.dtypes
```

Python

```
InvoiceNo       object
StockCode       object
Description     object
Quantity         int64
InvoiceDate     object
UnitPrice      float64
CustomerID     float64
Country         object
dtype: object
```

```
# Calculating the percentage of missing values for each column
print("\n Percentage of Missing values in each column:")
print((df.isnull().sum()/len(df))*100)
```

Python

```
 Percentage of Missing values in each column:
InvoiceNo       0.000000
StockCode       0.000000
Description     0.268311
Quantity        0.000000
InvoiceDate     0.000000
UnitPrice       0.000000
CustomerID     24.926694
Country         0.000000
dtype: float64
```

Handling Missing Values Strategy: CustomerID (24.93% missing values)

The CustomerID column contains nearly a quarter of missing data. This column is essential for clustering customers and creating a recommendation system. Imputing such a large percentage of missing values might introduce significant bias or noise into the analysis.

Description (0.27% missing values)

The Description column has a minor percentage of missing values. However, it has been noticed that there are inconsistencies in the data where the same StockCode does not always have the same Description. This indicates data quality issues and potential errors in the product descriptions.

```
print("\n Percentage of Missing values in each column:")
print((df.isnull().sum()/len(df))*100)
```

Python

```
 Percentage of Missing values in each column:
InvoiceNo      0.0
StockCode      0.0
Description    0.0
Quantity       0.0
InvoiceDate    0.0
UnitPrice      0.0
CustomerID     0.0
Country        0.0
dtype: float64
```

Handling Duplicate Values

```
# Finding duplicate rows (keeping all instances)
duplicate_rows = df[df.duplicated(keep=False)]

# Sorting the data by certain columns to see the duplicate rows next to each other
duplicate_rows_sorted = duplicate_rows.sort_values(by=['InvoiceNo', 'StockCode', 'Description', 'CustomerID', 'Quantity'])

# Displaying the first 10 records
duplicate_rows_sorted.head(10)
```

Python

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
494	536409	21866	UNION JACK FLAG LUGGAGE TAG	1	12/1/2010 11:45	1.25	17908.0	United Kingdom
517	536409	21866	UNION JACK FLAG LUGGAGE TAG	1	12/1/2010 11:45	1.25	17908.0	United Kingdom
485	536409	22111	SCOTTIE DOG HOT WATER BOTTLE	1	12/1/2010 11:45	4.95	17908.0	United Kingdom
539	536409	22111	SCOTTIE DOG HOT WATER BOTTLE	1	12/1/2010 11:45	4.95	17908.0	United Kingdom
489	536409	22866	HAND WARMER SCOTTY DOG DESIGN	1	12/1/2010 11:45	2.10	17908.0	United Kingdom
527	536409	22866	HAND WARMER SCOTTY DOG DESIGN	1	12/1/2010 11:45	2.10	17908.0	United Kingdom
521	536409	22900	SET 2 TEA TOWELS I LOVE LONDON	1	12/1/2010 11:45	2.95	17908.0	United Kingdom
537	536409	22900	SET 2 TEA TOWELS I LOVE LONDON	1	12/1/2010 11:45	2.95	17908.0	United Kingdom
578	536412	21448	12 DAISY PEGS IN WOOD BOX	1	12/1/2010 11:49	1.65	17920.0	United Kingdom
598	536412	21448	12 DAISY PEGS IN WOOD BOX	1	12/1/2010 11:49	1.65	17920.0	United Kingdom

```
# Displaying the number of duplicate rows
print(f"The dataset contains {df.duplicated().sum()} duplicate rows that need to be removed.")

# Removing duplicate rows
df.drop_duplicates(inplace=True)
```

Python

```
The dataset contains 5225 duplicate rows that need to be removed.
```

```
# Getting the number of rows in the dataframe
df.shape[0]
```

Python

```
401604
```

```
# Convert InvoiceDate to datetime
df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'])
```

Python

2. **RFM Metrics Calculation**:
   Calculated key RFM metrics where Recency measured the days since a customer's last purchase, Frequency counted the number of purchases, and Monetary represented total spending. These metrics were foundational for segmenting customers based on purchasing behavior.

- **Recency**: Days since the last purchase.

Recency (R)

```
current_date = df['InvoiceDate'].max()
```

```
recency = df.groupby('CustomerID')['InvoiceDate'].max().reset_index()
recency['Recency'] = (current_date - recency['InvoiceDate']).dt.days
recency.head()
```

	CustomerID	InvoiceDate	Recency
0	12346.0	2011-01-18 10:17:00	325
1	12347.0	2011-12-07 15:52:00	1
2	12348.0	2011-09-25 13:13:00	74
3	12349.0	2011-11-21 09:51:00	18
4	12350.0	2011-02-02 16:01:00	309

- **Frequency**: Number of orders per customer.

Frequency (f)

```
frequency = df.groupby('CustomerID')['InvoiceNo'].nunique().reset_index()
frequency.columns = ['CustomerID', 'Frequency']
frequency.head()
```

[22]

	CustomerID	Frequency
0	12346.0	2
1	12347.0	7
2	12348.0	4
3	12349.0	1
4	12350.0	1

- **Monetary**: Total spending of each customer.

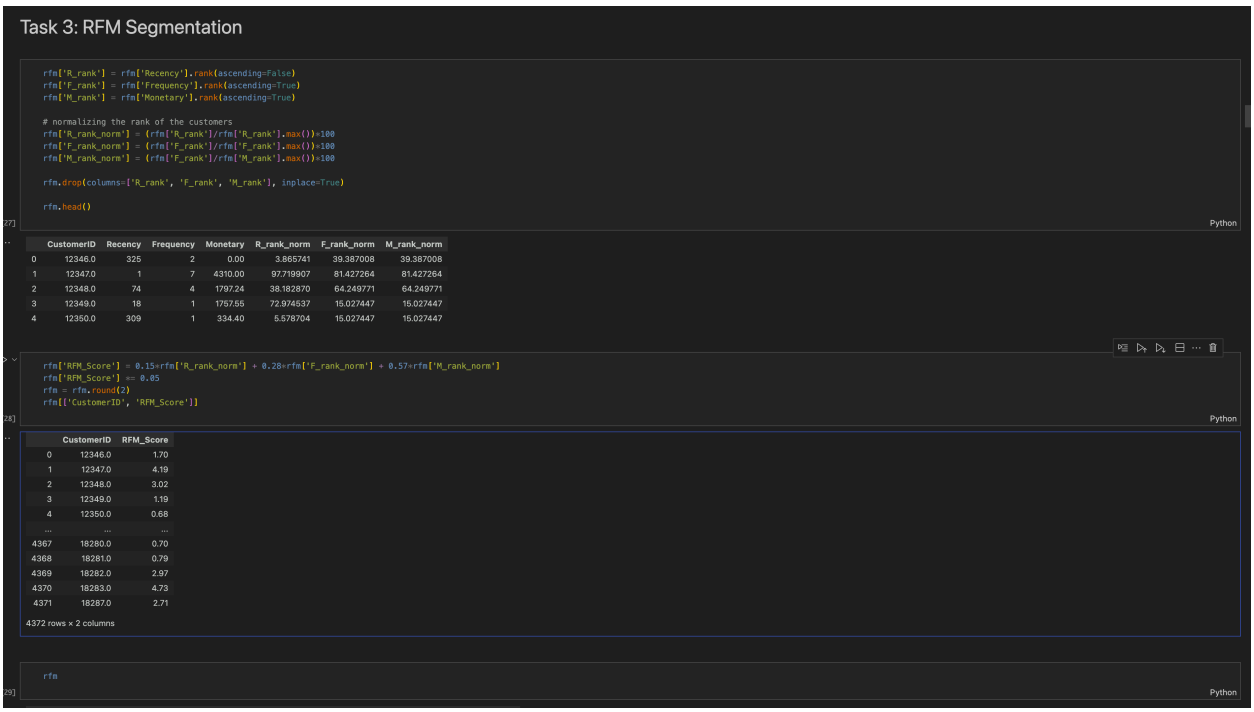
Monetary (m)

```
df['TotalPrice'] = df['Quantity'] * df['UnitPrice']
monetary = df.groupby('CustomerID')['TotalPrice'].sum().reset_index()
monetary.columns = ['CustomerID', 'Monetary']
monetary.head()
```

[24]

	CustomerID	Monetary
0	12346.0	0.00
1	12347.0	4310.00
2	12348.0	1797.24
3	12349.0	1757.55
4	12350.0	334.40

3. **RFM Segmentation**: Assign scores to RFM metrics and combine them into a unified score.
   By assigning scores to each RFM metric, We created a composite RFM score for each customer. This scoring system enabled the categorization of customers into distinct segments, facilitating targeted marketing strategies.



Task 3: RFM Segmentation

```
rfm['R_rank'] = rfm['Recency'].rank(ascending=False)
rfm['F_rank'] = rfm['Frequency'].rank(ascending=True)
rfm['M_rank'] = rfm['Monetary'].rank(ascending=True)

# normalizing the rank of the customers
rfm['R_rank_norm'] = (rfm['R_rank']/rfm['R_rank'].max())*100
rfm['F_rank_norm'] = (rfm['F_rank']/rfm['F_rank'].max())*100
rfm['M_rank_norm'] = (rfm['F_rank']/rfm['M_rank'].max())*100

rfm.drop(columns=['R_rank', 'F_rank', 'M_rank'], inplace=True)

rfm.head()
```

Python

	CustomerID	Recency	Frequency	Monetary	R_rank_norm	F_rank_norm	M_rank_norm
0	12346.0	325	2	0.00	3.865741	39.387008	39.387008
1	12347.0	1	7	4310.00	97.719907	81.427264	81.427264
2	12348.0	74	4	1797.24	38.182870	64.249771	64.249771
3	12349.0	18	1	1757.55	72.974537	15.027447	15.027447
4	12350.0	309	1	334.40	5.578704	15.027447	15.027447

```
rfm['RFM_Score'] = 0.15*rfm['R_rank_norm'] + 0.28*rfm['F_rank_norm'] + 0.57*rfm['M_rank_norm']
rfm['RFM_Score'] *= 0.05
rfm = rfm.round(2)
rfm[['CustomerID', 'RFM_Score']]
```

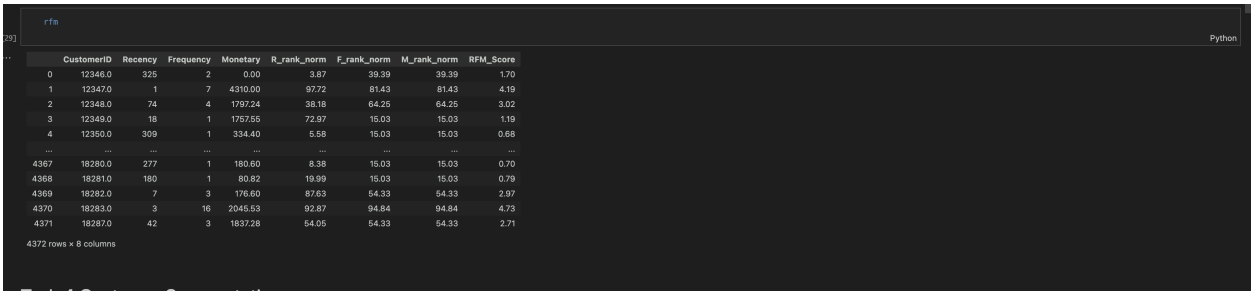
Python

	CustomerID	RFM_Score
0	12346.0	1.70
1	12347.0	4.19
2	12348.0	3.02
3	12349.0	1.19
4	12350.0	0.68
...	...	...
4367	18280.0	0.70
4368	18281.0	0.79
4369	18282.0	2.97
4370	18283.0	4.73
4371	18287.0	2.71

4372 rows × 2 columns

```
rfm
```

Python



```
rfm
```

Python

	CustomerID	Recency	Frequency	Monetary	R_rank_norm	F_rank_norm	M_rank_norm	RFM_Score
0	12346.0	325	2	0.00	3.87	39.39	39.39	1.70
1	12347.0	1	7	4310.00	97.72	81.43	81.43	4.19
2	12348.0	74	4	1797.24	38.18	64.25	64.25	3.02
3	12349.0	18	1	1757.55	72.97	15.03	15.03	1.19
4	12350.0	309	1	334.40	5.58	15.03	15.03	0.68
...	...	...	...	...	...	...	...	...
4367	18280.0	277	1	180.60	8.38	15.03	15.03	0.70
4368	18281.0	180	1	80.82	19.99	15.03	15.03	0.79
4369	18282.0	7	3	176.60	87.63	54.33	54.33	2.97
4370	18283.0	3	16	2045.53	92.87	94.84	94.84	4.73
4371	18287.0	42	3	1837.28	54.05	54.33	54.33	2.71

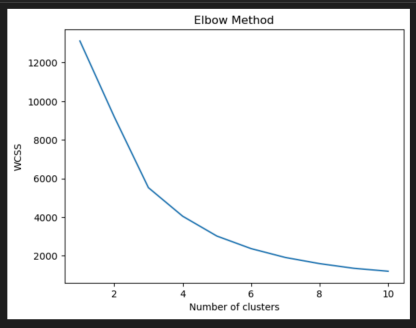
4372 rows × 8 columns

4. **Customer Clustering**: Apply clustering algorithms (e.g., K-Means) to segment customers.
   Applied clustering algorithms such as K-Means to group customers with similar purchasing patterns. This approach helped in identifying homogeneous segments that could be addressed with tailored marketing efforts.
   By experimenting with the k value we were able to conclude that 4 would be the optimal value in accordance to the elbow graph.

Task 4 Customer Segmentation

```
scaler = StandardScaler()
rfm_scaled = scaler.fit_transform(rfm[['Recency', 'Frequency', 'Monetary']])
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10, random_state=0)
    kmeans.fit(rfm_scaled)
    wcss.append(kmeans.inertia_)

plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()

k = 4

kmeans = KMeans(n_clusters=k, init='k-means++', max_iter=300, n_init=10, random_state=0)
kmeans.fit(rfm_scaled)

rfm['Cluster'] = kmeans.labels_
rfm
```

[31] Python



	CustomerID	Recency	Frequency	Monetary	R_rank_norm	F_rank_norm	M_rank_norm	RFM_Score	Cluster
0	12346.0	325	2	0.00	3.87	39.39	39.39	1.70	2
1	12347.0	1	7	4310.00	97.72	81.43	81.43	4.19	0

	CustomerID	Recency	Frequency	Monetary	R_rank_norm	F_rank_norm	M_rank_norm	RFM_Score	Cluster
0	12346.0	325	2	0.00	3.87	39.39	39.39	1.70	2
1	12347.0	1	7	4310.00	97.72	81.43	81.43	4.19	0
2	12348.0	74	4	1797.24	38.18	64.25	64.25	3.02	0
3	12349.0	18	1	1757.55	72.97	15.03	15.03	1.19	0
4	12350.0	309	1	334.40	5.58	15.03	15.03	0.68	2
...	...	...	...	...	...	...	...	...	...
4367	18280.0	277	1	180.60	8.38	15.03	15.03	0.70	2
4368	18281.0	180	1	80.82	19.99	15.03	15.03	0.79	2
4369	18282.0	7	3	176.60	87.63	54.33	54.33	2.97	0
4370	18283.0	3	16	2045.53	92.87	94.84	94.84	4.73	0
4371	18287.0	42	3	1837.28	54.05	54.33	54.33	2.71	0

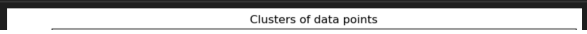
4372 rows × 9 columns

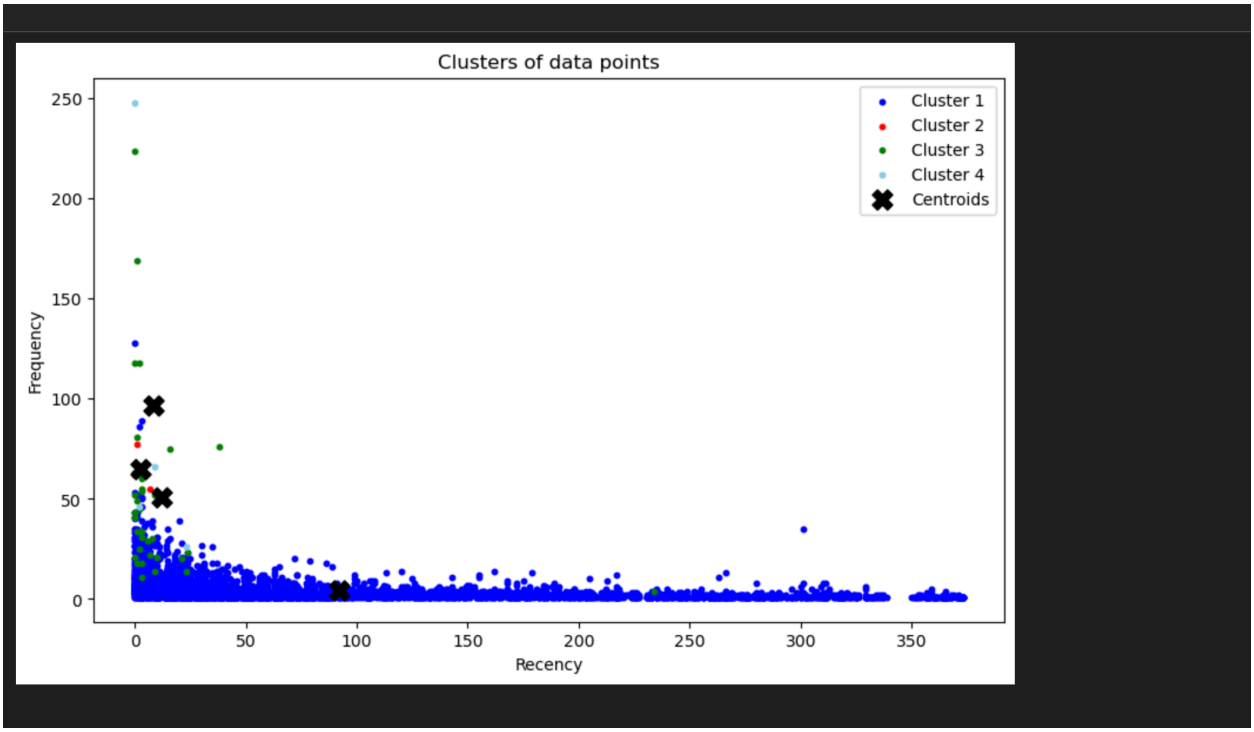
```
features = ['Recency', 'Frequency', 'Monetary']
X = rfm[features]
# Defining the number of clusters

k = 4

# Applying KMeans clustering
kmeans = KMeans(n_clusters=k, init='k-means++', random_state=42)
y_kmeans = kmeans.fit_predict(X)
```

[32] Python

```
# Visualizing the clusters and centroids
plt.figure(figsize=(10,6))
plt.scatter(X.iloc[y_kmeans == 0, 0], X.iloc[y_kmeans == 0, 1], s=10, c='blue', label='Cluster 1')
plt.scatter(X.iloc[y_kmeans == 1, 0], X.iloc[y_kmeans == 1, 1], s=10, c='red', label='Cluster 2')
plt.scatter(X.iloc[y_kmeans == 2, 0], X.iloc[y_kmeans == 2, 1], s=10, c='green', label='Cluster 3')
plt.scatter(X.iloc[y_kmeans == 3, 0], X.iloc[y_kmeans == 3, 1], s=10, c='skyblue', label='Cluster 4')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s=150, c='black', marker='X', label='Centroids')
plt.legend()
plt.xlabel('Recency')
plt.ylabel('Frequency')
plt.title("Clusters of data points")
plt.show()
```

[33] Python





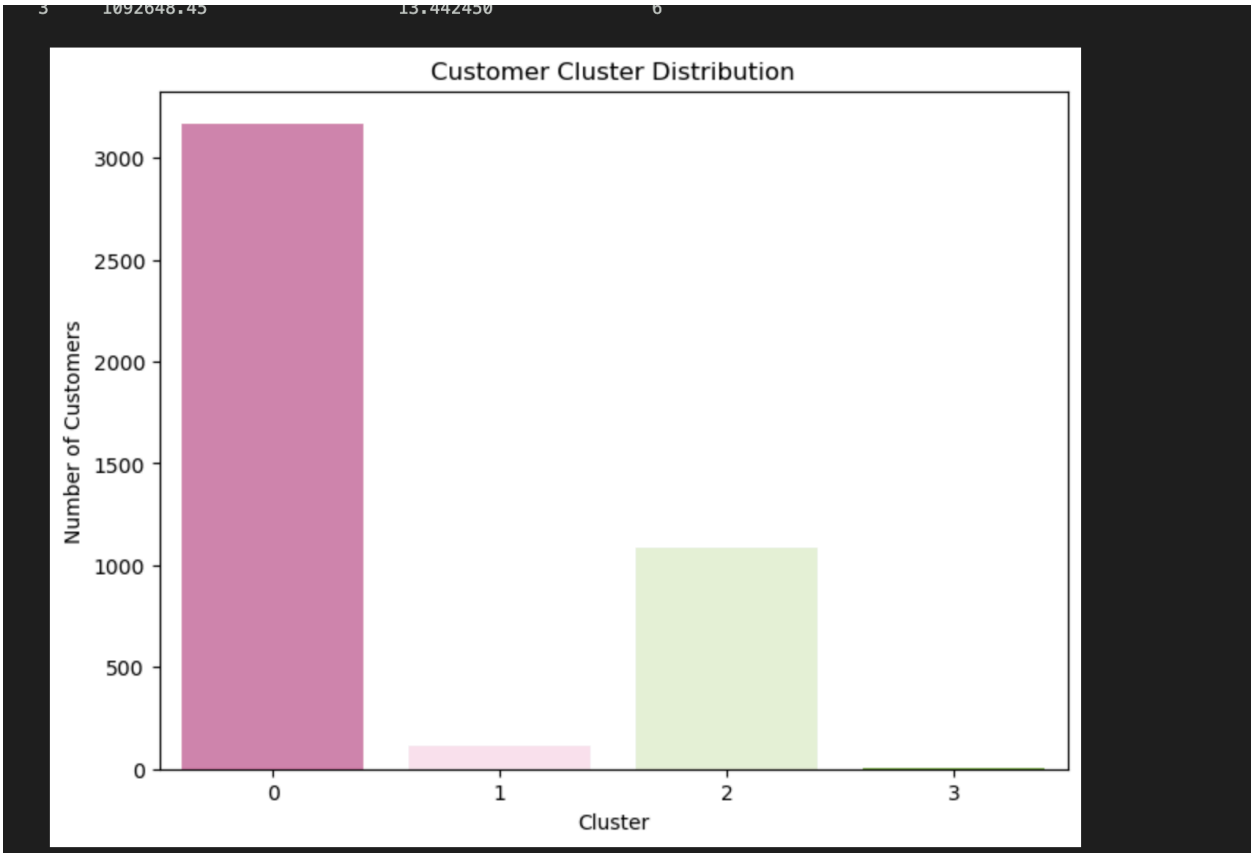
5. **Segment Profiling**: Analyze customer segments and their characteristics.
   Conducted an in-depth analysis of each customer segment to understand their unique characteristics and preferences. This profiling provided insights into how different segments contribute to overall business performance.

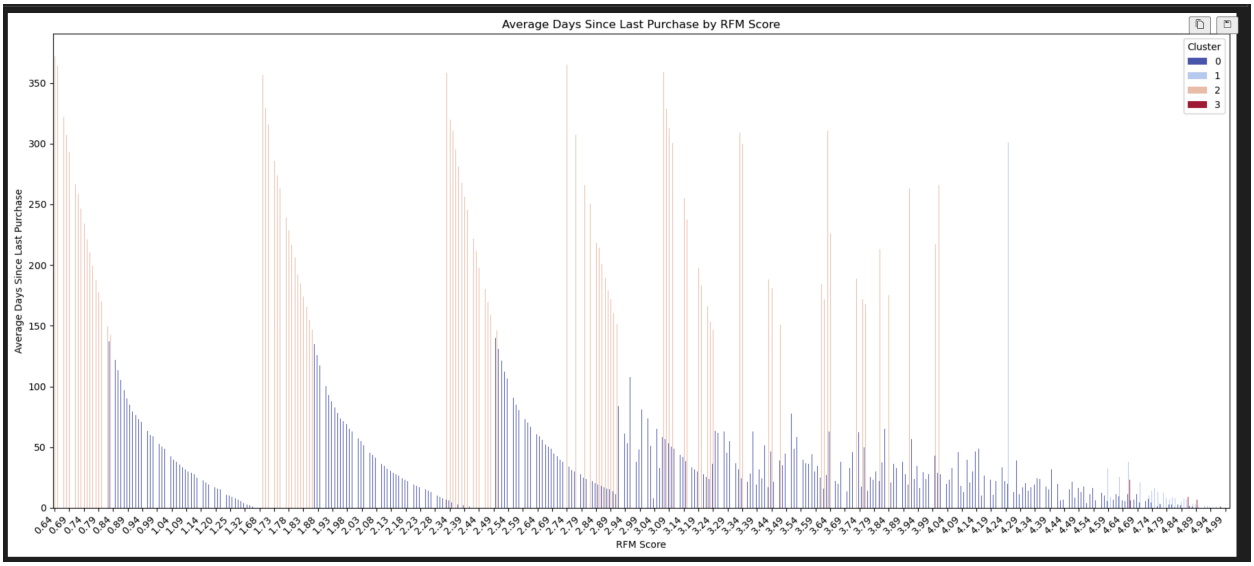
```
rfm["Customer_segment"] = np.where(rfm['RFM_Score'] >4.5, "Top Customers",
                                   (np.where(rfm['RFM_Score'] > 4,"High value Customer",
                                   (np.where(rfm['RFM_Score'] > 3,"Medium Value Customer",
                                   np.where(rfm['RFM_Score'] > 1.6, 'Low Value Customers', 'Lost Customers'))))))
rfm[['CustomerID', 'RFM_Score', 'Customer_segment']].head(20)
```

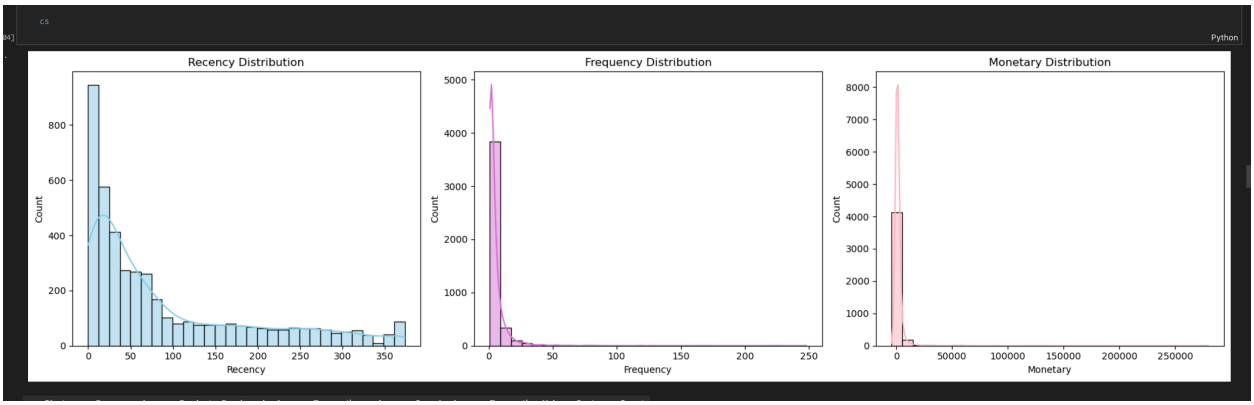
[35]

	CustomerID	RFM_Score	Customer_segment
0	12346.0	1.70	Low Value Customers
1	12347.0	4.19	High value Customer
2	12348.0	3.02	Medium Value Customer
3	12349.0	1.19	Lost Customers
4	12350.0	0.68	Lost Customers
5	12352.0	4.27	High value Customer
6	12353.0	0.77	Lost Customers
7	12354.0	0.74	Lost Customers
8	12355.0	0.76	Lost Customers
9	12356.0	2.83	Low Value Customers
10	12357.0	1.09	Lost Customers
11	12358.0	2.41	Low Value Customers
12	12359.0	3.95	Medium Value Customer
13	12360.0	2.68	Low Value Customers
14	12361.0	0.70	Lost Customers
15	12362.0	4.65	Top Customers
16	12363.0	1.90	Low Value Customers
17	12364.0	3.39	Medium Value Customer
18	12365.0	2.36	Low Value Customers
19	12367.0	1.34	Lost Customers

6. **Visualization**: Generate graphs and heatmaps for insights.
   Generated various graphs to visually represent the insights derived from the analysis. These visualizations made it easier to communicate findings and support decision-making processes.







Analysis Questions

1. **Data Overview**
   - What is the size of the dataset in terms of the number of rows and columns?
   - Can you provide a brief description of each column in the dataset?
   - What is the period covered by this dataset?

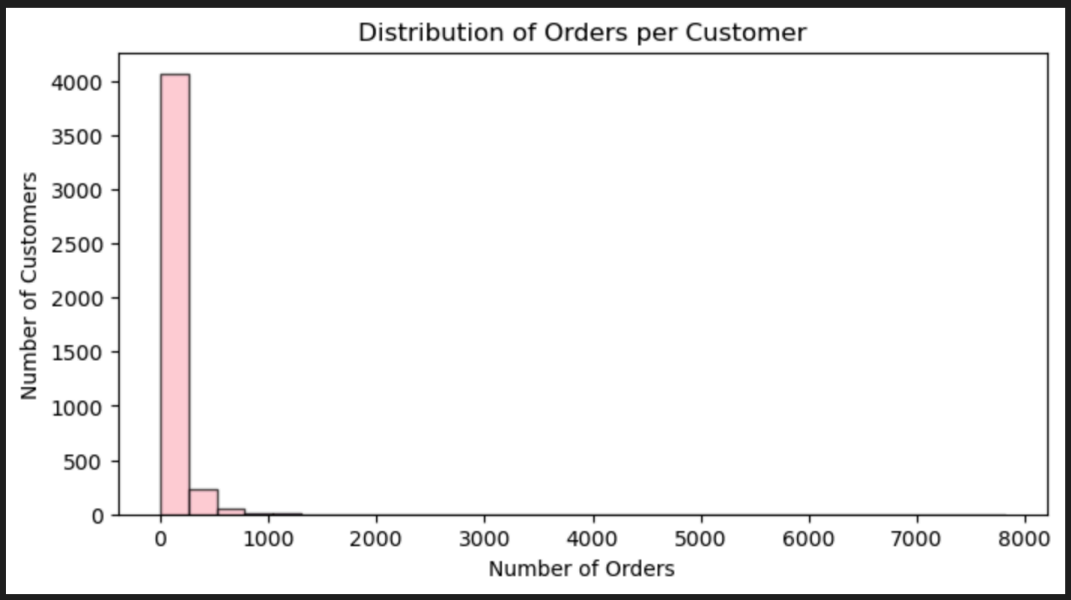
Assessed the dataset's dimensions, including its size in terms of rows and columns, and provided detailed descriptions of each column. Additionally, evaluated the time-period covered by the dataset to understand its temporal scope.

```
Size of the dataset: 401604 rows and 9 columns


Column Descriptions:
       InvoiceNo StockCode                      Description       Quantity  \
count     401604    401604                           401604  401604.000000
unique     22190      3684                             3896            NaN
top       576339    85123A  WHITE HANGING HEART T-LIGHT HOLDER            NaN
freq         542      2065                             2058            NaN
mean         NaN       NaN                              NaN      12.183273
min          NaN       NaN                              NaN  -80995.000000
25%          NaN       NaN                              NaN       2.000000
50%          NaN       NaN                              NaN       5.000000
75%          NaN       NaN                              NaN      12.000000
max          NaN       NaN                              NaN   80995.000000
std          NaN       NaN                              NaN     250.283037

                         InvoiceDate      UnitPrice     CustomerID  \
count                         401604  401604.000000  401604.000000
unique                           NaN            NaN            NaN
top                              NaN            NaN            NaN
freq                             NaN            NaN            NaN
mean    2011-07-10 12:08:23.848567552       3.474064   15281.160818
min               2010-12-01 08:26:00       0.000000   12346.000000
25%               2011-04-06 15:02:00       1.250000   13939.000000
...
std                NaN     430.352218


Time period covered by the dataset: From 2010-12-01 08:26:00 to 2011-12-09 12:50:00
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

2. **Customer Analysis**
   • How many unique customers are there in the dataset?
   • What is the distribution of the number of orders per customer?
   • Can you identify the top 5 customers who have made the most purchases by order count?

Identified the total number of unique customers and analyzed the distribution of orders per customer. Furthermore, we pinpointed the top 5 customers based on order count, highlighting key contributors to sales.

```
Number of unique customers: 4372
Distribution of orders per customer:
CustomerID
17841.0    7812
14911.0    5898
14096.0    5128
12748.0    4459
14606.0    2759
           ...
18068.0       1
13256.0       1
15590.0       1
16138.0       1
15389.0       1
Name: count, Length: 4372, dtype: int64
```

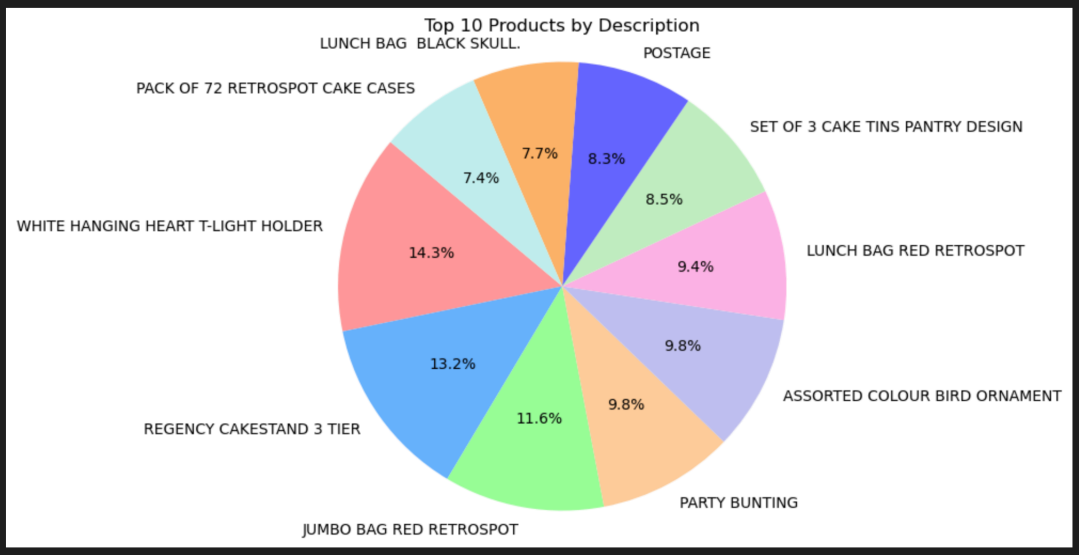



```
Top 5 customers with the most purchases by order count:
CustomerID
17841.0    7812
14911.0    5898
14096.0    5128
12748.0    4459
14606.0    2759
Name: count, dtype: int64
```

3. **Product Analysis**
   - What are the top 10 most frequently purchased products?
   - What is the average price of products in the dataset?
   - Can you find out which product category generates the highest revenue?

Listed the top 10 most frequently purchased products through pie-chart and calculated their average prices. Additionally, identified which product category generated the highest revenue, providing insights into product performance.

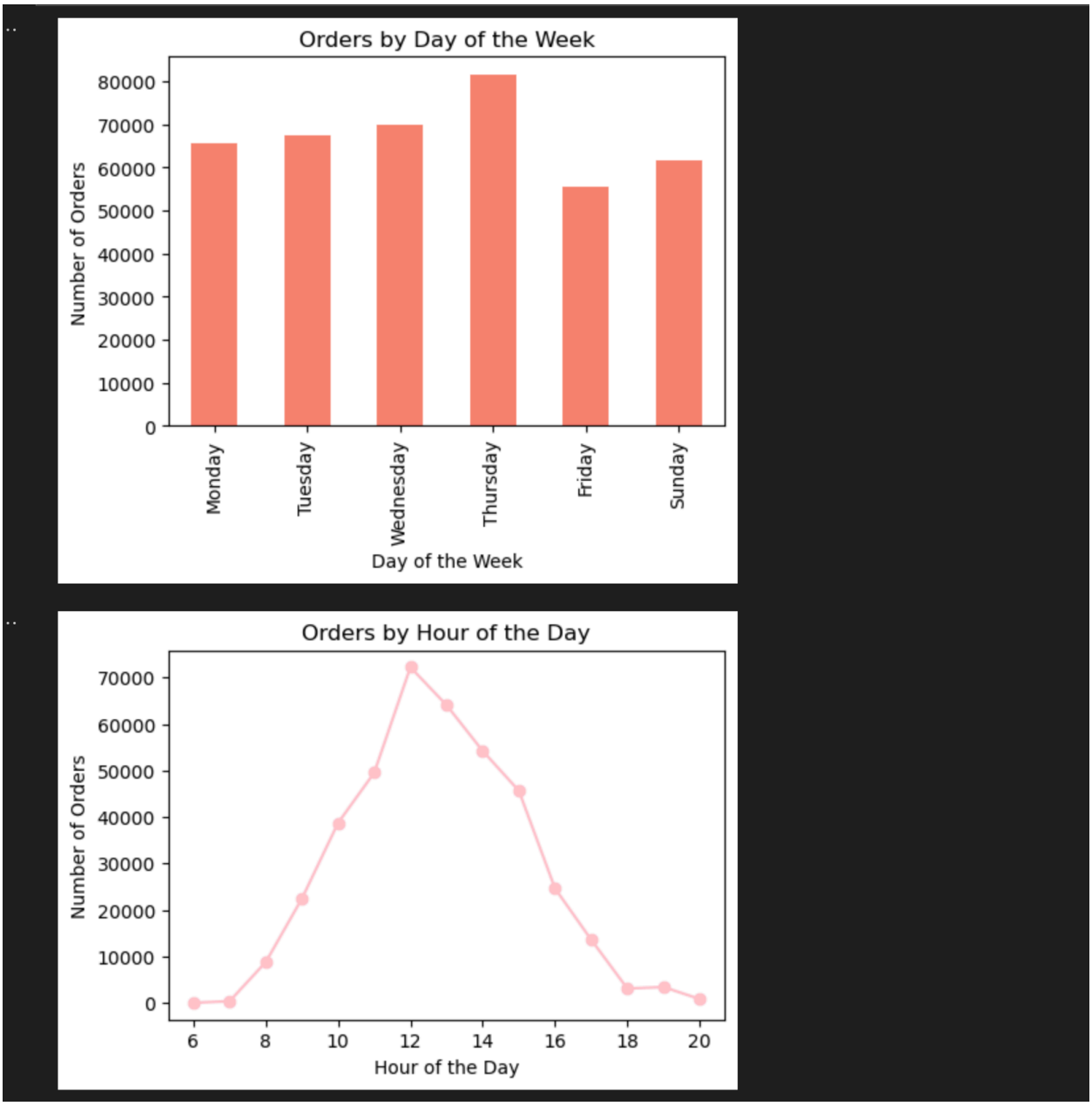
```
Description
WHITE HANGING HEART T-LIGHT HOLDER    2058
REGENCY CAKESTAND 3 TIER              1894
JUMBO BAG RED RETROSPOT               1659
PARTY BUNTING                         1409
ASSORTED COLOUR BIRD ORNAMENT         1405
LUNCH BAG RED RETROSPOT               1345
SET OF 3 CAKE TINS PANTRY DESIGN      1224
POSTAGE                               1196
LUNCH BAG  BLACK SKULL.               1099
PACK OF 72 RETROSPOT CAKE CASES       1062
Name: count, dtype: int64
```




```
Average Price of Products: Description
 4 PURPLE FLOCK DINNER CANDLES       2.312162
 50'S CHRISTMAS GIFT BAG LARGE       1.248091
 DOLLY GIRL BEAKER                   1.243796
 I LOVE LONDON MINI BACKPACK         4.138406
 I LOVE LONDON MINI RUCKSACK         4.150000
                                        ...
ZINC T-LIGHT HOLDER STARS SMALL      0.836888
ZINC TOP  2 DOOR WOODEN SHELF       16.768182
ZINC WILLIE WINKIE  CANDLE STICK     0.872344
ZINC WIRE KITCHEN ORGANISER          7.175000
ZINC WIRE SWEETHEART LETTER TRAY     3.454000
Name: UnitPrice, Length: 3896, dtype: float64
```

4. **Time Analysis**
   - Is there a specific day of the week or time of day when most orders are placed?
   - What is the average order processing time?
   - Are there any seasonal trends in the dataset?

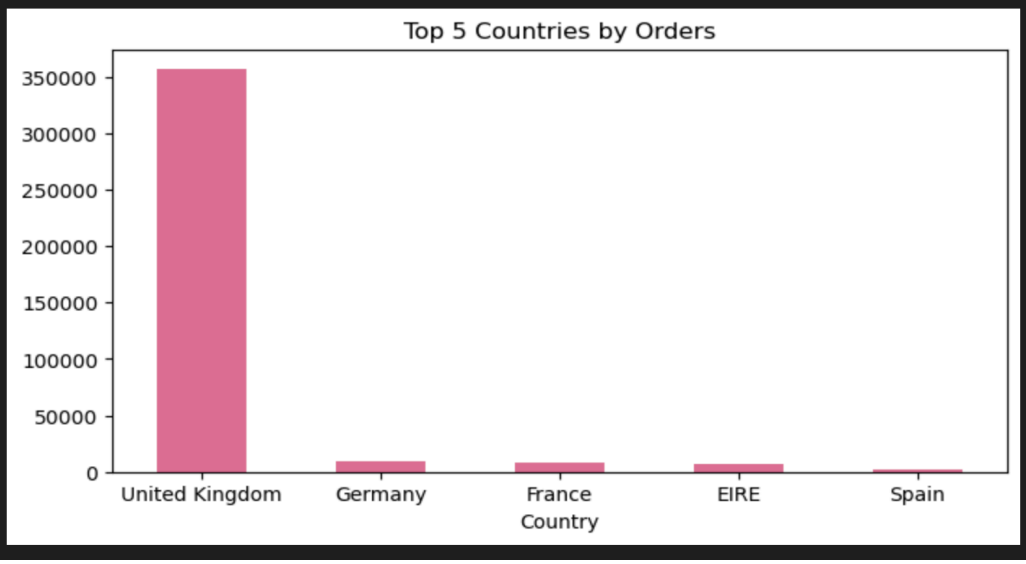
Explored ordering patterns by examining which days of the week or times of day saw the most orders. Also calculated average order processing times and identified any seasonal trends affecting sales.

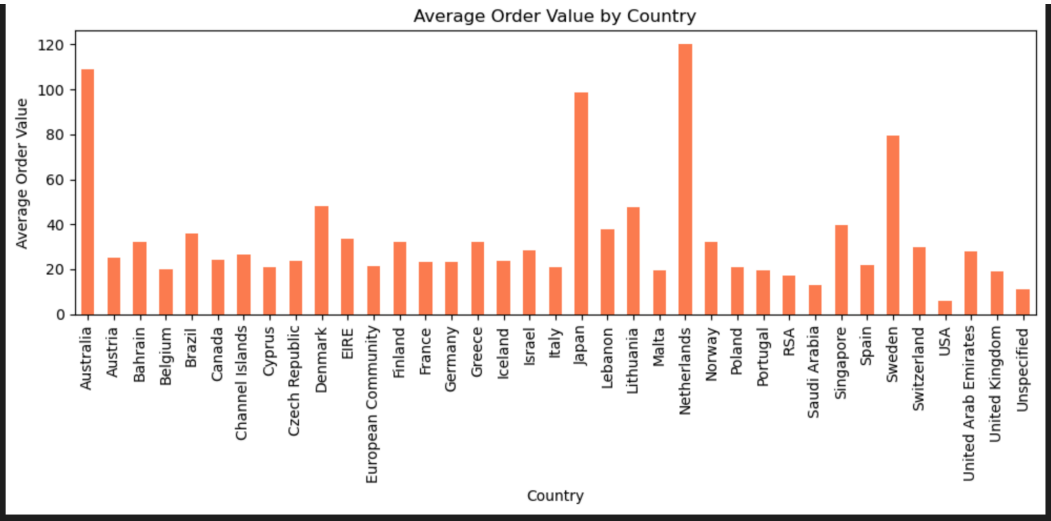


5. **Geographical Analysis**
   - Can you determine the top 5 countries with the highest number of orders?
   - Is there a correlation between the customer's country and the average order value?

Determined the top 5 countries with the highest order volumes and analyzed correlations between customer location and average order value, offering insights into geographic market dynamics.

```
Top 5 countries with highest number of orders
Country
United Kingdom    356728
Germany             9480
France              8475
EIRE                7475
Spain               2528
Name: count, dtype: int64

Text(0.5, 1.0, 'Top 5 Countries by Orders')
```

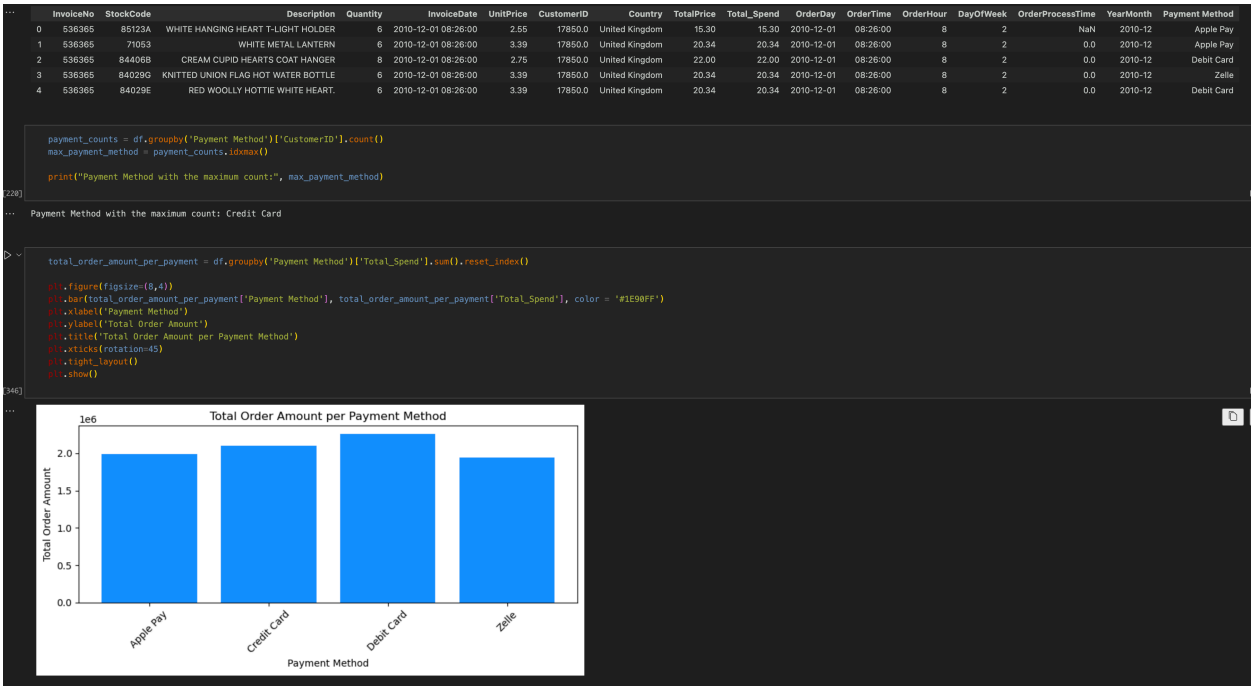





```
Correlation between Country and Average Order Value:
Country
Australia               108.910787
Austria                  25.322494
Bahrain                  32.258824
Belgium                  19.773301
Brazil                   35.737500
Canada                   24.280662
Channel Islands          26.520991
Cyprus                   21.045434
Czech Republic           23.590667
Denmark                  48.247147
EIRE                     33.445054
European Community       21.176230
Finland                  32.124806
France                   23.200714
Germany                  23.365978
Greece                   32.263836
Iceland                  23.681319
Israel                   28.293117
Italy                    21.034259
Japan                    98.716816
Lebanon                  37.641778
Lithuania                47.458857
Malta                    19.728110
...
United Arab Emirates     27.974706
United Kingdom           18.914008
Unspecified              11.040539
Name: Total_Spend, dtype: float64
```

6. **Payment Analysis**
   - What are the most common payment methods used by customers?
   - Is there a relationship between the payment method and the order amount?

Generated data for payment methods and investigated common payment methods used by customers and examined any relationships between payment methods and order amounts, providing insights into payment preferences.



	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	TotalPrice	Total_Spend	OrderDay	OrderTime	OrderHour	DayOfWeek	OrderProcessTime	YearMonth	Payment Method
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom	15.30	15.30	2010-12-01	08:26:00	8	2	NaN	2010-12	Apple Pay
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	20.34	20.34	2010-12-01	08:26:00	8	2	0.0	2010-12	Apple Pay
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom	22.00	22.00	2010-12-01	08:26:00	8	2	0.0	2010-12	Debit Card
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	20.34	20.34	2010-12-01	08:26:00	8	2	0.0	2010-12	Zelle
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	20.34	20.34	2010-12-01	08:26:00	8	2	0.0	2010-12	Debit Card

```
payment_counts = df.groupby('Payment Method')['CustomerID'].count()
max_payment_method = payment_counts.idxmax()

print("Payment Method with the maximum count:", max_payment_method)
```

```
Payment Method with the maximum count: Credit Card
```

```
total_order_amount_per_payment = df.groupby('Payment Method')['Total_Spend'].sum().reset_index()

plt.figure(figsize=(8,4))
plt.bar(total_order_amount_per_payment['Payment Method'], total_order_amount_per_payment['Total_Spend'], color = '#1E90FF')
plt.xlabel('Payment Method')
plt.ylabel('Total Order Amount')
plt.title('Total Order Amount per Payment Method')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```


7. **Customer Behavior**
   - How long, on average, do customers remain active (between their first and last purchase)?
   - Are there any customer segments based on their purchase behavior?

Measured how long customers remained active between their first and last purchase and identified segments based on purchasing behavior. This analysis helped in understanding customer lifecycle stages.

```
Average Duration of Customer Activity: 133.38586459286367 days
recency_threshold:  49.0
frequency_threshold:  41.0

Customer Segments:
            Recency  Frequency        Segment
CustomerID
12346.0         325          2   Low Activity
12347.0           1        182  High Activity
12348.0          74         31   Low Activity
12349.0          18         73  High Activity
12350.0         309         17   Low Activity
...             ...        ...            ...
18280.0         277         10   Low Activity
18281.0         180          7   Low Activity
18282.0           7         13   Low Activity
18283.0           3        721  High Activity
18287.0          42         70  High Activity

[4372 rows x 3 columns]
```

8. **Returns and Refunds**
   - What is the percentage of orders that have experienced returns or refunds?
   - Is there a correlation between the product category and the likelihood of returns?

Calculated the percentage of orders that experienced returns or refunds and explored correlations between product categories and return likelihood, aiming to improve product satisfaction.

```
Percentage of Orders with Returns or Refunds: 2.21%
Percentage of Returns by Product Category:

Description
 4 PURPLE FLOCK DINNER CANDLES           NaN
 50'S CHRISTMAS GIFT BAG LARGE      0.909091
 DOLLY GIRL BEAKER                  1.459854
 I LOVE LONDON MINI BACKPACK             NaN
 I LOVE LONDON MINI RUCKSACK             NaN
                                     ...
ZINC T-LIGHT HOLDER STARS SMALL     1.244813
ZINC TOP  2 DOOR WOODEN SHELF      18.181818
ZINC WILLIE WINKIE  CANDLE STICK    0.520833
ZINC WIRE KITCHEN ORGANISER             NaN
ZINC WIRE SWEETHEART LETTER TRAY        NaN
Name: InvoiceNo, Length: 3896, dtype: float64
```

```
Correlation between Product Category and Likelihood of Returns:
Quantity_ordered   -0.581692
Name: Quantity_returned, dtype: float64
```

9. Profitability Analysis

• Can you calculate the total profit generated by the company during the dataset's period?

• What are the top 5 products with the highest profit margins?

Assessed total profit generated during the dataset period and identified products with the highest profit margins. This analysis was crucial for understanding profitability drivers.

Total profit generated:
568857.0240000001
Top 5 products with higher profits:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	TotalPrice	Total_Spend	OrderDay	OrderTime	OrderHour	DayOfWeek	OrderProcessTime	YearMonth	Payment Method	CostUnitPrice	CostPrice	Profit
540421	581483	23843	PAPER CRAFT , LITTLE BIRDIE	80995	2011-12-09 09:15:00	2.08	16446.0	United Kingdom	168469.60	168469.60	2011-12-09	09:15:00	9	4	4919.383333	2011-12	Debit Card	1.95	157940.25	10529.35
61619	541431	23166	MEDIUM CERAMIC TOP STORAGE JAR	74215	2011-01-18 10:01:00	1.04	12346.0	United Kingdom	77183.60	77183.60	2011-01-18	10:01:00	10	1	NaN	2011-01	Zelle	0.97	71988.55	5195.05
222680	556444	22502	PICNIC BASKET WICKER 60 PIECES	60	2011-06-10 15:28:00	649.50	15098.0	United Kingdom	38970.00	38970.00	2011-06-10	15:28:00	15	4	0.100000	2011-06	Credit Card	607.97	36478.20	2491.80
173382	551697	POST	POSTAGE	1	2011-05-03 13:46:00	8142.75	16029.0	United Kingdom	8142.75	8142.75	2011-05-03	13:46:00	13	1	0.916667	2011-05	Debit Card	7622.08	7622.08	520.67
348325	567423	23243	SET OF TEA COFFEE SUGAR TINS PANTRY	1412	2011-09-20 11:05:00	5.06	17450.0	United Kingdom	7144.72	7144.72	2011-09-20	11:05:00	11	1	0.000000	2011-09	Zelle	4.74	6692.88	451.84

10. Customer Satisfaction

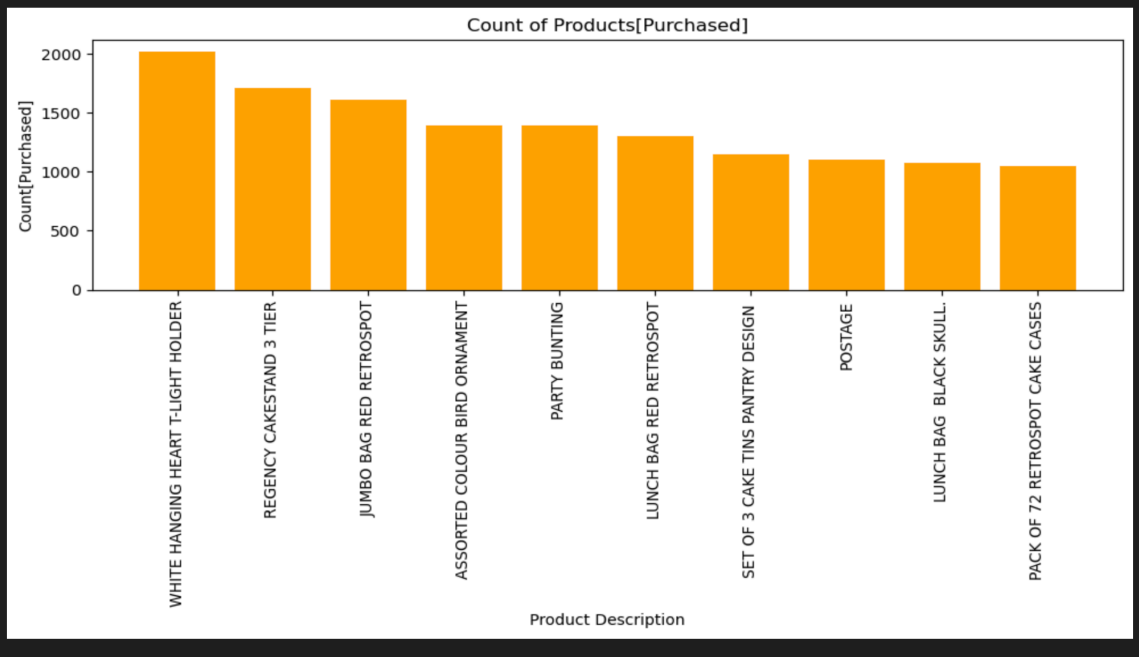
• Is there any data available on customer feedback or ratings for products or services?

• Can you analyze the sentiment or feedback trends, if available?

Analyzed available customer feedback or ratings to understand sentiment trends. This analysis provided valuable insights into customer satisfaction levels and potential areas for improvement.

Most frequently bought products:

	Description	count
0	WHITE HANGING HEART T-LIGHT HOLDER	2016
1	REGENCY CAKESTAND 3 TIER	1714
2	JUMBO BAG RED RETROSPOT	1615
3	ASSORTED COLOUR BIRD ORNAMENT	1395
4	PARTY BUNTING	1390
5	LUNCH BAG RED RETROSPOT	1303
6	SET OF 3 CAKE TINS PANTRY DESIGN	1152
7	POSTAGE	1099
8	LUNCH BAG BLACK SKULL.	1078
9	PACK OF 72 RETROSPOT CAKE CASES	1050



Most frequently retuned products:

	Description	count
0	REGENCY CAKESTAND 3 TIER	180
1	Manual	175
2	POSTAGE	97
3	JAM MAKING SET WITH JARS	86
4	Discount	77