Introduction

Customer Segmentation:

Customer segmentation is a method of dividing customers into groups or clusters based on common characteristics. The market researcher can segment customers into the B2C model using various customer's demographic characteristics such as occupation, gender, age, location, and marital status. Psychographic characteristics such as social class, lifestyle and personality characteristics and behavioral characteristics such as spending, consumption habits, product/service usage, and previously purchased products. In the B2B model using various company's characteristics such as the size of the company, type of industry, and location.

In this notebook I have done customer segmentation in two ways

1. Using K-means: To understand the customers like who can be easily converged [target customers] so that it can be passed on to the marketing team and plan the strategy accordingly.
2. We segment customers using RFM (Recency, Frequency, Monetary)

RFM (Recency, Frequency, Monetary) analysis is a behavior-based approach grouping customers into segments. It groups the customers based on their previous purchase transactions. How recently, how often, and how much did a customer buy. RFM filters customers into various groups for the purpose of better service. It helps managers to identify potential customers to do more profitable business.

Data acquisition and cleaning:

There are 2 datasets that were used to understand the customer segmentation process and uses. Both the datasets are attached in the repository for public use.

The first dataset is based of customers who spend in the mall. You are owing a supermarket mall and through membership cards, you have some basic data about your customers like Customer ID, age, gender, annual income and spending score. Spending Score is something you assign to the customer based on your defined parameters like customer behavior and purchasing data.

Problem Statement You own the mall and want to understand the customers like who can be easily converge [Target Customers] so that the sense can be given to marketing team and plan the strategy accordingly.

The second dataset was obtained online and it is online e-commerce transactional data attached in the repository.

The consists of the following features:

- 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: Invice 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.

Methodology:

Mall Customers:

All the relevant libraries that are needed for this project were imported

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

```

Regular data checks were made just to make sure the data is free of null values and does not have any false formatting of data or wrong that would lead to a wrong or bad model. After cleaning the data the features were selected from mall data with which the model was created to understand the behavior of the customer and how it can be clustered using KMeans algorithm so that it can be passed on to the marketing team to become more profitable.

For K means algorithm it is necessary to understand the data and decide an appropriate number of clusters hence elbow chart was used to accomplish this as shown below

```
from sklearn.cluster import KMeans
```

Building the model! Using KMeans algorithm to decide the optimum cluster numbers using an elbow method. Elbow method is plotted wcss vs number of cluster. WCSS: Within cluster sum of squares

The breaking point of elbow from which the error becomes almost consistent we choose that as the number of clusters.

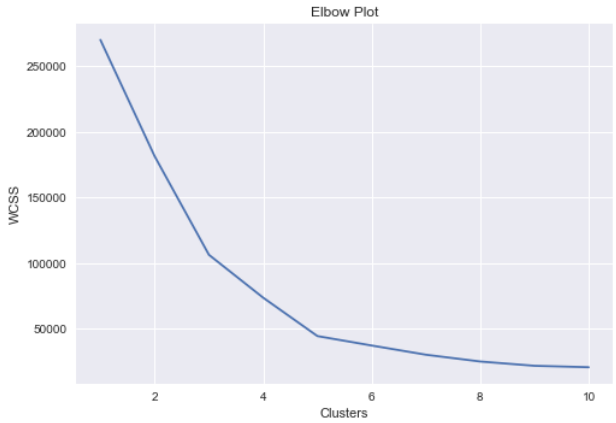
```
wcss=[] #creating an empty list to store wcss from each cluster
```

```
for i in range(1,11):
    kmeans = KMeans(n_clusters=i, random_state=0)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)

#inertia_ is the formula used to segregate the data points into clusters
```

```
#plotting the elbow graph

plt.plot(range(1,11),wcss)
plt.title("Elbow Plot")
plt.xlabel("Clusters")
plt.ylabel("WCSS")
plt.show()
```



It can be observed that the cluster's error becomes constant after k=5! Hence, k=5 will be appropriate for the k means algorithm for this data.

The model was built using the following code

```
#Building model
kmeans_model = KMeans(n_clusters=5, random_state=0)
y_kmeans = kmeans_model.fit_predict(X)
```

With this we can also see the cluster assigned for each customer and each cluster can be categorized into something meaningful that will be in the conclusion section

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)	clusters
0	1	Male	19	15	39	4
1	2	Male	21	15	81	3
2	3	Female	20	16	6	4
3	4	Female	23	16	77	3
4	5	Female	31	17	40	4
5	6	Female	22	17	76	3
6	7	Female	35	18	6	4

The clusters are then visualized with the respective centroids as shown below:

```
#plotting the clusters
plt.scatter(X[y_kmeans==0,0], X[y_kmeans==0,1], s=100, c='red', label = 'Cluster1')
plt.scatter(X[y_kmeans==1,0], X[y_kmeans==1,1], s=100, c='blue', label = 'Cluster2')
plt.scatter(X[y_kmeans==2,0], X[y_kmeans==2,1], s=100, c='green', label = 'Cluster3')
plt.scatter(X[y_kmeans==3,0], X[y_kmeans==3,1], s=100, c='orange', label='Cluster4')
plt.scatter(X[y_kmeans==4,0], X[y_kmeans==4,1], s=100, c='pink', label='Cluster5')

#clusters centers
plt.scatter(kmeans.cluster_centers_[:,0], kmeans.cluster_centers_[:,1], s= 250, c='yellow', label= 'Centroids')

plt.title("Customers based clusters")
plt.xlabel("Annual Income")
plt.ylabel("Spending score")
plt.legend()
plt.show()
```

Online/E-Commerce Customers:

RFM Analysis:

RFM (Recency, Frequency, Monetary) analysis is a behavior-based approach grouping customers into segments. It groups the customers on the basis of their previous purchase transactions. How recently, how often, and how much did a customer buy. RFM filters customers into various groups for the purpose of better service. It helps managers to identify potential customers to do more profitable business. There is a segment of customer who is the big spender but what if they purchased only once or how recently they purchased? Do they often purchase our product? Also, It helps managers to run an effective promotional campaign for personalized service.

Recency (R): Who have purchased recently? Number of days since last purchase (least recency)

Frequency (F): Who has purchased frequently? It means the total number of purchases. (high frequency)

Monetary Value(M): Who have high purchase amount? It means the total money customer spent (high monetary value)

Here, Each of the three variables (Recency, Frequency, and Monetary) consists of four equal groups, which creates 64 (4x4x4) different customer segments.

Steps of RFM (Recency, Frequency, Monetary):

- Calculate the Recency, Frequency, Monetary values for each customer.
- Add segment bin values to RFM table using quartile.
- Sort the customer RFM score in ascending order.

All the relevant libraries that are needed for this project were imported.

Regular data checks were made just to make sure the data is free of null values and does not have any false formatting of data or wrong that would lead to a wrong or bad model.

```
#import libraries

import pandas as pd
import numpy as np
import seaborn as sns
import datetime as dt
```

```
data_rfm = pd.read_csv('data.csv',encoding="ISO-8859-1", dtype={'CustomerID':str, 'InvoiceID':str})
```

```
data_rfm.head()
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850	United Kingdom

```
data_rfm.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
InvoiceNo      541909 non-null object
StockCode      541909 non-null object
Description    540455 non-null object
Quantity       541909 non-null int64
InvoiceDate    541909 non-null object
UnitPrice      541909 non-null float64
CustomerID     406829 non-null object
Country        541909 non-null object
dtypes: float64(1), int64(1), object(6)
memory usage: 33.1+ MB
```

Here in this data CustomerID is like the primary key for the table. I remove all the rows which did not have any customer id.

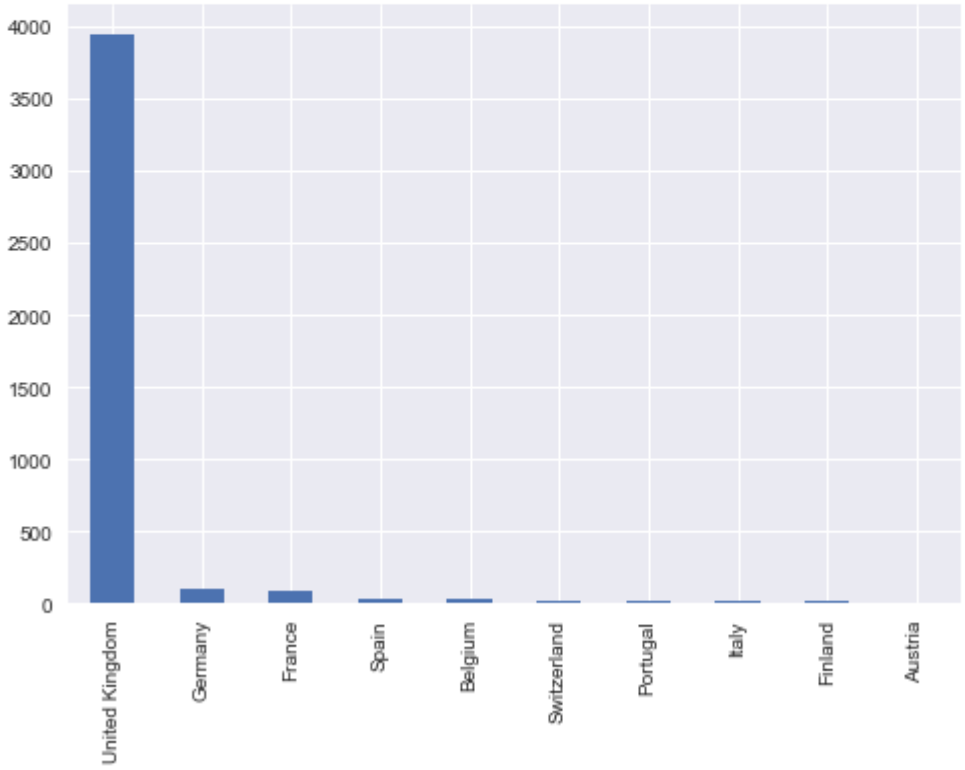
With further analysis it was found that the data was more for UK based customers hence, the further RFM analysis was done based on the UK customers data.

```
#there are many blanks in customer values. Removing it

data_rfm = data_rfm[pd.notnull(data_rfm['CustomerID'])]
```

```
filtered_data = data_rfm[['Country', 'CustomerID']].drop_duplicates()
```

```
#Top ten country's by customer
filtered_data.Country.value_counts()[:10].plot(kind = 'bar')
plt.show()
```



I then did some basic data manipulation and made sure that the data was clean and ready for analysis

```
uk_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 361878 entries, 0 to 541893
Data columns (total 8 columns):
InvoiceNo      361878 non-null object
StockCode      361878 non-null object
Description    361878 non-null object
Quantity       361878 non-null int64
InvoiceDate    361878 non-null object
UnitPrice      361878 non-null float64
CustomerID     361878 non-null object
Country        361878 non-null object
dtypes: float64(1), int64(1), object(6)
memory usage: 24.8+ MB
```

Filter required Columns

Here, you can filter the necessary columns for RFM analysis. You only need her five columns CustomerID, InvoiceDate, InvoiceNo, Quantity, and UnitPrice. CustomerId will uniquely define your customers, InvoiceDate help you calculate recency of purchase, InvoiceNo helps you to count the number of time transaction performed(frequency). Quantity purchased in each transaction and UnitPrice of each unit purchased by the customer will help you to calculate the total purchased amount.

- For Recency, Calculate the number of days between present date and date of last purchase each customer.
- For Frequency, Calculate the number of orders for each customer.
- For Monetary, Calculate sum of purchase price for each customer.

Customers with the lowest recency, highest frequency and monetary amounts considered as top customers. qcut() is Quantile-based discretization function. qcut bins the data based on sample quantiles. For example, 1000 values for 4 quantiles would produce a categorical object indicating quantile membership for each customer.

RFM Result Interpretation:

Combine all three quartiles(r_quartile,f_quartile,m_quartile) in a single column, this rank will help you to segment the customers well group.

Conclusion:

Mall customers

```
#plotting the clusters
plt.scatter(X[y_kmeans==0,0], X[y_kmeans==0,1], s=100, c='red', label = 'Cluster1')
plt.scatter(X[y_kmeans==1,0], X[y_kmeans==1,1], s=100, c='blue', label = 'Cluster2')
plt.scatter(X[y_kmeans==2,0], X[y_kmeans==2,1], s=100, c='green', label = 'Cluster3')
plt.scatter(X[y_kmeans==3,0], X[y_kmeans==3,1], s=100, c='orange', label='Cluster4')
plt.scatter(X[y_kmeans==4,0], X[y_kmeans==4,1], s=100, c='pink', label='Cluster5')

#clusters centers
plt.scatter(kmeans.cluster_centers_[:,0], kmeans.cluster_centers_[:,1], s= 250, c='yellow', label= 'Centroids')

plt.title("Customers based clusters")
plt.xlabel("Annual Income")
plt.ylabel("Spending score")
plt.legend()
plt.show()
```

Model Interpretation

Cluster 1 (Red Color) -> earning high but spending less

cluster 2 (Blue Color) -> average in terms of earning and spending

cluster 3 (Green Color) -> earning high and spending high [TARGET SET]

cluster 4 (Orange Color) -> earning less but spending more

Cluster 5 (Pink Color) -> Earning less, spending less

We can put Cluster 3 into some alerting system where email can be sent to them on daily basis as these re easy to converse wherein others we can set like once in a week or once in a month.

E-commerce Data

RFM analysis helps managers to identify potential customers to do more profitable business. There is a segment of customer who is the big spender but what if they purchased only once or how recently they purchased? Do they often purchase our product? Also, it helps managers to run an effective promotional campaign for personalized service.