SALES DATA ANALYSIS USING RFM

Submitted to : Tidyquant

Submitted By : Ayush Anand

RFM Analysis

MONETARY VALUE

FREQUENCY

RECENCY

Loading the Data

- The dataset was loaded and viewed using pandas and was stored in a dataframe called “Sales”.
- There we totale 5000 rows and 40 columns with a primary column named “CustomerID”.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

import sklearn
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score

from scipy.cluster.hierarchy import linkage
from scipy.cluster.hierarchy import dendrogram
from scipy.cluster.hierarchy import cut_tree
```

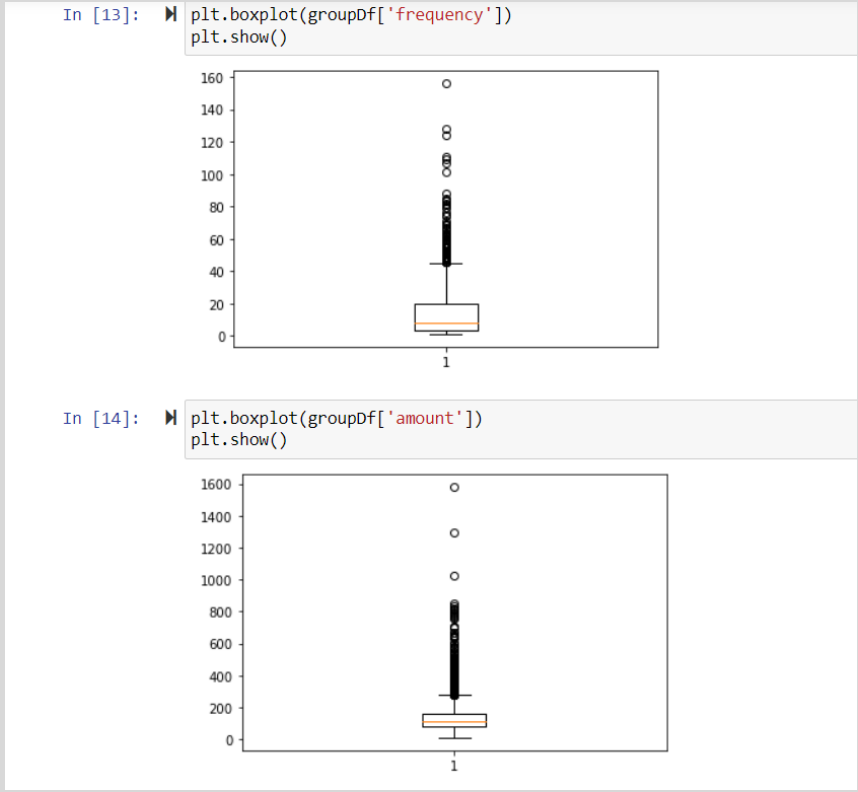
In [2]:

```
#Loading the data into sales as pandas dataframe
sales = pd.read_excel("sales_data.xlsx")
sales.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 40 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   CustomerID                5000 non-null   int64
 1   TOTAL_ORDERS              5000 non-null   int64
 2   REVENUE                   5000 non-null   float64
 3   AVERAGE_ORDER_VALUE       5000 non-null   float64
 4   CARRIAGE_REVENUE          5000 non-null   float64
 5   AVERAGESHIPPING           5000 non-null   float64
 6   FIRST_ORDER_DATE          5000 non-null   datetime64[ns]
 7   LATEST_ORDER_DATE         5000 non-null   datetime64[ns]
 8   AVGDAYSBETWEENORDERS      5000 non-null   float64
 9   DAYSSINCELASTORDER        5000 non-null   int64
```

Exploratory Data Analysis

- The dataset had no null values or missing values or false values.
- Then checked for Outliers and treated the Outliers.
- The data set was divided into "groupDf" Dataframe for RFM analysis.
- The outliers were found to be in "frequency" and "amount" columns.
- After removing outliers there were 4973 columns left.



Used K-Means Algorithm for Clustering

- Used K-Means Algorithm for clustering
- Checked if the data was clusterable using Hopkins test.
- The number was found to be 0.92, which was pretty close to 1. Thus the data was clusterable.
- Found the optimal number of clusters using the Elbow Curve

Using K-Means Algorithm

In [18]:
```
# Using an arbitrary number (n_cluster = 4)
kmeans = KMeans(n_clusters=4, max_iter=50)
kmeans.fit(rfmDfScaled)
```

Out[18]:
```
KMeans(max_iter=50, n_clusters=4)
```

In [19]:
```
kmeans.labels_
```

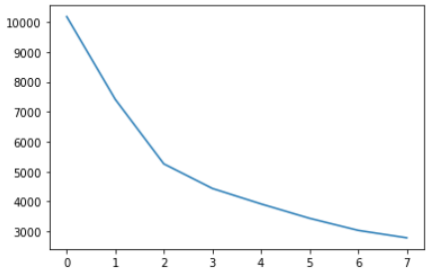
Out[19]:
```
array([3, 3, 3, ..., 1, 1, 0])
```

In [20]:
```
# sum of squared distance (ssd)
ssd=[]
range_n_clusters= [2,3,4,5,6,7,8,9]
for num_clusters in range_n_clusters:
    kmeans = KMeans(n_clusters = num_clusters, max_iter=50)
    kmeans.fit(rfmDfScaled)

    ssd.append(kmeans.inertia_)

plt.plot(ssd)
```

Out[20]:
```
[<matplotlib.lines.Line2D at 0x255bc4223a0>]
```



Silhouette's Analysis

- Using, the number of clusters being 4, plotted the box plots for each of the parameters

Silhouette's Analysis

From th elbow curve we found that the X-point (2) ,i.e, 4 clusters are optimal for the data. Thus using 4 as the optimal number.

In [23]:
```
kmeans = KMeans(n_clusters = 4,max_iter=50)
kmeans.fit(rfmDfScaled)
```

Out[23]:
```
KMeans(max_iter=50, n_clusters=4)
```

In [24]:
```
kmeans.labels_
```

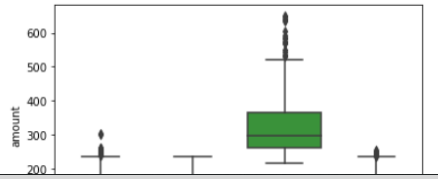
Out[24]:
```
array([2, 2, 2, ..., 3, 3, 1])
```

In [25]:
```
# Assigning the cluster label to the groupDf
groupDf['clusterID'] = kmeans.labels_
groupDf.head()
```

Out[25]:

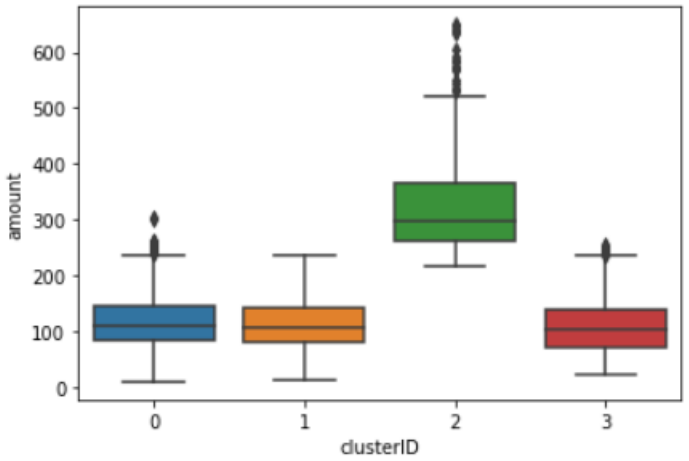
	CustomerID	amount	frequency	recency	clusterID
0	1	571.27	61	53	2
1	2	550.63	59	94	2
2	3	456.21	53	53	2
3	4	220.89	84	5	0
4	5	649.42	26	130	2

In [26]:
```
#plotting
sns.boxplot(x='clusterID',y='amount',data=groupDf)
plt.show()
```

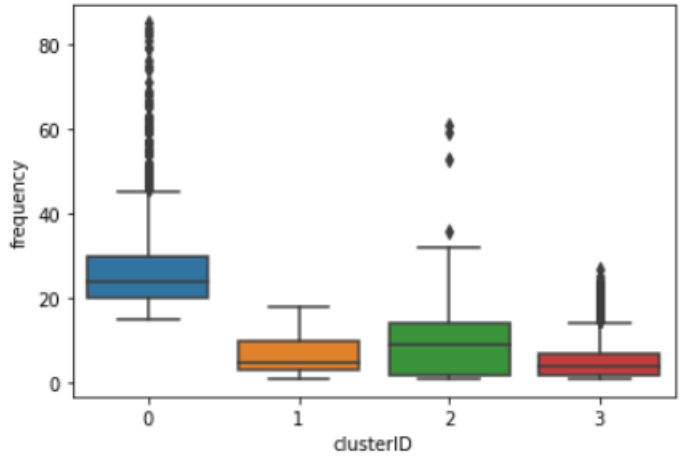


Box Plots

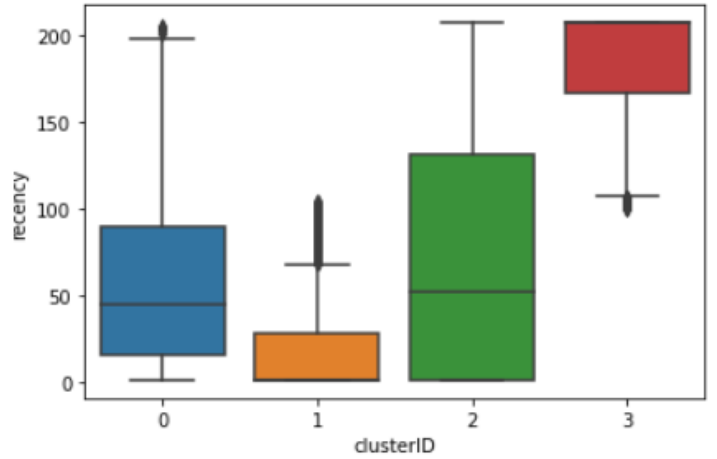
In [26]:
```
#plotting
sns.boxplot(x='clusterID',y='amount',data=groupDf)
plt.show()
```



In [27]:
```
sns.boxplot(x='clusterID',y='frequency',data=groupDf)
plt.show()
```

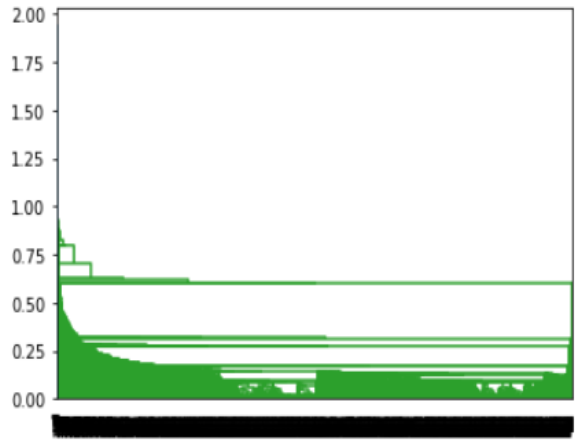


In [28]:
```
sns.boxplot(x='clusterID',y='recency',data=groupDf)
plt.show()
```

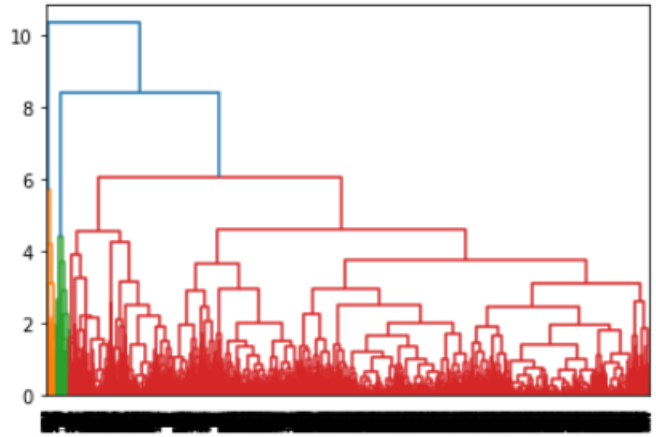


Hierarchical Clustering and Dendrogram

In [32]:
```
# performing the Single linkage
mergings = linkage(rfmDfScaled , method = 'single' , metric = 'euclidean')
dendrogram(mergings)
plt.show()
```

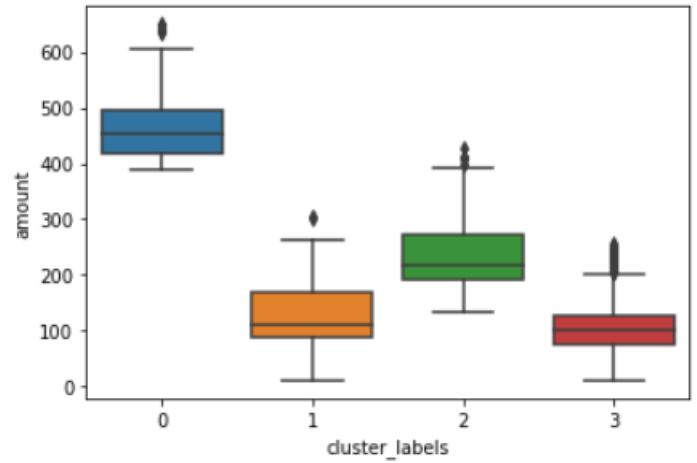


In [33]:
```
# Complete Linkage
mergings = linkage(rfmDfScaled, method = 'complete', metric = 'euclidean')
dendrogram(mergings)
plt.show()
```

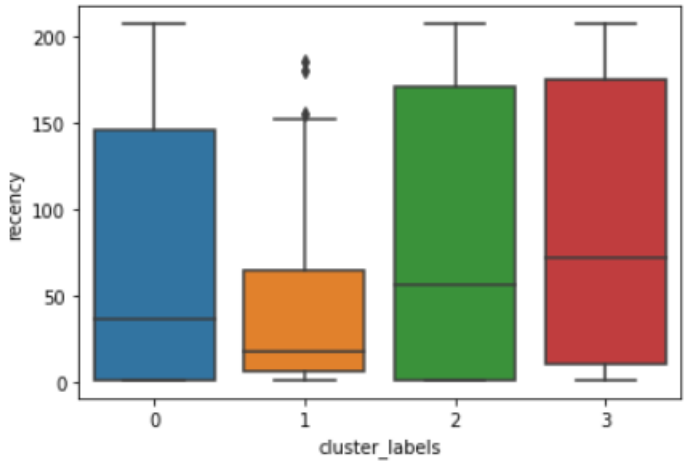


The Final 3 Boxplots

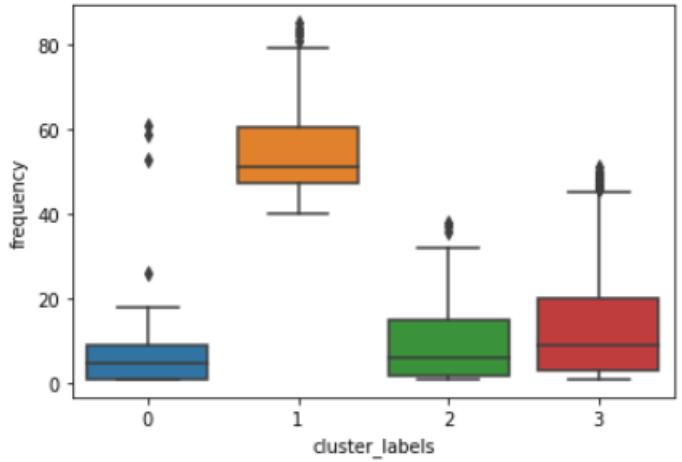
In [38]:
```
sns.boxplot(x = 'cluster_labels', y = 'amount', data = groupDf)
plt.show()
```



In [36]:
```
sns.boxplot(x = 'cluster_labels', y = 'recency', data = groupDf)
plt.show()
```



In [37]:
```
sns.boxplot(x = 'cluster_labels', y = 'frequency', data = groupDf)
plt.show()
```



Conclusion

As we can see from the above three boxplots, the Cluster 4 (cluster_label = 3) has made the most recent order and it also makes the orders more frequently as compared to others.

However when we compare the average amount per order it turns out to be the least for the customers in the cluster 4. We can also consider cluster 3 (cluster_label = 2) for the most recent order after cluster 4 and cluster 3 has made the most frequent orders after cluster 4 however when we compare the average amount per order cluster 3 (cluster_labels = 2) scores a second position again. So we should be focusing on Cluster 3 (cluster_labels = 2) if we want to have a good amount of sales and profits.

We could also see that cluster 1 (cluster_labels = 0) has made the purchase for highest amounts but have made the order less frequently. We could ask our marketing team to roll out more attention grabbing advertisements for those particular set of customers in order to have more revenue as they have also made orders recently which means Cluster 1 customers are loyal too.

Thank You!