Capstone Project - Project Online Retail

Table of Contents

1.Problem Statement

The company is suffering from a financial crisis, and they are looking to come out of the crisis by analyzing their customer purchase pattern using the various modelling methods and overcome the crisis.

2.Project Objective

The objective of this project is to try and understand the various customer purchase patterns for their firm based on the insights by analyzing the data and know the most profitable product.

3.Data Description

The dataset available is a Secondary dataset provided by the Intellipaat's support team for the project and study purpose.

The data consists of 8 columns:

1.InvoiceNo: Invoice number. Nominal, a 6-digit integral number uniquely assigned to each transaction. If this code starts with letter 'c', it indicates a cancellation.

2.StockCode: Product (item) code. Nominal, a 5-digit integral number uniquely assigned to each distinct product.

3.Description: Product (item) name. Nominal.

4.Quantity: The quantities of each product (item) per transaction. Numeric.

5.InvoiceDate: Invoice Date and time. Numeric, the day and time when each transaction was generated.

6.UnitPrice: Unit price. Numeric, Product price per unit in sterling.

7.CustomerID: Customer number. Nominal, a 5-digit integral number uniquely assigned to each customer.

8.Country: Country name. Nominal, the name of the country where each customer resides.

4.Data Preprocessing Steps And Inspiration

The preprocessing of the data included the following steps:

1. Reading the CSV file.

```
data = pd.read_csv("OnlineRetail.csv", encoding = "ISO-8859-1")
print(data.info())
data.head()
```

```
<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  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
None
```

2. Checking the data types and Null value information using info

```
print(np.sum(data.isnull()), "\n")
print("Percentage of customers missing: ", round(data['CustomerID'].isnull().sum() * 100 / len(data),2),"%" )
```

```
InvoiceNo           0
StockCode           0
Description      1454
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     135037
Country             0
dtype: int64

Percentage of customers missing:  25.16 %
```

3. Data cleaning

4.Checking for duplicates if any present and dropping them.

```
print("Shape before dropping duplicates", data.shape)
data = data.drop_duplicates()
print("Shape after dropping duplicates", data.shape)
```

```
Shape before dropping duplicates (541909, 8)
Shape after dropping duplicates (536641, 8)
```

5. Checking Data shape and Unique Values in different columns.

6. Checking for nulls in CustomerID.

```
print(np.sum(data.isnull()), "\n")
print("Percentage of customers missing: ", round(data['CustomerID'].isnull().sum() * 100 / len(data),2),"%" )
```

```
InvoiceNo           0
StockCode           0
Description      1454
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     135037
Country             0
dtype: int64

Percentage of customers missing:  25.16 %
```

5.Choosing the Algorithm For the Project

Description for the Online Retail algorithm for the project.

I have chosen the RFM segmentation algorithm for this project for the following reasons:

1. Used to analyze customer Value.
2. Recency of the customer.
3. Frequency of the bookings of a customer.
4. Monetary i.e. The total turnover of a customer.

K- Means Clustering Algorithm.

1. To properly find out segments on our RFM values.
2. Which Customer falls under which Cluster.

6.Assumptions

The following assumptions were made in order to create the model for Online retail project.

1. All the customer with customer ID null were assumed to be New Customers and hence didn’t any ID prior.

7.Model Evaluation and Technique

RFM Segmentation

RFM is a data modeling method used to analyze customer value. It stands for recency, frequency, and monetary, which are just three metrics that describe what your customers did.

- Recency (R) of a customer: Days since the last purchase.
- Frequency (F) of the bookings/turnover of a customer: Number of purchases.
- Monetary (M) - The total turnover of a customer: Sum of sales.

For the analysis, we need to define a 'analysis date', which is the day on which we are conducting this analysis which I am taking as the next to last date in data and taking 1 year previous data from the selected date for recency calculation

```
analysis_date = uk_df["InvoiceDate"].max() + pd.DateOffset(1)
print("RFM Analysis Date :", analysis_date)

start_date = analysis_date - pd.DateOffset(days = 365)
print("Start Date when taking 1 year data for analysis :", start_date)
```

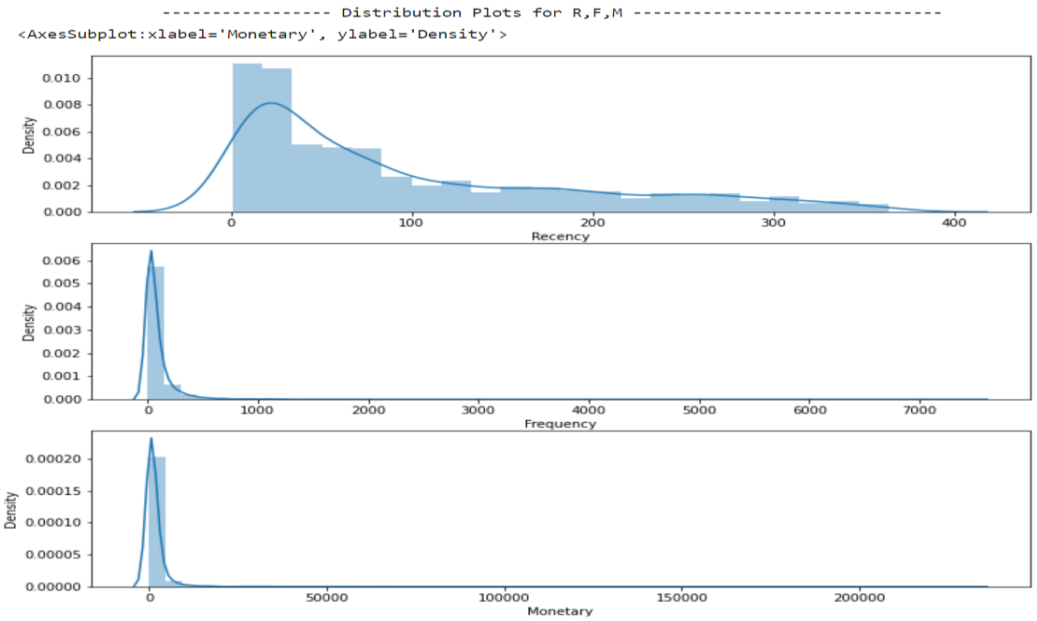
```
RFM Analysis Date : 2011-12-11 17:19:00
Start Date when taking 1 year data for analysis : 2010-12-11 17:19:00
```

The RFM_Score values will range from 0 (0+0+0) to 9 (3+3+3). Now grouping by the RFM scores to check the mean values of recency, frequency, and monetary corresponding to each score.

K - Means Clustering

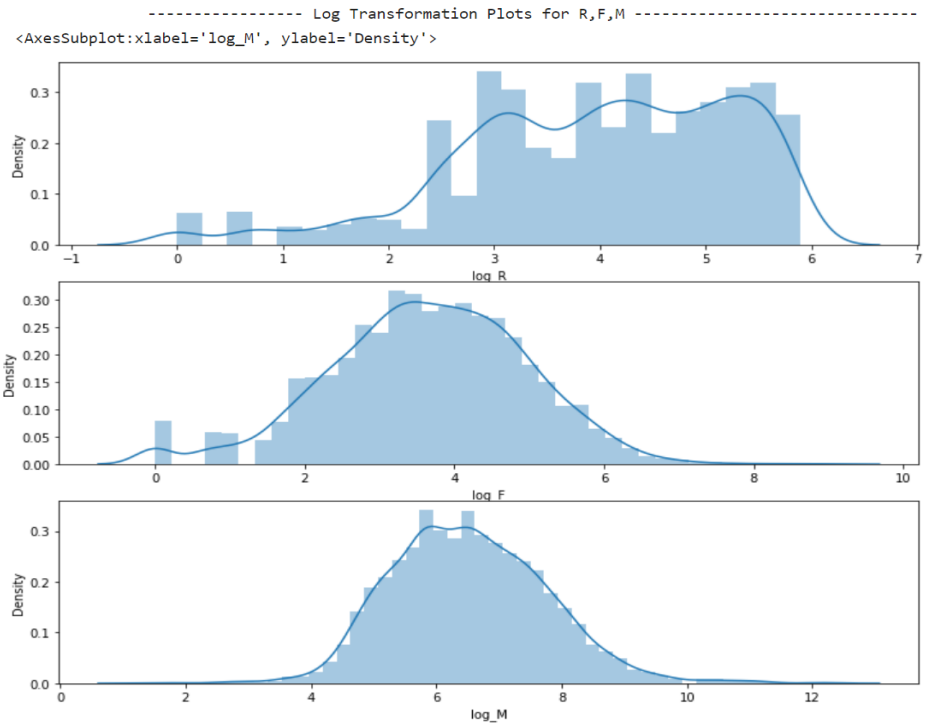
To properly find out segments on our RFM values, we can use a clustering algorithm like K-means.

```
print("               ---------------- Distribution Plots for R,F,M ------------------------------")

# Checking the distribution of Recency, Frequency and MonetaryValue variables.
plt.figure(figsize=(12,10))
# Plot distribution of var1
plt.subplot(3, 1, 1); sns.distplot(data_rfm['Recency'])
# Plot distribution of var2
plt.subplot(3, 1, 2); sns.distplot(data_rfm['Frequency'])
# Plot distribution of var3
plt.subplot(3, 1, 3); sns.distplot(data_rfm['Monetary'])
```



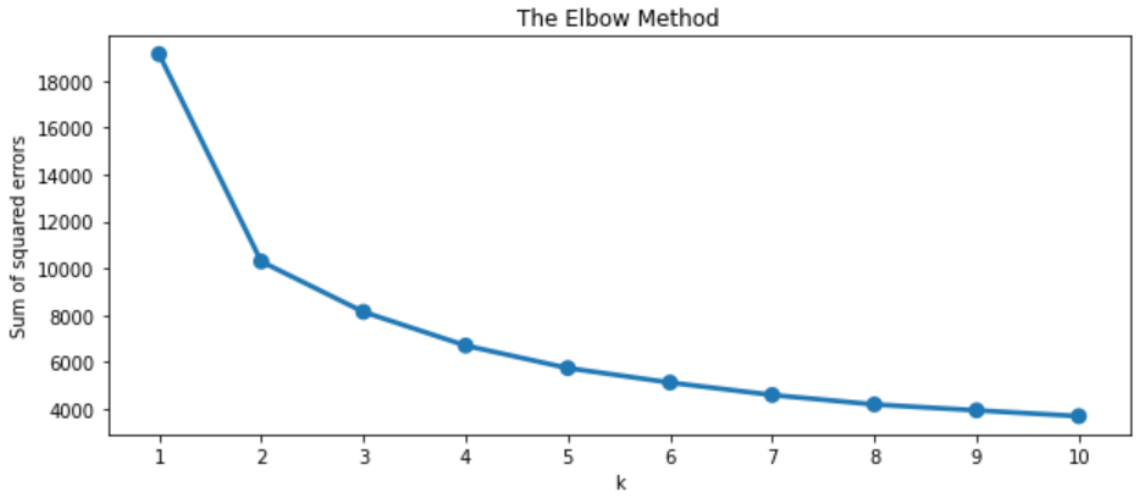
From the above figure, all the variables do not have a symmetrical distribution. All of them are skewed to the right. Since clustering algorithms require a normal distribution, normalization of the data is required.I am using Log transformation to deal with the skewness of data

```
# Taking Log of columns
data_rfm["log_R"] = np.log(data_rfm.Recency)
data_rfm["log_F"] = np.log(data_rfm.Frequency)
data_rfm["log_M"] = np.log(data_rfm.Monetary)

# Checking the distribution of Recency, Frequency and Monetary variables after Log transformation
print("                ---------- Log Transformation Plots for R,F,M ----------")
plt.figure(figsize=(12,10))
plt.subplot(3, 1, 1)
sns.distplot(data_rfm['log_R'])
plt.subplot(3, 1, 2)
sns.distplot(data_rfm['log_F'])
plt.subplot(3, 1, 3)
sns.distplot(data_rfm['log_M'])
```



To Find optimal number of clusters, I will use Elbow method where errors are plotted against K (cluster value) to identify optimal number of clusters

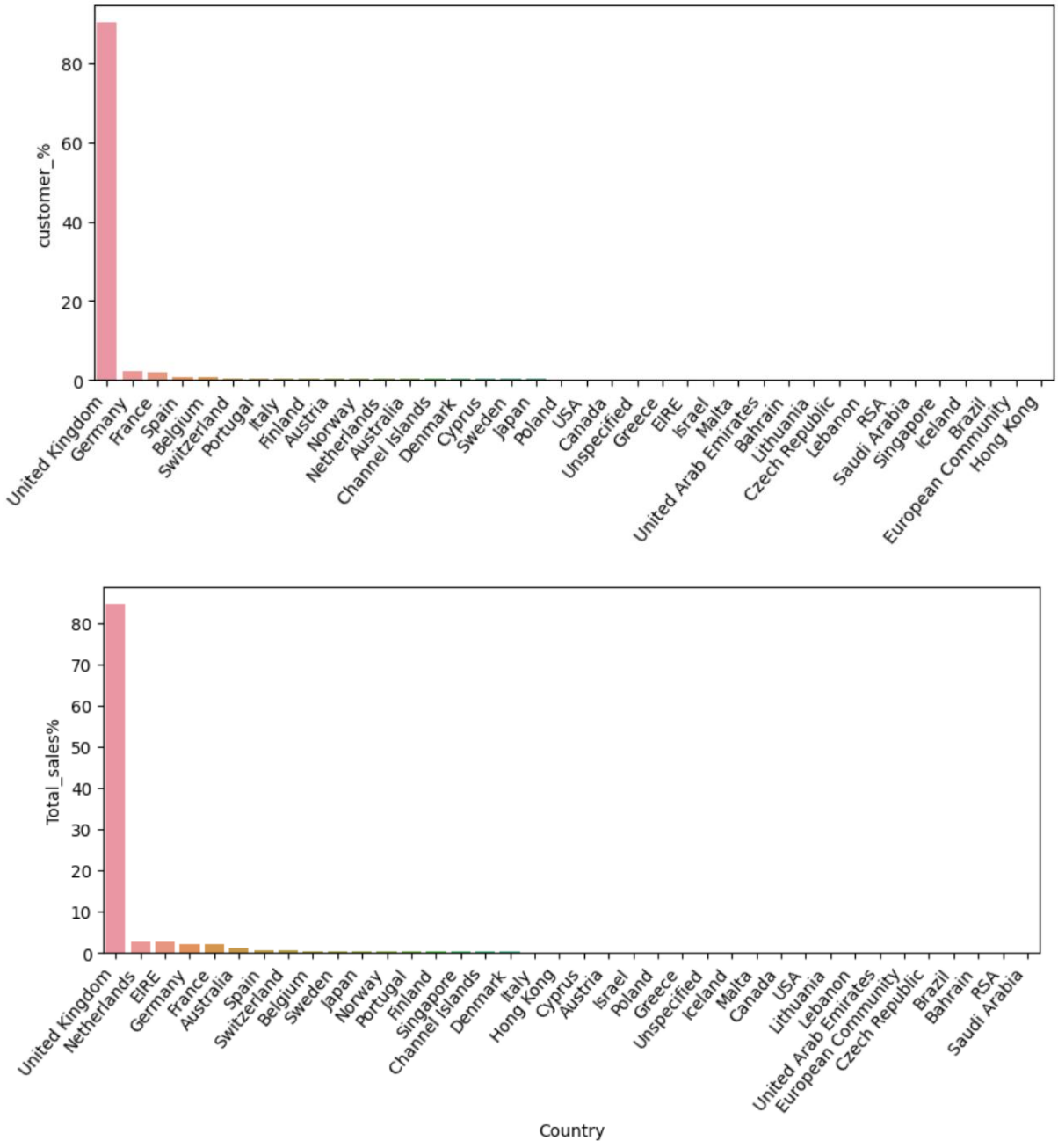
```
### Features Used in training K Means - Log Transformed Recency, Frequency and Monetary values
data_norm = data_rfm[["log_R", "log_F", "log_M"]]

sse = {}
# Fit KMeans and calculate SSE for each k
for k in range(1, 11):

    # Initialize KMeans with k clusters
    kmeans = KMeans(n_clusters=k, random_state=1)

    # Fit KMeans on the normalized dataset
    kmeans.fit(data_norm)

    # Assign sum of squared distances to k element of dictionary
    sse[k] = kmeans.inertia_

# Plotting the elbow plot
plt.figure(figsize=(10,4))
plt.title('The Elbow Method')
plt.xlabel('k');
plt.ylabel('Sum of squared errors')
sns.pointplot(x=list(sse.keys()), y=list(sse.values()))
plt.show()
```



From the above plot, we can see that the optimal number of clusters can be taken as 3. Now I am building the K Means model using 3 clusters

8.Inferences from the Project

Customer and total revenue split with respect to country



The United Kingdom not only has the most sales revenue, but also the most customers. Therefore, for the purpose of this analysis, I will be taking data corresponding to orders from the United Kingdom.

```
uk_product = uk_df.groupby(['StockCode','Description'],as_index= False)['Quantity'].sum().sort_values(by='Quantity', \
                                                                                                       ascending=False)
uk_product.head(5)
```

	StockCode	Description	Quantity
2653	23843	PAPER CRAFT , LITTLE BIRDIE	80995
2112	23166	MEDIUM CERAMIC TOP STORAGE JAR	77036
3113	84077	WORLD WAR 2 GLIDERS ASSTD DESIGNS	49430
3601	85099B	JUMBO BAG RED RETROSPOT	44161
3622	85123A	WHITE HANGING HEART T-LIGHT HOLDER	35430

	Number of products per Order
count	18019.000000
mean	26.351129
std	48.882851
min	1.000000
25%	6.000000
50%	15.000000
75%	29.000000
max	1110.000000

	Number of products per Order
count	18019.000000
mean	26.351129
std	48.882851
min	1.000000
25%	6.000000
50%	15.000000
75%	29.000000
max	1110.000000

The average number of orders per customer is 1 and average number of products per Order is 15.

```
n_clusters = 3
kmeans = KMeans(n_clusters = n_clusters, random_state=1)
kmeans.fit(data_norm)
data_rfm["cluster"] = kmeans.predict(data_norm)
```

Checking Mean RFM values in different clusters to understand cluster properties

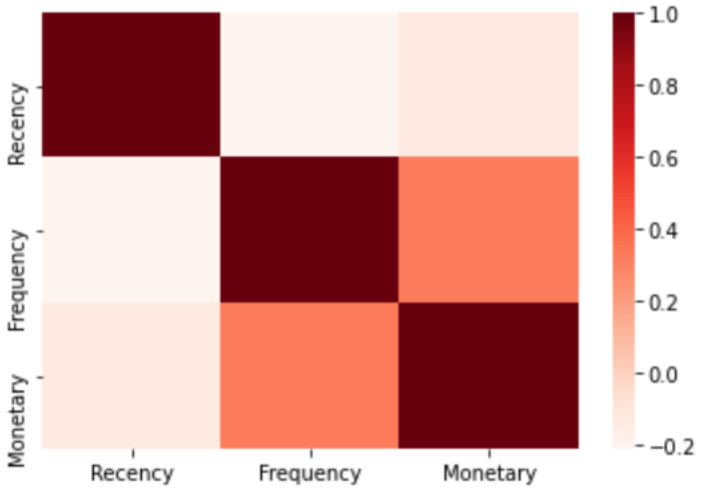
```
data_rfm.groupby(["cluster"])[['Recency','Frequency', 'Monetary']].mean()
```

cluster	Recency	Frequency	Monetary
0	89.224907	58.619579	1014.150875
1	20.415306	228.291837	5108.000082
2	168.994422	13.216733	267.958104

This is the mean value of RFM in different clusters. Clearly there is correlation between Frequency and Monetary - how their value varies between clusters. This can also be seen in the heat map of these three features shown below. Low value of recency means recent transactions while high frequency means more frequent transactions and high Monetary value means high spending in transactions

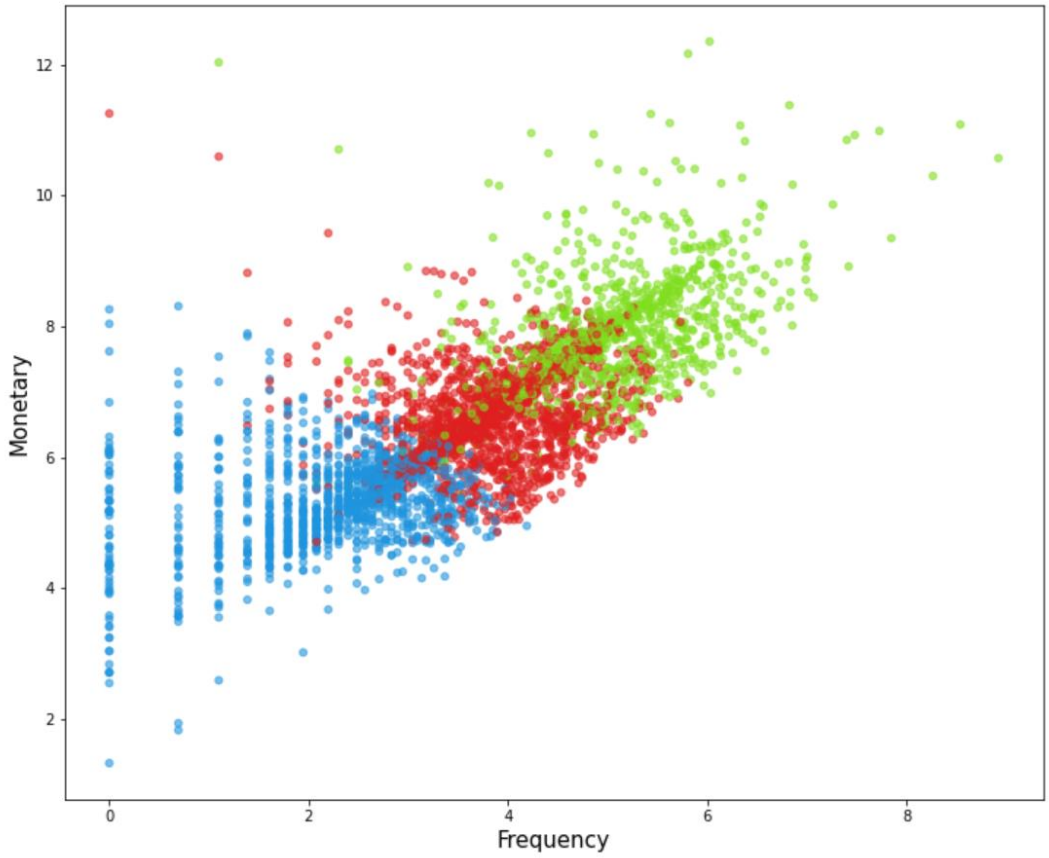
```
sns.heatmap(data_rfm[['Recency','Frequency', 'Monetary']].corr(),cmap="Reds")
```

```
<AxesSubplot:>
```



There is a negative correlation between Recency - Frequency and Recency - Monetary, but there is a positive correlation between Frequency - Monetary which can also be seen in the variation of mean

values in clusters



This plot clearly indicates that green cluster is high value customer cohort (Cluster 1) and blue cluster is Lost/low value customer cohort (Cluster 2) while red cluster is average value customer cohort (Cluster 0)

- High Value :- Frequent spending customers with high monetary transactions and had transacted recently
- Low Value/ Lost :- Customers with least frequency and monetary spending and made purchase long time ago. These customers might be lost from the customer base.
- Average Value :- Customers who made their transactions some time ago with less frequency and monetary value. These customers could become high risk and we could aggressively market towards them with great deals so we don't lose them forever

9.Future Possibilities

customers with the lowest RFM scores have the highest recency value and the lowest frequency and monetary value, and the vice-versa is true as well. This can be used to create manual segments based on RFM scores like Loyal Customers Group where Frequancy is High, High Spending Group where Monetary is High, Lost Customers Group where Recency is High etc.

Loyal and Good Customers which have High RFM values could be rewarded and heavy discounts are not needed for them. Customers with High Recency (and High Frequency and Monetary values) could be targeted aggresively with discounts so that they are not lost

10.Conclusion

The project shows us that:

1.The United Kingdom not only has the most sales revenue, but also the most customers.

2.Customer purchased Stock Code 23843 which is Paper Craft, Little Birdie the most with 80995 in quantity.

3. The average number of orders per customer is 1 and average number of products per Order is 15

4. There is a negative correlation between Recency - Frequency and Recency - Monetary, but there is a positive correlation between Frequency - Monetary.