Department of Electrical and Computer Engineering

North South University



CSE 533: Machine Learning Project

Customer Segmentation for E-commerce Businesses using K-means Algorithm

Name: HM Adnan Zami **ID # 2115164650**

Faculty Advisor:

Dr. Mohammad Ashrafuzzaman Khan

Assistant Professor

Department of ECE

Spring 2021

Table of Contents

CHAPTER 1: INTRODUCTION 4

1.1 E-Commerce Industry of Bangladesh 4

1.2 Challenges for E-Commerce Business 4

1.3 Personalized Marketing 5

1.4 Customer Segmentation 5

1.5 Project Goals 5

CHAPTER 2: LITERATURE REVIEW 6

2.1 RFM Analysis 6

2.1.1 Recency 6

2.1.2 Frequency 6

2.1.3 Monetary 7

2.1.4 RFM Score 7

2.2 K-MEANS ALGORITHM 7

2.3 Advantages of k-means: 8

2.4 Disadvantages of k-means: 9

- Manually selecting k: To find the best solution, use the “Loss vs. Clusters” plot (k) 9

CHAPTER 3: METHODOLOGY 10

3.1 Method for Analysis 10

3.1.2 Feature Selection 10

3.1.3 Dataset 11

3.1.4 Data Cleaning 13

3.2 Recency, Frequency and Monetary Value 14

3.2.1 Recency 14

3.2.2 Frequency 15

3.2.3 Monetary Value 16

3.3 RFM Score 17

3.3.1 Set RFM Ranks using Quintile method 18

3.3.1 Gaussian Distribution 20

3.4 Segmentation with K-means Clustering 21

3.4.1 Recency & Frequency 21
3.4.2 Recency & M Rank 24
3.4.3 Frequency & Monetary Rank 25
3.4.4 Recency & RFM Score 27
3.4.5 Recency Rank & Monetary Rank 30
3.4.4 Recency Rank & Frequency Rank 32
3.4.4 Recency Rank & Frequency Rank 33
CHAPTER 4: RESULTS 37
4.1 Recency Rank & Monetary Rank 37
4.2 Country & RFM Score 38
CHAPTER 5: CONCLUSION 40
5.1 Conclusion 40

CHAPTER 1: INTRODUCTION

This chapter will briefly describe the e-commerce industry of Bangladesh along with the challenges. Methods for overcoming those challenges will be discussed and an outline of the project objective will also be mentioned.

1.1 E-Commerce Industry of Bangladesh

Ecommerce growth began with the rise of the internet and continues to this day, especially in B2C ecommerce (Business to Customer). When a user uses ecommerce to shop, he or she finds it to be simple and fast. Customers are more likely to shop from ecommerce because it is easy to use. This has led to the growth of e-commerce in Bangladesh. “According to the latest data from Statista, the e-commerce market in Bangladesh in 2019 stood at 1,648 million USD which will increase to 2,077 million USD this year and in 2023, the market size will be 3,077 million USD.” [New Age BD, 2020]

1.2 Challenges for E-Commerce Business

Due to such fast growth of e-commerce businesses and the changes in customer behavior due to the changes in technology, it has become difficult for businesses to identify patterns in customer habits. Businesses, therefore, need to excel in providing the best user experience to stay in the competition. Some of the common challenges e-commerce businesses face are

- Attracting the perfect customer
- Providing superior Customer Experience
- Retaining Customer Loyalty
- Generating targeted traffic
- Capturing quality leads
- Converting Shoppers into Paying Customers

1.3 Personalized Marketing

E-commerce businesses needs to create personalized offers for each customer type to ensure effective marketing. Personalization in ecommerce services, such as offering product recommendations, links recommendations, advertisements, or text and graphics that conform to the users' characteristics and needs, can help to alleviate information overload. Furthermore, Personalized services in ecommerce can sustain consumer loyal ty of established customers and attract new customers by delivering service to customers based on their preferences and characteristics, which helps to solve the problem of overloaded details.

1.4 Customer Segmentation

Customer segmentation is needed before personalization can be enforced. Customer segmentation is accomplished by analyzing the customer record, which includes demographic information and purchasing history like shows the number of days after a customer's last transaction, the total number of invoices he or she has received, and the total amount spent by a customer. Business Rule, Quantile Participation, Supervised Clustering, Unsupervised Clustering, Customer Profiling, RFM Cell Classification Grouping, Customer Likeness Clustering, and Purchase Affinity Clustering are examples of segmentation approaches.

1.5 Project Goals

We will perform customer segmentation on an e-commerce business dataset to identify customer habits. Based on data analysis, the project will identify customer segmentation approaches. We will aim to segment customers based on their spending habits, their online activity, their online presence and their location. Afterwards, we will try to create marketing strategies to assist businesses in creating marketing campaigns exclusively for each segment of customers.

CHAPTER 2: LITERATURE REVIEW

Before starting our project, we studied various methods of customer segmentation and research papers to find the best approach to perform this project. We analyzed RFM method for ranking customers and K-means algorithm to perform segmentation.

2.1 RFM Analysis

RFM stands for Recency, Frequency, and Monetary Value, each of which refers to a distinct consumer characteristic. Since frequency and monetary value affect a customer's lifetime value, and recency effects retention, a measure of engagement, these RFM metrics are important measures of a customer's conduct. These three RFM considerations can be used to estimate how likely (or unlikely) a customer is to do business with a company again or, in the case of a charitable organization, make another donation. As a result, we've chosen to use RFM in our project.

2.1.1 Recency

The amount of time since your last order or interaction with the product. The more recently a customer has made a purchase with a company, the more likely they are to remember the company and its brand for future purchases. Recent customers have a greater chance of participating in future purchases than customers who have not purchased from the company in months or even longer periods.Such data can be used to alert recent customers to return to the store as soon as possible to complete their purchases. To ensure that lapsed consumers are not forgotten, marketing campaigns should be made to remind them that it has been a while since their last purchase while also providing an opportunity to rekindle their patronage.

2.1.2 Frequency

The total number of transactions or the average time between transactions/engaged visits is used to calculate the total number of transactions. Factors such as the type of product, the sales price point, and the need for replenishment or replacement may all influence how often a consumer transacts. If a customer's shopping period can be anticipated, such as when they need fresh groceries, marketing campaigns may be focused on reminding them to visit the store when products like eggs or milk are running low.

2.1.3 Monetary

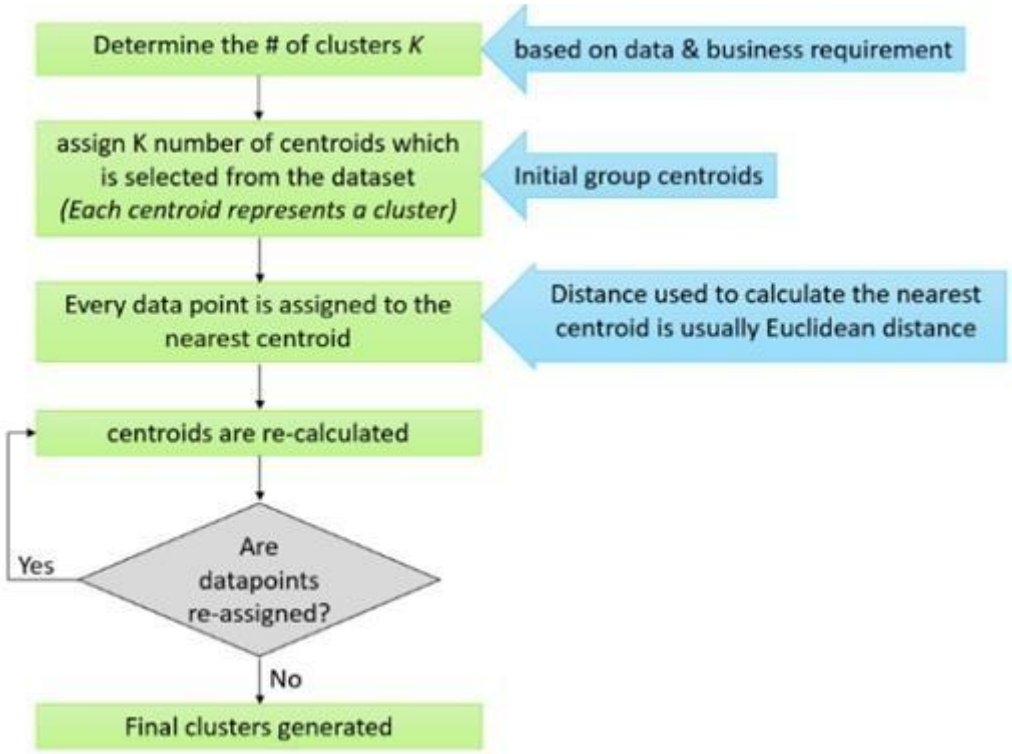
The total or average value of transactions. The lucrativeness of the customer's purchases with the company during their transactions determines monetary value. The natural tendency is to place a greater focus on motivating consumers who spend the most money to keep doing so. While this may result in a higher return on marketing and customer service investments, it also runs the risk of alienating consumers who have been loyal but have not invested as much on each purchase.

2.1.4 RFM Score

An RFM study identifies the customers who are most likely to respond to a new bid. Each customer receives a three-digit RFM score (ranging from 111 to 555) as a result of the RFM review. A 555 means that a consumer has bought a product or service the most recently, regularly, and for the most money. Finally, we can rate these customers by adding their individual R, F, and M scores together to get an RFM score. This RFM score is simply the sum of the individual R, F, and M scores, calculated by giving each RFM attribute equal weight.

2.2 K-MEANS ALGORITHM

K-means clustering is an iterative clustering algorithm in which the number of clusters K is predetermined and each data point is assigned to one of the K clusters iteratively based on feature similarity. In data training, k-Means is one of the most basic clustering algorithms. It can be used to automatically identify groups of related instances/items/objects/points. The algorithm divides instances into a set of clusters based on the number of clusters listed by the user (e.g. assume k clusters). The first and most critical step is to select a set of k centroids at random, normally one for each cluster as far apart as possible. The most common method for calculating the distance between an instance and its centroid is the Euclidean distance. After each instance addition, the cluster centroids are still recalculated. This procedure is repeated until no further changes are needed.



Broad steps of the k-means algorithm:

2.3 Advantages of k-means:

1. It can handle massive data sets. Convergence is guaranteed. It's possible to warm up the locations of centroids. Adapts quickly to new situations. Generalizes to various shapes and sizes of clusters, such as elliptical clusters.
2. If the variables are large, K-Means is always computationally faster than hierarchical clustering if the k smalls are held.
3. K-Means clusters are more compact than hierarchical clustering, especially when the clusters are globular.

2.4 Disadvantages of k-means:

- ***Manually selecting k:*** *To find the best solution, use the "Loss vs. Clusters" plot (k).*

- **Being reliant on starting values:** You can reduce this reliance for a low by running k-means many times with different initial values and selecting the best result. As the number of variables increases, more advanced versions of k-means are needed to choose better initial centroids values (called k-means seeding).
- **Data of various sizes and densities are clustered:** When clustering data with different sizes and densities, k-means has trouble. To cluster such data, k-means must be generalized.
- **Outliers are grouped together:** Outliers can drag centroids, or outliers can be given their own cluster rather than being ignored. Until clustering, consider deleting or clipping outliers.
- **Scaling in relation to the number of measurements:** Outliers can drag centroids, or outliers can be given their own cluster rather than being ignored. Until clustering, consider deleting or clipping outliers.

CHAPTER 3: METHODOLOGY

The primary objective of this project is to identify customer habits to develop targeted marketing to increase conversion rates of ecommerce businesses. The reason for choosing ecommerce businesses is because it is easier to collect customer data from online sources. Basic information on customer purchases is enough to design the model for customer segmentation for this project. For this reason, the dataset used in this project is from an online retail shop in the UK. It contains all the transactions occurring for a UK-based and registered, non-store online retail between 01/12/2009 and 09/12/2011. The company mainly sells unique all-occasion gift-ware. It has 8 attributes and a total of 1,067,371 instances of transactions.

3.1 Method for Analysis

The technique used for segmentation in this project is K-means clustering. K-means algorithm is quite effective for this project as it is an unsupervised machine learning algorithm. Since we are using a dataset without any labels, we will use unsupervised machine learning to deduce the patterns in customer purchase behaviour.

To understand the data, we will be using RFM analysis which stands for Recency, Frequency and Monetary Value. This analysis based on the three characteristics of a customer will provide us with the following information.

- Recency - days since last purchase
- Frequency - total number of invoices
- Monetary Value - total amount spent by a customer.

After computing each value, we will combine these 3 features to determine the RFM score. From a scale of 1-5, each customer will be given a score for Recency, Frequency and Monetary with 5 being the highest. All three scores will be averaged to determine the final RFM score. The corresponding values for RFM to it range from 1-5 will be determined based on the percentile of the RFM values. This is further explained during classification.

3.1.2 Feature Selection

For our particular project, we will be performing segmentations based on these three characteristics and country of origin as that feature is available in our dataset. The features we chose for this project and the attributes that we will use from the dataset to define these features are shown in the following table.

No.	Attributes	Feature Analysis
1.	Invoice Date	Recency
2.	Invoices	Frequency
3.	Unit Price	Monetary Value
4.	Purchase Quantity	
5.	Country	Country

Based on these features, we will perform K-means clustering to create the following segmentations.

1. Recency & Frequency
2. Recency & Monetary
3. Frequency & Monetary
4. RFM Score
5. RFM Score & Country

The results from these segmentations will be used to identify classes of customers on whom specific marketing strategies can be implemented to create targeted marketing campaigns.

3.1.3 Dataset

The first step to performing this analysis is exploring and cleaning the dataset. We will import the Pandas and Numpy libraries for data manipulation and mathematical calculations respectively. To explore our dataset, we will first load it into the dataframe and run the following commands.

```python
#load libraries
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

#read and set dataset = df
df = pd.read_csv(r'C:/Users/LENOVO/Python/Project/test5000.csv')

#explore dataset
df.head() #top 5 rows
df.shape #number of (rows, columns)

#descriptive statistics of data
df.describe().transpose()
```

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

df.head()

	count	mean	std	min	25%	50%	75%	max
Quantity	541910.0	9.552234	218.080957	-80995.00	1.00	3.00	10.00	80995.0
Price	541910.0	4.611138	96.759765	-11062.06	1.25	2.08	4.13	38970.0
Customer ID	406830.0	15287.684160	1713.603074	12346.00	13953.00	15152.00	16791.00	18287.0

df.describe().transpose()

There are a total of 397,925 instances and 8 columns with the following attributes:

- **InvoiceNo**: Invoice number. Nominal. A 6-digit integral number uniquely assigned to each transaction. If this code starts with the 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 a transaction was generated.
- **UnitPrice**: Unit price. Numeric. Product price per unit in sterling (Â£).

- **CustomerID**: Customer number. Nominal. A 5-digit integral number uniquely assigned to each customer.
- **Country**: Country name. Nominal. The name of the country where a customer resides.

3.1.4 Data Cleaning

We will be using Customer ID as the index for the rows. All the rows must be checked to see if any of the values are missing. Also, there were orders which were cancelled. These orders have 'c' assigned to their invoice and their Quantity value is negative. We need to find and remove the rows with NULL values for Customer ID and negative values for Quantity.

```
Invoice            0
StockCode          0
Description     1454
Quantity           0
InvoiceDate        0
Price              0
Customer ID    135080
Country            0
dtype: int64
```

df.isnull().sum()

```
#total NULL values in each column
df.isnull().sum()

#drop all rows with NULL values
df = df.dropna()

#drop negative values for Quantity
df = df[df['Quantity'] > 0]

#remaining (rows, columns)
df.shape
```

After removing the rows with missing Customer ID and negative Quantity values, we remain with the following

Rows	Columns
397,925	8

3.2 Recency, Frequency and Monetary Value

Now that the dataset is ready, we will begin by finding the RFM values. Using Customer ID as the index, we will compute and store the values for Recency, Frequency and Monetary values in separate dataframes. Afterwards, we will rank the customers from 1-5 with 5 being the highest. The value ranges for the ranks will be determined by finding the top percentiles. Those in the top 20 percentile will get a rank of 5 and then the top 40 percentile will receive rank 4 and so on.

3.2.1 Recency

The Recency will be calculated based on the date of purchase. The latest date of purchase will be taken and compared with today's date to find the number of days since last purchase. The more recent the latest purchase date, the higher the customer will rank.

This dataset contains the dates from 2010 to 2011. Generally, the current date is used to determine the number of days since the last purchase but for our purpose, we will compare the purchase date with 1/1/2012 to keep the values within the range of 1 to 365.

```
#Recency

rec = []  #empty list to store number of days since last purchase

date_from = ['1/1/2012  00:00']   #present date, compared with purchased date
date_from = pd.to_datetime(date_from)

#loop unique customers
for x in u_customer:
    row_customer = df.loc[df['Customer ID'] == x]  #select rows with Customer ID
    inv_dates = row_customer.InvoiceDate  #store invoice date
    recent_date = max(inv_dates)   #find latest purchase date
    recent_date = pd.to_datetime(recent_date)
    num_days = date_from - recent_date    #calculate number of days
    conv_days = num_days.days[0]
    rec.append(conv_days)       #store number of days

recency = {
    'Customer ID': u_customer,
    'Days Since Last Purchase': rec
}


r_analysis = pd.DataFrame(recency, columns = ['Customer ID', 'Days Since Last Purchase'])
r_analysis
```

	Customer ID	Days Since Last Purchase
0	12346.0	347
1	12347.0	151
2	12348.0	97
3	12349.0	40
4	12350.0	332
...	...	...
4334	18280.0	299
4335	18281.0	202
4336	18282.0	148
4337	18283.0	117
4338	18287.0	223

4339 rows × 2 columns

Recency output

3.2.2 Frequency

For frequency, we will determine the number of purchases made by a customer. For this, we need to count the number of invoices. Each invoice has multiple instances of items for which we will select the unique invoices for each customer and count them to determine the frequency.

```
# Frequency

items = [] #empty list to store all invoices of each  customer
invoices = [] #empty list to store all unique invoices of each customer

for x in u_customer: #iterate through unique customers
    row_customer = df.loc[df['Customer ID'] == x]
    inv = row_customer.Invoice #select all invoices for each customer
    s_inv = set(inv)  #select unique invoices each customer
    f_inv = inv.count() #count total invoices for each customer
    u_inv = len(s_inv) #count total unique invoices
    items.append(f_inv) #add invoices to empty item list
    invoices.append(u_inv) #add invoices to empty invoice list


inv_record = {
    'Customer ID': u_customer,
    'Items': items,
    'Invoices': invoices
}

f_analysis = pd.DataFrame(inv_record, columns = ['Customer ID', 'Items', 'Invoices'])

f_analysis
```

	Customer ID	Items	Invoices
0	12346.0	1	1
1	12347.0	182	7
2	12348.0	31	4
3	12349.0	73	1
4	12350.0	17	1
...	...	...	...
4334	18280.0	10	1
4335	18281.0	7	1
4336	18282.0	12	2
4337	18283.0	756	16
4338	18287.0	70	3

4339 rows × 3 columns

Frequency output

3.2.3 Monetary Value

The Monetary Value is the total amount spent by a customer. To determine this, we will iterate through unique Customer ID and select the Quantity and Price attributes. Then we will multiply the values to determine the Monetary Value.

```
#Monetary

monetary_value = [] #empty list to store total money spent

for x in u_customer:   #iterate through unique customers
    row_customer = df.loc[df['Customer ID'] == x]  #iterate unique customers
    unit_price = row_customer.Price  #store price for unique customer
    quantity = row_customer.Quantity #store quantity for unique customer
    invoice_money = unit_price * quantity #calculate money spent for each row
    total_money = invoice_money.sum() #sum total money spent
    monetary_value.append(total_money) #store total money spent


cus_mon = {
    'Customer ID': u_customer,
    'Total Money Spent': monetary_value,
}

m_analysis = pd.DataFrame(cus_mon, columns = ['Customer ID', 'Total Money Spent'])

m_analysis
```

	Customer ID	Total Money Spent
0	12346.0	77183.60
1	12347.0	4310.00
2	12348.0	1797.24
3	12349.0	1757.55
4	12350.0	334.40
...	...	...
4334	18280.0	180.60
4335	18281.0	80.82
4336	18282.0	178.05
4337	18283.0	2094.88
4338	18287.0	1837.28

4339 rows × 2 columns

Monetary Value output

3.3 RFM Score

After computing each feature, we combined the results to show the RFM table that shows all the values. From this table, we will determine the rank for each customer by finding the ranking of each RFM value and then combining them to determine the RFM score.

```
rfm_records = {
    'Customer ID': u_customer,
    'Items': items,
    'Invoices': invoices,
    'Total Money Spent': monetary_value,
    'Days Since Last Purchase': rec
}

rfm_analysis = pd.DataFrame(rfm_records, columns=['Customer ID',
                                                  'Items', 'Invoices',
                                                  'Total Money Spent',
                                                  'Days Since Last Purchase'])
rfm_analysis
```

	Customer ID	Items	Invoices	Total Money Spent	Days Since Last Purchase
0	12346.0	1	1	77183.60	347
1	12347.0	182	7	4310.00	151
2	12348.0	31	4	1797.24	97
3	12349.0	73	1	1757.55	40
4	12350.0	17	1	334.40	332
...	...	...	...	...	...
4334	18280.0	10	1	180.60	299
4335	18281.0	7	1	80.82	202
4336	18282.0	12	2	178.05	148
4337	18283.0	756	16	2094.88	117
4338	18287.0	70	3	1837.28	223

4339 rows × 5 columns

RFM Output

3.3.1 Set RFM Ranks using Quintile method

We will use the quintile method to set the ranking. The quintile method selects the top percentile of a dataset. In our cases, there will be a vector containing 4 values that will set the ranges for creating partitions or ranks (1-5). Customers in the top 20 percentile will receive the highest rank of 5. Since these percentiles are dependent on the dataset values, this method will work better in ranking the customers rather than selecting partitioning the customers into groups by dividing the maximum value

We will find the range partition vector of each of the RFM values by running the quantile () function on our RFM analysis dataset.

	Customer ID	Items	Invoices	Total Money Spent	Days Since Last Purchase	Recency Rank	Frequency Rank	Monetary Rank	RFM Score
0.2	13505.6	14.0	1.0	250.106	93.0	1.0	2.0	1.6	6.0
0.4	14708.2	29.0	2.0	489.724	114.0	2.0	3.0	2.2	9.0
0.6	15882.8	58.0	3.0	941.942	158.0	3.0	4.0	3.8	10.0
0.8	17080.4	121.0	6.0	2057.914	240.0	4.0	5.0	4.4	12.0

rfm_analysis.quantile([0.2, 0.4, 0.6, 0.8])

Based on the results, the following table shows the ranges for each rank for RFM. Here, the R, F and M represent the corresponding values for days since last purchase, invoices and total money spent respectively.

Ranks	**Recency (R)**	**Frequency (F)**	**Monetary Value (M)**
1	(0.8) =< R1	F < (0.2)	M < (0.2)
2	(0.6) <= R < (0.8)	(0.2) <= F < (0.4)	(0.2) <= M < (0.4)
3	(0.4) <= R < (0.6)	(0.4) <= F < (0.6)	(0.4) <= M < (0,6)
4	(0.2) <= R < (0.4)	(0.6) <= F < (0.8)	(0.6) <= M < (0.8)
5	R < (0.2)	(0.8) =< F	(0.8) =< M

```
qtile = rfm_analysis.quantile([0.2, 0.4, 0.6, 0.8]) #quantile function to split records into percentile
qtile_mon = qtile['Total Money Spent']

monetary_rank = []  #list for storing M rank

#monetary value rank computing loop
for x in rfm_analysis['Total Money Spent']:
    if x < qtile_mon[0.2]:
        m_rank = 1
    elif ((qtile_mon[0.2] <= x) & (x < qtile_mon[0.4])):
        m_rank = 2
    elif ((qtile_mon[0.4] <= x) & (x < qtile_mon[0.6])):
        m_rank = 3
    elif ((qtile_mon[0.6] <= x) & (x < qtile_mon[0.8])):
        m_rank = 4
    elif qtile_mon[0.8]<=x:
        m_rank = 5
    monetary_rank.append(m_rank) #add rank to list

```

The final results after computing the ranks for R,F,M is shown in the following table. The RFM Score is the sum of frequency, recency and monetary ranks.

	Customer ID	Items	Invoices	Total Money Spent	Days Since Last Purchase	Recency Rank	Frequency Rank	Monetary Rank	RFM Score	Country
0	12346.0	1	1	77183.60	347	1	2	5	8	United Kingdom
1	12347.0	182	7	4310.00	151	3	5	5	13	Iceland
2	12348.0	31	4	1797.24	97	4	4	4	12	Finland
3	12349.0	73	1	1757.55	40	5	2	4	11	Italy
4	12350.0	17	1	334.40	332	1	2	2	5	Norway
...	...	...	...	...	...	...	...	...	...	...
4334	18280.0	10	1	180.60	299	1	2	1	4	United Kingdom
4335	18281.0	7	1	80.82	202	2	2	1	5	United Kingdom
4336	18282.0	12	2	178.05	148	3	3	1	7	United Kingdom
4337	18283.0	756	16	2094.88	117	3	5	5	13	United Kingdom
4338	18287.0	70	3	1837.28	223	2	4	4	10	United Kingdom

4339 rows × 10 columns

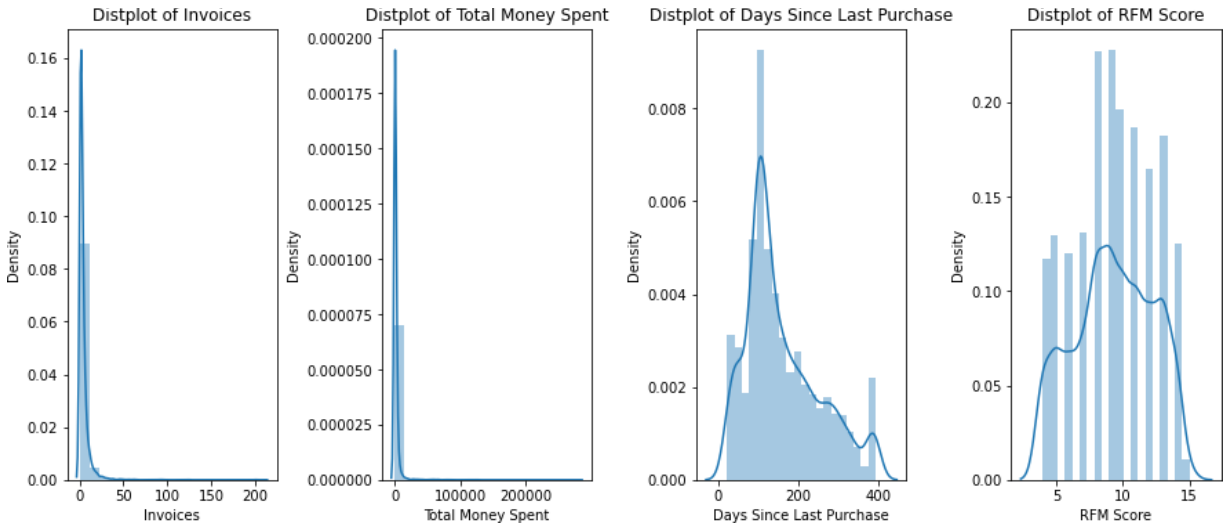
RFM Score

3.3.1 Gaussian Distribution

After finding the RFM ranks and RFM Score, we performed the Gaussian distribution to observe the distribution of the data. Using the seaborn library, we plotted the invoices (F), total money spent (M), days since last purchase (R) and RFM score.

```
plt.figure(1 , figsize = (15 , 6))
n = 0
for x in ['Invoices', 'Total Money Spent' , 'Days Since Last Purchase', 'RFM Score']:
    n += 1
    plt.subplot(1 , 4 , n)
    plt.subplots_adjust(hspace =0.5 , wspace = 0.5)
    sns.distplot(rfm_analysis[x] , bins = 20)
    plt.title('Distplot of {}'.format(x))
plt.show()
```

From the plot, we can see that the RFM score has the highest density between 8-10 which is a good score suggesting that the business is doing well. The days since last purchase shows that majority of the customer's falls around 100 days as the highest density. The invoices and total money spent plot does not offer a distinct result as this graph's 'wspace' is not enough because the values are spread apart.



3.4 Segmentation with K-means Clustering

We imported the following libraries to perform segmentation and data visualization:

- matplotlib.pyplot - creating static, animated, and interactive visualizations
- seaborn - data visualization library based on matplotlib.
- Sklearn - for machine learning and statistical modeling

3.4.1 Recency & Frequency

Our first segmentation was between Recency, Days Since Last Purchase, and Frequency, Invoices. The first step to performing k-means is to find the value for k. There are 2 methods for finding the value for K which

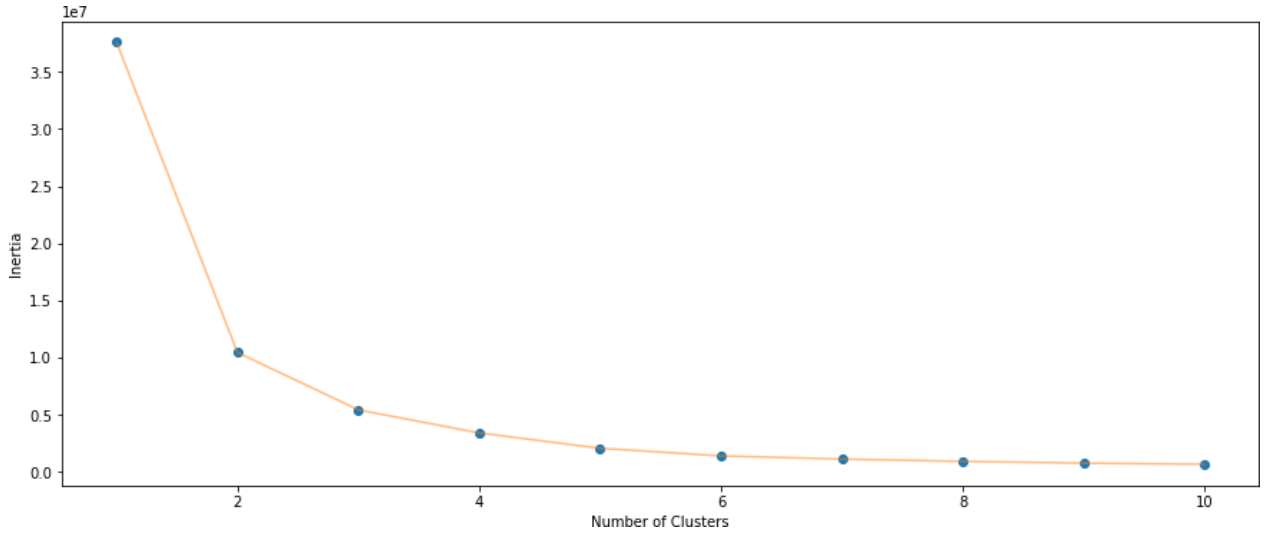
- Elbow method and Silhouette Method

For our purpose, we chose the elbow method. In Elbow method, we need to calculate the WCSS, within-cluster sum of squares, also known as inertia using the following formula

$$WCSS = \sum_{i \in n} (X_i - Y_i)^2$$

.

It will take all the datapoints for a number of clusters and calculated the square sum of the datapoints. As the number of clusters increases, the WCSS value will start to decrease. WCSS value is largest when K = 1. When we analyze the graph, we can see that the graph will rapidly change at a point and thus creating an elbow shape. From this point, the graph starts to move almost parallel to the X-axis. The K value corresponding to this point is the optimal K value or an optimal number of clusters.



From the graph, we can select 4 as our value for K. We perform K-means clustering using the following steps.

1. Select random centroids
    a. Define boundaries by finding the average Euclidean distance between other centroids
2. Find Euclidean distance of data points within cluster and average the values
3. Adjust centroid to the averaged value and iterate Step 1

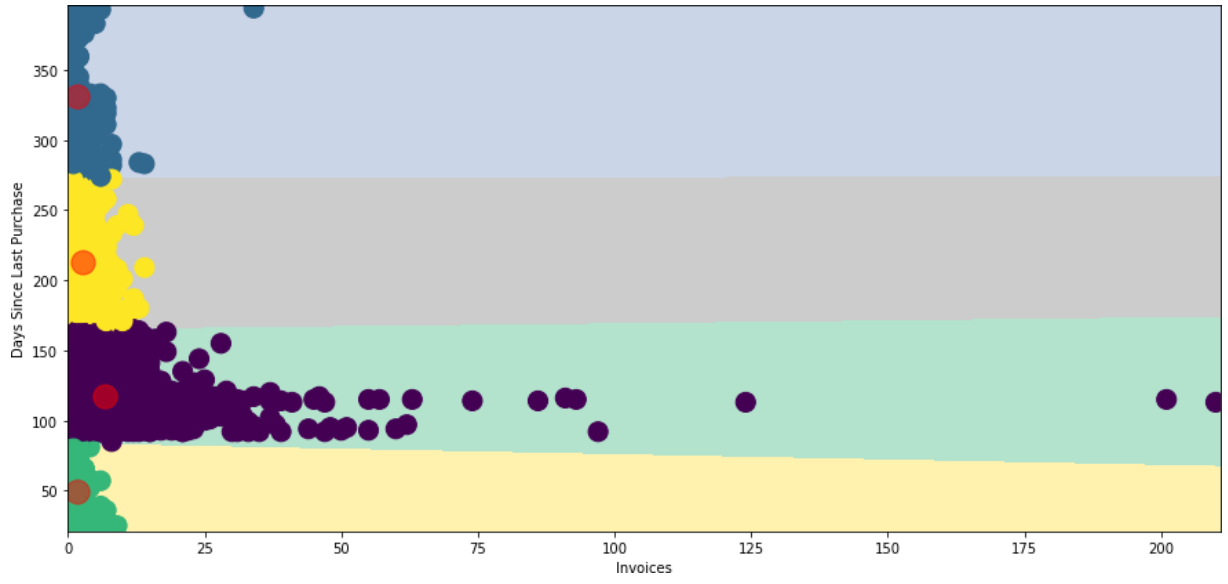
We will now implement k-means clustering algorithm. In k-means clustering. Here, we set the n_cluster = 4 as we chose 4 clusters. The maximum iteration is set to 300. After applying running the code, we plotted the results using the matplot library to plot the results.

```
# select 4 clusters
algorithm = (KMeans(n_clusters = 4 ,init='k-means++', n_init = 10 ,max_iter=300,
                    tol=0.0001,  random_state= 111  , algorithm='elkan') )
algorithm.fit(X1)
labels1 = algorithm.labels_
centroids1 = algorithm.cluster_centers_

h = 0.02
x_min, x_max = X1[:, 0].min() - 1, X1[:, 0].max() + 1
y_min, y_max = X1[:, 1].min() - 1, X1[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])
```

```
plt.figure(1 , figsize = (15 , 7) )
plt.clf()
Z = Z.reshape(xx.shape)
plt.imshow(Z , interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

plt.scatter( x = 'Invoices' ,y = 'Days Since Last Purchase' , data = rfm_analysis, c = labels1 ,
            s = 200 )
plt.scatter(x = centroids1[: , 0] , y =  centroids1[: , 1] , s = 300 , c = 'red' , alpha = 0.5)
plt.ylabel('Days Since Last Purchase') , plt.xlabel('Invoices')
plt.show()
```

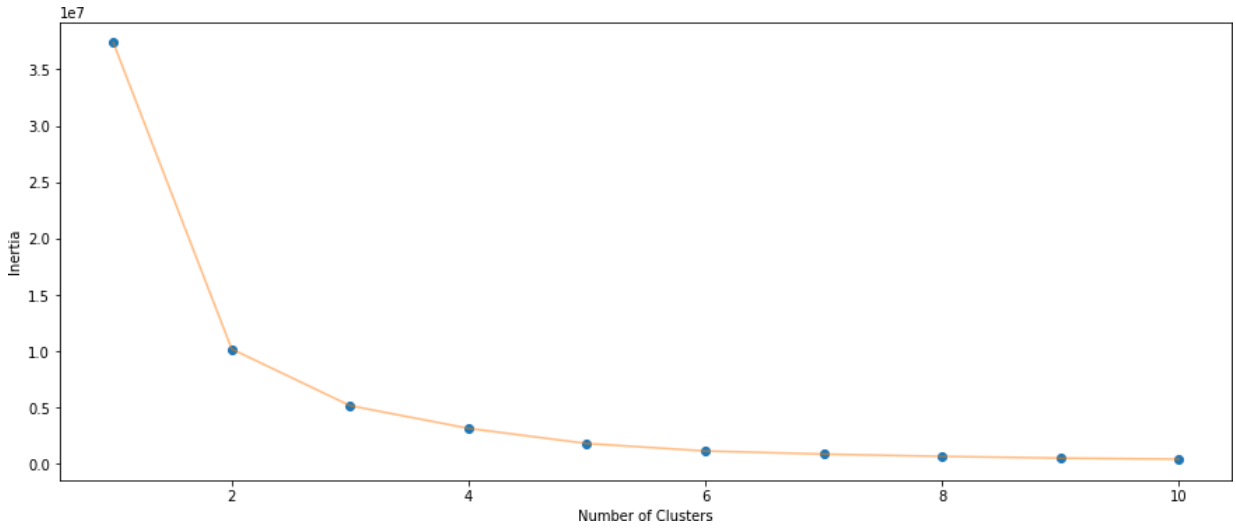


3.4.2 Recency & M Rank

Similarly, we performed the k-means clustering for Recency, days since last purchase, and M Rank. By performing the same steps as 3.4.1, we found the following results.

```
# Recency & Monetary Rank
X1 = rfm_analysis[['Days Since Last Purchase', 'Monetary Rank']].iloc[: , :].values
inertia = []
for n in range(1 , 11):
    algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init = 10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  , algorithm='elkan') )
    algorithm.fit(X1)
    inertia.append(algorithm.inertia_)

plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(1 , 11) , inertia , 'o')
plt.plot(np.arange(1 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Number of Clusters') , plt.ylabel('Inertia')
plt.show()
```



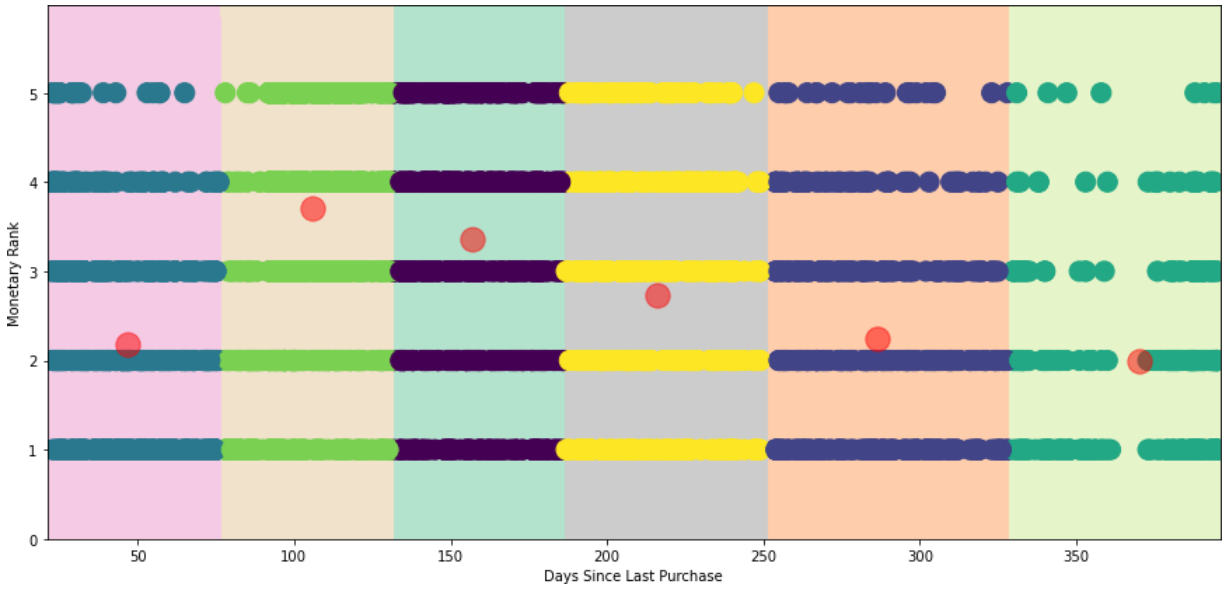
Here, we selected 6 clusters.

```
# select 4 clusters
algorithm = (KMeans(n_clusters = 6 ,init='k-means++', n_init = 10 ,max_iter=300,
                    tol=0.0001,  random_state= 111  , algorithm='elkan') )
algorithm.fit(X1)
labels1 = algorithm.labels_
centroids1 = algorithm.cluster_centers_

h = 0.02
x_min, x_max = X1[:, 0].min() - 1, X1[:, 0].max() + 1
y_min, y_max = X1[:, 1].min() - 1, X1[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])
```

```
# Recency & Monetary Rank
plt.figure(1 , figsize = (15 , 7) )
plt.clf()
Z = Z.reshape(xx.shape)
plt.imshow(Z , interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

plt.scatter( x = 'Days Since Last Purchase' ,y = 'Monetary Rank' , data = rfm_analysis, c = labels1 ,
            s = 200 )
plt.scatter(x = centroids1[: , 0] , y =  centroids1[: , 1] , s = 300 , c = 'red' , alpha = 0.5)
plt.ylabel('Monetary Rank') , plt.xlabel('Days Since Last Purchase')
plt.show()
```

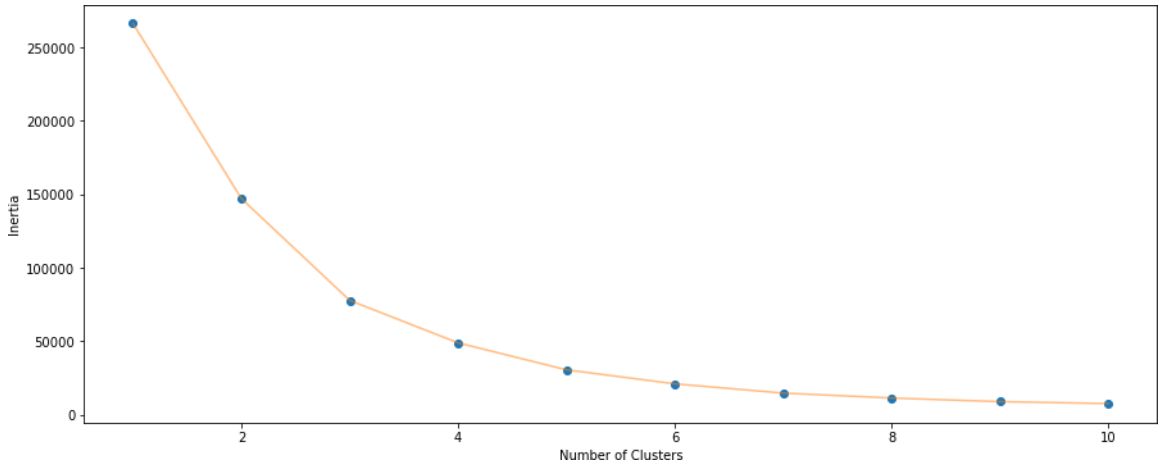


3.4.3 Frequency & Monetary Rank

Similarly, we performed the k-means clustering for Recency, days since last purchase, and M Rank. By performing the same steps as 3.4.1, we found the following results.

```
# Frequency & Monetary Rank
X1 = rfm_analysis[['Invoices', 'Monetary Rank']].iloc[: , :].values
inertia = []
for n in range(1 , 11):
    algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init = 10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  , algorithm='elkan') )
    algorithm.fit(X1)
    inertia.append(algorithm.inertia_)

plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(1 , 11) , inertia , 'o')
plt.plot(np.arange(1 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Number of Clusters') , plt.ylabel('Inertia')
plt.show()
```



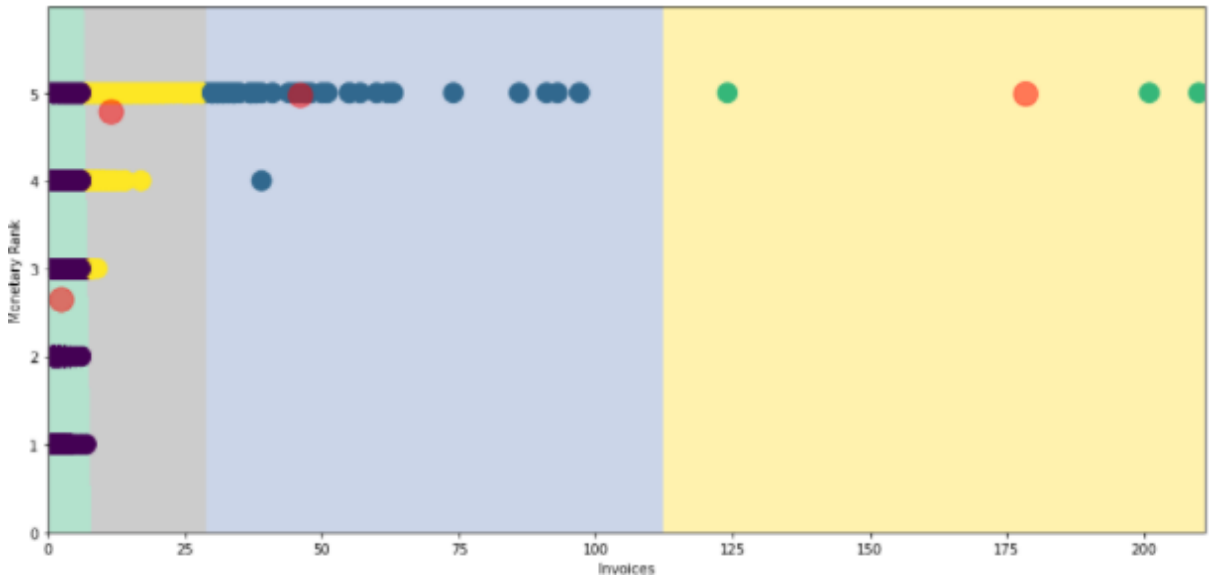
Here we selected 4 as the number of clusters.

```
# select 4 clusters
algorithm = (KMeans(n_clusters = 4 ,init='k-means++', n_init = 10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  , algorithm='elkan') )
algorithm.fit(X1)
labels1 = algorithm.labels_
centroids1 = algorithm.cluster_centers_

h = 0.02
x_min, x_max = X1[:, 0].min() - 1, X1[:, 0].max() + 1
y_min, y_max = X1[:, 1].min() - 1, X1[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])
```

```
# Frequency & Monetary Rank
plt.figure(1 , figsize = (15 , 7) )
plt.clf()
Z = Z.reshape(xx.shape)
plt.imshow(Z , interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

plt.scatter( x = 'Invoices' ,y = 'Monetary Rank' , data = rfm_analysis, c = labels1 ,
            s = 200 )
plt.scatter(x = centroids1[: , 0] , y =  centroids1[: , 1] , s = 300 , c = 'red' , alpha = 0.5)
plt.ylabel('RFM Score') , plt.xlabel('Days Since Last Purchase')
plt.show()
```

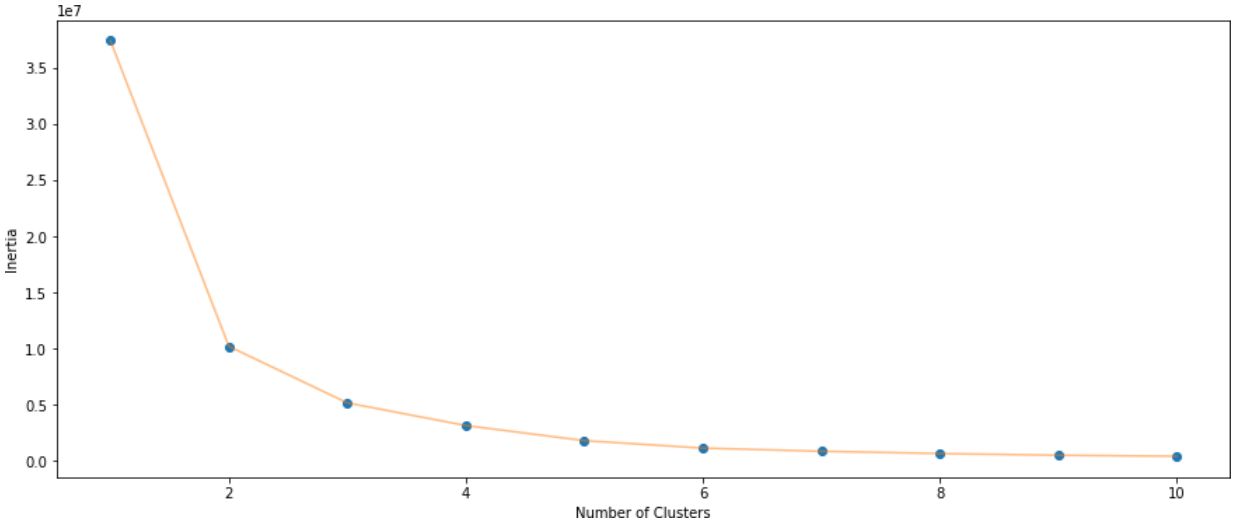


3.4.4 Recency & RFM Score

Similarly, we performed the k-means clustering for Recency, days since last purchase, and M Rank. By performing the same steps as 3.4.1, we found the following results.

```python
# Recency & RFM Score
X1 = rfm_analysis[['Days Since Last Purchase', 'RFM Score']].iloc[: , :].values
inertia = []
for n in range(1 , 11):
    algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init = 10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  , algorithm='elkan') )
    algorithm.fit(X1)
    inertia.append(algorithm.inertia_)

plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(1 , 11) , inertia , 'o')
plt.plot(np.arange(1 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Number of Clusters') , plt.ylabel('Inertia')
plt.show()
```

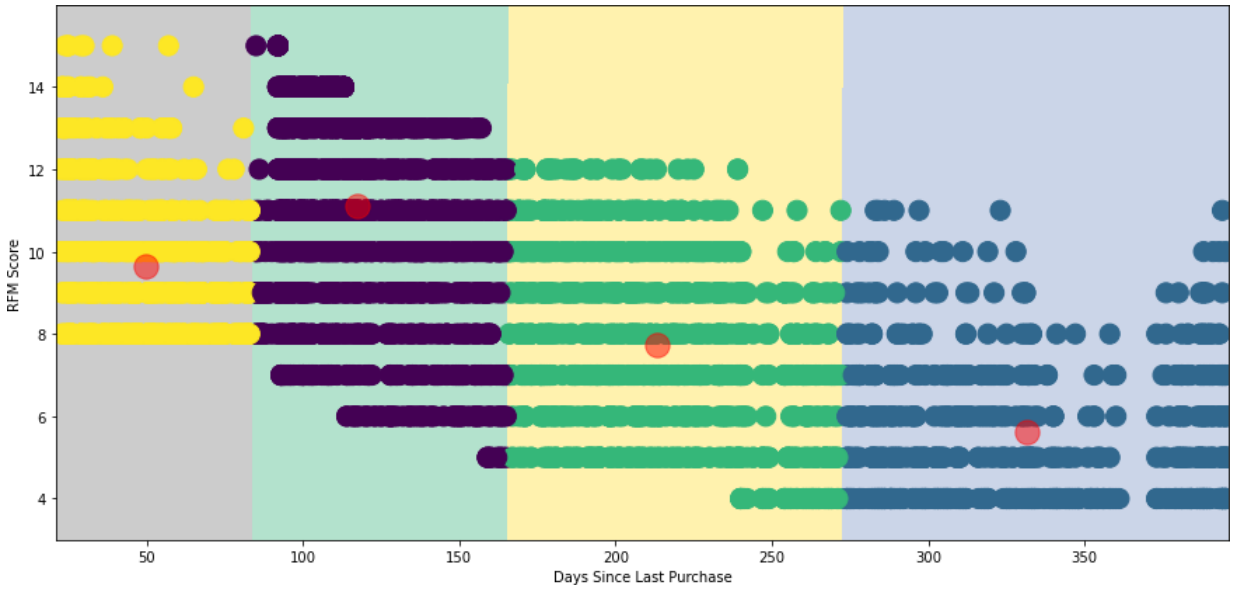



```python
# select 4 clusters
algorithm = (KMeans(n_clusters = 4 ,init='k-means++', n_init = 10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  , algorithm='elkan') )
algorithm.fit(X1)
labels1 = algorithm.labels_
centroids1 = algorithm.cluster_centers_

h = 0.02
x_min, x_max = X1[:, 0].min() - 1, X1[:, 0].max() + 1
y_min, y_max = X1[:, 1].min() - 1, X1[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])
```

```
# Recency & RFM Score
plt.figure(1 , figsize = (15 , 7) )
plt.clf()
Z = Z.reshape(xx.shape)
plt.imshow(Z , interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

plt.scatter( x = 'Days Since Last Purchase' ,y = 'RFM Score' , data = rfm_analysis, c = labels1 ,
            s = 200 )
plt.scatter(x = centroids1[: , 0] , y =  centroids1[: , 1] , s = 300 , c = 'red' , alpha = 0.5)
plt.ylabel('RFM Score') , plt.xlabel('Days Since Last Purchase')
plt.show()
```

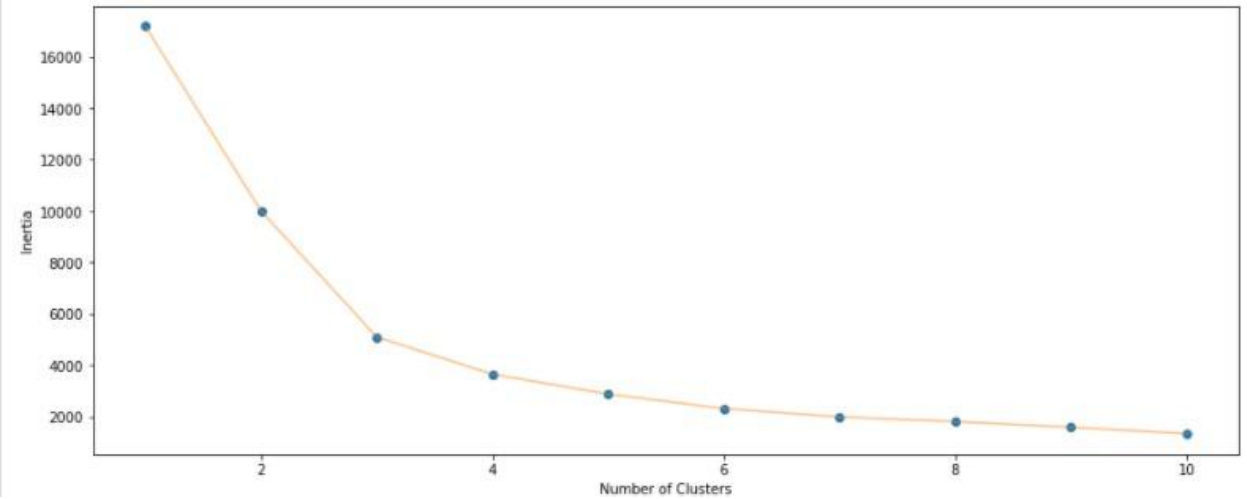


3.4.5 Recency Rank & Monetary Rank

We segmented the recency and monetary ranks for this segment. By performing the same steps as 3.4.1, we found the following results.

```
# Recency rank & Monetary Rank
X1 = rfm_analysis[['Recency Rank', 'Monetary Rank']].iloc[: , :].values
inertia = []
for n in range(1 , 11):
    algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init = 10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  , algorithm='elkan') )
    algorithm.fit(X1)
    inertia.append(algorithm.inertia_)

plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(1 , 11) , inertia , 'o')
plt.plot(np.arange(1 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Number of Clusters') , plt.ylabel('Inertia')
plt.show()
```

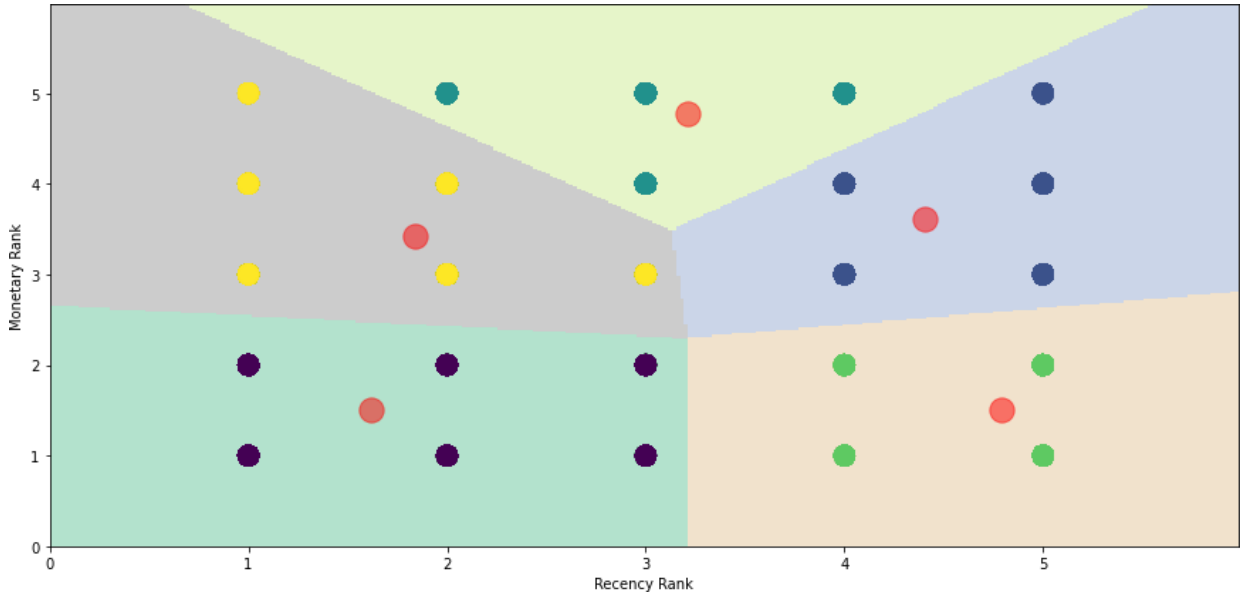


```
# select 5 clusters
algorithm = (KMeans(n_clusters = 5 ,init='k-means++', n_init = 10 ,max_iter=300,
                    tol=0.0001,  random_state= 111  , algorithm='elkan') )
algorithm.fit(X1)
labels1 = algorithm.labels_
centroids1 = algorithm.cluster_centers_

h = 0.02
x_min, x_max = X1[:, 0].min() - 1, X1[:, 0].max() + 1
y_min, y_max = X1[:, 1].min() - 1, X1[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])
```

```
# Recency rank & Monetary rank
plt.figure(1 , figsize = (15 , 7) )
plt.clf()
Z = Z.reshape(xx.shape)
plt.imshow(Z , interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

plt.scatter( x = 'Recency Rank' ,y = 'Monetary Rank' , data = rfm_analysis, c = labels1 ,
             s = 200 )
plt.scatter(x = centroids1[: , 0] , y =  centroids1[: , 1] , s = 300 , c = 'red' , alpha = 0.5)
plt.ylabel('Monetary Rank') , plt.xlabel('Recency Rank')
plt.show()
```

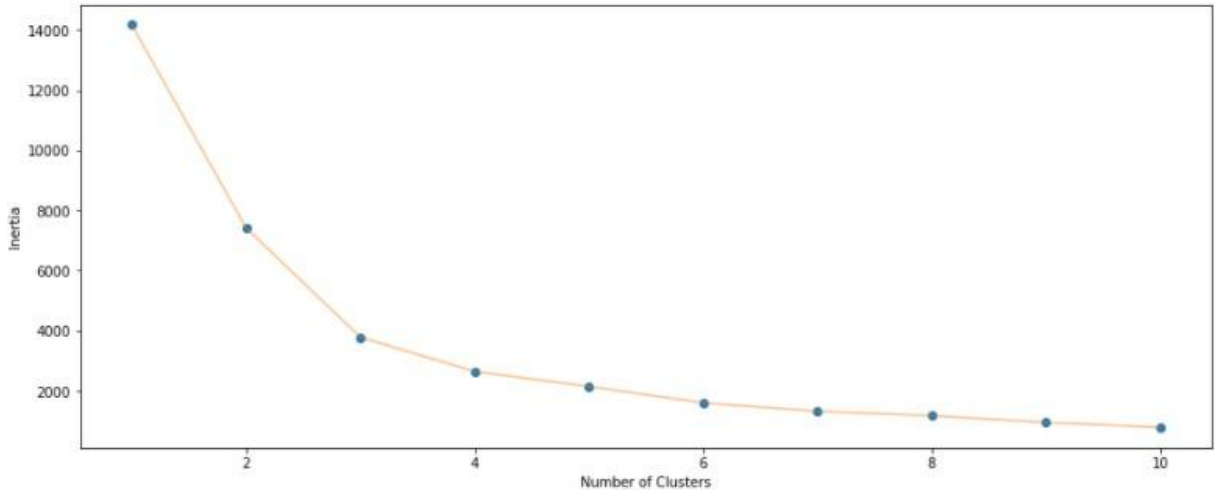


3.4.4 Recency Rank & Frequency Rank

Similarly, we performed the k-means clustering for Recency Rank and Frequency Rank. By performing the same steps as 3.4.1, we found the following results.

```
# Recency rank & Frequency Rank
X1 = rfm_analysis[['Recency Rank', 'Frequency Rank']].iloc[: , :].values
inertia = []
for n in range(1 , 11):
    algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init = 10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  , algorithm='elkan') )
    algorithm.fit(X1)
    inertia.append(algorithm.inertia_)

plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(1 , 11) , inertia , 'o')
plt.plot(np.arange(1 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Number of Clusters') , plt.ylabel('Inertia')
plt.show()
```

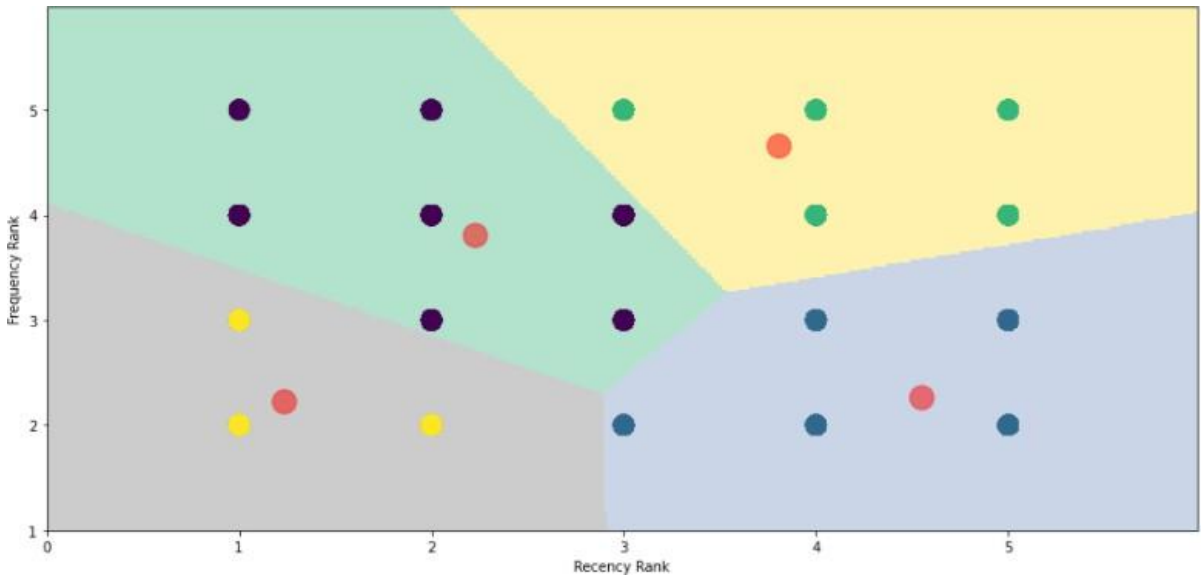


```
# select 4 clusters
algorithm = (KMeans(n_clusters = 4 ,init='k-means++', n_init = 10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  , algorithm='elkan') )
algorithm.fit(X1)
labels1 = algorithm.labels_
centroids1 = algorithm.cluster_centers_

h = 0.02
x_min, x_max = X1[:, 0].min() - 1, X1[:, 0].max() + 1
y_min, y_max = X1[:, 1].min() - 1, X1[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])

# Recency rank & Monetary rank
plt.figure(1 , figsize = (15 , 7) )
plt.clf()
Z = Z.reshape(xx.shape)
plt.imshow(Z , interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

plt.scatter( x = 'Recency Rank' ,y = 'Frequency Rank' , data = rfm_analysis, c = labels1 ,
            s = 200 )
plt.scatter(x = centroids1[: , 0] , y =  centroids1[: , 1] , s = 300 , c = 'red' , alpha = 0.5)
plt.ylabel('Frequency Rank') , plt.xlabel('Recency Rank')
plt.show()
```

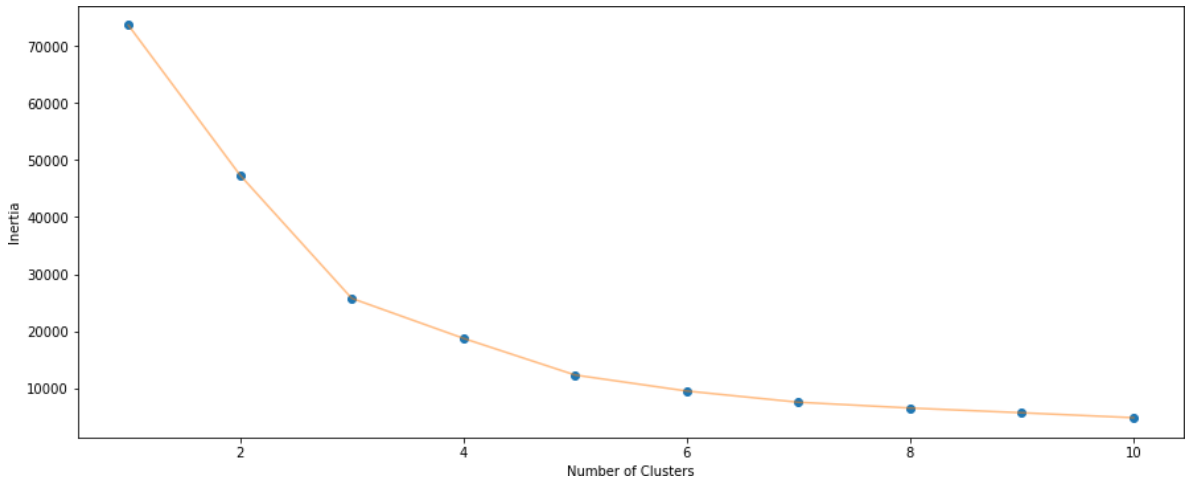


3.4.4 Recency Rank & Frequency Rank

Similarly, we performed the k-means clustering for Country Code and RFM. For each country, we set a Country Code since only numerical values can be plotted. Afterwards, we performed k-means algorithm for segmentation and found the following results.

```python
# Recency rank & Monetary Rank
X1 = rfm_analysis[['RFM Score', 'Country Code']].iloc[: , :].values
inertia = []
for n in range(1 , 11):
    algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init = 10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  , algorithm='elkan') )
    algorithm.fit(X1)
    inertia.append(algorithm.inertia_)

plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(1 , 11) , inertia , 'o')
plt.plot(np.arange(1 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Number of Clusters') , plt.ylabel('Inertia')
plt.show()
```

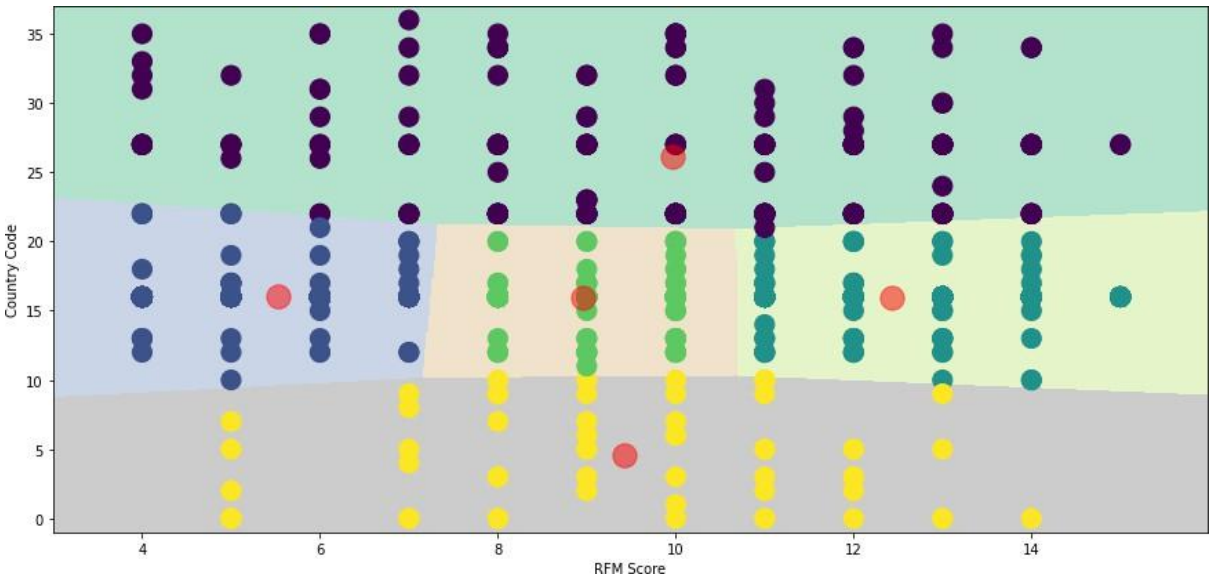


```
# select 5 clusters
algorithm = (KMeans(n_clusters = 5 ,init='k-means++', n_init = 10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  , algorithm='elkan') )
algorithm.fit(X1)
labels1 = algorithm.labels_
centroids1 = algorithm.cluster_centers_

h = 0.02
x_min, x_max = X1[:, 0].min() - 1, X1[:, 0].max() + 1
y_min, y_max = X1[:, 1].min() - 1, X1[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])

# Frequency rank & Monetary rank
plt.figure(1 , figsize = (15 , 7) )
plt.clf()
Z = Z.reshape(xx.shape)
plt.imshow(Z , interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

plt.scatter( x = 'RFM Score' ,y = 'Country Code' , data = rfm_analysis, c = labels1 ,
            s = 200 )
plt.scatter(x = centroids1[: , 0] , y =  centroids1[: , 1] , s = 300 , c = 'red' , alpha = 0.5)
plt.ylabel('Country Code') , plt.xlabel('RFM Score')
plt.show()
```



	Country	Country Code
0	Australia	0
1	Austria	1
2	Bahrain	2
3	Belgium	3
4	Brazil	4
5	Canada	5
6	Channel Islands	6
7	Cyprus	7
8	Czech Republic	8
9	Denmark	9
10	EIRE	10
11	European Community	11
12	Finland	12
13	France	13
14	Germany	14
15	Greece	15
16	Iceland	16
17	Israel	17
18	Italy	18
19	Japan	19
20	Lebanon	20
21	Lithuania	21
22	Malta	22
23	Netherlands	23
24	Norway	24
25	Poland	25
26	Portugal	26
27	RSA	27
28	Saudi Arabia	28
29	Singapore	29
30	Spain	30
31	Sweden	31
32	Switzerland	32
33	USA	33
34	United Arab Emirates	34
35	United Kingdom	35
36	Unspecified	36

CHAPTER 4: RESULTS

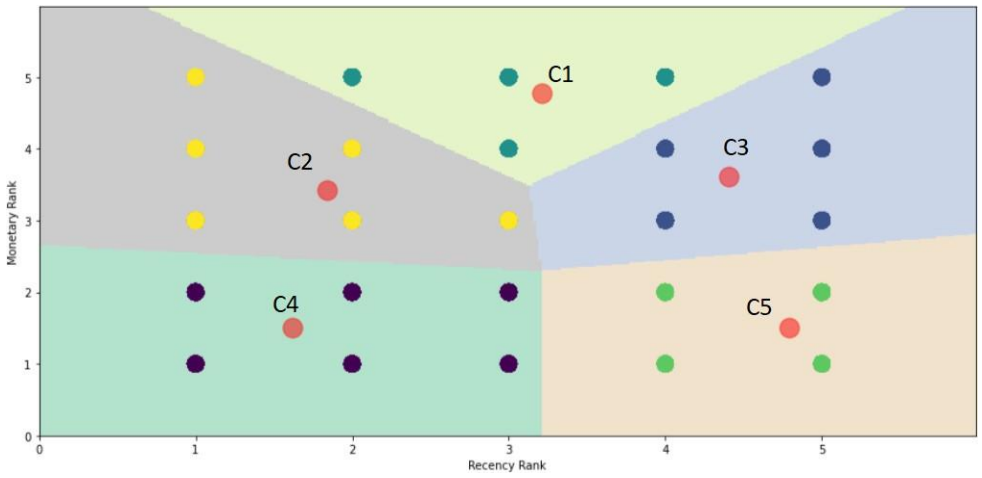
In this section, we will analyze the clusters and find probably marketing strategies that we can implement in our business.

4.1 Recency Rank & Monetary Rank

For targeted market analysis we have divided our customers into 5 parts. We found after the analysis, the following C1 are those customers who have high spending but their activity is average. On the other hand, C2 indicates high spending customers who are not active. Then C3 shows that the customers who have high frequency and also their spending is high. Again from the analysis, we have got C4 implies the customers who have low spending and are not active. And lastly, C5 gives us the idea of which customers are highly active with low spending.

In Summary:

1. C1- high spending customers with average activity
2. C2 - high spending customer but not active
3. C3 - highly frequent customer with high spending
4. C4 - low spending customer who are not active
5. C5 - highly active customer with low spending



Recency Rank (x-axis) & Monetary Rank (y-axis)

For the targeted market analysis, we have divided our customers into 5 parts. Firstly, those who are the customers of the C1 category, for them we will Advertise high priced products once in a while as their activity is average but spending rank is high. On the other hand, for the customers of the C2 category, we will create marketing campaigns to convert them into loyal customers as these users have a tendency to spend high but are not recent. Then for the customer of the C3 category which is the Best customer group we have got from the analysis. We will always include them when marketing high-priced products. Provide loyalty benefits. Again, for the category of C4, all Customers are at risk. So to avoid risk we will provide discounts or special offers to increase engagement. And Lastly C5 category customers we will Provide discounts or offers as these customers are highly active. We Can perform A/B testing for marketing campaigns of new products using these customers.

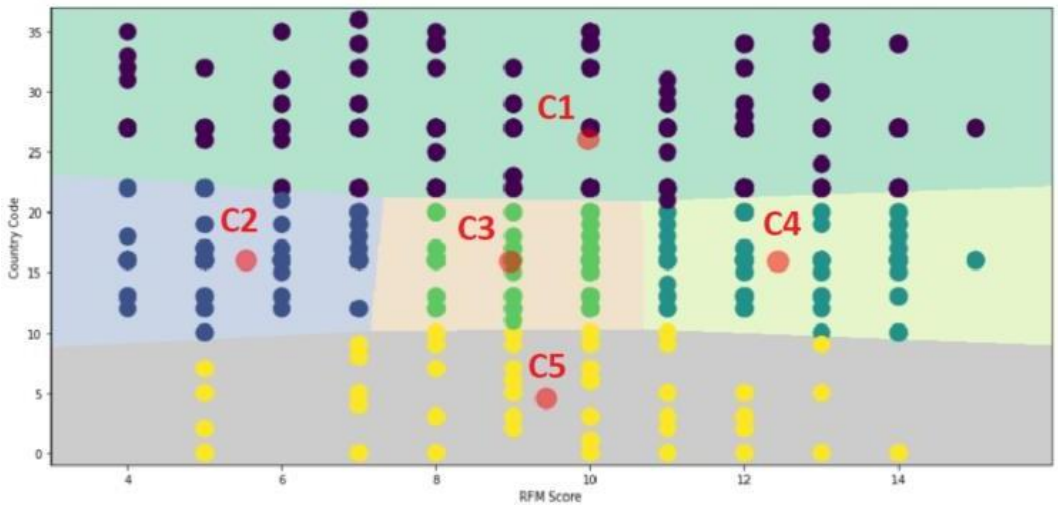
The following list summarizes the marketing strategies discussed.

1. C1- Advertise high priced products once in a while as their activity is average but spending rank is high
2. C2 - Must create marketing campaigns to convert them into loyal customers as these users have a tendency to spend high but are not recent.
3. C3 - Best customer group. Always include them when marketing high priced products. Provide loyalty benefits
4. C4 - Customers at risk. Provide discounts or special offers to increase engagement.
5. C5 - Provide discounts or offers as these customers are highly active. Can perform A/B testing for marketings campaigns of new products using these customers

4.2 Country & RFM Score

For the 36 countries mentioned in our dataset, we divided the groups into 5 clusters. From the results, we can see that C1 and C5 do not provide any useful results. This is because the values are distributed across the whole RFM score axis. However, for C2, C3 and C4, we can infer that

the best customer group is C4 and the customer's who are at risk are in C2. C3 falls in between these groups which means they are our average customer group.



In summary, we can make the following assumption about our customer groups.

1. C1 - Covers all RFM score range. This data is not helpful as we cannot differentiate whether the customers are in the higher RFM range or lower.
2. C2 - Need's attention. Try to create marketing campaigns targeted to gain these customers interest for your businesses.
3. C3 - Regularly send marketing content to these customers and try to improve their activity in your business. These customers can be converted into high value customers.
4. C4 - Best customer segment. Provide loyalty bonuses to maintain these customers in your business
5. C5 - Similar to C1, this cluster does not provide enough information as customers from these countries fall in all RFM score range.

CHAPTER 5: CONCLUSION

5.1 Conclusion

Customer segmentation is a method of improving customer relations by learning about the customer's wants and activities so that appropriate communication can be established. To increase sales, customer segmentation was needed to get potential customers used to the product. Potential consumer data, such as ecommerce platforms such as media purchasing and selling online, can be used to deliver services tailored to the needs of customers. Customer segmentation is an operation that divides customers or items into classes that have similar characteristics. This project explores many components to do customer segmentation, including identifying customer habits to develop targeted marketing to increase conversion rates for e-commerce businesses. For this, we used an RFM Analysis, which shows the number of days after a customer's last transaction, the total number of invoices he or she has received, and the total amount spent by a customer. In order to find patterns in a non-labeled dataset, unsupervised machine learning must be used. To classify the data, we used k-means. Internal data and external data are also needed for customer segmentation. Customers may be segmented in a variety of ways. Defining business objectives, gathering data, data planning, evaluating variables, data analysis, and performance assessment are all steps in the Customer Segmentation process.

References

1. 4 Types of Market Segmentation With Real-World Examples | Yieldify. (2021). Retrieved 20 May 2021, from https://www.yieldify.com/blog/types-of-market-segmentation/

2. 21 Customer Segmentation Examples for eCommerce Excellence. (2021). Retrieved 20 May 2021, from https://blog.crobox.com/article/customer-segmentation-examples

3. (2021). Retrieved 20 May 2021, from https://www.researchgate.net/publication/326706602_Approaches_to_Clustering_in_Customer_Segmentation

4. Academy, T., Hassan?, W., Alaa, B., Challenge, B., Sell, F., & Sells, C. et al. (2021). The 10 Most Common Challenges Faced by eCommerce Businesses. Retrieved 20 May 2021, from https://ecommercetrainingacademy.com/ecommerce-business-challeges/

5. Clustering algorithms for customer segmentation. (2021). Retrieved 20 May 2021, from https://towardsdatascience.com/clustering-algorithms-for-customer-segmentation-af637c6830ac

6. Customer Segmentation Analysis with Python. (2021). Retrieved 20 May 2021, from https://towardsdatascience.com/customer-segmentation-analysis-with-python-6afa16a38d9e

7. Customer segmentation: How machine learning makes marketing smart. (2021). Retrieved 20 May 2021, from https://bdtechtalks.com/2020/12/28/machine-learning-customer-segmentation/

8. Inside Recency, Frequency, Monetary Value (RFM). (2021). Retrieved 20 May 2021, from https://www.investopedia.com/terms/r/rfm-recency-frequency-monetary-value.asp

9. k-Means Advantages and Disadvantages | Clustering in Machine Learning. (2021). Retrieved 20 May 2021, from https://developers.google.com/machine-learning/clustering/algorithm/advantages-disadvantages

10. K-means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks. (2021). Retrieved 20 May 2021, from https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a

11. Python, R. (2021). K-Means Clustering in Python: A Practical Guide – Real Python. Retrieved 20 May 2021, from https://realpython.com/k-means-clustering-python/

12. RFM Analysis for Customer Segmentation | Clever Tap. (2021). Retrieved 20 May 2021, from https://clevertap.com/blog/rfm-analysis/?fbclid=IwAR24iAh4Qz7qxanTZFp5pmnOX9ourJHNTkpwxL-9Jaipf2SN4YTpdqSdjGU

13. Shaw, N. (2021). Machine Learning For Ecommerce: How Does it Work?. Retrieved 20 May 2021, from https://www.bigcommerce.com/blog/ecommerce-machine-learning/#use-cases-for-ecommerce-machine-learning

14. The growth of e-commerce during the pandemic in Bangladesh. (2021). Retrieved 20 May 2021, from https://www.newagebd.net/article/114200/the-growth-of-e-commerce-during-the-pandemic-in-bangladesh

15. UCI Machine Learning Repository: Online Retail II Data Set. (2021). Retrieved 20 May 2021, from https://archive.ics.uci.edu/ml/datasets/Online+Retail+II