Customer Segmentation

Team Members:

- **Krishna Koushik Parimi**
- **Rupa Dinesh Lnu**
- **Ravi Tarun Prasad Nimmalapudi**

Objective:

Customer segmentation is the process of dividing customers into groups based on shared characteristics. The objective of this customer segmentation project is to use machine learning techniques to categorize online retail customers into distinct and meaningful groups based on their purchasing behavior and characteristics. The goal is to enhance the understanding of the customer base, identify trends and behaviors, and develop personalized marketing efforts for each segment. This segmentation will, in turn, lead to improved customer satisfaction and increased profitability for the online retail business.

Problem Statement:

The challenge of this project is to effectively analyze the transactional trends of different products in a particular time frame that are bought by people residing in different locations. So, coming to **why segment customers?**

From a business perspective, mainly to get a **competitive market edge** and moreover putting people into categories is only one aspect of consumer segmentation. Customers are better understood when they are segmented, and you can use this knowledge to develop content that addresses the demands and difficulties of each section.

Data Sources:

The dataset, titled "Online Retail," is sourced from the UCI Machine Learning Repository, providing a valuable collection of transnational data capturing transactions from December 1, 2010, to September 9, 2011, for a UK-based non-store online retail business specializing in unique all-occasion gifts. The data offers a glimpse into the actual operations and customer interactions of this retail company, shedding light on purchase behaviors, transaction details, and customer demographics.

Dataset/Features Description:

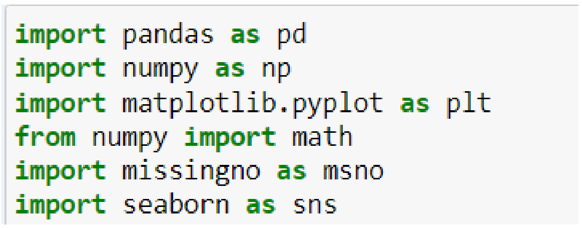
The dataset includes 8 features like InvoiceNo, StokeCode, Description, Quantity, InvoiceData, UnitPrice, CustomerID and the country. It is important to understand what each feature is:

1. **InvoiceNo:**
   - Unique identifier for each transaction or invoice.
   - Helps in the monitoring and evaluation of specific transactions. Transactional patterns can reveal information about seasonality, promotional efficacy, and client loyalty.
2. **StockCode:**
   - Code representing the specific product in the transaction.
   - It helps to find product preferences and associations among various client segments by enabling the examination of which goods are usually purchased together.
3. **Description:**
   - Brief description of the product involved in the transaction.
   - It gives the StockCode more context, which makes it easier to comprehend the kinds of goods that buyers are purchasing. This can be quite important for figuring out trends and preferences.
4. **Quantity:**
   - The quantity of the product purchased in the transaction.
   - It helps in comprehending how clients behave when making purchases. Low amounts might represent lone clients, whereas high quantities might imply wholesalers or bulk buyers.
5. **InvoiceDate:**
   - Date and time when the transaction was recorded.
   - It makes possible to analyze temporal patterns, including seasonality, peak purchasing periods, and long-term trends. In order to customize marketing and promotional methods, this is essential.
6. **UnitPrice:**
   - Price per unit of the product.
   - It helps to analyze how different client categories spend their money. Lower unit pricing may be linked to customers on a tight budget, whereas higher unit prices may imply premium customers.

7. **CustomerID:**
   - Unique identifier for each customer.
   - It is necessary for analysis that is customer-centric. aids in detecting repeat consumers, comprehending client loyalty, and classifying transactions per customer.
8. **Country:**
   - The country where the customer is located.
   - It allows for segmentation by location. It is possible for different nations to display unique buying habits, cultural influences, and reactions to marketing tactics.

Businesses can identify client categories based on variables like purchasing frequency, product preferences, spending patterns, temporal activity, and geographic location by studying these elements combined. Improved consumer happiness and business performance are the results of this segmentation, which in turn makes focused and customized marketing strategies possible.

Workflow Brief Walkthrough:



```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from numpy import math
import missingno as msno
import seaborn as sns
```

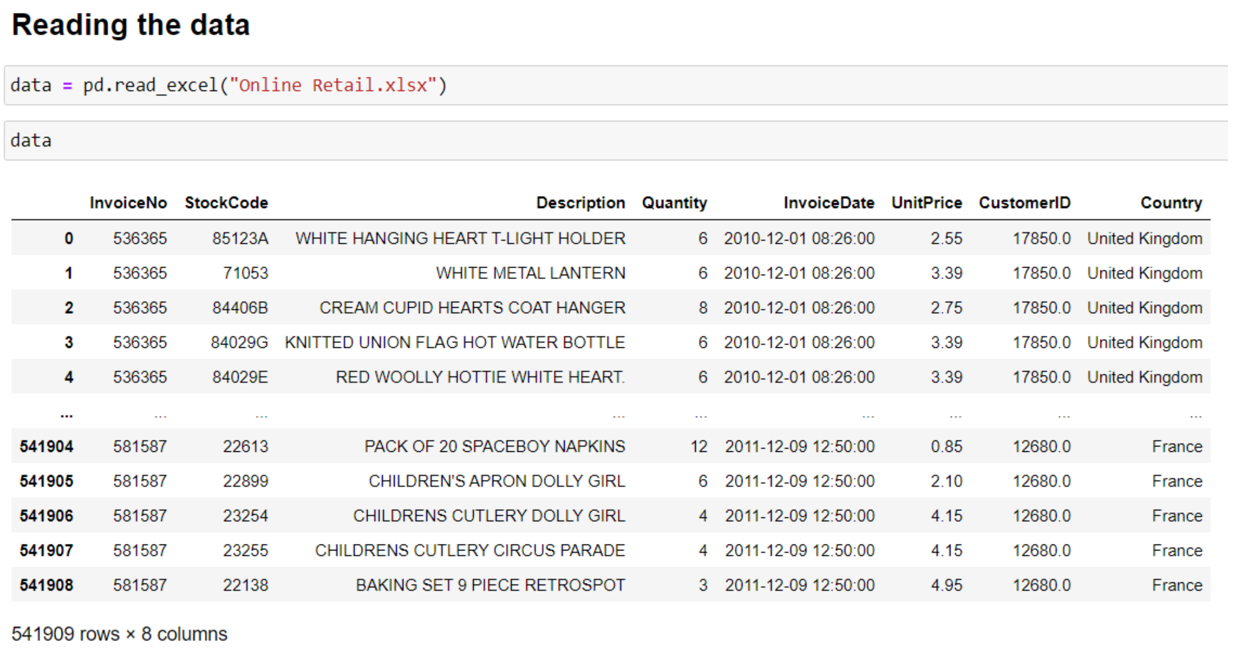
Fig 1

Fig 1: Importing the required libraries. As of now we have used six libraries like pandas, NumPy, matplotlib, missingno and seaborn. Here is a description of each library:

- **pandas:** Data manipulation and analysis, providing powerful structures for efficient cleaning, exploration, and transformation of datasets.
- **numpy:** Support for large, multi-dimensional arrays and mathematical functions, essential for numerical operations in scientific computing and data analysis.
- **matplotlib.pyplot:** Plotting library for creating visualizations, crucial for conveying insights and patterns in the data.
- **math (from numpy):** Additional mathematical functions for advanced operations beyond core Python math module.
- **missingno:** Data visualization library for a quick summary of missing values,

aiding in decision-making during data preprocessing.

- **seaborn:** Statistical data visualization library based on matplotlib, simplifying the creation of informative and attractive graphics during exploratory data analysis.
- **scikit-learn:** It provides simple and efficient tools for data analysis and modeling, including various machine learning algorithms, preprocessing techniques, and model evaluation methods.



Reading the data

```
data = pd.read_excel("Online Retail.xlsx")
```

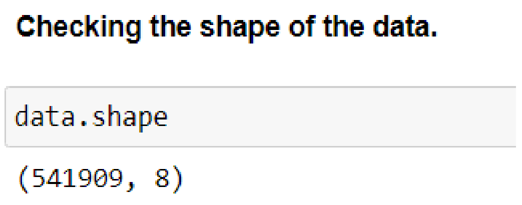
```
data
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
...	...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	2011-12-09 12:50:00	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	2011-12-09 12:50:00	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	2011-12-09 12:50:00	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	2011-12-09 12:50:00	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	2011-12-09 12:50:00	4.95	12680.0	France

541909 rows × 8 columns

Fig 2

Fig 2: Pandas efficiently read an Excel file into a panda Data Frame, facilitating easy data exploration and analysis in Python.



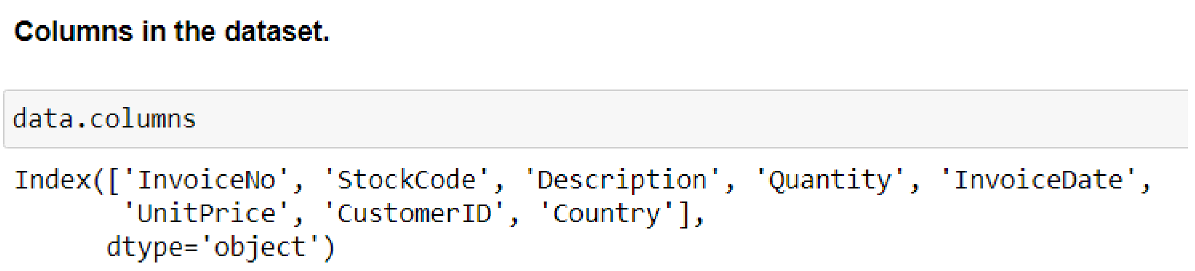
Checking the shape of the data.

```
data.shape
```

```
(541909, 8)
```

Fig 3

Fig 3: Shape of the data frame helps us know the number of features(columns) and the total number of data entries.



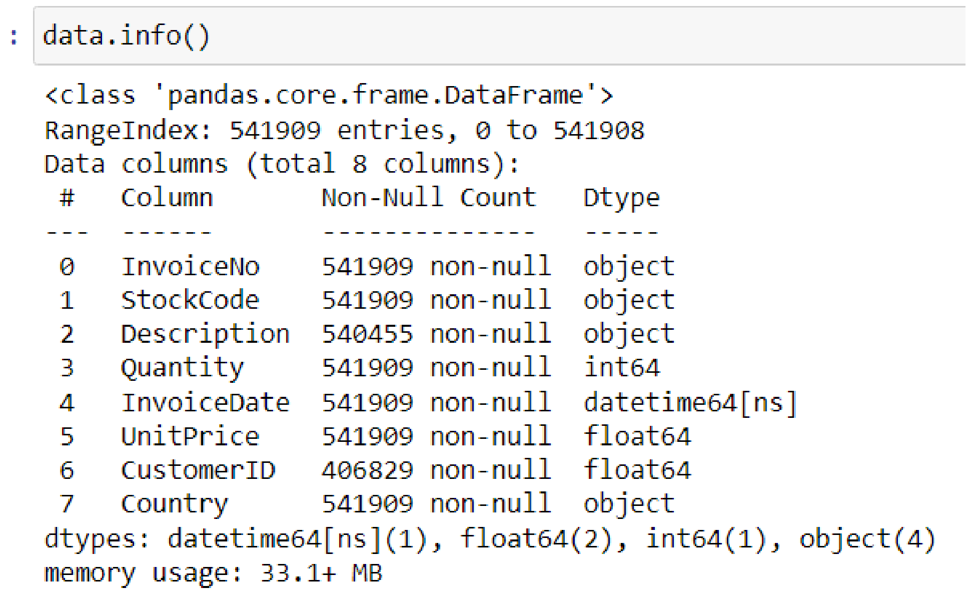
Columns in the dataset.

```
data.columns
```

```
Index(['InvoiceNo', 'StockCode', 'Description', 'Quantity', 'InvoiceDate',
       'UnitPrice', 'CustomerID', 'Country'],
      dtype='object')
```

Fig 4

Fig 4: The column labels of a DataFrame, providing a list of column names for quick reference and manipulation in data analysis.

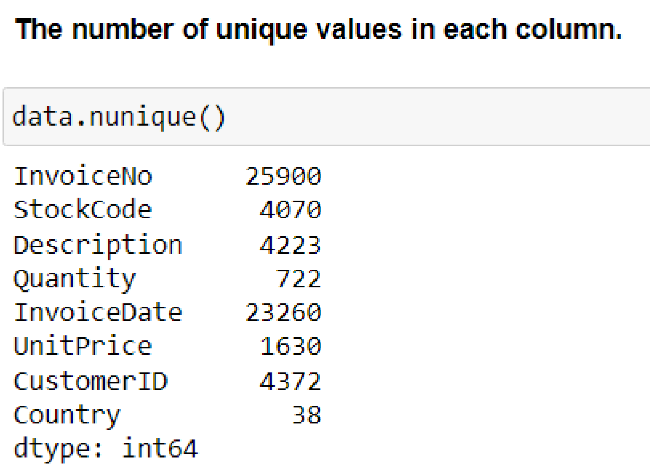


```
: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
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  datetime64[ns]
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  object
dtypes: datetime64[ns](1), float64(2), int64(1), object(4)
memory usage: 33.1+ MB
```

Fig 5

Fig 5: It Provides a concise summary of a pandas DataFrame, displaying information on data types, non-null values, and memory usage, offering a quick overview of the dataset's structure.



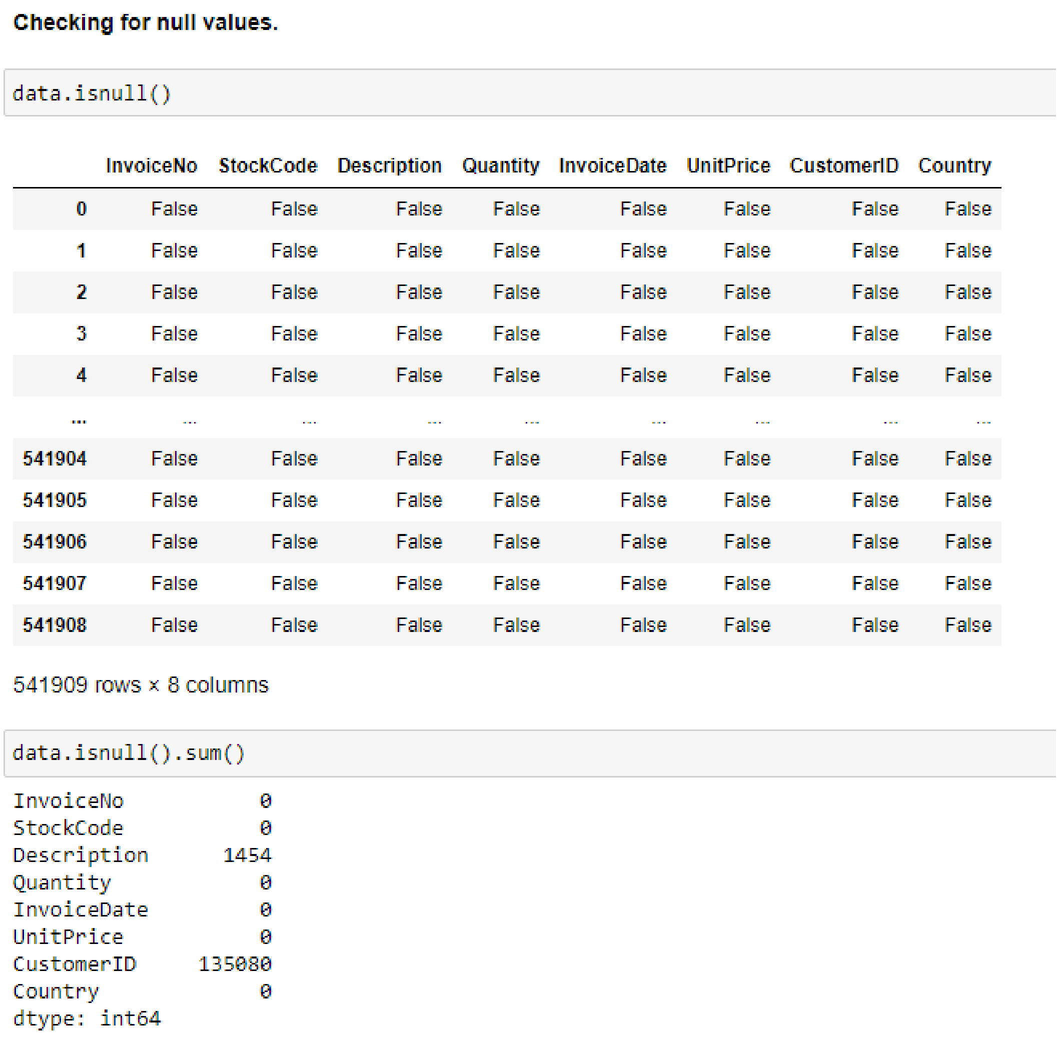
The number of unique values in each column.

```
data.nunique()
```

```
InvoiceNo      25900
StockCode       4070
Description     4223
Quantity         722
InvoiceDate    23260
UnitPrice       1630
CustomerID      4372
Country           38
dtype: int64
```

Fig 6

Fig 6: It returns the number of unique values in each column of a DataFrame, providing insights into the diversity of data within each feature.



Checking for null values.

```
data.isnull()
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...
541904	False	False	False	False	False	False	False	False
541905	False	False	False	False	False	False	False	False
541906	False	False	False	False	False	False	False	False
541907	False	False	False	False	False	False	False	False
541908	False	False	False	False	False	False	False	False

541909 rows × 8 columns

```
data.isnull().sum()
```

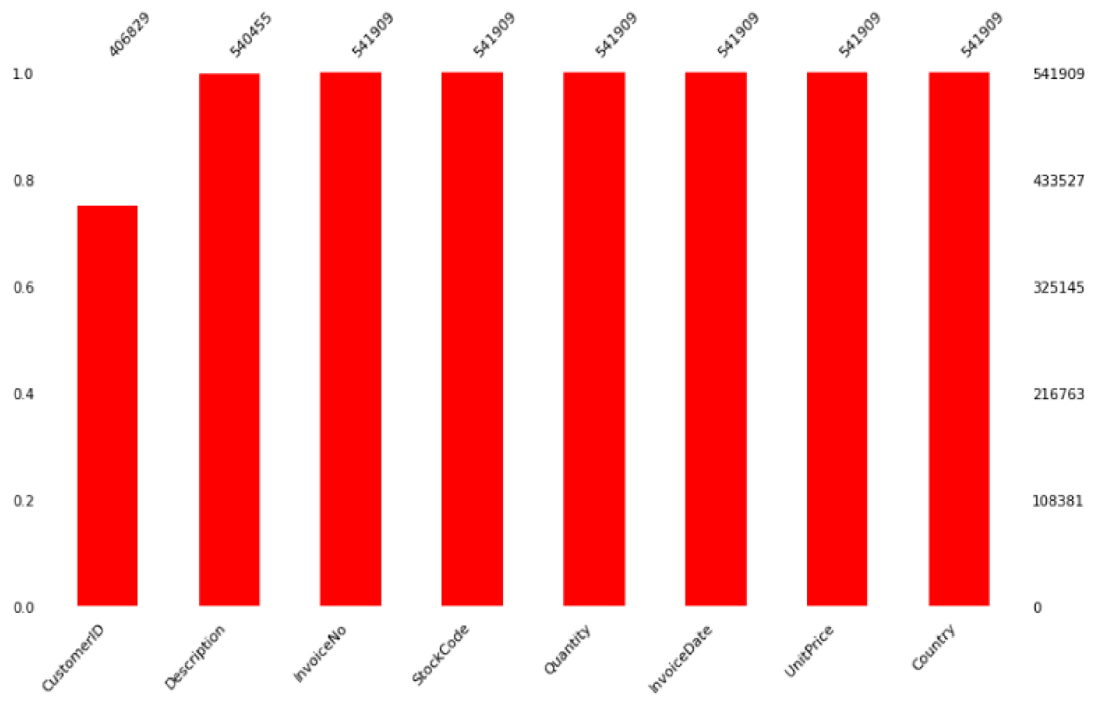
```
InvoiceNo           0
StockCode           0
Description      1454
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     135080
Country             0
dtype: int64
```

Fig 7

Fig 7: It returns a DataFrame of the same shape as data with Boolean values indicating the presence of missing values (True for missing, False for not missing) and .sum() gives the sum of missing values for each column in the dataframe which provides a quick idea of missing data across different features.

```
msno.bar(data, figsize=(10,5), sort="ascending", color='r', fontsize=8)
```

```
<Axes: >
```



As we can see the column containing the most number of null values is ['CustomerID'] and the next is ['Description'] column which differ with the total shape value that is 541909.

The presence of missing data and incorrect values within a dataset is common as the real world contains noisy data. These anomalies can significantly impact the accuracy and reliability of any subsequent analysis or modeling effort.

Fig 8

Dealing with the missing values.

- The column ['CustomerID'] has the most number of null values, we need to think about either dropping the whole column or the rows instead.
- We think that the column ['CustomerID'] is a crucial column and would be a key column that would help identify a customer therefore we go with dropping the null values instead. (Doesn't even constitute 5% of whole data).
- Coming to the ['Description'] column we have comparitively very less values so we proceed with dropping them as well.

```
data.dropna(inplace=True)
```

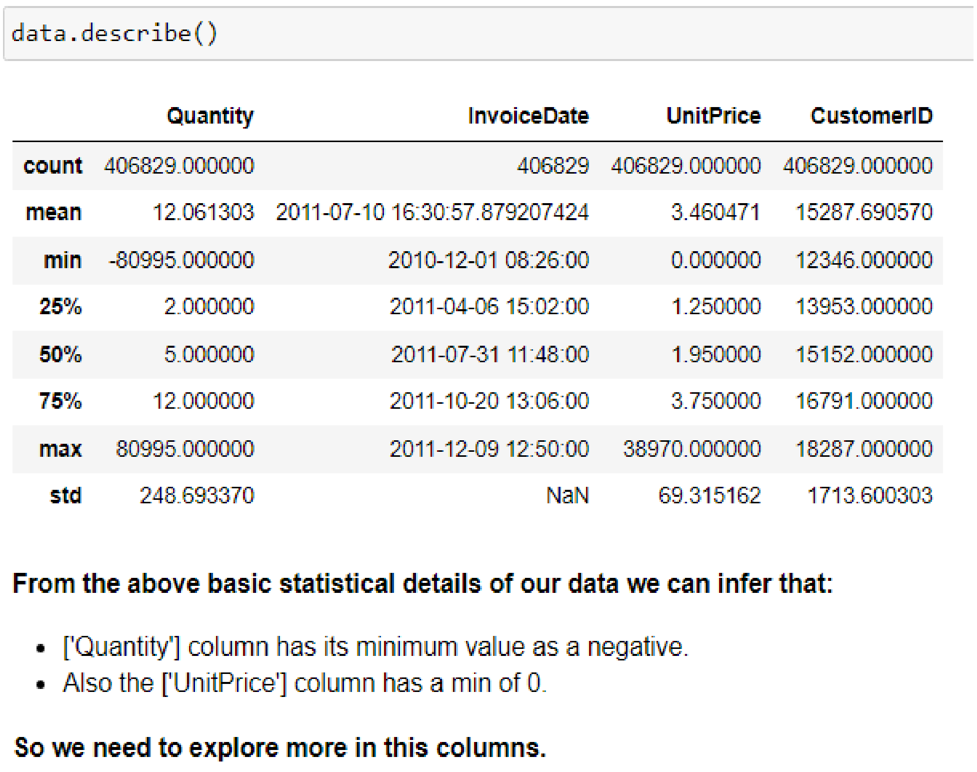
Fig 9

```
data.shape
```

```
(406829, 8)
```

As we can see we have dropped the rows having null values so we get the updated shape of our dataframe as (406829, 8).

Fig 10



```
data.describe()
```

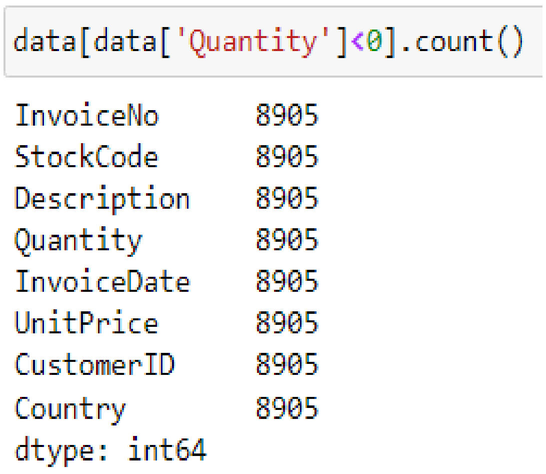
	Quantity	InvoiceDate	UnitPrice	CustomerID
count	406829.000000	406829	406829.000000	406829.000000
mean	12.061303	2011-07-10 16:30:57.879207424	3.460471	15287.690570
min	-80995.000000	2010-12-01 08:26:00	0.000000	12346.000000
25%	2.000000	2011-04-06 15:02:00	1.250000	13953.000000
50%	5.000000	2011-07-31 11:48:00	1.950000	15152.000000
75%	12.000000	2011-10-20 13:06:00	3.750000	16791.000000
max	80995.000000	2011-12-09 12:50:00	38970.000000	18287.000000
std	248.693370	NaN	69.315162	1713.600303

From the above basic statistical details of our data we can infer that:

- ['Quantity'] column has its minimum value as a negative.
- Also the ['UnitPrice'] column has a min of 0.

So we need to explore more in this columns.

Fig 11

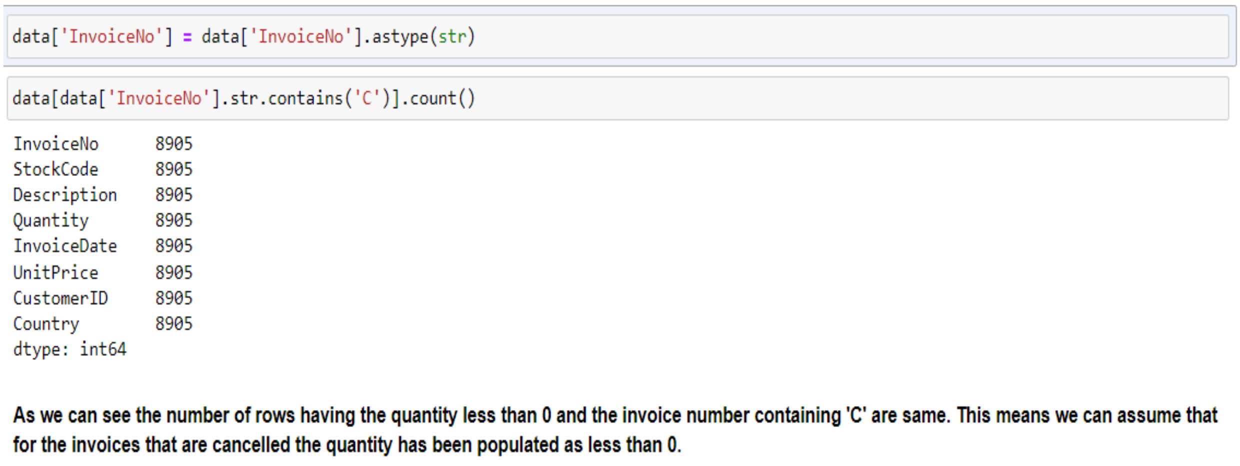


```
data[data['Quantity']<0].count()
```

```
InvoiceNo      8905
StockCode      8905
Description    8905
Quantity       8905
InvoiceDate    8905
UnitPrice      8905
CustomerID     8905
Country        8905
dtype: int64
```

Fig 12

Fig 12: Counts the number of rows in the DataFrame where the 'Quantity' column has values less than zero, indicating the occurrences of negative quantities in the dataset.

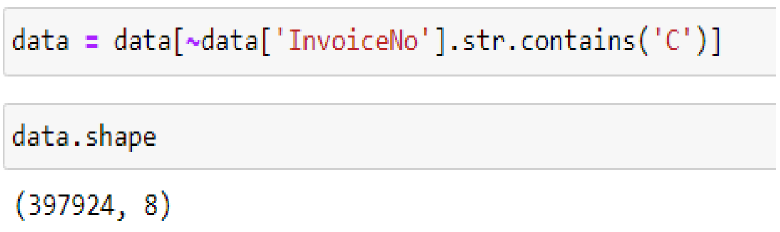

```
data['InvoiceNo'] = data['InvoiceNo'].astype(str)
```

```
data[data['InvoiceNo'].str.contains('C')].count()
```

```
InvoiceNo      8905
StockCode      8905
Description    8905
Quantity       8905
InvoiceDate    8905
UnitPrice      8905
CustomerID     8905
Country        8905
dtype: int64
```

As we can see the number of rows having the quantity less than 0 and the invoice number containing 'C' are same. This means we can assume that for the invoices that are cancelled the quantity has been populated as less than 0.

Fig 13

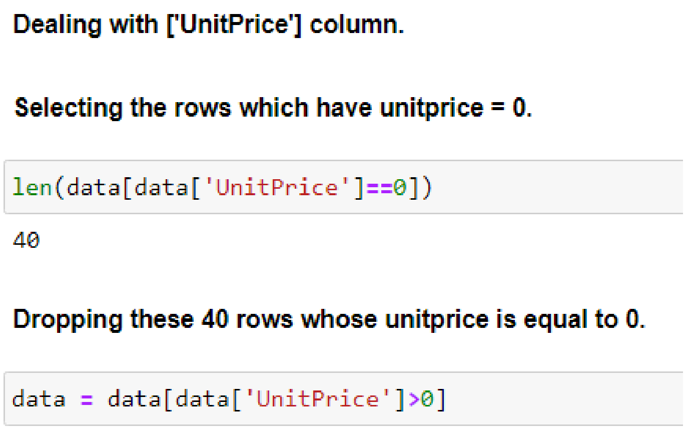

```
data = data[~data['InvoiceNo'].str.contains('C')]
```

```
data.shape
```

```
(397924, 8)
```

Fig 14

Fig 14: Previously, we saw that the number of rows that have negative quantity and the number of rows that contains "C" in the invoice number are the same. So, we drop the rows that contains the C in the invoice number. We only consider the rows which doesn't contain the letter C in the invoice number and drop the ones that contains C in the invoice number. The shape of the data frame lets us know the current number of rows and columns in the data frame.


Dealing with ['UnitPrice'] column.

Selecting the rows which have unitprice = 0.

```
len(data[data['UnitPrice']==0])
```

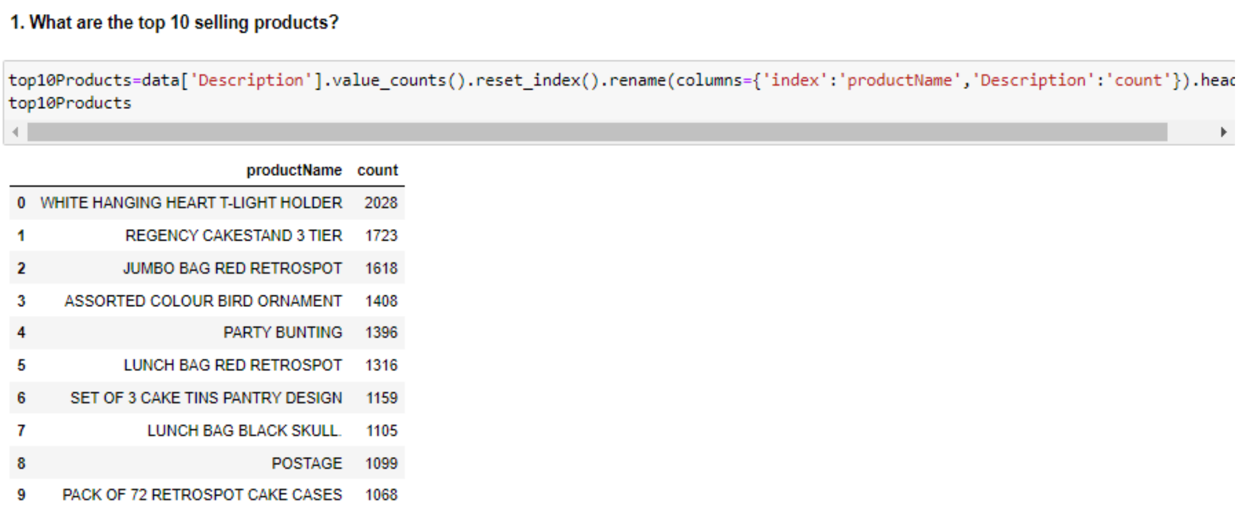
```
40
```

Dropping these 40 rows whose unitprice is equal to 0.

```
data = data[data['UnitPrice']>0]
```

Fig 15

Fig 15: There are few rows which have the unit price to be 0. The cost of any product cannot be zero. So, this data need to be eliminated. So, we just consider the rows with the unit price to be greater than 0.



1. What are the top 10 selling products?

```
top10Products=data['Description'].value_counts().reset_index().rename(columns={'index':'productName','Description':'count'}).heac
top10Products
```

	productName	count
0	WHITE HANGING HEART T-LIGHT HOLDER	2028
1	REGENCY CAKESTAND 3 TIER	1723
2	JUMBO BAG RED RETROSPOT	1618
3	ASSORTED COLOUR BIRD ORNAMENT	1408
4	PARTY BUNTING	1396
5	LUNCH BAG RED RETROSPOT	1316
6	SET OF 3 CAKE TINS PANTRY DESIGN	1159
7	LUNCH BAG BLACK SKULL.	1105
8	POSTAGE	1099
9	PACK OF 72 RETROSPOT CAKE CASES	1068

Fig 16

Fig 16: It is important that we know the top selling products to know the sales trend.

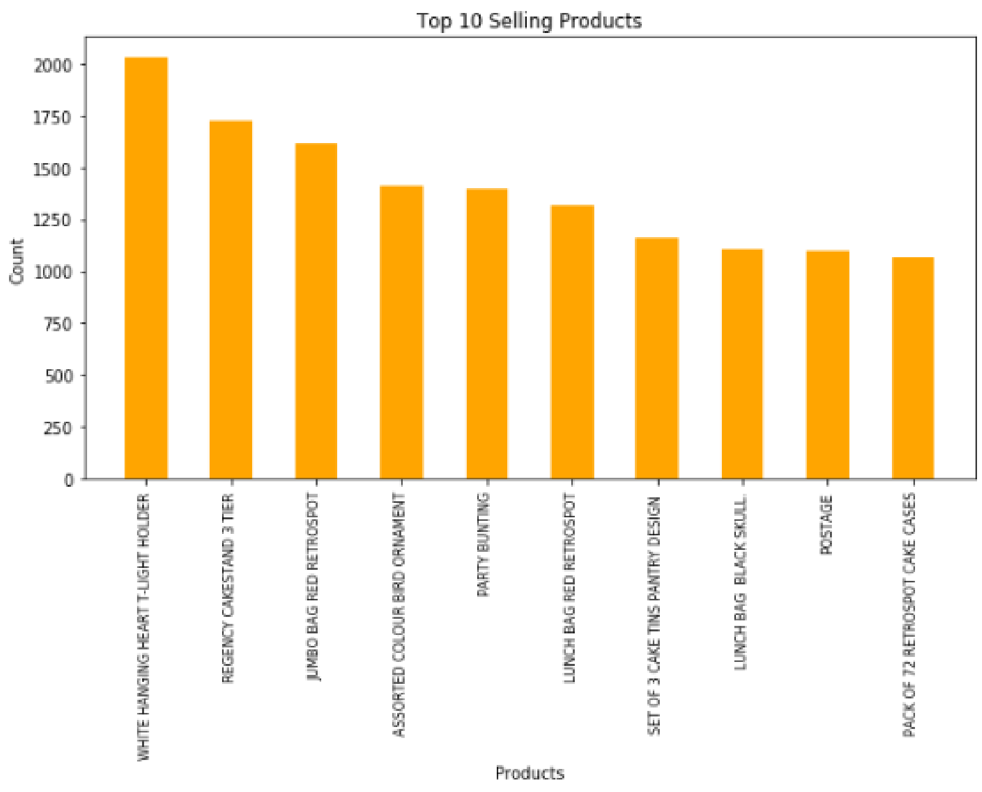


Fig 17

Fig 17: Bar plot to visualize the top 10 selling products.

```
Text(0.5,1,'Least 10 Selling Products')
```

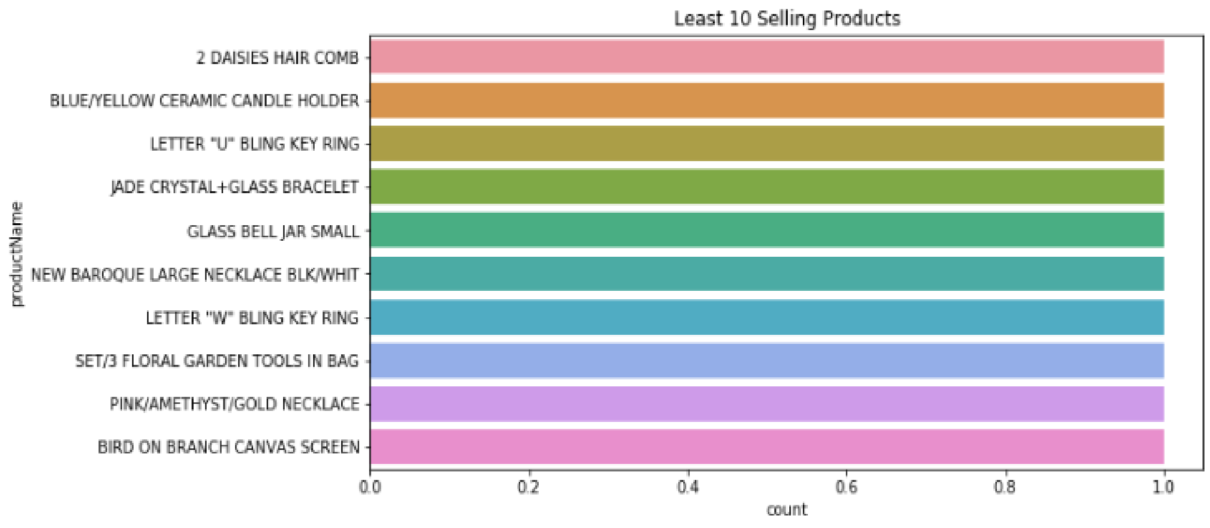


Fig 18

Fig 18: Bar plot to visualize the 10 least sold products.

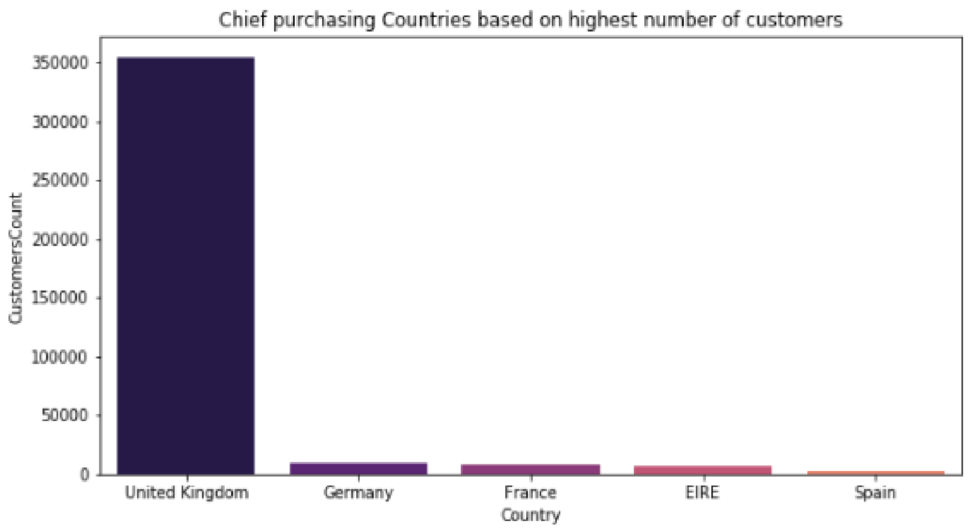


Fig 19

Fig 19: Bar plot to visualize the top 5 countries based on the number of customers.

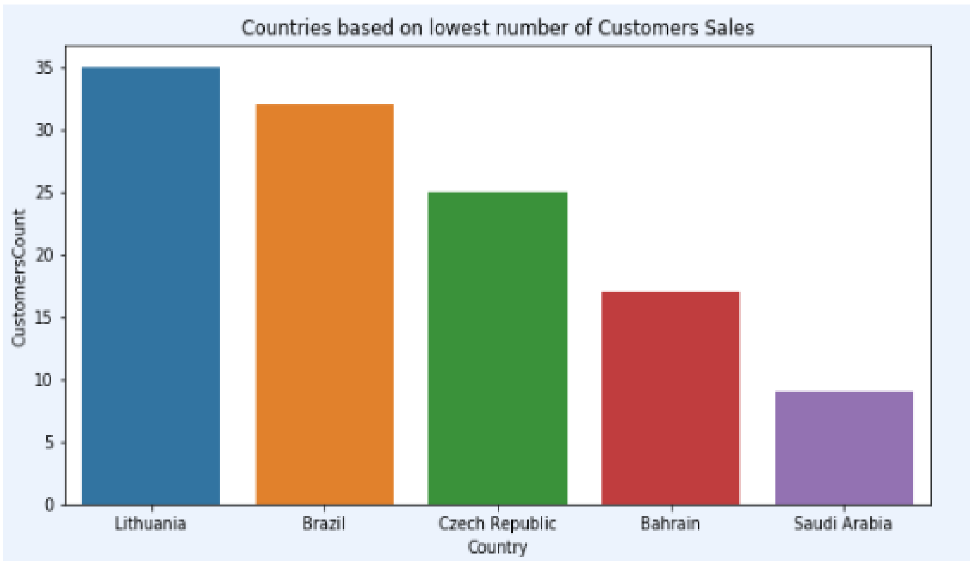
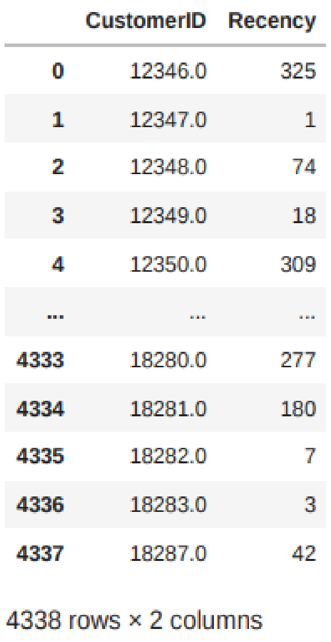


Fig 20

Fig 20: Bar plot to visualize the least 5 countries based on the number of customers.

Building the model:



	CustomerID	Recency
0	12346.0	325
1	12347.0	1
2	12348.0	74
3	12349.0	18
4	12350.0	309
...	...	...
4333	18280.0	277
4334	18281.0	180
4335	18282.0	7
4336	18283.0	3
4337	18287.0	42

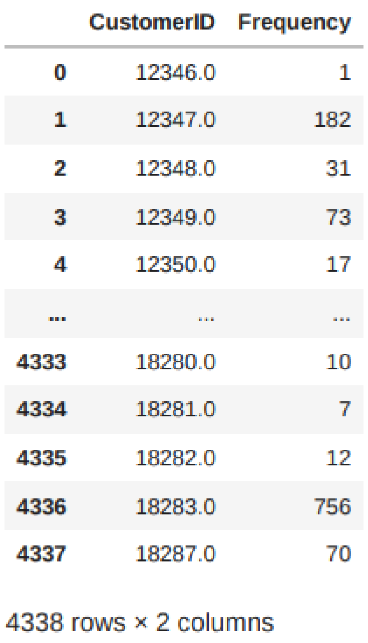
4338 rows × 2 columns

Fig 21

Fig 21: Recency for each customer is calculated. Recency tells us how recently a customer has made a purchase. It measures the time elapsed since the last customer transaction. This is one of the steps while we do RFM analysis.

Considering the exploration done on the data we can see that this data mostly contains customers from 'United Kingdom' that is the reason we cannot segment customers based on demography or geography therefore we choose the 'RFM' ('Recency', 'Frequency' and 'Monetary') Model.

RFM is a popular method for creating and allocating a score to each client in RFM analysis.

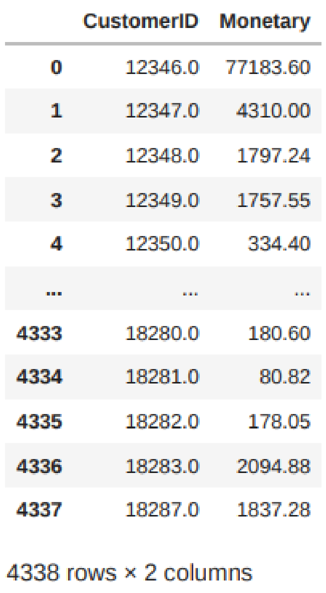


	CustomerID	Frequency
0	12346.0	1
1	12347.0	182
2	12348.0	31
3	12349.0	73
4	12350.0	17
...	...	...
4333	18280.0	10
4334	18281.0	7
4335	18282.0	12
4336	18283.0	756
4337	18287.0	70

4338 rows × 2 columns

Fig 22

Fig 22: Frequency for each customer is calculated. Frequency is how often a customer makes purchases within a specific period. It counts the total number of transactions or visits made by a customer. We group the data by 'CustomerID' and the corresponding 'Frequency' of transactions which results in the total number of transactions made by a customer.

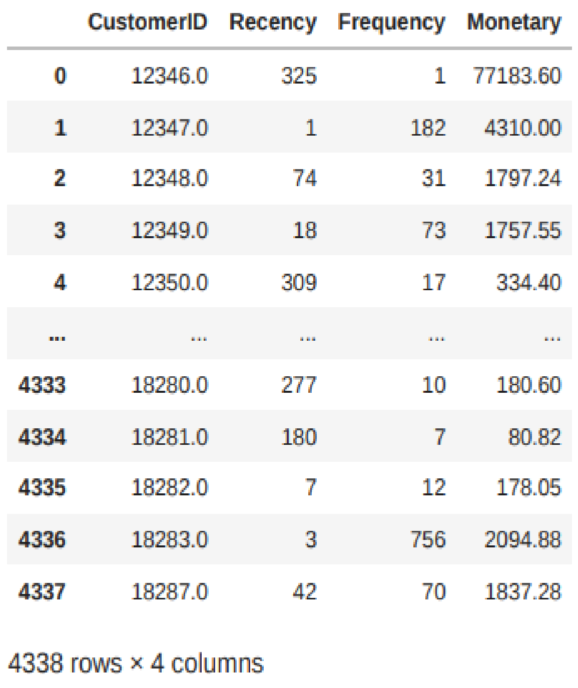


	CustomerID	Monetary
0	12346.0	77183.60
1	12347.0	4310.00
2	12348.0	1797.24
3	12349.0	1757.55
4	12350.0	334.40
...	...	...
4333	18280.0	180.60
4334	18281.0	80.82
4335	18282.0	178.05
4336	18283.0	2094.88
4337	18287.0	1837.28

4338 rows × 2 columns

Fig 23

Fig 23: Monetary for each customer is calculated. Monetary is the total amount a customer has spent within a certain period. It reflects the customer's purchasing power and contribution to revenue. It summarizes customer transaction data by calculating the total monetary value of purchases made by each unique customer.



	CustomerID	Recency	Frequency	Monetary
0	12346.0	325	1	77183.60
1	12347.0	1	182	4310.00
2	12348.0	74	31	1797.24
3	12349.0	18	73	1757.55
4	12350.0	309	17	334.40
...	...	...	...	...
4333	18280.0	277	10	180.60
4334	18281.0	180	7	80.82
4335	18282.0	7	12	178.05
4336	18283.0	3	756	2094.88
4337	18287.0	42	70	1837.28

4338 rows × 4 columns

Fig 24

Fig 24: Combining all the 'RFM' metrics into a single data frame. We are doing this because we can store all the analysis from the RFM in one data frame so that it would be easy to plot.

Analyzing Distribution:

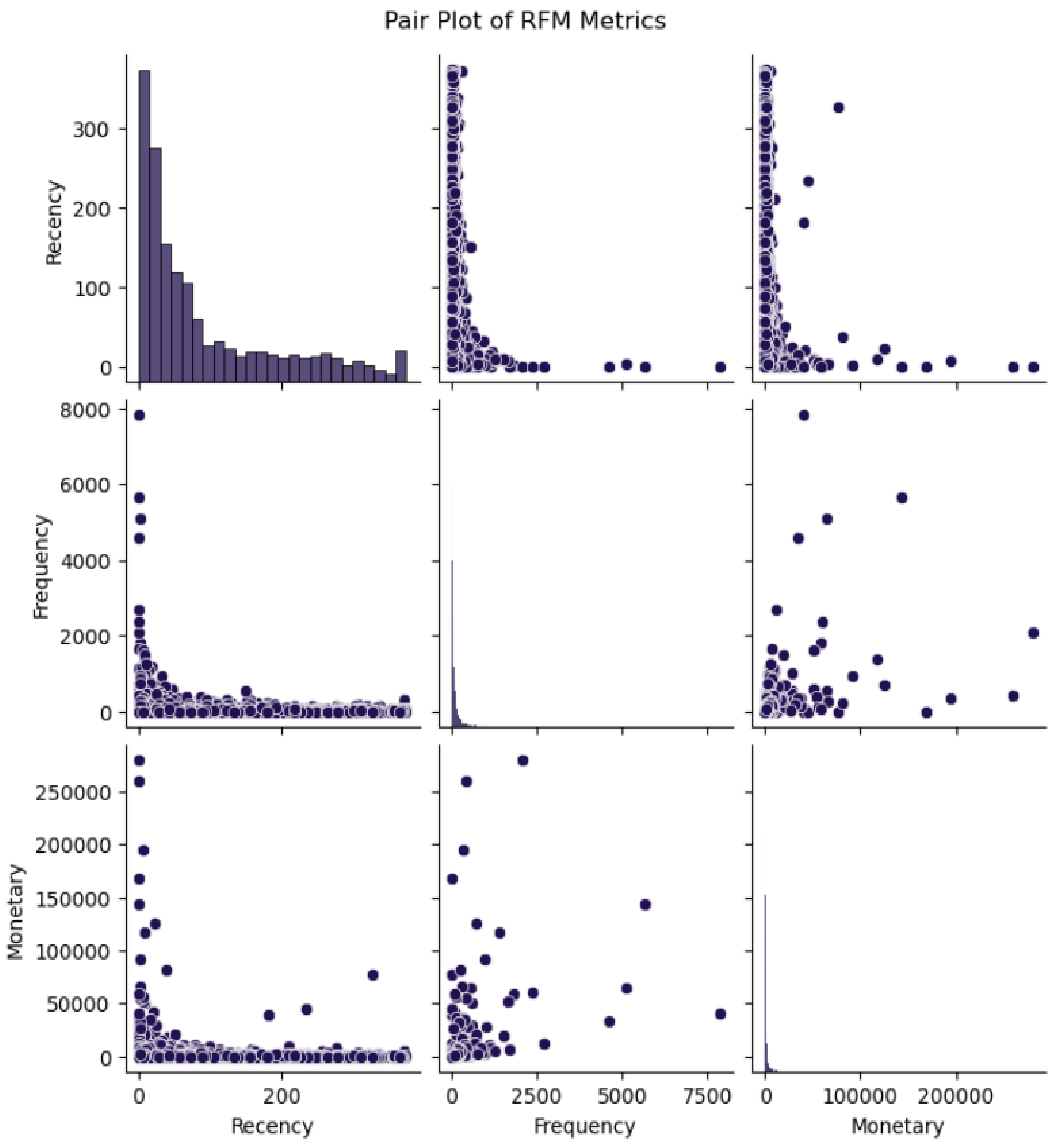
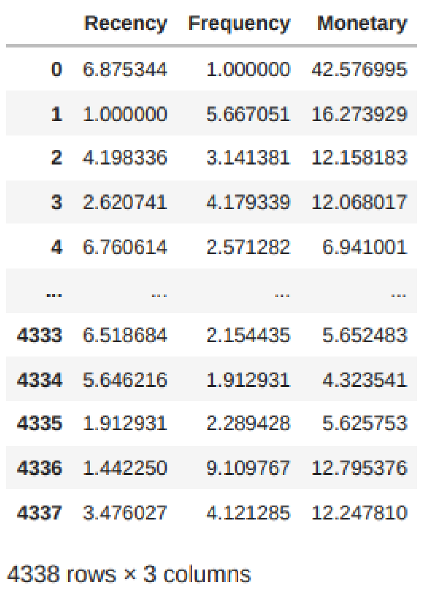


Fig 25

Fig 25: A 'Pair Plot' which displays scatterplots for pairs of variables and histograms for individual variables. In the context of 'RFM' data we can visualize the interrelationships among Recency, Frequency and Monetary metrics in the dataset.



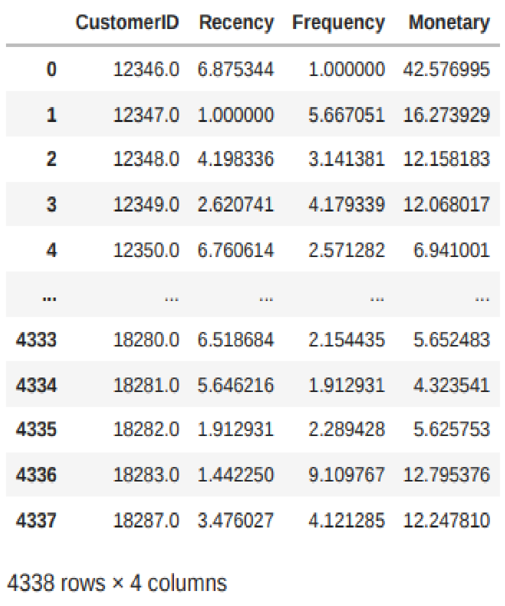
	Recency	Frequency	Monetary
0	6.875344	1.000000	42.576995
1	1.000000	5.667051	16.273929
2	4.198336	3.141381	12.158183
3	2.620741	4.179339	12.068017
4	6.760614	2.571282	6.941001
...	...	...	...
4333	6.518684	2.154435	5.652483
4334	5.646216	1.912931	4.323541
4335	1.912931	2.289428	5.625753
4336	1.442250	9.109767	12.795376
4337	3.476027	4.121285	12.247810

4338 rows × 3 columns

Fig 26

Fig 26: As we see in **Fig 25,** the data lacks symmetry in the distribution when the data is plotted using a pair plot. Observing the distribution, we can infer that it a 'Positive Skew' indicating a bulk of data on the left with a tail to its right.

One of the efficient ways of addressing the skewness is **cube root transformation**. Cube Root Transformation is one of the techniques to address 'Positive Skew' data and is a relatively simple and straightforward method.



	CustomerID	Recency	Frequency	Monetary
0	12346.0	6.875344	1.000000	42.576995
1	12347.0	1.000000	5.667051	16.273929
2	12348.0	4.198336	3.141381	12.158183
3	12349.0	2.620741	4.179339	12.068017
4	12350.0	6.760614	2.571282	6.941001
...	...	...	...	...
4333	18280.0	6.518684	2.154435	5.652483
4334	18281.0	5.646216	1.912931	4.323541
4335	18282.0	1.912931	2.289428	5.625753
4336	18283.0	1.442250	9.109767	12.795376
4337	18287.0	3.476027	4.121285	12.247810

4338 rows × 4 columns

Fig 27

Fig 27: After eliminating the skewness**,** the data frame is concatenated with the CutomerID. The cube root transformation involves taking the cube root of each data point. This transformation is particularly useful when dealing with positively skewed data, as it tends to reduce the right skewness.

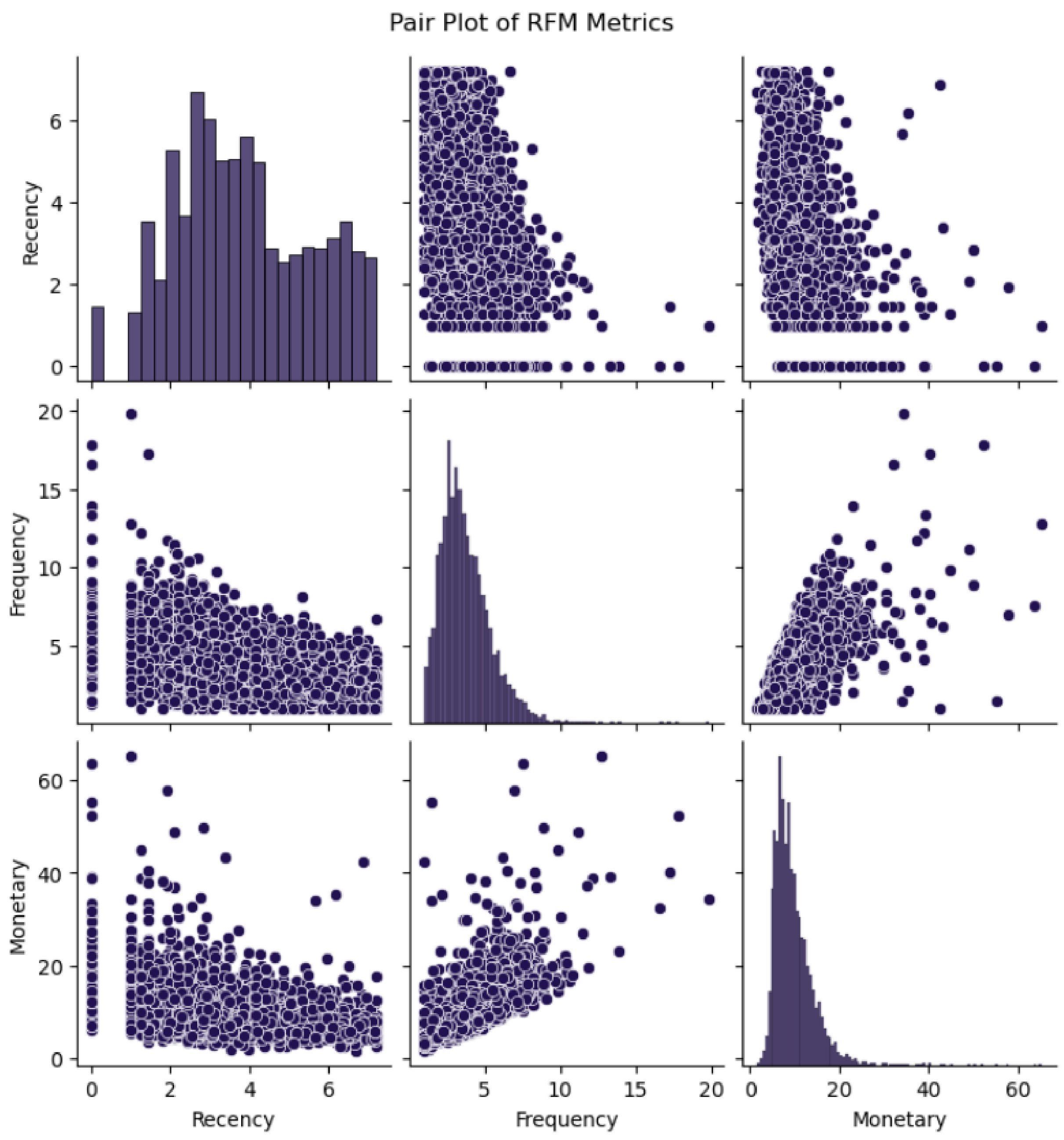
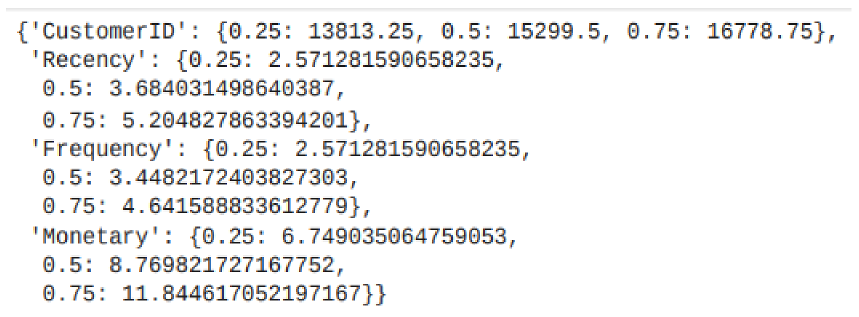


Fig 27

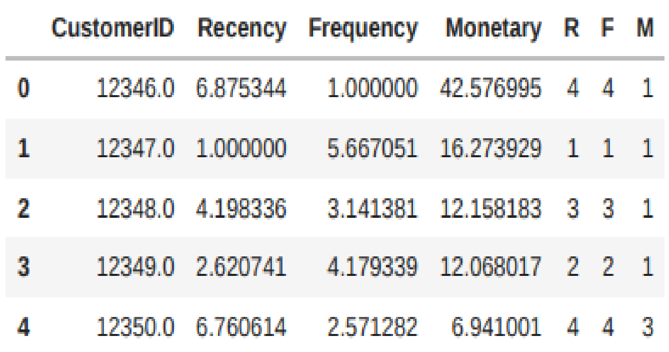
Fig 27: Pair plot of RFM metrics after eliminating the positive skewness.



```
{'CustomerID': {0.25: 13813.25, 0.5: 15299.5, 0.75: 16778.75},
 'Recency': {0.25: 2.571281590658235,
  0.5: 3.684031498640387,
  0.75: 5.204827863394201},
 'Frequency': {0.25: 2.571281590658235,
  0.5: 3.4482172403827303,
  0.75: 4.641588833612779},
 'Monetary': {0.25: 6.749035064759053,
  0.5: 8.769821727167752,
  0.75: 11.844617052197167}}
```

Fig 28

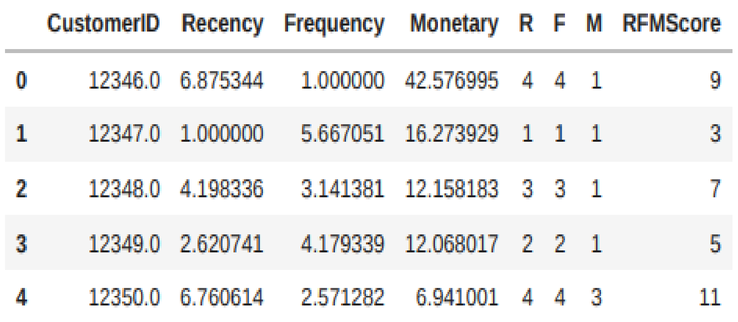
Fig 28: We will be assigning scores to each customer by dividing each RFM Metric into quartiles.



	CustomerID	Recency	Frequency	Monetary	R	F	M
0	12346.0	6.875344	1.000000	42.576995	4	4	1
1	12347.0	1.000000	5.667051	16.273929	1	1	1
2	12348.0	4.198336	3.141381	12.158183	3	3	1
3	12349.0	2.620741	4.179339	12.068017	2	2	1
4	12350.0	6.760614	2.571282	6.941001	4	4	3

Fig 29

Fig 29: Score 1 - Value is less than 25th percentile. Score 2 - Value is greater than 25th percentile but less than or equal to the 50th percentile. Score 3 - Value is greater than 50th percentile but less than or equal to the 75th percentile. Score 4 - Value is greater than the 75th percentile. The above display data is only for the first 5 customers as we used head while displaying.



	CustomerID	Recency	Frequency	Monetary	R	F	M	RFMScore
0	12346.0	6.875344	1.000000	42.576995	4	4	1	9
1	12347.0	1.000000	5.667051	16.273929	1	1	1	3
2	12348.0	4.198336	3.141381	12.158183	3	3	1	7
3	12349.0	2.620741	4.179339	12.068017	2	2	1	5
4	12350.0	6.760614	2.571282	6.941001	4	4	3	11

Fig 30

Fig 30: As we can see that we have assigned scores considering each segment for a customer. Now, we will be combining the scores to get the total 'RFM Score'.

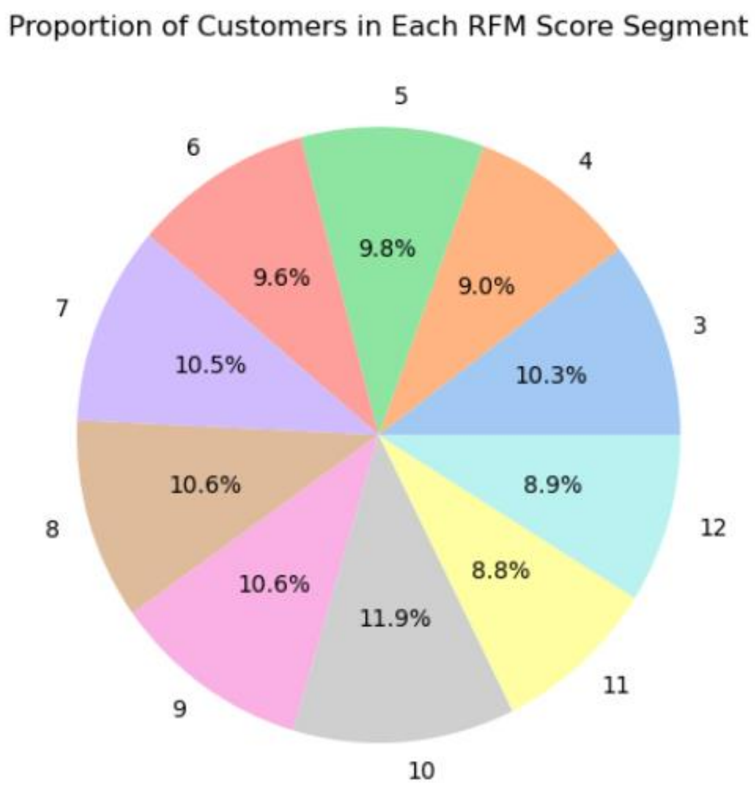


Fig 31

Fig 31: Looking at the proportion of customers in each RFM score segment helps in knowing which RFM score group has the highest number of customers.

Multivariate Analysis/Feature Selection:

We can see a strong positive correlation between 'Frequency' and 'Monetary' (0.72). Indicating that customers who make more frequent purchases also tend to spend more money. This suggests that combining these two metrics can capture a significant portion of customer behavior. The goal of the segmentation project is to enhance customer satisfaction and increase profitability. Frequency and Monetary metrics directly relate to the financial impact of customers on the business, making them relevant for strategic decisions. So, this is the reason we chose frequency and monetary for our segmentation.

We can even answer the question that why only ‘Frequency’ and ‘Monetary’ were the features selected for clustering.

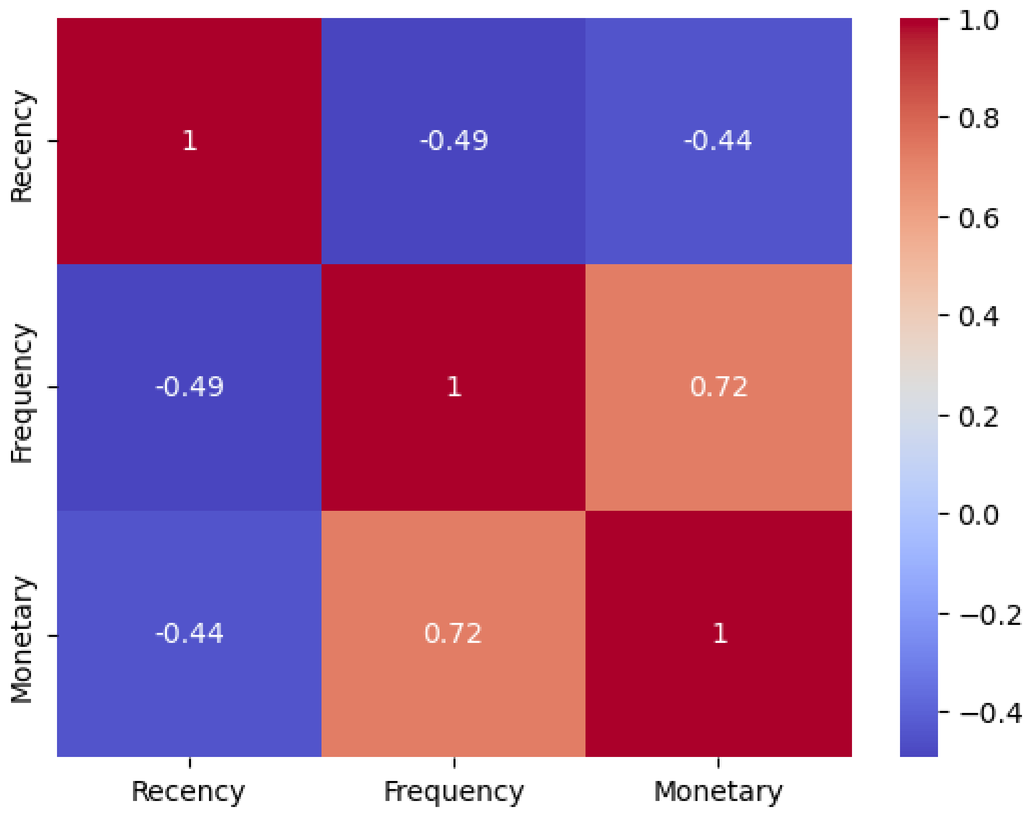


Fig 32

Fig 32: Heatmap depicting correlation between features (RFM Metrics).

Algorithm / Solution Technologies:

- As we know, 'K-Means Clustering' is a popular choice for customer segmentation. Even we will be proceeding with the K-Means algorithm as it is computationally efficient and straightforward. Also 'K-Means' produces easily interpretable results especially when working with numeric data such as RFM metrics.
- Utilizing RFM (Recency, Frequency, Monetary) metrics for segmentation, focusing specifically on monetary and frequency features.
- K-means clustering is ideal for numeric feature-based segmentation, aligning well with RFM metrics, allowing for a data-driven approach.
- K-means is computationally efficient and scalable, making it suitable for handling large datasets with ease. This indicates the algorithm's efficiency.
- K-means provides easily interpretable results, generating distinct clusters with clear boundaries, aiding in understanding segment characteristics.
- K-means allows for iterative improvement of cluster assignments, facilitating adjustments and refinements based on changing business needs or evolving customer behavior.
- The algorithm's simplicity and ease of implementation make it accessible and adaptable to diverse business environments.

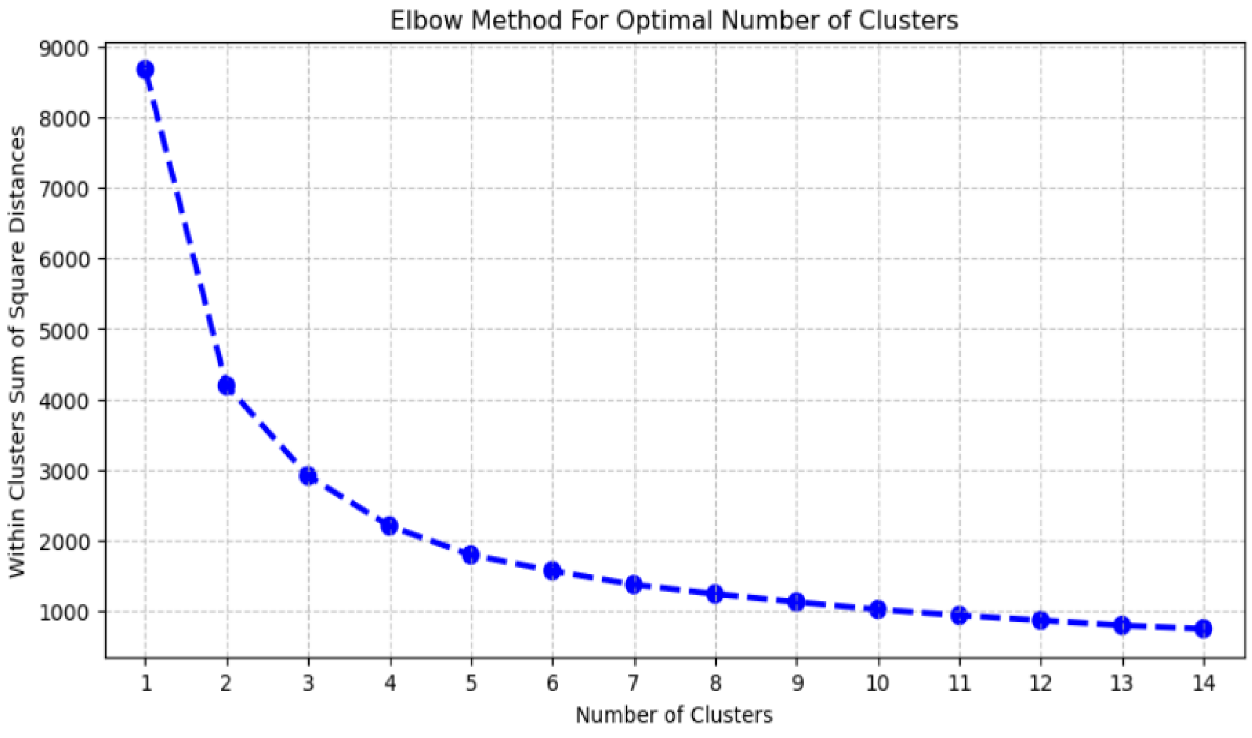


Fig 33

Fig 33: The elbow method is a technique used to find the optimal number of clusters (k) in a K-means clustering analysis. As we can see that this graph displays a discernible "elbow-like" bend where the rate of decrease in the within cluster sum of squares starts to slow down notably. We can see that bend at '2' or '3'. Therefore, first we would be assuming that the optimum number of clusters is '2'.

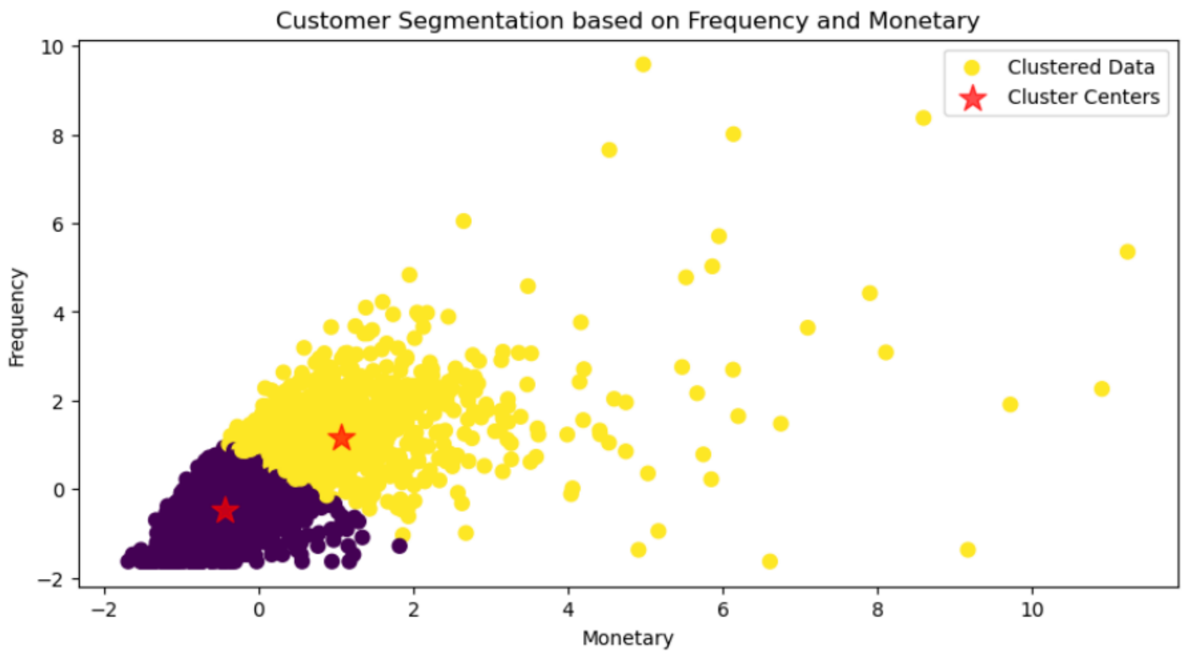


Fig 34

Fig 34: Customer Segmentation based on frequency and monetary. Assuming the ideal number of clusters is equal to 2.

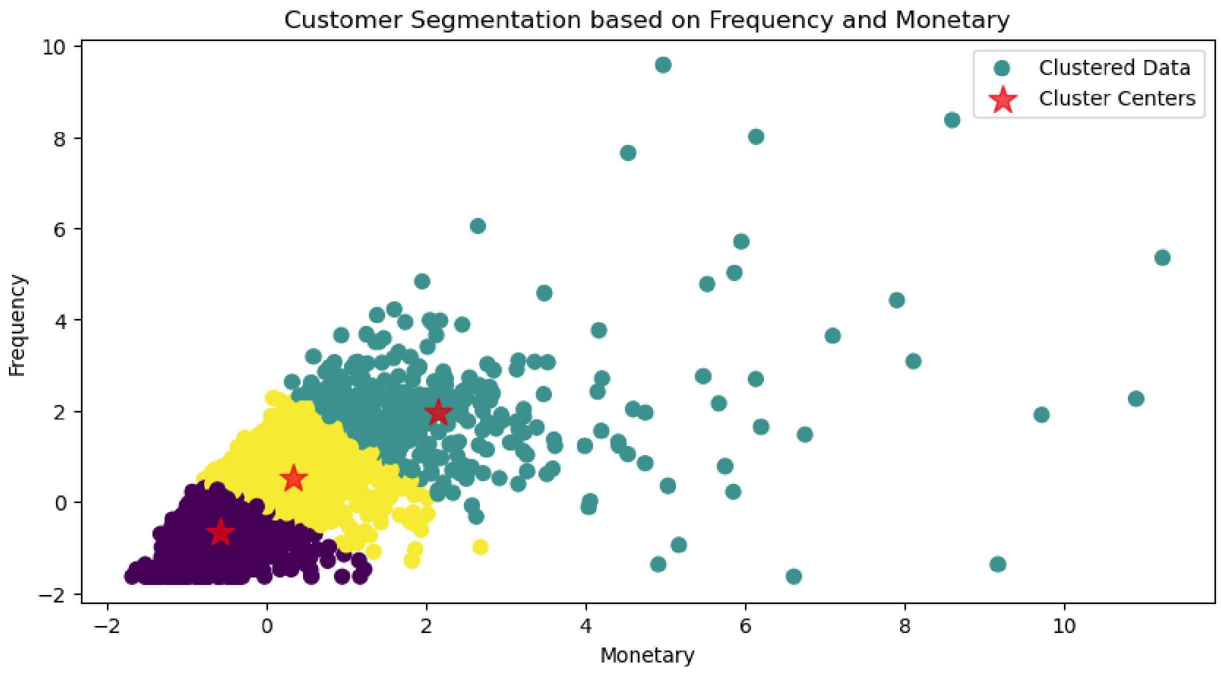
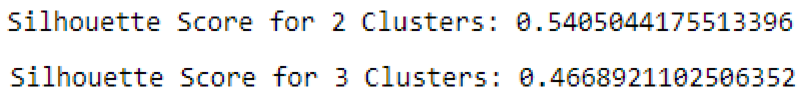


Fig 35

Fig 35: Customer Segmentation based on frequency and monetary. Assuming the ideal number of clusters is equal to 3.

Validation:



```
Silhouette Score for 2 Clusters: 0.5405044175513396
Silhouette Score for 3 Clusters: 0.4668921102506352
```

Fig 36

Fig 36: The silhouette score is a metric used to calculate the goodness of a clustering technique, such as K-means clustering, by measuring how well-separated the clusters are. Silhouette score when the number of clusters is 2 and 3 respectively.

Conclusion/Summary: -

Customer segmentation serves as a pivotal strategy to better comprehend and cater to the diverse needs and behaviors of consumers. By employing machine learning techniques to segment online retail customers based on their purchasing patterns and characteristics, this project aims to derive actionable insights for personalized marketing approaches and enhanced customer satisfaction.
The primary objective is to unravel trends within the retail data, particularly product preferences over a specific timeframe.
Understanding these trends and clustering customers enables the identification of distinct consumer groups, providing a competitive advantage and fostering tailored marketing initiatives.

Segmentation isn't just about dividing customers into groups; it's about gaining a comprehensive understanding of their preferences, challenges, and buying behaviors. This detailed knowledge facilitates the creation of targeted content and strategies that precisely address the unique needs of each segment. Ultimately, through strategic segmentation, the goal is to elevate customer satisfaction, drive profitability, and gain a competitive edge in the online retail landscape.
We found enhanced understanding of the customer base by gaining insights into customer purchases through "Exploratory Data Analysis".
From the cluster analysis segmentation, we can infer the importance of fostering customer loyalty and engagement for maximizing revenue.
Also, we identified two distinct customer segments:

Cluster 1 - Customers with lower spending frequency and monetary value.
Cluster 2 - Customers with higher spending frequency and monetary value.

In summary, segmenting customers based on their spending behavior allows for tailored strategies, fostering stronger customer relationships, increased satisfaction, and improved business performance within the online retail landscape.

Lessons Learned:

Understanding Data Patterns:

Analyzing customer behavior patterns through segmentation taught us how to identify underlying trends and distinguish distinct customer groups.

Normalizing the data:

Even addressing ‘Skewness’ was the biggest challenge we faced. We had to get along with new methods to address the ‘Skewness’.
If possible, we want to try normalizing the data using other methods like ‘Log Transformation’ and ‘Square Root Transformation’ the next time and compare the results.

All things considered, our exploration of consumer segmentation through unsupervised learning—specifically, K-means clustering with RFM metrics—has revealed the incredible potential of this methodology. It greatly facilitated the process of identifying various client segments from a business standpoint, according to their spending habits and purchasing patterns. We would be able to customize marketing plans and provide individualized methods that addressed the needs of segments thanks to this independent strategy. This experience demonstrated how important unsupervised learning is for deriving useful insights from complex datasets, which improves our comprehension of consumer behavior and makes data-driven decision-making possible.

Risks:

The data we are dealing with has approximately more than 5 lakh records. So, while dealing with this huge load of data, the data needs to be cleaned and preprocessed before analyzing it as it might have a lot of anomalies which might lead to inaccurate results. Incomplete, inaccurate, or inconsistent data can lead to biased analysis and incorrect conclusions. Ensuring data quality through proper validation, cleaning, and preprocessing is essential.

Challenges:

As mentioned above, we would be dealing with a huge volume of data. Handling and analyzing large volumes of data efficiently can be challenging and can be computationally intensive. Processing, cleaning, aggregating, and extracting patterns from such a large dataset can be time-consuming. Efficiently representing and summarizing data while maintaining visual clarity is a common challenge in real-time.

Team Contributions:

Almost all the team members contributed equally to the project.

Rupa Dinesh Lnu (30%)

Groundwork for the idea of Customer Segmentation problem, EDA and analyzing customer trends in data, writing reports and making presentations etc.

Krishna Koushik Parimi (35%)

Collecting and analyzing dataset, choosing and building the appropriate model, selecting features, analyzing segments etc.

Ravi Tarun Prasad Nimmalapudi (35%)

Parts of pre-processing data, normalizing the data, performing segmentation, evaluating the model, parts of making presentations and reports.

Citations:

https://www.kaggle.com/datasets/vjchoudhary7/customer-segmentation-tutorial-in-python

Ref - We came across a similar problem of customer segmentation, but this data only contained 200 observations or so, which we felt wouldn't be enough for a thorough analysis.