
# Customer Segmentation using RFM Analysis

Group 35

**Project 2**
**Final Report**

**Course Code:** IE6400
**Course Name:** Foundation of Data Analytics
Fall 2023

**Team Members:**

Ginnegolla, Hema Venkata Sai Teja
Gopanwar, Anuj
Nalam, Lokhi
Rasineni, Neha
Sanapureddy, Sree Dharani Reddy

# IE 6400 Foundations Data Analytics Engineering

## Project 2 - Report

### Task 1: Data Preprocessing

1. We initiated data preprocessing by displaying data to check columns and data:

| | 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 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 541904 | 581587 | 22613 | PACK OF 20 SPACEBOY NAPKINS | 12 | 12/9/2011 12:50 | 0.85 | 12680.0 | France |
| 541905 | 581587 | 22899 | CHILDREN'S APRON DOLLY GIRL | 6 | 12/9/2011 12:50 | 2.10 | 12680.0 | France |
| 541906 | 581587 | 23254 | CHILDRENS CUTLERY DOLLY GIRL | 4 | 12/9/2011 12:50 | 4.15 | 12680.0 | France |
| 541907 | 581587 | 23255 | CHILDRENS CUTLERY CIRCUS PARADE | 4 | 12/9/2011 12:50 | 4.15 | 12680.0 | France |
| 541908 | 581587 | 22138 | BAKING SET 9 PIECE RETROSPOT | 3 | 12/9/2011 12:50 | 4.95 | 12680.0 | France |

The total number of entries in the dataset are 541909 records across 8 columns.

2. We then checked data.info() to inspect and identify the data type and count of each column.

```
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   InvoiceNo    541909 non-null  object
 1   StockCode    541909 non-null  object
 2   Description  540455 non-null  object
 3   Quantity     541909 non-null  int64
 4   InvoiceDate  541909 non-null  object
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  object
dtypes: float64(2), int64(1), object(5)
memory usage: 33.1+ MB
```

3. Handling missing values:

We then checked for missing values in the dataset: data.isnull().sum()

```
InvoiceNo           0
StockCode           0
Description      1454
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     135080
Country             0
dtype: int64
```

We identified missing values in the “Description” and “CustomerID” columns which needs to be handled.

4. Converting data types:

‘CustomerID’ has been converted from float to integer data type for consistency and ease of analysis.‘InvoiceDate’ has been converted to datetime format for more efficient handling.

```
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   InvoiceNo    406829 non-null  object
 1   StockCode    406829 non-null  object
 2   Description  406829 non-null  object
 3   Quantity     406829 non-null  int64
 4   InvoiceDate  406829 non-null  datetime64[ns]
 5   UnitPrice    406829 non-null  float64
 6   CustomerID   406829 non-null  int32
 7   Country      406829 non-null  object
```

5. Data Cleaning:

Rows with missing Description and CustomerID were removed, as this information is essential for item identification. Data has been cleaned and handled successfully:

```
InvoiceNo      0
StockCode      0
Description    0
Quantity       0
InvoiceDate    0
UnitPrice      0
CustomerID     0
Country        0
dtype: int64
```

## Task 2: RFM Calculation

1. Recency (days since last purchase), Frequency (total number of orders for each customer) and Monetary (sum of total price for each customer) values are calculated.

| CustomerID | Recency | Frequency | Monetary |
|---|---|---|---|
| **12346** | 326 | 2 | 0.00 |
| **12347** | 2 | 7 | 4310.00 |
| **12348** | 75 | 4 | 1797.24 |
| **12349** | 19 | 1 | 1757.55 |
| **12350** | 310 | 1 | 334.40 |

2. **Inferences from the Output:**

   Customer 12346: This customer has a high recency value (326 days), indicating they haven't made a purchase recently. Despite having 2 orders (frequency), their monetary value is 0, suggesting returned or canceled orders.

   Customer 12347: With a recency of 2 days and 7 orders, this customer is both recent and frequent, with a significant total spend, indicating a highly engaged and valuable customer.

   Customer 12348: They have a moderate recency and frequency but a relatively high monetary value, suggesting fewer but larger purchases.

   Customer 12349: This customer has made a recent purchase (19 days ago) but has only shopped once, spending a substantial amount in that single order.

   Customer 12350: With a high recency (310 days), this customer might be at risk of churn, having made only one purchase.

## Task 3: RFM Segmentation

1. We initiate RFM Segmentation based on rank-based quartiles:

| CustomerID | Recency | Frequency | Monetary | R_quartile | F_quartile | M_quartile |
|---|---|---|---|---|---|---|
| **12346** | 326 | 2 | 0.00 | 1 | 3 | 4 |
| **12347** | 2 | 7 | 4310.00 | 4 | 1 | 1 |
| **12348** | 75 | 4 | 1797.24 | 2 | 2 | 1 |
| **12349** | 19 | 1 | 1757.55 | 3 | 3 | 1 |
| **12350** | 310 | 1 | 334.40 | 1 | 3 | 3 |

**Explanation:**

1. Rank-Based Quartiles: This method first ranks customers based on their RFM scores and then divides these ranks into quartiles. This approach is particularly useful when there are many duplicate values in the dataset.

    a. Recency (R_quartile): Customers are ranked based on how recently they've made a purchase, with more recent purchasers receiving a lower rank. These ranks are then divided into quartiles.

    b. Frequency (F_quartile): Customers are ranked based on how often and how much they spend, respectively, with higher ranks indicating more frequency or spending. These ranks are then segmented into quartiles.

    c. Monetary (M_quartile): Customers are ranked based on how often and how much they spend, respectively, with higher ranks indicating more frequency or spending. These ranks are then segmented into quartiles.

2. For Recency, lower values indicate better outcomes, while for Frequency and Monetary values, higher values indicates better results.

3. We then combine RFM scores into a single RFM score for each customer:

| CustomerID | Recency | Frequency | Monetary | R_quartile | F_quartile | M_quartile | RFM_Score |
|---|---|---|---|---|---|---|---|
| **12346** | 326 | 2 | 0.00 | 1 | 3 | 4 | 134 |
| **12347** | 2 | 7 | 4310.00 | 4 | 1 | 1 | 411 |
| **12348** | 75 | 4 | 1797.24 | 2 | 2 | 1 | 221 |
| **12349** | 19 | 1 | 1757.55 | 3 | 3 | 1 | 331 |
| **12350** | 310 | 1 | 334.40 | 1 | 3 | 3 | 133 |

4. NOTE: Combining RFM Scores: The individual R, F, and M quartiles are concatenated to form a single RFM score for each customer. This score is a string with three characters, each representing the quartile ranking for Recency, Frequency, and Monetary, in that order.

    EXAMPLE :

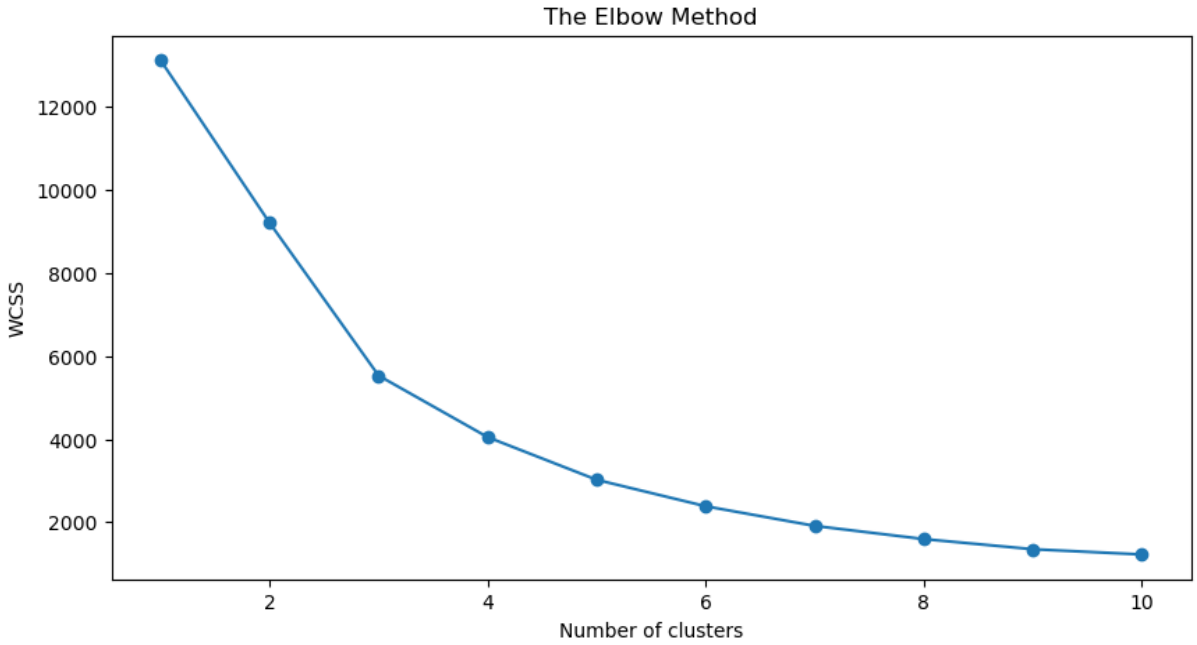
    In in the above table we can see the RFM_Score for the CustomerID 12346 is 134 i.e. ( R_quartile = 1 , F_quartile = 3 , M_quartile = 4 )

**Task 4: Customer Segmentation**

1. Initially we standardize the data. The StandardScaler from sklearn.preprocessing is used to standardize the RFM data. K-Means clustering is applied to the standardized RFM data for a range of cluster numbers from 1-10.

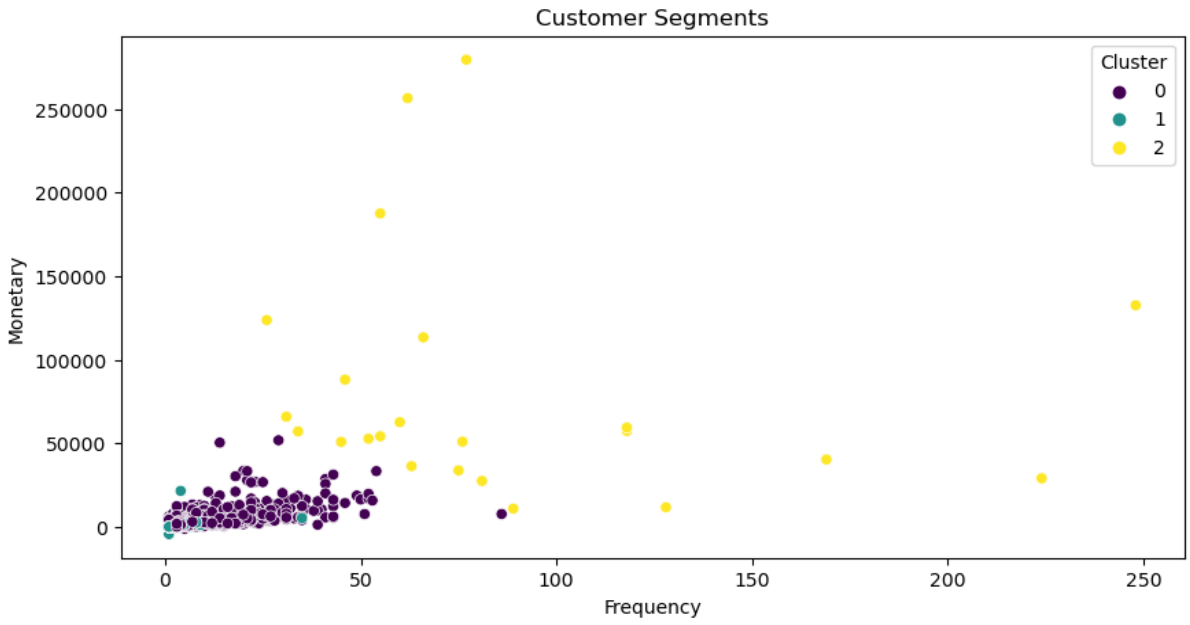
    For each value of k, the Within-Cluster Sum of Squares (WCSS) is calculated and stored in the list WCSS. It is a measure of variance within each cluster, and the goal of K-Means is to minimize this value.

2. We then utilize the Elbow method to find the optimal number of clusters and plot the results onto a line graph to observe ‘The Elbow’. The x-axis represents the number of clusters, while the y-axis represents the WCSS for each cluster number. The 'elbow point' on the graph is where the rate of decrease sharply changes, indicating a good balance between the number of clusters and the within-cluster variance.



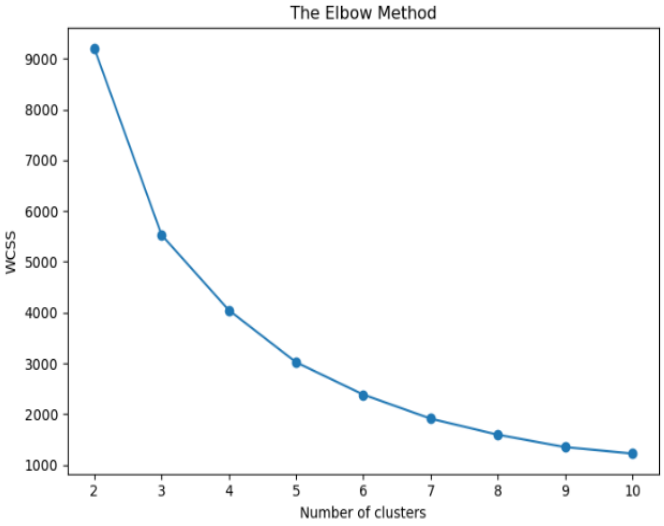

```
0    3241
1    1108
2      23
Name: Cluster, dtype: int64
```

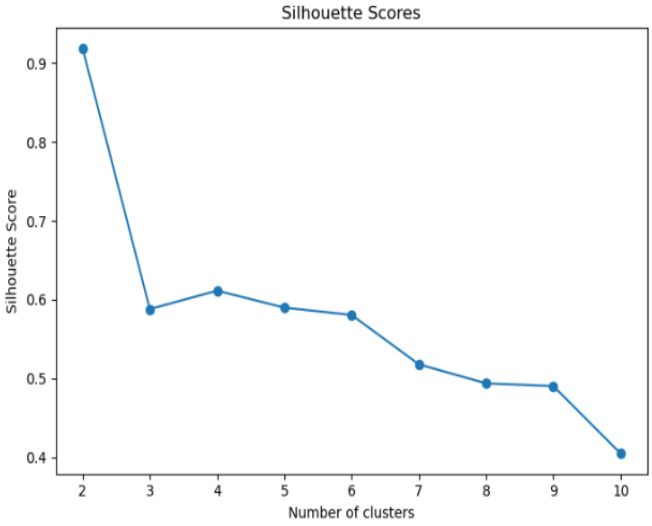
In the graph above we can see that as the number of clusters increases from 1 to 10, the WCSS decreases rapidly at first and then more slowly after around 3 clusters. The 'elbow' of the graph seems to be at the point where k=3, which suggests that choosing 3 clusters may be a good trade-off between having too many clusters and too much variance within the clusters.



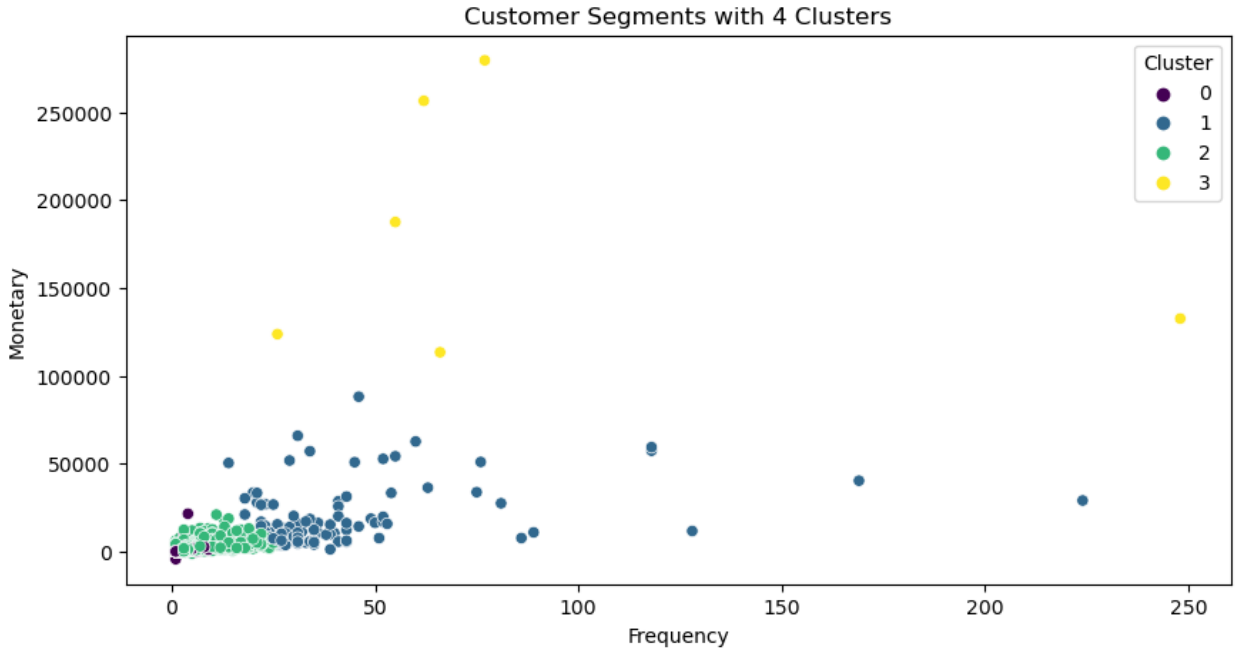
The scatter plot above visualizes customer segments created by K-Means clustering based on purchase Frequency and Monetary value. Three clusters are used to plot the scatter plot: Cluster 0 represents infrequent, low-spending customers; Cluster 1 includes moderately frequent and spending customers; and Cluster 2 consists of high-spending, frequent customers. From the above scatter plot we can conclude that most customers fall into the lower spending and less frequent purchase category, while a smaller segment represents high-value customers.

3. We then calculate the Silhouette scores to verify/check the accuracy of the Elbow method. Then we apply K-Means clustering with the chosen number of clusters and check the number of customers in each cluster. The values are then plotted.




```
2    3169
0    1087
1     110
3       6
Name: Cluster, dtype: int64
```



Cluster 0 (Purple): Customers in this cluster have low frequency and low to moderate monetary values. This group is likely made up of new or occasional customers.

Cluster 1 (Blue): This cluster includes customers with moderate frequency and a wide range of monetary values, including some of the highest values. They could be considered valuable customers due to their higher monetary contribution, even if they don't purchase as frequently.

Cluster 2 (Green): These customers have lower frequency and monetary values. They might be infrequent shoppers or those who make smaller purchases.

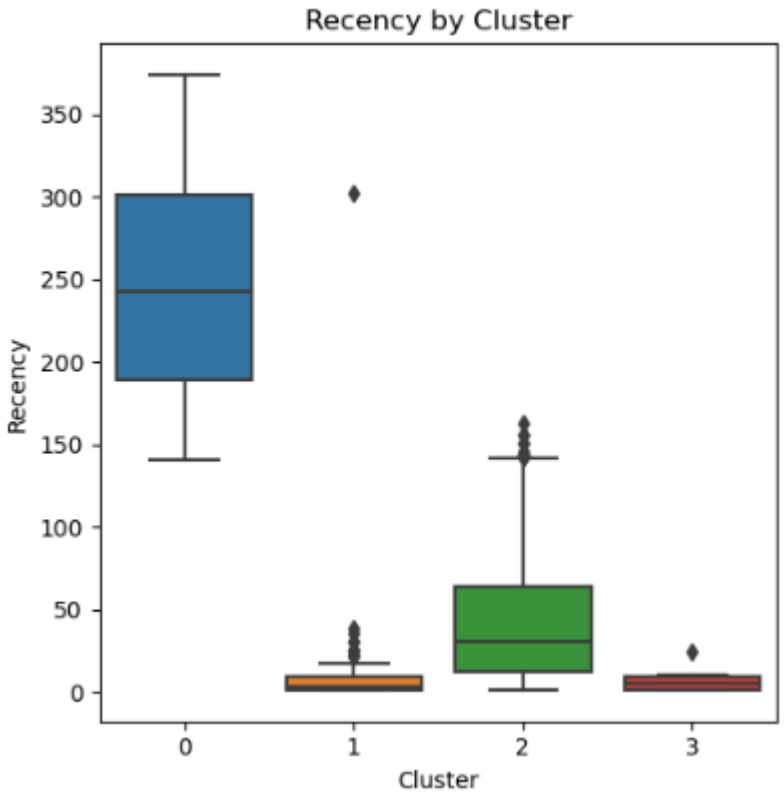
Cluster 3 (Yellow): Customers in this cluster have very high frequency and monetary values. This segment is likely the most valuable in terms of revenue, consisting of highly loyal and high-spending customers.

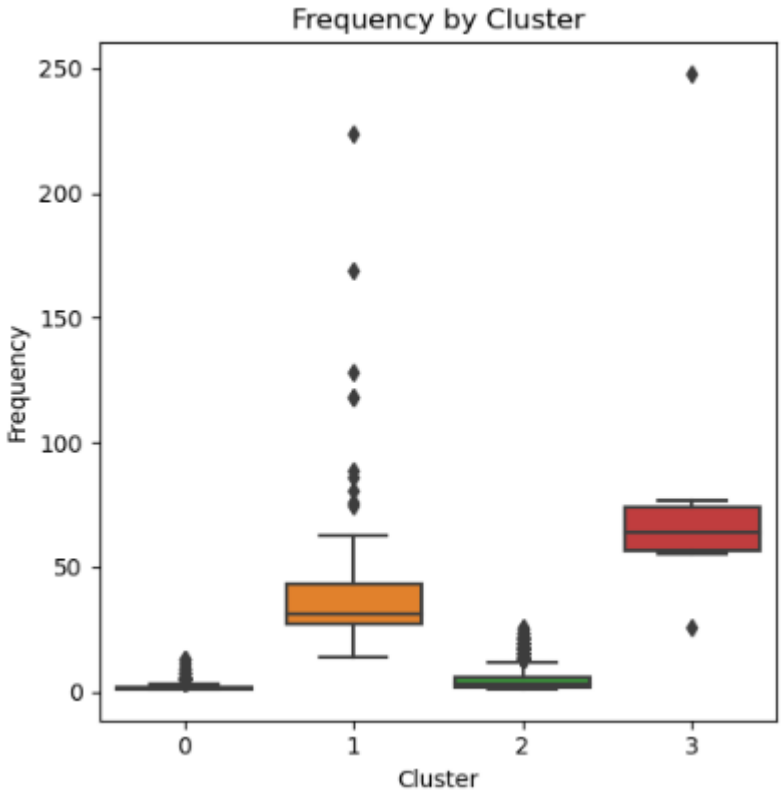
The businesses can use this information to tailor specific marketing strategies for each cluster, like focusing retention efforts on the high-value customers in Cluster 3 or trying to increase the purchase frequency of customers in Cluster 1.

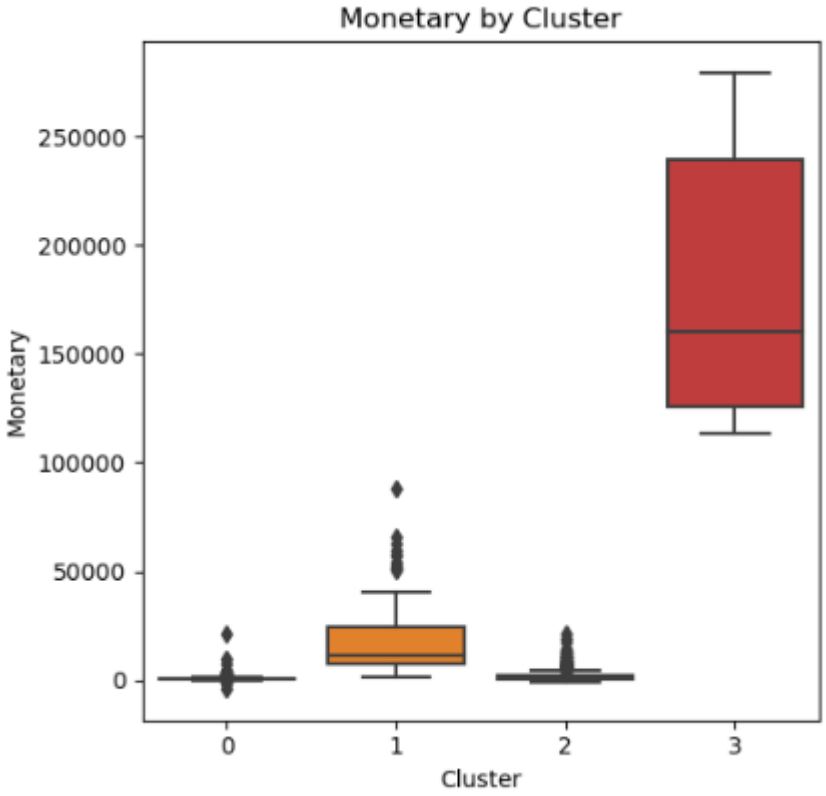
**Task 5: Segment Profiling**

1. Analyzing and profiling each customer segment: We display the cluster profiles and utilize box plots for deeper insights, each for Recency, Frequency and Monetary values.

| | Cluster | Average Recency | Average Frequency | Average Monetary | Count |
|---|---|---|---|---|---|
| 0 | 0 | 247.951242 | 1.805888 | 453.488888 | 1087 |
| 1 | 1 | 9.181818 | 40.672727 | 18441.961455 | 110 |
| 2 | 2 | 41.606500 | 4.802461 | 1478.515539 | 3169 |
| 3 | 3 | 7.666667 | 89.000000 | 182181.981667 | 6 |






2. **Interpretation of Segment Profiles:**

   Average Recency: This metric shows how recent the purchases are on average within each cluster. A lower average recency indicates a cluster with more recent activity.

Average Frequency: This represents the average number of purchases in each cluster. Higher frequency indicates a cluster with more frequent purchases.

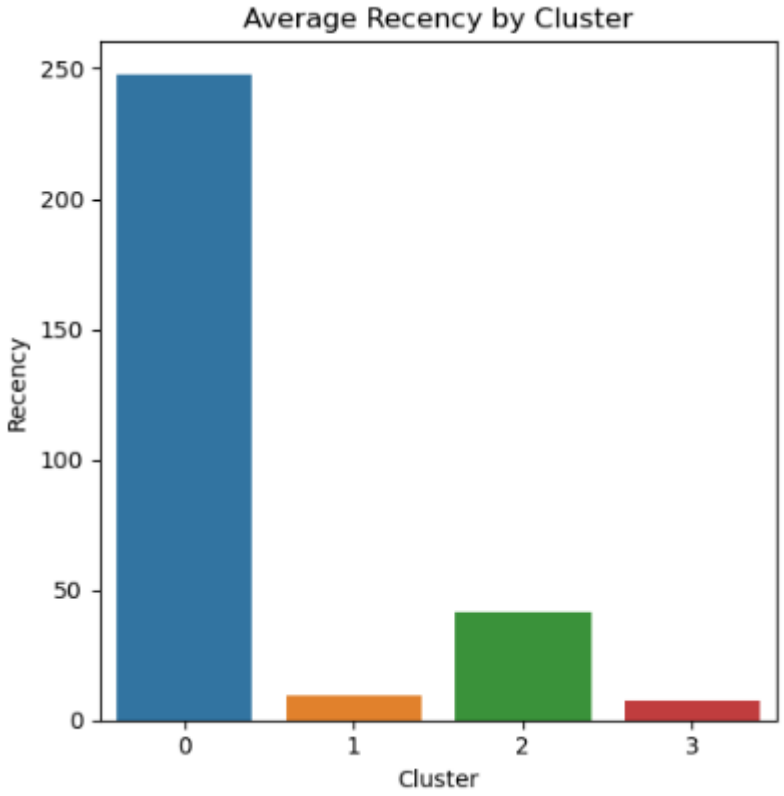
Average Monetary: This indicates the average spending of customers in each cluster. Higher values suggest clusters with higher spending customers.

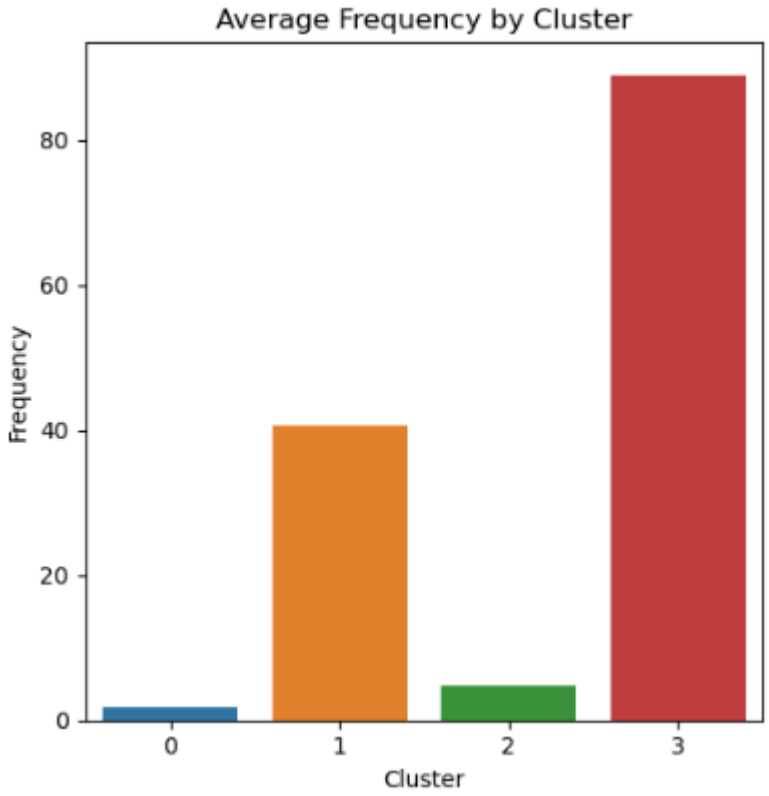
Count: The number of customers in each cluster, indicating the size of each segment.

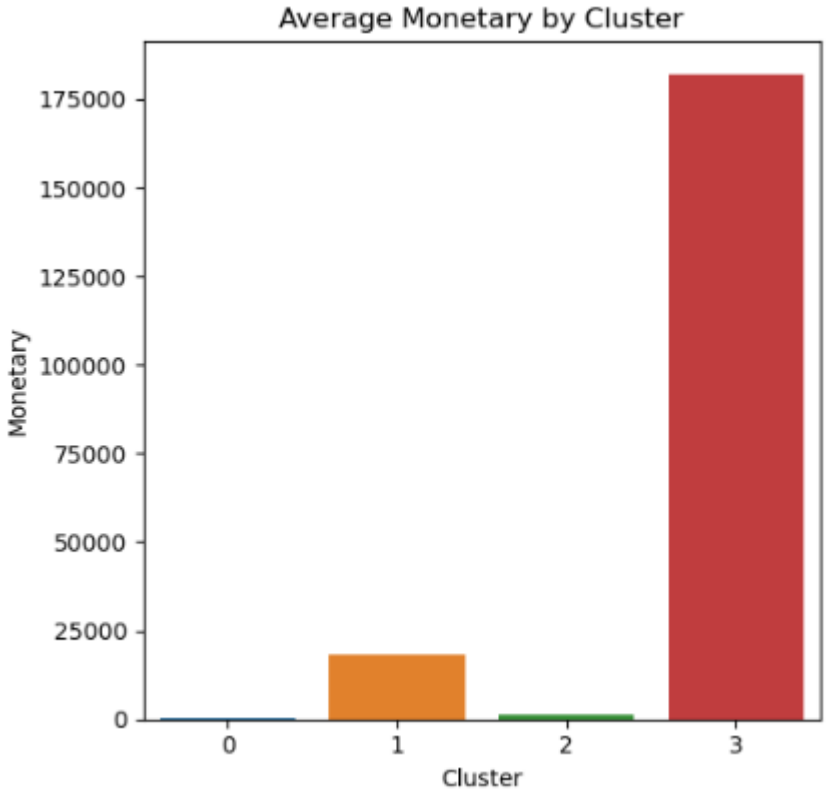
Box Plots: These provide visual insights into the distribution of RFM metrics within each cluster. They can show the range, median, and any potential outliers in each segment.

**Task 6: Marketing Recommendations**

1. We group data by cluster to find average RFM values and count of customers in each cluster and then plot the values individually for Recency, Frequency and Monetary.






| | Cluster | Recency | Frequency | Monetary | Count |
|---|---|---|---|---|---|
| 0 | 0 | 247.951242 | 1.805888 | 453.488888 | 1087 |
| 1 | 1 | 9.181818 | 40.672727 | 18441.961455 | 110 |
| 2 | 2 | 41.606500 | 4.802461 | 1478.515539 | 3169 |
| 3 | 3 | 7.666667 | 89.000000 | 182181.981667 | 6 |

2. We define thresholds for each segment and convert the RFM quartile columns to integers for comparison. We then categorize customers based on defined thresholds and display the segment distribution.

```
Potential Loyalists    1080
Loyal Customers        1046
At-Risk Customers       985
Churned Customers       790
Other                   371
New Customers           100
Name: Segment, dtype: int64
```

3. Based on the typical profiles, here are some **marketing strategies** for each segment:
    a. High-Value Customers (Low Recency, High Frequency, High Monetary)

        - Loyalty Programs: Offer exclusive rewards or loyalty programs to enhance their buying experience.
        - Upsell and Cross-sell: Recommend premium products or complementary items as they are more likely to make frequent and high-value purchases.
        - Personalized Communication: Engage with personalized emails or messages that reflect their interests and past purchasing behavior.
        - Exclusive Offers: Provide early access to new products or exclusive deals to make them feel valued.

    b. Loyal Customers (Moderate to Low Recency, High Frequency, Moderate Monetary)

        - Engagement Campaigns: Run regular engagement campaigns to maintain their interest and encourage continuous interaction with your brand.
        - Feedback and Reviews: Encourage them to provide feedback or reviews, enhancing their sense of belonging to the brand community.
        - Referral Incentives: Implement referral programs as loyal customers can be great brand ambassadors.

    c. Potential Loyalists (Low Recency, Moderate Frequency, Moderate Monetary)

        - Welcome Offers: Acknowledge their recent interaction with welcome-back offers or discounts on their next purchase.
        - Product Recommendations: Use their past purchase history to recommend products, increasing the likelihood of repeat purchases.
        - Membership Programs: Introduce them to membership programs that offer incremental benefits with more purchases.

    d. New Customers (Low Recency, Low Frequency, Low to Moderate Monetary)

        - First-Time Buyer Offers: Provide special offers or discounts on their next purchase to encourage them to come back.
        - Onboarding Series: Use an email onboarding series to educate them about different products and offers.
        - Social Proof: Share customer testimonials and reviews to build trust.

e. At-Risk Customers (High Recency, Moderate Frequency, Moderate Monetary)

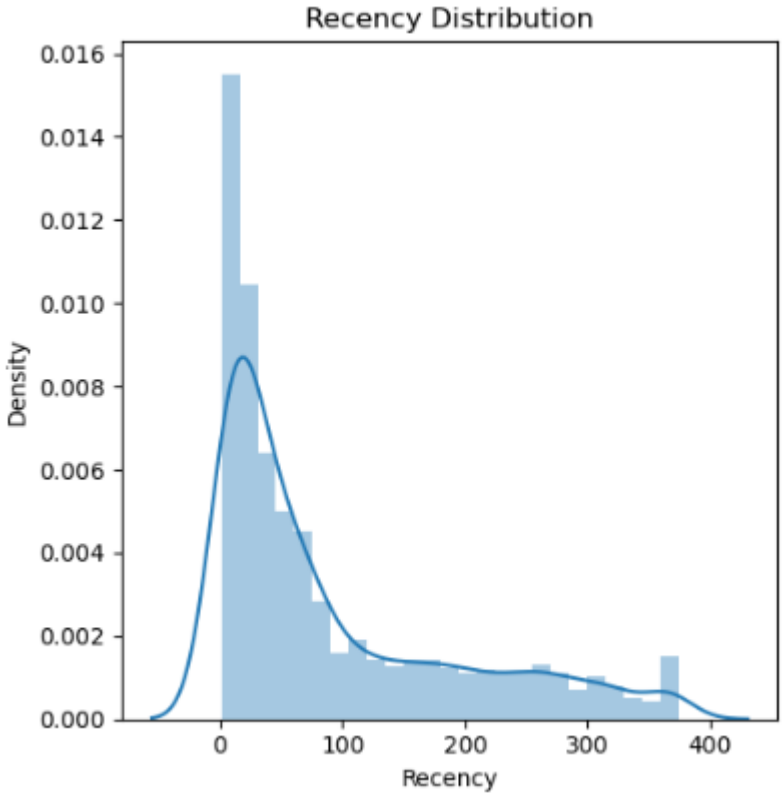
- Reactivation Campaigns: Send re-engagement emails or messages highlighting what they've missed.
- Survey for Feedback: Understand their inactivity reasons through feedback surveys and address their concerns.
- Win-Back Offers: Provide compelling offers or discounts to encourage them to make a purchase.

f. Churned Customers (High Recency, Low Frequency, Low Monetary)

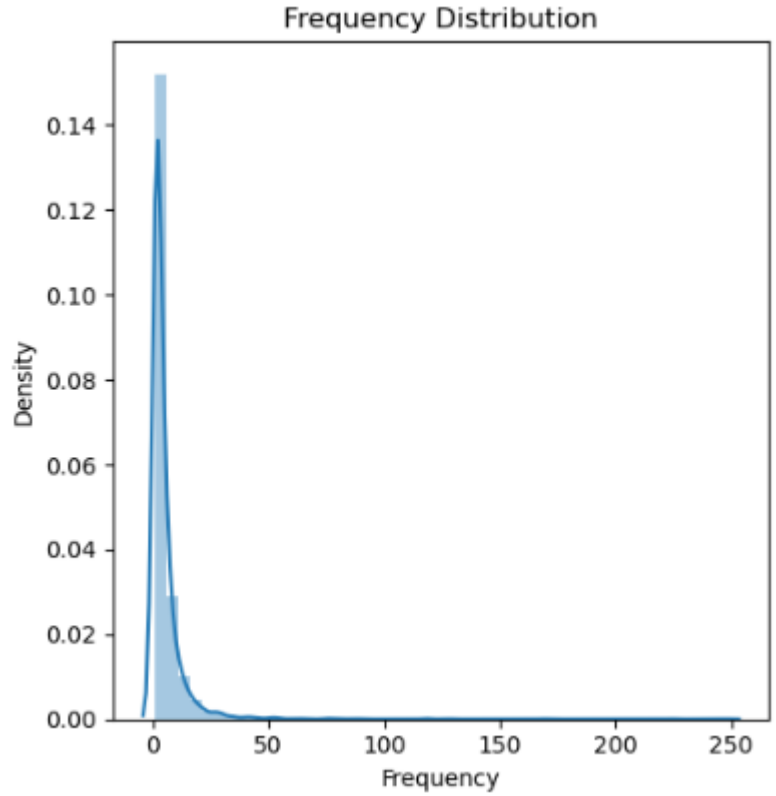
- Re-engagement Offers: Reach out with special offers to renew their interest.
- Market Research: Conduct research to understand their needs and preferences better.
- Improve Product/Service: Use insights from research to improve offerings and communicate these changes to them.

**Task 7: Visualization**

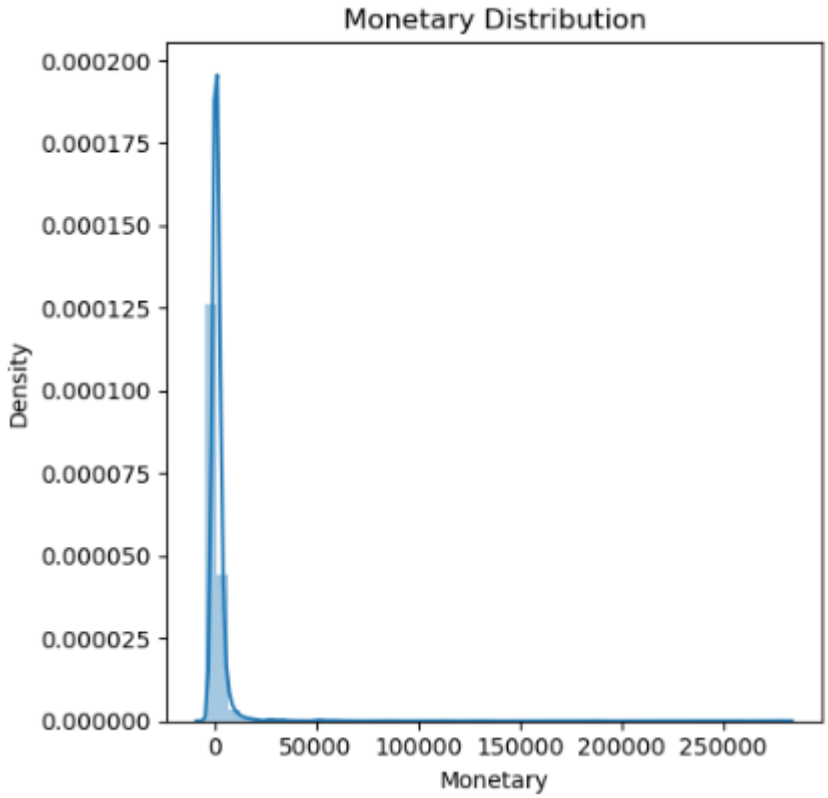
1. **RFM Distribution Plots**:
   a. Recency Distribution:



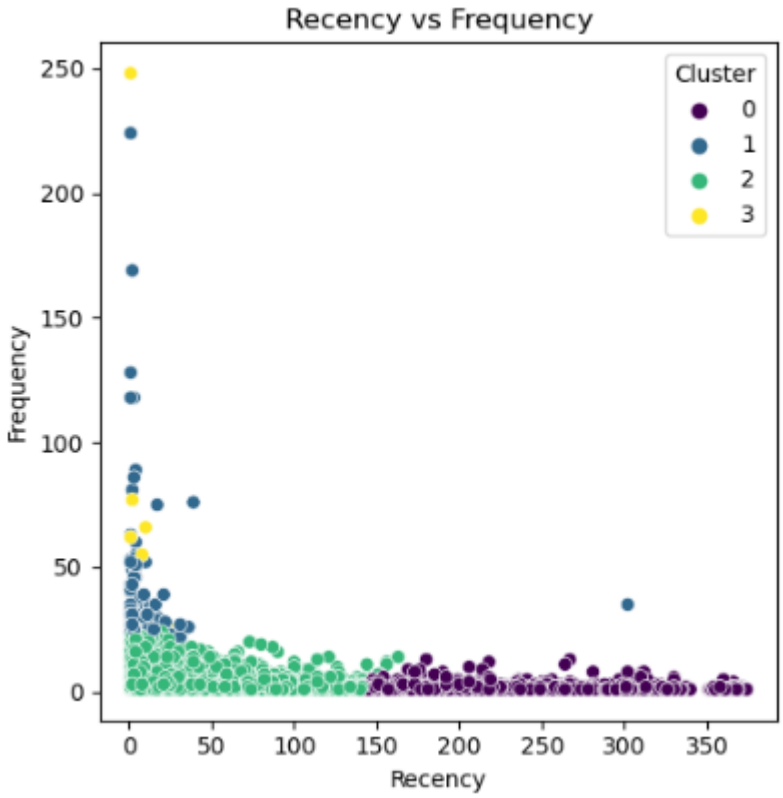
b. Frequency Distribution:



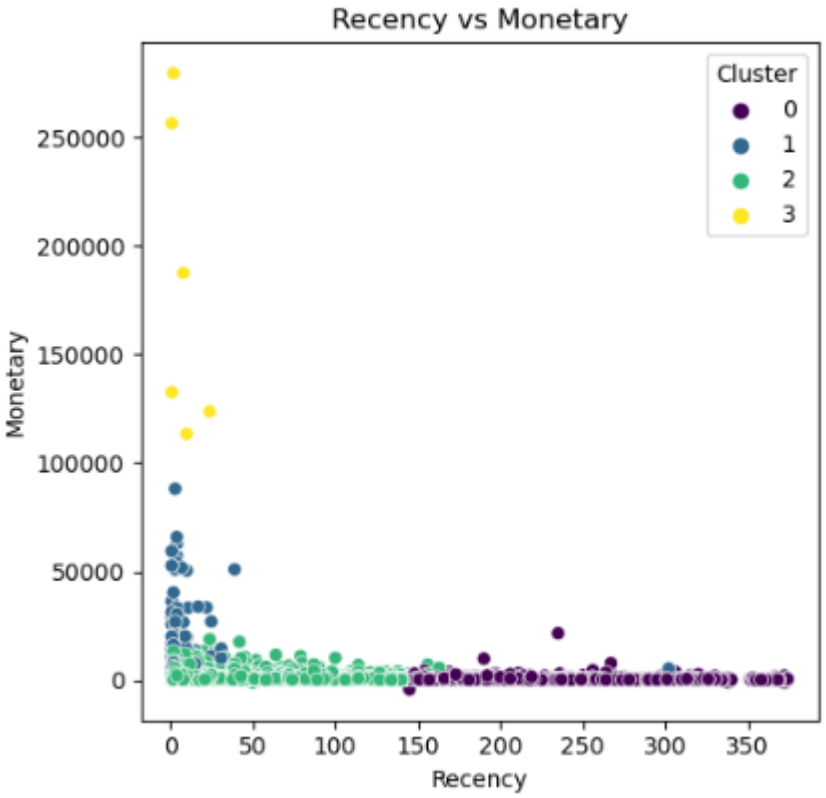

c. Monetary Distribution:



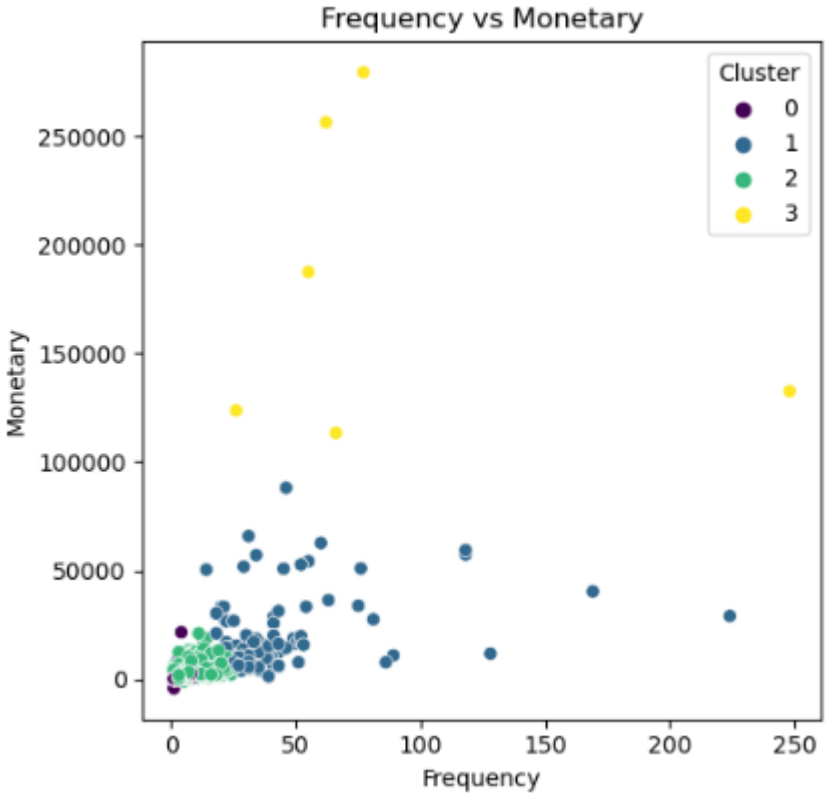

2. **RFM Cluster Scatterplots:**
   a. Scatter plot for Recency vs Frequency:




b. Scatter plot for Recency vs Monetary:



c. Scatter plot for Frequency vs Monetary:




## 1. Data Overview

```
Dataset Size:
Rows: 406829 Columns: 9

Column Descriptions:
InvoiceNo: object
StockCode: object
Description: object
Quantity: int64
InvoiceDate: datetime64[ns]
UnitPrice: float64
CustomerID: int32
Country: object
TotalPrice: float64

Date Ranges:
InvoiceDate: 2010-12-01 08:26:00 to 2011-12-09 12:50:00
```

i. What is the size of the dataset in terms of the number of rows and columns?

- The dataset contains a total of 406,829 rows and 9 columns, indicating a substantial amount of data that could be used for detailed analysis.

ii. Can you provide a brief description of each column in the dataset?

- The dataset details include the data type for each column, indicating how the information is stored and potentially how it can be manipulated for analysis.

iii. What is the period covered by this dataset?

- The data spans from December 1, 2010, at 08:26:00 to December 9, 2011, at 12:50:00, providing just over a year's worth of transaction data.

## 3. Customer Analysis

```
Number of Unique Customers: 4372

Distribution of Orders Per Customer:
 count    4372.000000
mean        5.075480
std         9.338754
min         1.000000
25%         1.000000
50%         3.000000
75%         5.000000
max       248.000000
Name: InvoiceNo, dtype: float64

Top 5 Customers by Order Count:
 CustomerID
14911    248
12748    224
17841    169
14606    128
13089    118
Name: InvoiceNo, dtype: int64
```

i. How many unique customers are there in the dataset?
   - There are 4,372 unique customers documented in the dataset.

ii. What is the distribution of the number of orders per customer?

   - On average, a customer has placed about 5.08 orders (mean). The standard deviation is approximately 9.34, indicating a wide variation in the number of orders per customer. The minimum number of orders a customer has placed is 1 (min). The 25th percentile is 1, meaning that 25% of the customers have placed only 1 order. The median number of orders per customer is 3 (50%), suggesting that half of the customers have placed 3 orders or fewer. The 75th percentile is 5, indicating that 75% of customers have placed 5 orders or fewer. The maximum number of orders placed by a single customer is 248 (max).

iii. Can you identify the top 5 customers who have made the most purchases by order count?

   - The dataset has the above customers as the most frequent purchasers, ranked by their total number of orders. These customers are likely to be highly valuable to the business given their frequent purchases and could be targeted for loyalty programs or special promotions to further enhance their engagement with the company.

## 4. Product Analysis

```
Top 10 Most Frequently Purchased Products:
 WHITE HANGING HEART T-LIGHT HOLDER    2369
REGENCY CAKESTAND 3 TIER              2200
JUMBO BAG RED RETROSPOT               2159
PARTY BUNTING                         1727
LUNCH BAG RED RETROSPOT               1638
ASSORTED COLOUR BIRD ORNAMENT         1501
SET OF 3 CAKE TINS PANTRY DESIGN      1473
PACK OF 72 RETROSPOT CAKE CASES       1385
LUNCH BAG  BLACK SKULL.               1350
NATURAL SLATE HEART CHALKBOARD        1280
Name: Description, dtype: int64

Average Price of Products: 4.61

Product Generating the Highest Revenue:
 Description
DOTCOM POSTAGE    206245.48
Name: Revenue, dtype: float64
```

i. What are the top 10 most frequently purchased products?

- The dataset reveals the top 10 products with the highest frequency of purchase.

ii. What is the average price of products in the dataset?

- The average price of the product is 4.61.

iii. Can you find out which product category generates the highest revenue?

- The category contributing the most to revenue is 'Dotcom Postage', which has produced a total revenue of $206,245.48.

## 5. Time Analysis

```
Orders by Day of Week:
 Thursday     103857
Tuesday      101808
Monday        95111
Wednesday     94565
Friday        82193
Sunday        64375
Name: DayOfWeek, dtype: int64

Orders by Hour of Day:
 12    78709
15     77519
13     72259
14     67471
11     57674
16     54516
10     49037
9      34332
17     28509
8       8909
18      7974
19      3705
20       871
7        383
6         41
Name: HourOfDay, dtype: int64
```

```
Average Order Processing Time: 34.86 hours

Orders by Month:
 Year  Month
2010  12       42481
2011  1        35147
      2        27707
      3        36748
      4        29916
      5        37030
      6        36874
      7        39518
      8        35284
      9        50226
      10       60742
      11       84711
      12       25525
dtype: int64
```

i. Is there a specific day of the week or time when most orders are placed?

- Orders by Day of the Week:

The data suggests that Thursday is the busiest day with the highest number of orders (103,857). The next busiest days are Tuesday and Monday with 101,808 and 95,111 orders,

respectively. Wednesday and Friday have a slightly lower number of orders, with Wednesday at 94,565 and Friday at 82,193. Sunday has the fewest orders with 64,375.

- Orders by Hour of Day:

The busiest hour of the day for placing orders is noon, with 78,709 orders. The hours from 3 PM to 11 AM are also busy, but there's a gradual decrease in the number of orders as the hours progress from afternoon to late morning. The hours with the least number of orders are early morning (6 AM and 7 AM) and late evening (8 PM to 9 PM), with the lowest being at 6 AM with only 41 orders.

ii. What is the average processing time?

- The average processing time is 34.86 hours.

iii. Are there any seasonal trends in the dataset?

- The data indicates that November 2011 had the highest number of orders in that year, while December 2011 experienced a sharp decrease. This pattern may suggest seasonal trends or could be influenced by specific events or promotions (like Black Friday or Cyber Monday sales). The significant drop in December could be due to several factors, such as the dataset not covering the whole month or a natural decline in orders after the November peak.

## 6. Geographical Analysis:

```
Top 5 Countries with the Highest Number of Orders:
 United Kingdom    361878
Germany             9495
France              8491
EIRE                7485
Spain               2533
Name: Country, dtype: int64
```

```
Average Order Value by Country:
 Country
Netherlands             2818.431089
Australia               1986.627101
Lebanon                 1693.880000
Japan                   1262.165000
Israel                  1165.708333
Brazil                  1143.600000
RSA                     1002.310000
Singapore                912.039000
Denmark                  893.720952
Norway                   879.086500
Sweden                   795.563261
Greece                   785.086667
Switzerland              785.061972
EIRE                     784.593166
Cyprus                   647.314500
United Arab Emirates     634.093333
Iceland                  615.714286
Canada                   611.063333
Channel Islands          608.675455
Austria                  534.437895
Spain                    521.662667
Finland                  465.140417
France                   429.504017
Lithuania                415.265000
Portugal                 415.140143
Germany                  367.658723
Belgium                  343.789580
United Kingdom           340.830609
Unspecified              333.383750
Italy                    307.100182
Poland                   300.547500
Bahrain                  274.200000
European Community       258.350000
Malta                    250.547000
USA                      247.274286
Czech Republic           141.544000
Saudi Arabia              65.585000
dtype: float64
```

i. Can you determine the top 5 countries with the highest number of orders?

- The top 5 countries with the highest number of orders are the United Kingdom, Germany, France, Eire, and Spain.

ii. Is there a correlation between the country of the customer and the average order value?

- It appears that there is no direct correlation between the number of orders and the average order value. The country with the highest number of orders, the United Kingdom, does not have the highest average order value. It has one of the lower average order values compared to other countries.

- Additionally, countries like the Netherlands have a high average order value but do not rank among the top in order count, indicating that while they may place fewer orders, the orders they do place are of higher value on average.

## 7. Payment Analysis

```
# For most common payment methods
payment_method_counts = data['PaymentMethod'].value_counts()

# To assess the relationship between payment method and order amount
average_order_amount_by_payment = data.groupby('PaymentMethod')['TotalOrderValue'].mean()
```

- There is no dataset for payment method, this would be the necessary steps if the payment method was available in the dataset. However, if we had access to such a dataset, the steps for conducting payment analysis would typically follow a process similar to what's depicted in the image above.

## 8. Customer Behavior

```
Average Customer Activity Duration: 133 days 17:25:29.204025618
```

| CustomerID | Recency | Frequency | Monetary |
|---|---|---|---|
| **12346.0** | 326 | 2 | 0.00 |
| **12347.0** | 2 | 7 | 4310.00 |
| **12348.0** | 75 | 4 | 1797.24 |
| **12349.0** | 19 | 1 | 1757.55 |
| **12350.0** | 310 | 1 | 334.40 |

```
Potential Loyalists    1080
Loyal Customers        1046
At-Risk Customers       985
Churned Customers       790
Other                   371
New Customers           100
Name: Segment, dtype: int64
```

| CustomerID | Recency | Frequency | Monetary | R_quartile | F_quartile | M_quartile | RFM_Score | Cluster | Segment |
|---|---|---|---|---|---|---|---|---|---|
| **12346** | 326 | 2 | 0.00 | 1 | 3 | 4 | 134 | 0 | Potential Loyalists |
| **12347** | 2 | 7 | 4310.00 | 4 | 1 | 1 | 411 | 2 | Churned Customers |
| **12348** | 75 | 4 | 1797.24 | 2 | 2 | 1 | 221 | 2 | Other |
| **12349** | 19 | 1 | 1757.55 | 3 | 3 | 1 | 331 | 2 | At-Risk Customers |
| **12350** | 310 | 1 | 334.40 | 1 | 3 | 3 | 133 | 0 | Potential Loyalists |

i. How long on average, do customers remain active (between the first and the last purchase)?

- On an average customers stay active for 133 days 17 hours and 25 minutes between the first and the last purchase.

ii. Are there any customer segments based on their purchase behavior

- Customer segments have been identified based on purchasing behavior, and categorized using RFM (Recency, Frequency, Monetary value) metrics. These segments are:

- Potential Loyalists: Customers with recent and frequent transactions, indicating a trend towards loyalty.
- Loyal Customers: Those who consistently show high values across all RFM metrics, demonstrating both frequent patronage and significant spending.
- Churned Customers: Customers with low recent engagement, suggesting they are no longer active.
- At-Risk Customers: Individuals who have historically been good customers but have shown a decrease in recent activity, indicating a risk of churn.
- New Customers: Recently acquired customers with high spending but low frequency, indicating they are new to the business.

  This segmentation helps in understanding customer habits and tailoring business strategies to each distinct group.

## 9. Returns & Refunds

```
# If there was a 'Quantity' column then negative values would indicate returns
# Calculating the total number of orders and the number of returned orders
total_orders = data['InvoiceNo'].nunique()
returned_orders = data[data['Quantity'] < 0]['InvoiceNo'].nunique()
percentage_returns = (returned_orders / total_orders) * 100

# If there was a 'ProductCategory' column then
# Calculating the percentage of returns by product category
returns_by_category = data[data['Quantity'] < 0].groupby('ProductCategory')['InvoiceNo'].nunique()
orders_by_category = data.groupby('ProductCategory')['InvoiceNo'].nunique()
percentage_returns_by_category = (returns_by_category / orders_by_category) * 100
```

- The Return and Refund can't be found as the dataset is not available but if there was a ProductCategory column then we would have considered the negative values in the Quantity column as the return and the steps for conducting Returns and Refund would typically follow a process similar to what's depicted in the image above.

## 10. Profitability Analysis

```
# TOTAL PROFIT CALCULATION
# If the dataset had 'ProfitMargin' columns
# Total Sales Revenue = UnitPrice * Quantity
data['SalesRevenue'] = data['UnitPrice'] * data['Quantity']

# If COGS data is not available but Profit Margin percentage is available
# COGS = SalesRevenue - (SalesRevenue * ProfitMargin)
data['COGS'] = data['SalesRevenue'] - (data['SalesRevenue'] * data['ProfitMargin'])

# Total Profit = Total Sales Revenue - Total COGS
total_profit = data['SalesRevenue'].sum() - data['COGS'].sum()
```

```
# TOP 5 PRODUCTS WITH HIGHEST PROFIT MARGIN
# Calculating Profit Margin for each product
# Profit Margin = (SalesRevenue - COGS) / SalesRevenue
data['ProductProfitMargin'] = (data['SalesRevenue'] - data['COGS']) / data['SalesRevenue']

# Group by Product and calculate average profit margin
average_profit_margin_by_product = data.groupby('Product')['ProductProfitMargin'].mean()

# Finding top 5 products with the highest profit margins
top_5_products_profit_margin = average_profit_margin_by_product.sort_values(ascending=False).head(5)
```

- There is no revenue and cost associated column in the dataset to perform profitability analysis, as we need both revenue and the cost associated with each sale to calculate profit. However, if we had access to such a dataset, the steps for conducting profitability analysis would typically follow a process similar to what's depicted in the image above.

## 11. Customer Satisfaction

```
from textblob import TextBlob

# Assuming 'data' has a 'CustomerFeedback' column
# Apply TextBlob to each feedback to get sentiment polarity
data['SentimentPolarity'] = data['CustomerFeedback'].apply(lambda x: TextBlob(x).sentiment.polarity)

# Categorize sentiments as Positive, Neutral, or Negative
data['Sentiment'] = pd.cut(data['SentimentPolarity'], bins=3, labels=["Negative", "Neutral", "Positive"])

# Analyze sentiment distribution
sentiment_distribution = data['Sentiment'].value_counts()

sentiment_distribution
```

```
# Assuming 'data' has a 'Rating' column
# Calculate the average rating for each product
average_rating_by_product = data.groupby('Product')['Rating'].mean()

# Analyzing rating distribution
rating_distribution = data['Rating'].value_counts()

average_rating_by_product, rating_distribution
```

- The dataset does not have a customer feedback and rating column to perform the task. However, if we had access to such a dataset with the required columns, the steps for conducting customer satisfaction analysis would typically follow a process similar to what's depicted in the image above.