INFORMATICS INSTITUTE OF TECHNOLOGY

UNIVERSITY OF WESTMINSTER

# 6BUIS017W.1

# Customer Relationship and Change Management

CW - 1

Module Leader: Mr. Trevor Soris

Degree: BSc (Hons) in Business Data Analytics

Name:

Ahmed Ammar Ismeth Mohideen

UOW ID - w1869538

IIT ID – 20210536

## Task 1 – A

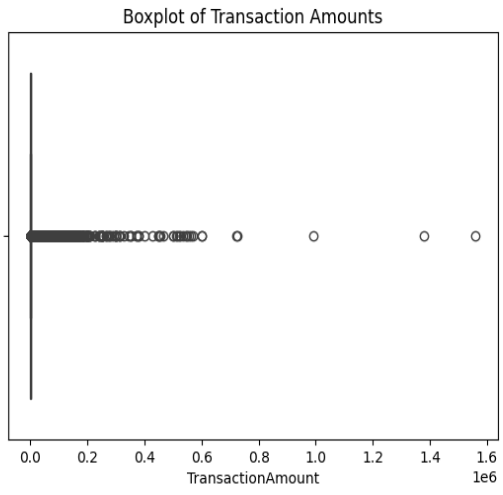
CustomerDOB,CustGender,CustLocation and CustAccountBalance each had null values before cleaning was done, CustDOB was then dropped as assigning a default value could lead to mislead downstream processing. CustGender,CustLocation were then imputed with mode to preserve the data distribution and avoid data loss. CustAccountBalance was imputed by median as it was skewed numerical data and this ensure to maintain the central tendency without being influenced by outliers.

```
Null values after cleaning:
TransactionID         0
CustomerID            0
CustomerDOB           0
CustGender            0
CustLocation          0
CustAccountBalance    0
TransactionDate       0
TransactionTime       0
```

## Task 1 – B

Using Inter quartile range is a common method to deal with removing outliers for transaction amounts as it focuses on the middle 50% of data(bulk typical transactions), it handles skewed distributions effectively and is an widely accepted easy to implement data cleaning technique.

```
import seaborn as sns
import matplotlib.pyplot as plt # Import the matplotlib library

# Boxplot Before Cleaning (Before removing outliers)
sns.boxplot(data = df2, x = 'TransactionAmount')

plt.title('Boxplot of Transaction Amounts')
plt.show()
```



**Removing Invalid Transaction Amounts**

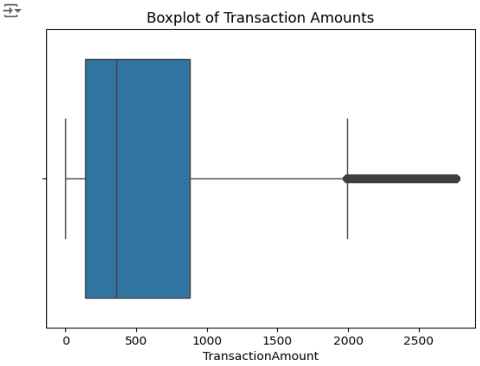
```
# Step 1: Apply IQR method to detect and remove outliers
Q1 = df2['TransactionAmount'].quantile(0.25)
Q3 = df2['TransactionAmount'].quantile(0.75)
IQR = Q3 - Q1

# Define lower and upper bounds
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

print(lower_bound)
print(upper_bound)
```

```
-1400.0
2760.0
```

```
#Remove transaction amounts outside the IQR range
df3 = df2[(df2['TransactionAmount'] > lower_bound) & (df2['TransactionAmount'] < upper_bound)]

# Display the cleaned dataset size
print(f"Dataset size after removing invalid transaction amounts: {df3.shape}")
```

```
Dataset size after removing invalid transaction amounts: (933919, 9)
```

```
# Boxplot After Cleaning (After removing outliers)
sns.boxplot(data = df3, x = 'TransactionAmount')
plt.title('Boxplot of Transaction Amounts')
plt.show()
```



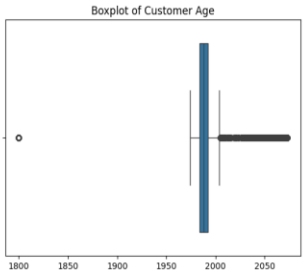

## Task 1 – C

The invalid age was dealt with by separating the year of birth and this added a new useful feature to the dataset where the age was calculated based on the year in concern during when the transactions occurred. The IQR method was used to remove outliers and unrealistic age values. Negative ages and ages above 110 years or ages too young were also considered an error as values were assumed to be taken inside a given threshold to ensure data reflects realistic ages.

```
df3['CustYearOfBirth'] = pd.DatetimeIndex(df3['CustomerDOB']).year
```

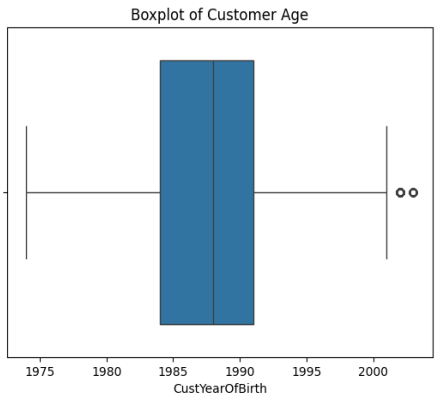
```
<ipython-input-17-0b561a2693ed>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df3['CustYearOfBirth'] = pd.DatetimeIndex(df3['CustomerDOB']).year
```

```
# Boxplot Before Cleaning (Before removing outliers)
sns.boxplot(data = df3, x = 'CustYearOfBirth')

plt.title('Boxplot of Customer Age')
plt.show()
```




Connected to Python 3 Google Compute Engine backend

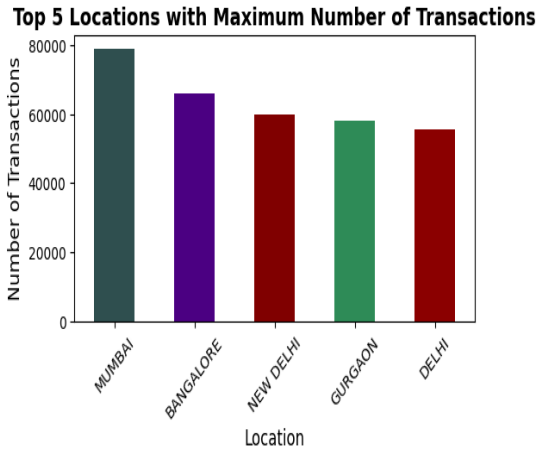
```
# Boxplot After Cleaning (Before removing outliers)
sns.boxplot(data = df4, x = 'CustYearOfBirth')

plt.title('Boxplot of Customer Age')
plt.show()
```




## Task 1 – C

The code identifies and displays the top 5 locations with the highest number of transactions from the dataset (df4). It uses value_counts() to count occurrences of each location, selects the top 5, and visualizes the result using a bar plot. Mumbai had the highest transactions while Delhi had the 5$^{th}$ highest.

```
# Display the top 5 locations with the maximum number of transactions
top_locations = df4['CustLocation'].value_counts().head(5)

# Display the result
print("Top 5 Locations with Maximum Number of Transactions:")
print(top_locations)

# Define a list of colors
d_colors = ['#2F4F4F', '#4B0082', '#800000', '#2E8B57', '#8B0000']  # Dark colors

# Create a bar plot for the top 5 locations
top_locations.plot(kind='bar', color=d_colors, figsize=(6, 3))

# Title and labels for the plot
plt.title('Top 5 Locations with Maximum Number of Transactions', fontsize=14, fontweight='bold')
plt.xlabel('Location', fontsize=12)
plt.ylabel('Number of Transactions', fontsize=12)
plt.xticks(rotation=45)

# Show the plot
plt.show()
```




## Task 2 – A

**Recency metric** is about the duration of a customer's purchase. It would be used to measure a period of time since their last purchase. In usual RFM analysis, customers who have acted more recently are considered more values since they may be in a better position to order again or respond well to marketing campaigns. Given that, Recency in this exercise is the time that has elapsed since the most recent purchase of the member.

**Frequency metric** refers to the number of times a customer has contacted a product or service or completed a purchase in a specific period. This is a measure that determines how deeply involved a customer is, or how loyal. The consumers who have contact with a brand more frequentlyor who make purchases regularly are normally considered more precious and loyal. Considering the present exercise Frequency is how many times members purchased during the last period.

**Monetary metric** is the total dollar amount that a customer has spent on purchases over some time. It is an indicator of how much money a consumer is worth. Customers who have spent more with the company in the recent past are often considered the most valuable to a company. Considering this exercise in mind: Monetary is the member's money purchases in the previous period.

```
# RFM analysis and group transactions by Customerid
rfm_data = pd.read_sql(''' SELECT CustomerID,
                  JULIANDAY('2016-12-31') - JULIANDAY(MAX(TransactionDate)) AS recency,
                  COUNT(*) AS frequency,
                  SUM(TransactionAmount) AS monetary
            FROM data
            WHERE TransactionAmount != 0
            GROUP BY CustomerID ''', conn)
rfm_data
```

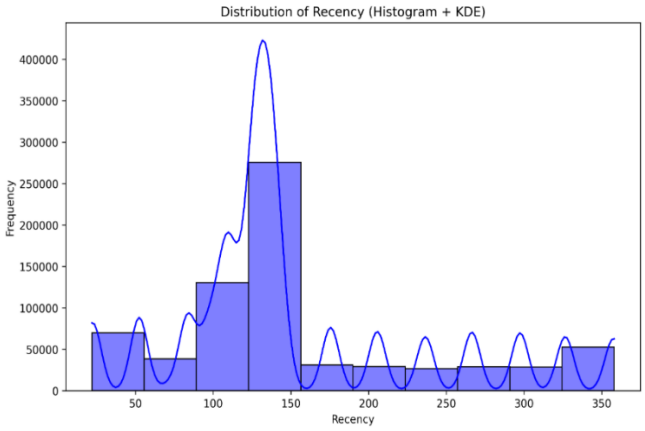
| | CustomerID | recency | frequency | monetary |
|---|---|---|---|---|
| 0 | C1010011 | 114.0 | 1 | 356.0 |
| 1 | C1010012 | 139.0 | 1 | 1499.0 |
| 2 | C1010014 | 176.0 | 2 | 1455.0 |
| 3 | C1010018 | 107.0 | 1 | 30.0 |
| 4 | C1010028 | 124.0 | 1 | 557.0 |
| ... | ... | ... | ... | ... |
| 711596 | C9099836 | 176.0 | 1 | 691.0 |
| 711597 | C9099877 | 107.0 | 1 | 222.0 |
| 711598 | C9099919 | 135.0 | 1 | 126.0 |
| 711599 | C9099941 | 125.0 | 1 | 50.0 |
| 711600 | C9099956 | 97.0 | 1 | 855.0 |

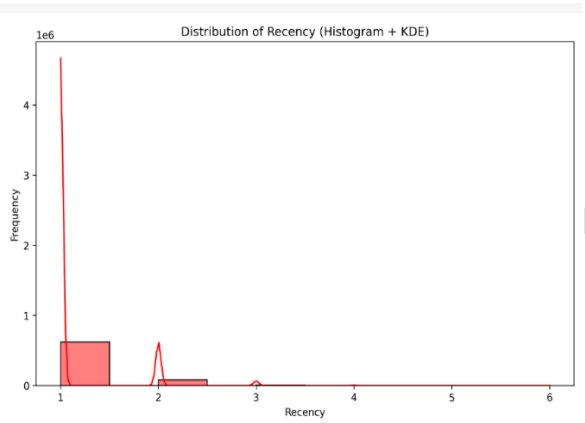
711601 rows × 4 columns

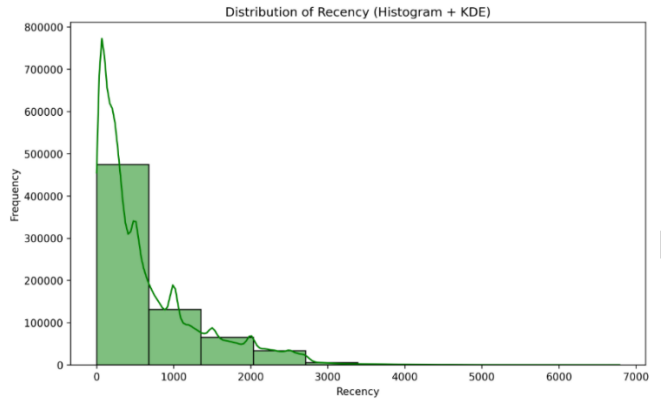
This query is designed to perform RFM analysis, which is a customer segmentation technique often used in marketing to evaluate and group customers based on transaction behaviors. The assumption taken here for recency is that the query compares the last transaction date (MAX(TransactionDate)) for each customer against **December 31, 2016**, the reference date as at the end of the transaction year.

Some insights would be that lower recency values could indicate customer has made a recent transaction, making them relatively active and vice versa. Higher frequency means customer makes frequent purchases and likely loyal and vice versa. Higher monetary values means that customer has spent more money indicating a high - value customer.

## Task 2 – B



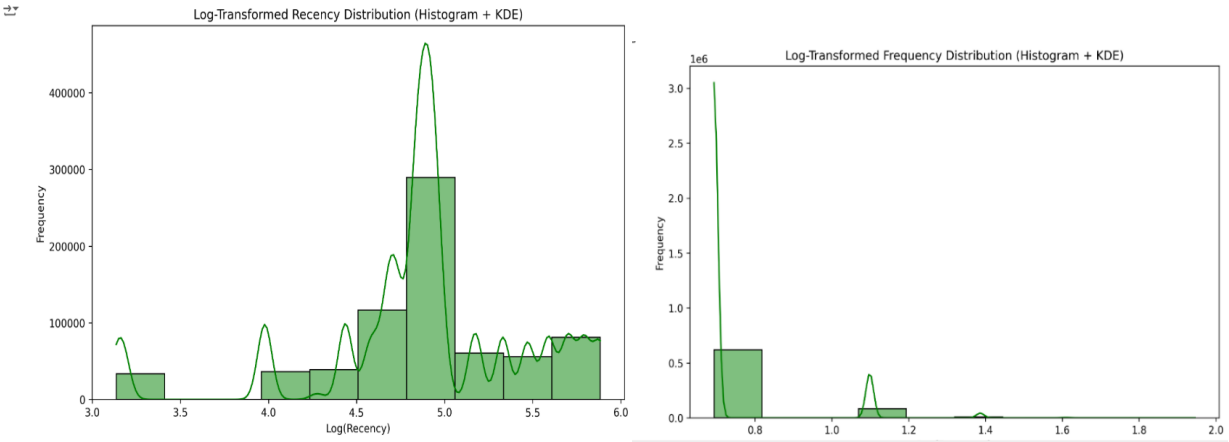



This visualization helps to understand the distribution of each metric and identify patterns such as skewness or unusual concentrations in data. The data in the 3 plots seem positively skewed with most customers clustering at one end (recent customers, frequent buyers or high spenders). The plots come in handy as it helps to form customer segments in order to perform target marketing or customer retention strategies.

## Task 2 – C

Skewness refers to the asymmetry in the distribution of data and means that data is not evenly distributed around the mean and the tail of the distribution is longer on one side. In context it distort central tendency measures like the mean and could misinterpret the typical value. For instance, a few high spenders could disproportionately increase average spend, making it less representative of majority customers. It could also mean outliers or extreme values in the tail and can influence statistical measures. This could impact hypothesis testing used for marketing campaigns as well.

After the data was normalized and skewness was delt with the following results were obtained



Skewness before normalization :

| | 0 |
|---|---|
| recency | 0.993699 |
| frequency | 2.896057 |
| monetary | 1.612569 |

dtype: float64

Skewness after normalization :

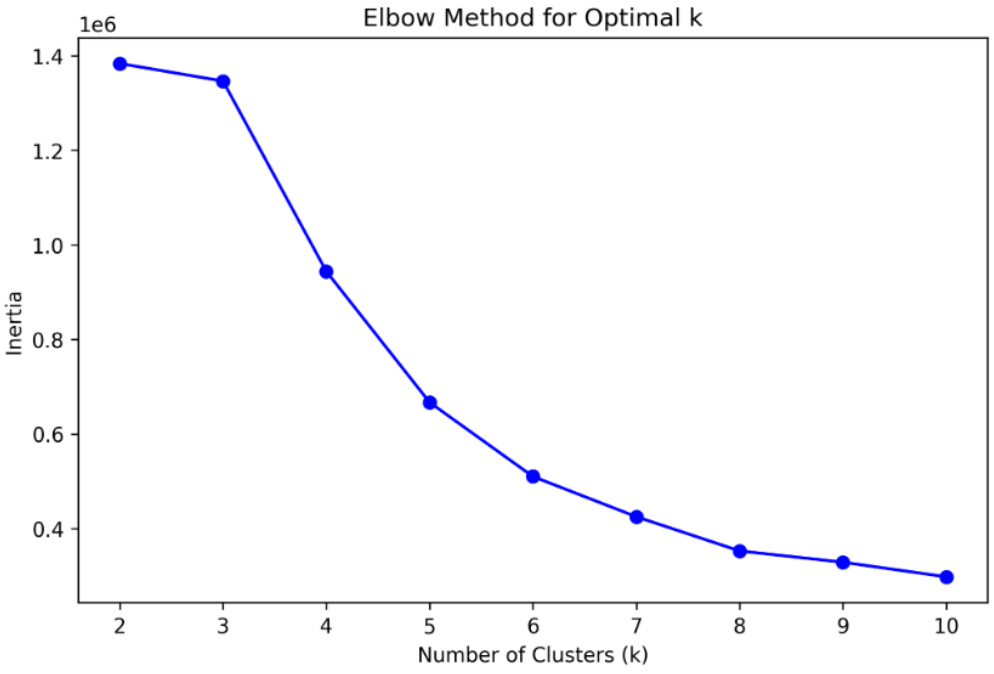
| | 0 |
|---|---|
| recency | 0.993699 |
| frequency | 2.896057 |
| monetary | 1.612569 |
| log_recency | -0.889852 |
| log_frequency | 2.531610 |
| log_monetary | -0.721287 |

dtype: float64

## Task 3 – A

K-means clustering is a widely used unsupervised machine learning algorithm that is used for segmentation tasks and it is a reasonable choice for dividing customers into clusters based on their RFM metrics. Som of it's strengths are that it's well-suited for numerical data as each metric can be treated as a numerical feature thus allowing the algorithm to compute distances effectively, it's efficiency and scalability can handle large datasets and this is a practical choice for customer segmentation, segmentation assigns customers to clusters ensuring similar behaviors among customers in terms of recency, frequency and monetary, cluster numbers can be adjusted based on business requirement. Customers can be classified into loyal, at-risk and average customers etc.

## Task 3 – B



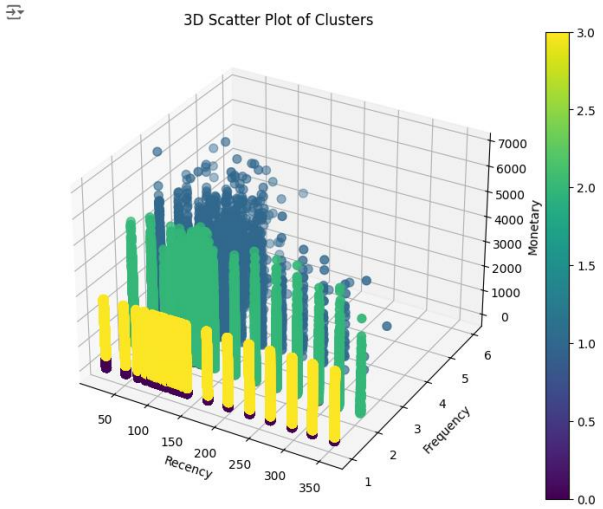
The elbow method was used as it is a straightforward way to determine numbers of clusters by plotting the within-cluster sum of squares (inertia) against the number of clusters. It also quantifies how compact a cluster is. It focuses on diminishing returns as when the clusters increase the WCSS decreases as data points are assigned to closer centroids. It prevents overfitting that comes as a result of too many clusters. The elbow method can be applied to any clustering algorithm that computes cluster compactness.

## Task 3 – C

```
# Use the optimal k
optimal_k = 4
kmeans = KMeans(n_clusters=optimal_k, random_state=42)
kmeans.fit(rfm_scaled)
```

```
KMeans
KMeans(n_clusters=4, random_state=42)
```

K-means was computed using the optimal clusters as 4.




The following 3D Scatter scatter plot was obtained for the RFM showing the clear number of clusters and customer groupings based on similarities between those groups.

# Task 4

**Identifying Top Locations:**

The code identifies the top 5 locations with the most transactions using df5['CustLocation'].value_counts().head(5). This will give you an idea about the locations where your business has the highest concentration of customers.

This information is highly valuable for running focused marketing campaigns, resource allocation, and understanding regional customer behavior.

**Cluster Distribution within Top Locations:**

By combining the results of the RFM analysis with location data, you can analyze how the different customer segments (clusters) are distributed across these top locations. The top5_location_data will help in identifying which clusters dominate at each location. This enables you to tailor your strategies to specific locations and customer groups within those locations.

For example, if any location has a high volume of "Loyal Customers", you might want to run retention strategies and loyalty programs more aggressively in that location.

**Dominant Cluster in Top Locations:**

max_transaction_clust and max_transaction_clust_sorted show the cluster with the most transactions inside each of the top locations.

This identifies the dominant type of customer in each of these locations and gives valuable insight into the primary form of customer behavior and preferences in those locations.

**Cluster Profiling:**

clust_profile gives the average Recency, Frequency, and Monetary values for each cluster. This characterizes each customer segment based on their purchasing patterns.

For instance, "Loyal Customers" can have high frequency and monetary values, indicating frequent and high-value purchases. "Dormant Customers" can have high recency, suggesting a lack of recent activity.

Labeling Clusters:

**Identifying Top Locations:**

The code identifies the top 5 locations with the most transactions using df5['CustLocation'].value_counts().head(5). This gives you an idea of the location where your business has the highest concentration of customers.

This information is important for targeted marketing campaigns, resource allocation, and understanding regional customer behavior.

**Cluster Distribution within Top Locations**:

You can merge the RFM analysis results, rfm_analysis, with location data, df5, to analyze how different customer segments or clusters are distributed across these top locations.

top5_location_data would give insight into which cluster is dominant in a given location. This would provide an opportunity to have a more location-based strategy while targeting specific customer groups within these locations.

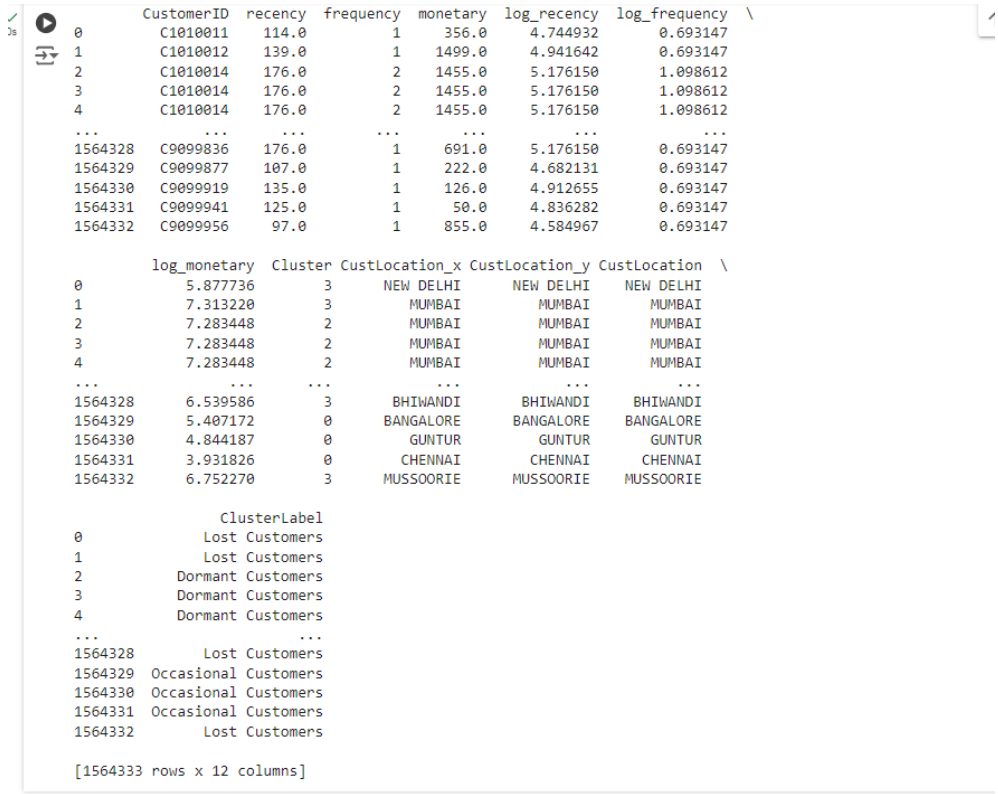
For example, if a location has more "Loyal Customers", you might focus on retention strategies and loyalty programs within that

**Targeted Marketing**: Utilize cluster information and location data to target specific customer segments in each location with appropriate marketing campaigns.

**Customer Retention**: Emphasize strategies for retaining "Loyal Customers" and reactivating "Dormant Customers" in their respective locations.

**Customer Service**: Adapt customer service approaches based on the dominant cluster in a location.

**New Customer Acquisition**: Target marketing efforts toward the acquisition of new customers in high-potential locations based on cluster distribution.



| | CustomerID | recency | frequency | monetary | log_recency | log_frequency | log_monetary | Cluster | CustLocation_x | CustLocation_y | CustLocation | ClusterLabel |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | C1010011 | 114.0 | 1 | 356.0 | 4.744932 | 0.693147 | 5.877736 | 3 | NEW DELHI | NEW DELHI | NEW DELHI | Lost Customers |
| 1 | C1010012 | 139.0 | 1 | 1499.0 | 4.941642 | 0.693147 | 7.313220 | 3 | MUMBAI | MUMBAI | MUMBAI | Lost Customers |
| 2 | C1010014 | 176.0 | 2 | 1455.0 | 5.176150 | 1.098612 | 7.283448 | 2 | MUMBAI | MUMBAI | MUMBAI | Dormant Customers |
| 3 | C1010014 | 176.0 | 2 | 1455.0 | 5.176150 | 1.098612 | 7.283448 | 2 | MUMBAI | MUMBAI | MUMBAI | Dormant Customers |
| 4 | C1010014 | 176.0 | 2 | 1455.0 | 5.176150 | 1.098612 | 7.283448 | 2 | MUMBAI | MUMBAI | MUMBAI | Dormant Customers |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 1564328 | C9099836 | 176.0 | 1 | 691.0 | 5.176150 | 0.693147 | 6.539586 | 3 | BHIWANDI | BHIWANDI | BHIWANDI | Lost Customers |
| 1564329 | C9099877 | 107.0 | 1 | 222.0 | 4.682131 | 0.693147 | 5.407172 | 0 | BANGALORE | BANGALORE | BANGALORE | Occasional Customers |
| 1564330 | C9099919 | 135.0 | 1 | 126.0 | 4.912655 | 0.693147 | 4.844187 | 0 | GUNTUR | GUNTUR | GUNTUR | Occasional Customers |
| 1564331 | C9099941 | 125.0 | 1 | 50.0 | 4.836282 | 0.693147 | 3.931826 | 0 | CHENNAI | CHENNAI | CHENNAI | Occasional Customers |
| 1564332 | C9099956 | 97.0 | 1 | 855.0 | 4.584967 | 0.693147 | 6.752270 | 3 | MUSSOORIE | MUSSOORIE | MUSSOORIE | Lost Customers |

[1564333 rows x 12 columns]

# Task 5

Customer Dimension, Transaction Dimension, Date Dimension, Marketing - Campaign Dimension, Loyalty Dimension

**Justification of Dimensions**

**Customer Dimension**

- Captures individual customer attributes such as demographics: age, gender, location, and segmentation details.
- Enables location-based analysis that includes determining the high-revenue places.

**Transaction Dimension**

- Essential for analyzing purchasing patterns, average spend, and product preferences
- Facilitates profitability analysis by linking transactions to marketing campaigns or customer segments.

**Date Dimension**

- Provides a temporal context to transactions and campaigns for time-series analysis.
- Tracks attributes such as day, week, month, quarter, year, and holiday indicators.

**Marketing - Campaign Dimension**

- Captures marketing campaigns, including, but not limited to, campaign ID, name, target audience, duration, and channels used-e.g., email, social media
- Links campaigns to the right transaction and customers for campaign performance analysis.

**Loyalty Dimension**

- Captures customer loyalty attributes such as loyalty program enrollment, tier levels, and points earned/redeemed.
- Enables tracking of customer retention rates and loyalty-driven revenue.

**Justification of Measures**

**Customer Dimension - Churn Rate:**

Critical for identifying segments at risk of disengagement and improving retention strategies.

**Transaction Dimension - Total Sales Revenue:**

Fundamental metric for evaluating business performance.

**Date Dimension - Sales by Day/Week/Month/Quarter/Year:**

Identifies seasonal trends and peak periods for targeted campaigns.

**Marketing-Campaign - Response Rate:**

Indicates campaign engagement success and helps refine targeting strategies.

**Loyalty Dimension  - Points Earned/Redeemed:**

Tracks customer participation in loyalty programs and engagement.