Pattern Recognition and Machine Learning

Minor Project Report

Roll Number	B21CS092, B21CS093, B21CS094
Name	G Mukund, Shashank Shekhar Asthana, Shreshth Vatsal Sharma
Project Number	04
Lab Title	Online Retail Data Analysis

Problem Statement

A company that sells some of the product, and you want to know how well the selling performance of the product. You have the data that we can analyze, but what kind of analysis can we do? Well, we can segment customers based on their buying behavior on the market. Your task is to classify the data into the possible types of customers that the retailer can encounter.

Introduction

The report describes how a company selling products can segment its customers based on buying behavior and geographic location. The purpose of this report is to provide an overview of the analysis performed on customer data for a company that sells products. To achieve this, we have analyzed customer data using a customer segmentation technique called RFM analysis based on their buying behavior in the market.. The report explains how the RFM analysis works, how the data was preprocessed, and how the RFM analysis results were interpreted to create a segmented RFM table. Another strategy we used on the project was categorizing clients according to their frequency, monetary values as features, and countries as class labels.

We have a Random Forest Classifier, SVD and KNN Classifier, PCA and Decision Tree Classifier, LDA and KNN Classifier, and PCA and Decision Tree Classifier to implement and evaluate classification systems.

Data Pre-processing

The key steps that we took in pre-processing were-

- Mounting Google Drive: The first step in the code is to mount the Google Drive account. This is done using the "drive.mount('/content/drive')" command in Google Colab. This step is necessary to access the retail dataset stored in the Google Drive.

- Importing libraries: The next step is to import necessary libraries such as NumPy, Pandas, warnings, datetime, Matplotlib, and scikit-learn modules

- Loading the dataset: The retail dataset is loaded into a Pandas DataFrame using the "pd.read_excel()" function. The dataset is stored in an Excel file named "Online Retail.xlsx" and is located in the Colab Notebook's directory on Google Drive.

- Copying the dataset: A copy of the original dataset is made using the assignment operation "df1 = df". This step is done to keep a backup of the original dataset, in case any changes are made during preprocessing.

- Exploratory Data Analysis (EDA): The dataset is analyzed to gain insights into its structure and contents. The following EDA steps are performed:

- The number of unique countries in the dataset is found using the "nunique()" function, which returns the count of distinct values in a column. In this case, it is the count of unique countries.

- The unique countries in the dataset are displayed using the "unique()" function, which returns an array of unique values in a column.

- The number of customers from each country is computed using the "groupby()" function, which groups the data by a specified column and returns aggregate statistics. Here, the "count()" function is used to count the number of CustomerIDs for each country.

- The result is sorted in descending order based on the number of customers using the "sort_values()" function.

This enabled us to get the countries with the largest number of Customer ID, thus we can observe which countries have the maximum number of customers of our company and what type of analysis we can do for that country.

United Kingdom

The analysis was performed on customer data for the United Kingdom. More than 90% of the customers in the data are from the United Kingdom. There's some research indicating that customer clusters vary by geography.

There were 133,600 missing values in the CustomerID column, and these were removed using the isnull() function since the analysis is based on customers. The data was cleaned to remove negative values in the Quantity column. After cleaning up, 354,345 rows and 9 columns were left in the dataset.

Then we added a column for total price by multiplying the unit price column with the quantity column.

RFM Analysis:

A Recency-Frequency-Monetary value (RFM) table was created to segment customers based on their behavior. RFM is a customer segmentation technique that groups customers based on their purchase history.

We used the RFM (Recency - Frequency, - Monetary Value) framework to segment the customers. The RFM analysis is a method used to evaluate customer value based on three criteria:

Recency - How recently has the customer made a purchase?

Frequency - How often does the customer make purchases?

Monetary Value - How much money does the customer spend on purchases?

We calculated the RFM metrics for each customer using the following steps:

- We calculated the recency, frequency, and monetary value for each customer.
- We split the metrics into segments using tertiles, dividing the data into three equal divisions.
- We assigned a score to each segment based on the tertile range.

Segmentation:

We used the scores to create segments for each customer. The thought process that we put behind the segments is based on the RFM scores and is as follows:

- Best Customers - High monetary value, high frequency, and low recency (111)
- Loyal Customers - High monetary value, high frequency, and high recency (113)
- Potential Loyalist - High monetary value, low frequency, and low recency (131)
- Big Spenders - High monetary value, low frequency, and high recency (133)
- Almost Lost - Low monetary value, high frequency, and high recency (313)
- Lost Customers - Low monetary value, low frequency, and high recency (333)
- Low-Value Customers - Low monetary value, low frequency, and low recency (331)

We used the scores to create segments for each customer. The segments are based on the RFM scores and are categorized as follows:

- WholeSaler or Corporate Gifter

```
if (rfm_arr_score[i] == 231 or rfm_arr_score[i]== 131 or rfm_arr_
score[i]== 331)
```

- Regular customer with interest towards affordable gifting

```
if (rfm_arr_score[i] == 113 or rfm_arr_score[i]== 213)
```

- Regular Customer with high demand of expensive gifts

```
if (rfm_arr_score[i] == 111 or rfm_arr_score[i]== 211) :
```

- Moderate Customer with moderate level demands

```
if rfm_arr_score[i] ==  122 or rfm_arr_score[i]== 222
```

- Potential emerging customer, inprofitable right now but can be profitable in coming times

```
if rfm_arr_score[i] == 133 or rfm_arr_score[i] == 233
```

- Regular customer with high demand of average price gifts

```
if rfm_arr_score[i] == 212 or rfm_arr_score[i]== 112
```

- Emerging Customer, profitable with moderate buying demands

```
if rfm_arr_score[i] == 132 or rfm_arr_score[i]== 232 or rfm_arr_score[i
]== 123 or rfm_arr_score[i]== 223
```

- Retailers or the ones with shops in city who buy weekly or monthly to run their shops

```
if rfm_arr_score[i] == 121 or rfm_arr_score[i]== 221
```

- Else Nearly lost or inactive customers, not bought anything since a long time

We then implemented a function named "plot," which we used to visualize the results of a customer segmentation analysis. The function takes a segmented_rfm dataframe as input, which contains the RFM scores and labels for each customer. The RFM scores are used as features for the clustering algorithm, while the labels are used to determine the number of clusters.

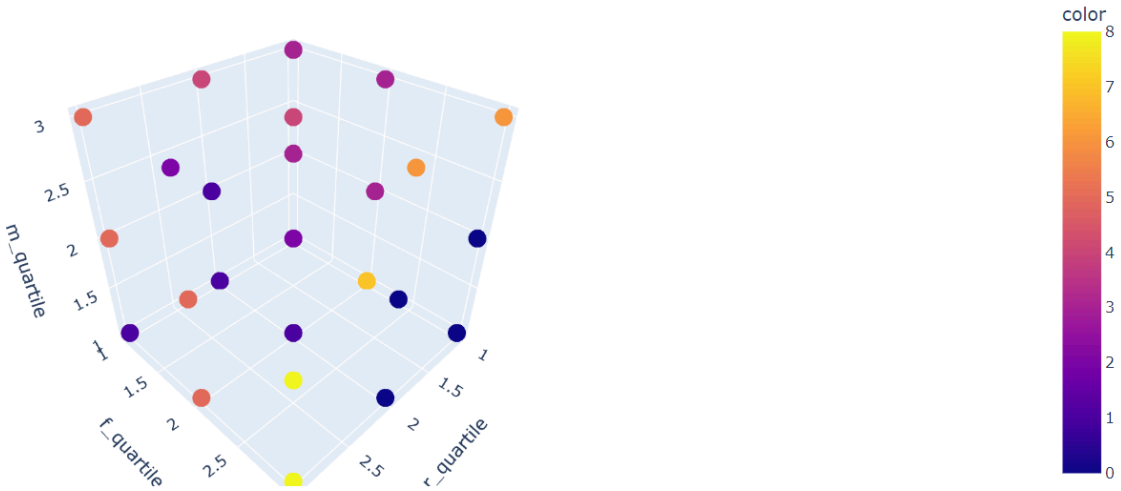
The function first separates the labels and features from the input dataframe. It then initializes a KMeans clustering algorithm with the number of clusters equal to the number of unique labels in the input dataframe. The algorithm is fit to the features using the fit() method of the KMeans class, and the predicted cluster labels are obtained using the labels_ attribute of the fitted KMeans object.

The function then creates a 3D scatter plot using the px.scatter_3d() function from the Plotly Express library. The x, y, and z axes of the scatter plot correspond to the three RFM scores, and the color of each point corresponds to the predicted cluster label.

Finally, the function returns the scatter plot using the show() method of the fig object.

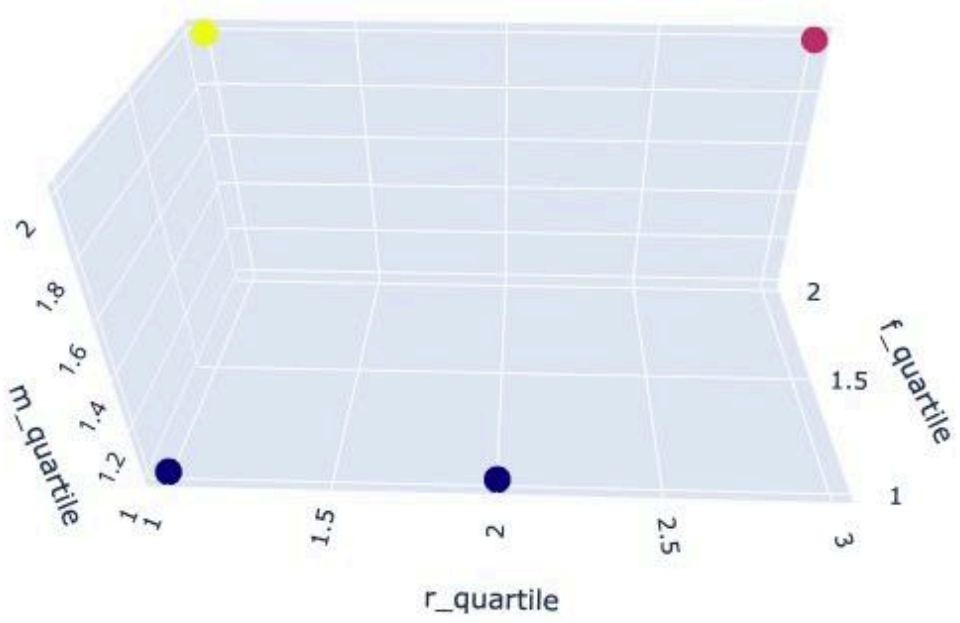
Overall, the plot function provided a useful way to visualize the results of a customer segmentation analysis, allowing us to quickly identify different customer segments based on their RFM scores.

The final plot for **United Kingdom** was-



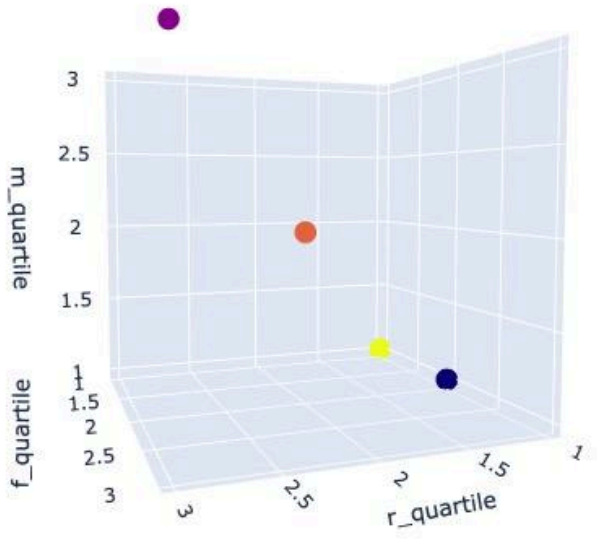
Germany

Performing the similar process for Germany as well we get the plot as-



France

Performing the similar process for France as well we get the plot as-



In the COLAB file, the plots of rest of the countries are displayed. You may not be able to see all of the plots owing to network or memory problems, but by running the same code block on fewer countries, we are able to obtain all of the plots for each country and, as a result, customer types.

Implementation of Classifiers

We made the decision to classify consumers based on their recentness, frequency, and monetary values, which are the model's features, and their respective countries, which are the class labels. What we intended to do was segment clients so that businesses could predict. For example, if a business receives a dataset of customers and wants to know where country each customer is from, our model may assist the business in making the same determination.

We started off by copying the dataset and made a dataframe which we would be using further for preprocessing, we dropped the lower column since it's values were of no use to us while learning our model and we would not obtain efficient information by keeping them

Then, we dropped the null values of customer Ids since we are learning our model for each customer and if customer if itself is a null value, then it is of no point to us

Then, what we did was some preprocessing as we did earlier, slicing the data frame for quantity greater than zero, changing the format of invoice date time, and adding a Total Price column

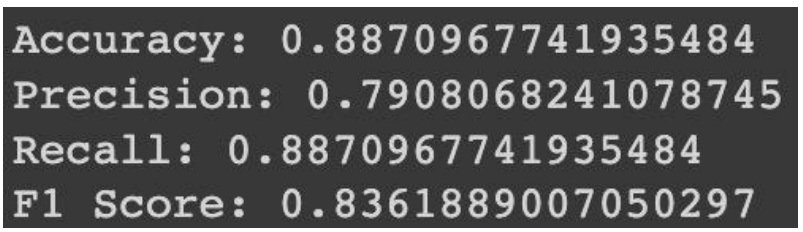
Then, we made a similar RFM Table that we made earlier in our first approach towards project along with the respective country the customer belongs to so that we obtain the required dataframe at last upon which we would be implementing the classifiers

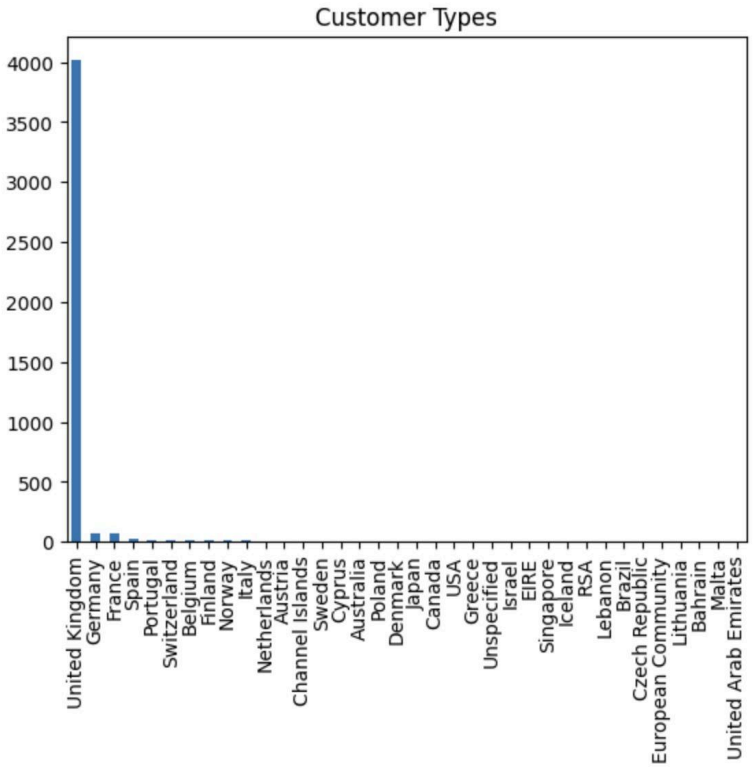
Then I split the data set into training and testing sets, with features being **recency, frequency, monetary value,** and class being **Country**

Then onwards, I implemented several classifiers that we have been taught in this course and obtained their accuracies, precision scores, recall scores and F1 scores

The same has been shown below, respectively :

- Random Forest Classifier :





- SVD and KNN Classifier :

```
Accuracy: 0.8847926267281107
Precision Score: 0.7905785970302098
Recall: 0.8847926267281107
F1 Score: 0.8350365621443783
```

- PCA and Decision Tree Classifier :

```
Accuracy: 0.805299539170507
Precision Score: 0.793512825637167
Recall: 0.805299539170507
F1 Score: 0.799333530023115
```

- LDA and KNN Classifier :

```
Accuracy: 0.805299539170507
Precision Score: 0.793512825637167
Recall: 0.805299539170507
F1 Score: 0.799333530023115
```

Conclusion

In conclusion, we have successfully segmented the customers based on their buying behavior and geographic location. The segmentation will help the company tailor its marketing efforts based on the customers' preferences and increase customer retention. It will also provide insights into the customers' behavior and help the company identify the high-value customers that require special attention. As determined by our machine learning model, the company can also start and carry out efficient marketing for the clients it may be losing. As a result, the model may be advantageous to the business.

Another strategy we used involved the company using our model to identify the country that a consumer comes from based on their recent behavior, frequency, and monetary values.

The SVD-KNN classifier and the Random Forest classifier both did well on this dataset with higher frequencies than the competitors; therefore, they can be used to continue learning our model for future requirements.

THANK YOU !