CAPSTONE PROJECT
(Unsupervised Machine Learning)
CUSTOMER SEGMENTATION

Created by -
Aditya Chandraprakash Tadas

STEPS IN PROJECT

- **INTRODUCTION**
- **DATA CLEANING**
- **EXPLORATORY DATA ANALYSIS**
- **FEATURE ENIGINEERING**
- **CLUSTERING BY RFM SCORING**
- **ELBOW METHOD**
- **K-MEANS CLUSTERING**
- **SILHOUETTE ANALYSIS**
- **DENDOGRAM**
- **PRINCIPAL COMPONENT ANALYSIS**

INTRODUCTION

- Businesses all over the world are growing every day. With the help of technology, they have access to a wider market and hence, a large customer base.
- Customer segmentation refers to categorizing customers into different groups with similar characteristics.
- Customer segmentation can help businesses focus on each customer group in a different way, in order to maximize benefits for customers as well as the business
- This project mainly deals in segmenting customers of an online business store in the UK.

DATA

- The data being used here is a transnational data of an online store based in the UK, which mainly sells unique all-occasion gifts.
- The data has 5,41,908 rows and 8 columns:

➢ Invoice No
: Number uniquely assigned to each transaction. If this code starts with letter 'c', it indicates a cancellation

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

➢ Description: Product (item) name. Nominal.

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

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

➢ Unit Price: Unit price. Numeric, Product price per unit in sterling.

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

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

DATA SUMMARY

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	01-12-2010 08:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	01-12-2010 08:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	01-12-2010 08:26	3.39	17850.0	United Kingdom

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

```
# Dropping the null values of the description column
data.dropna(subset = ['Description'], inplace = True)
```

```
# Dropping the rows which contain null values in the Customer ID column
data.dropna(subset=['CustomerID'], axis = 0, inplace = True)
```

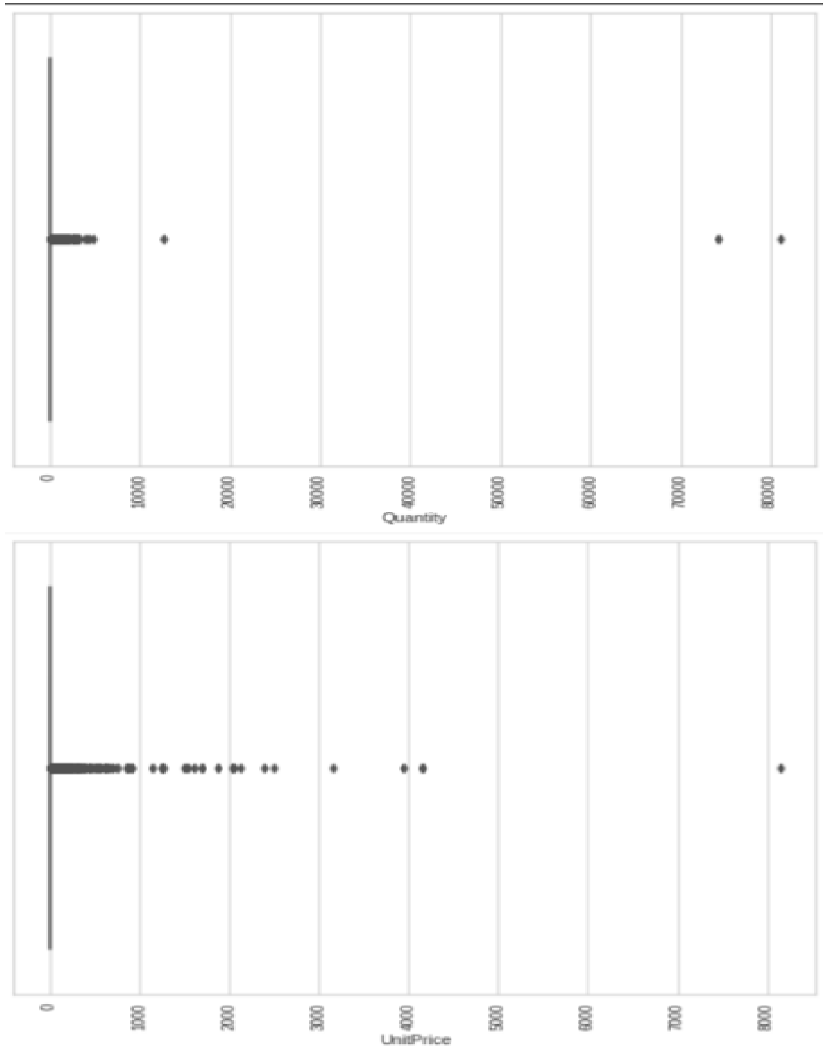
```
# Dropping the cancelled orders
data.drop(data[data['Cancelled'] == 'YES'].index, inplace=True)
```

Before handling null values data contains 541909 but after dropping null values present in 2 columns which are description and customer id now data contains 406829 r ecords.
Unfortunately 135080 Records have been lost in the process

After dropping cancelled order unfortunately 8905 records are lost in this process.

```
data.shape
(397924, 8)
```

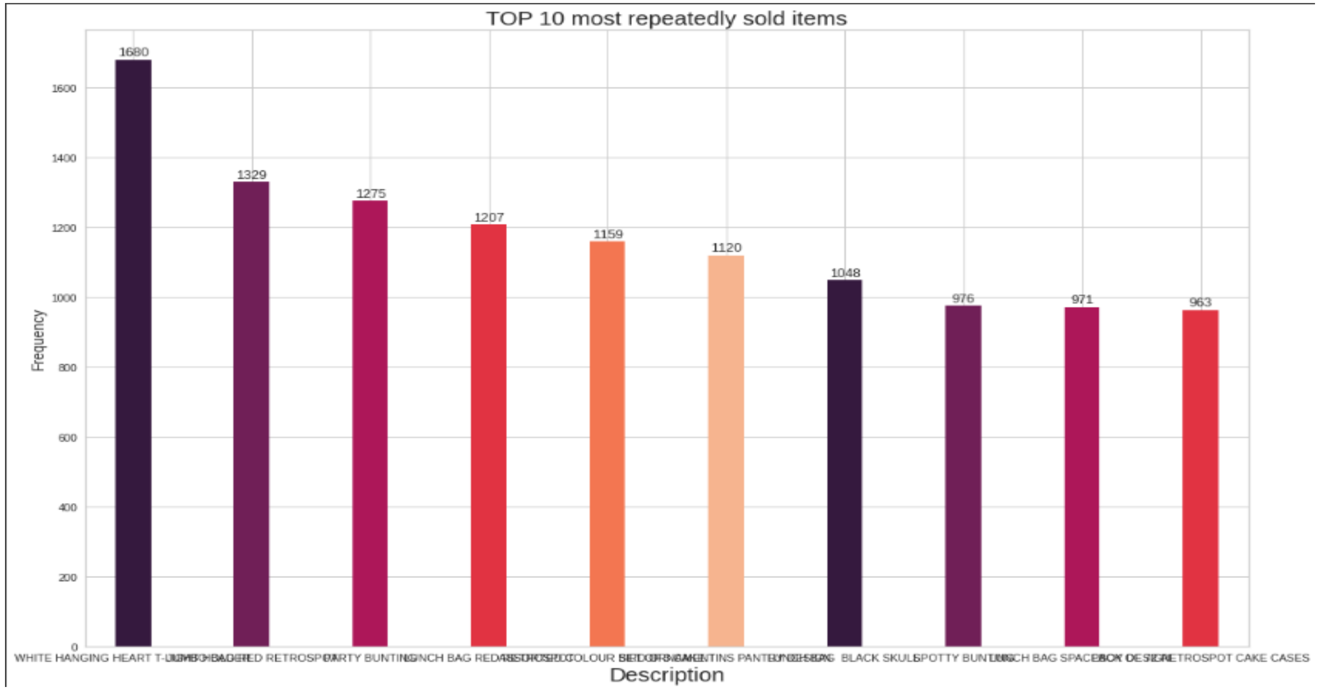
REMOVING OUTLIER

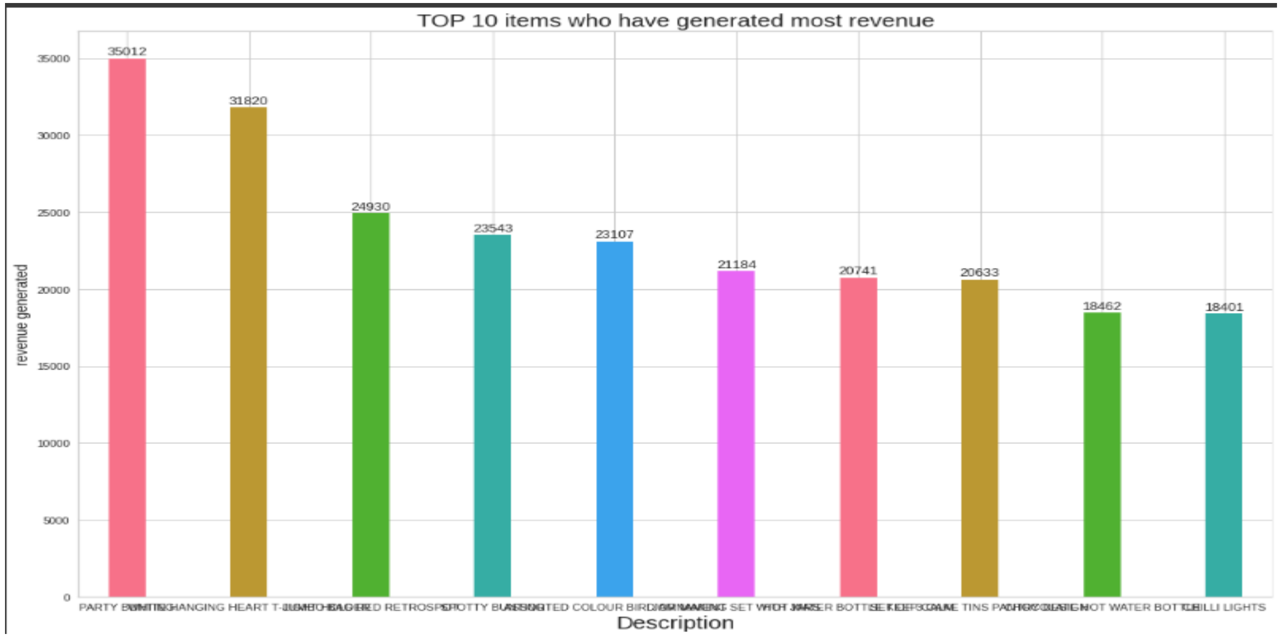



```python
# Creating a function to remove outliers
def remove_outliers(df , column):
    '''Removes outliers in given the dataframe and column'''
    q3 = df[column].quantile(0.75)
    q1 = df[column].quantile(0.25)
    iqr = q3 - q1
    upper_limit = q3 + (1.5 * iqr)
    lower_limit = q1 - (1.5 * iqr)

    if lower_limit < 0:
      df = df[df[column] <= upper_limit]
    else:
      df = df[(df[column] >= lower_limit) & (df[column] <= upper_limit)]

    return df
```

```python
# Removing the outliers using the function created
data = remove_outliers(df = data, column = 'Quantity')
data = remove_outliers(df = data, column = 'UnitPrice')
```

```python
# Using the Invoice date column to extract
data['InvoiceDate']=data['InvoiceDate'].apply(pd.to_datetime)
data['Day'] = data['InvoiceDate'].dt.day
data['Month'] = data['InvoiceDate'].dt.month
data['Year'] = data['InvoiceDate'].dt.year
data['day_name'] = data['InvoiceDate'].dt.day_name()
data['Quarter']=data['InvoiceDate'].dt.quarter
data['hour']=data['InvoiceDate'].dt.hour
data['week'] = data['InvoiceDate'].dt.week
```

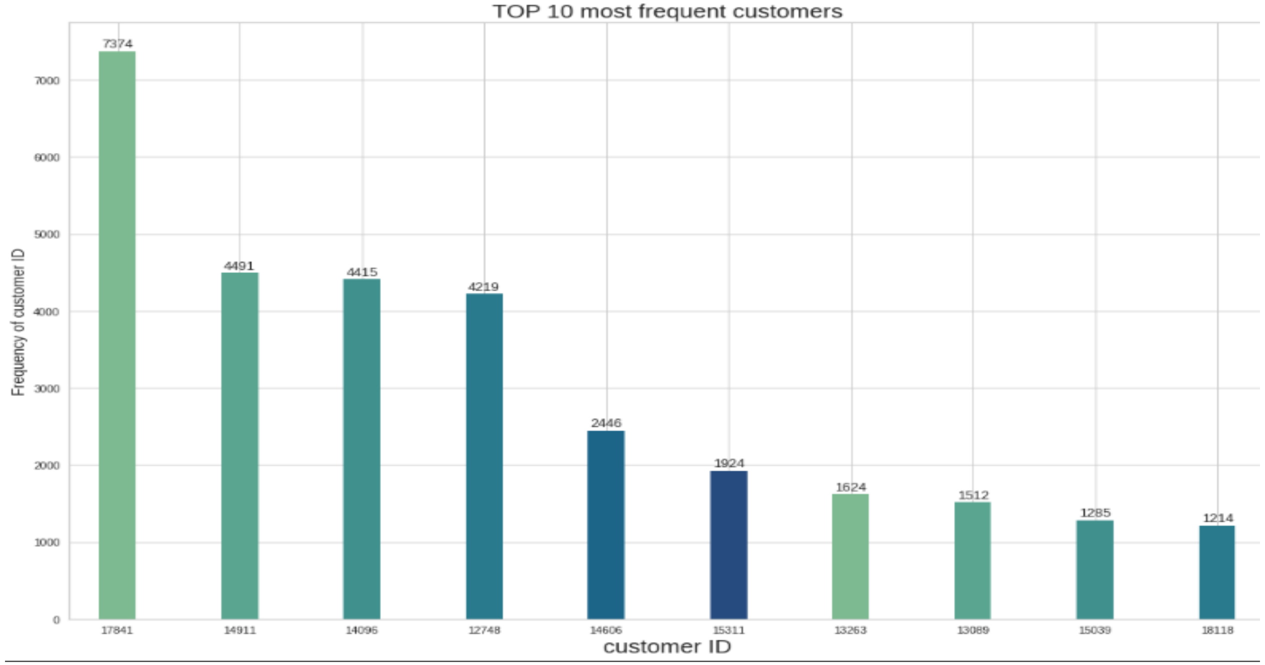


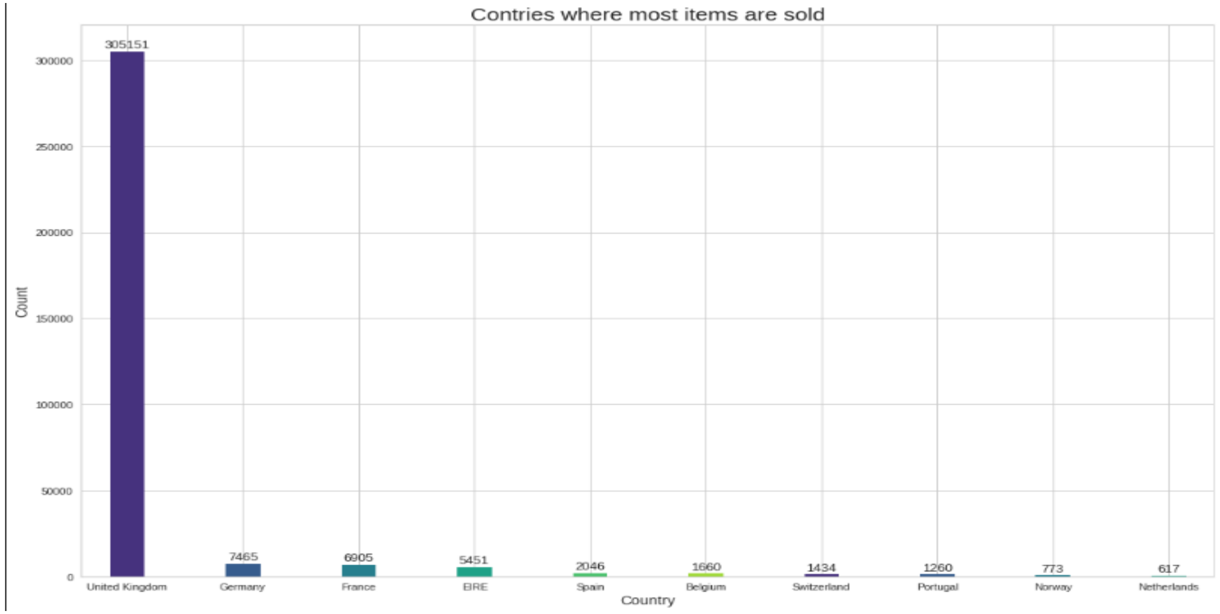


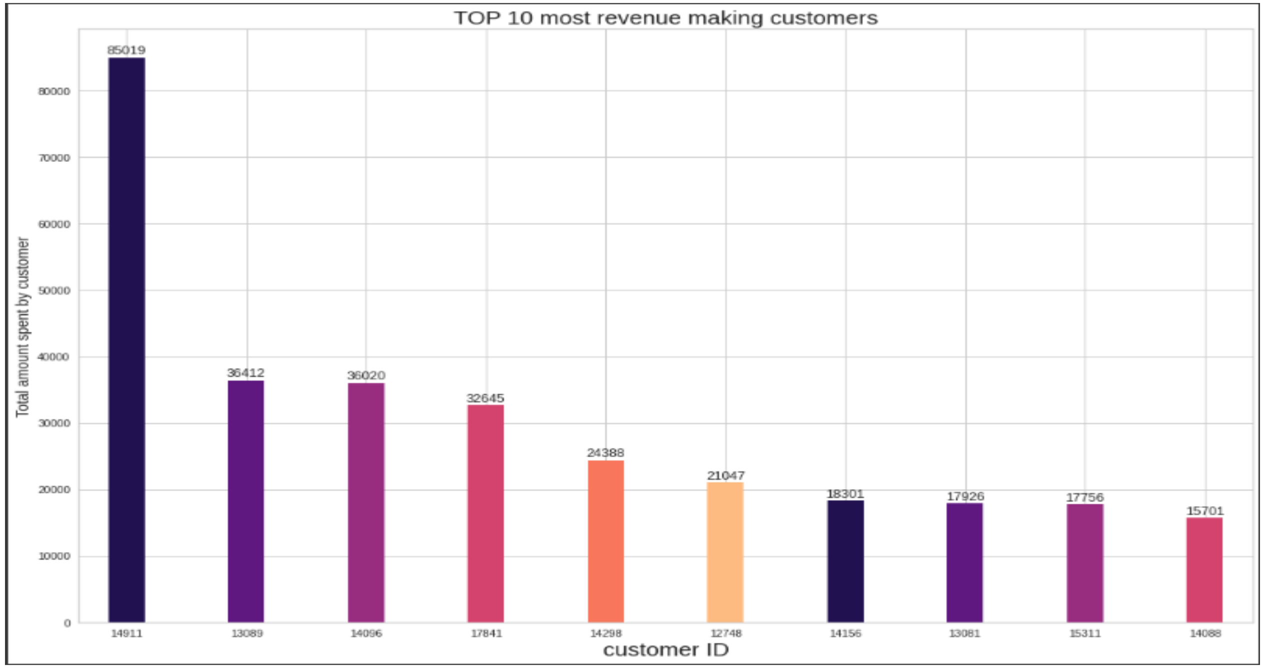
- From the bar
  plot of top 10 most frequently sold items it is clearly seen that WHITE HANGING HEART T-LIGHT HOLDER is the most reapetedly sold items.
- Hence company should generate stratergies to increase the supply of all those 10 most frequently sold items.

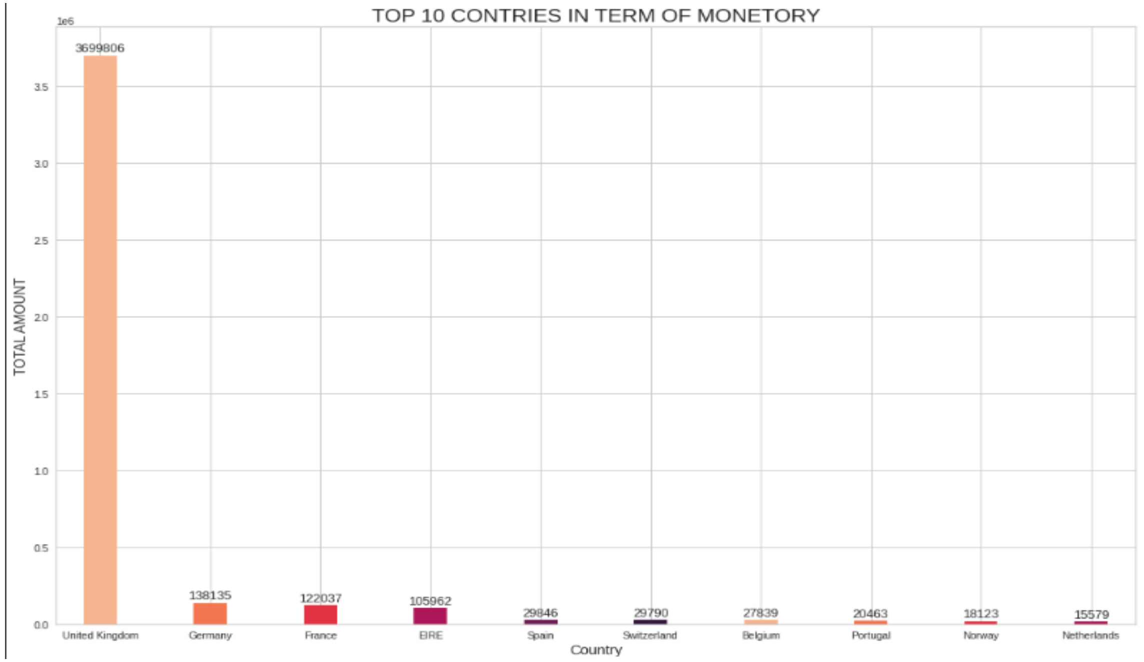
- From the bar
  plot of Top 10 most revenue generated items we can clearly seen the PAPER CRAFT , LITTLE BIRDIE is the item which have generated most revenue for company.
- Hence company needs to work on that product and improve quality and supply further more to generate more revenue.

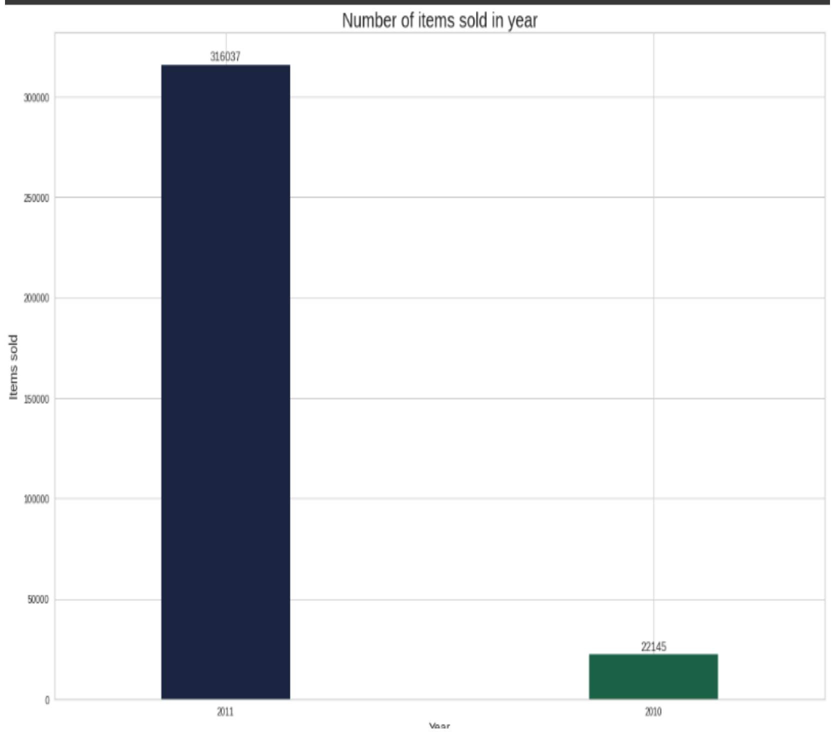
- But one important point in that the products called REGENCY CAKESTAND 3 TIER , WHITE HANGING HEART T-LIGHT HOLDER , JUMBO BAG RED RETROSPOT these are the top 3 products who sold more frequently and also comes under the top 5 product who generated most revenue for company so company need to maintain good suppLy and improve quality of product so the company make more profit by selling those 3 products because these 3 products are most important products for the company.

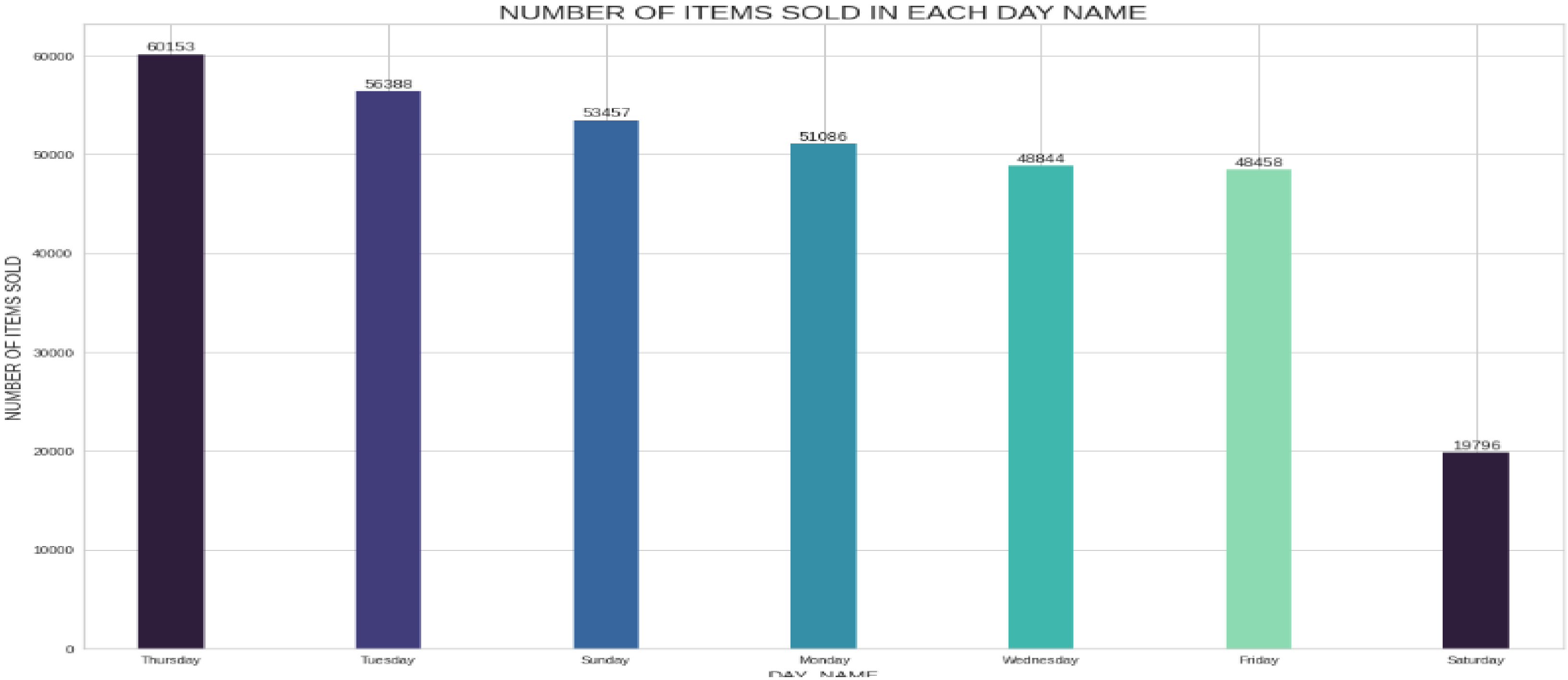

TOP 10 most frequent customers
Frequency of customer ID
customer ID
7374
4491
4415
4219
2446
1924
1624
1512
1285
1214
17841
14911
14096
12748
14606
15311
13263
13089
15039
18118

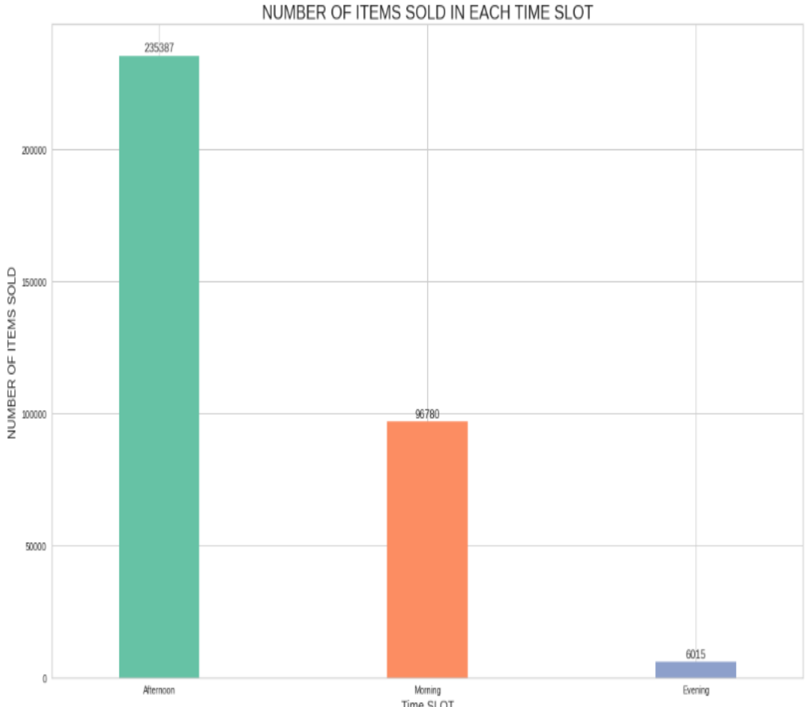

Contries where most items are sold
Count
Country
305151
7465
6905
5451
2046
1660
1434
1260
773
617
United Kingdom
Germany
France
EIRE
Spain
Belgium
Switzerland
Portugal
Norway
Netherlands

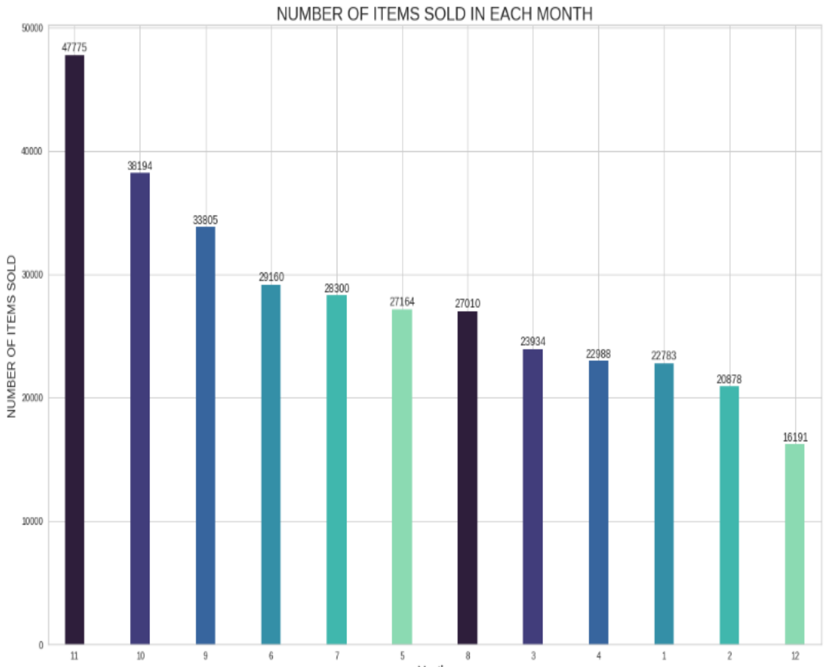

TOP 10 most revenue making customers
Total amount spent by customer
customer ID
85019
36412
36020
32645
24388
21047
18301
17926
17756
15701
14911
13089
14096
17841
14298
12748
14156
13081
15311
14088

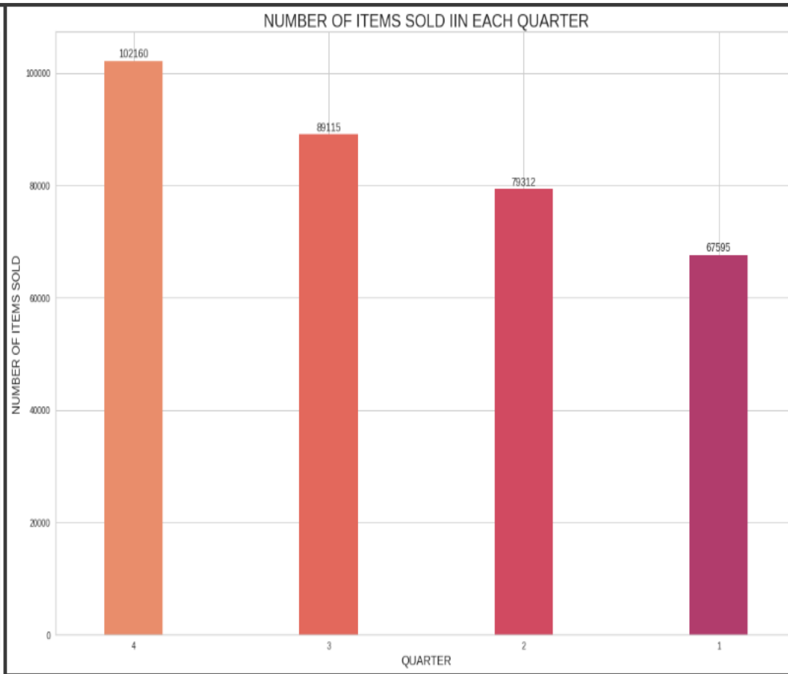

TOP 10 CONTRIES IN TERM OF MONETORY
TOTAL AMOUNT
Country
1e6
3699806
138135
122037
105962
29846
29790
27839
20463
18123
15579
United Kingdom
Germany
France
EIRE
Spain
Switzerland
Belgium
Portugal
Norway
Netherlands

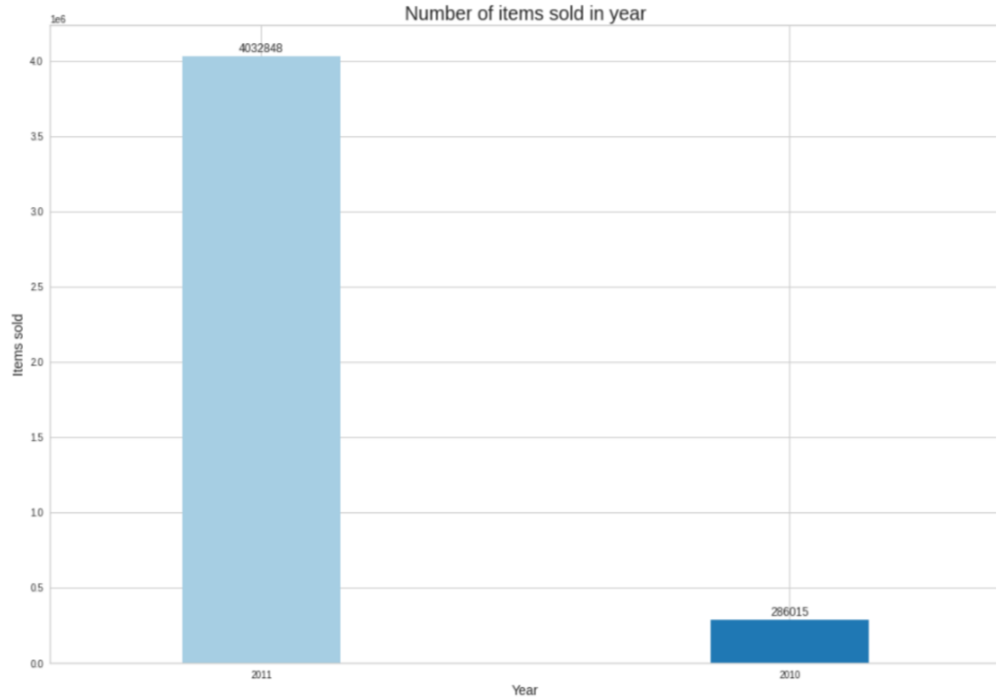

Number of items sold in year
Items sold
316037
22145
2011
2010


NUMBER OF ITEMS SOLD IN EACH DAY NAME
NUMBER OF ITEMS SOLD
60153
56388
53457
51086
48844
48458
19796
Thursday
Tuesday
Sunday
Monday
Wednesday
Friday
Saturday

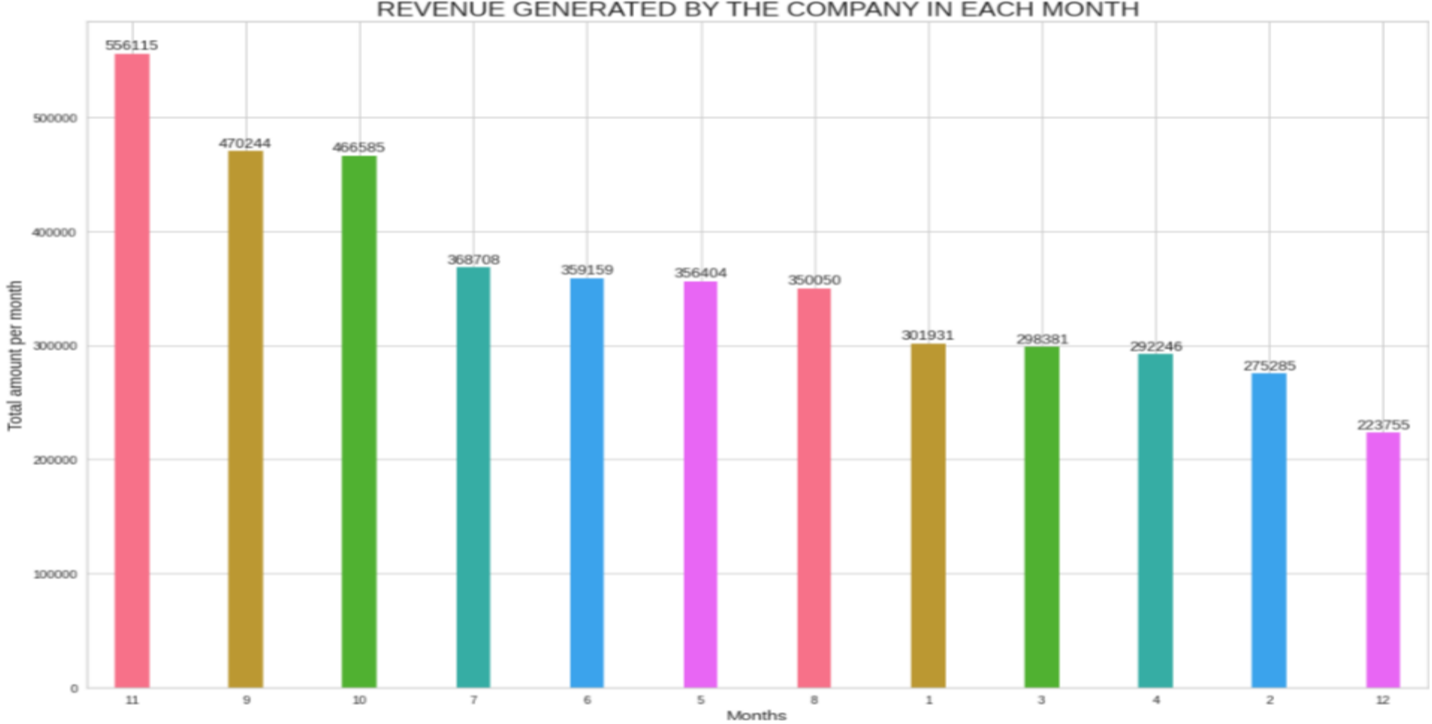

NUMBER OF ITEMS SOLD IN EACH TIME SLOT
NUMBER OF ITEMS SOLD
235387
96780
6015
Afternoon
Morning
Evening
Time SLOT


NUMBER OF ITEMS SOLD IN EACH MONTH
NUMBER OF ITEMS SOLD
47775
38194
33805
29160
28300
27164
27010
23934
22988
22783
20878
16191
11
10
9
6
7
5
8
3
4
1
2
12


NUMBER OF ITEMS SOLD IIN EACH QUARTER
NUMBER OF ITEMS SOLD
102160
89115
79312
67595
4
3
2
1
QUARTER

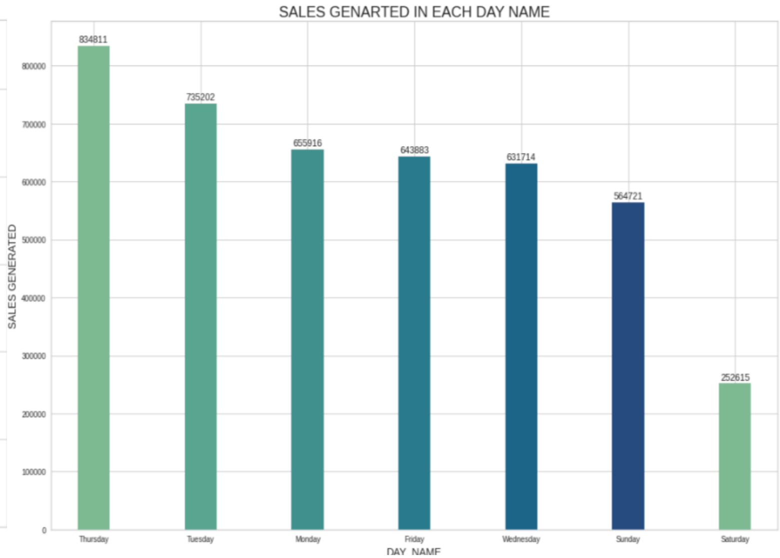


Number of items sold in year
1e6
Items sold
4032848
286015
2011
2010
Year

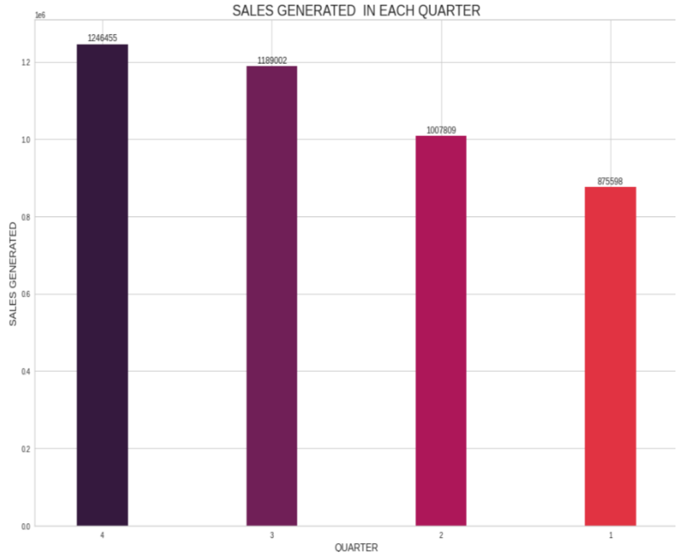


REVENUE GENERATED BY THE COMPANY IN EACH MONTH
Total amount per month
556115
470244
466585
368708
359159
356404
350050
301931
298381
292246
275285
223755
11
9
10
7
6
5
8
1
3
4
2
12
Months

REVENUE GENERATED IN EACH TIME SLOT
1e6
Total amount
2756409
1500691
61762
Afternoon
Morning
Evening
Time SLOT



SALES GENARTED IN EACH DAY NAME
SALES GENERATED
834811
735202
655916
643883
631714
564721
252615
Thursday
Tuesday
Monday
Friday
Wednesday
Sunday
Saturday
DAY_NAME



SALES GENERATED IN EACH QUARTER
1e6
SALES GENERATED
1246455
1189002
1007809
875598
4
3
2
1
QUARTER

Co-relation Plot

From the above co-
relation plot we can see that most of the fea
tures are highly co-
relation but we require only few features to
cluster our customers.

so we can ignore this co-relation

Develop Recency , Frequency and Monetary Data Frame

```
# Creating a dataframe to find the most recent purchase
recency_df = pd.DataFrame(data.groupby('CustomerID').max()['InvoiceDate'], columns = ['InvoiceDate'])
recency_df.reset_index(inplace = True)

# Calculating days from most recent purchase
recency_df['Recency'] = recency_df['InvoiceDate'].apply(lambda x: (latest_Date - x).days)
recency_df = recency_df.loc[: , ['CustomerID', 'Recency']]
recency_df.head()
```

	CustomerID	Recency
0	12347	40
1	12348	220
2	12349	19
3	12350	311
4	12352	73

```
# Creating a frequency dataframe
freq_df = pd.DataFrame(data = data.groupby('CustomerID').nunique()['InvoiceNo'])
freq_df.reset_index(inplace = True)
freq_df.columns = ['CustomerID', 'Frequency']
freq_df.head()
```

	CustomerID	Frequency
0	12347	7
1	12348	3
2	12349	1
3	12350	1
4	12352	7

```
# Grouping by customer ID to find total billed amount per customer
monetary_df = pd.DataFrame(data.groupby('CustomerID').sum()['TotalAmount'])
monetary_df.reset_index(inplace = True)
monetary_df.columns = ['CustomerID', 'Monetary']
monetary_df.head()
```

	CustomerID	Monetary
0	12347	3314.73
1	12348	90.20
2	12349	999.15
3	12350	294.40
4	12352	1130.94

CREATING FUNCTION TO CATEGORISE CUSTOMERS

```
#Functions to create R , F ,M segments
def RScoring(x,p,d):
    if x <= d[p][0.25] :
        return 1
    elif x <= d[p][0.50] :
        return 2
    elif x <= d[p][0.75] :
        return 3
    else :
        return 4
def FnMScoring(x,p,d):
    if x <= d[p][0.25] :
        return 4
    elif x <= d[p][0.50]:
        return 3
    elif x <= d[p][0.75]:
        return 2
    else:
        return 1
```

Extracting new variables from recency , freaquency and monetory

```
rfm_df['R']=rfm_df['Recency'].apply(RScoring , args = ('Recency',quantiles,))
rfm_df['F']=rfm_df['Frequency'].apply(FnMScoring , args = ('Frequency',quantiles,))
rfm_df['M']=rfm_df['Monetary'].apply(FnMScoring , args = ('Monetary',quantiles,))
rfm_df.head()
```

	CustomerID	Recency	Frequency	Monetary	R	F	M
0	12347	40	7	3314.73	2	1	1
1	12348	220	3	90.20	4	2	4
2	12349	19	1	999.15	1	4	2
3	12350	311	1	294.40	4	4	3
4	12352	73	7	1130.94	3	1	2

	CustomerID	Recency	Frequency	Monetary	R	F	M	RFMScore
count	4192.000000	4192.000000	4192.000000	4192.000000	4192.000000	4192.000000	4192.000000	4192.000000
mean	15290.259065	105.616651	4.015983	1030.263007	2.487834	2.660782	2.500000	7.648616
std	1719.353408	114.120616	7.022919	2205.355349	1.121510	1.192880	1.118167	2.911787
min	12347.000000	0.000000	1.000000	0.000000	1.000000	1.000000	1.000000	3.000000
25%	13808.750000	22.000000	1.000000	207.850000	1.000000	2.000000	1.750000	5.000000
50%	15280.500000	61.000000	2.000000	468.665000	2.000000	3.000000	2.500000	8.000000
75%	16770.250000	162.000000	4.000000	1136.625000	3.000000	4.000000	3.250000	10.000000
max	18287.000000	697.000000	197.000000	85018.780000	4.000000	4.000000	4.000000	12.000000

CUSTOMER SEGMENTATION USING RFM SCORE

Diamond Class-

The customers which have a RFM score greater than equal to 10 and till the end(in these scenario maximum RFM score is 12) those customers are belong to Diamond Category.
Those customers are the most important customers for the company with respect to sales.

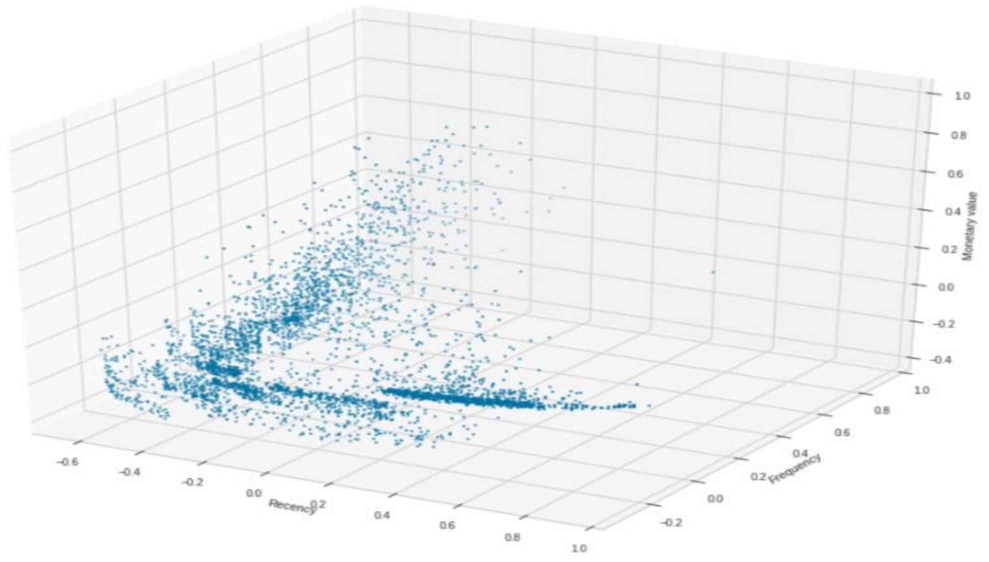
Platinum Class-

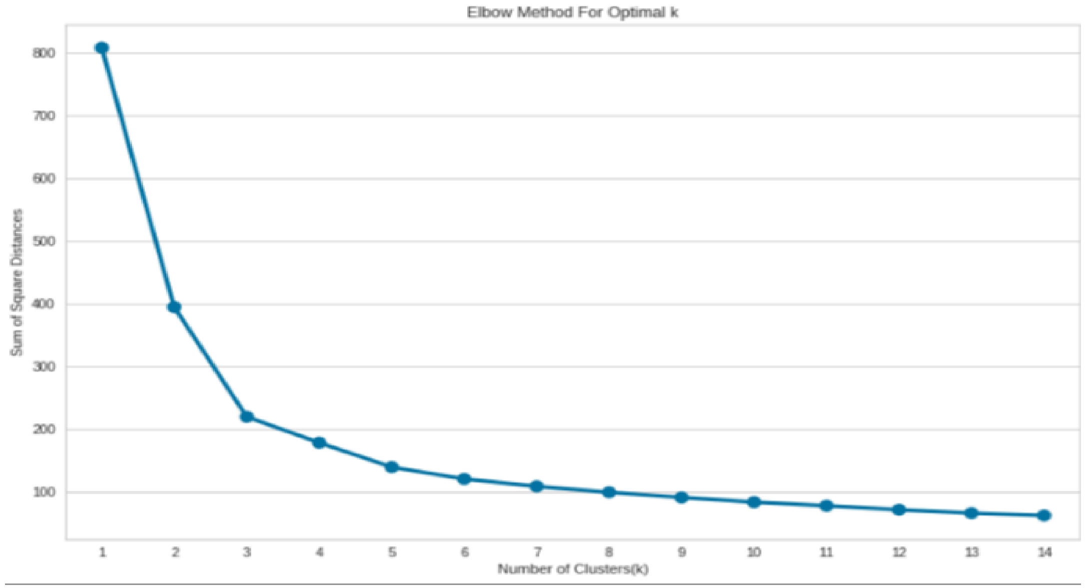
The customers whose RFM score is equal to 10 those customers are belong to platinum Category.
Those customer are second most important customers to generate sales for the company.
Company should target those customers to expensive products and medium expensive products.
also company should maintain good relationship with these customers category stay in contact with these customers regularly.
There is a possibility that these customers are retailers who bought a items in huge amount so these customer categories are important categories to generate sales for the company.

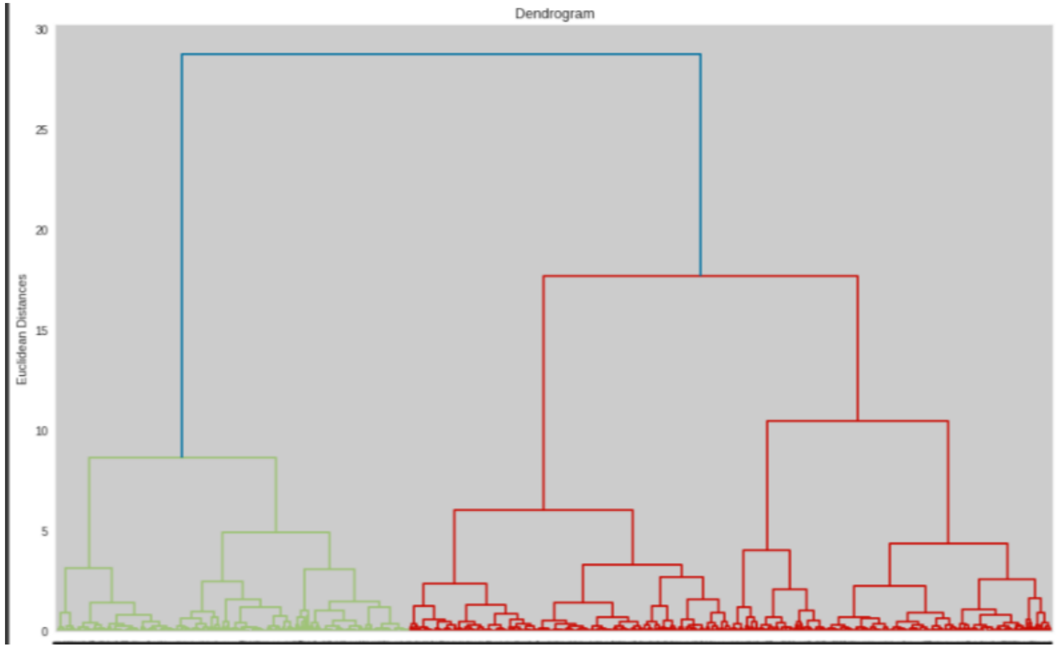
Gold Class -

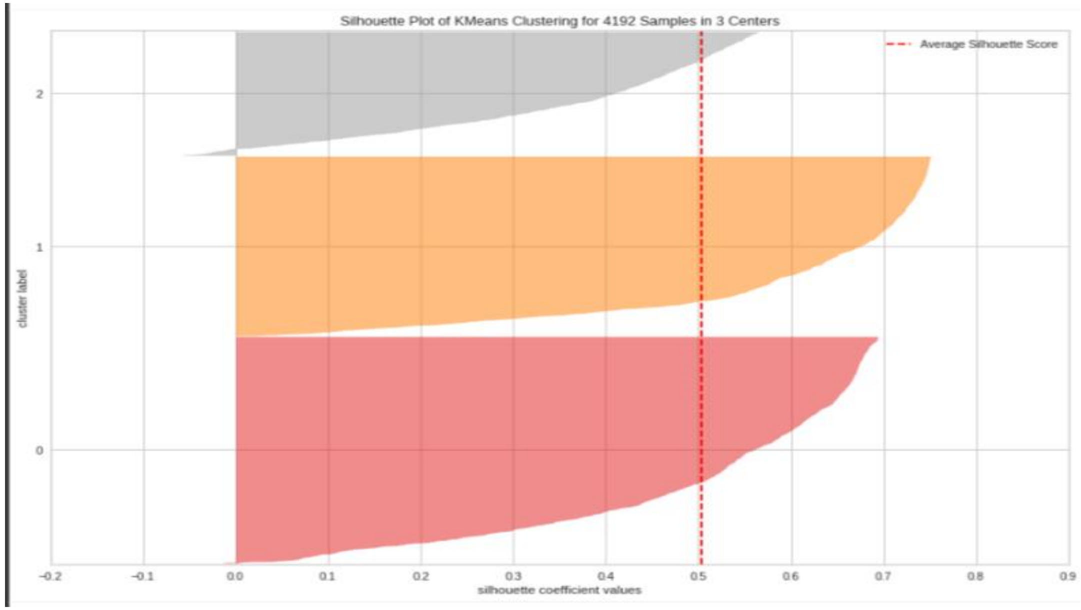
The customers which have a RFM score less than 8 all these customer are belong to category gold class.
Most number of customer are belong to these class .
These are the regular customers of company but these customers are not bought items in huge frequency.
Company don't need to target these customer for expensive product.
Also when company launch a marketing campaign. No need to target these customer at the level of Diamond and Platinum class customers.

ESTIMATION OF NUMBER OF CLUSTERS









```
# Grouping by clusters to understand the profiles
rfm_df_1.groupby('Cluster').mean()
```

Cluster	Recency	Frequency	Monetary
0	234.659155	1.661268	364.716945
1	43.076923	2.338350	492.434555
2	33.046012	10.512270	2983.167556

Diamond Customers – Cluster 2

These are the most valuable customers for the company. These customers are very Recent also bought items very frequently and these customers bought items in a mass valume.

So company need to maintain good relationship with these customers they have to offer discounts and better deals to that customer so that company able to generate more sales from that customer.

This customers are more likely to be whole sellers or distributors.

Also when company develop any marketing campaign these customers should be the first priority of the company.

Platinum customers – Cluster 1

These customers segment is more recent after diamond cluster also they bought more items in in bulk volume .

These customer are second most important customer for company after diamond customers.

Also company have to built strategy to convert these customers into Diamond Customers.

Gold customers – Cluster 0

These customers are not regular customers of company also these customers does not bought items very frequently and they d o not bought items in bulk volume.

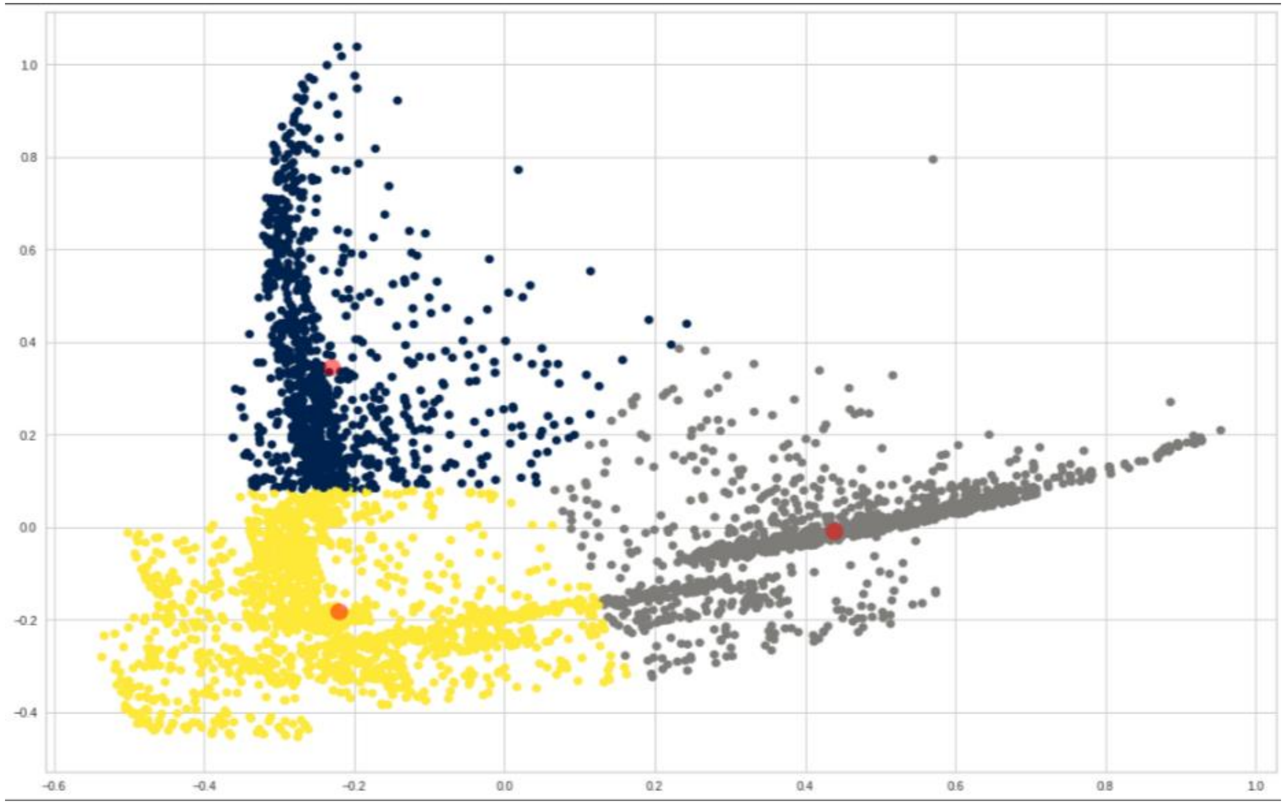
So company give less priority to this customers.

Also company built strategy to convert these customer in Platinum customers.

DIMENTINALITY REDUSCTION USING PRINCIPLE COMPONET ANALYSIS AND VISUALISING CUSTOMERS SEGMENTATION CLUSTERS FORM BY K-MENS CLUSTERING.

```python
#fit RFM data into PCA
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
pca.fit(rfm_scaled)
```

```python
#tranforn data into 2 dimentions
X_pca = pca.transform(rfm_scaled)
```



By visualizing K-means cluster we can clearly seen that it performs very well job.

T

he cluster Centroids are at long distance and there are   some outlier but it is not affection to form cluster in that case.