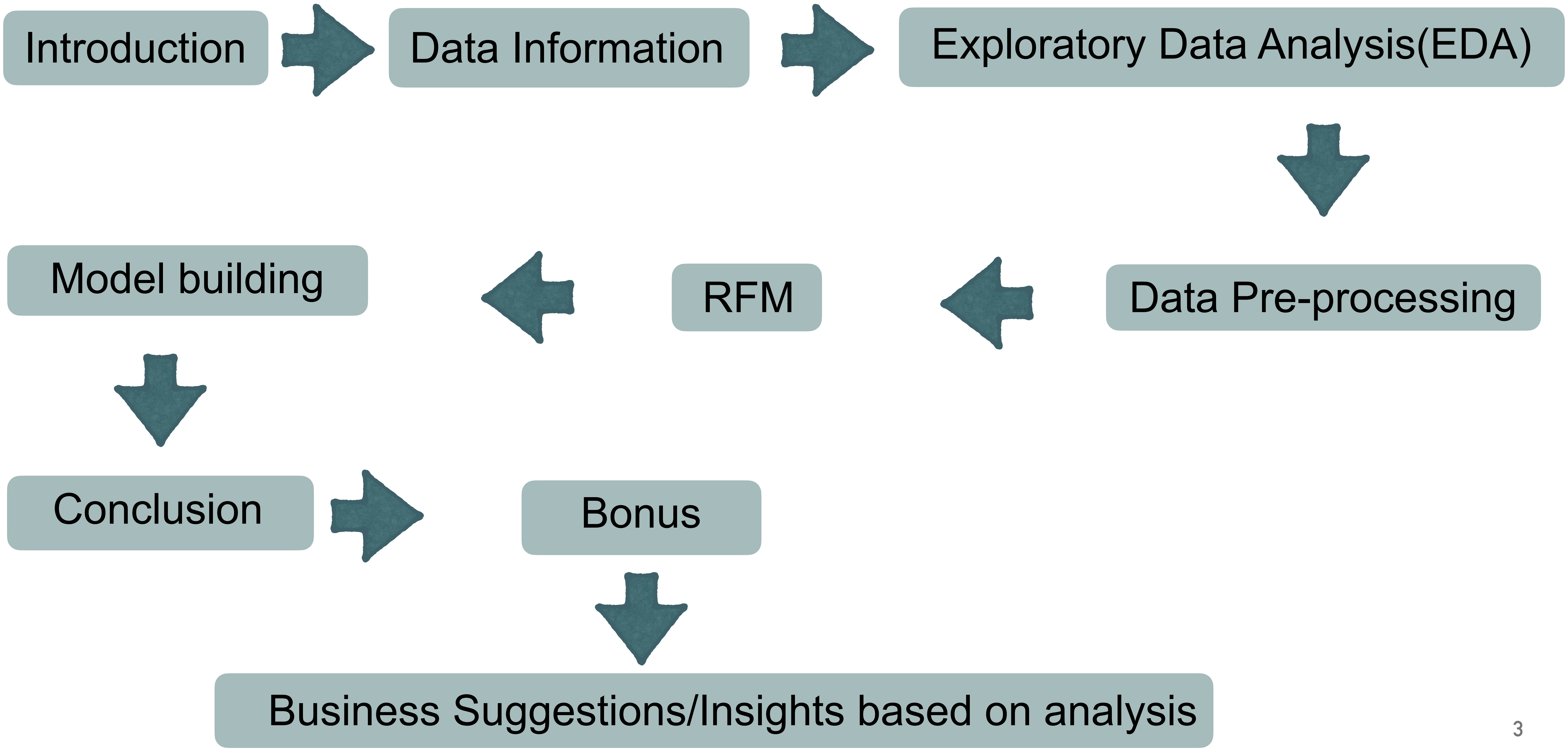
# E-Commerce Analytics
## DATA SCIENCE PRODEGREE PROJECT
## GROUP- 1


| |
|---|
| Dhanyashree Gowda |
| Nitin Satija |
| Sahil Behra |
| Shilpa Pelwar |

# TABLE OF CONTENTS

# WORK FLOW


Introduction
Data Information
Exploratory Data Analysis(EDA)
Data Pre-processing
RFM
Model building
Conclusion
Bonus
Business Suggestions/Insights based on analysis

# INTRODUCTION

➤ **Aim of the project**

Build an unsupervised learning model which can enable your company to analyze their customers via RFM (recency, frequency and monetary value) approach.

➤ **Problem**

We have to draw meaningful insights from 1 year of data & provide brief details based on the monetary value, frequency of buy, etc.

➤ **Format**

| System Elements | Details |
|---|---|
| Designing Tool | Jupyter Notebook, Tableau |
| Programming Language | Python |
| Dataset Format | CSV Files |

# *Data information*

- Data Info: (537979,12)

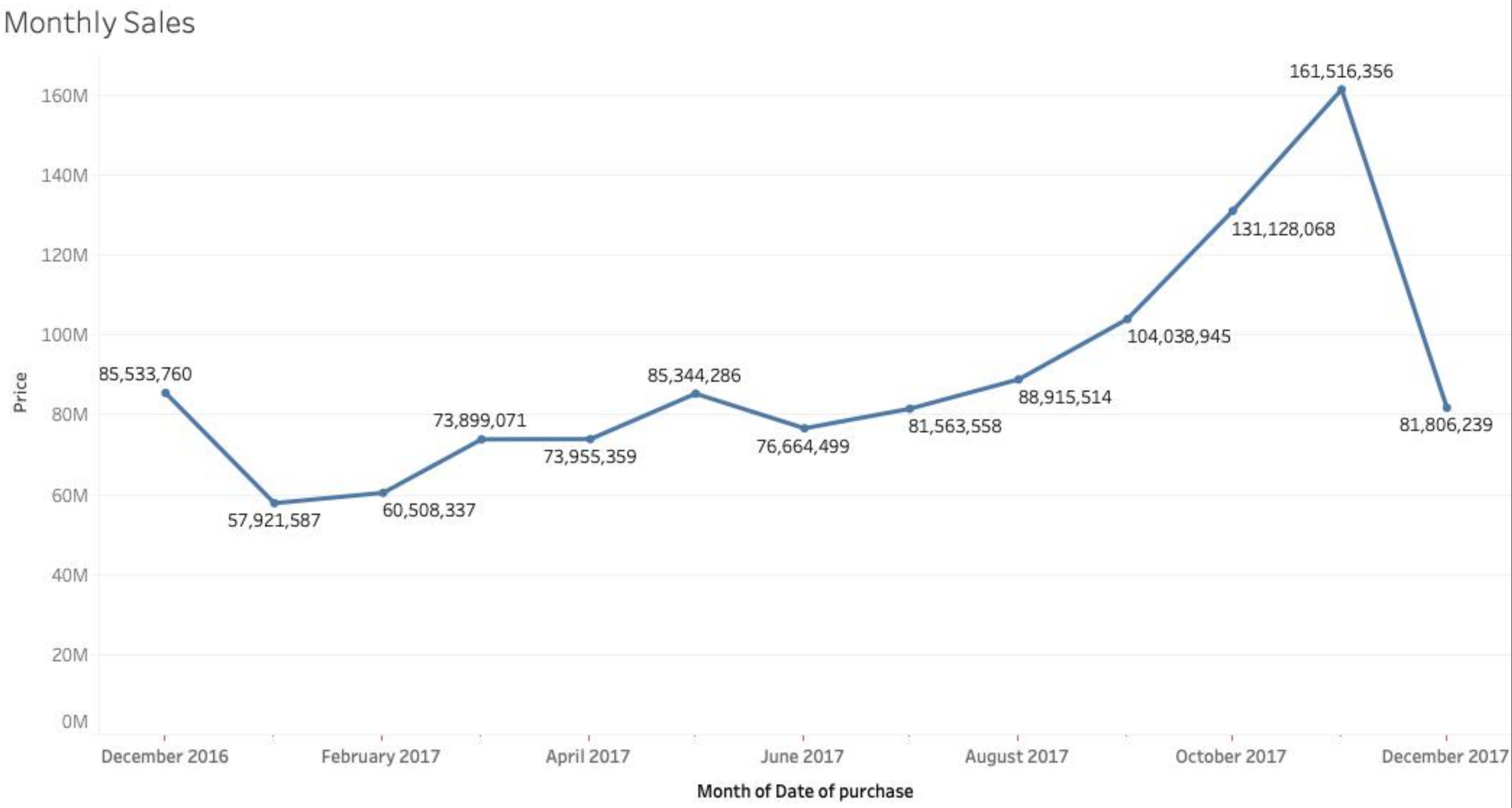
| Columns(Original) | Data Type | Null Values | Unique Values |
|---|---|---|---|
| CustomerID | Float64 | 133790 | 4349 |
| Item Code | Object | 0 | 4009 |
| InvoieNo | Float64 | 0 | 24928 |
| Date of purchase | Object | 0 | 381 |
| Quantity | Float64 | 0 | 462 |
| Time | Object | 0 | 770 |
| price per Unit | Float64 | 0 | 2900 |
| Price | Float64 | 0 | 13529 |
| Shipping Location | Object | 0 | 20 |
| Cancelled_status | Object | 529634 | 1 |
| Reason of return | Object | 537979 | 2 |
| Sold as set | Float64 | 537979 | 0 |

- The data representation is from 2016-12-02 to 2017-12-19

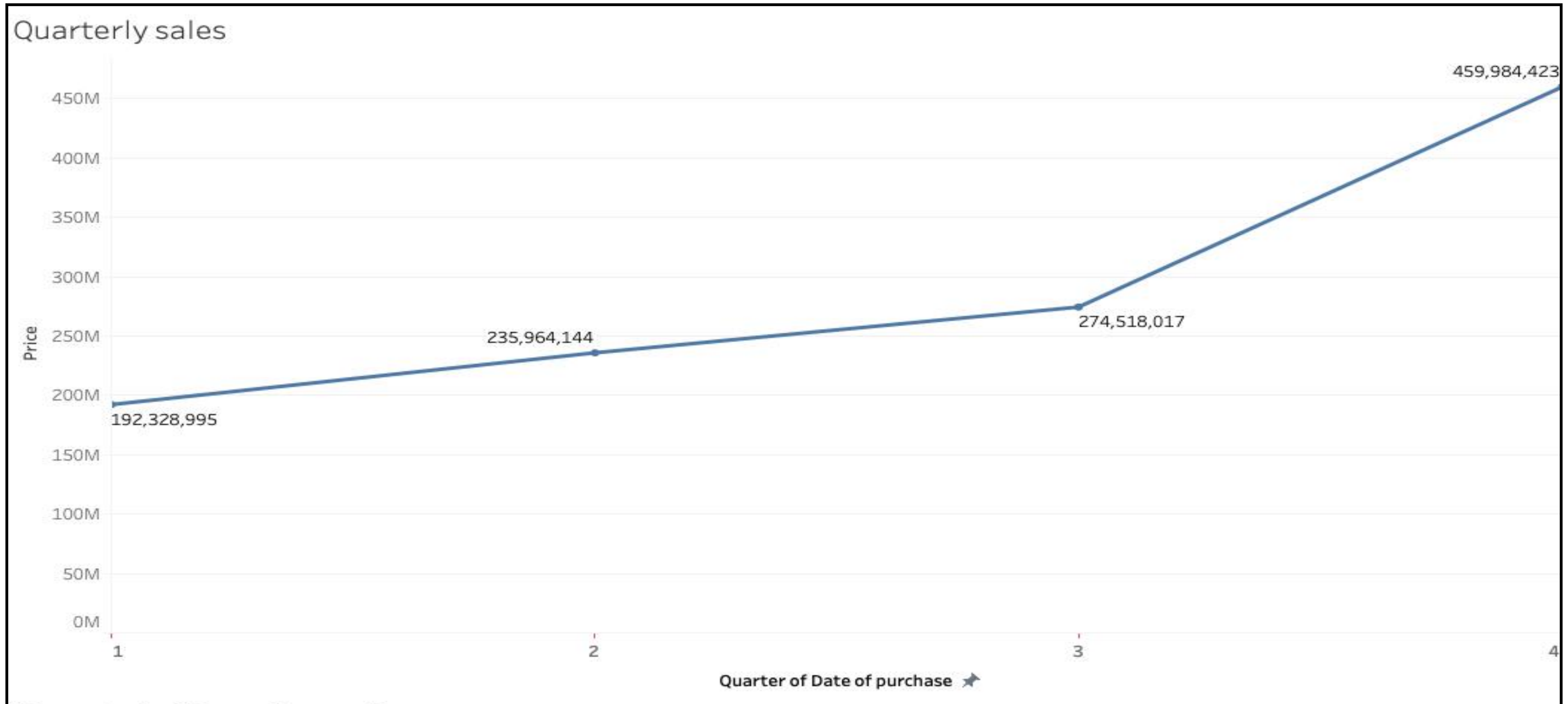
# EXPLORATORY DATA ANALYSIS (EDA)

*Before Data Pre-Processing*

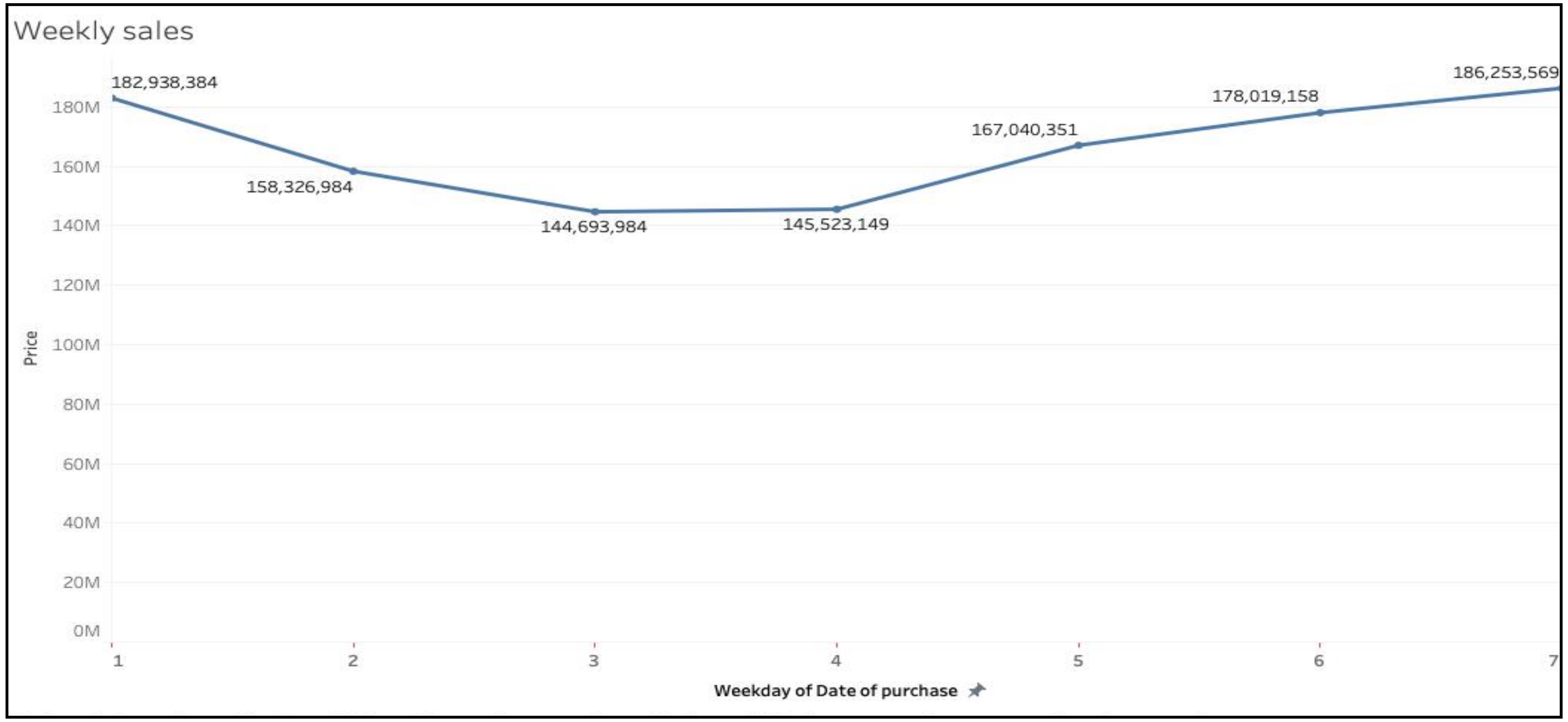
- There are 4349 unique customers
- There are 9 duplicate rows
- There are 24.9%(136927) of missing customer id
- There are 4009 unique items
- There are 24928 unique invoice numbers
- Highest number of products shipped is from location 36 (501963:quantities)
- There are 4 continuous and 5 non-continuous features.




**As our report is from December 2016 to December 2017.**
**The sales was highest in the month of November 2017 and lowest in the month of January 2017.**

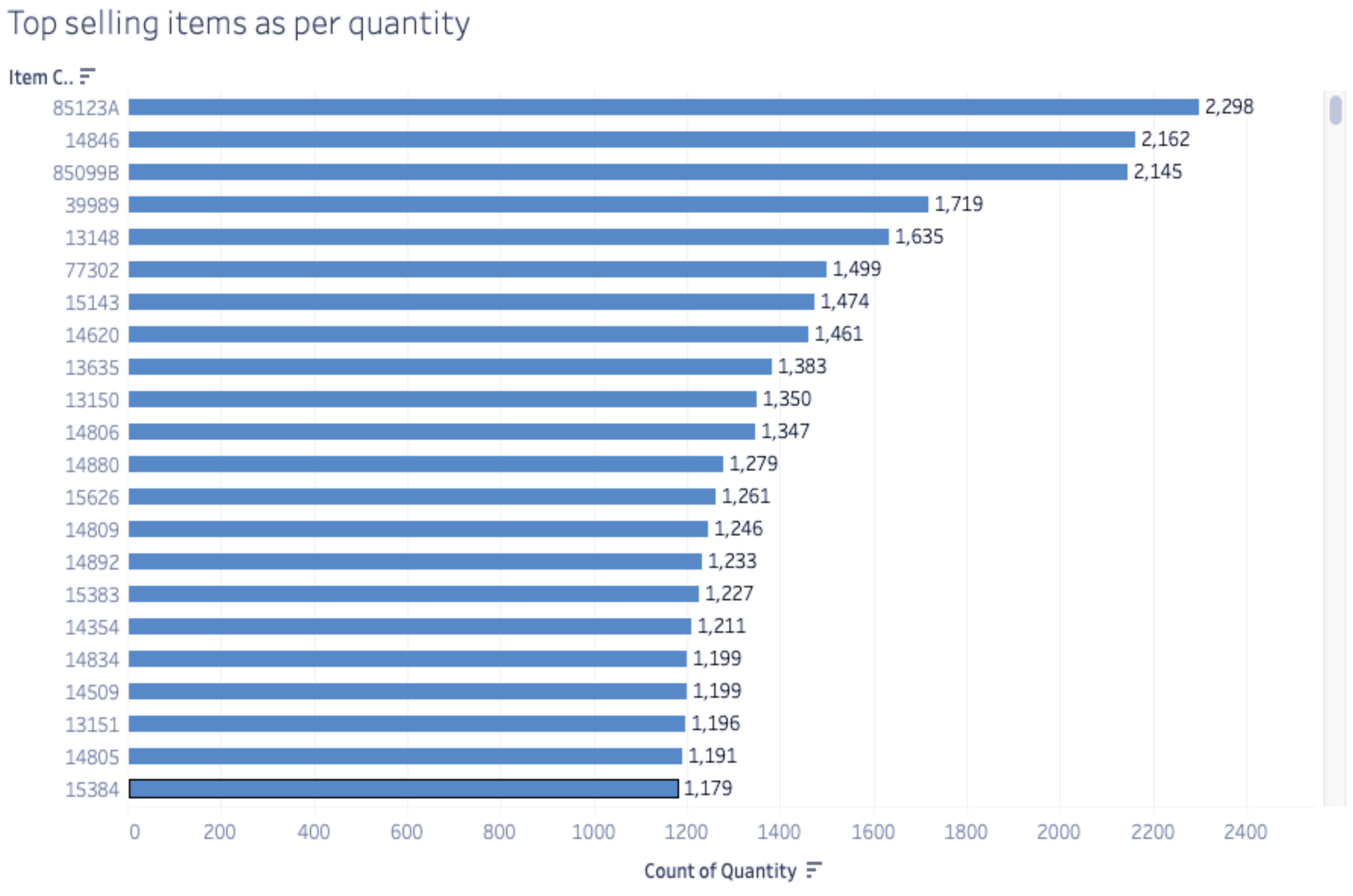



**4th quarter had the maximum sales.**



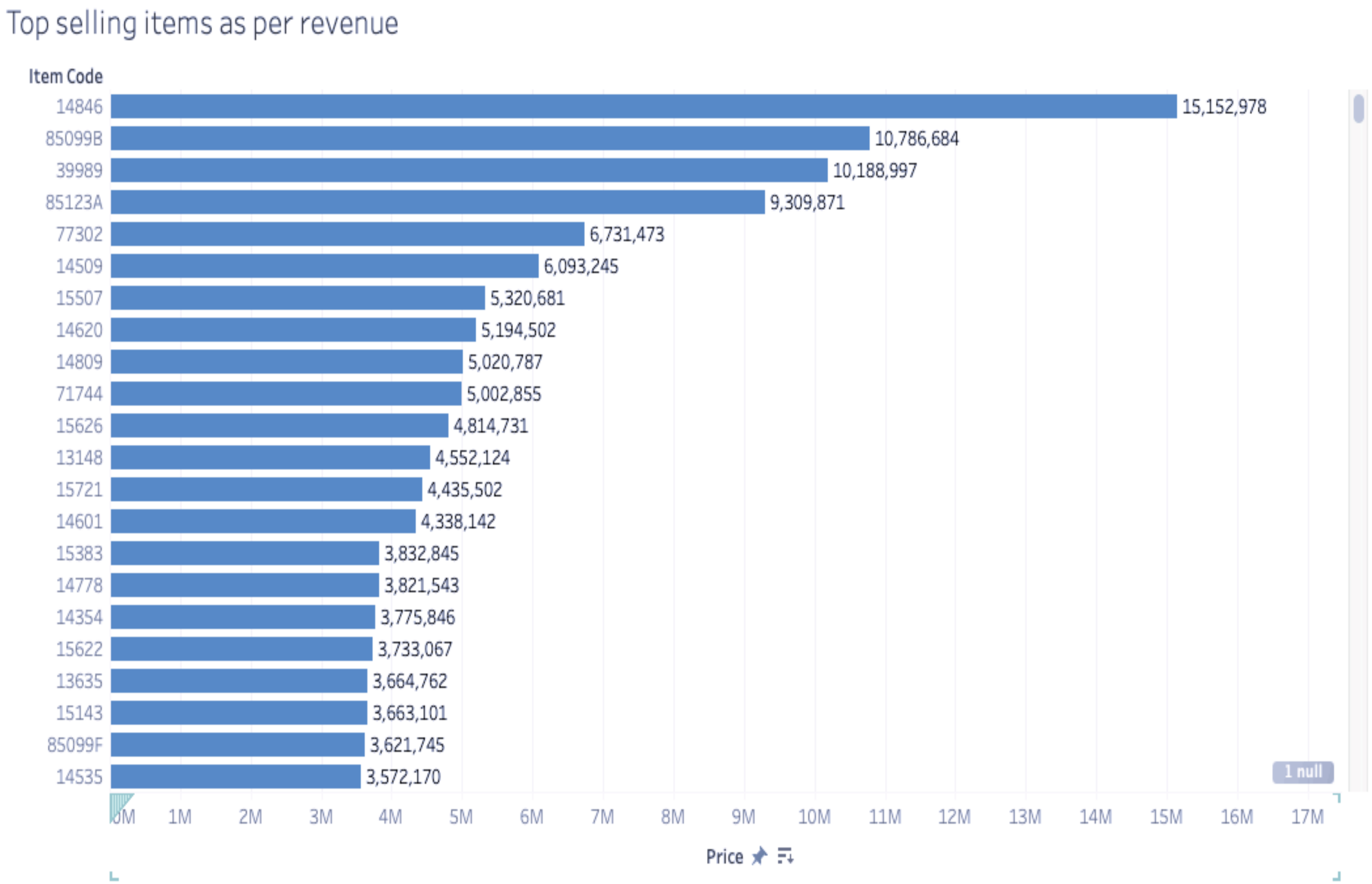



EDA Continuous...




Caption

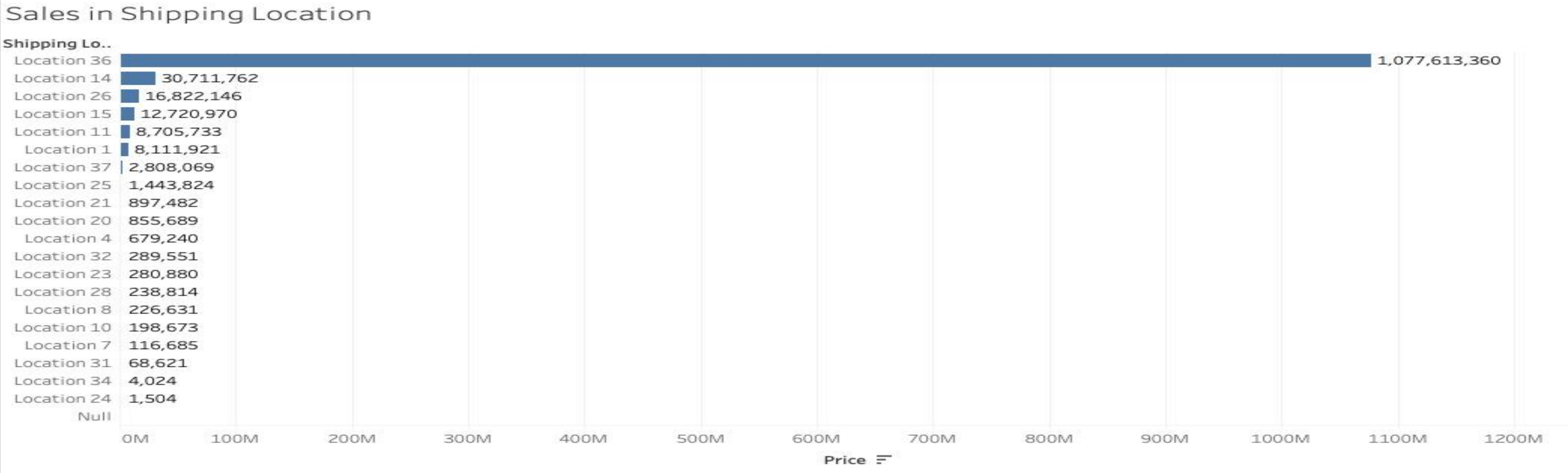
2,298 quantity of 85123A item code was sold.




Caption

**The sale of Item Code 14846 was Highest of Rs.15,152,978**

EDA Continuous...




**Highest contribution to sales is from Shipping location 36.**

# DATA PRE-PROCESSING

*Dropping missing values:*

**<u>CustomerID</u>:** We check in the data files for the null values and we remove them from those data files containing them. We delete 133790 empty rows from CustomerID, because of its unique feature, we are unable to impute it.

**<u>Dropping Features</u>**: As we can see from above that **'Cancelled status**', '**Reason of Return', 'Sold as set' and 'Price per Unit**' are mostly null and not provide any information for decision making so we can remove them.

**<u>Dropping Duplicate Values:</u>** Remove all duplicate rows except the first one. There are 9 duplicate values.

**<u>Dropping Negative values:</u>** Drop all negative values from Quantity which indicates returned product and not impacting business in Sales/Monetary Terms

**<u>Change Data Type</u>:** We will convert the datatype of 'Date of purchase' to datetime format.

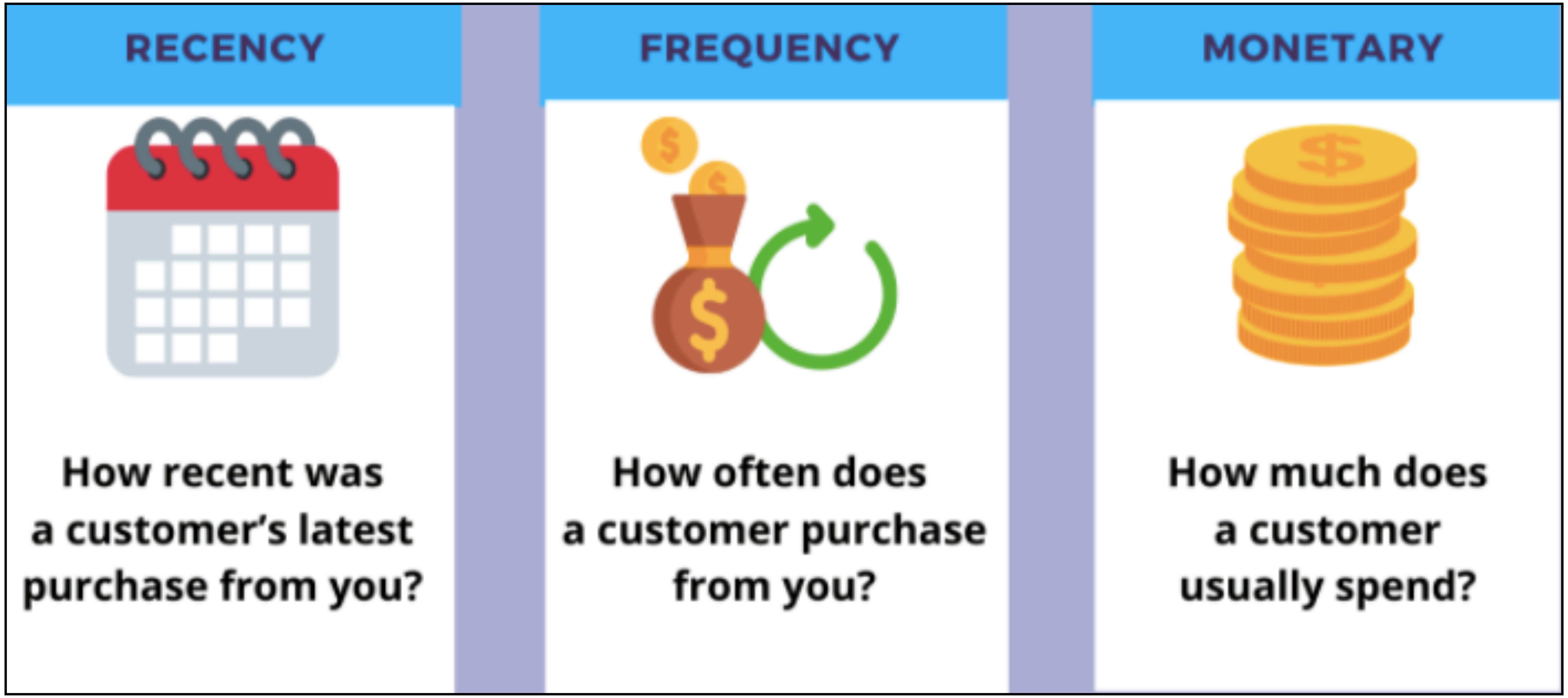
# *After Data Pre-processing :*

- ➤ There are 4324 unique customers
- ➤ There are 0 duplicate rows
- ➤ There are no missing customer id
- ➤ There are 3637 unique items
- ➤ There are 18305 unique invoice numbers
- ➤ Highest number of Products is shipping from Location 36 (368829 - quantities)
- ➤ Data shape is (395998,9)

| Columns(Original) | Columns(New) | Data Type | Null Values | Unique Values | |
|---|---|---|---|---|---|
| CustomerID | CustomerID | Float64 | 0 | 4324 | Continuous |
| Item Code | Item_Code | Object | 0 | 3637 | Non continuous |
| InvoieNo | InvoiceNo | Float64 | 0 | 18305 | Continuous |
| Date of purchase | Date_of_purchase | Object | 0 | 381 | Non continuous |
| Quantity | Quantity | Float64 | 0 | 280 | Continuous |
| Time | Time | Object | 0 | 740 | Non continuous |
| Price | Revenue | Float64 | 0 | 10681 | Continuous |
| Shipping Location | Shipping_Location | Object | 0 | 20 | Non continuous |

# RFM

- Recency, frequency, monetary value is a Marketing analysis tool used to identify an organization's best customers by measuring and analyzing spending habits.
- RFM metrics:



## ➤ *Steps of RFM(Recency, Frequency, Monetary):*

1. Calculate the Recency, Frequency, Monetary values for each customer.
2. We will Make RFM Table.

In [35]:
```
RFM = FullRaw.groupby('CustomerID').agg({'Date_of_purchase':lambda x:(maxi-x.max()).days,
                                        'InvoiceNo':lambda x:x.nunique(),'Revenue':lambda x:x.sum()})
RFM = RFM.reset_index()
RFM.head()
```

Out[35]:

| | CustomerID | Date_of_purchase | InvoiceNo | Revenue |
|---|---|---|---|---|
| **0** | 2.0 | 4 | 7 | 553704.0 |
| **1** | 3.0 | 77 | 4 | 257404.0 |
| **2** | 4.0 | 19 | 1 | 176613.0 |
| **3** | 5.0 | 311 | 1 | 41976.0 |
| **4** | 6.0 | 37 | 7 | 166972.0 |

In [36]:
```
RFM.columns = ['CustomerID', 'Recency', 'Frequency', 'Monetary']
RFM.head()
```

Out[36]:

| | CustomerID | Recency | Frequency | Monetary |
|---|---|---|---|---|
| **0** | 2.0 | 4 | 7 | 553704.0 |
| **1** | 3.0 | 77 | 4 | 257404.0 |
| **2** | 4.0 | 19 | 1 | 176613.0 |
| **3** | 5.0 | 311 | 1 | 41976.0 |
| **4** | 6.0 | 37 | 7 | 166972.0 |

3. Now we will calculate score for each customer.
4. Add segment bin values to RFM table using quartile.

| | CustomerID | Recency | Frequency | Monetary | r_quartile | f_quartile | m_quartile | RFM_Score |
|---|---|---|---|---|---|---|---|---|
| **0** | 2.0 | 4 | 7 | 553704.0 | 4 | 4 | 4 | 444 |
| **1** | 3.0 | 77 | 4 | 257404.0 | 2 | 3 | 4 | 234 |
| **2** | 4.0 | 19 | 1 | 176613.0 | 4 | 1 | 3 | 413 |
| **3** | 5.0 | 311 | 1 | 41976.0 | 1 | 1 | 2 | 112 |
| **4** | 6.0 | 37 | 7 | 166972.0 | 3 | 4 | 3 | 343 |

## 5. To make score simple and easy to interpret, we add the score(4+4+4).

```
In [46]: RFM_Score_Seg['RFM_Score_Sum'] = RFM_Score_Seg[['r_quartile','f_quartile','m_quartile']].sum(axis=1)
         RFM_Score_Seg.head()
```

Out[46]:

| | CustomerID | Recency | Frequency | Monetary | r_quartile | f_quartile | m_quartile | RFM_Score | RFM_Score_Sum |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 2.0 | 4 | 7 | 553704.0 | 4 | 4 | 4 | 444 | 12 |
| **1** | 3.0 | 77 | 4 | 257404.0 | 2 | 3 | 4 | 234 | 9 |
| **2** | 4.0 | 19 | 1 | 176613.0 | 4 | 1 | 3 | 413 | 8 |
| **3** | 5.0 | 311 | 1 | 41976.0 | 1 | 1 | 2 | 112 | 4 |
| **4** | 6.0 | 37 | 7 | 166972.0 | 3 | 4 | 3 | 343 | 10 |

## 6. As per score, we segment each customer into four categories as: Platinum, Gold, Silver and Bronze.

```
In [47]: Loyalty_Level = ['Bronze','Silver','Gold','Platinum']
         Score_Qant = pd.qcut(RFM_Score_Seg.RFM_Score_Sum, q = 4, labels = Loyalty_Level)
         RFM_Score_Seg['RFM_Loyalty_Level'] = Score_Qant.values
         RFM_Score_Seg.head()
```

Out[47]:

| | CustomerID | Recency | Frequency | Monetary | r_quartile | f_quartile | m_quartile | RFM_Score | RFM_Score_Sum | RFM_Loyalty_Level |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2.0 | 4 | 7 | 553704.0 | 4 | 4 | 4 | 444 | 12 | Platinum |
| **1** | 3.0 | 77 | 4 | 257404.0 | 2 | 3 | 4 | 234 | 9 | Gold |
| **2** | 4.0 | 19 | 1 | 176613.0 | 4 | 1 | 3 | 413 | 8 | Gold |
| **3** | 5.0 | 311 | 1 | 41976.0 | 1 | 1 | 2 | 112 | 4 | Bronze |
| **4** | 6.0 | 37 | 7 | 166972.0 | 3 | 4 | 3 | 343 | 10 | Gold |

# MODEL BUILDING

- We are using unsupervised Modelling (K-Means clustering)
- Start clustering with main data set. Shape is (395998,9)
- Remove Features which are not contributing in decision making. (Time, price per unit and Quantity columns)
- Above step is crucial as we start to transform our raw data to the data with the appropriate format for the upcoming clustering algorithm to consume. New shape is (395998,6).

- ➢ Changing Non-Continuous to Continuous variables.

```
In [56]: encoder = LabelEncoder()

         FullRaw_Clustering['Item_Code'] = encoder.fit_transform(FullRaw_Clustering['Item_Code'])
         FullRaw_Clustering['Date_of_purchase'] = encoder.fit_transform(FullRaw_Clustering['Date_of_purchase'])
         FullRaw_Clustering['Shipping_Location'] = encoder.fit_transform(FullRaw_Clustering['Shipping_Location'])
```

- ➢ Now we standardize data.

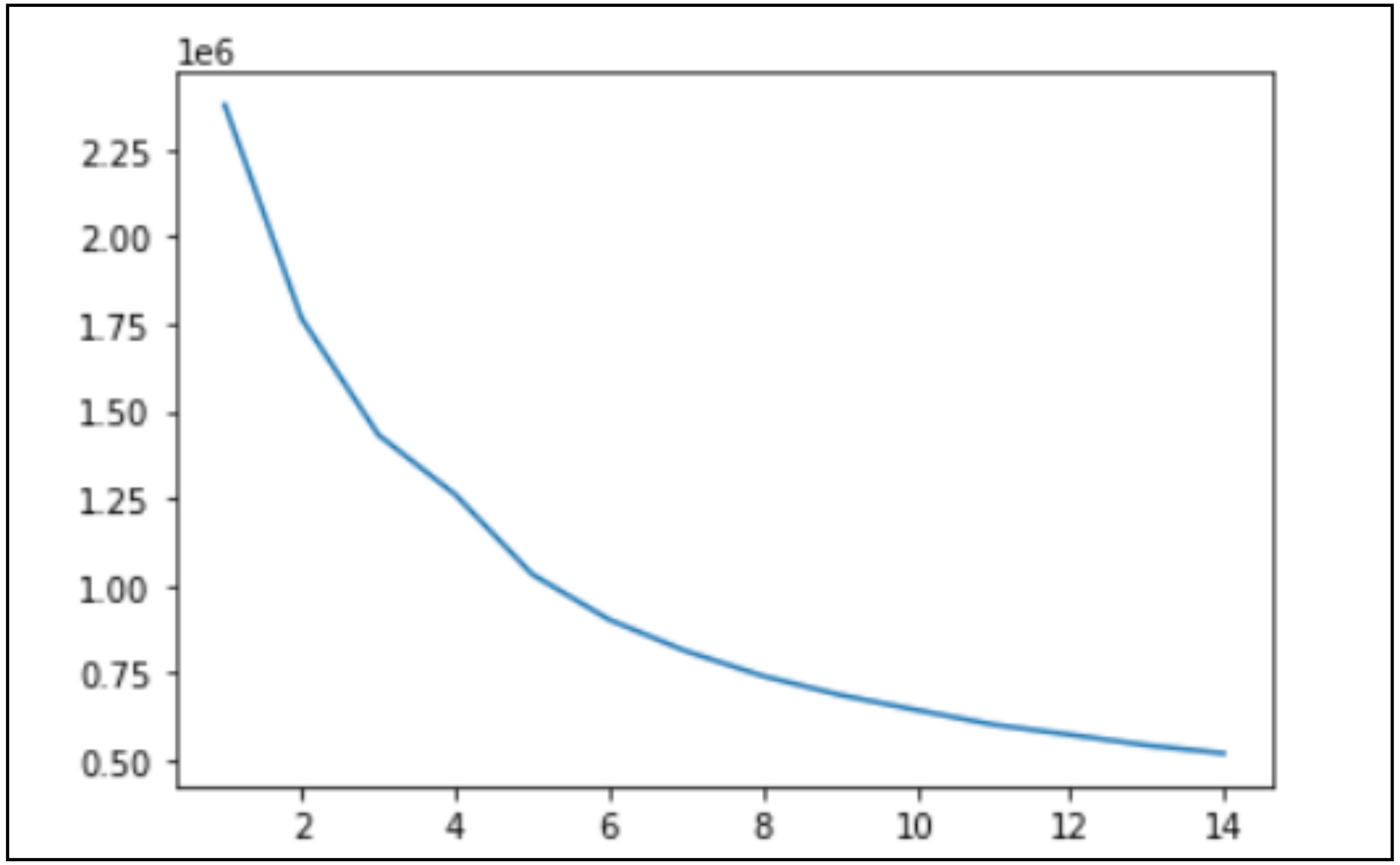
```
In [59]: FullRaw_Clustering_Scaling = StandardScaler().fit(FullRaw_Clustering)
         FullRaw_Clustering_Std = FullRaw_Clustering_Scaling.transform(FullRaw_Clustering)
         FullRaw_Clustering_Std = pd.DataFrame(FullRaw_Clustering_Std, columns = FullRaw_Clustering.columns)

         FullRaw_Clustering_Std.head()
```

- ➤ Calculating number of cluster by “Elbow Method”.

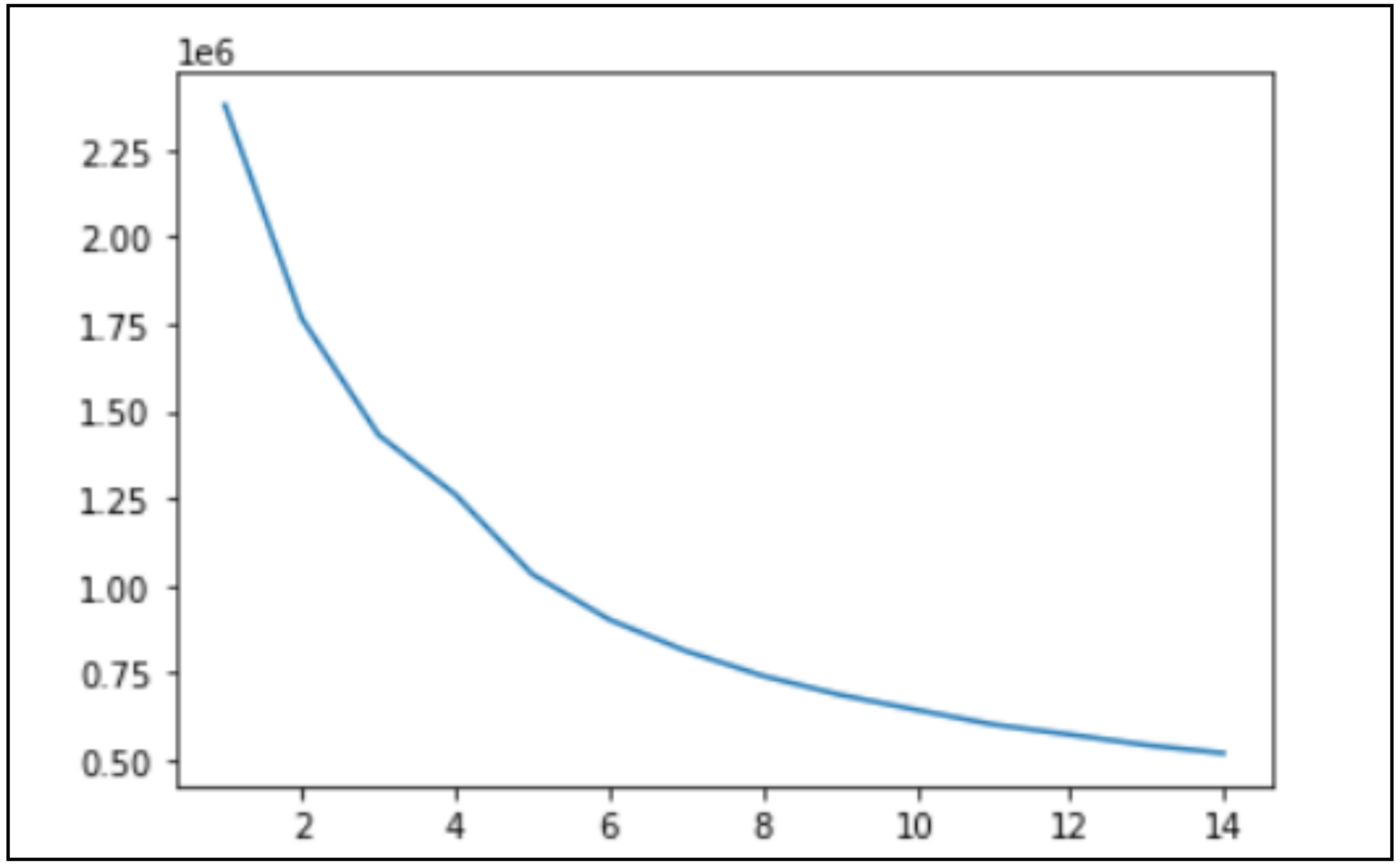
```
In [60]: WSS = []
         for k in range(1, 15):
             kmeans = KMeans(n_clusters=k, random_state = 123).fit(FullRaw_Clustering_Std)
             WSS.append(kmeans.inertia_)

         WSS
Out[60]: [2375987.9999999353,
          1764897.677933506,
          1431517.6495252794,
          1260673.6357591867,
          1032561.022396422,
          902787.7785971275,
          812463.5072577209,
          741282.3224805972,
          687240.3039619938,
          643877.5153796239,
          601870.2663092331,
          572995.8100097922,
          542821.8734511508,
          519725.1596351662]
```




- ➤ From above graph, number of clusters are not clear. So based on Silhouette Score we are going with 3 cluster model.

- Now we standardize data.
- Calculating number of cluster by “Elbow Method”.




- From above graph, number of clusters are not clear. So based on silhouette we are going with 3 cluster model.

- Combining cluster info with the main data (395598,7)
- Below is the cluster size.

*CLUSTER SIZES*

```
In [65]: FullRaw2['Cluster'].value_counts()

Out[65]: 0    213312
         1    164027
         2     18659
         Name: Cluster, dtype: int64
```

- Number of unique customers in each clusters.

```
In [66]: Num_of_Unique_Customers_in_Each_Clusters = FullRaw2.groupby('Cluster')['CustomerID'].nunique()
         Num_of_Unique_Customers_in_Each_Clusters

Out[66]: Cluster
         0    3403
         1    3013
         2    2832
         Name: CustomerID, dtype: int64
```

# ➤ Merging loyalty levels from RFM segmentation data with clustering data.

```
In [68]: newdf = pd.DataFrame(RFM_Score_Seg, columns = ['CustomerID','RFM_Loyalty_Level'])
         newdf
```

Out[68]:

| | CustomerID | RFM_Loyalty_Level |
|---|---|---|
| 0 | 2.0 | Platinum |
| 1 | 3.0 | Gold |
| 2 | 4.0 | Gold |
| 3 | 5.0 | Bronze |
| 4 | 6.0 | Gold |
| ... | ... | ... |
| 4319 | 4368.0 | Bronze |
| 4320 | 4369.0 | Bronze |
| 4321 | 4370.0 | Silver |
| 4322 | 4371.0 | Platinum |
| 4323 | 4372.0 | Gold |

4324 rows × 2 columns

```
In [69]: FullRaw2
```

Out[69]:

| | CustomerID | Item_Code | InvoiceNo | Date_of_purchase | Revenue | Shipping_Location | Cluster |
|---|---|---|---|---|---|---|---|
| 0 | 4355.0 | 2031 | 398177 | 329 | 1926.0 | 0 | 2 |
| 1 | 4352.0 | 975 | 394422 | 305 | 1740.0 | 0 | 2 |
| 2 | 4352.0 | 973 | 394422 | 312 | 1866.0 | 0 | 2 |
| 3 | 4352.0 | 3303 | 388633 | 261 | 1869.0 | 0 | 2 |
| 4 | 4352.0 | 1685 | 394422 | 310 | 1888.0 | 0 | 2 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 395993 | 37.0 | 1040 | 402292 | 359 | 384.0 | 19 | 0 |
| 395994 | 37.0 | 1040 | 402292 | 358 | 398.0 | 19 | 0 |
| 395995 | 21.0 | 2939 | 363890 | 19 | 2464.0 | 19 | 1 |
| 395996 | 21.0 | 3412 | 363890 | 19 | 4068.0 | 19 | 1 |
| 395997 | 21.0 | 1040 | 363890 | 15 | 4940.0 | 19 | 1 |

395998 rows × 7 columns

```
In [70]: Final_df = pd.merge(FullRaw2,newdf, on = "CustomerID")
         Final_df
```

Out[70]:

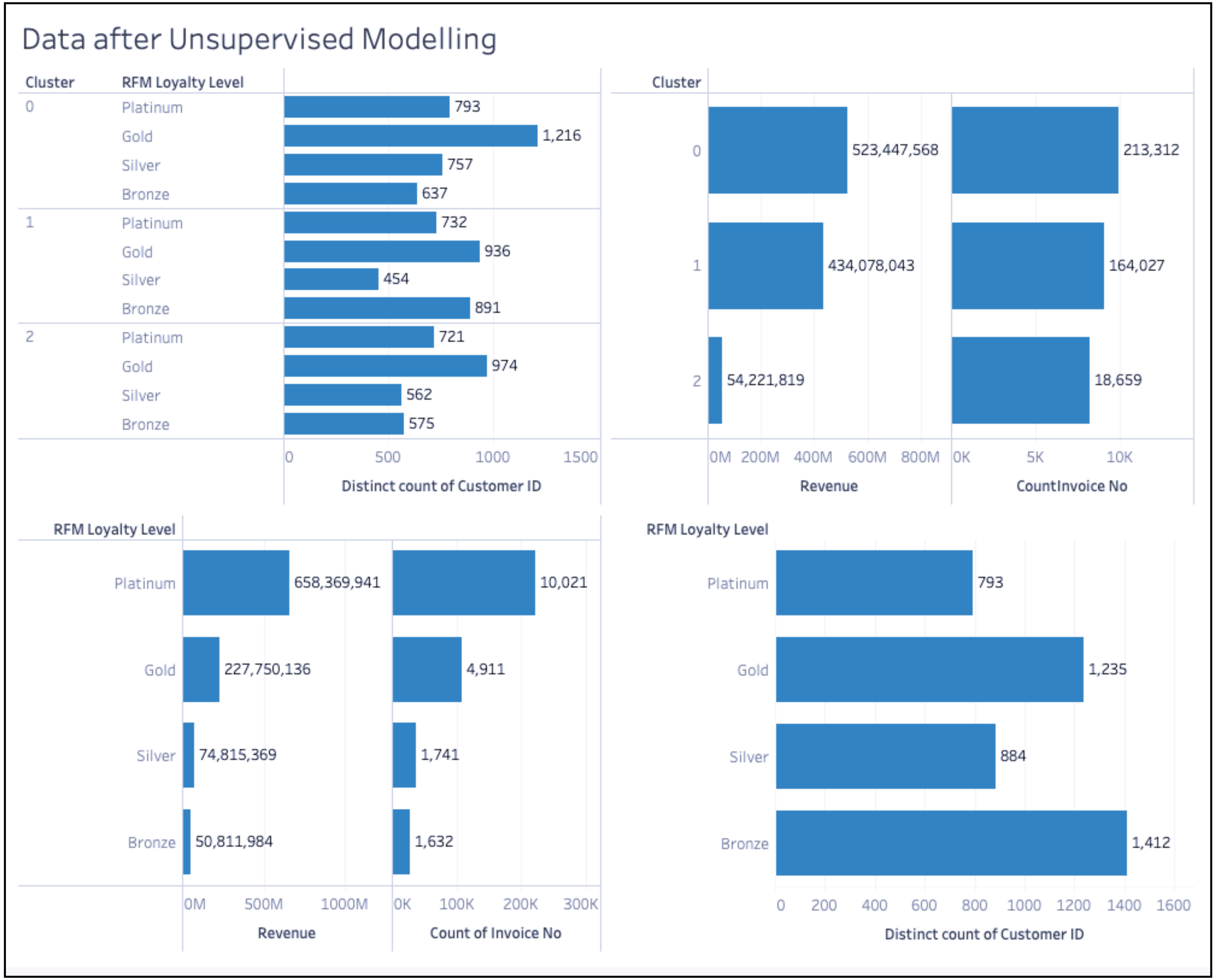
| | CustomerID | Item_Code | InvoiceNo | Date_of_purchase | Revenue | Shipping_Location | Cluster | RFM_Loyalty_Level |
|---|---|---|---|---|---|---|---|---|
| 0 | 4355.0 | 2031 | 398177 | 329 | 1926.0 | 0 | 2 | Silver |
| 1 | 4355.0 | 1239 | 390525 | 285 | 1020.0 | 15 | 0 | Silver |
| 2 | 4355.0 | 1311 | 390525 | 278 | 1032.0 | 15 | 0 | Silver |
| 3 | 4355.0 | 1312 | 390525 | 277 | 1392.0 | 15 | 0 | Silver |
| 4 | 4355.0 | 2018 | 390525 | 278 | 1491.0 | 15 | 0 | Silver |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 395993 | 5.0 | 4 | 368101 | 66 | 3168.0 | 15 | 1 | Bronze |
| 395994 | 5.0 | 1303 | 368101 | 64 | 3180.0 | 15 | 1 | Bronze |
| 395995 | 5.0 | 1114 | 368101 | 66 | 3672.0 | 15 | 1 | Bronze |
| 395996 | 5.0 | 762 | 368101 | 66 | 3696.0 | 15 | 1 | Bronze |
| 395997 | 3244.0 | 2568 | 369050 | 74 | 501.0 | 17 | 1 | Bronze |

395998 rows × 8 columns

➤ Below, we can see Unique customers with their loyalty levels in each clusters.

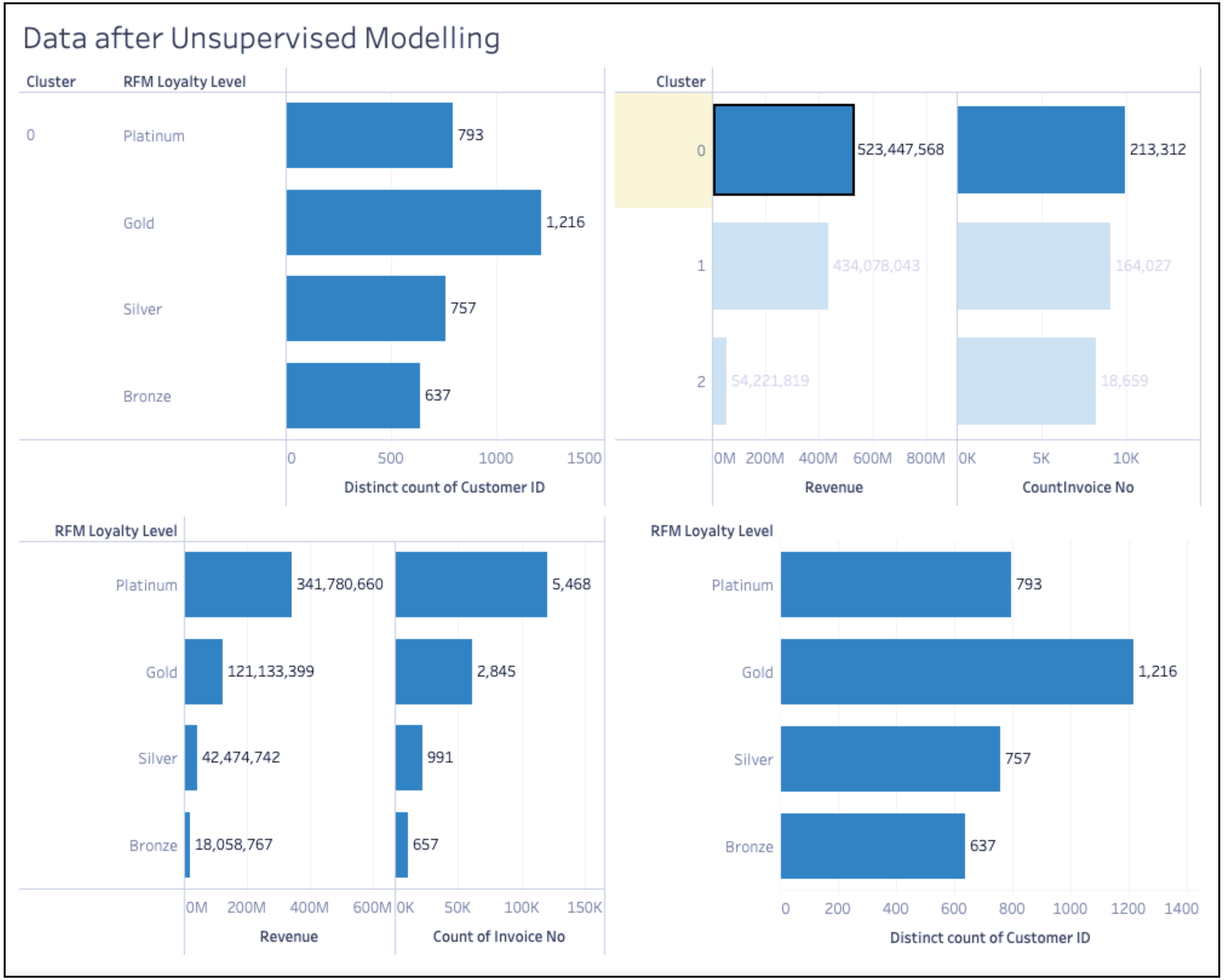
```
In [73]: Clusters_Seg = Final_df.groupby(['Cluster', 'RFM_Loyalty_Level'])['CustomerID'].nunique()
         Clusters_Seg.columns = ['Cluster', 'RFM_Loyalty_Level', 'Unique Customers']
         Clusters_Seg

Out[73]: Cluster  RFM_Loyalty_Level
         0        Bronze                637
                  Silver                757
                  Gold                 1216
                  Platinum              793
         1        Bronze                891
                  Silver                454
                  Gold                  936
                  Platinum              732
         2        Bronze                575
                  Silver                562
                  Gold                  974
                  Platinum              721
         Name: CustomerID, dtype: int64
```




INSIGHTS AFTER CLUSTERING…


INSIGHTS AFTER CLUSTERING…



Data after Unsupervised Modelling
Cluster
RFM Loyalty Level
0
Platinum
793
Gold
1,216
Silver
757
Bronze
637
0
500
1000
1500
Distinct count of Customer ID
Cluster
0
523,447,568
213,312
1
434,078,043
164,027
2
54,221,819
18,659
0M
200M
400M
600M
800M
Revenue
0K
5K
10K
CountInvoice No
RFM Loyalty Level
Platinum
341,780,660
5,468
Gold
121,133,399
2,845
Silver
42,474,742
991
Bronze
18,058,767
657
0M
200M
400M
600M
Revenue
0K
50K
100K
150K
Count of Invoice No
RFM Loyalty Level
Platinum
793
Gold
1,216
Silver
757
Bronze
637
0
200
400
600
800
1000
1200
1400
Distinct count of Customer ID

# CONCLUSION

After observation, we have concluded from out model that,

1. RFM analysis can help in answering many questions with respect to their customers and this can help companies to make marketing strategies for their customers, retaining their slipping customers and providing recommendations to their customer based on their interest.

2. From RFM segmentation, we got 4324 unique customers into 4 levels :Platinum(793), Gold(884),Silver(1,235) and Bronze(1,412)

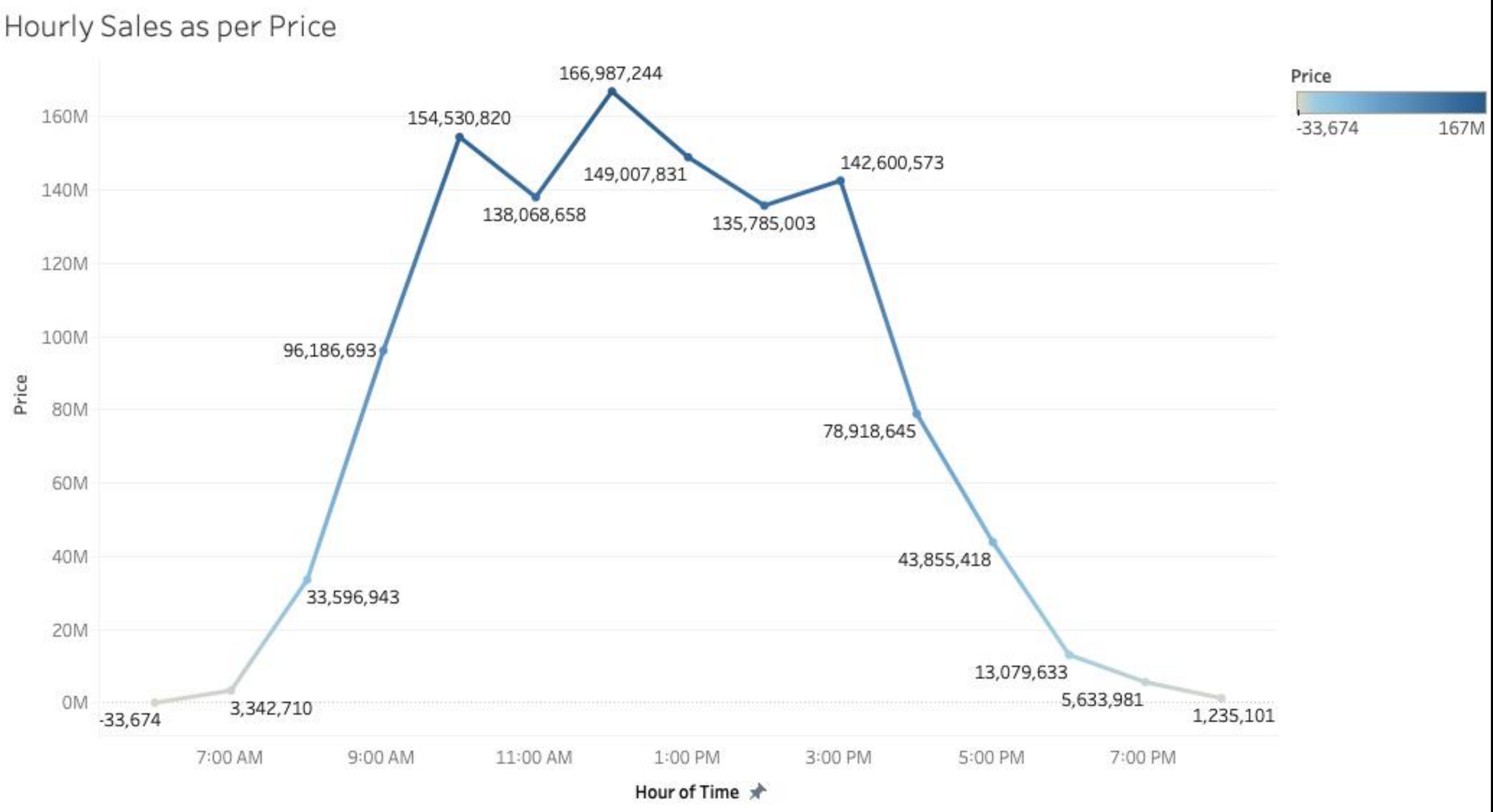
3. After modeling we have got 3 clusters. 0:213312, 1:164027, 2:18659.

4. Customers in Cluster 0 has the maximum revenue generation as well as Maximum number of transactions.

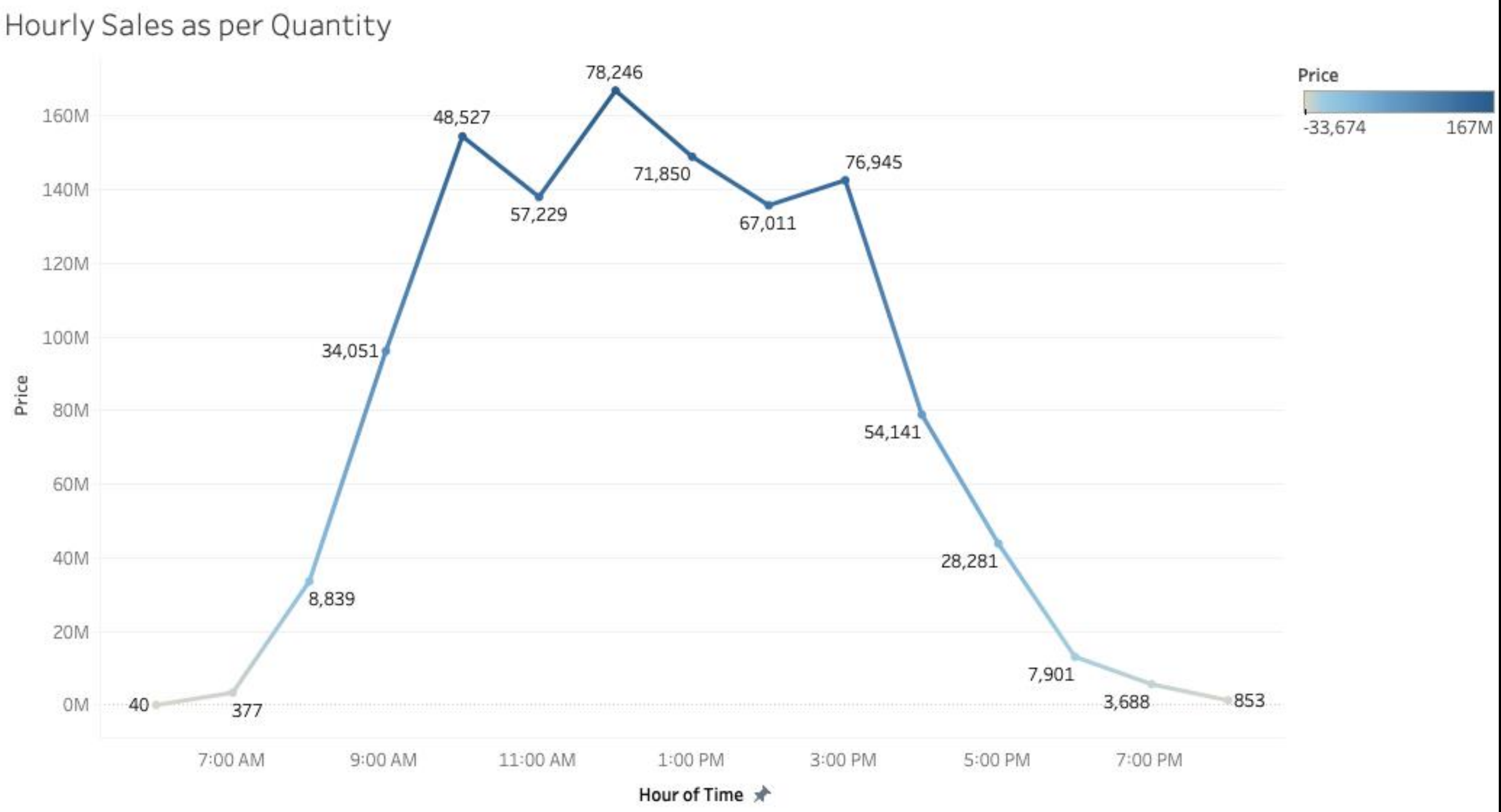
5.Cluster 0 and 1 has the maximum revenue generation as one of the major factor is **Location 36** which is contributing the most in it and it is not included in Cluster 2 hence the revenue is less.

| **CUSTOMER SEGMENT** | **ACTIVITY** | **ACTIONABLE TIP** |
|---|---|---|
| PLATINUM | Bought recently, buy often and spend the most! | Reward them. Can be early adopters for new products. Will promote your brand. |
| GOLD | Spend good money with us often. Responsive to promotions. | Upsell higher value products. Ask for reviews. Engage them. |
| SILVER | Recent shoppers, but haven’t spent much. | Create brand awareness, offer free trials |
| BRONZE | Lowest recency, frequency and monetary scores. | Send personalized emails to reconnect, offer renewals, provide helpful resources. |

# Bonus




At 12pm the sale is Highest.
Between 10am and 3pm there is maximum engagement of the Customers.




At 12pm the sale is Highest.
Between 10am and 3pm there is maximum engagement of the Customers.

# *Business Suggestions/Insights Based on Analysis*

- Company should provide offers and discounts in the months like in the **first quartile** of the year so as to increase sales as these months have the lowest sales.
- Location 24, 34, 31, 7, 10, 8 and 28 have fewer sales as compared to other Locations. The company should look for the **reasons** behind it to boost up the sales.
- 85123A is the maximum sold items. Hence, company should take care of **inventory** of 85123A.
- Item code 14846 is the maximum revenue generated item.
- Company can work on **price—strategy** as per insights.
- Around 10:00 am to 3:00 pm there has been maximum sale.
- And the least is before 10:00 am and after 3:00 pm  so we can focus during these time to offer deals or discounts by advertising. To maximize the likelihood of customers buying the product/s.

# THANK YOU!