Capstone Project-4

Online Retail Customer Segmentation

Unsupervised Machine Learning

BY

Nehal S Jambhulkar

❖ Problem Statement:

- To identify major customer segments on a transnational data set.
- Data set contains all the transactions occurring between 1st December 2010 and 9th December 2011 for a UK-based and registered non-store online retail.
- The company mainly sells unique all-occasion gifts.
- Many customers of the company are wholesalers.

❖ Data Description:

Total Rows= 541909
Total features=8

- **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.
- **StockCode:** 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.
- **InvoiceDate**: Invoice Date and time. Numeric, the day and time when each transaction was generated.
- **UnitPrice:** Unit price. Numeric, Product price per unit in sterling.
- **CustomerID:** 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 Wrangling

❖ Information of the data

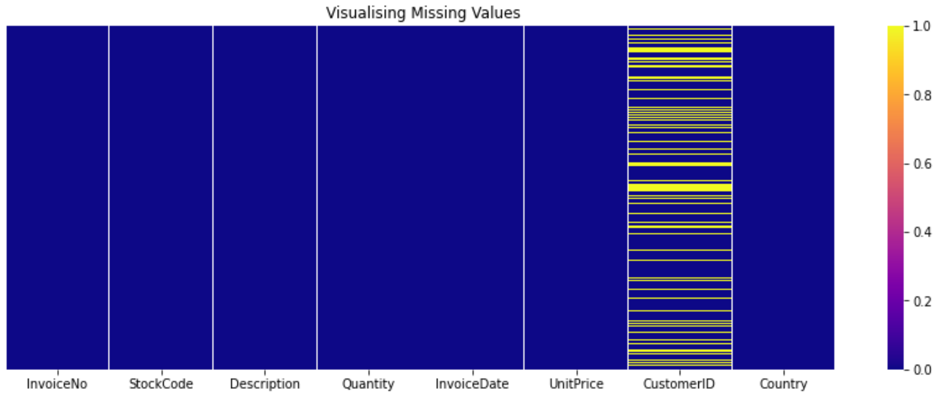
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   InvoiceNo    541909 non-null  object
 1   StockCode    541909 non-null  object
 2   Description  540455 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
```

- Invoicedate to datetime.
- If InvoiceNo starts with C means it's a cancellation.
- Shape of data after dropping entries=392692

❖ Null values

```
In [8]:  # Let's check the null values count.
         df.isnull().sum().sort_values(ascending=False)

Out[8]:  CustomerID     135080
         Description      1454
         InvoiceNo           0
         StockCode           0
         Quantity            0
         InvoiceDate         0
         UnitPrice           0
         Country             0
         dtype: int64
```



❖ Data Wrangling :

```
df[df['Quantity']<0]
```

Out[24]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
141	C536379	D	Discount	-1	2010-12-01 09:41:00	27.50	14527.0	United Kingdom
154	C536383	35004C	SET OF 3 COLOURED FLYING DUCKS	-1	2010-12-01 09:49:00	4.65	15311.0	United Kingdom
235	C536391	22556	PLASTERS IN TIN CIRCUS PARADE	-12	2010-12-01 10:24:00	1.65	17548.0	United Kingdom
236	C536391	21984	PACK OF 12 PINK PAISLEY TISSUES	-24	2010-12-01 10:24:00	0.29	17548.0	United Kingdom
237	C536391	21983	PACK OF 12 BLUE PAISLEY TISSUES	-24	2010-12-01 10:24:00	0.29	17548.0	United Kingdom
...	...	...	...	...	...	...	...	...
540449	C581490	23144	ZINC T-LIGHT HOLDER STARS SMALL	-11	2011-12-09 09:57:00	0.83	14397.0	United Kingdom
541541	C581499	M	Manual	-1	2011-12-09 10:28:00	224.69	15498.0	United Kingdom
541715	C581568	21258	VICTORIAN SEWING BOX LARGE	-5	2011-12-09 11:57:00	10.95	15311.0	United Kingdom
541716	C581569	84978	HANGING HEART JAR T-LIGHT HOLDER	-1	2011-12-09 11:58:00	1.25	17315.0	United Kingdom
541717	C581569	20979	36 PENCILS TUBE RED RETROSPOT	-5	2011-12-09 11:58:00	1.25	17315.0	United Kingdom

- Invoice No starting with C had negative entries in the quantity column means negative values in quantity column indicates cancellations.

Feature Engineering:

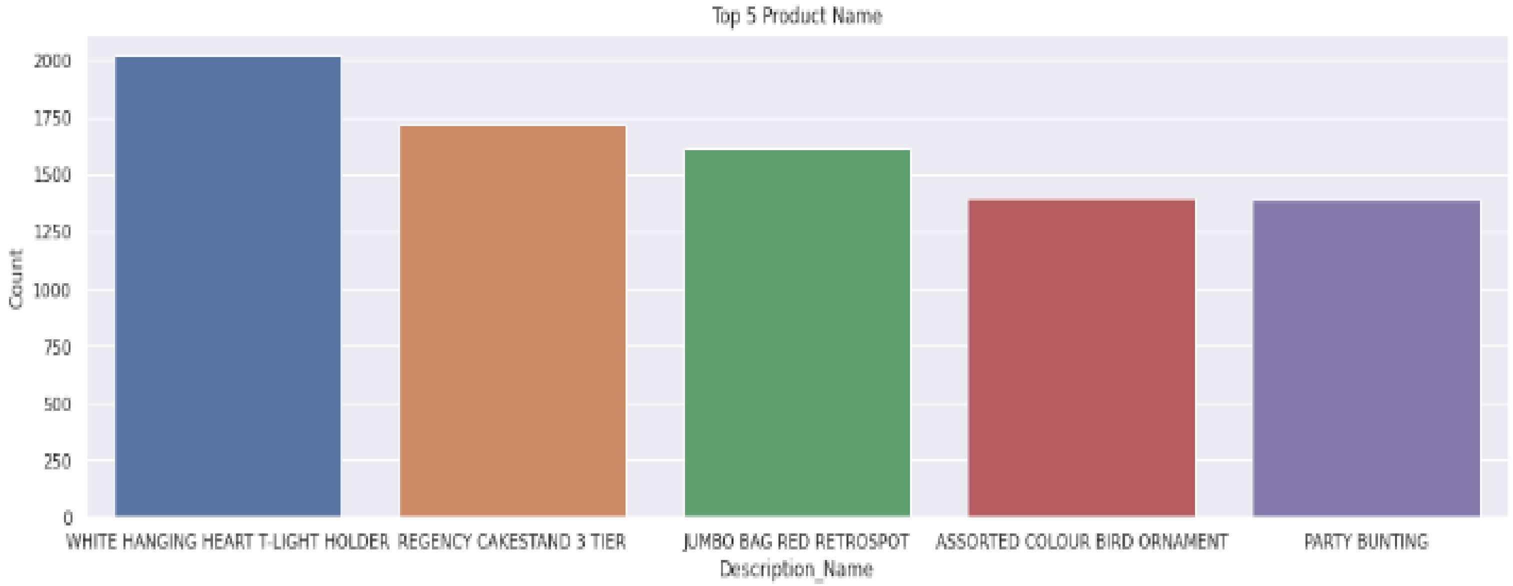
- Changed the datatype of Invoice Date column into datetime .

```
# Create some new features from Invoicedate like hours,year,month_num,day_num
df["year"] = df["InvoiceDate"].apply(lambda x: x.year)
df["month_num"] = df["InvoiceDate"].apply(lambda x: x.month)
df["day_num"] = df["InvoiceDate"].apply(lambda x: x.day)
df["hour"] = df["InvoiceDate"].apply(lambda x: x.hour)
df["minute"] = df["InvoiceDate"].apply(lambda x: x.minute)
```

```
#creating new feature (TotalAmount)
df['TotalAmount']=df['Quantity']*df['UnitPrice']
```

```
def time_type(time):
  if(time==6 or time==7 or time==8 or time==9 or time==10 or time==11):
    return 'Morning'
  elif(time==12 or time==13 or time==14 or time==15 or time==16 or time==17):
    return 'Afternoon'
  else:
    return 'Evening'
```

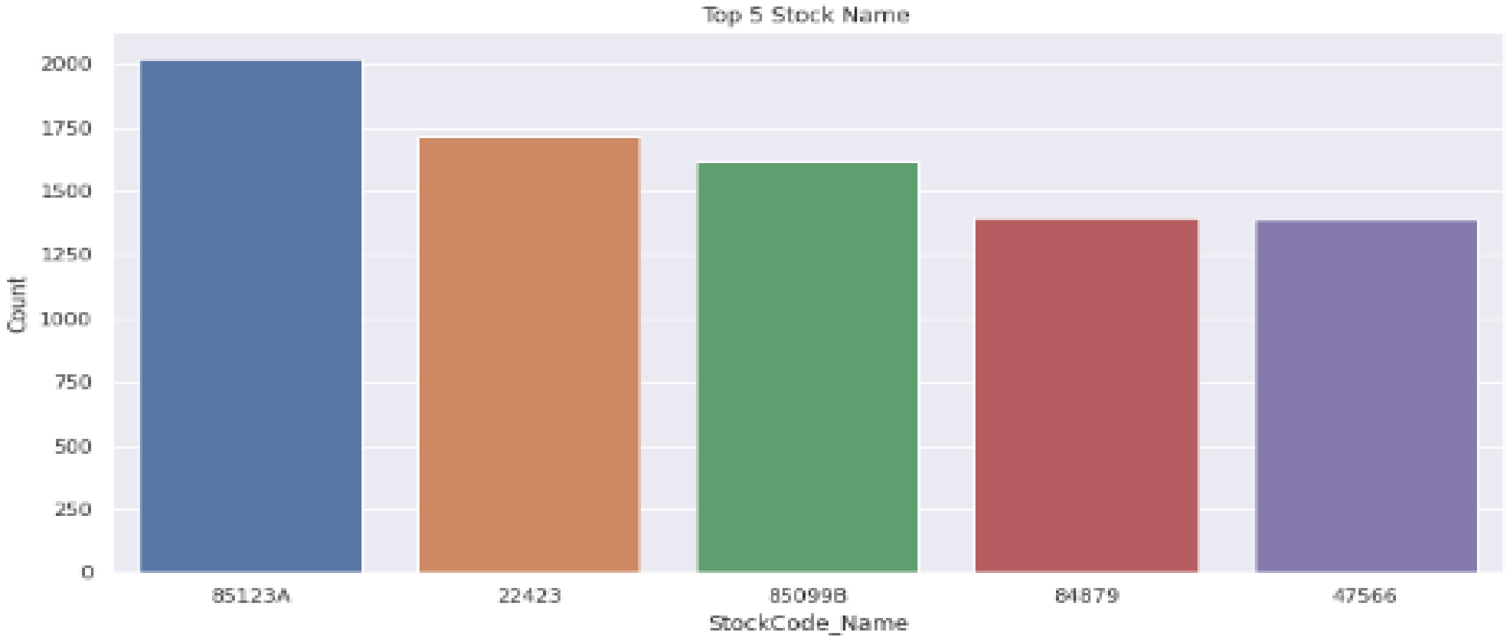
❖EDA(Exploratory Data Analysis):



Observations

- WHITE HANGING HEART T-LIGHT HOLDER is the highest selling product almost 2018 units were sold
- "REGENCY CAKESTAND 3 TIER is the 2nd highest selling product almost 1723 units were sold"

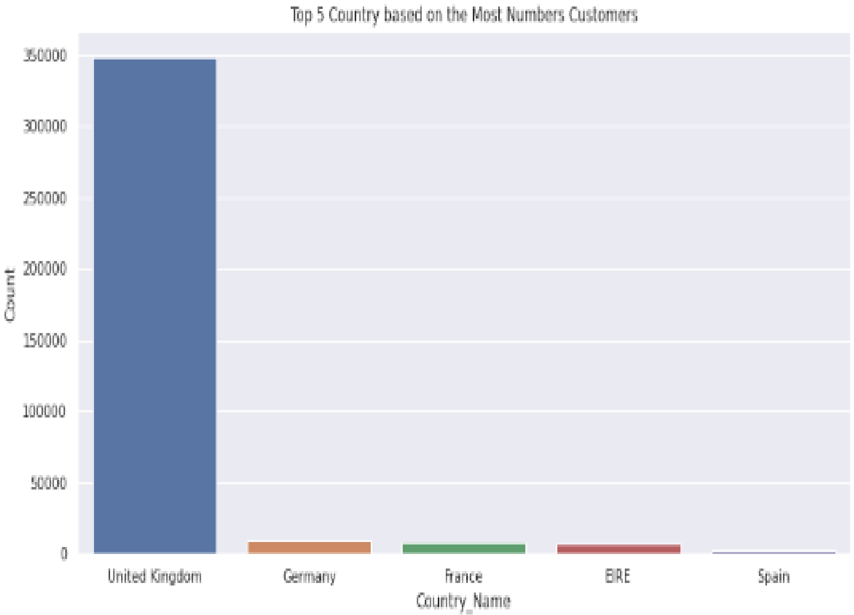
❖EDA(Exploratory Data Analysis):

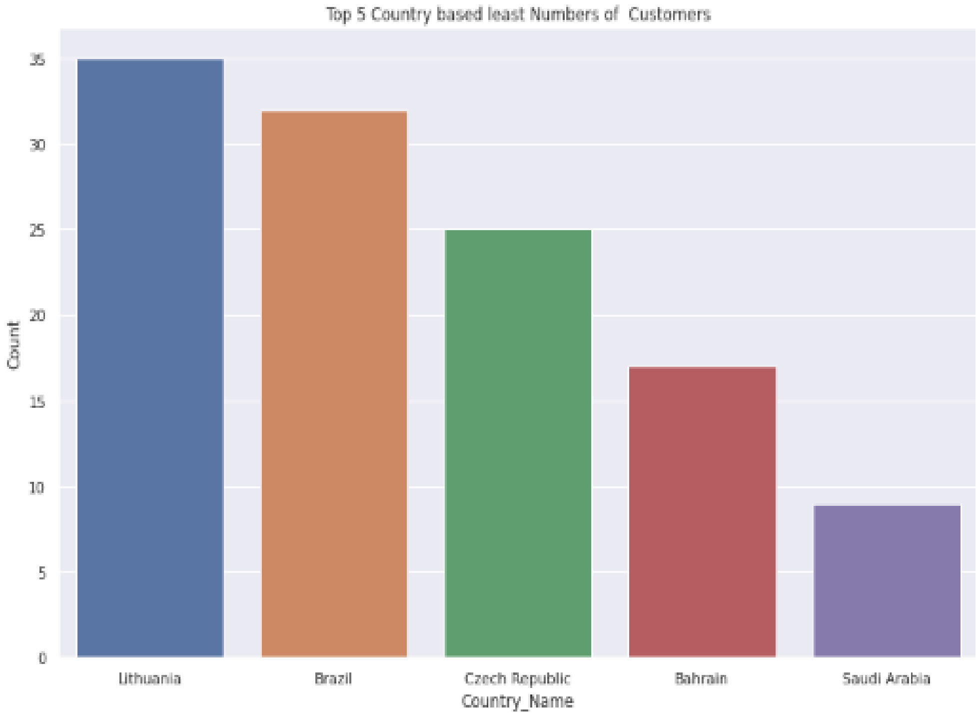


Observations

- StockCode-85123Ais the first highest selling product.
- StockCode-22423 is the 2nd highest selling product.

❖EDA(Exploratory Data Analysis):





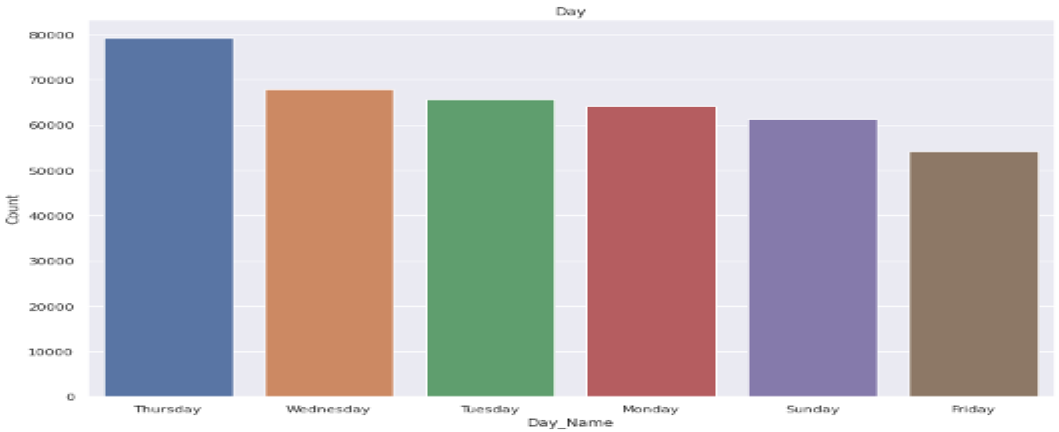
Observation

- UK has highest number of customers
- Germany,France and IreLand has almost equal number of customers

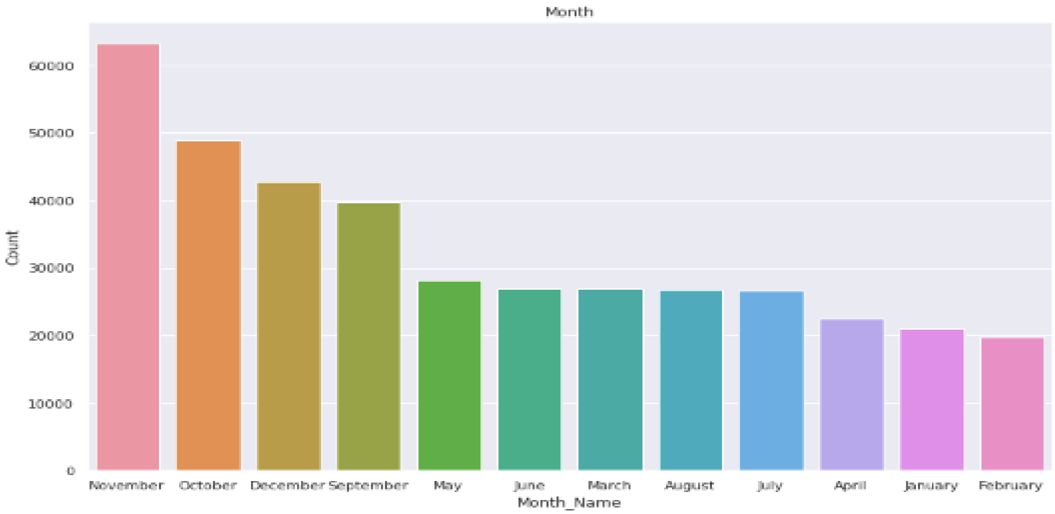
Observations

- There are very less customers from Saudi Arabia
- Bahrain is the 2nd Country having least number of customers

❖EDA(Exploratory Data Analysis):

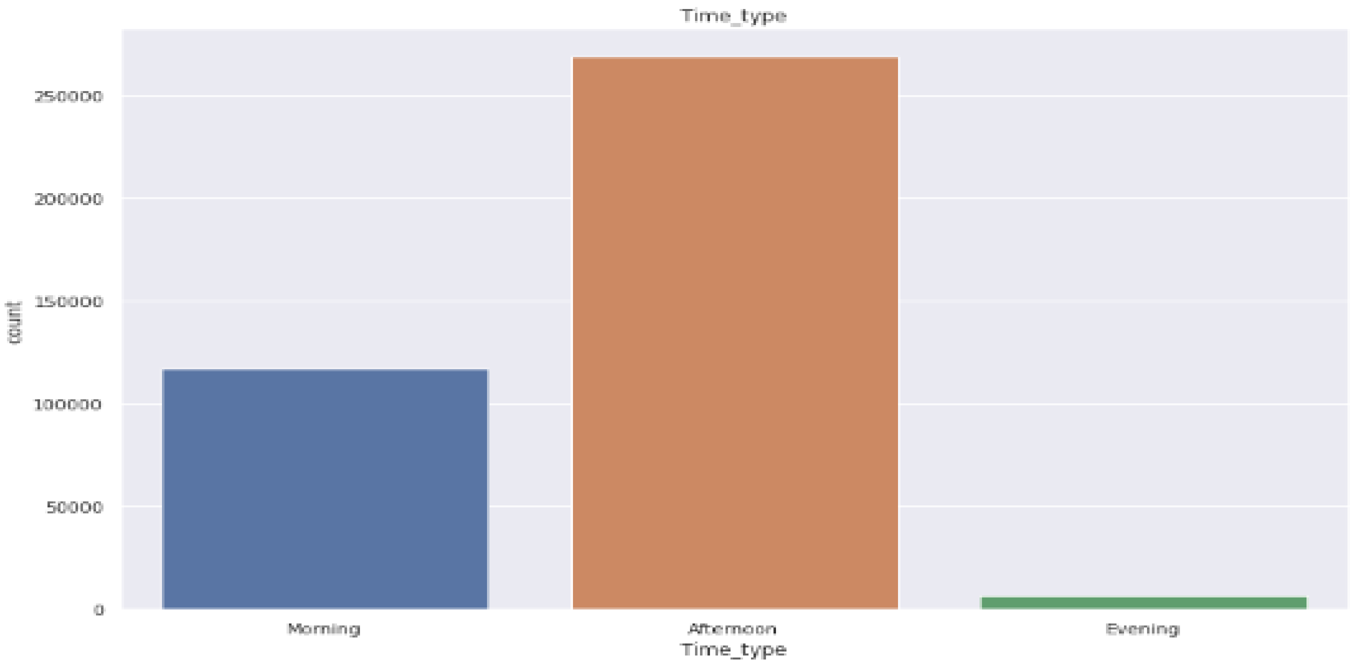


- Sales On Thursdays are very high.
- Sales On Fridays are very less.



- Most of the sales happened in November month.
- February Month had least sales.

❖EDA(Exploratory Data Analysis):



- **Most of the sales happens in the afternoon.**
- **Least sales happens in the evening.**

❖ Model Building:

RFM Model Analysis:

What is RFM?

•RFM is a method used to analyze customer value. RFM stands for RECENCY, Frequency, and Monetary.
•RECENCY: How recently did the customer visit our website or how recently did a customer purchase?
•Frequency: How often do they visit or how often do they purchase?
•Monetary: How much revenue we get from their visit or how much do they spend when they purchase?

Why it is Needed?

RFM Analysis is a marketing framework that is used to understand and analyze customer behavior based on the above three factors RECENCY, Frequency, and Monetary.
The RFM Analysis will help the businesses to segment their customer base into different homogenous groups so that they can engage with each group with different targeted marketing strategies.

❖ Model Building:

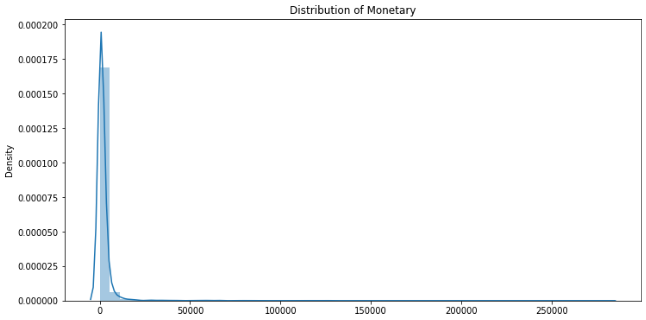
RFM Model Analysis:

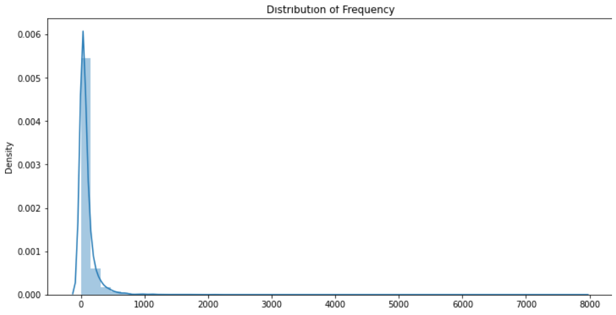
- Recency = Latest Date - Last Invoice Data.
- Frequency = Count of invoice no. of transaction(s).
- Monetary = Sum of Total Amount for each customer.

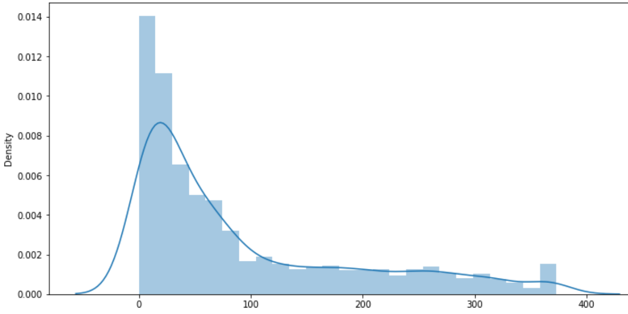
```
quantiles
```

```
{'Recency': {0.25: 17.0, 0.5: 50.0, 0.75: 141.75},
 'Frequency': {0.25: 17.0, 0.5: 41.0, 0.75: 98.0},
 'Monetary': {0.25: 306.48249999999996,
  0.5: 668.5700000000002,
  0.75: 1660.5974999999999}}
```

	CustomerID	Recency	Frequency	Monetary	R	F	M	RFMGroup	RFMScore	RFM Loyalty Level
0	14646.0	1	2076	280206.02	1	1	1	111	3	Platinaum
1	18102.0	0	431	259657.30	1	1	1	111	3	Platinaum
2	17450.0	8	336	194390.79	1	1	1	111	3	Platinaum
3	14911.0	1	5670	143711.17	1	1	1	111	3	Platinaum
4	14156.0	9	1395	117210.08	1	1	1	111	3	Platinaum
5	17511.0	2	963	91062.38	1	1	1	111	3	Platinaum
6	16684.0	4	277	66653.56	1	1	1	111	3	Platinaum
7	14096.0	4	5111	65164.79	1	1	1	111	3	Platinaum
8	13694.0	3	568	65039.62	1	1	1	111	3	Platinaum
9	15311.0	0	2366	60632.75	1	1	1	111	3	Platinaum





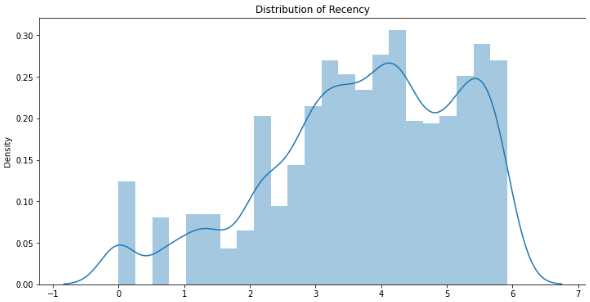


❖ Model Building:

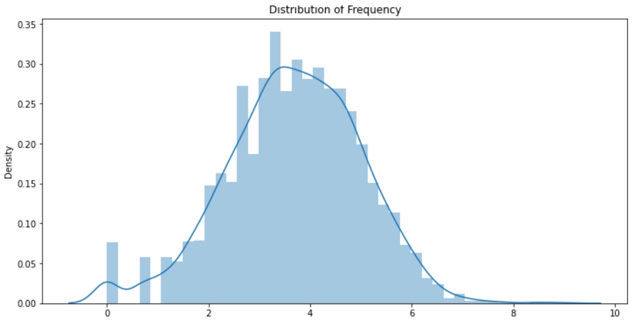
RFM Model Analysis:

.

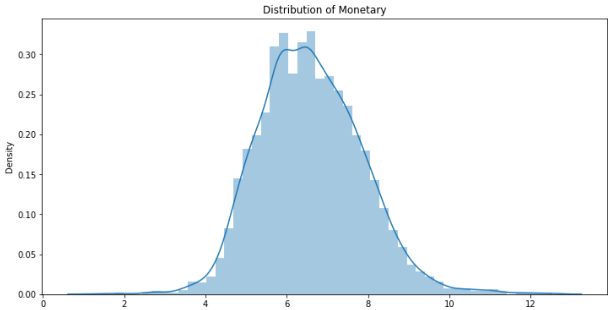
- Log transformation on Frequency, Recency and Monetary.



Recency



Frequency



Monetary

❖ Model Building:

RFM Model Analysis:

- So just using RFM Model analysis we created 4 clusters namely Platinum, Gold, Silver and Bronze.

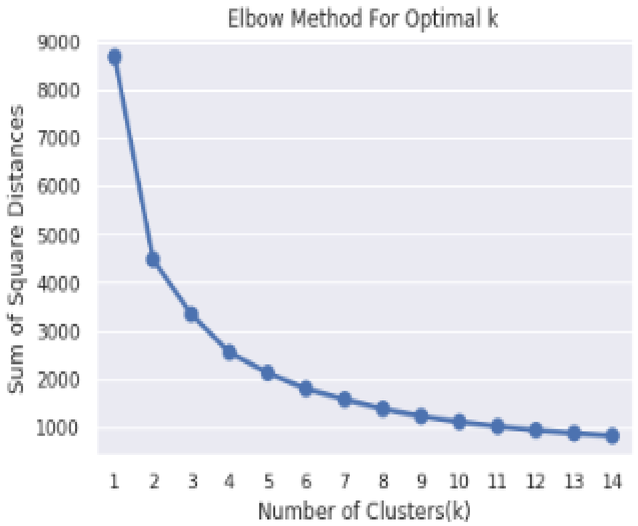
count
1118
1300
1024
896

❖ Model Building:

AI

K-means Clustering: (Recency and Monetary)

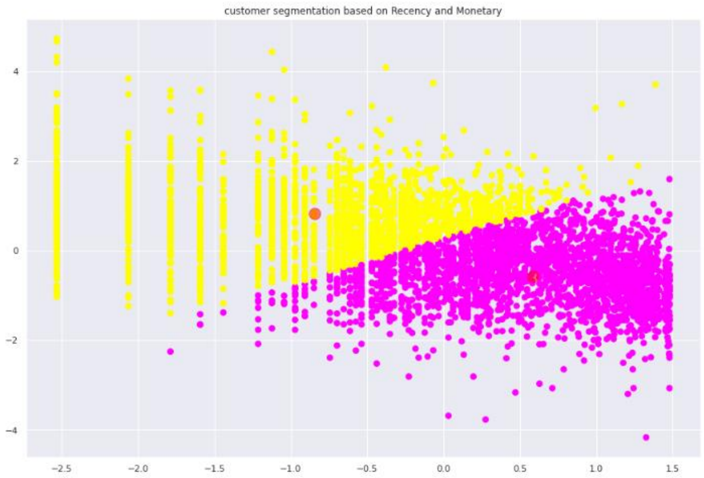
- Finding the Optimal value of cluster using Elbow method and Silhouette Score.




```
For n_clusters = 2, silhouette score is 0.4207311225472853
For n_clusters = 3, silhouette score is 0.3427491939502174
For n_clusters = 4, silhouette score is 0.365040273826764
For n_clusters = 5, silhouette score is 0.33996351493307714
For n_clusters = 6, silhouette score is 0.34473476099101463
For n_clusters = 7, silhouette score is 0.34892162760042844
For n_clusters = 8, silhouette score is 0.3379633550451048
For n_clusters = 9, silhouette score is 0.3458690365018091
For n_clusters = 10, silhouette score is 0.34770111777016427
For n_clusters = 11, silhouette score is 0.33663230013727924
For n_clusters = 12, silhouette score is 0.33999432805353114
For n_clusters = 13, silhouette score is 0.34198059345025084
For n_clusters = 14, silhouette score is 0.3455527746016807
For n_clusters = 15, silhouette score is 0.33500286931259143
```

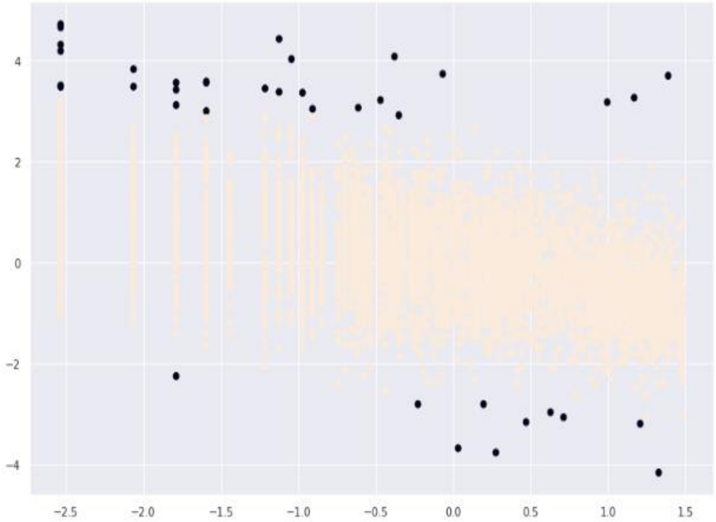
❖ Model Building:

K-means Clustering: (Recency and Monetary)



we see that ,Customers are well separate when we cluster them by Recency and Monetary

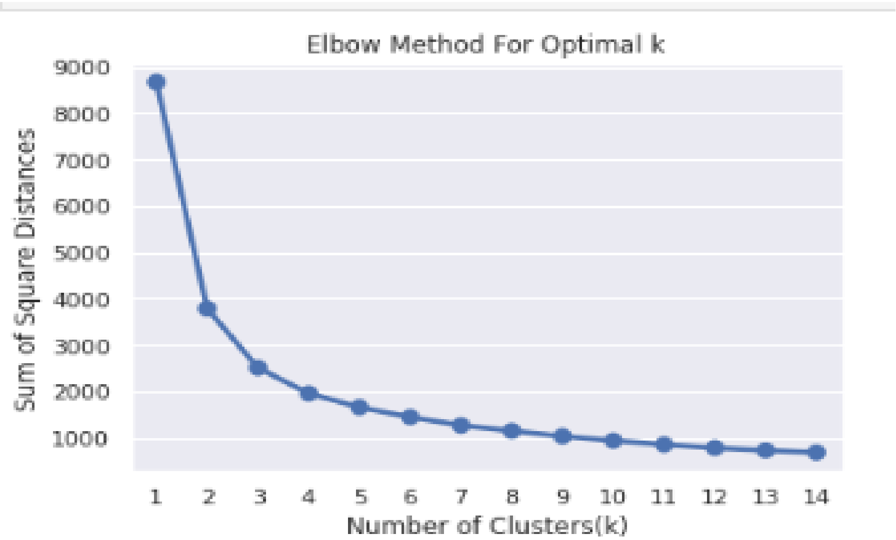
DBSCAN Algorithm (Recency and Monetary)



❖ Model Building:

K-means Clustering: (Frequency and Monetary)

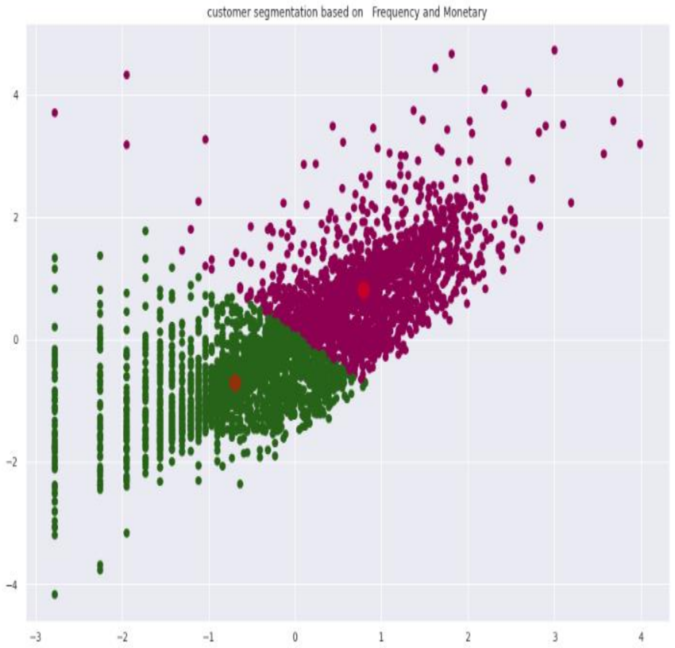
- Finding the Optimal value of cluster using Elbow method and Silhouette Score.



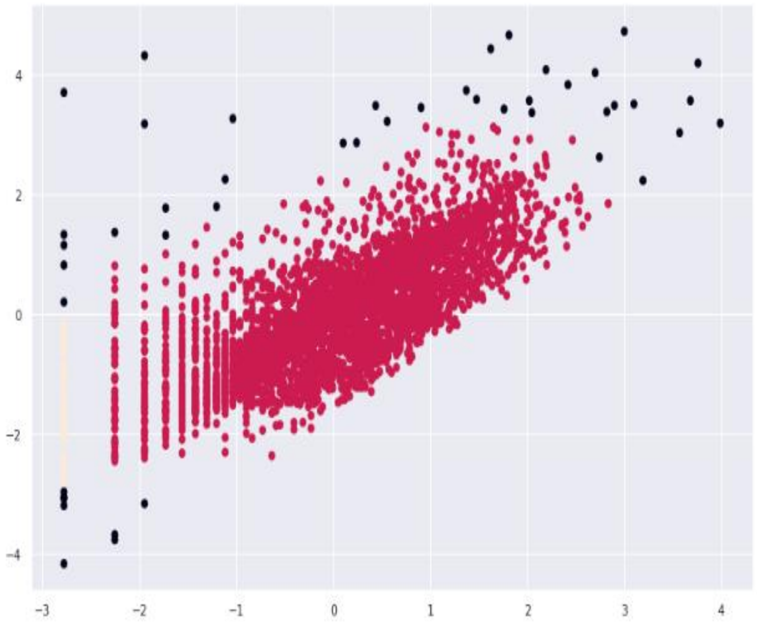

```
For n_clusters = 2, silhouette score is 0.47878955165437487
For n_clusters = 3, silhouette score is 0.40765185551085575
For n_clusters = 4, silhouette score is 0.37209909785837036
For n_clusters = 5, silhouette score is 0.3467394021038058
For n_clusters = 6, silhouette score is 0.364430854308104
For n_clusters = 7, silhouette score is 0.34437827513458963
For n_clusters = 8, silhouette score is 0.35204886038508226
For n_clusters = 9, silhouette score is 0.34573854951019156
For n_clusters = 10, silhouette score is 0.35964275798657014
For n_clusters = 11, silhouette score is 0.34127529518002286
For n_clusters = 12, silhouette score is 0.3548585654353163
For n_clusters = 13, silhouette score is 0.36337273595005365
For n_clusters = 14, silhouette score is 0.35734609483539337
For n_clusters = 15, silhouette score is 0.35722667224544963
```

❖ Model Building:

K-means Clustering: (Frequency and Monetary)



DBSCAN Algorithm (Frequency and Monetary)

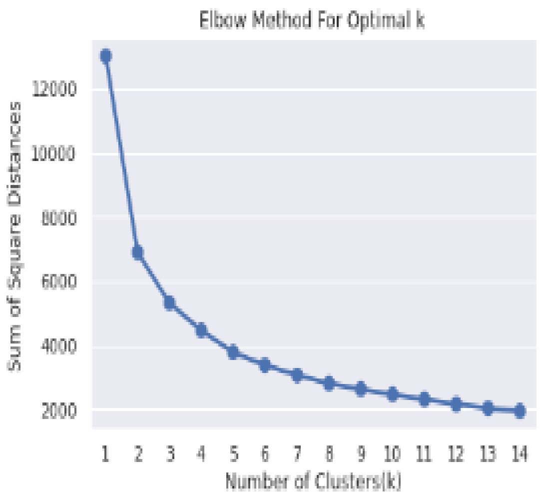


❖ Model Building:

K-means Clustering: (Recency, Frequency and Monetary)

- Finding the Optimal value of cluster using Elbow method and Silhouette Score.

.



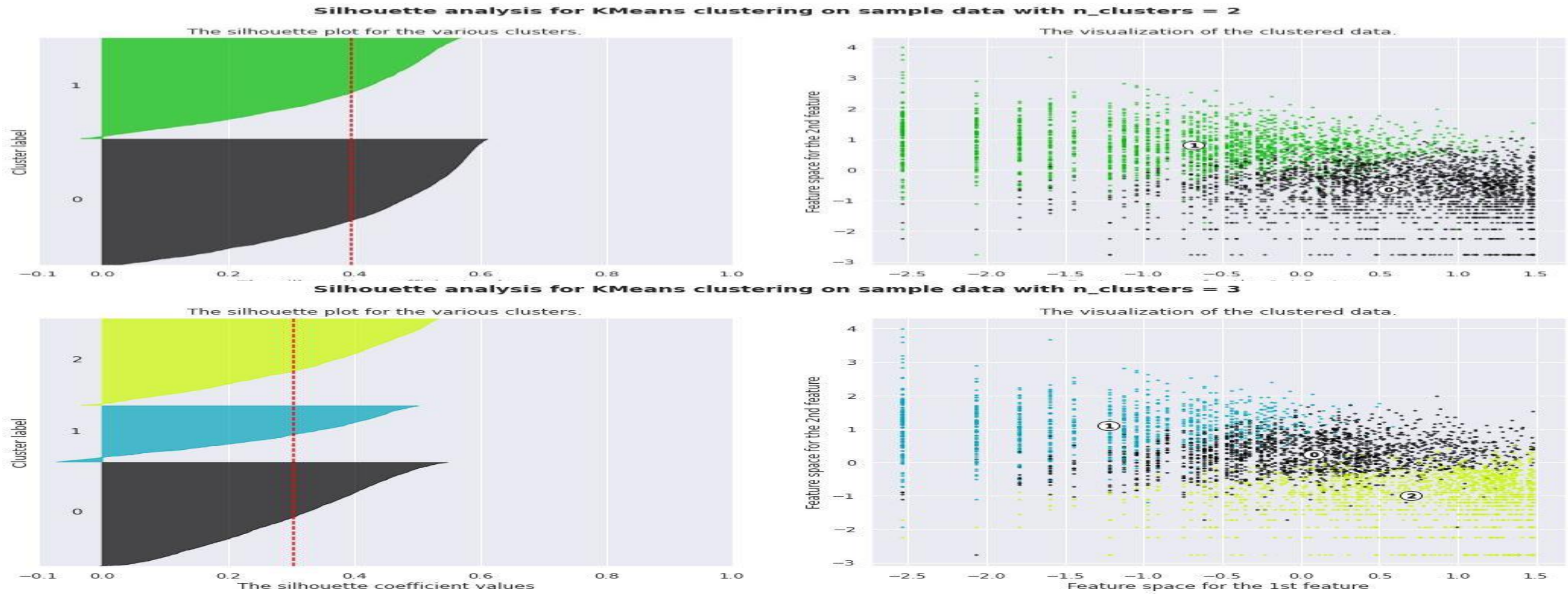

```
For n_clusters = 2 The average silhouette_score is : 0.39559432494517566
For n_clusters = 3 The average silhouette_score is : 0.305887663773877З
For n_clusters = 4 The average silhouette_score is : 0.3028850229495839
For n_clusters = 5 The average silhouette_score is : 0.2792649772843255
For n_clusters = 6 The average silhouette_score is : 0.27928761515967193
For n_clusters = 7 The average silhouette_score is : 0.2682462889803735
For n_clusters = 8 The average silhouette_score is : 0.2642066237713513
For n_clusters = 9 The average silhouette_score is : 0.25352442378461204
For n_clusters = 10 The average silhouette_score is : 0.2648431840242154
For n_clusters = 11 The average silhouette_score is : 0.2591906637856541
For n_clusters = 12 The average silhouette_score is : 0.26301779821497384
For n_clusters = 13 The average silhouette_score is : 0.2625620366156386
For n_clusters = 14 The average silhouette_score is : 0.2541667131220692
For n_clusters = 15 The average silhouette_score is : 0.25483561850201053
```

❖ Model Building:

K-means Clustering: (Recency, Frequency and Monetary)

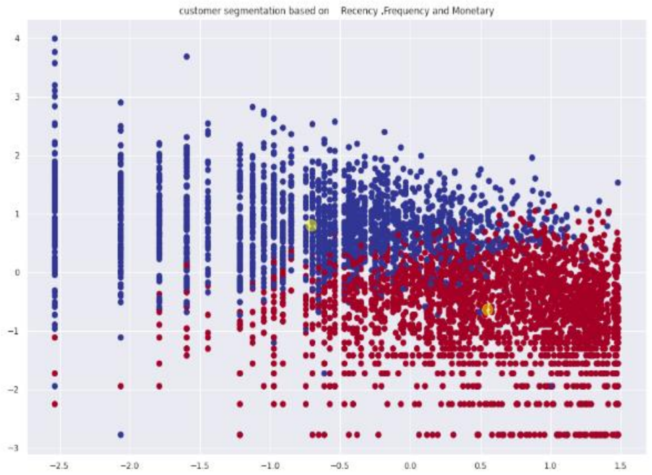
.

.



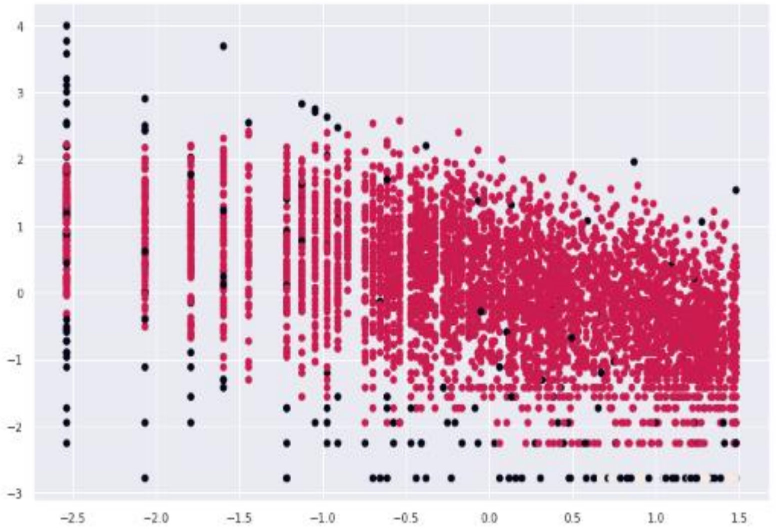
❖ Model Building:

K-means Clustering: (Recency,Frequency and Monetary)



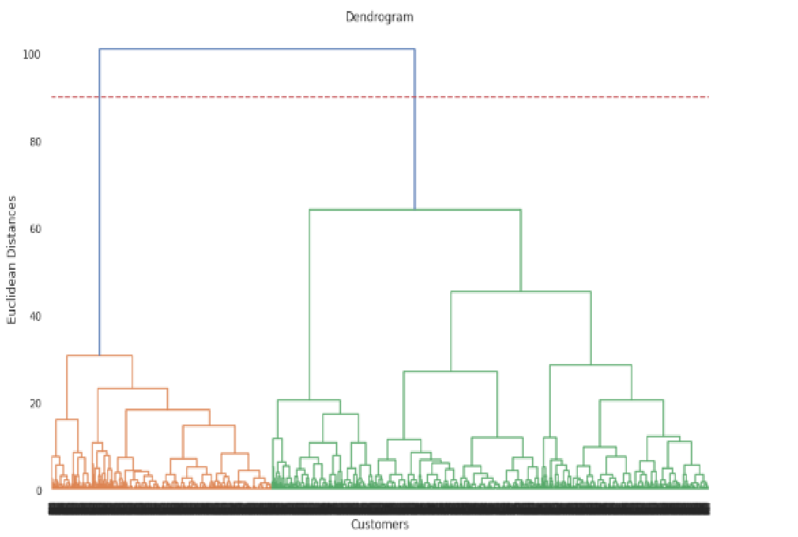
we see that ,Customers are well separate when we cluster them by Recency ,Frequency and Monetary

DBSCAN Algorithm (Recency,Frequency and Monetary)



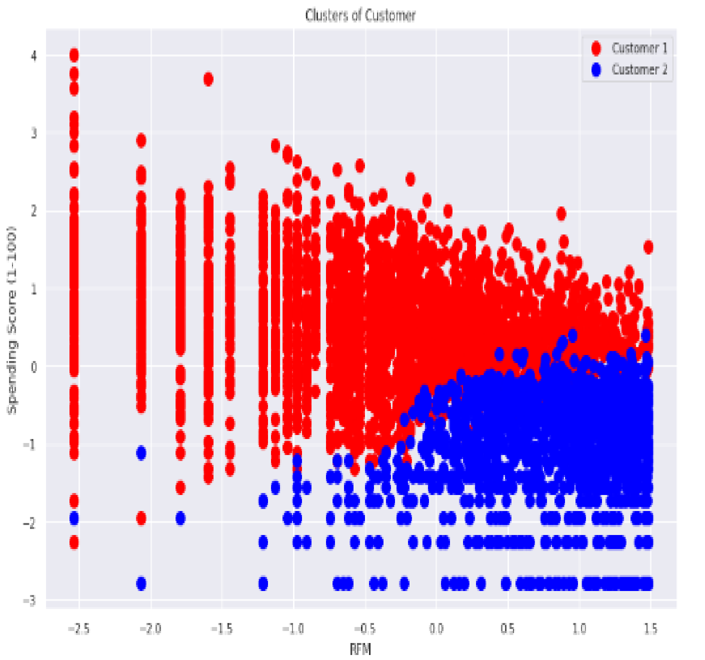
❖ Model Building:

Hierarchical Clustering(Recency, Frequency and Monetary)



The number of clusters will be the number of vertical lines which are being intersected by the line drawn using the threshold=90

No. of Cluster = 2



Optimal Number of clusters using Dendogram.(Optimal Clusters=2)

❖Summary and Conclusion:

AI

- **Firstly we did clustering based on RFM analysis. We had 4 clusters/Segmentation of customers based on RFM score.**

	Recency			Frequency			Monetary			
	mean	min	max	mean	min	max	mean	min	max	count
RFM_Loyalty_Level										
Platinaum	18.207513	0	140	238.480322	20	7676	5660.267764	316.25	280206.02	1118
Gold	63.789231	0	372	68.151538	1	521	1350.118809	114.34	168472.50	1300
Silver	121.260742	1	373	27.217773	1	98	614.235178	35.40	77183.60	1024
Bronz	191.853795	18	373	10.717634	1	39	195.198951	3.75	660.00	896

•Platinum customers=1118 (less recency but high frequency and heavy spendings)
•Gold customers=1300 (good recency,frequncy and moentary)
•Silver customers=1024(high recency, low frequency and low spendings)
•Bronz customers=896 (very high recency but very less frequency and spendings)

- ***Later we implemented the machine learning algorithms to cluster the customers.***

SL No.	Model_Name	Data	Optimal_Number_of_cluster
1	K-Means with silhouette_score	RM	2
2	K-Means with Elbow methos	RM	2
3	DBSCAN	RM	2
4	K-Means with silhouette_score	FM	2
5	K-Means with Elbow methos	FM	2
6	DBSCAN	FM	2
7	K-Means with silhouette_score	RFM	2
8	K-Means with Elbow methos	RFM	2
9	Hierarchical clustering	RFM	2
10	DBSCAN	RFM	3

❖ Summary and Conclusion:

	Recency			Frequency			Monetary			
	mean	min	max	mean	min	max	mean	min	max	count
Cluster_base_on-freq_mon_rec										
0	140.602226	1	373	24.976092	1	174	470.947643	3.75	77183.60	2426
1	30.485356	1	372	173.692469	1	7676	4050.570038	150.61	280206.02	1912

- **Above clustering is done with recency, frequency and monetary data(Kmeans Clustering) as all 3 together will provide more information.**
- **Cluster 0 has high recency rate but very low frequency and monetary. Cluster 0 contains 2426 customers.**
- **Cluster 1 has low recency rate but they are frequent buyers and spends very high money than other customers as mean monetary value is very high. Thus generates more revenue to the retail business.**

- ***With this, we are done. Also, we can use more robust analysis for the clustering, using not only RFM but other metrics such as demographics or product features.***

THANK YOU