Customer Segmentation using RFM Analysis and K-Means Clustering in Python

Omotolani Kehinde · Follow
8 min read · Jan 3, 2023

92 1

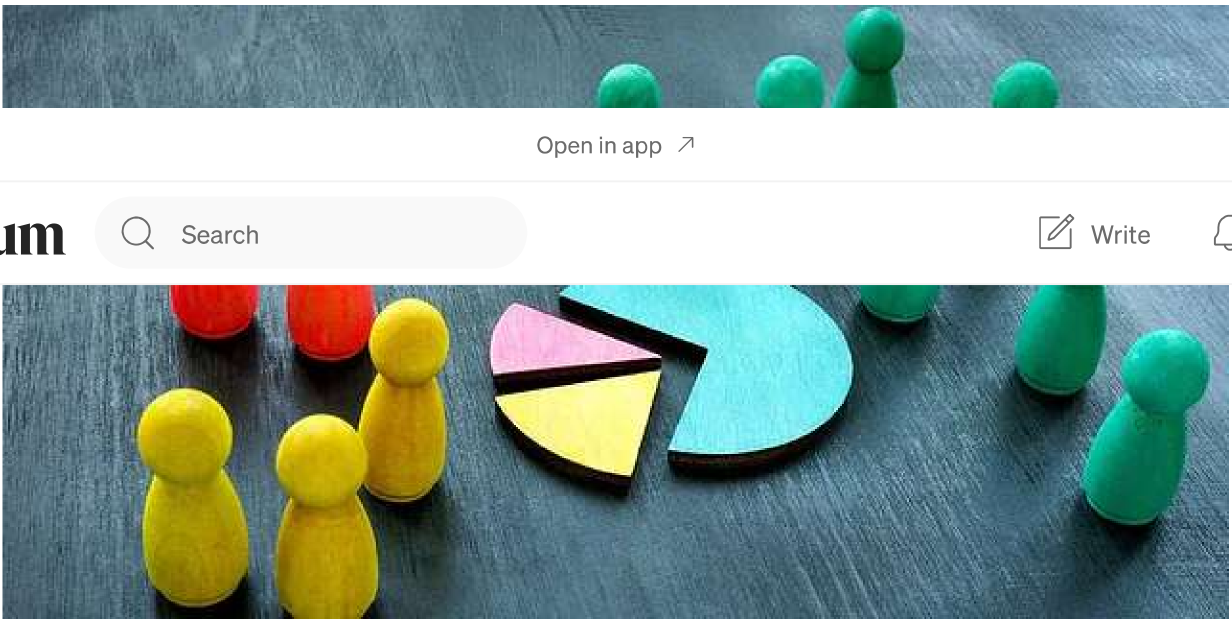


Table of Contents

Customer segmentation is the process of dividing your users or customers into segments using similar characteristics like purchase frequency, the amount spent, demographics, behavioral patterns and so much more.

This allows organizations to develop more targeted sales, retention, and marketing strategies for customer segments.

Customer segmentation can be broken down into two types:

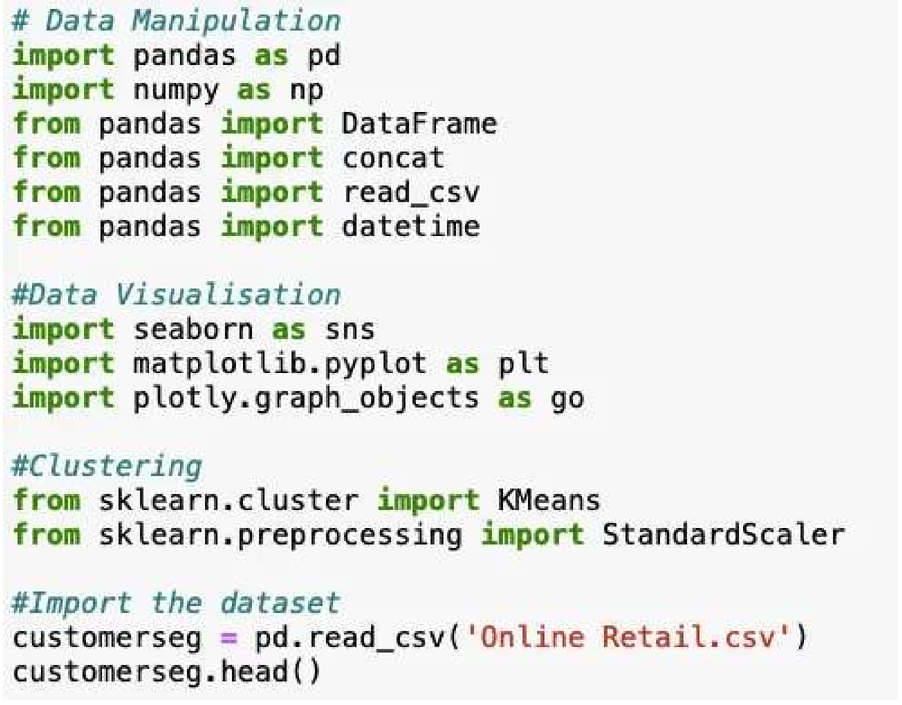
1. Segmenting customers based on their personas: these segments customers based on relevant user data criteria like demographics and geographic-related information.
2. Segmenting customers based on their behavior: these segments customers based on their spending behavior, how often, and what products they buy.

This analysis focuses on segmenting our customers using the behavioral patterns seen in their purchase history. We analyzed the trends seen in each demographic to help understand the market performance of each product in the operating countries but we decided to do a behavioral segmentation because understanding your customers' patterns makes it easier to create a better-personalized experience for each segment. We created a Recency frequency monetary pie chart and cluster pie chart(using k means clustering) for this project.

We got the data from the UCI Machine Learning website. We are working with the Online Retail dataset. The dataset is an e-commerce dataset that contains transactions from December 1st, 2010 until December 9th, 2011 for a UK-based online retail store. You can access the dataset from here.

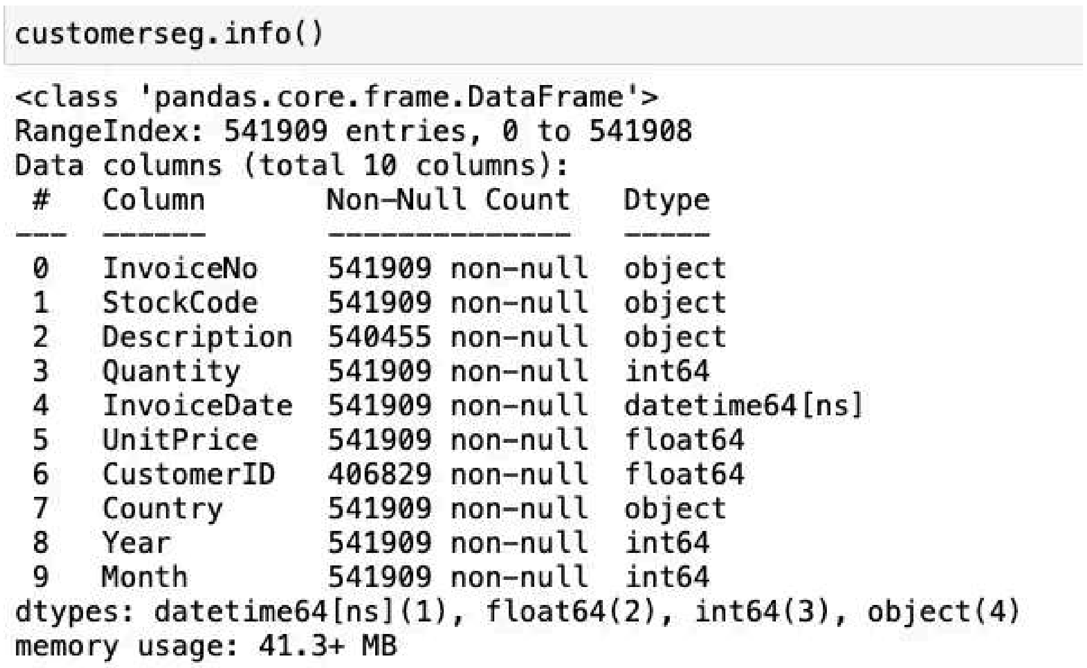
Imports

So we had to bring in a couple of guys to help with data processing, visualizations of our insights, and some other components to help perform some tasks along the way.



```python
# Data Manipulation
import pandas as pd
import numpy as np
from pandas import DataFrame
from pandas import concat
from pandas import read_csv
from pandas import datetime

#Data Visualisation
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.graph_objects as go

#Clustering
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler

#Import the dataset
customerseg = pd.read_csv('Online Retail.csv')
customerseg.head()
```

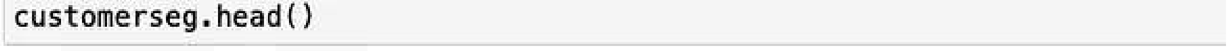
Here is some basic information about our dataset(Month and year were extracted by me)



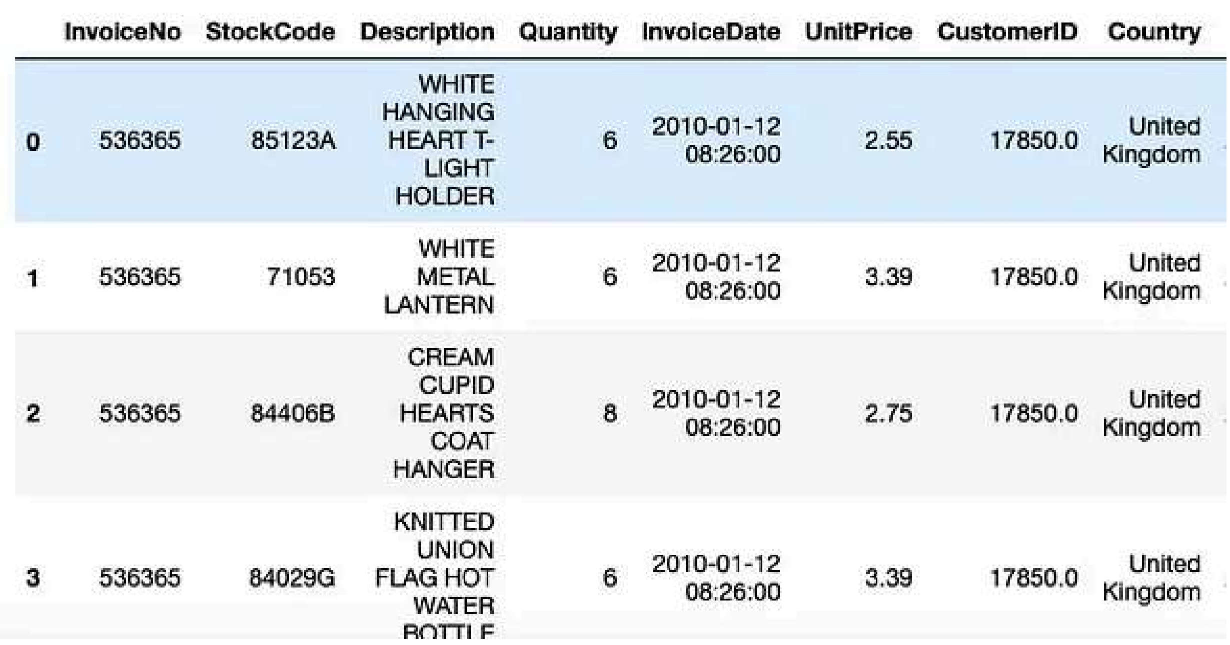
```
customerseg.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 10 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  datetime64[ns]
 5   UnitPrice    541909 non-null  float64       
 6   CustomerID   406829 non-null  float64       
 7   Country      541909 non-null  object        
 8   Year         541909 non-null  int64         
 9   Month        541909 non-null  int64         
dtypes: datetime64[ns](1), float64(2), int64(3), object(4)
memory usage: 41.3+ MB
```

There are 541909 observations for 8 predictors.



```
customerseg.head()
```

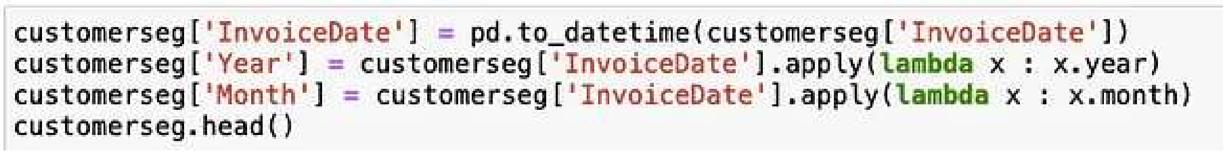


	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-01-12 08:26:00	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-01-12 08:26:00	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-01-12 08:26:00	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-01-12 08:26:00	3.39	17850.0	United Kingdom

. . .

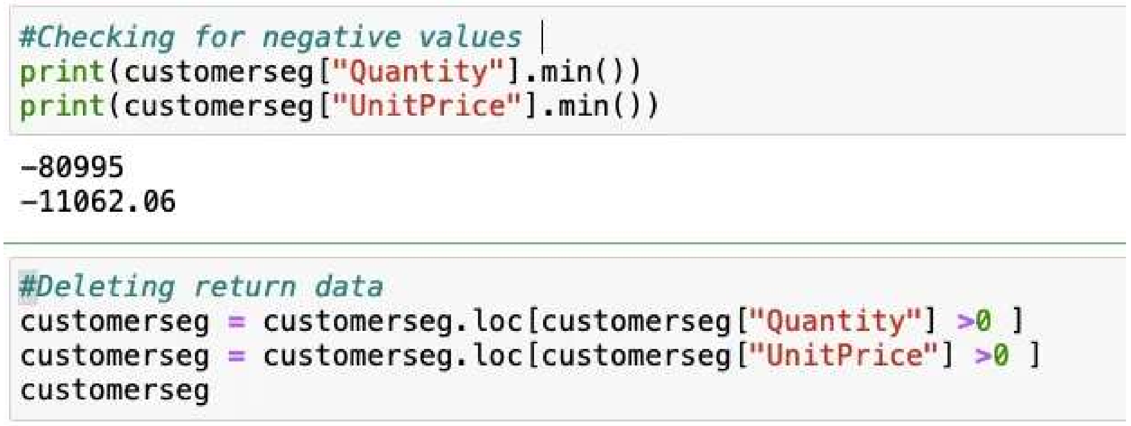
Data Preparation and Cleaning

- **Converting InvoiceDate to the date format and extracting years and months**



```
customerseg['InvoiceDate'] = pd.to_datetime(customerseg['InvoiceDate'])
customerseg['Year'] = customerseg['InvoiceDate'].apply(lambda x : x.year)
customerseg['Month'] = customerseg['InvoiceDate'].apply(lambda x : x.month)
customerseg.head()
```

- **Checking and removing negative values in our dataset:-** it can be seen that there are orders with negative quantity and unit price — most likely returns. While returns can be analyzed to gather some insights and trends, this study won't be exploring that.



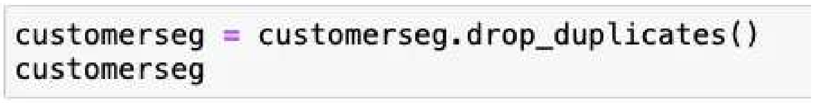
```
#Checking for negative values
print(customerseg["Quantity"].min())
print(customerseg["UnitPrice"].min())
```

```
-80995
-11062.06
```

```
#Deleting return data
customerseg = customerseg.loc[customerseg["Quantity"] >0 ]
customerseg = customerseg.loc[customerseg["UnitPrice"] >0 ]
customerseg
```

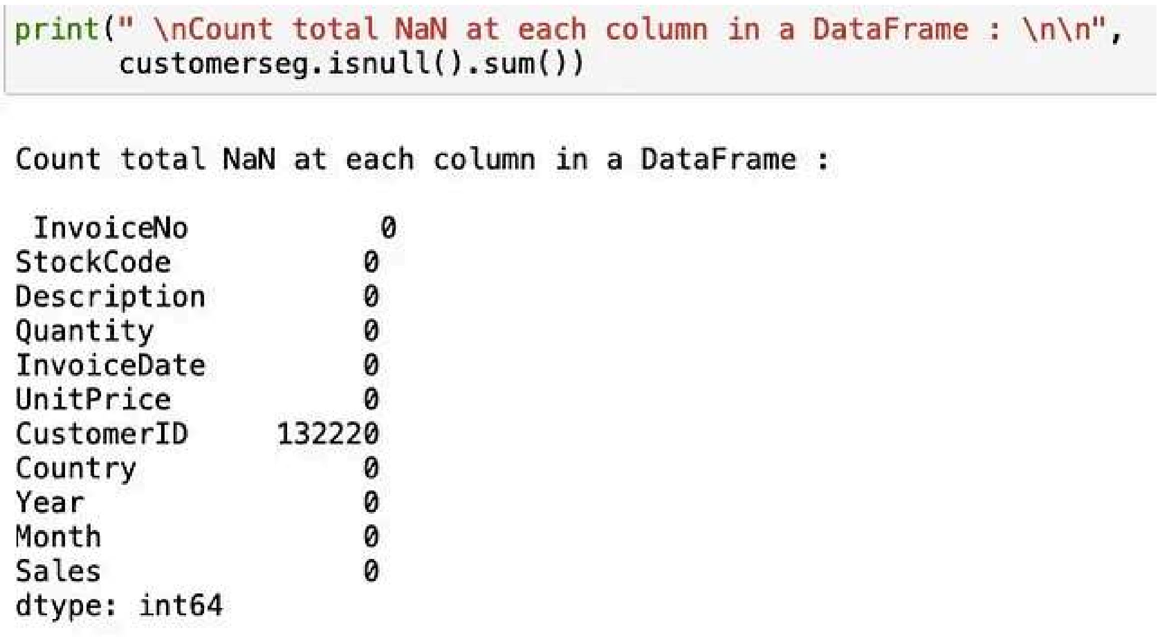
- **Dropping Duplicates**

We must drop repeated information to avoid contaminating our dataset.



```
customerseg = customerseg.drop_duplicates()
customerseg
```

- **Missing Values**



```
print(" \nCount total NaN at each column in a DataFrame : \n\n",
      customerseg.isnull().sum())
```

```
Count total NaN at each column in a DataFrame :

  InvoiceNo           0
 StockCode            0
 Description          0
 Quantity             0
 InvoiceDate          0
 UnitPrice            0
 CustomerID      132220
 Country              0
 Year                 0
 Month                0
 Sales                0
 dtype: int64
```

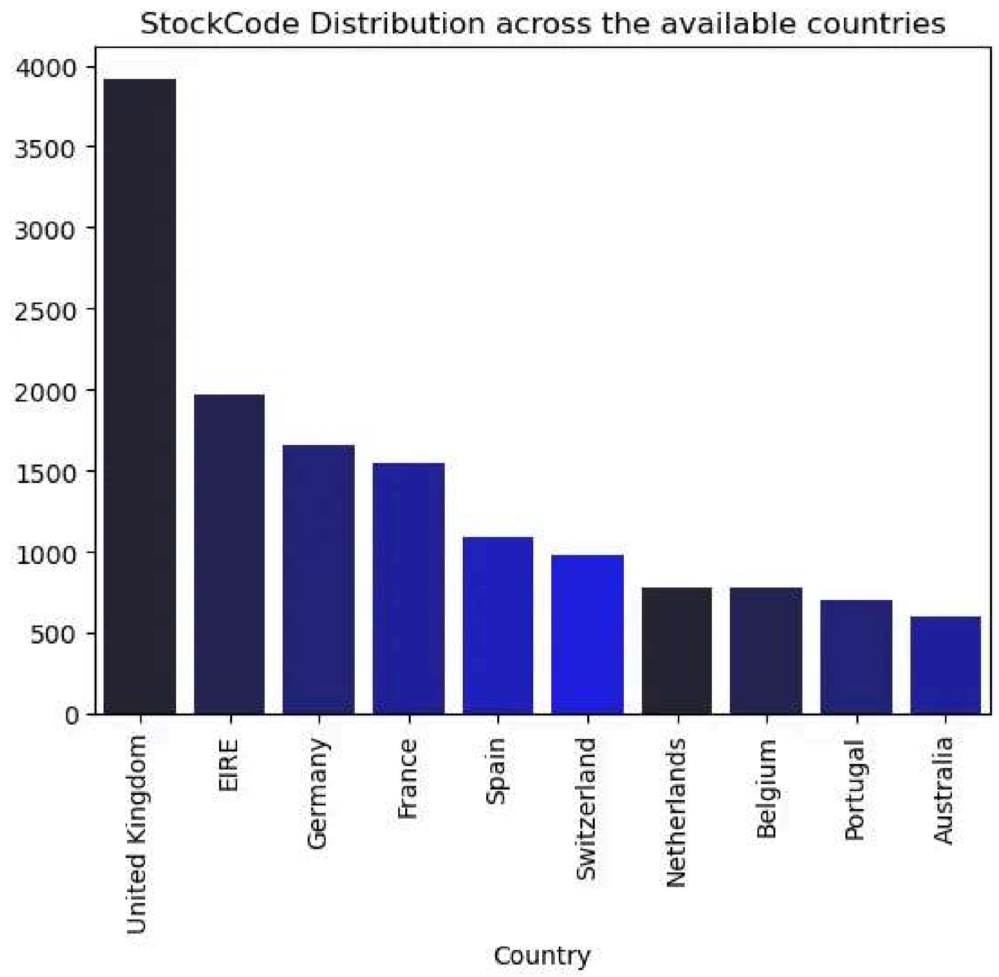
We had a total of 135080 missing Customer IDs but I decided not to drop this because this information is not completely missing, while we have the IDs missing we have all the major sales data for each transaction and these sales still translated to profit for the online retail store and it can help our analysis.

. . .

Exploratory Data Analysis

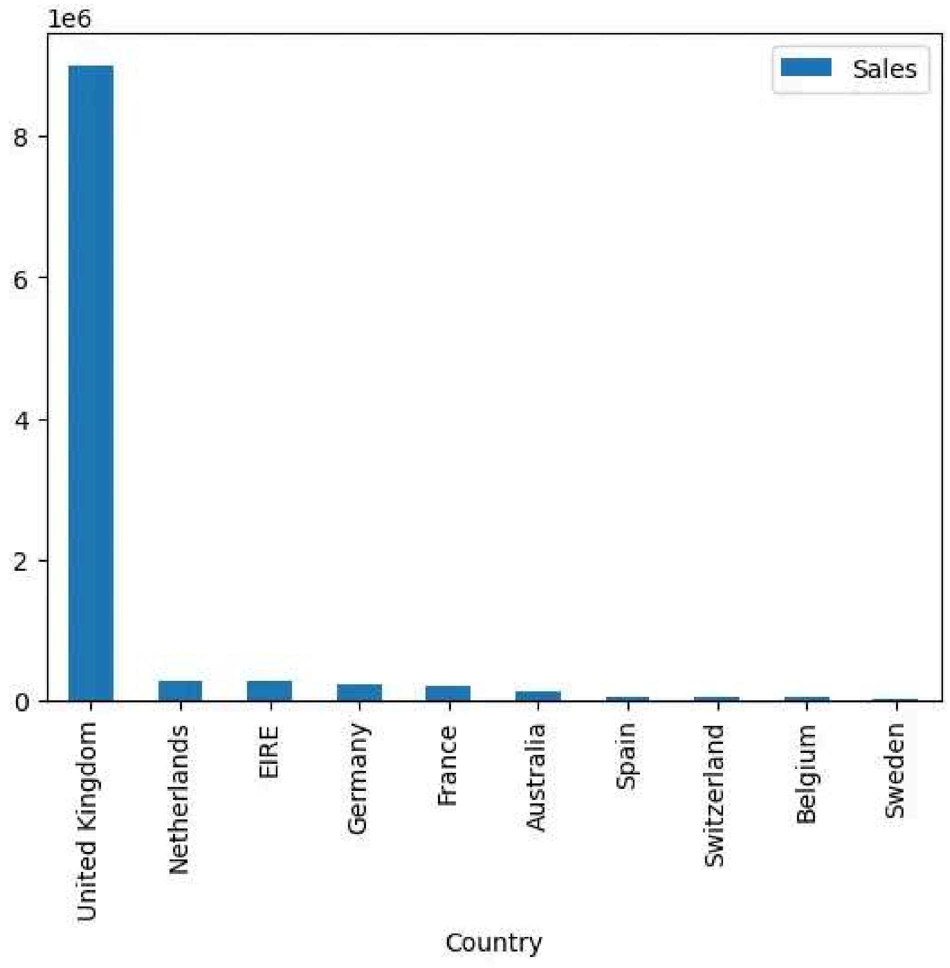
- **Stock Code Distribution across the available countries**

Stock Code is the Product(item) code uniquely assigned to each distinct product. This analysis gave us a general idea of the total amount of unique products ordered per country.



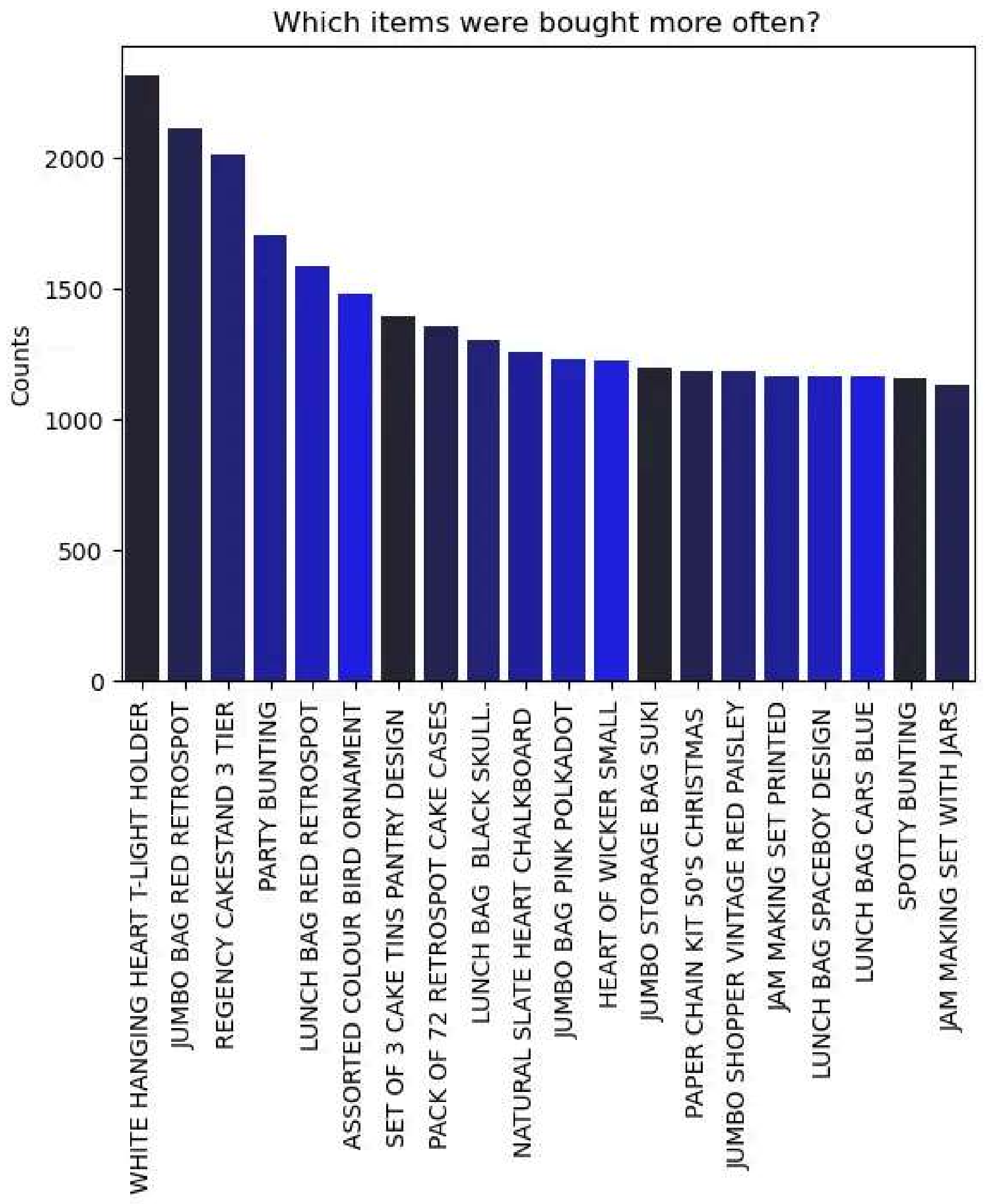
- **Top 10 Countries with the highest sales.**

Our data shows that the majority of the sales are made from the United Kingdom which was a total of 9,001,744.094 for 12 months



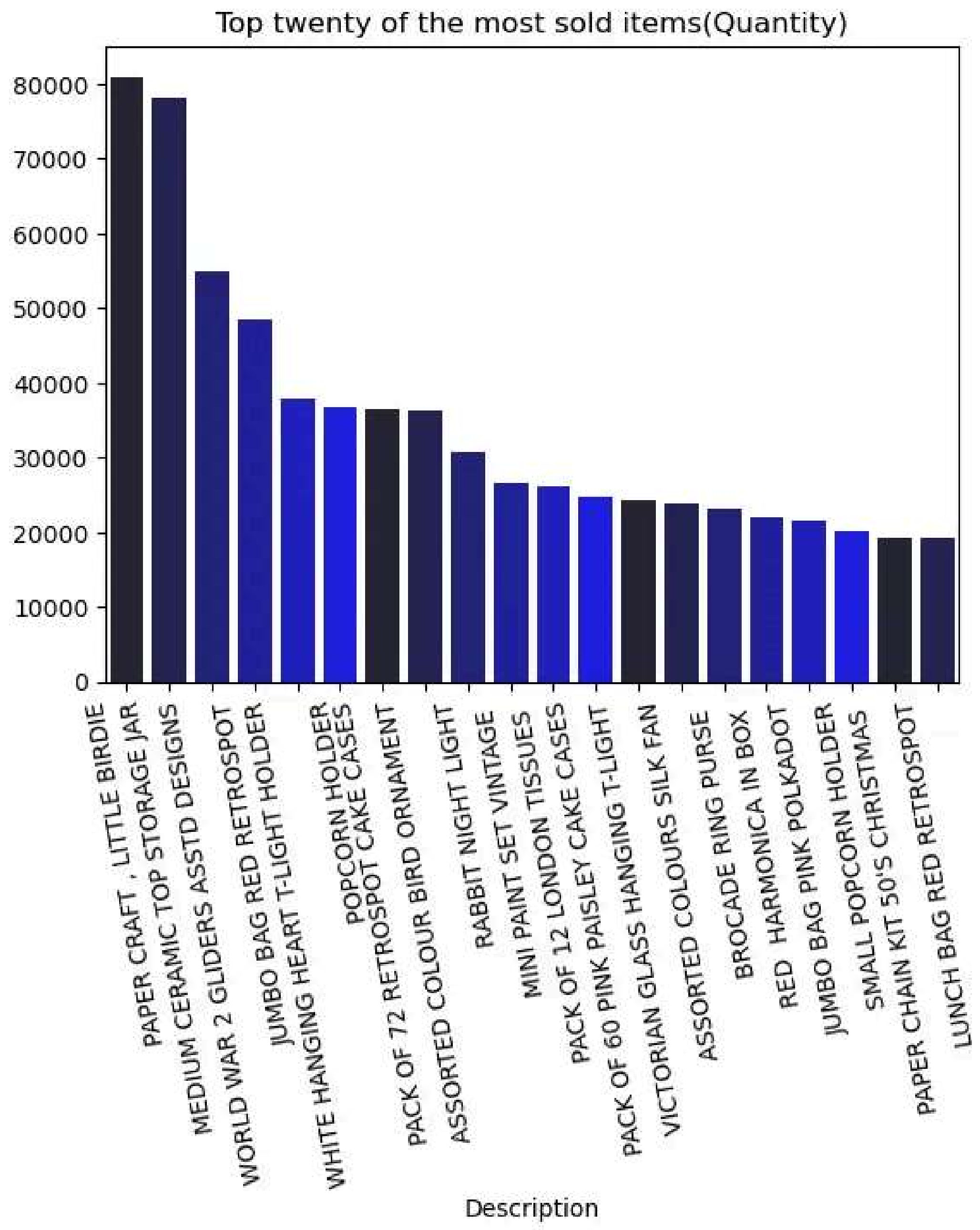
- **Which Item was purchased more often?**

From the results, we observe that customers ordered WHITE HANGING HEART T-LIGHT HOLDER 2028 times. This shows the order counts not the quantity or total amount purchased.

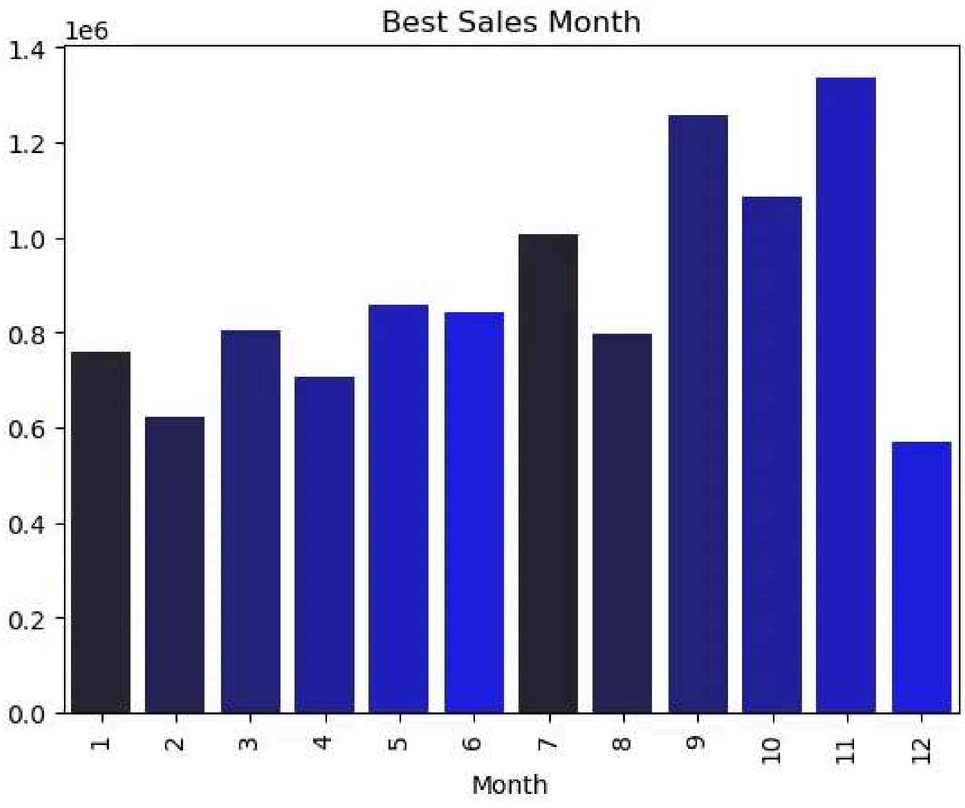


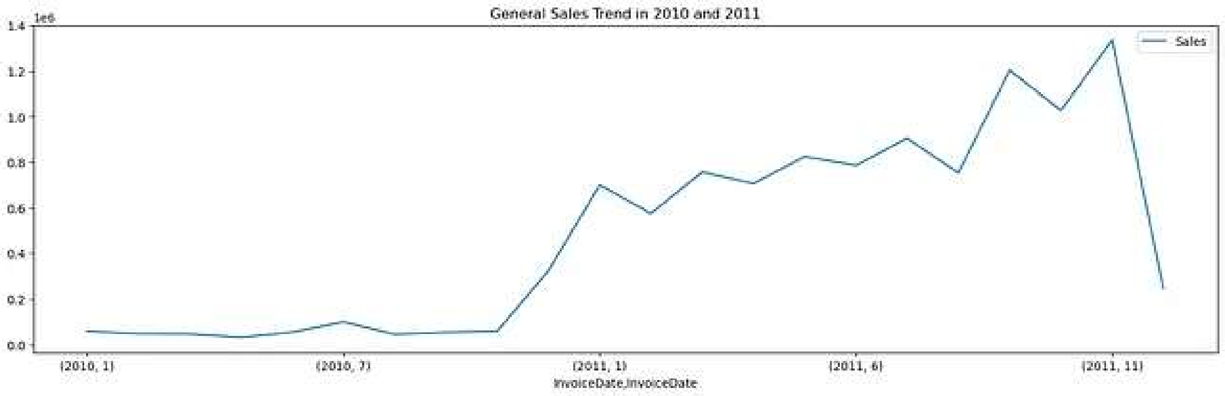
- **Top twenty of the most sold items(In terms of quantity)**

While WHITE HANGING HEART T-LIGHT HOLDER was purchased more often, The online retail store sold more Paper Craft Little Birdie than any other product.



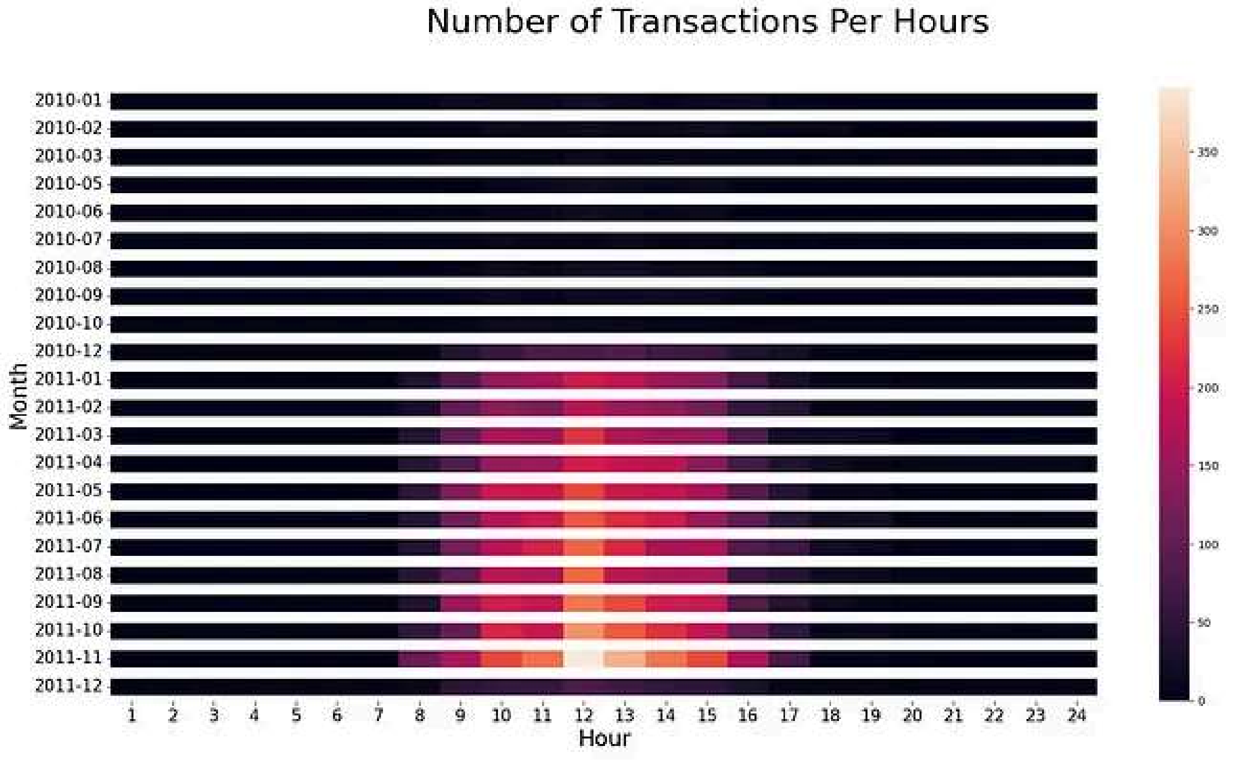
- **What month did we have our best sales?**





From our results, we see that the company made the highest sales in November 2011.

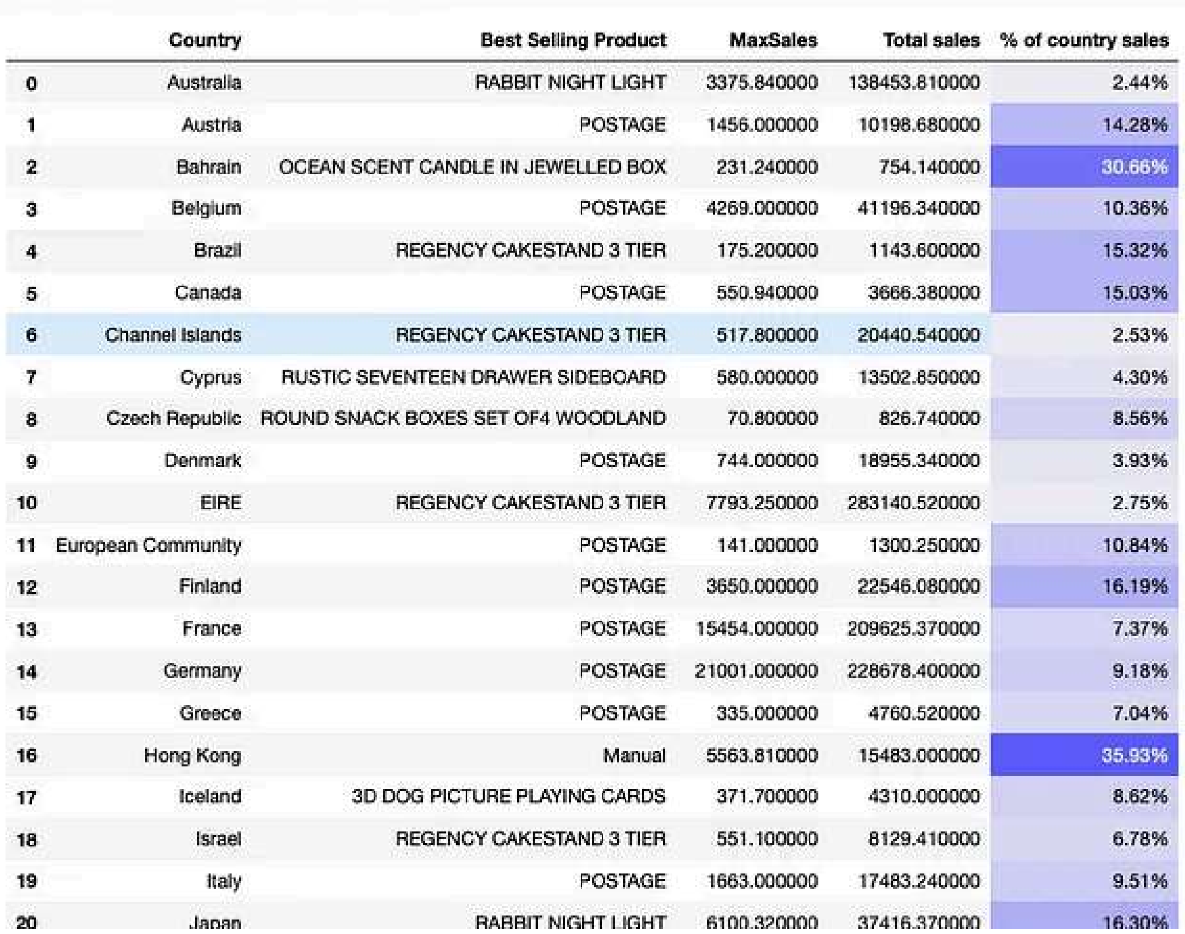
- **When do customers tend to purchase products?**



There are no orders between the hours of 19:0pm and 6:00 am on the online retail store. We can see that 12:o0pm is the most active hour for the online retail store. We have the highest amount of orders between the hours of 11 am to 2 pm each day.

- **Best Selling Product for each Country**

This shows the best-selling product for each country and its percentage contribution to the total amount of sales for the country.



	Country	Best Selling Product	MaxSales	Total sales	% of country sales
0	Australia	RABBIT NIGHT LIGHT	3375.840000	138453.810000	2.44%
1	Austria	POSTAGE	1456.000000	10198.680000	14.28%
2	Bahrain	OCEAN SCENT CANDLE IN JEWELLED BOX	231.240000	754.140000	30.66%
3	Belgium	POSTAGE	4269.000000	41196.340000	10.36%
4	Brazil	REGENCY CAKESTAND 3 TIER	175.200000	1143.600000	15.32%
5	Canada	POSTAGE	550.940000	3666.380000	15.03%
6	Channel Islands	REGENCY CAKESTAND 3 TIER	517.800000	20440.540000	2.53%
7	Cyprus	RUSTIC SEVENTEEN DRAWER SIDEBOARD	580.000000	13502.850000	4.30%
8	Czech Republic	ROUND SNACK BOXES SET OF4 WOODLAND	70.800000	826.740000	8.56%
9	Denmark	POSTAGE	744.000000	18955.340000	3.93%
10	EIRE	REGENCY CAKESTAND 3 TIER	7793.250000	283140.520000	2.75%
11	European Community	POSTAGE	141.000000	1300.250000	10.84%
12	Finland	POSTAGE	3650.000000	22546.080000	16.19%
13	France	POSTAGE	15454.000000	209625.370000	7.37%
14	Germany	POSTAGE	21001.000000	228678.400000	9.18%
15	Greece	POSTAGE	335.000000	4760.520000	7.04%
16	Hong Kong	Manual	5563.810000	15483.000000	35.93%
17	Iceland	3D DOG PICTURE PLAYING CARDS	371.700000	4310.000000	8.62%
18	Israel	REGENCY CAKESTAND 3 TIER	551.100000	8129.410000	6.78%
19	Italy	POSTAGE	1663.000000	17483.240000	9.51%
20	Japan	RABBIT NIGHT LIGHT	6100.320000	37416.370000	16.30%

. . .

Recency Frequency Monetary (RFM) Analysis

RFM (Recency, Frequency, Monetary) analysis is a customer segmentation technique that uses past purchase behavior to divide customers into groups.

RECENCY (R): Days since last purchase

FREQUENCY (F): Total number of purchases

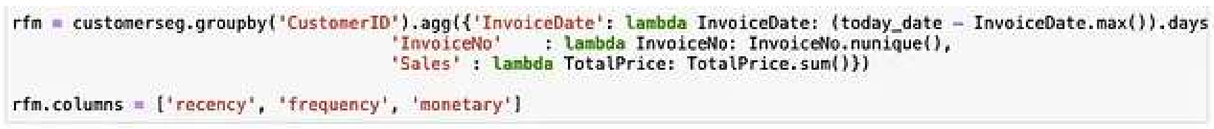
MONETARY VALUE (M): Total money this customer spent.

These RFM metrics are important indicators of a customer's behavior because the frequency and monetary value affect a customer's lifetime value, and recency affects retention, a measure of engagement.

The RFM value was assigned to each customer as follows:

- Recency = Number of days the customer's latest invoice date compared to the latest invoice date among all customers.

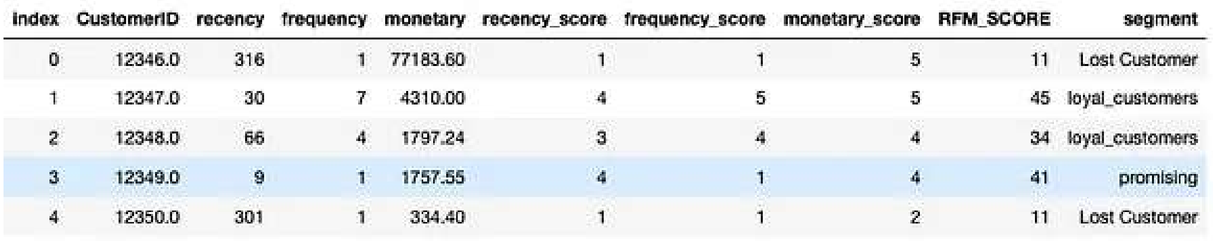
- Frequency = Number of transactions of the customers.
- Monetary=Spend of the customers.



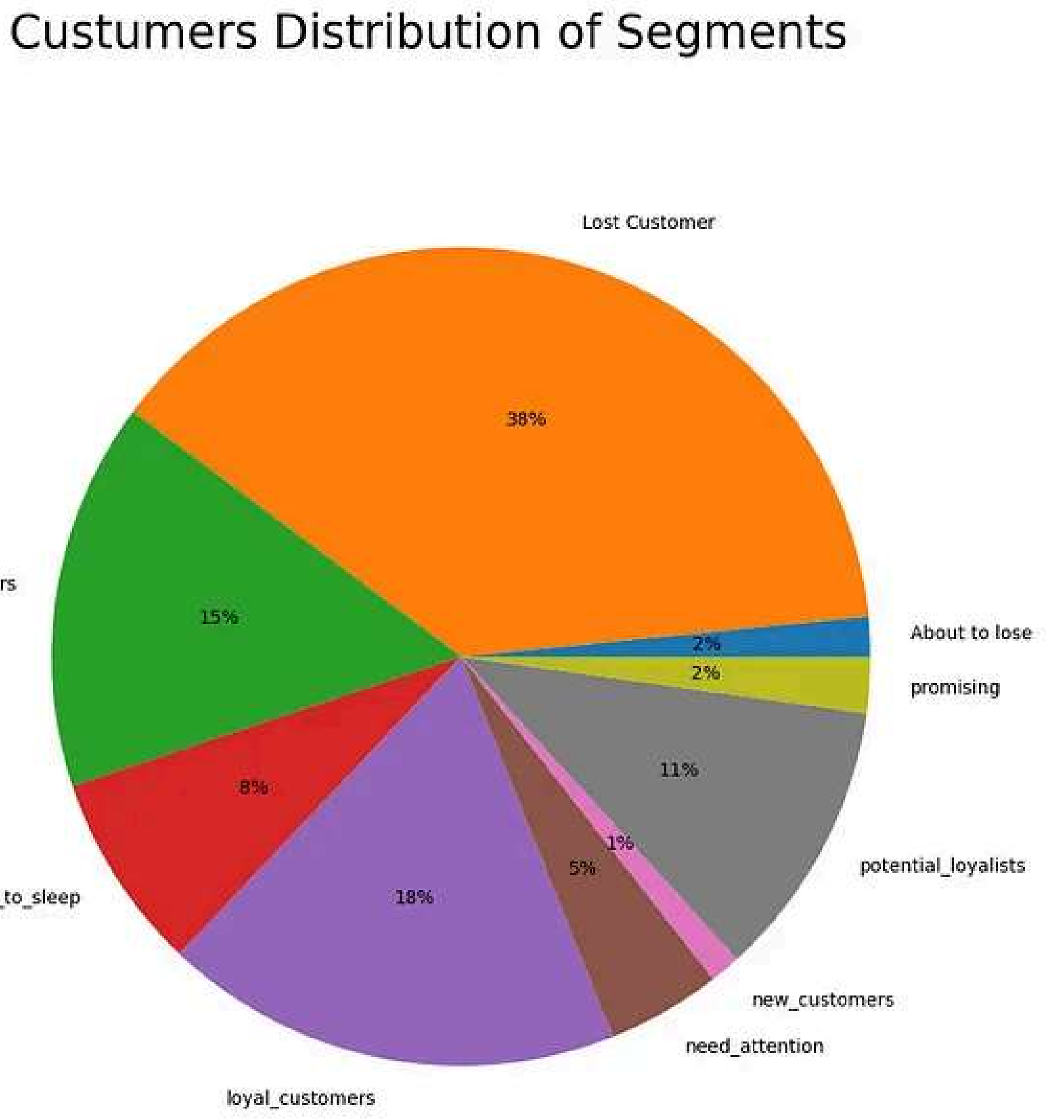
```
rfm = customerseg.groupby('CustomerID').agg({'InvoiceDate': lambda InvoiceDate: (today_date - InvoiceDate.max()).days
                                             'InvoiceNo'    : lambda InvoiceNo: InvoiceNo.nunique(),
                                             'Sales' : lambda TotalPrice: TotalPrice.sum()})

rfm.columns = ['recency', 'frequency', 'monetary']
```

After that, scoring was assigned to each metric using qcut(Quantile-based discretization function) where q was equal to 5. This helps create an unbiased segmentation by letting pandas figure out how to divide up the data based on the distribution of the data.

Higher frequency and monetary will have higher scores while lower recency will have higher scores (recent purchase). After that, an RFM score will be determined by combining the score of all RF(Recency and Frequency) metrics for each customer after which we grouped each RFM score into segments for better storytelling.



index	CustomerID	recency	frequency	monetary	recency_score	frequency_score	monetary_score	RFM_SCORE	segment
0	12346.0	316	1	77183.60	1	1	5	11	Lost Customer
1	12347.0	30	7	4310.00	4	5	5	45	loyal_customers
2	12348.0	66	4	1797.24	3	4	4	34	loyal_customers
3	12349.0	9	1	1757.55	4	1	4	41	promising
4	12350.0	301	1	334.40	1	1	2	11	Lost Customer



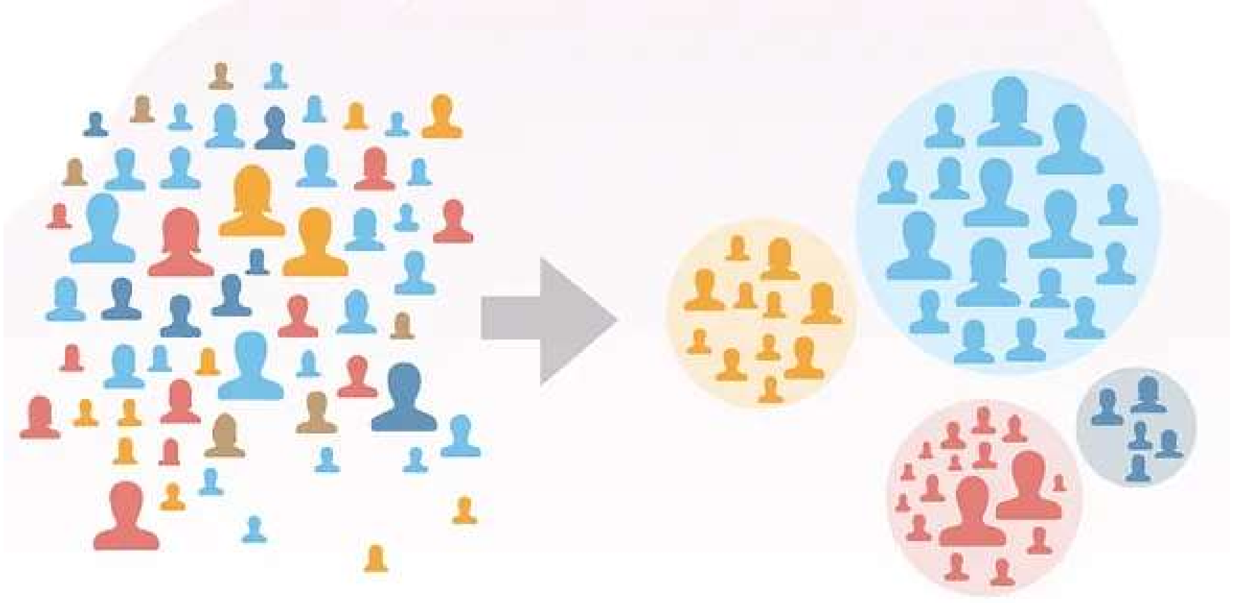
We have 15% of customers considered as Retained Customers. We can put a lot of effort into improving the experience of these customers since they account for a large portion of our revenue.

38% of customers here are considered Lost customers due to an extremely low RFM Score. We can also put in the effort to bring a percentage of this group back on board with us.

In summary, this result can help the product or marketing team design strategies to fit each customer segment.

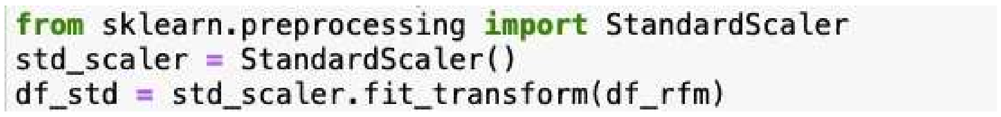
. . .

K-Means Clustering



In addition to segmentation by RFM analysis, K-means clustering can also be used to understand customer segmentation. Standardization is useful when your data has varying scales and the algorithm you are using does make assumptions about your data having a Gaussian distribution.

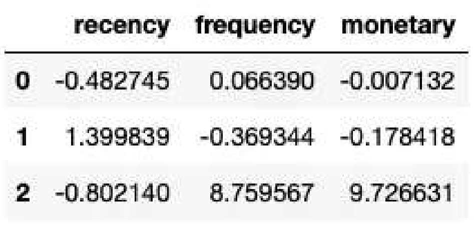
K-Means clustering is greatly influenced by the scale of the data, it assumes that each cluster adheres to a unimodal distribution, such as Gaussian and our RFM score has varying scales so we decided to standardize the data before clustering.



```
from sklearn.preprocessing import StandardScaler
std_scaler = StandardScaler()
df_std = std_scaler.fit_transform(df_rfm)
```

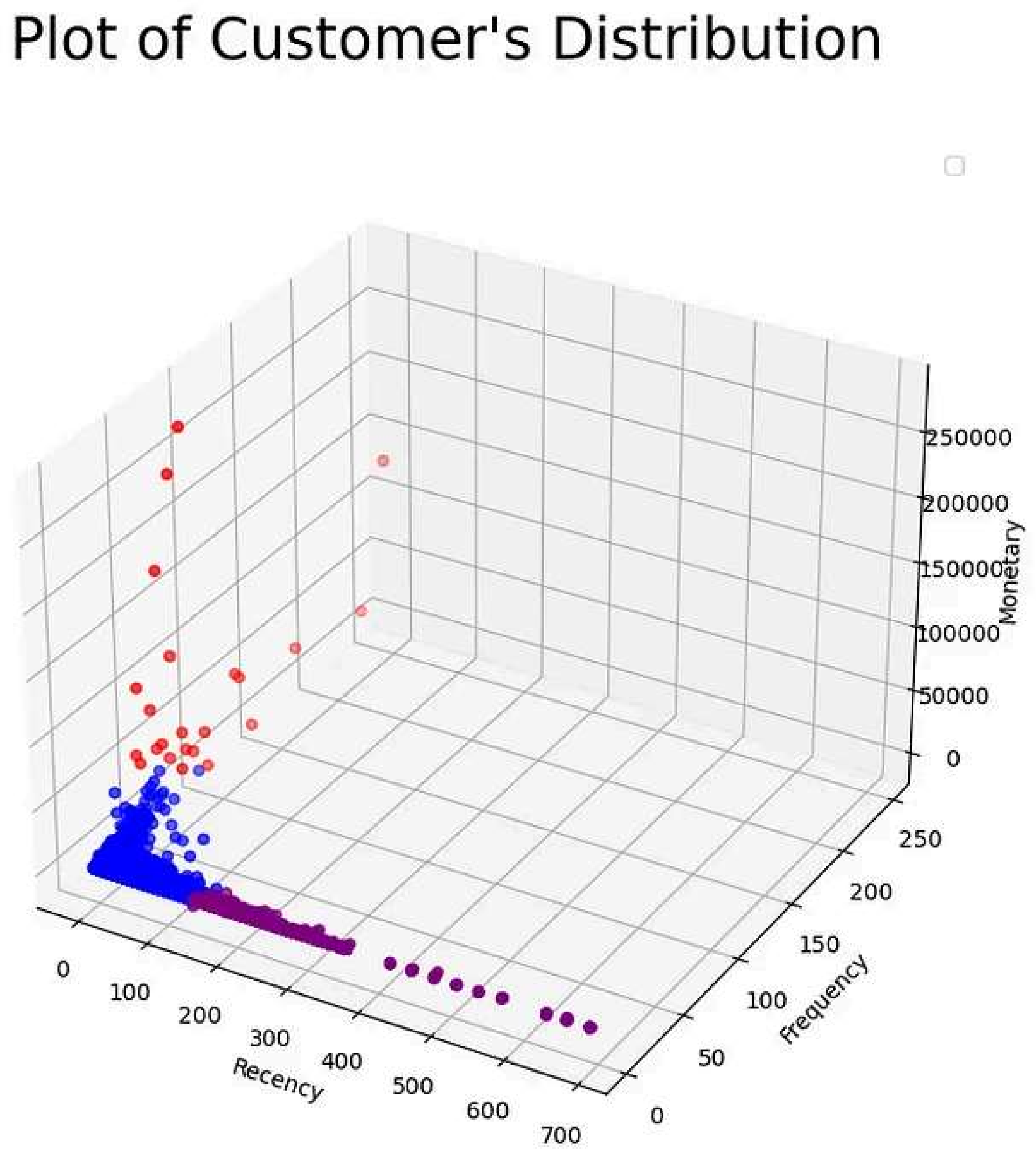
From the KMeans clustering, We can sort every customer into 3 different clusters based on the similarity seen in their RFM results and calculate the mean of each output.

The final output should look like this:

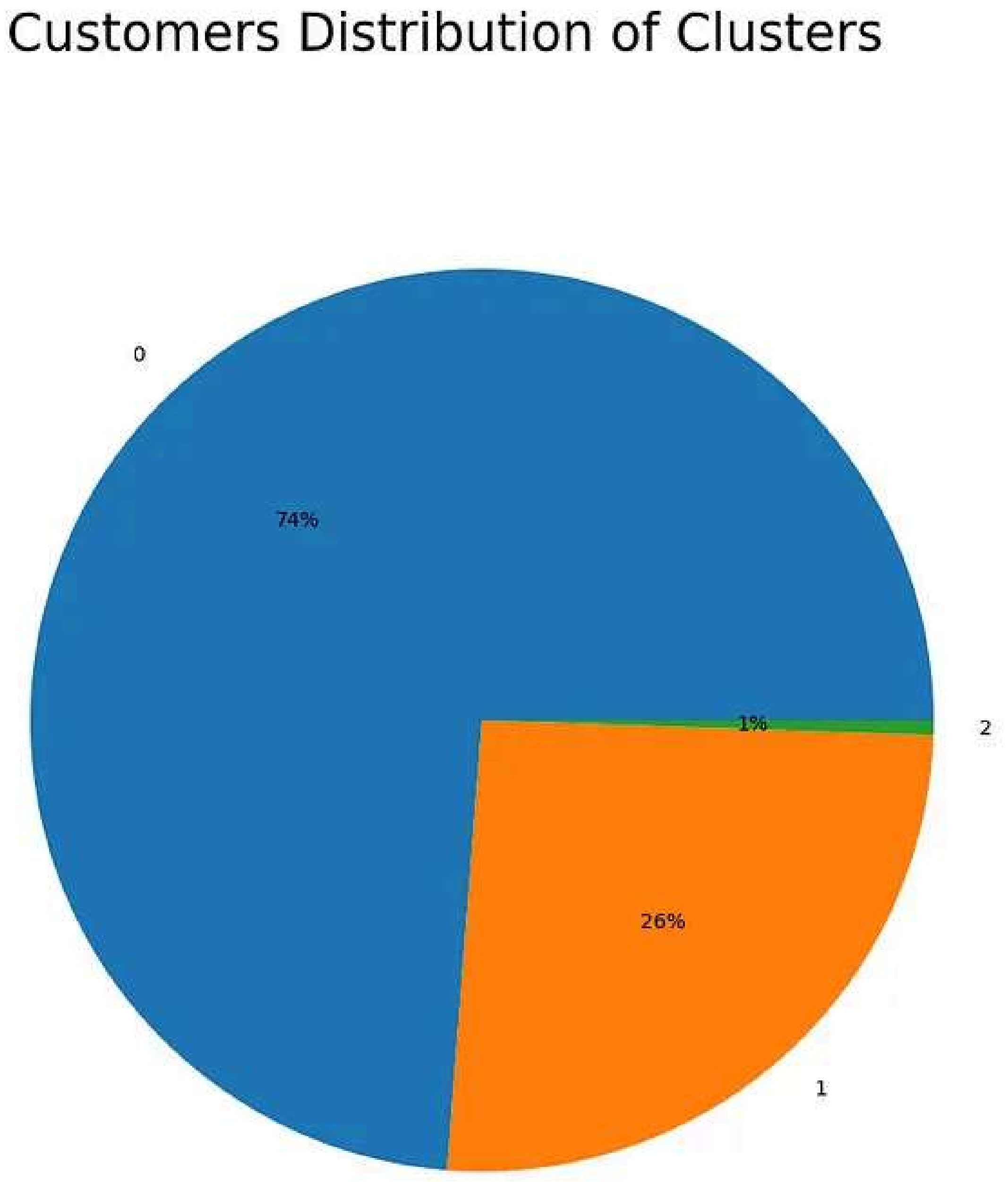


	recency	frequency	monetary
0	-0.482745	0.066390	-0.007132
1	1.399839	-0.369344	-0.178418
2	-0.802140	8.759567	9.726631

Besides the output, we can analyze the segments using a 3D scatter plot. This gives us a good visualization of how the cluster differs from each other and how they were segmented using the RFM metrics.



We can also create a pie chart to calculate and assign a percentage to each cluster. This helps the necessary team understand the store's sales data



In summary, **Cluster 0**(74% of our customers) is most likely a cluster of new customers. They are customers with very low recency and a good level of frequency, which means they have been active recently and it would be a great idea to design retention strategies to ensure they stay active.

Cluster 1(26% are the customers) is most likely a cluster of churned customers. Our analysis shows that they buy at the lowest frequency, spend the least money, and have not purchased anything in a long time.

Cluster 2(1% of our customers) is most likely our exceptional customers. They are the most active set of customers, they buy at the highest frequency, and spend the most money.

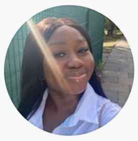
Conclusion

We explored the dataset to make sure we get a detailed understanding of our customer personas before trying to segment these customers. This is important because it helps you make data-driven decisions throughout your implementation process. We segmented our customers into 9 groups in our RFA analysis and 3 clusters in our clustering analysis. This type of analysis helps marketing and product teams to design more efficient and personalized strategies while saving cost and money.

You can find the jupyter notebook on my GitHub.

Thank you for reading!

Customer Segmentation K Means Clustering Rfm Analysis
Exploratory Data Analysis Python



Written by Omotolani Kehinde

Follow

50 Followers

Data Scientist

More from Omotolani Kehinde