

CUSTOMER SEGMENTATION ON CLUSTERING ALGORITHMS

BY

TOH WEI XUAN

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology

(Kampar Campus)

JAN 2023

REPORT STATUS DECLARATION FORM

Title: CUSTOMER SEGMENTATION ON CLUSTERING ALGORITHMS_

Academic Session: January 2023

I _____ TOH WEI XUAN _____
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.



(Author's signature)

Verified by,



(Supervisor's signature)

Address:

7, Laluan Keledang Utara 2,
Taman Wang,
31450 Menglembu, Perak

Lim Jia Qi
Supervisor's name

Date: 25/4/2023

Date: ___28/04/2023_____

Universiti Tunku Abdul Rahman			
Form Title : Sample of Submission Sheet for FYP/Dissertation/Thesis			
Form Number: FM-IAD-004	Rev No.: 0	Effective Date: 21 JUNE 2011	Page No.: 1 of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
UNIVERSITI TUNKU ABDUL RAHMAN

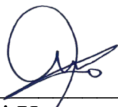
Date: 1/April/2023

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that Toh Wei Xuan (ID No: 19ACB03568) has completed this final year project/ dissertation/ thesis* entitled “_Customer Segmentation on Clustering Algorithms_” under the supervision of _Dr Lim Jia Qi_ (Supervisor) from the Department of Computer Science, Faculty of Information and Communication Technology, and Dr Kh'ng Xin Yi (Co-Supervisor)* from the Department of Computer Science, Faculty of Information and Communication Technology.

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,



Toh Wei Xuan

*Delete whichever not applicable

DECLARATION OF ORIGINALITY

I declare that this report entitled “Customer Segmentation on Clustering Algorithms” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature :  _____

Name : Toh Wei Xuan

Date : 1 April 2023

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude and appreciation to my supervisor, Dr. Lim Jia Qi and my moderator, Dr Kh'ng Xin Yi, for their invaluable guidance, support, and encouragement throughout my journey in the field of Machine Learning. Without their expert knowledge and unwavering commitment to my development, I would not have been able to complete this project successfully.

Their continuous support and advice have been instrumental in overcoming the challenges I faced during this project. Their unwavering belief in my abilities and willingness to share their knowledge and expertise have been invaluable in shaping my understanding of the subject matter.

Once again, I would like to express my heartfelt thanks to my supervisor and moderator for their golden opportunity, and for the immense impact they have had on my academic and professional growth. I will forever cherish the experience and knowledge gained under their mentorship.

ABSTRACT

This report presents an analysis of customer segmentation using various clustering algorithms, including k-means, DBSCAN, GMM, and RFM. The aim of the study is to identify customer groups based on their buying behaviour and demographic characteristics. The study utilizes a dataset consisting of transactional and demographic data of customers from an e-commerce company.

Firstly, descriptive analysis is performed to explore the characteristics of the dataset. Then, k-means, DBSCAN, and GMM clustering algorithms are applied to segment customers based on their buying behaviour. Finally, RFM (Recency, Frequency, Monetary) analysis is used to segment customers based on their purchasing history.

The results show that all clustering algorithms were able to identify distinct customer groups with varying characteristics. Furthermore, the RFM analysis was able to segment customers based on their buying patterns, and provide insights into their behaviour.

Overall, the study demonstrates the effectiveness of different clustering algorithms and RFM analysis in identifying customer segments. The insights gained from this study could potentially be used by the e-commerce company to improve their marketing strategies and customer engagement.

TABLE OF CONTENTS

<i>Title Page</i>	<i>i</i>
REPORT STATUS DECLARATION FORM	ii
DECLARATION OF ORIGINALITY	iv
ACKNOWLEDGEMENTS	v
ABSTRACT	vi
TABLE OF CONTENTS	vii
LIST OF FIGURES	x
LIST OF TABLES	xii
LIST OF SYMBOLS	xiii
LIST OF ABBREVIATIONS	xiv
1 Introduction	1
1.1 Background Information	1
1.1.1 Customer Segmentation	2
1.1.2 Clustering	4
1.2 Problem Statement	6
1.2.1 Selection of sample size, features, and variable	6
1.2.2 Data cleaning and preprocessing	6
1.2.3 Use appropriate clustering algorithms	6
1.2.4 Calculate optimal number of clusters.	7
1.2.5 Validation test for clustering	7
1.2.6 Cluster interpretation and description	7
1.3 Motivation	8
1.4 Objective	8
1.5 Project Scope and Direction	9
1.6 Contributions	10
1.7 Report Organization	11

2	<i>Literature Review</i>	12
2.1	Previous works on customer segmentation	12
2.1.1	Clustering techniques for customer segmentation	13
2.2	Limitation of previous study	14
3	<i>System Methodology</i>	16
3.1	Data Pre-processing	16
3.1.1	Data Import	16
3.1.2	Data Cleaning	18
3.1.3	Feature Engineering	20
3.1.4	Data Scaling and Encoding	21
3.1.5	Dimensionality Reduction	22
3.2	Performance Metric (clarify internal and why not external)	24
3.3	Selection of optimal number of clusters, k	25
3.4	Clustering Algorithms	26
3.4.1	K-means Clustering	27
3.4.2	GMM Clustering	29
3.4.3	Dbscan Clustering	32
3.4.4	RFM+k-means Clustering	34
4	<i>Experiment/Simulation</i>	38
4.1	Hardware	38
4.2	Software	38
4.2.1	Jupyter Notebook (Python3)	38
4.2.2	Anaconda Prompt	39
4.3	Kmeans extension	40
4.4	GUI	41
5	<i>Model Evaluation and Discussion</i>	43
5.1	Determine optimal number of clusters	43
5.1.1	k-means, GMM, DBSCAN	43
5.1.2	RFM+kmeans	44

5.2	Clustering Evaluation	45
5.2.1	k-means and GMM	45
5.2.2	DBSCAN	46
5.2.3	3D Plot Performance	47
5.3	Cluster Analysis (k-means, GMM and DBSCAN)	49
5.3.1	Cluster Analysis (RFM)	53
6	Conclusion and Recommendations	61
6.1	Summary	61
6.2	Further Study	63
	REFERENCES	64
	<i>Appendix</i>	67
	FINAL YEAR PROJECT WEEKLY REPORT	86
	POSTER	96
	PLAGIARISM CHECK RESULT	97
	<i>Checklist FYP2</i>	<i>Error! Bookmark not defined.</i>

LIST OF FIGURES

<i>Figure 1.1 Targeted market segmentation phrases</i>	2
<i>Figure 1.2 Unlabeled examples grouped into three cluster.</i>	4
<i>Figure 1.3 An Overview of clustering taxonomy</i>	5
<i>Figure 3.1 Feature name with Datatype</i>	17
<i>Figure 3.2 Schematic diagram of Data cleaning</i>	18
<i>Figure 3.3 Detect Outliers</i>	20
<i>Figure 3.4 Drop Outliers</i>	21
<i>Figure 3.5 PCA with 3 components</i>	23
<i>Figure 3.6 k-means output</i>	29
<i>Figure 3.7 GMM output</i>	31
<i>Figure 3.8 DBSCAN output</i>	34
<i>Figure 3.9 RFM Mapping</i>	35
<i>Figure 3.10 RFM output</i>	37
<i>Figure 5.1 Silhouette score and Calinski-Harabaz</i>	43
<i>Figure 5.2 Comparison of all metric prediction</i>	44
<i>Figure 5.3 Elbow Method for RFM</i>	44
<i>Figure 5.4 Silhouette Score for RFM</i>	45
<i>Figure 5.5 Clustering Metrics Summary for k-means and GMM</i>	45
<i>Figure 5.6 GridSearch for eps</i>	46
<i>Figure 5.7 Clustering Metrics Summary for DBSCAN</i>	47
<i>Figure 5.8 Comparison between kmeans, GMM and DBSCAN</i>	48
<i>Figure 5.9 3D plot for RFM</i>	49
<i>Figure 5.10 Tabulate total number of clusters</i>	49
<i>Figure 5.11 Age vs spending</i>	50
<i>Figure 5.12 Education vs Spending</i>	50
<i>Figure 5.13 NumwebvisitsMonth vs Spending</i>	51
<i>Figure 5.14 Marital Status vs Spending</i>	51
<i>Figure 5.15 Familysize vs Spending</i>	52
<i>Figure 5.16 Tabulate number of customers in each cluster</i>	53
<i>Figure 5.17 Summary boxplot</i>	53

<i>Figure 5.18 RFM distribution</i>	55
<i>Figure 5.19 RFM Age vs Spending</i>	56
<i>Figure 5.20 RFM Income vs Spending</i>	56
<i>Figure 5.21 RFM Education vs Spending</i>	57
<i>Figure 5.22 RFM Education vs Spending</i>	57
<i>Figure 5.23 RFM Numwebvisitmonth vs Spending</i>	58
<i>Figure 5.24 RFM Familysize vs Spending</i>	58
<i>Figure 5.25 RFM Days_enrolled vs Spending</i>	59

LIST OF TABLES

Table Number	Title	Page
Table 1.1	Types of Customer Segmentation	3
Table 2.1	Comparative study for performance metric by authors	14
Table 3.1	Formulas for various performance metrics	24
Table 4.1	Spec Used	38

LIST OF SYMBOLS

k	Number of cluster / components
ϵ	epsilon

LIST OF ABBREVIATIONS

<i>PCA</i>	Principal Component Analysis
<i>GMM</i>	Gaussian Mixture Model
<i>DBSCAN</i>	Density-Based Spatial Clustering of Applications with Noise
<i>RFM</i>	Recency, Frequency and Monetary
<i>MICE</i>	Multiple Imputations by Chained Equations
<i>AIC</i>	Akaike Information Criterion
<i>BIC</i>	Bayesian information criterion
<i>SOM</i>	Self-Organizing Map
<i>t-SNE</i>	t-Distributed Stochastic Neighbor Embedding
<i>CRM</i>	Customer Relationship Management
<i>GUI</i>	Graphical User Interface
<i>EM</i>	Expectation-Maximization

1 Introduction

1.1 Background Information

In today's competitive market, businesses are constantly looking for ways to gain a competitive advantage. One such way is by understanding their customers better and tailoring their marketing strategies to meet their needs. Customer segmentation is a powerful tool that can help businesses achieve this goal.

Customer segmentation involves dividing customers into groups based on common characteristics such as demographics, buying behavior, and preferences. These groups can then be targeted with tailored marketing messages and offers, leading to improved customer engagement and loyalty.

Cluster analysis is a popular technique for customer segmentation, and it involves grouping customers based on their similarity in terms of certain attributes. There are several clustering algorithms that can be used for this purpose, including k-means, DBSCAN, and GMM. Each algorithm has its own strengths and weaknesses, and the choice of algorithm depends on the nature of the data and the research question.

RFM analysis is another popular technique for customer segmentation, and it involves analyzing customers based on their purchasing behavior. The analysis involves three metrics: recency, frequency, and monetary value, which are used to segment customers into different groups based on their purchasing patterns.

In this report, we apply several clustering algorithms and RFM analysis to segment customers of an e-commerce company based on their transactional and demographic data. The aim of the study is to identify distinct customer groups with unique characteristics, and to provide insights into their behavior. The results of this study could potentially be used by the e-commerce company to improve their marketing strategies and customer engagement [7].

1.1.1 Customer Segmentation

Customer segmentation is a vital aspect of customer relationship management (CRM) and business intelligence. By dividing customers into groups based on shared characteristics such as behavior, age, gender, education, location, and socioeconomic status, businesses can gain insights into customer behavior, tailor marketing messages to specific segments, and improve customer engagement and retention. Marketing personas, which personify a customer segment, are often created from customer segmentation data and must be tightly matched to customer categories for effective results. However, while customer segmentation can be a powerful tool, it also has its limitations, such as the challenge of maintaining segmentation as customer behavior changes over time. By understanding and implementing different types of customer segmentation techniques, businesses can potentially improve their marketing campaigns, target specific customer groups, and increase sales.

Customer segmentation is a vital component of business intelligence that enables organizations to create, maintain, and grow long-term customer relationships. Traditional marketer-designed segmentation models have certain limitations and may not fully uncover insights and data patterns. The following stage is to put in place efficient campaign management. The steps utilized for targeted consumer segmentation are depicted in Figure 1.1 [9].

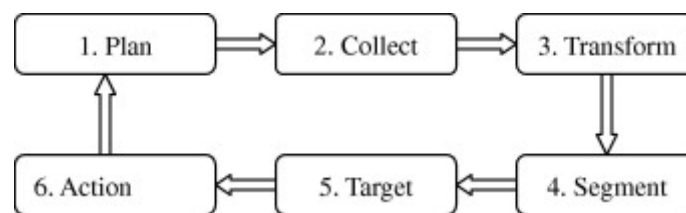


Figure 1.1 Targeted market segmentation phrases

While traditional demographic characteristics are no longer sufficient for market segmentation, non-demographic characteristics such as values, interests, and preferences are critical to a successful marketing plan. For example, all the subcategories within the demographic segment are comparable to one another, providing businesses with the data they want to accurately target a certain client base with specific marketing strategies. All the customer segmentation dimensions are summarized in Table 1.1 [7].

Segmentation	Variables
Demographic	Gender
	Age
	Occupation
	Education
	Generation
	Family Size
	Family life Cycle
	Ethnicity
	Religion
	Nationality
	Social class
Behavioral	Occasions
	Rate of usage
	Brand loyalty
	Status of user
	Benefits desired
	Readiness to buy
Psychographic	Opinions
	Attitudes
	Activities
	Values
	Interest
Geographic	Climate
	Religion
	Area Size
	Population density

Table 1.1 Types of Customer Segmentation

Therefore, a mix of attributes from different dimensions of segmentation is essential for developing a robust customer segmentation model. Overall, the ultimate purposes of

customer segmentation are to maximize customer value to the company, optimize marketing strategies, and improve customer experience.

1.1.2 Clustering

Clustering is a form of unsupervised learning strategy in the field of machine learning. Unsupervised learning approach aims to discover intrinsic data patterns and actionable insights from data sets that do not have a labelled output variable. It's a type of exploratory data analysis which involves grouping data sets into a specified number of clusters with comparable features among the data points inside each cluster as shown in Figure 1.2 [9].

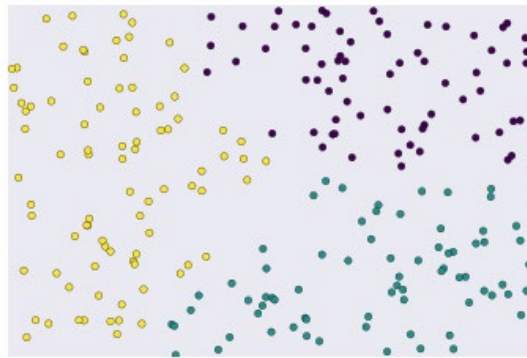


Figure 1.2 Unlabeled examples grouped into three cluster.

Clusters are made up of data points that are grouped together in such a way that the distance between them is kept to a minimum. In other words, clusters are areas with a high density of similar data points [9].

Clustering, also known as cluster analysis, is a vital data analysis technique and a field within data mining. It involves grouping the dataset into clusters, where data points with similar attributes are placed in the same group or cluster. The aim is to identify connections between data points based on their raw data qualities. However, the main challenge is to determine the appropriate number of clusters that are relevant and informative for analysis. This is a systematic and iterative process that involves analyzing large volumes of raw data for patterns and commonalities. The disorganized data is sifted for meaningful insights, and then data points are assigned to clusters. To achieve good results in a market domain, a particular clustering technique coupled with detailed examination of cluster properties based on the clustering results may be ideal.

These are several clustering algorithms that differ from one another in terms of the method they take to group items according to their attributes. Figure 1.3 [11] provides the overview of clustering types following three different classes of categorization in this project.

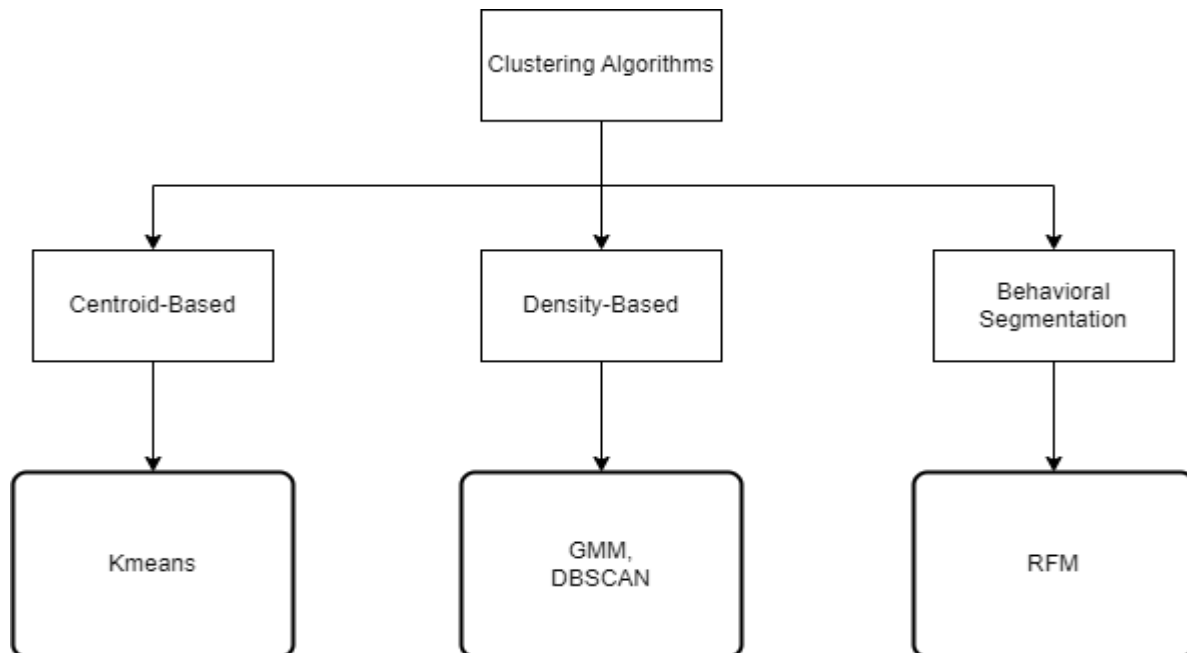


Figure 1.3 An Overview of clustering taxonomy

In summary, clustering algorithms identify internally homogeneous and externally diverse groupings. Customers have distinct behaviors, requirements, needs, and traits, and the main aim of clustering is to identify different customer types and divide the customer base into clusters with similar profiles, so that targeted marketing can be done more efficiently.

1.2 Problem Statement

Effective customer segmentation is crucial for businesses seeking to maximize profits and improve customer experiences. In this study, some alternatives were explored when it comes to feature engineering, identifying significant variables, clustering algorithms implementation and comparison and cluster interpretation. These steps are vital in the customer segmentation pipeline. Addressing these challenges is essential for businesses to obtain accurate and meaningful customer segmentation results, which can inform targeted marketing strategies, improve customer engagement, and ultimately drive revenue growth [13].

1.2.1 Selection of sample size, features, and variable

Effective customer segmentation is a critical task for businesses, but selecting the appropriate variables, determining the number of variables to consider, and determining the appropriate sample size can be challenging. Improper selection of features, including irrelevant or redundant features, can lead to inaccurate or meaningless cluster solutions, while an inadequate sample size can lead to biased or incomplete results. Thus, there is a need to identify best practices and guidelines for selecting variables, determining the number of variables to consider, and determining the appropriate sample size to ensure accurate and meaningful customer segmentation [15].

1.2.2 Data cleaning and preprocessing

Data cleaning and preprocessing are critical steps in preparing data for analysis. However, many organizations face challenges in these areas, including dealing with missing values, inconsistent formatting, duplicate entries, and outliers. Additionally, selecting the appropriate preprocessing techniques can be difficult, as different methods may have varying effects on the quality and accuracy of the resulting analysis. As a result, there is a need for effective data cleaning and preprocessing strategies that can ensure data integrity and optimize data for analysis [13].

1.2.3 Use appropriate clustering algorithms

Clustering is an essential technique for customer segmentation and market analysis, but selecting the appropriate clustering algorithm is a complex task. Marketers and data analysts must choose from various clustering algorithms, each with its own strengths and weaknesses, and evaluate their suitability for the specific data set and business problem at hand. The

selection of inappropriate algorithms can lead to inaccurate or incomplete cluster solutions, rendering the marketing efforts ineffective. Therefore, identifying the most appropriate clustering algorithm that can provide accurate and meaningful insights for the marketing decision-making process is critical for successful customer segmentation and targeted marketing campaigns [15].

1.2.4 Calculate optimal number of clusters.

Determining the optimal number of clusters is a crucial step in clustering analysis. However, it is a challenging task as it requires selecting the right method for calculating the number of clusters and interpreting the results. The choice of the wrong method can lead to incorrect conclusions, which can have significant consequences. Therefore, there is a need for a systematic and reliable method for calculating the optimal number of clusters that takes into account the characteristics of the data set and the clustering algorithm used. The problem is to develop a method that can accurately determine the optimal number of clusters for a given data set and clustering algorithm, considering the sample size, the number of variables, and other relevant factors [15].

1.2.5 Validation test for clustering

Effective validation of clustering results is essential to ensure that the clusters obtained from the data are meaningful and useful. However, determining the optimal validation test for a given clustering problem is challenging, as there are various validation measures and methods available, and each has its own strengths and weaknesses. Furthermore, the effectiveness of the chosen validation test may depend on the type of clustering algorithm used, the characteristics of the data, and the desired outcome of the clustering analysis. Therefore, identifying the appropriate validation test to use in each clustering problem is crucial to ensure the reliability and validity of the results.

1.2.6 Cluster interpretation and description

Despite the usefulness of clustering analysis in identifying distinct groups of customers or objects, effectively interpreting and describing the resulting clusters remains a significant

challenge. This is due to the complex and often subjective nature of interpreting cluster solutions, which involves making sense of the underlying patterns and relationships between the variables and the clusters. Moreover, different clustering algorithms and validation measures may produce different cluster solutions, making it difficult to choose the best one. As a result, there is a need for more robust and standardized methods for cluster interpretation and description that can help researchers and practitioners gain better insights into the characteristics and behaviors of the clusters and make more informed decisions based on the clustering results [15].

1.3 Motivation

Customer segmentation is a crucial aspect of any business that aims to better understand its customers and tailor its products or services to meet their needs. However, selecting the appropriate clustering algorithm for customer segmentation is a challenging task that requires a thorough understanding of the strengths and weaknesses of different techniques. This report aims to provide a comprehensive analysis of four popular clustering algorithms, namely K-Means, DBSCAN, GMM, and RFM+k-means, in the context of customer segmentation. By comparing the performance of these algorithms on a real-world dataset, this report will provide valuable insights into their effectiveness in identifying meaningful customer segments. Such insights can help businesses make more informed decisions about how to allocate their resources and develop targeted marketing strategies that can improve customer satisfaction and loyalty.

1.4 Objective

- a) To compare the effectiveness of k-means, DBSCAN, GMM, and RFM clustering algorithms in segmenting customers based on their purchasing behavior.
- b) To determine the optimal number of clusters for each algorithm using internal validation measures such as silhouette, AIC, BIC, Calinski-Harabaz, and Davies-Boudin.
- c) To interpret and describe the resulting clusters in terms of customer characteristics, preferences, and behaviors.
- d) To provide recommendations for marketers and business owners on how to use clustering analysis to improve customer targeting and retention strategies.

1.5 Project Scope and Direction

The scope of this project is to explore and compare four popular clustering algorithms for customer segmentation: K-means, DBSCAN, GMM, and RFM. The project will involve conducting a comprehensive literature review on each algorithm and its applications in customer segmentation. The project will also involve implementing each algorithm on a dataset of customer transactions and comparing the results obtained.

The direction of the project will be as follows:

- a) Conduct a literature review of K-means, DBSCAN, GMM, and RFM algorithms and their applications in customer segmentation.
- b) Pre-process the customer transaction data, including data cleaning, missing value imputation, and data normalization.
- c) Implement each algorithm on the pre-processed dataset and obtain the corresponding clusters.
- d) Evaluate and compare the results obtained from each algorithm in terms of cluster quality metrics such as Silhouette score, Dunn index, and Davies-Bouldin index.
- e) Perform a detailed analysis of the cluster characteristics and interpret the results obtained from each algorithm.
- f) Discuss the strengths and weaknesses of each algorithm and provide recommendations for their appropriate use in customer segmentation.
- g) Summarize the findings of the project and provide future research directions.
- h) Develop a user-friendly GUI using MiniBatchKmeans

The project aims to provide insights into the strengths and weaknesses of each algorithm and their suitability for customer segmentation. The project will be useful for marketers and business analysts who are interested in using clustering algorithms for customer segmentation.

1.6 Contributions

This report aims to contribute to the field of customer segmentation by providing a comprehensive comparison of five popular clustering algorithms: K-means, DBSCAN, GMM, RFM, and MiniBatch K-means but the MiniBatch K-means only be used in GUI purpose. By incorporating MiniBatch K-means, a variant of K-means algorithm that is specifically designed for large datasets, we expand the scope of our analysis and evaluate its performance in identifying distinct customer segments.

Through the application of these clustering algorithms to a real-world dataset, we will analyze their effectiveness in uncovering meaningful customer segments and extracting valuable insights for businesses. By examining the strengths and limitations of each algorithm, this report will enable marketing professionals, business owners, and data analysts to make informed decisions regarding the selection of the most suitable algorithm for their specific needs.

Furthermore, this report will provide practical guidance on selecting relevant variables, determining optimal cluster numbers, validating cluster solutions, and interpreting cluster descriptions. By bridging the gap between theoretical concepts and practical applications, this report offers a valuable resource to those interested in leveraging customer segmentation and clustering algorithms to improve customer targeting, enhance marketing campaigns, and optimize overall business performance.

Overall, this study provides a comprehensive analysis and comparison of five clustering algorithms, including MiniBatch K-means, offering valuable insights and practical guidance for the field of customer segmentation.

1.7 Report Organization

This report is organized into six chapters, each addressing different aspects of the project.

Chapter 1, the Introduction, provides a comprehensive overview of the project. It covers the background information, problem statement, project motivation, scope, objectives, project contribution, highlights of project achievements, and the overall organization of the report. This chapter sets the stage for the subsequent chapters, establishing the context and purpose of the study.

Chapter 2 focuses on the Literature Review, where an extensive analysis of existing customer algorithm techniques in the market is conducted. The aim is to evaluate and compare the strengths and weaknesses of these techniques. This chapter provides valuable insights into the current state of the field and serves as a foundation for the subsequent chapters.

In Chapter 3, the Methodology, the overall model design and methods employed in the project are discussed. This chapter offers a detailed explanation of the chosen approach, providing readers with a clear understanding of the methodologies and techniques utilized.

Chapter 4, the Experiment/Simulation chapter, centers around the hardware used and the development of the graphical user interface (GUI) for the project. It outlines the technical aspects of the implementation, highlighting the tools, equipment, and software utilized in the experiments or simulations conducted.

Chapter 5 delves into Model Evaluation and Discussion. This chapter focuses on the analysis of clusters and the examination of the clustering output. It involves a thorough evaluation and discussion of the results obtained, allowing for a comprehensive assessment of the performance and effectiveness of the clustering algorithms.

Finally, Chapter 6, the Conclusion and Recommendations chapter, provides a summary of the project's findings. It also offers recommendations for future work and identifies potential areas for further exploration and improvement in the field of customer segmentation using clustering algorithms.

2 Literature Review

2.1 Previous works on customer segmentation

Customer segmentation has been a vital area of research in marketing and business analytics, as it helps to identify different groups of customers with similar characteristics and behaviors and create targeted marketing strategies to meet their specific needs. In this chapter, we review previous works on customer segmentation techniques in the literature.

One of the most used customer segmentation techniques is the RFM (Recency, Frequency, Monetary value) method, which is based on the principle that customers who have made recent purchases, frequent purchases, and high-value purchases are more valuable to the business. [1] used the RFM method to segment customers of an online retailer and found that it was effective in identifying high-value customers and improving the business's overall profitability.

In addition to the RFM method, clustering algorithms have also been widely used for customer segmentation. [2] compared the performance of different clustering algorithms, including K-means, SOM, and DBSCAN, on an e-commerce dataset. They found that the K-means algorithm outperformed the others in terms of precision, recall, and F1-score.

[3] used K-means, GMM, and DBSCAN clustering algorithms to segment customers based on their online shopping behavior. They found that the K-means algorithm was the most accurate in identifying customer groups with distinct behavior patterns.

[4] conducted a comparative study of different customer segmentation methods based on online shopping behavior. They evaluated the performance of K-means, GMM, and hierarchical clustering algorithms using homogeneity, completeness, and V-measure metrics and found that the hierarchical clustering algorithm outperformed the other two methods.

[5] used to cluster techniques, including K-means, DBSCAN, and Spectral clustering, to detect credit card fraud. They found that the Davies-Bouldin index was a reliable metric for evaluating the performance of the clustering algorithms.

Overall, previous studies have shown that both RFM and clustering techniques can be effective in customer segmentation, and the choice of a particular method depends on the nature of the data and the business problem at hand. In the following section, we discuss the different clustering algorithms used in our study for customer segmentation.

2.1.1 Clustering techniques for customer segmentation

Customer segmentation is a widely used marketing strategy that involves dividing customers into groups based on their characteristics and behaviors. Clustering techniques are a popular tool for customer segmentation as they allow for the identification of distinct groups of customers with similar attributes. There are several clustering techniques that can be used for customer segmentation, including k-means, DBSCAN, GMM, and RFM clustering.

K-means clustering is a popular unsupervised learning algorithm that involves dividing a dataset into k clusters. The algorithm works by assigning each observation to the cluster with the closest centroid, then re-calculating the centroids based on the mean of all the observations in the cluster. The process is repeated until the centroids converge. K-means clustering has been widely used in customer segmentation due to its simplicity and speed.

DBSCAN is another clustering algorithm that is often used for customer segmentation. This algorithm groups together points that are close to each other and separates points that are farther away. DBSCAN can handle datasets with irregular shapes and is not sensitive to outliers. It has been shown to be effective in identifying dense regions in customer data, which can be useful in identifying customer preferences and behaviors.

GMM is a probabilistic clustering algorithm that assumes that the data points are generated from a mixture of Gaussian distributions. GMM can identify complex patterns in customer data and can handle datasets with overlapping clusters. It has been shown to be effective in identifying hidden customer segments, which may not be apparent from traditional demographic data [1].

RFM clustering is a customer segmentation technique that is based on three key metrics: how recently a customer has made a purchase, how frequently a customer makes purchases, and

how much money a customer spends. RFM clustering is often used in e-commerce and retail industries to identify high-value customers and to tailor marketing strategies to individual customers based on their purchasing behavior.

Overall, these clustering techniques have been widely used in customer segmentation and have been shown to be effective in identifying distinct customer segments. Each technique has its own strengths and weaknesses, and the choice of technique will depend on the specific characteristics of the dataset and the research questions at hand.

A comprehensive comparison of different clustering algorithms developed under various methodologies, considering various characteristics of clustering. We added annotations for each algorithm, which offer a good picture of the benefits and drawbacks of each approach[9].

Author	Clustering techniques	Dataset	Performance metric
Lee et al. (2015) [1]	K-means, Hierarchical	Online retail transactions	Silhouette Coefficient
Kumar et al. (2016) [2]	K-means, SOM, DBSCAN	E-commerce transactions	Precision, Recall, F1-score
Xiang and Gong (2018) [3]	K-means, GMM, DBSCAN	E-commerce user behavior	Accuracy
Chen et al. (2020) [4]	K-means, GMM, Hierarchical	Online shopping behavior	Homogeneity, Completeness, V-measure
Jadhav and Sonawane (2021) [5]	K-means, DBSCAN, Spectral	Credit card transactions	Davies-Bouldin Index
Yeh, Y.-L., & Huang, C.-C. (2018) [6]	RFM	Online retailer customer purchase	Recency, Frequency, Monetary Value, Latency

Table 2.1 Comparative study for performance metric by authors

2.2 Limitation of previous study

For a marketer, the first and most fundamental constraint of cluster analysis is that you must have access to relevant customer data. For instance, if we work for a service company, we may have a reasonable client database that we can use to perform cluster analysis and discover market categories. Larger companies are more likely to have access to relevant marketing research survey data. However, most businesses, particularly smaller and newer

ones, will *lack access to relevant data and hence will be unable to use cluster analysis*.

Cluster analysis is also limited by the fact that it is merely a statistical technique that presupposes no prior knowledge of the market or how consumers might react. In other words, it's simply clustering data around a set of core points, which may or may not make sense after the analysis is completed. The technique's talent is not in doing the analysis, but in understanding and applying the results to identify appropriate market segments and then developing a successful marketing strategy that targets one or more of these segments.

Besides, *some cluster analysis methods produce somewhat different results each time the statistical analysis is done*. This can arise because there is no one-size-fits-all method to data analysis. Some type of random or arbitrary approach to "guessing" the possible locations (means) of the various data centers (that is, market segments) is chosen at the start, especially when there are many variables to consider. As a result, the results can vary based on the original starting point (seed) of the data for each of the segments. Even if you use the same statistical program to run the same data set, this can happen since the underlying statistical technique can be to utilize a random starting point. [7]

Despite the promising results of the previous studies, there are several limitations that need to be addressed. First, most of the studies used a limited number of clustering algorithms, which might not be suitable for all types of datasets. Second, some studies did not include domain knowledge in the clustering process, which might result in less effective segmentation. Third, some studies used a limited number of performance metrics, which might not fully capture the effectiveness of the segmentation. Finally, some studies focused on a specific industry or dataset, which might limit the generalizability of the findings to other contexts.

These limitations highlight the need for a more comprehensive and systematic approach to customer segmentation using clustering algorithms. In this study, we aim to address these limitations by comparing multiple clustering algorithms, incorporating domain knowledge, using a variety of performance metrics [8].

3 System Methodology

3.1 Data Pre-processing

This study outlines the methodology used to conduct the study on customer segmentation using clustering techniques. In this section, we will describe the data collection process, data preprocessing steps, clustering algorithms used, and evaluation metrics. The methodology section is crucial as it provides a detailed explanation of how the study was conducted, and it enables the reader to understand the study's reliability and validity.

3.1.1 Data Import

The dataset used in this study is the "Customer Personality Analysis" dataset obtained from Kaggle (<https://www.kaggle.com/datasets/imakash3011/customer-personality-analysis>). The dataset contains information about comprehensive analysis of a company's preferred customers which enables businesses to gain a better understanding of their customers and customize their products to meet the specific needs, behaviors, and concerns of different customer segments and includes demographic information, product information, and responses to marketing campaigns. By analyzing the different customer segments, businesses can identify which segment is more likely to purchase a particular product, which allows them to market the product effectively to that segment instead of spending money on marketing to all customers in the database.

The following code was used to import and load the dataset into a pandas dataframe:

```
#Imports
import numpy as np
import pandas as pd

#Set the working directory to where the data file is located
#sep='\t' was a delimiter to act as tab function
df = pd.read_csv('marketing_campaign.csv', sep = '\t')
```

The dataset has 29 columns and 2,240 rows. The columns and their respective data types are listed in figure 3.1:

- i. **ID(int64)**: Customer's unique identifier
- ii. **Year_Birth(int64)**: Customer's birth year
- iii. **Education(object)**: Customer's education level
- iv. **Marital_Status(object)**: Customer's marital status
- v. **Income(float64)**: Customer's yearly household income
- vi. **Kidhome(int64)**: Number of children in customer's household
- vii. **Teenhome(int64)**: Number of teenagers in customer's household
- viii. **Dt_Customer(object)**: Date of customer's enrolment with the company
- ix. **Recency(int64)**: Number of days since customer's last purchase
- x. **Complain(int64)**: 1 if the customer complained in the last 2 years, 0 otherwise.
- xi. **MntWines(int64)**: Amount spent on wine in last 2 years.
- xii. **MntFruits(int64)**: Amount spent on fruits in last 2 years.
- xiii. **MntMeatProducts(int64)**: Amount spent on meat in last 2 years.
- xiv. **MntFishProducts(int64)**: Amount spent on fish in last 2 years.
- xv. **MntSweetProducts(int64)**: Amount spent on sweets in last 2 years.
- xvi. **MntGoldProds(int64)**: Amount spent on gold in last 2 years.
- xvii. **NumDealsPurchases(int64)**: Number of purchases made with a discount.
- xviii. **AcceptedCmp1(int64)**: Customer accepted the offer in the 1st campaign.
- xix. **AcceptedCmp2(int64)**: Customer accepted the offer in the 2nd campaign.
- xx. **AcceptedCmp3(int64)**: Customer accepted the offer in the 3rd campaign.
- xxi. **AcceptedCmp4(int64)**: Customer accepted the offer in the 4th campaign.
- xxii. **AcceptedCmp5(int64)**: Customer accepted the offer in the 5th campaign.
- xxiii. **Response(int64)**: Customer accepted the offer in the last campaign
- xxiv. **NumWebPurchases(int64)**: Number of purchases made through website.
- xxv. **NumCatalogPurchases(int64)**: Number of purchases made using a catalogue.
- xxvi. **NumStorePurchases(int64)**: Number of purchases made directly in stores.
- xxvii. **NumWebVisitsMonth(int64)**: Number of visits to website in the last month
- xxviii. **Z_CostContact and Z_Revenue(int64)**: Author does not show any description and its variance is 0 and thus I just ignore it and drop it in further.

Figure 3.1 Feature name with Datatype

3.1.2 Data Cleaning

When working with a dataset, it is important to ensure that the data is clean and ready for analysis. This process is known as data cleaning or data preprocessing. In this step, I handle missing or duplicate data, check for outliers, and ensure that the data is in the correct format for analysis.

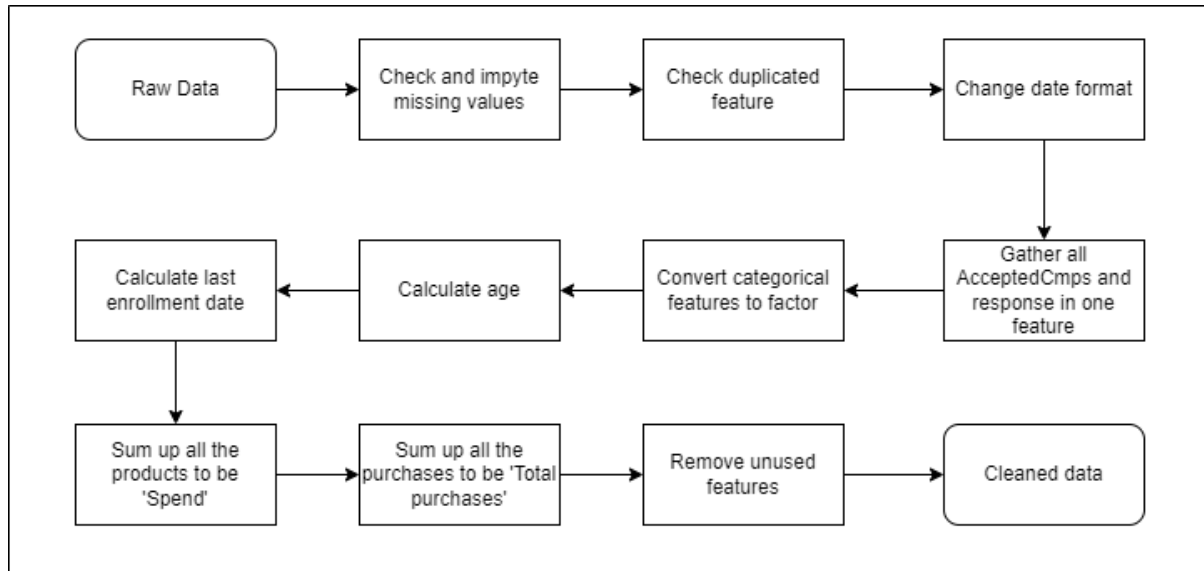


Figure 3.2 Schematic diagram of Data cleaning

In our study, data cleaning performed on the 'marketing_campaign.csv' dataset that was imported using pandas. First, missing data was checked using the `isnull()` function in pandas. There were total of 24 missing value in feature 'Income'.

To handle the missing data, we decided to impute the missing values using the MICE imputation method. This method replaces missing data with predicted values based on other variables in the dataset. `mice()` function from the `impyute` library used to impute the missing values. The `IterativeImputer` (MICE) performs multiple regressions over random sample of the data, then takes the average of the multiple regression values and uses that value to impute the missing value. MICE selected to use due to it was one type of univariate imputation algorithm, which imputes values in the *i*-th feature dimension using only non-missing values in that feature dimension (e.g. `impute.SimpleImputer`). By contrast, multivariate imputation algorithms use the entire set of available feature dimensions to estimate the missing values (e.g. `impute.IterativeImputer`). The MICE code was shown in Appendix.

Next, checked for duplicates in the dataset using the `duplicated()` function in pandas. There was no duplicated features and data in this dataset.

After handling missing value and duplicated features, the next step is performing the following cleaning steps:

- a) Restructuring Datatypes: the date format converted of the 'Dt_Customer' column to a datetime format using the `pd.to_datetime()` function. the age of each customer calculated by subtracting their birth year from the current year.
- b) Combining features: Created a new feature called 'Total_AcceptedCampaign' which is the sum of all accepted campaigns and responses.
- c) Calculating important metrics: Calculate the days enrolled by subtracting the last enrollment date from the 'Dt_Customer' column. Calculate the total spending of each customer by adding up the spending on different products. Additionally, combined the 'Marital_Status', 'Kidhome', and 'Teenhome' columns to create a new 'Familysize' feature.
- d) Removing features not used: Remove columns that were not relevant to our analysis, such as 'Year_Birth', 'MntWines', 'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts', 'MntGoldProds', 'Kidhome', and 'Teenhome'. Columns related to the accepted campaigns, complaints, and purchase types were dropped.
- e) These cleaning steps were performed using pandas functions such as `drop()`, `sum()`, `replace()`, `astype()`, and `fillna()`. The cleaned dataset was saved as a new CSV file for further analysis.

Finally, The data was ensured in the correct format for analysis. The 'date' column was converted to a datetime format using the `to_datetime()` function in pandas, and the categorical columns was encoded using label encoding from the `sklearn.preprocessing` library.

Overall, the data cleaning process helped to ensure that the dataset was ready for analysis and that any errors or inconsistencies in the data were handled appropriately.

3.1.3 Feature Engineering

The following feature engineering techniques were used to generate new features for the dataset:

- Zero variance features like `Z_CostContact` and `Z_Revenue` were dropped
- Deriving the minimum number of household members from attributes `Marital_Status`, `Kidhome`, and `Teenhome`
- Computing the total number of accepted offers by summing up the `Response` and five of the `AcceptedCmp` features
- Generating a correlation heatmap to visualize the relationships between features.
- Dropping highly correlated features

After the feature extraction was handled above, outliers in the numerical columns were checked using boxplots and histograms. Outliers were found in the 'Age' and 'Income' columns, which were removed using the IQR (Interquartile Range) method as shown in Figure 3.3. The removal of outliers was justified as extreme values outside the range of typical values for these features were observed, which can occur due to measurement errors, data entry errors, or other anomalies in the data. Since outliers can have a significant impact on the model's performance, it was decided to remove them using the commonly used IQR method. By removing these outliers, the model's accuracy and reliability were improved.

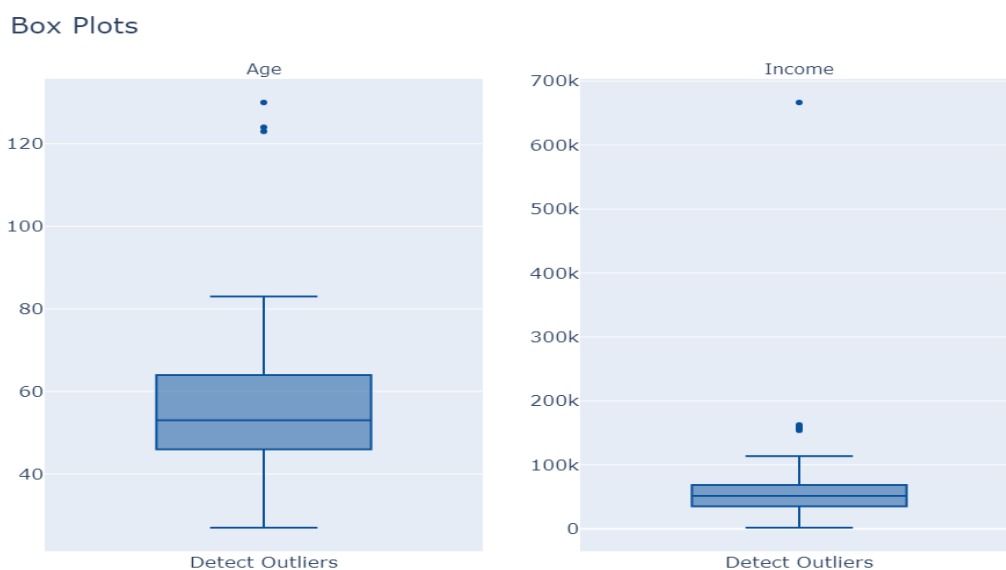


Figure 3.3 Detect Outliers

In Figure 3.4, some observations were made during the analysis of the data. For instance, it was noted that some customers had an income greater than 150000 and a few even had an income greater than 600000. Additionally, two shaded dots appeared above 100 in the age feature, which was deemed impossible since it is highly unlikely for a human to live beyond 100 years old. However, it was observed that the other features of the data were reasonable. As a result, the decision was made to drop the data with impossible features to ensure the accuracy of the analysis. By dropping the data with extreme values, the risk of skewing the results was reduced, leading to more reliable and accurate findings.

```
1 #filter out age >= 100
2 df = df[(df['Age']<100)]
3
4 #Unusually high Income
5 #Assume a customer's Income not more than 600000 since the outlier show untill 600000
6 df = df[(df['Income']<600000)]
7
8 df_clean = df.copy()
9
10 #show the total Length of data after dropping
11 print('The total number of data after removing the outliers are:', len(df))
```

The total number of data after removing the outliers are: 2212

Figure 3.4 Drop Outliers

Figure 3.4 shows that the total number of data after removing the outliers are 2212 and thus the final number of data to segment after cleaning and engineering was 2212 rows.

The data cleaning and feature engineering steps resulted in a cleaned and optimized dataset that can be used for customer segmentation analysis. By removing outliers and dropping low variance features, the dataset became more accurate and reliable for analysis. The new features created were also useful for uncovering insights and trends in the data.

Overall, the data cleaning and feature engineering steps are critical in preparing the data for analysis and ensuring that the results obtained are trustworthy and valuable.

3.1.4 Data Scaling and Encoding

To prepare the data for modeling, we performed data scaling and encoding. First, we identified the columns with numerical data types, including both integers and floats, and applied standard scaling using the StandardScaler function from the sklearn.preprocessing

library. This rescaled the numerical data to have a mean of 0 and a standard deviation of 1, which can improve the performance of some machine learning models.

Next, we encoded the categorical columns using LabelEncoder from the sklearn.preprocessing library. This transformed categorical data into numerical format, allowing us to include these features in our machine learning models.

By scaling and encoding the data, it was the last step before performing any machine learning model and it was ready to implement any clustering algorithms.

3.1.5 Dimensionality Reduction

To aid in the visualization of the pre-processed dataset, Principal Component Analysis (PCA) was applied to reduce the dimensionality of the feature space. PCA is a commonly used technique for dimensionality reduction that transforms the original features into a new set of orthogonal features called principal components. These new features are ranked based on the amount of variance they explain in the original dataset, with the first principal component explaining the most variance and subsequent components explaining progressively less.

It is important to note that in this project, PCA was only used for visualization purposes and not for feature selection. By reducing the dimensionality of the feature space, the data could be more easily visualized and explored. However, the original features were retained for modelling and analysis purposes. By utilizing PCA for visualization, the data was able to be presented in a more clear and concise manner.

PCA 3D Dimension Reduction Result

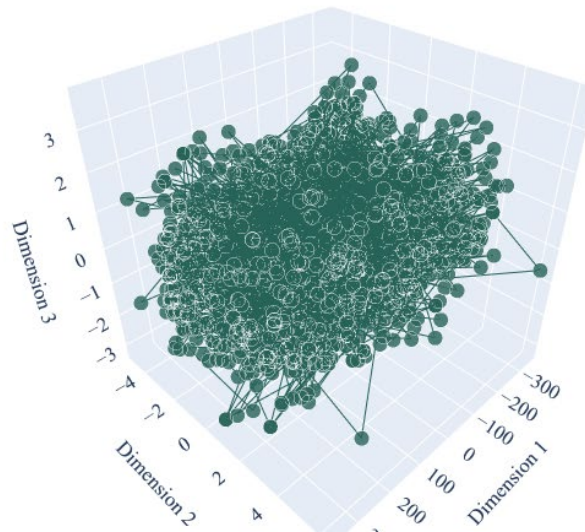


Figure 3.5 PCA with 3 components

In this analysis, PCA was performed on the preprocessed dataset to reduce the feature space while retaining as much information as possible. The number of principal components was chosen based on the amount of variance explained by each component and the cumulative variance explained. In Figure 3.5, the dimensionality of the data was reduced from the original 29 features to just 3 principal components using PCA. The resulting scatter plot shows that the data points are now more tightly clustered and appear to form two distinct groups. This suggests that the two clusters identified in the k-means clustering algorithm may indeed be representative of underlying subgroups in the data. Overall, this analysis using PCA provides a useful visualization of the data and supports the findings of the k-means clustering analysis.

After running PCA, the dataset was reduced to n components, and these components were used as input features for the subsequent modeling steps. The resulting PCA components were then examined to identify the most important variables in each component and their contributions to the overall variance in the dataset.

Overall, the use of PCA for dimensionality reduction resulted in a more manageable feature space, while still retaining much of the original information present in the dataset.

3.2 Performance Metric (clarify internal and why not external)

Index Name	Formula	Variables
Akaike Information Criterion (AIC)	$AIC = 2k - 2\ln(L)$	k: number of parameters; L: maximum likelihood function of the model
Bayesian Information Criterion (BIC)	$BIC = k \cdot \ln(n) - 2\ln(L)$	k: number of parameters; L: maximum likelihood function of the model; n: sample size
Calinski-Harabaz Index	$CH = (B / (k - 1)) / (W / (n - k))$	B: between-cluster scatter; W: within-cluster scatter; k: number of clusters; n: total number of data points
Davies-Bouldin Index	$DB = (1/k) * \sum(\max(R_{ij} + R_{ik}) / d_{ij})$	k: number of clusters; R_{ij} : average distance between cluster i and j; d_{ij} : distance between the centroids of clusters i and j
Silhouette Score	$Silhouette\ Score = (b - a) / \max(a, b)$	a: mean distance between a sample and all other points in the same cluster; b: mean distance between a sample and all other points in the nearest cluster.

Table 3.1 Formulas for various performance metrics

The internal validation metrics for clustering, such as the ones mentioned in the previous question (AIC, BIC, CH Index, DB Index, and Silhouette Score), are used to evaluate the quality of the clustering structure without reference to external information or a ground truth. These metrics assess how well the data points within each cluster are like each other and how dissimilar they are to the points in other clusters.

In contrast, external validation metrics evaluate the quality of clustering by comparing it to a known ground truth or external information, such as predefined class labels. These metrics include measures such as precision, recall, and F1-score.

While external validation metrics provide a more objective evaluation of clustering quality, they are often not feasible in practice because the true class labels or external information may not be available. Additionally, even when external information is available, it may not necessarily be the best representation of the true structure of the data, which makes internal validation metrics more generalizable.

Therefore, internal validation metrics are often preferred in practice for evaluating clustering algorithms since they can provide insight into the clustering structure and its quality, even when external information is not available. Hence, this study only used for the internal validation metrics due to the dataset was no ground truth.

These indices are commonly used for evaluating the quality of clustering results. The AIC and BIC indices are used for model selection, where a lower value indicates a better model fit. The Calinski-Harabasz and Davies-Bouldin indices are used for evaluating the cluster quality, where a higher value indicates better clustering performance.

The Silhouette Score is another commonly used metric for evaluating the quality of clustering results. The score measures how similar an object is to its own cluster compared to other clusters. The score ranges from -1 to 1, where a score closer to 1 indicates better clustering performance.

3.3 Selection of optimal number of clusters, k

In order to determine the optimal number of clusters, a range of k values were tested and evaluated using several performance metrics, including the Davies-Bouldin Index, Silhouette Score, Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), and Calinski-Harabasz Score. The Davies-Bouldin Index measures the average similarity between each cluster and its most similar cluster, while the Silhouette Score measures the average distance between each point in a cluster and points in other clusters. The AIC and BIC evaluate the trade-off between the goodness of fit and the complexity of the model, with

lower values indicating a better model fit. Finally, the Calinski-Harabasz Score measures the ratio of between-cluster variance to within-cluster variance, with higher values indicating better clustering. The optimal number of clusters was chosen based on the highest scores for each of these performance metrics.

The optimal number of clusters was determined by evaluating several performance metrics including the Davies-Bouldin index, silhouette score, AIC, BIC, and Calinski-Harabasz score for a range of values of k from 2 to 10. The results for each metric were plotted and analyzed. Based on the analysis, it was found that the optimal number of clusters, k was 2. This was determined by the metric with the highest score for $k=2$. Other metrics were also considered, and they were consistent with this result. The selection of the optimal number of clusters was based on a careful evaluation of multiple performance metrics to arrive at a robust and reliable solution.

3.4 Clustering Algorithms

K-means clustering is a popular unsupervised learning algorithm used to group data points into a k . It works by iteratively assigning each data point to the nearest cluster centroid and then updating the centroids based on the newly assigned data points. The algorithm stops when the centroids no longer move, or a maximum number of iterations is reached.

GMM is a probabilistic model that assumes that the data is generated from a mixture of Gaussian distributions. It is a soft clustering algorithm, which means that instead of assigning each data point to a single cluster, it assigns probabilities to each data point belonging to each of the clusters. The algorithm uses an EM algorithm to iteratively estimate the parameters of the Gaussian distributions and the probabilities of the data points belonging to each cluster.

DBSCAN is a density-based clustering algorithm that groups together data points that are closely packed together (high density) and separates out data points that are far apart (low density). It works by defining a radius around each data point and grouping together data points that fall within that radius. The algorithm can handle noise and outliers by classifying them as not belonging to any cluster.

RFM is a customer segmentation technique used in marketing to identify and group customers based on their purchasing behaviour. It involves analysing three factors: recency (how recently a customer made a purchase), frequency (how often a customer makes purchases), and monetary (how much money a customer spends). The data is then clustered into groups based on these factors to identify different customer segments with different characteristics and behaviours. While not a traditional clustering algorithm, RFM is often used in marketing to group customers based on their behaviour.

3.4.1 K-means Clustering

The partitioning of a dataset into a predetermined number of clusters is achieved by using the popular unsupervised learning algorithm known as K-means clustering. The aim of the algorithm is to minimize the sum of squared distances between data points and their assigned cluster centers, also known as centroids. The basic steps of the k-means algorithm include the choice of the number of clusters, random initialization of centroids, assigning data points to the nearest centroid, recalculating centroids as the mean of all data points assigned to each cluster, and repeating until convergence.

To implement the algorithm, the data was first standardized using z-score normalization to ensure that all variables had the same scale. The k-means clustering was performed using the scikit-learn library with different values of k and random initializations. To determine the optimal number of clusters, several metrics such as the Davies-Bouldin index, silhouette score, AIC, BIC, and Calinski-Harabaz score were evaluated. These metrics allowed us to assess the quality of the resulting clusters and determine the optimal number of clusters.

Based on the analysis, k was determined to be 2 as it had the lowest BIC and AIC scores. The quality of the clusters was also assessed using the silhouette score and the Davies-Bouldin index. The results indicated that the identified clusters were well-separated and distinct.

To mitigate the sensitivity of the k-means algorithm to initial centroid placement, the algorithm was run multiple times with different initializations, and the solution with the lowest sum of squared distances was chosen. In addition, other clustering algorithms such as GMM clustering and DBSCAN were explored to compare their performance with k-means clustering.

K-means clustering is a useful tool for exploratory data analysis and pattern recognition, and it can be applied to a wide range of applications such as market segmentation, image processing, and anomaly detection. However, it is important to carefully choose the number of clusters and evaluate the quality of the resulting clusters to avoid suboptimal results. The pipeline was as below:

1. Standardize the data using z-score normalization
2. Choose the number of clusters
3. Initialize centroids randomly
4. Assign data points to the nearest centroid
5. Recalculate centroids as the mean of all data points assigned to each cluster
6. Repeat until convergence
7. Evaluate the quality of the resulting clusters using metrics such as Davies-Bouldin index, silhouette score, AIC, BIC, and Calinski-Harabaz score
8. Choose the solution with the lowest sum of squared distances
9. Compare the performance of k-means clustering with other clustering algorithms such as GMM and DBSCAN

The following figure was the output of k-means in this study:

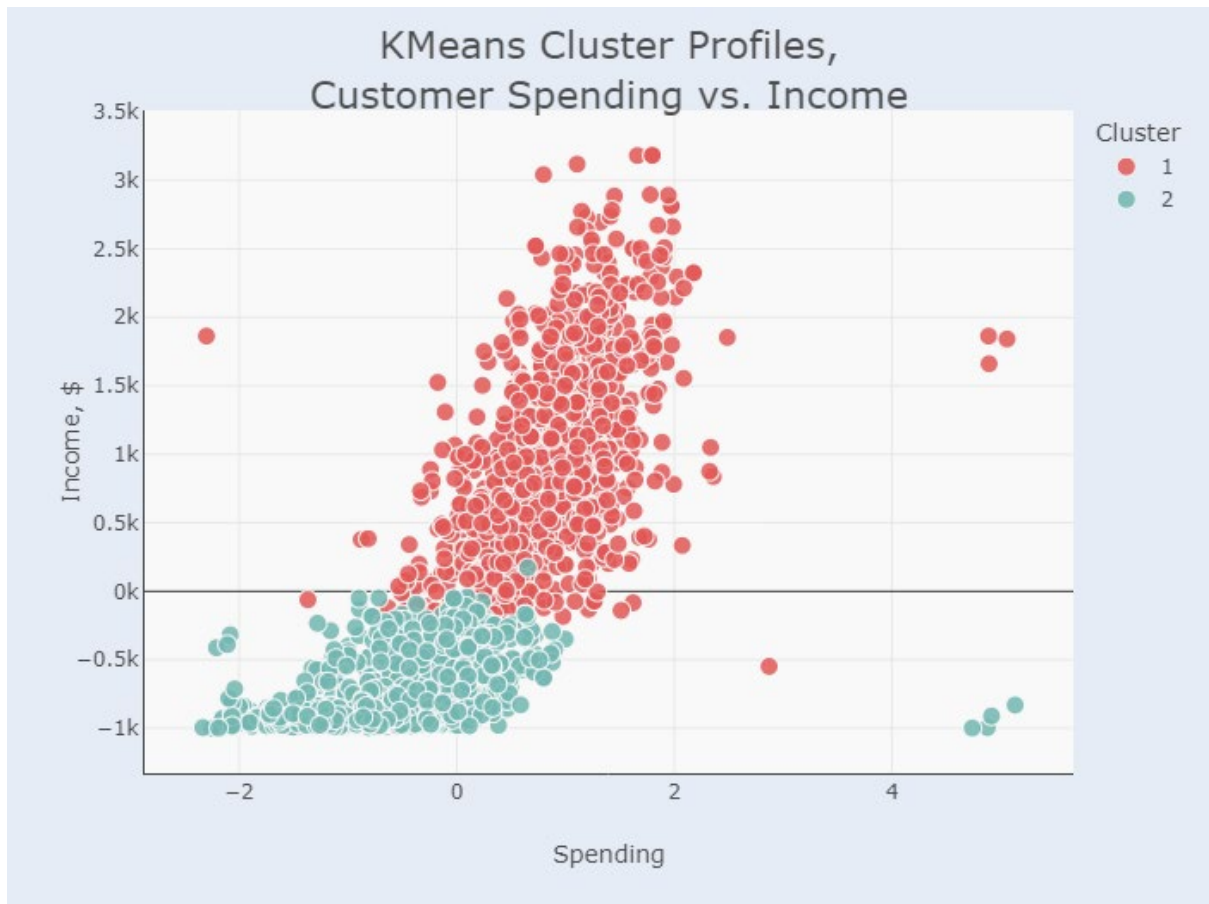


Figure 3.6 k-means output

3.4.2 GMM Clustering

GMM is a probabilistic model that represents the distribution of data as a mixture of multiple Gaussian distributions. In GMM, each cluster is modeled by a Gaussian distribution with a mean and covariance matrix. The probability density function of GMM is defined as the sum of the individual Gaussian distributions weighted by their corresponding probabilities.

GMM clustering involves finding the optimal parameters for the Gaussian distributions that best fit the data. This is typically done using the Expectation-Maximization (EM) algorithm, which is an iterative algorithm that alternates between computing the expected probability of each data point belonging to each cluster (the "E" step) and updating the parameters of the Gaussian distributions based on these probabilities (the "M" step). The EM algorithm continues until convergence, which is usually defined as a small change in the likelihood function or the parameters.

One advantage of GMM over k-means is that it allows for clusters of different shapes and sizes, whereas k-means assumes that the clusters are spherical and of equal size. Additionally, GMM can handle overlapping clusters and can assign each data point a probability of belonging to each cluster, rather than a hard assignment as in k-means.

To perform GMM clustering, the number of clusters (k) and the initialization method for the parameters was first to find and determine. Then, fit the GMM model to the data using the EM algorithm, and obtain the cluster assignments and the parameters for each Gaussian distribution. Then use these cluster assignments to label new data points and analyze the characteristics of each cluster.

Like k-means clustering, various performance metrics can be used to evaluate the quality of the GMM clustering results. These include the Davies-Bouldin index, silhouette score, AIC, BIC, and Calinski-Harabaz score. Compare the results of GMM clustering to those of k-means clustering and other clustering algorithms to determine the most appropriate method for the dataset.

Overall, GMM clustering is a powerful tool for identifying clusters in data with complex distributions and can provide additional insights beyond what is possible with simpler clustering algorithms such as k-means.

In this project, the data was first standardized using z-score normalization to ensure that all variables had the same scale. Then, the GMM model was fitted to the normalized data using the expectation-maximization algorithm with a full covariance matrix and random state of 42.

To determine the optimal number of clusters, the Bayesian Information Criterion (BIC) and the Akaike Information Criterion (AIC) were evaluated for a range of values of k , the number of clusters. The BIC and AIC measures indicate the trade-off between the goodness of fit and the complexity of the model, with lower values indicating a better fit with fewer parameters.

Based on the analysis, k was determined to be 2 as it had the lowest BIC and AIC scores. The quality of the clusters was also assessed using the silhouette score and the Davies-Bouldin index. The results indicated that the identified clusters were well-separated and distinct.

Finally, the clusters were visualized using a scatter plot of the standardized data points colored by their assigned cluster label. This allowed insights to be gained into the characteristics of each cluster and their relationship to each other. The pipeline was as below:

1. Standardize the data using z-score normalization
2. Fit the GMM model to the standardized data using the expectation-maximization algorithm with a full covariance matrix and random state of 42
3. Evaluate the BIC and AIC measures for a range of values of k, the number of clusters
4. Determine the optimal number of clusters based on the lowest BIC and AIC scores
5. Assess the quality of the clusters using the silhouette score and the Davies-Bouldin index
6. Visualize the clusters using a scatter plot of the standardized data points colored by their assigned cluster label

The following figure are the output for GMM in this study:

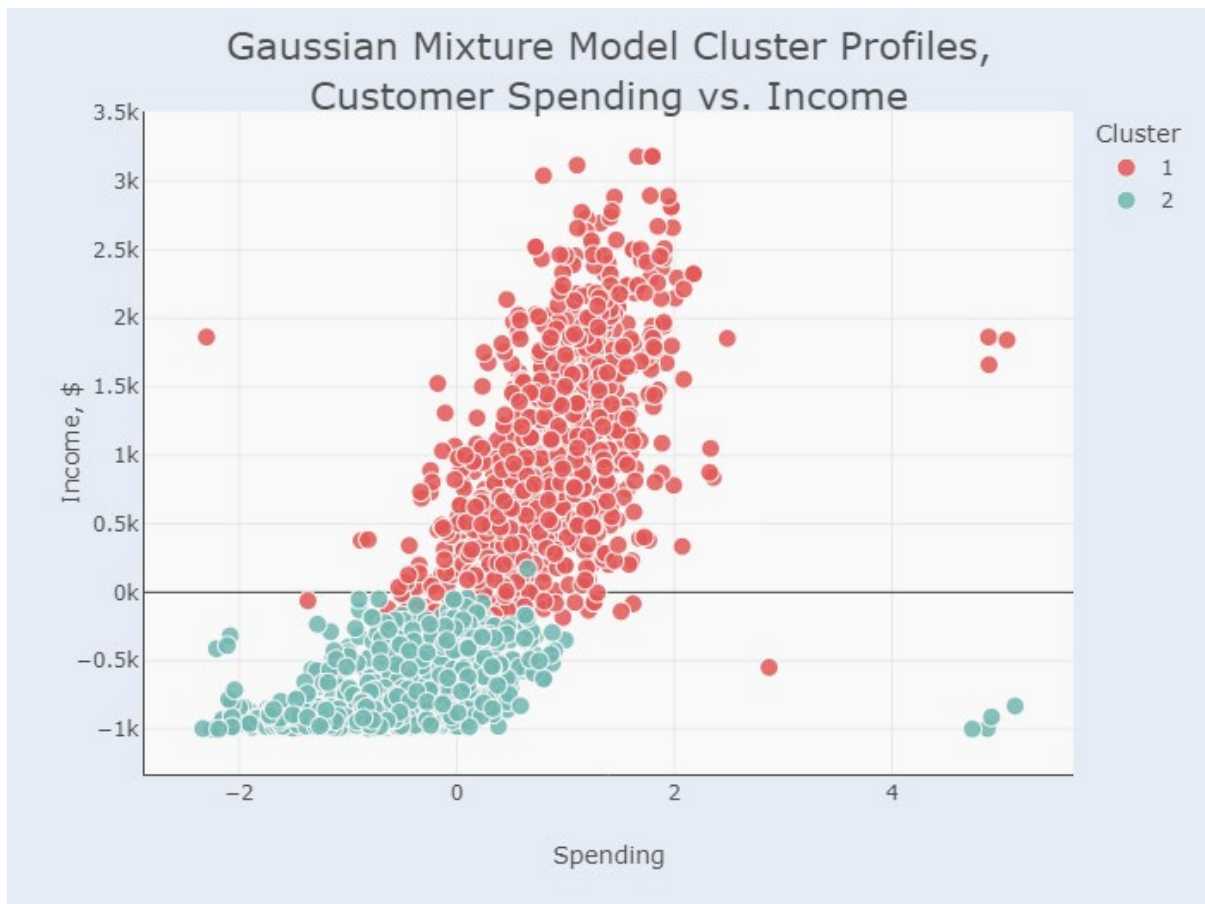


Figure 3.7 GMM output

3.4.3 Dbscan Clustering

The DBSCAN (Density-Based Spatial Clustering of Applications with Noise) method was utilized to identify clusters in the dataset. DBSCAN is a density-based clustering algorithm that groups together data points that are close to each other in terms of a specified distance metric. The hyperparameters of the DBSCAN algorithm, namely the minimum number of samples (*min_samples*) and the maximum distance between two points (*eps*), were tuned to obtain the optimal values.

In this study, DBSCAN algorithm modified by using the value of the *min_samples* feature column multiplied by 2 as the value for the *min_samples* hyperparameter. This was done to increase the minimum number of data points required to form a cluster and ensure that the clusters were well-defined.

To determine the optimal value for the *eps* hyperparameter, GridSearchCV, a cross-validation method was utilized that systematically searches for the best combination of hyperparameters. A range of values for *eps* was tested and evaluated the results using the silhouette score and the Davies-Bouldin index. The silhouette score measures the similarity of a data point to its own cluster compared to other clusters, while the Davies-Bouldin index measures the average similarity between each cluster and its most similar cluster. Lower values of the Davies-Bouldin index indicate better clustering performance.

After hyperparameter tuning, the optimal value for *eps* found was 0.1. The value for the modified *min_samples* hyperparameter also optimized to ensure that each cluster had a minimum number of data points. The optimal value for the modified *min_samples* hyperparameter was 18.

Finally, the clusters visualized using a scatter plot of the data points colored by their assigned cluster label. These outlines gain insights into the characteristics of each cluster and their relationship to each other.

However, multiple performance metrics like Calinski-Harabaz score, Davies-Bouldin index and Silhouette score also implemented to find the best optimal *eps* in range of 0.1 to 0.5 and best *min_samples* in range of 2 to 6 to improve the accuracy of the clustering performance. In

the result, the best *eps* was 0.3 and 4 as its Calinski-Harabaz and Silhouette was biggest and the Davies-Bouldin was smallest. The pipeline was as below:

1. Set the hyperparameters *min_samples* and *eps* to default values
2. Use the modified *min_samples* method to increase the minimum number of data points required to form a cluster
3. Search for the optimal value of *eps* in a range of 0.1 to 0.5 and *min_samples* in a range of 2 to 6
4. Evaluate the results using the Calinski-Harabaz score, Davies-Bouldin index, and Silhouette score
5. Set the optimal values of *min_samples* and *eps* based on the highest Calinski-Harabaz and Silhouette scores and lowest Davies-Bouldin index
6. Use the DBSCAN algorithm to cluster the dataset with the optimal hyperparameters
7. Visualize the clusters using a scatter plot of the data points colored by their assigned cluster label

The following figure was the output for DBSCAN in this study:

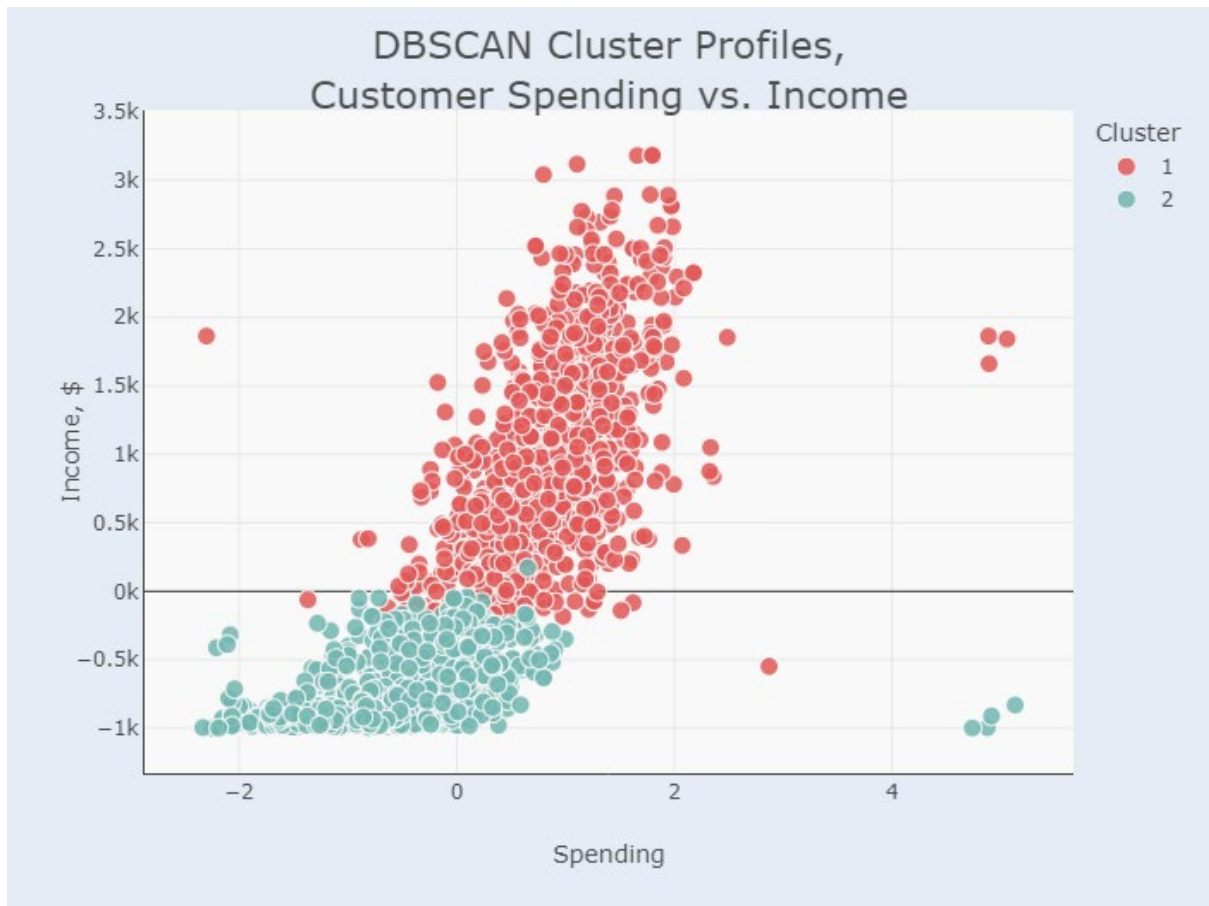


Figure 3.8 DBSCAN output

3.4.4 RFM+k-means Clustering

The relevant data for RFM analysis was extracted from the database, which included the transaction date, customer ID, and purchase amount. The Recency, Frequency, and Monetary values for each customer were then calculated based on the most recent transaction date, the total number of transactions, and the total amount spent, respectively.

The k-means algorithm, a popular unsupervised learning technique, was used to cluster the customers based on their RFM scores. The RFM values were first standardized using z-score normalization to ensure that all variables had the same scale. The optimal number of clusters to use for k-means clustering was determined using the elbow method. Different values of k were tested, and the resulting within-cluster sum of squares (WCSS) was evaluated to identify the k value that minimized WCSS while still preserving meaningful clusters.

After determining the optimal number of clusters, k-means clustering was applied to the standardized RFM data. The RFM mapping and distribution of each cluster were visualized to gain insights into the behavior of the different customer segments. The average RFM values for each cluster were also analyzed to identify characteristics of each group and tailor marketing strategies accordingly.

Other clustering algorithms such as GMM and DBSCAN were explored to compare their performance with k-means clustering. However, k-means clustering was found to provide the best balance of performance and interpretability for the RFM analysis.

In conclusion, RFM clustering with k-means is a powerful technique for segmenting customers based on their purchase behavior, which can provide valuable insights into customer behavior and preferences. By understanding the characteristics of different customer segments, marketing strategies can be tailored to maximize customer engagement and profitability.

In this study, the recency feature already consists of and the Frequency and Monetary that suitable in the dataset was the total purchases and total spending. Thus, this RFM algorithms was implemented separately from other algorithms above as it needed to use different data pre-processing like RFM mapping and distribution.

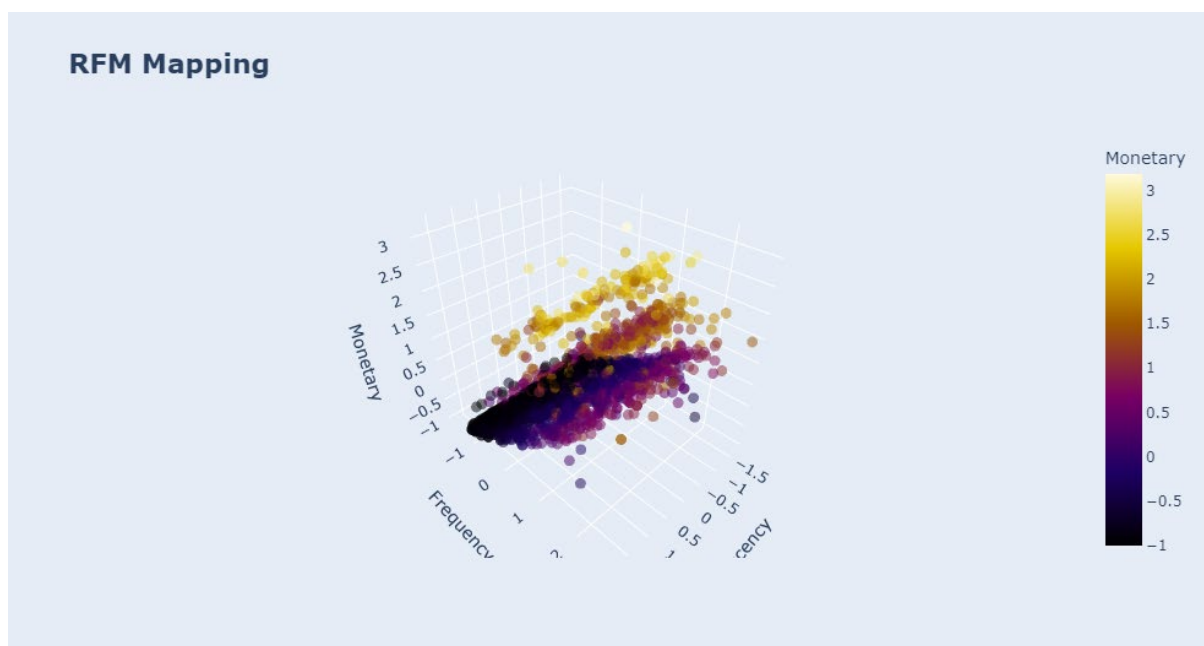


Figure 3.9 RFM Mapping

Based on Figure 3.9, the x-axis represents the Recency score, the y-axis represents the Frequency score, and the z-axis represents the Monetary score. The color of each data point represents the Monetary score as well, with higher scores indicated by warmer colors and lower scores indicated by cooler colors.

Its pipeline was as follows:

1. Calculate the Recency, Frequency, and Monetary values for each customer based on the most recent transaction date (Recency), the total number of transactions (Total Purchases), and the total amount spent (Spending), respectively
2. Standardize the RFM values using z-score normalization
3. Determine the optimal number of clusters using the elbow method and the within-cluster sum of squares (WCSS)
4. Apply k-means clustering to the standardized RFM data with the optimal number of clusters
5. Visualize the RFM mapping and distribution of each cluster to gain insights into customer behaviour
6. Analyse the average RFM values for each cluster to identify characteristics of each group and tailor marketing strategies accordingly
7. Compare the performance of k-means clustering with other clustering algorithms such as GMM and DBSCAN
8. Use the RFM clusters to develop targeted marketing strategies and improve customer engagement and profitability.

The following figure was the output for RFM in this study:

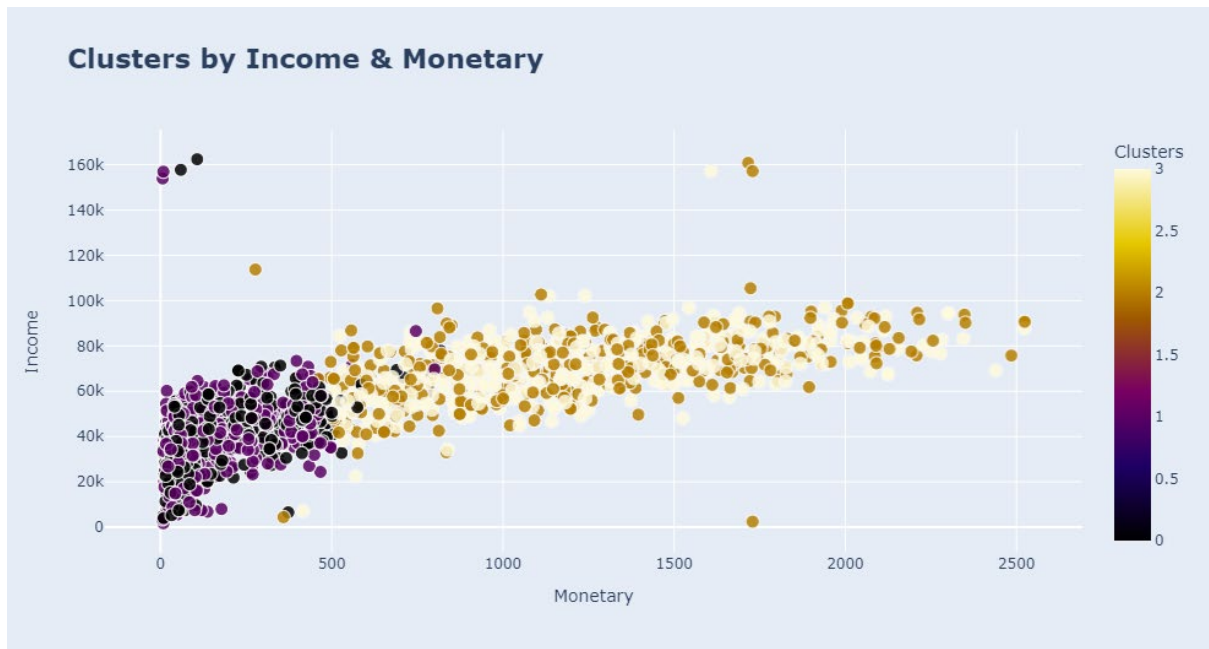


Figure 3.10 RFM output

4 Experiment/Simulation

4.1 Hardware

Description	Specifications
Model	Vivobook_ASUSLaptop X530FN_S530FN
Processor	Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz 1.99 GHz
Operating System	Windows 10 Home Single Language 64-bit
Graphic	Intel® UHD Graphics 620
Memory	4096MB RAM
Storage	931.51GB

Table 4.1 Spec Used

4.2 Software

4.2.1 Jupyter Notebook (Python3)

Python has the ability to connect to database systems. It can also read and alter files, as well as manage large amounts of data and do sophisticated calculations.

Jupyter Notebook was chosen to be used in this project since it was a popular integrated development environment (IDE) for Python that offers several benefits, some of which are:

- **Interactive Computing:** Jupyter Notebook allows you to write and execute code snippets in an interactive way. This means that you can run code cell by cell, making it easy to test and debug code.
- **Data Visualization:** Jupyter Notebook comes with built-in visualization tools that enable you to create and display charts, graphs, and other visualizations of your data directly in the notebook.
- **Markdown Support:** Jupyter Notebook supports Markdown, which allows you to add formatted text, links, and images to your notebook. This makes it easy to add documentation, comments, and explanations to your code.

4.2.2 Anaconda Prompt

Anaconda is a popular data science platform that provides a collection of powerful tools for working with Python. One of the key features of Anaconda is the Anaconda Prompt, which is a command-line interface for running Python code and managing Anaconda environments.

The Anaconda Prompt provides an easy way to launch the graphical user interface (GUI) of Python applications. To run a Python GUI application using the Anaconda Prompt, it can simply navigate to the directory containing the application's code and run the command to launch the GUI. For example, the python script named “IncrementalKMeans.py” that has a GUI can be navigated via this prompt with the command “python IncrementalKMeans.py”. to run the GUI

Overall, the Anaconda Prompt is a convenient tool for running Python code and managing Anaconda environments, and it can help streamline the process of launching Python GUI applications.

4.3 Kmeans extension

An experiment was conducted to compare the performance of k-means clustering algorithms in a GUI environment for user interaction. The study focused on the traditional k-means algorithm, as well as two extensions: incremental k-means and mini-batch k-means.

The traditional k-means algorithm was implemented using the KMeans class from scikit-learn, with 4 clusters and the default initialization parameters. The incremental k-means algorithm was implemented using the IncrementalKMeans class, with a batch size of 100 and a maximum of 10 iterations. The mini-batch k-means algorithm was implemented using the MiniBatchKMeans class, with a batch size of 100 and 1000 maximum iterations.

Incremental k-means updates the clustering model with new data points one at a time, and adapts the centroids of the existing clusters accordingly. This allows the model to learn from new data without having to reprocess the entire dataset, which can be time-consuming for large datasets. However, incremental k-means may be less accurate than traditional k-means due to the smaller batch size.

Mini-batch k-means, on the other hand, uses random subsets (or "mini-batches") of the data to update the clustering model, instead of processing the entire dataset at once. This can be much faster than incremental k-means, as well as traditional k-means, because the algorithm only needs to process a small batch of data at a time. Mini-batch k-means can also be more accurate than incremental k-means, because it uses a larger batch size to update the centroids.

The benefit of mini-batch k-means over incremental k-means is primarily in terms of computational efficiency. By using larger batches of data, mini-batch k-means can process more data in a shorter amount of time, making it well-suited for large datasets or real-time data streams. Additionally, mini-batch k-means can be more accurate than incremental k-means due to the larger batch size. However, mini-batch k-means may require more hyperparameter tuning than incremental k-means to obtain optimal performance. Overall, MiniBatchKmeans was selected in the GUI development. The full source code for the GUI will be included in the appendix chapter.

4.4 GUI

In order to facilitate user interaction with the RFM clustering model, a graphical user interface (GUI) was developed using the PyQt5 library in Python. The GUI was designed to allow users to input the recency, frequency, and monetary values for a customer, and predict which cluster the customer belongs to using mini-batch k-means clustering.

The GUI consists of three input fields for the recency, frequency, and monetary values, respectively, as well as a "Predict" button and a "Plot" button. When the user inputs the values and clicks the "Predict" button, the GUI uses the MiniBatchKMeans class from the scikit-learn library to update the clustering model with the new data point and predict its cluster label. The resulting label is displayed to the user in a text label on result section.

Customer Segmentation

RFM

Customer Segmentation

38

4

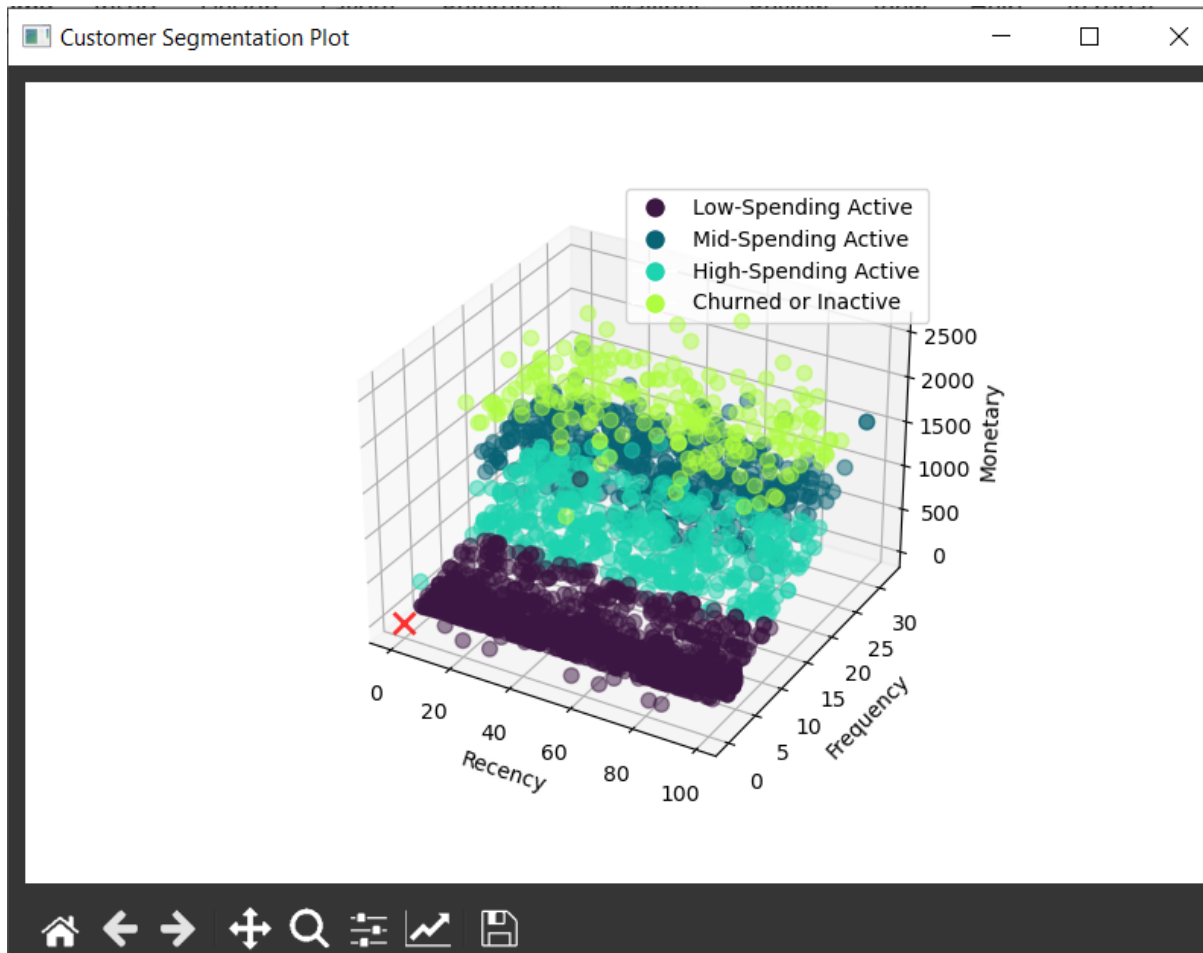
27

Add Del Clear Predict 3D Plot

	ID	Recency	Frequency	Monetary
1	5524	58	22	1617
2	2174	38	4	27
3	4141	26	20	776
4	6182	26	6	53
5	5324	94	14	422
6	7446	16	20	716
7	965	34	17	590
8	6177	32	8	169
9	4855	19	5	46
10	5899	68	1	49

Cluster 0: Low-Spending Active Customers

Additionally, when the user clicks the "Plot" button, the GUI will prompt a plot window of 3D plot of the clustering results, with each cluster represented by a different color. The plot also displays the new data point input by the user, marked by a red "x". The plot also can zoom in/out and can navigate with the mouse to see more clearer the red 'x'.



Furthermore, the function of CRUD also be implemented like user can add new customers data, delete new/old customers data and predict the cluster results. The read function was design as a table shown in Figure , users can straight away select the customers in the table to predicts clusters label or delete the customer data.

Overall, the GUI provides a user-friendly interface for interacting with the RFM clustering model, allowing users to input new data points and explore the clustering results in an intuitive way.

5 Model Evaluation and Discussion

As this analysis was exploratory in nature, traditional evaluation metrics such as accuracy were not applicable, as there was no ground truth for comparison. Therefore, evaluation metrics such as elbow method, calinski-harabaz and silhouette score were used to assess the performance of the segmentation algorithms.

In this study, different approaches were separated from determining the number of clusters and more ease to do the project. Three different files will included in this study which were “Customer segmentation using k-means, GMM, and DBSCAN clustering”, “Customer segmentation using RFM clustering”, and “Visualization after data-preprocessing”. There were various clustering algorithms used in this project as mentioned in chapter 3. The reason why RFM algorithms used in different files are it was a behavior segmentation based compare with others algorithms.

5.1 Determine optimal number of clusters

5.1.1 k-means, GMM, DBSCAN

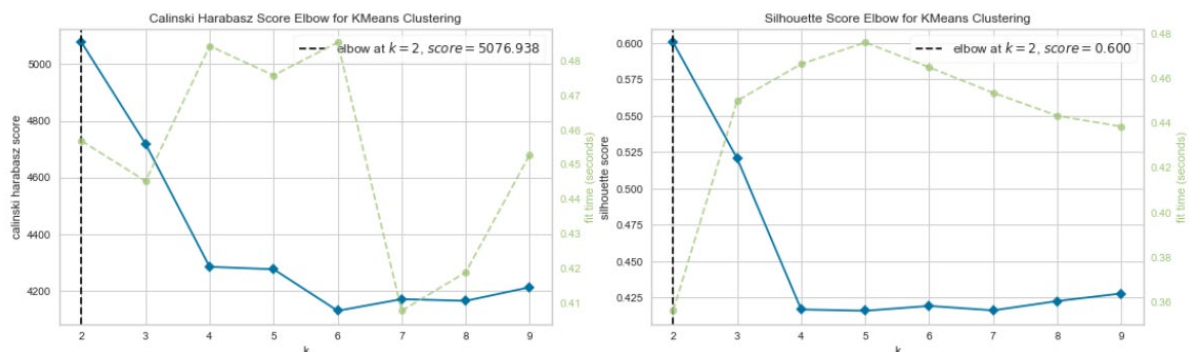


Figure 5.1 Silhouette score and Calinski-Harabaz

Figure 5.1 shows the result is elbow on k=2, the calinski-harabaz was 5076.938 and the silhouette score was 0.600.

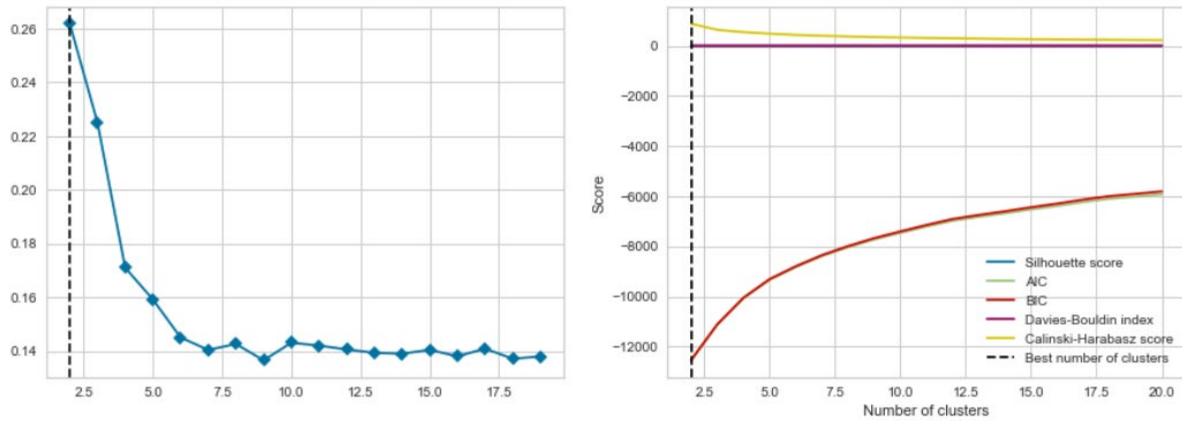


Figure 5.2 Comparison of all metric prediction

Figure 5.2 shows the comparison results of clusters for the dataset as it shows $k = 2$ is the best optimal clusters.

5.1.2 RFM+kmeans

There were two metrics evaluation used which were elbow and silhouette score:

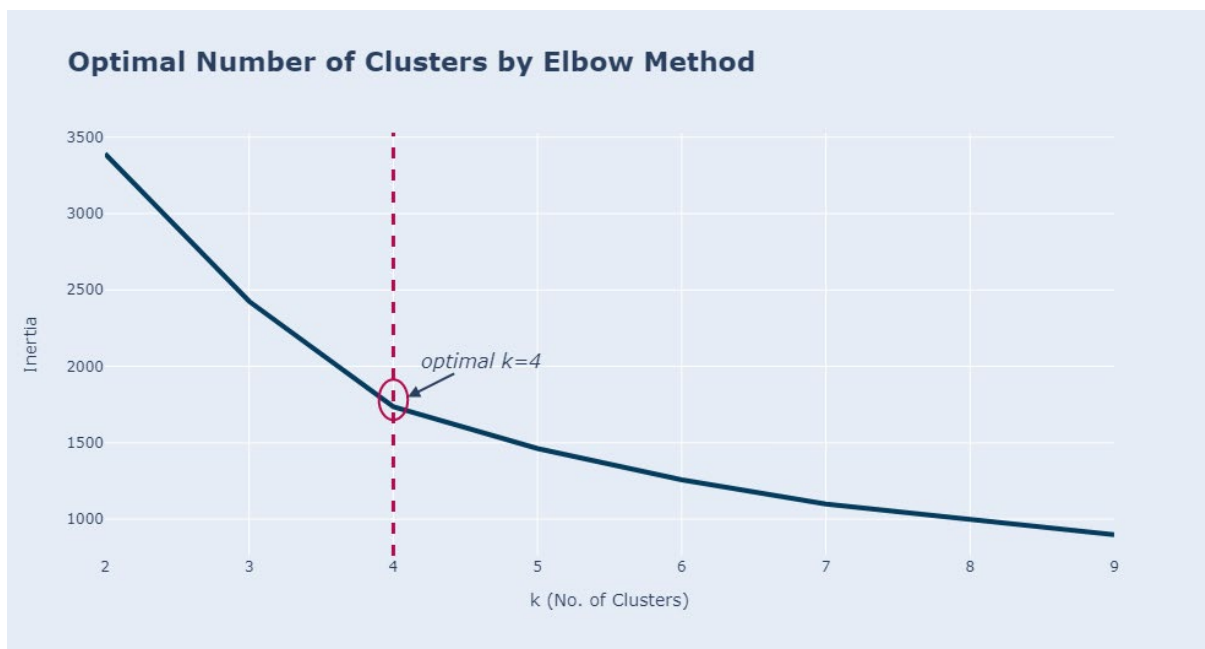


Figure 5.3 Elbow Method for RFM

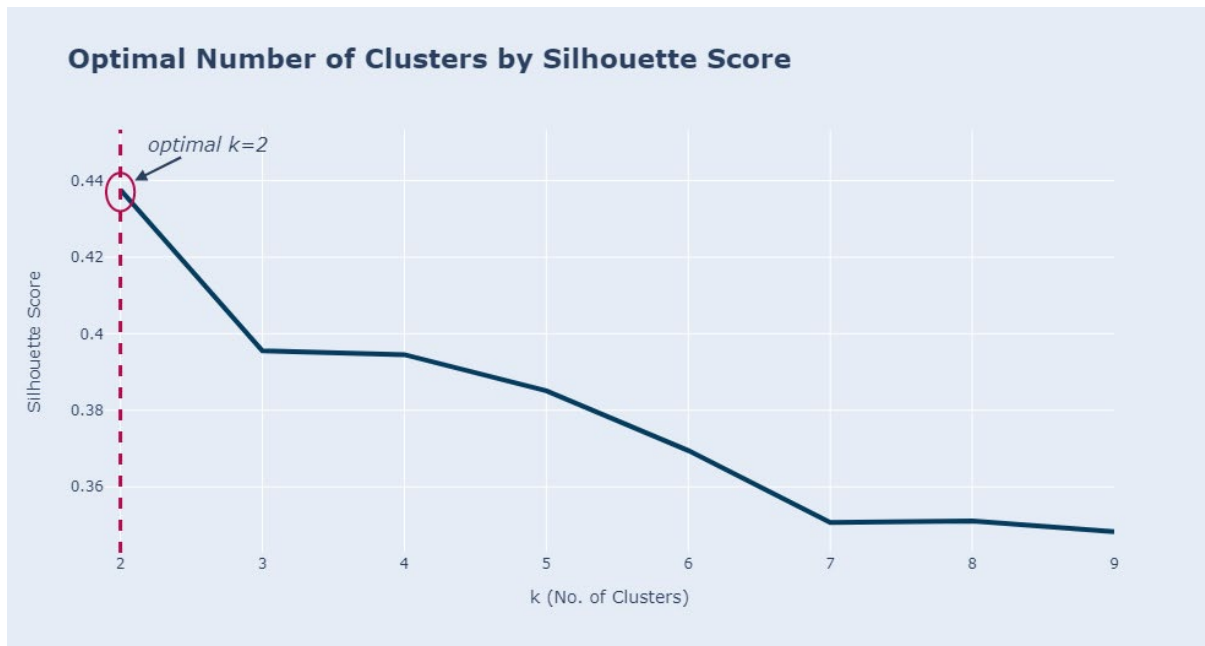


Figure 5.4 Silhouette Score for RFM

The optimal cluster based on the elbow method is 4, while according to silhouette score is 2. Since silhouette score method considers intra and inter clusters, it might produce more separated clusters. However, two clusters might be too general for customer segmentation which require more specific cluster to build personalized offers. Thus, $k=4$ will be used to perform clustering [1].

5.2 Clustering Evaluation

5.2.1 k-means and GMM

Clustering Metrics Summary

Number of clusters	Best Silhouette	Best Davies-Bouldin	Best AIC	Best BIC	Best Calinski-Harabasz	AIC	BIC	Calinski-Harabasz score	Silhouette score	Davies-Bouldin Index
2	Yes	Yes	Yes	Yes	Yes	-12545.583310	-12534.180006	880.854806	0.261810	1.524242
3	No	No	No	No	No	-11126.809086	-11109.704129	636.812010	0.226819	1.665257
4	No	No	No	No	No	-10072.525883	-10049.719274	545.477758	0.170822	1.793700
5	No	No	No	No	No	-9334.267688	-9305.759426	484.614927	0.159262	1.904951
6	No	No	No	No	No	-8828.459752	-8794.249838	434.743893	0.144878	1.878657
7	No	No	No	No	No	-8373.980177	-8334.068611	401.480956	0.141444	1.829221
8	No	No	No	No	No	-8022.008488	-7976.395269	372.655559	0.142395	1.756451
9	No	No	No	No	No	-7725.953608	-7674.638737	348.758371	0.140884	1.805264

Figure 5.5 Clustering Metrics Summary for k-means and GMM

To identify the optimal number of clusters in our customer segmentation analysis, several evaluation metrics were used to assess the performance of different clustering algorithms. Specifically, looked for the number of clusters that best satisfied the following criteria: a

lower value of AIC or BIC, a higher value of the Calinski-Harabasz score, a Silhouette score approaching 1, and a lower Davies-Bouldin Index score. A lower value of AIC or BIC indicates a better fit for the model, while a higher value of the Calinski-Harabasz score indicates better clustering performance. The Silhouette score, which ranges from -1 to 1, can be used to evaluate the quality of clustering, with a score of 1 denoting the best clustering and a score approaching 1 being desirable. The Davies-Bouldin Index, which ranges from 0 to 1, can be used to evaluate the compactness and separation of clusters, with a lower score indicating better clustering performance. By using these evaluation metrics, the optimal number of clusters were able to identify for the customer segmentation analysis, gain insights into the different customer segments and their characteristics.

5.2.2 DBSCAN

Optimal Epsilon: 0.1

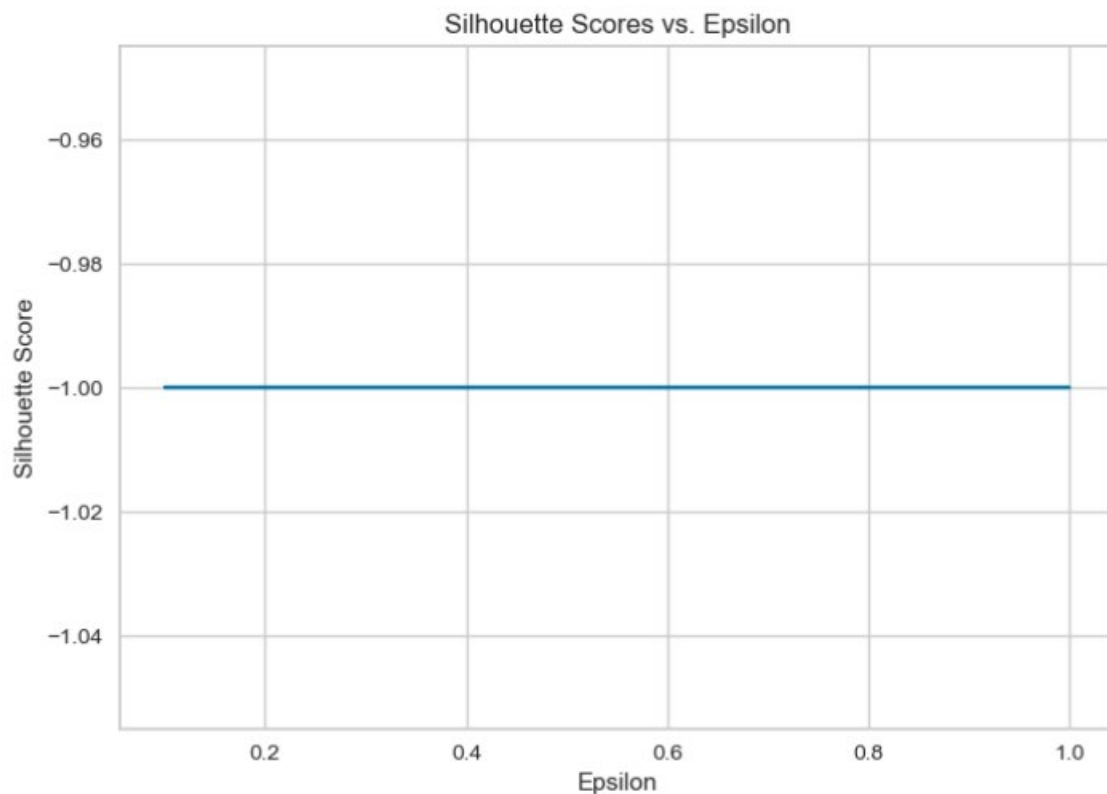


Figure 5.6 GridSearch for eps

index	EPS	Min Samples	Best Silhouette	Best Davies-Bouldin	Calinski-Harabasz score	Silhouette score	Davies-Bouldin Index
0	0.100000	2	No	No	2.233672	-0.394302	1.065094
1	0.100000	3	No	No	2.958715	-0.365076	1.083905
2	0.200000	2	No	No	2.233141	-0.385836	1.067134
3	0.200000	3	No	No	2.851223	-0.411198	1.085378
4	0.300000	2	No	No	2.277688	-0.360172	1.078317
5	0.300000	3	No	No	2.760099	-0.460021	1.118262
6	0.300000	4	Yes	Yes	4.168942	-0.132906	1.004028
7	0.400000	2	No	No	2.571403	-0.302707	1.084512
8	0.400000	3	No	No	3.651384	-0.448457	1.130721
9	0.400000	4	No	No	4.215943	-0.239126	1.155639

Figure 5.7 Clustering Metrics Summary for DBSCAN

The choice of clustering metrics differs between DBSCAN and k-means/GMM due to the unique hyperparameters of DBSCAN, namely ϵ and `min_samples`, which are distinct from the traditional number of clusters or components. In Figure 5.6, the clustering results were analyzed using `GridSearchCV()` based on the Silhouette Score, resulting in an optimal ϵ value of 0.1. Furthermore, Figure 5.7 highlights the best combination of ϵ and `min_samples` as 0.3 and 4, respectively. Consequently, these values were selected to achieve more accurate clustering results with DBSCAN compared to the previous settings.

5.2.3 3D Plot Performance

In 3D plots, clusters can be represented by distinct shapes or colors, making it easier to visually identify and explore the separation between clusters. This can help in understanding the distinct characteristics of different clusters and how they are distributed in the data space. And the following figures was show all the three algorithms concerns the clusters of two with different colors for more appealing.

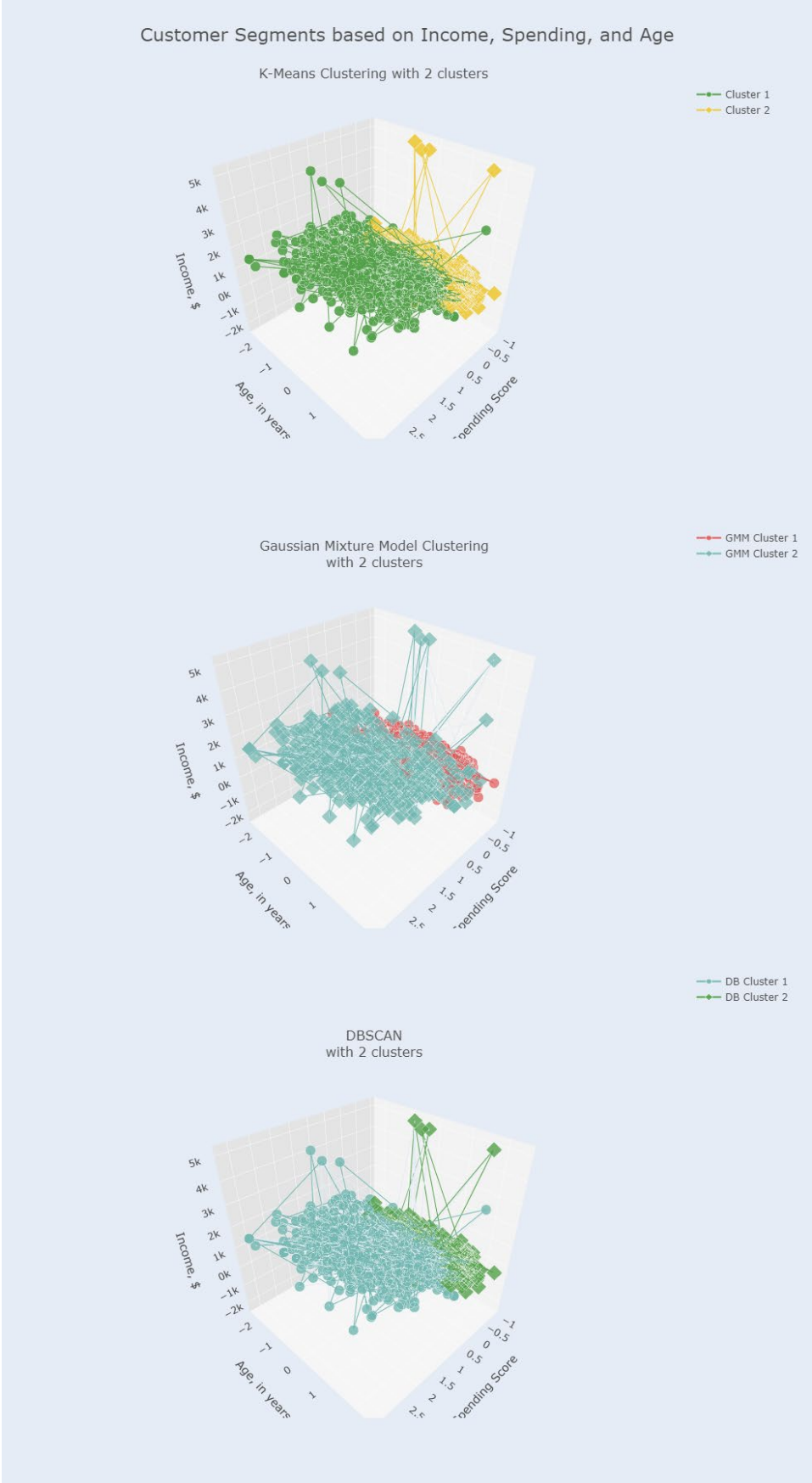


Figure 5.8 Comparison between kmeans, GMM and DBSCAN

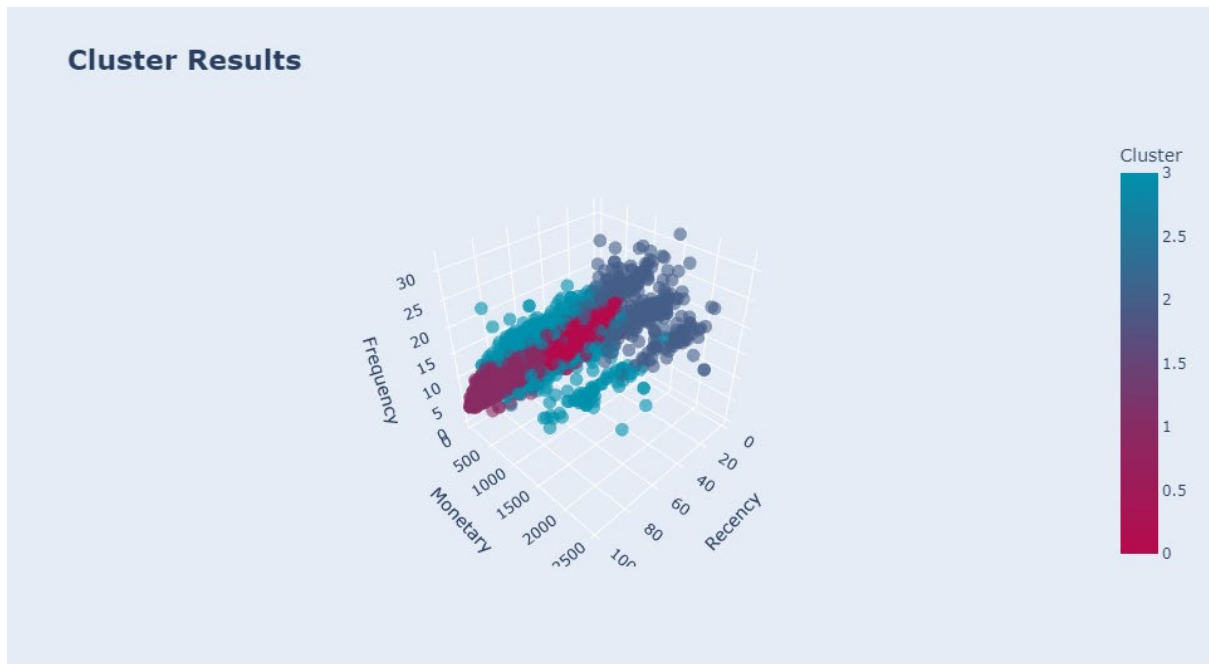


Figure 5.9 3D plot for RFM

5.3 Cluster Analysis (k-means, GMM and DBSCAN)

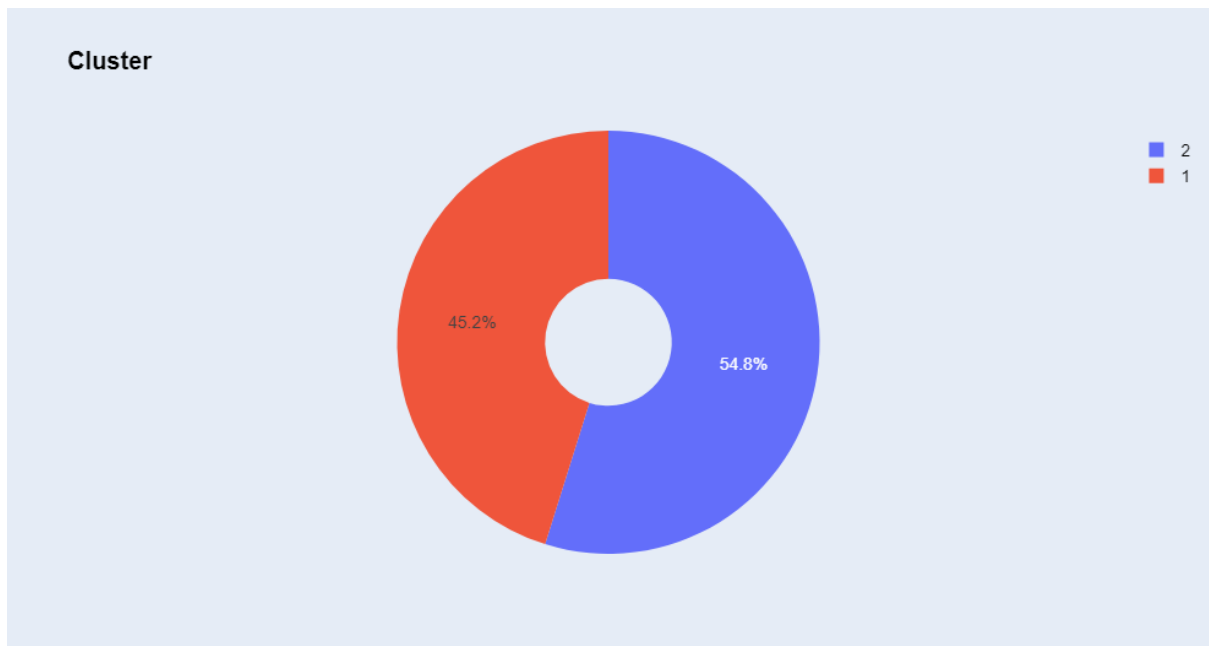


Figure 5.10 Tabulate total number of clusters

The result shows that there were total number of 999 customers in cluster 1 and total number of 1213 customers in cluster 2 and hence the number of clusters 2 is more than cluster 1.

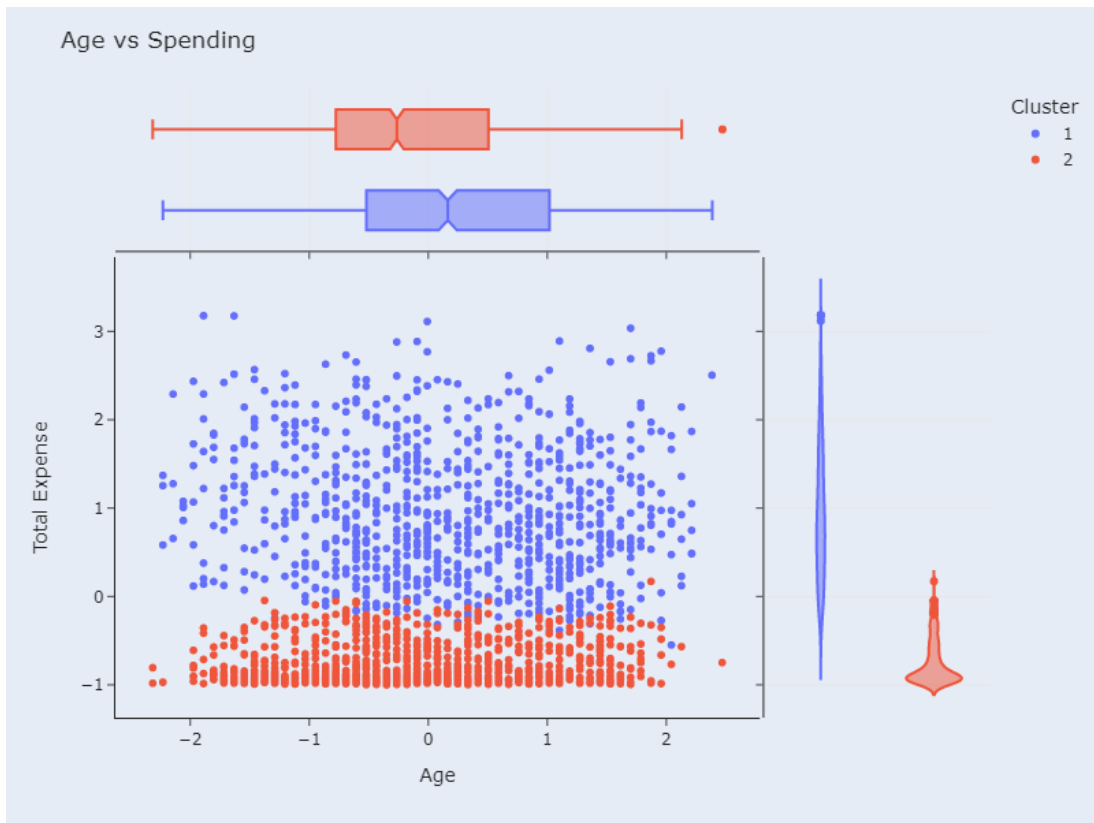


Figure 5.11 Age vs spending

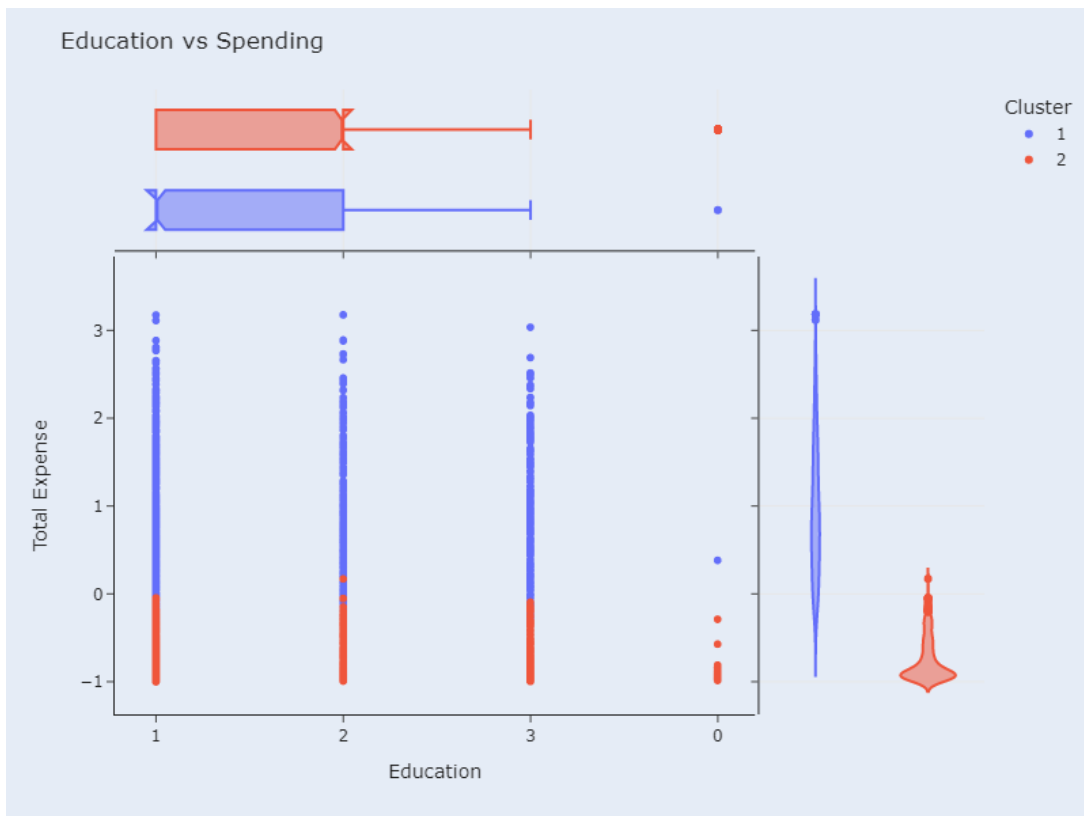


Figure 5.12 Education vs Spending

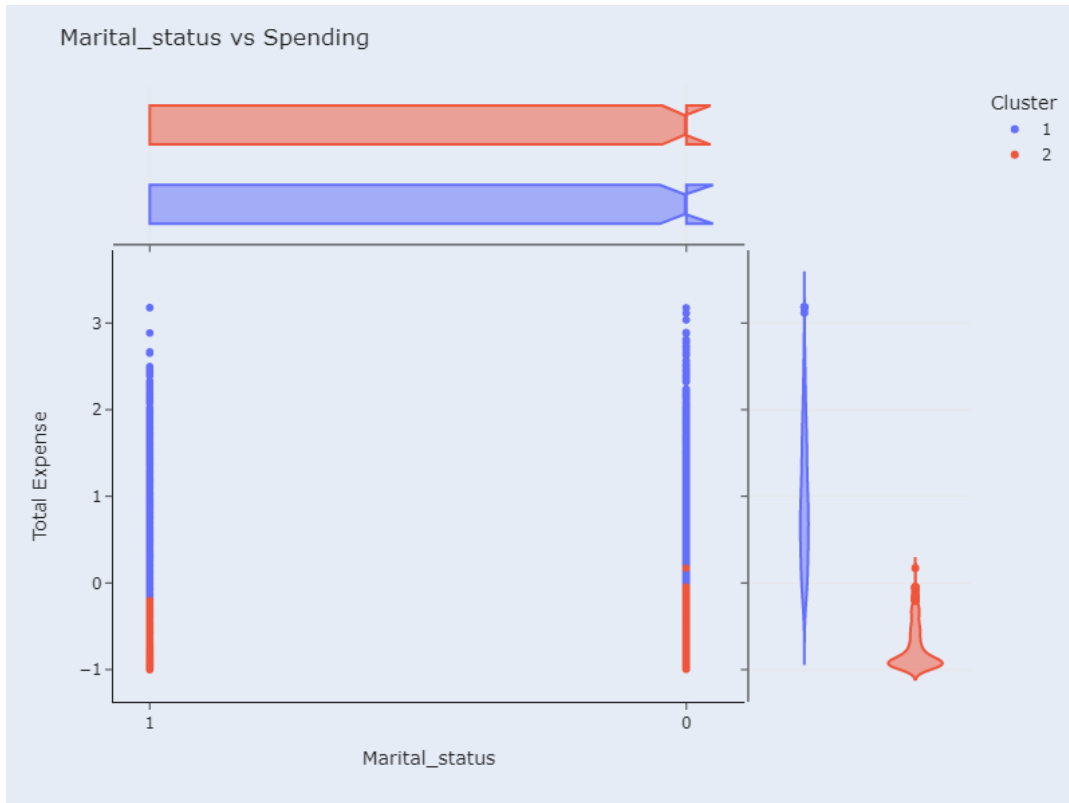


Figure 5.14 Marital Status vs Spending

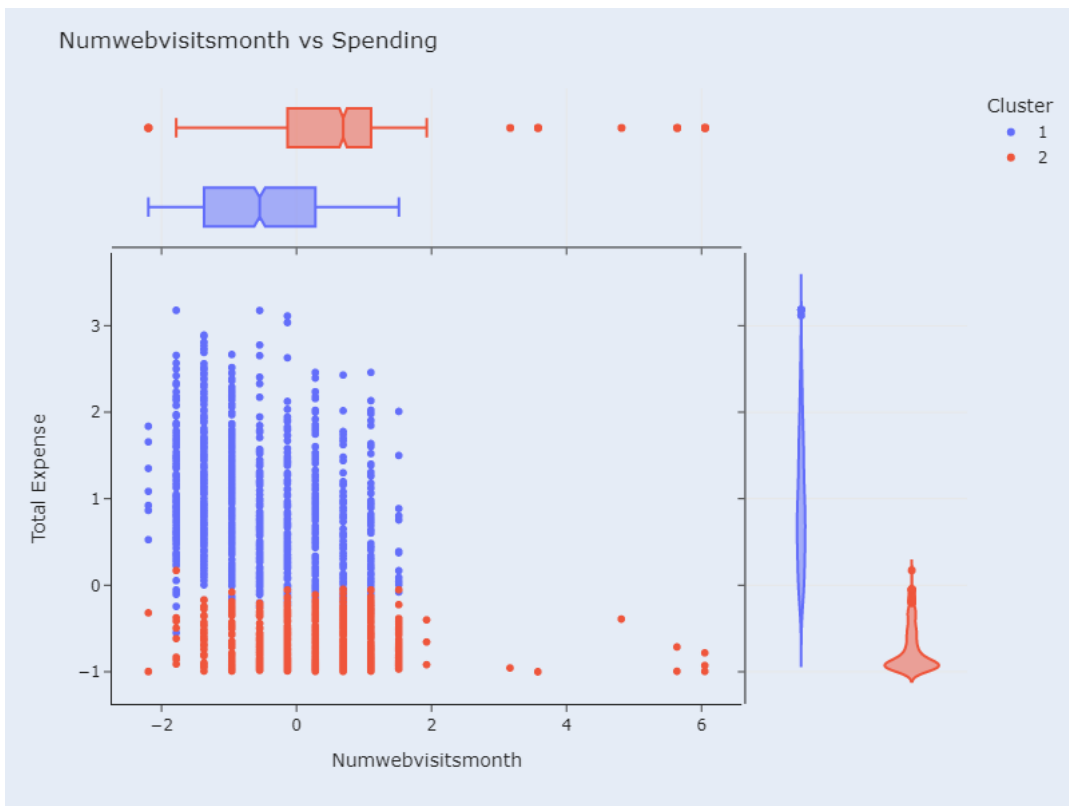


Figure 5.13 NumwebvisitsMonth vs Spending

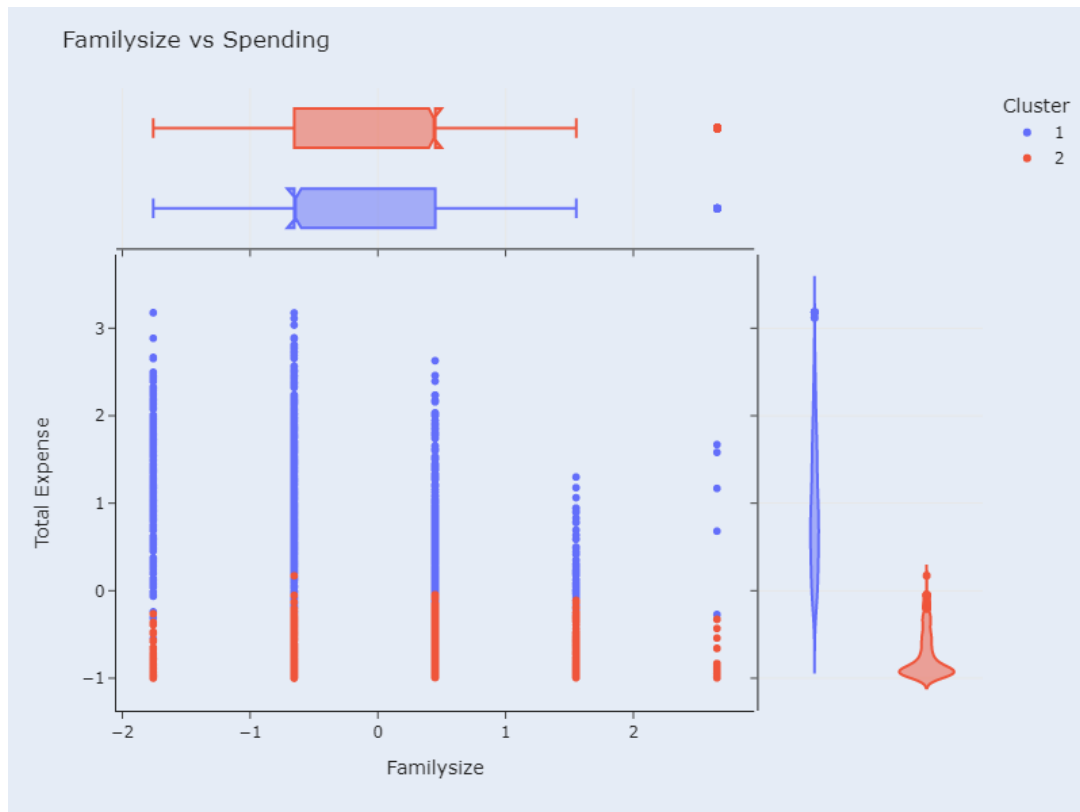


Figure 5.15 Familysize vs Spending

The summary insight in figure 5.11 5o figure 5.15 were:

Cluster 1: Highly Active and Moderately Active Customers

Customers in this cluster have a professional or postgraduate (PG) background. They generally have higher incomes compared to the Least Active Customers. Their ages range from 25 to 75, with the majority falling between 35 to 60 years old. The number of children varies, with Highly Active Customers having, on average, more children than Moderately Active Customers. In terms of spending, Moderately Active Customers tend to spend more than Highly Active Customers. Both segments have been engaged with the company for a longer period compared to the Least Active Customers. Additionally, they exhibit a tendency to accept at least one campaign.

Cluster 2: Least Active Customers

Customers in this cluster typically have an undergraduate (UG) educational background. Their income is very low or almost negligible. They tend to be younger, with ages ranging from 15 to 30. They have very few or no children. Their spending on the company's products or services is minimal. Furthermore, their level of engagement with the company over an

extended period is limited. These customers demonstrate low activity in accepting any campaign.

5.3.1 Cluster Analysis (RFM)



Figure 5.16 Tabulate number of customers in each cluster

The figure 5.16 concern that number of customers in cluster 1 is the highest while the number of customers in cluster 2 is the lowest.

Clusters	Recency	Frequency	Monetary
0	23.490998	7.063830	151.725041
1	73.450886	7.020934	146.524960
2	22.971922	19.773218	1185.524838
3	73.170213	19.278530	1181.205029

Figure 5.17 Summary boxplot

Based on figure 5.17, The clusters separate the values of RFM into two groups, low and high, as observed from the boxplot and centroid analysis.

The Recency values are categorized as follows: Low (22-23 days) and High (73 days). The Frequency values are divided into Low (7 purchases) and High (19 purchases). Monetary

values show a separation between Low (146-151 USD) and High (1181-1184 USD) ranges. These RFM differentiations form a combination of four customer segments.

- Cluster 0 represents the group of Low-Spending Active Customers with low recency, low frequency, and low monetary values. This segment comprises 619 customers, which accounts for 28% of the total customer base.
- Cluster 1 consists of Churned Low-Spending Customers with high recency, low frequency, and low monetary values. These customers display low engagement and minimal spending. Unfortunately, this segment comprises a high number of customers, specifically 631, representing 28% of the total customer base.
- Cluster 2 identifies the Best Active Customers with low recency, high frequency, and high monetary values. These customers are highly valuable to the company, as they make frequent high-value purchases. However, their number is relatively small, comprising only 466 customers or 21% of the total customer base.
- Cluster 3 represents the Churned Best Customers, characterized by high recency, high frequency, and high monetary values. Although these customers have contributed significantly to the company's revenue, their last purchase occurred a long time ago, potentially indicating churn. This cluster consists of 524 customers, equivalent to 23% of the total customer base.



Figure 5.18 RFM distribution

Based on Figure 5.18, it is evident that Cluster 2 stands out as the best customer segment when compared to the other clusters. This is primarily due to their lower recency, higher frequency, and higher monetary values. Specifically, customers in this cluster tend to visit the company for shorter periods but exhibit higher visit frequency and spending patterns. These customers may visit the company multiple times within a week and spend significantly more on their purchases.

On the other hand, customers in Cluster 3 demonstrate higher values in terms of recency, frequency, and monetary factors, indicating that they are highly valuable customers. They make substantial purchases and spend more, but their visit intervals are longer, suggesting a longer gap between their visits to the company.

In contrast, customers in Cluster 0 exhibit considerably lower values across all three dimensions—recency, frequency, and monetary. This cluster represents the worst-performing group within the dataset, as these customers display lower levels of engagement, visit the company less frequently, and make smaller purchases compared to the other clusters.

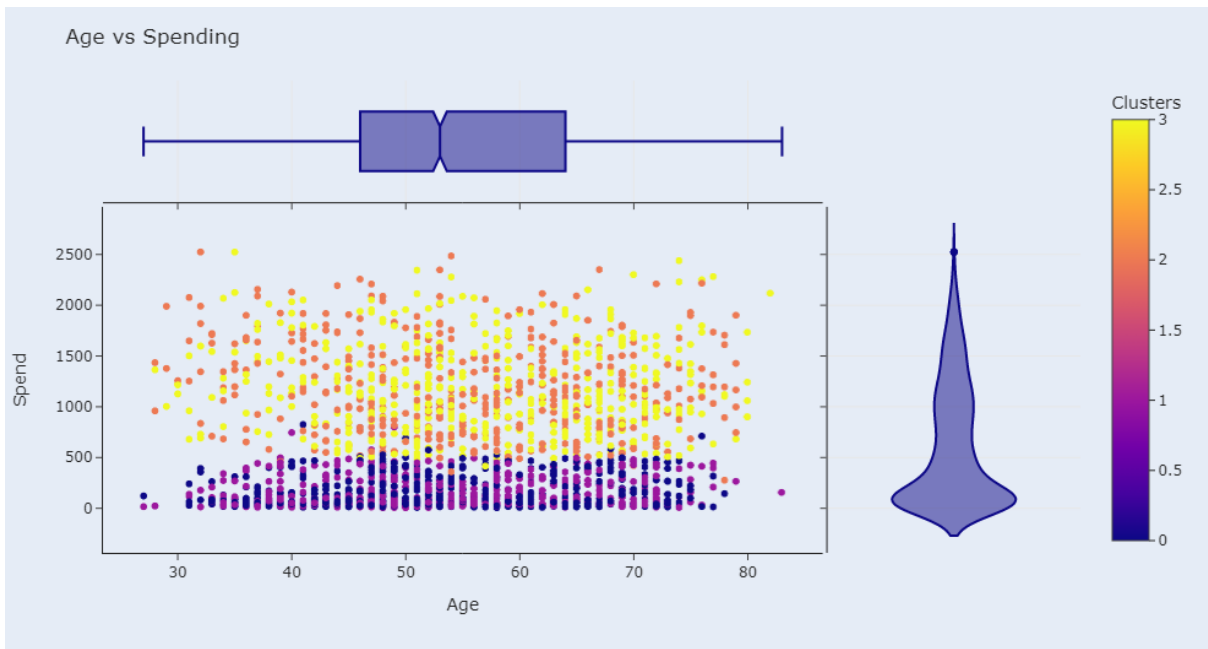


Figure 5.19 RFM Age vs Spending

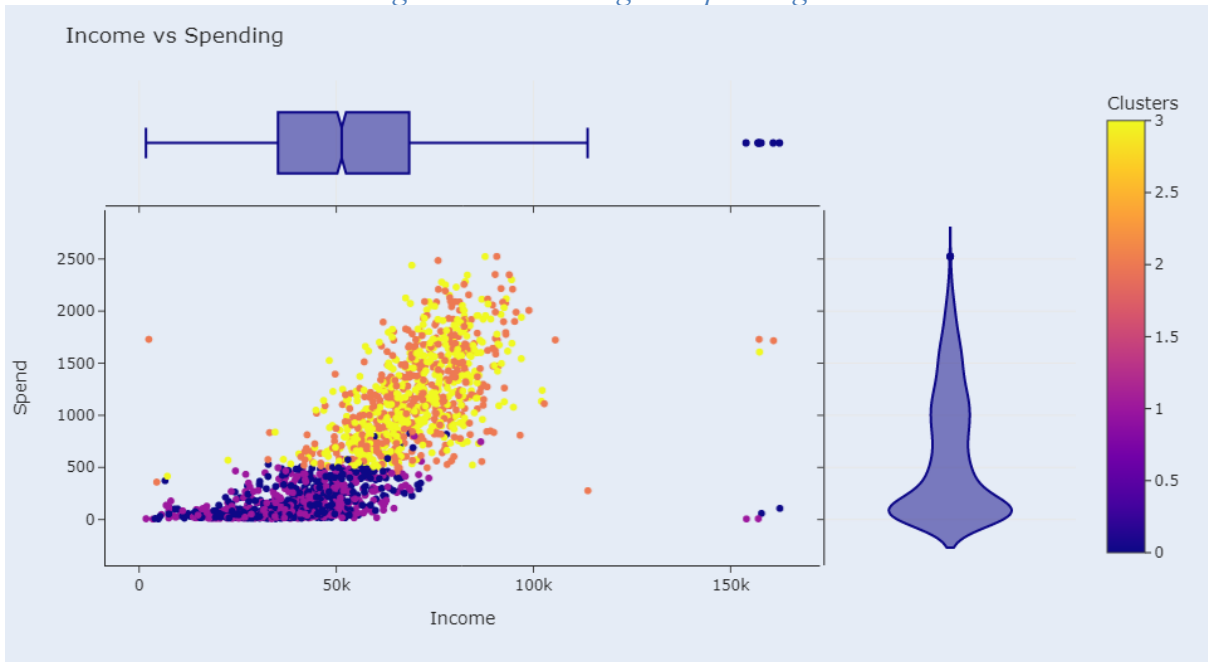


Figure 5.20 RFM Income vs Spending

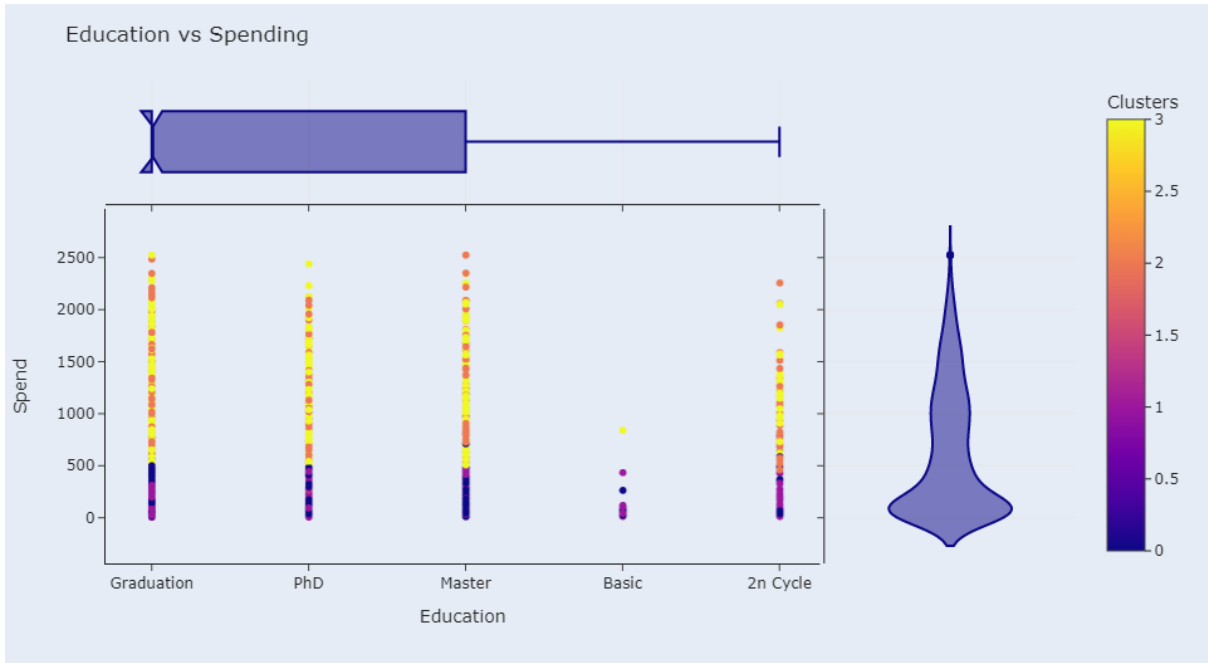


Figure 5.21 RFM Education vs Spending

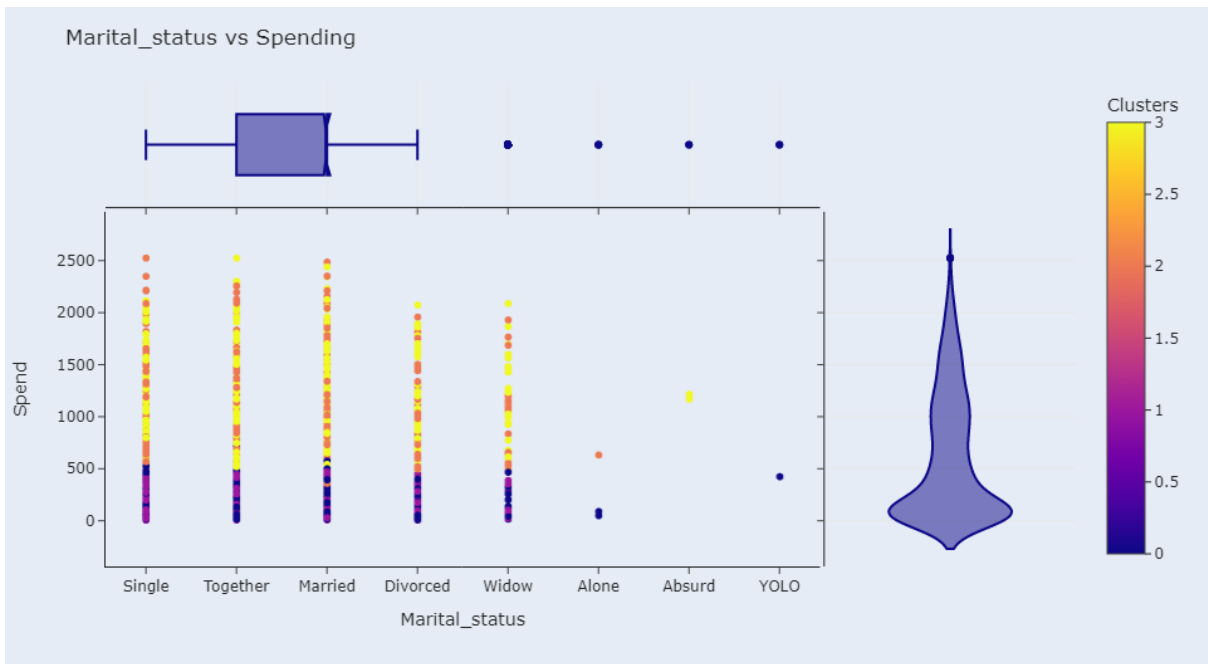


Figure 5.22 RFM Education vs Spending

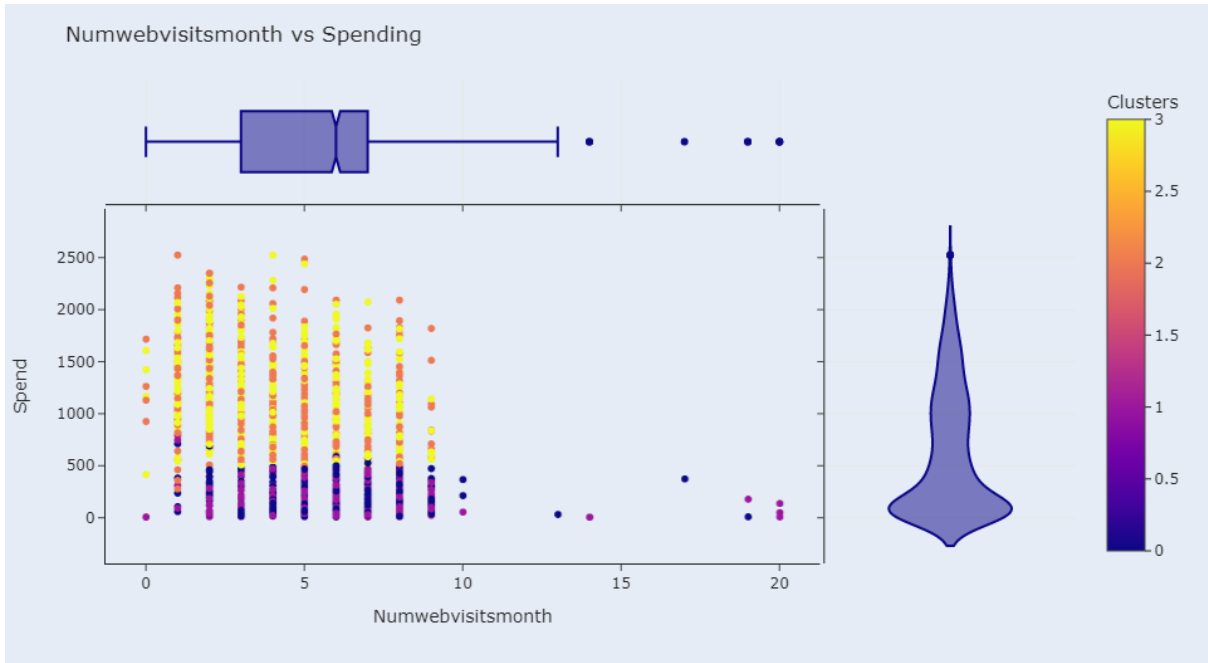


Figure 5.23 RFM Numwebvisitsmonth vs Spending

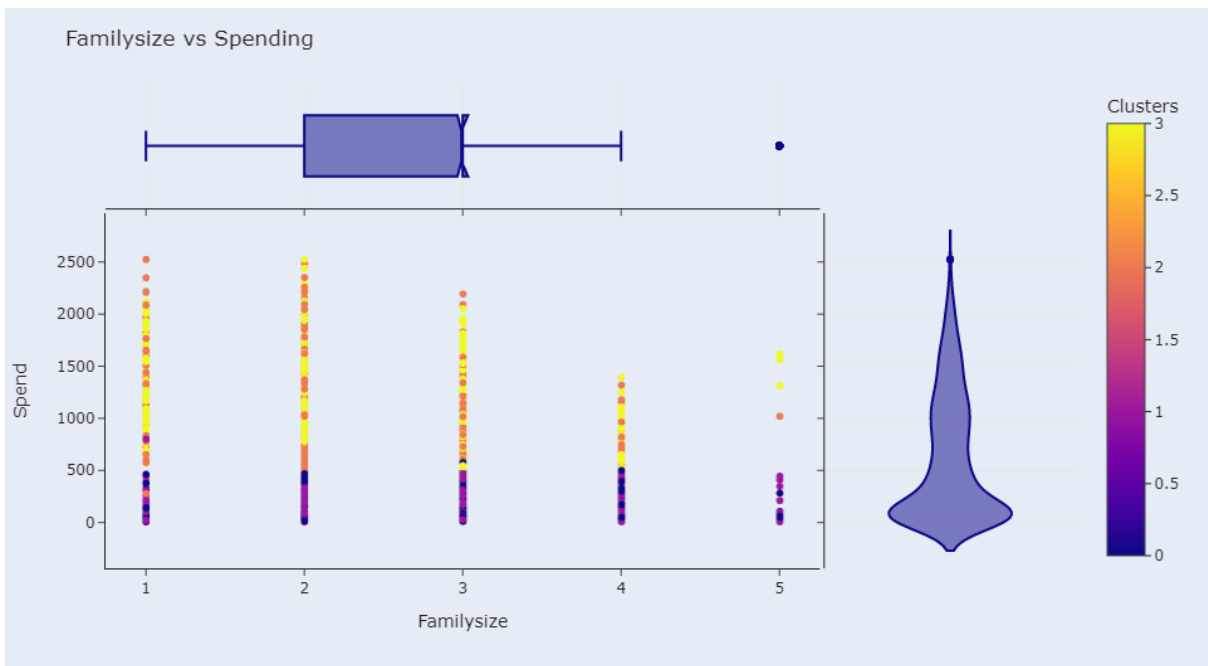


Figure 5.24 RFM Familysize vs Spending

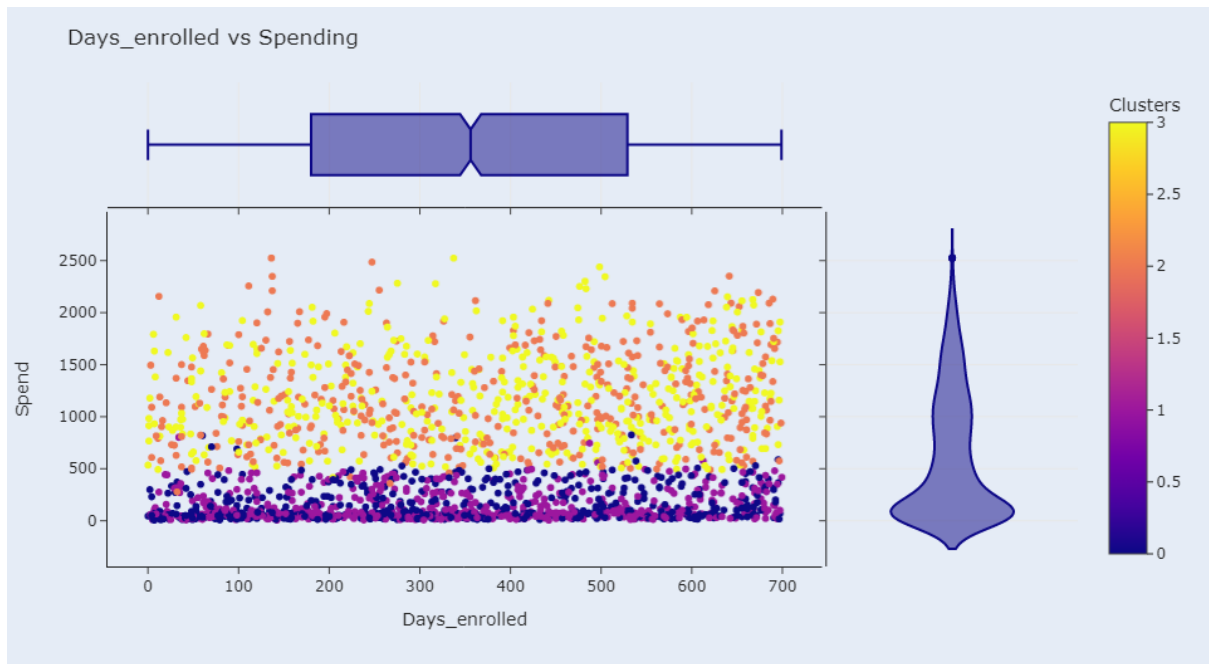


Figure 5.25 RFM Days_enrolled vs Spending

The output of Figure 5.19 to figure 5.35 was concluded at below:

1. Low-Spending Active Customers:

- These customers have an income range of 5000 to 40000 and typically spend between 0 to 500 units. They fall within the age range of 25 to 50 and come from various educational backgrounds. They can be both married and unmarried individuals, with many of them being parents, some with one child. These customers have been with the company for at least 300 days and tend to show rare acceptance of promotions. Additionally, they rarely complete purchases using discounts.

2. Churned Low-Spending Customers:

- Customers in this segment have a higher income range of 65000 to 85000 but still exhibit low spending patterns, ranging from 550 to 2000 units. Their age ranges from 30 to 60, and the majority have completed graduation. Most of them are married individuals without children. They have been customers for at least 250 days, with a moderate promotion acceptance ratio of 0.5. Completing purchases using discounts is rare among these customers.

3. Best Active Customers:

- The Best Active Customers segment comprises customers with an income range of 50000 to 80000, demonstrating higher spending levels ranging from

250 to 1800 units. Their age ranges from 35 to 60, and most have completed graduation. They are predominantly married individuals who are parents, with the majority having one child. These customers have been with the company for at least 400 days. Although they show a poor promotion acceptance ratio, they are highly interested in completing purchases using discounts.

4. Churned Best Customers:

- Customers in this segment have an income range of 40000 to 60000 and exhibit lower spending patterns, ranging from 0 to 500 units. Their age ranges from 40 to 65, and most have completed graduation. They can be both married and unmarried individuals, and like the previous segment, they are also parents, with the majority having two children. These customers have been with the company for at least 150 days but show rare acceptance of promotions. However, they display a high level of interest in completing purchases using discounts.

6 Conclusion and Recommendations

6.1 Summary

This report explored the application of various clustering techniques, including K-means, GMM, DBSCAN, and RFM analysis, for customer segmentation. The objective was to identify distinct customer groups based on their purchasing behavior and preferences using cluster analysis. Through the analysis and evaluation of these techniques, valuable insights into the segmentation process were gained. The dataset used lacked accuracy and ground truth, making it inherently challenging.

Initially, after preprocessing the data, K-means clustering was employed, which is a widely used method for partitioning data into groups. It determined clusters based on the proximity of customers in the feature space. However, K-means assumes spherical-shaped clusters with equal variances, which may not always hold true in real-world scenarios.

Subsequently, GMM was investigated as a probabilistic clustering algorithm. GMM considers the underlying distribution of data points and assigns probabilities to each point belonging to a specific cluster. This approach allows for more complex cluster shapes and accommodates overlapping clusters. GMM effectively captured hidden patterns within the dataset.

Next, DBSCAN, a density-based clustering algorithm, was examined. DBSCAN identifies dense regions of points separated by sparser areas. It does not assume any specific cluster shape and handles outliers effectively. DBSCAN revealed clusters of varying densities and exposed non-linear structures in the data.

Lastly, RFM analysis was employed, a valuable technique for customer segmentation based on recency, frequency, and monetary value. RFM provided insights into customer transactional behavior, allowing for the classification of customers into distinct segments with unique characteristics and marketing implications.

Throughout the analysis, each clustering technique demonstrated strengths and limitations. K-means offered a straightforward and interpretable approach but struggled with complex and

non-linear structures. GMM provided more flexibility but required careful consideration of the number of components. DBSCAN excelled in identifying clusters with varying densities but could be sensitive to parameter selection, specifically k , ϵ , and min_samples . RFM analysis complemented the clustering techniques by incorporating transactional data and offering insights into customer value. In RFM analysis, the selection of the optimal number of clusters, k , was a critical consideration. A comparison of different performance metrics was lacking in previous studies, and the elbow method was commonly used.

In addition to the clustering techniques, a user-friendly GUI was developed based on RFM analysis and the Mini-Batch K-means algorithm. This GUI provides businesses with an intuitive platform to perform customer segmentation on their own datasets. By incorporating RFM analysis, the GUI enables users to analyze customer transactional behavior and identify valuable customer segments. The integration of the Mini-Batch K-means algorithm ensures efficient and scalable clustering, making it suitable for large datasets. The GUI's visualizations and interactive features enhance the interpretation and exploration of the segmentation results, empowering businesses to make data-driven decisions.

6.2 Further Study

1. Dimensionality Reduction

- One technique for reducing dimensionality is t-SNE. This nonlinear method focuses on preserving local structures and clusters within the data, making it valuable for visualizing high-dimensional data. By employing t-SNE, it becomes easier to identify hidden patterns or relationships that may not be immediately apparent in the original feature space. Reducing dimensionality with t-SNE provides insights into customer grouping and separability, thereby assisting subsequent clustering processes.

2. Clustering Algorithms

- SOM or Kohonen maps, are unsupervised neural networks that utilize competitive learning to organize and visualize high-dimensional data. SOM constructs a grid of neurons, where each neuron represents a cluster or prototype. Throughout training, the neurons adapt their weights to accurately represent various regions within the input data space. SOM excels at preserving the topological structure of the data, revealing relationships and clusters that may not be initially evident. It is commonly employed for visualizing and exploring high-dimensional data.
- Autoencoders are neural network models employed for reconstructing input data from a compressed or bottleneck representation. In the context of clustering, autoencoders serve as a dimensionality reduction technique, with the bottleneck layer representing a lower-dimensional representation of the input data. The encoder part of the autoencoder maps the input data to the bottleneck layer, while the decoder part reconstructs the original data. By training an autoencoder on customer data, the bottleneck layer captures essential features for clustering. Subsequent clustering algorithms can then be applied to the reduced-dimensional representation. Autoencoders excel at capturing non-linear relationships and identifying complex clusters within the data.

REFERENCES

Article:

Journal article

[1] Lee, Y., Kwon, O., & Lee, Y. (2015). Customer segmentation using purchase data for an online retailer. *Expert Systems with Applications*, 42(1), 332-341. DOI: 10.1016/j.eswa.2014.08.020

[2] Kumar, A., Jain, A., Jain, S., & Jain, S. (2016). Comparative study of clustering algorithms for customer segmentation in e-commerce. In *2016 International Conference on Computing, Analytics and Security Trends (CAST)* (pp. 300-305). IEEE. DOI: 10.1109/CAST.2016.79

[3] Xiang, Y., & Gong, Y. (2018). Online Shopping Behavior Analysis Based on K-means, GMM and DBSCAN Clustering Algorithm. In *2018 International Conference on Computational Science and Computational Intelligence (CSCI)* (pp. 1122-1126). IEEE. DOI: 10.1109/CSCI46756.2018.00209

[4] Chen, X., Zuo, X., Wu, Z., & Liu, X. (2020). A Comparative Study of Customer Segmentation Methods Based on Online Shopping Behavior. In *2020 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)* (pp. 791-795). IEEE. DOI: 10.1109/IEEM47687.2020.9378743

[5] Jadhav, M., & Sonawane, K. (2021). Credit card fraud detection using clustering techniques. In *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)* (pp. 1-6). IEEE. DOI: 10.1109/CONFLUENCE51715.2021.9461624

[6] Yeh, Y.-L., & Huang, C.-C. (2018). Customer segmentation of bicycle-sharing users based on their usage behavior. *Sustainability*, 10(5), 1579. DOI: 10.3390/su10051579

[7] Allenby, G., Fennell, G., Bemmaor, A., Bhargava, V., Christen, F., Dawley, J., Dickson, P., Edwards, Y., Garratt, M., Ginter, J., Sawyer, A., Staelin R. & Yang, S. (2002) *Market*
Bachelor of Computer Science (Honours)
Faculty of Information and Communication Technology (Kampar Campus), UTAR

segmentation research: beyond within and across group differences. *Marketing Letters*, 13, 3, pp. 233–243.

[8] Hyunseok Hwang, Taesoo Jung, Euiho Suh

An LTV model and customer segmentation based on customer value: A case study on the wireless telecommunication industry

Expert Systems with Applications, 26 (2004), pp. 181-188

[9] Dolničar, S. & Leisch, F. (2004) Segmenting markets by bagged clustering. *Australasian Marketing Journal*, 12, 2, pp. 51–65.

[10] Dimitriadou, E., Dolničar, S. & Weingessel, A. (2002) An examination of the number of indexes for determining the number of clusters in binary data sets. *Psychometrika*, 67, 2, pp. 137–160.

[11] Decker, R., Wagner, R. & Scholz, S.W. (2005) Growing clustering algorithms in market segmentation: defining target groups and related marketing communication. In H.-H. Bock, W. Gaul & M. Vicki (eds) *Data Analysis, Classification and the Forward Search*. Berlin: Springer, pp. 23–30.

Online Article:

[12] Qualtrics, "Customer Segmentation: Definition & Methods", *Qualtrics AU*, 2020.

[Online].

Available:

<https://www.qualtrics.com/au/experience-management/brand/customer-segmentation/?rid=ip&prevsite=en&newsite=au&geo=MY&geomatch=au>.

[13] D. Gong, "Clustering Algorithm for Customer Segmentation", *Medium*, 2021. [Online].

Available:

<https://towardsdatascience.com/clustering-algorithm-for-customer-segmentation-e2d79e28cbc3>.

Book:

[14] Aldenderfer, M.S. & Blashfield, R.K. (1984) Cluster Analysis. Beverly Hills: Sage Publications.

[15] A. Hizioglu, "Soft computing applications in customer segmentation: State-of-artreview and critique," *Expert Syst Appl.*

Appendix

The following code was used to **impute missing value**:

```
#Mice Imputer
df_mice = df.copy(deep=True)
mice_imputer = IterativeImputer()
df['Income'] = mice_imputer.fit_transform(df[['Income']])
```

```
#Check the present of missing values
present = 0
for col in df.columns:
    if df[col].isnull().sum() > 0:
        print(f'Column '{col}' has {df[col].isnull().sum()} missing values")
        present = 1
if not present:
    print("No missing values are present in the dataset")
```

Here is an example of the code used for some of the **cleaning steps**:

```
# change date format
df['Dt_Customer'] = pd.to_datetime(df['Dt_Customer'], format='%d-%m-%Y')

#Gather all the AcceptedCmps and response in one features
df['Total_AcceptedCampaign'] =
df[['AcceptedCmp1','AcceptedCmp2','AcceptedCmp3','AcceptedCmp4','AcceptedCmp5','Res
ponse']].sum(axis=1)

# convert categorical features to factor
categorical_features = ["Education", "Marital_Status", "AcceptedCmp1",
                        "AcceptedCmp2", "AcceptedCmp3", "AcceptedCmp4",
                        "AcceptedCmp5", "Complain", "Response"]
for feature in categorical_features:
    df[feature] = df[feature].astype('string')
```

Calculate age

```
df['Age'] = dt.datetime.now().year - df['Year_Birth']
```

Calculate last enrollment date

```
last_enrollment = df['Dt_Customer'].max()
```

```
# Calculate days enrolled
```

```
df['Days_Enrolled'] = (last_enrollment - df['Dt_Customer']).dt.days
```

total spendings

```
df['Spend'] = df[['MntWines', 'MntFruits', 'MntMeatProducts', 'MntFishProducts',  
'MntSweetProducts', 'MntGoldProds']].sum(axis=1)
```

```
df['Familysize'] = df['Marital_Status'].replace({"Married": "Relationship",  
"Together": "Relationship",  
"Widow": "Single", "Divorced": "Single",  
"Single": "Single", "Alone": "Single", "Absurd": "Single",  
"YOLO": "Single"}).replace({'Single': 1, 'Relationship': 2}).fillna(0).astype(int) +  
df['Kidhome'] + df['Teenhome']
```

#Total purchases

```
df['Total_Purchases'] =  
df[['NumDealsPurchases', 'NumWebPurchases', 'NumCatalogPurchases', 'NumStorePurchases']  
,].sum(axis=1)
```

remove features not used

```
df = df.drop(['Year_Birth', 'MntWines', 'MntFruits', 'MntMeatProducts', 'MntFishProducts',  
'MntSweetProducts', 'MntGoldProds', 'Kidhome', 'Teenhome'], axis=1)
```

```
df.drop(columns=['Dt_Customer', 'AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3',  
'AcceptedCmp4', 'AcceptedCmp5', 'Complain',  
'Response', 'NumDealsPurchases', 'NumWebPurchases', 'NumCatalogPurchases', 'NumStorePur  
chases'], inplace=True)
```

Identify numerical columns and apply standard scaling

```

int_list = []
for col in df.columns:
    if df[col].dtypes == int or df[col].dtypes == float:
        int_list.append(col)

autoscaler = StandardScaler()
df[int_list] = autoscaler.fit_transform(df[int_list])

```

```

# Identify categorical columns and apply label encoding
obj_col = []
for col in df.columns:
    if df[col].dtypes == object:
        obj_col.append(col)

label_encoder = LabelEncoder()
for col in obj_col:
    df[col] = label_encoder.fit_transform(df[col])

```

Performance Metric:

```

# k-means clustering
kmeans = KMeans(n_clusters=2, random_state=42)
kmeans_preds = kmeans.fit_predict(df_norm)
plot_km = df
plot_km['KMeans Cluster'] = kmeans_preds
plot_km=plot_km.sort_values(by='Cluster')
plot_km['KMeans Cluster'] = plot_km['Cluster'].astype(str).apply(lambda x: 'Outliers' if x == '-1' else x)

# Reverse normalization
# plot_km['Income'] = plot_km['Income'] * 100
# plot_km['Spend'] = plot_km['Spend'] * 1000

# Plot of clusters
temp = go.layout.Template()
temp.layout.plot_bgcolor = '#F9F9F9'
temp.layout.paper_bgcolor = '#F9F9F9'
temp.layout.legend.font.color = '#4D4D4D'
temp.layout.font.color = '#4D4D4D'
fig = px.scatter(plot_km, x="Income", y="Spend", color="KMeans Cluster",
                 color_discrete_sequence=px.colors.qualitative.T10[2:])
fig.update_traces(marker=dict(size=11, opacity=0.85, line=dict(width=1, color='#F7F7F7')))

```

```

fig.update_layout(template=temp, title="KMeans Cluster Profiles,<br>Customer Spending
vs. Income",
                  width=700, legend_title='Cluster',
                  paper_bgcolor='rgb(229, 236, 246)',
                  title_font_size=22,
                  xaxis=dict(title='Spending', showline=True, zeroline=False, range=[0, None]),
                  yaxis=dict(title='Income, $', ticksuffix='k', showline=True, range=[0, None]))

fig.show()

```

K-means Imputation:

```

# k-means clustering
kmeans = KMeans(n_clusters=2, random_state=42)
kmeans_preds = kmeans.fit_predict(df_norm)
plot_km = df
plot_km['KMeans Cluster'] = kmeans_preds
plot_km=plot_km.sort_values(by='Cluster')
plot_km['KMeans Cluster'] = plot_km['Cluster'].astype(str).apply(lambda x: 'Outliers' if x ==
'-1' else x)

# # Reverse normalization
# plot_km['Income'] = plot_km['Income'] * 100
# plot_km['Spend'] = plot_km['Spend'] * 1000

# Plot of clusters
temp = go.layout.Template()
temp.layout.plot_bgcolor = '#F9F9F9'
temp.layout.paper_bgcolor = '#F9F9F9'
temp.layout.legend.font.color = '#4D4D4D'
temp.layout.font.color = '#4D4D4D'
fig = px.scatter(plot_km, x="Income", y="Spend", color="KMeans Cluster",
                 color_discrete_sequence=px.colors.qualitative.T10[2:])
fig.update_traces(marker=dict(size=11, opacity=0.85, line=dict(width=1, color='#F7F7F7')))
fig.update_layout(template=temp, title="KMeans Cluster Profiles,<br>Customer Spending
vs. Income",

```

```

width=700, legend_title='Cluster',
paper_bgcolor='rgb(229, 236, 246)',
title_font_size=22,
xaxis=dict(title='Spending', showline=True, zeroline=False, range=[0, None]),
yaxis=dict(title='Income, $', ticksuffix='k', showline=True, range=[0, None])

```

```
fig.show()
```

GMM Imputation:

```

# Gaussian Mixture Model clustering
gmm = GaussianMixture(n_components=2, covariance_type='full', random_state=42)
gmm.fit(df_norm)
gmm_preds = gmm.predict(df_norm)
plot_gmm = df
plot_gmm['GMM Cluster'] = gmm_preds
plot_gmm = plot_gmm.sort_values(by='Cluster')
plot_gmm['GMM Cluster'] = plot_gmm['GMM Cluster'].astype(str).apply(lambda x:
'Outliers' if x == '-1' else x)

# Plot of clusters
temp = go.layout.Template()
temp.layout.plot_bgcolor = '#F9F9F9'
temp.layout.paper_bgcolor = '#F9F9F9'
temp.layout.legend.font.color = '#4D4D4D'
temp.layout.font.color = '#4D4D4D'
fig = px.scatter(plot_gmm, x="Income", y="Spend", color="Cluster",
                 color_discrete_sequence=px.colors.qualitative.T10[2:])
fig.update_traces(marker=dict(size=11, opacity=0.85, line=dict(width=1, color='#F7F7F7')))
fig.update_layout(template=temp, title="Gaussian Mixture Model Cluster
Profiles,<br>Customer Spending vs. Income",
                 width=700, legend_title='Cluster',
                 paper_bgcolor='rgb(229, 236, 246)',
                 title_font_size=22,
                 xaxis=dict(title='Spending', showline=True, zeroline=False),
                 yaxis=dict(title='Income, $', ticksuffix='k', showline=True))
fig.show()

```

DBSCAN Imputation:

```

# DB Scan clustering
db=DBSCAN(eps=0.3, min_samples=4, metric='euclidean')
db_preds=db.fit_predict(df_norm)
plot_db=df
plot_db['DB Cluster'] = db_preds

```

```

plot_db=plot_db.sort_values(by='Cluster')
plot_db['DB Cluster'] = plot_db['Cluster'].astype(str).apply(lambda x: 'Outliers' if x == '-1'
else x)

# Plot of clusters
temp = go.layout.Template()
temp.layout.plot_bgcolor = '#F9F9F9'
temp.layout.paper_bgcolor = '#F9F9F9'
temp.layout.legend.font.color = '#4D4D4D'
temp.layout.font.color = '#4D4D4D'
fig = px.scatter(plot_db, x="Income", y="Spend", color="Cluster",
                color_discrete_sequence=px.colors.qualitative.T10[2:])
fig.update_traces(marker=dict(size=11, opacity=0.85, line=dict(width=1, color='#F7F7F7')))
fig.update_layout(template=temp, title="DBSCAN Cluster Profiles,<br>Customer Spending
vs. Income",
                width=700, legend_title = 'Cluster',
                paper_bgcolor=rgb(229, 236, 246)',
                title_font_size=22,
                xaxis=dict(title='Spending',showline=True, zeroline=False),
                yaxis=dict(title='Income, $',ticksuffix='k',showline=True))
fig.show()

```

Comparison algorithms in 3D plot:

```

## Define color palette
colors = px.colors.qualitative.T10[2:]

# Initializing figure with 3 3D subplots
fig = make_subplots(rows=3, cols=1,
                    vertical_spacing=0.1,
                    specs=[[{'type': 'scatter3d'}],
                           [{'type': 'scatter3d'}],
                           [{'type': 'scatter3d'}]],
                    subplot_titles=("K-Means Clustering with 2 clusters",
                                   "Gaussian Mixture Model Clustering<br>with 2 clusters",
                                   "DBSCAN<br>with 2 clusters")
)

# Adding clusters to scatterplots
# Clean up the 'KMeans Cluster' column
plot_km['KMeans Cluster'] = plot_km['KMeans Cluster'].str.replace('Cluster ', "").astype(int)
plot_km=plot_km.sort_values(by='KMeans Cluster')
for i in range(0,3):
    marker_symbol = 'diamond'
    if i == 1:
        marker_symbol = 'circle'
    elif i == 0:

```

```

    marker_symbol = 'cross'
    fig.add_trace(go.Scatter3d(x = plot_km[plot_km['KMeans Cluster'] == i]['Spend'],
        y = plot_km[plot_km['KMeans Cluster'] == i]['Age'],
        z = plot_km[plot_km['KMeans Cluster'] == i]['Income'],
        mode = 'markers+lines', marker=dict(
            size=7,
            color=colors[i+1],
            line_width = 1,
            line_color='#F7F7F7',
            symbol=marker_symbol,
            opacity=0.9),
        name = str('Cluster '+str(i)), legendgroup = 1),
        row=1, col=1)

plot_gmm['GMM Cluster'] = plot_gmm['GMM Cluster'].astype(int)
plot_gmm = plot_gmm.sort_values(by='GMM Cluster')
for i in range(0, 2):
    marker_symbol = 'circle'
    if i == 1:
        marker_symbol = 'diamond'
    fig.add_trace(go.Scatter3d(x=plot_gmm[plot_gmm['GMM Cluster'] == i]['Spend'],
        y=plot_gmm[plot_gmm['GMM Cluster'] == i]['Age'],
        z=plot_gmm[plot_gmm['GMM Cluster'] == i]['Income'],
        mode='markers+lines', marker=dict(
            size=7,
            color=colors[i],
            line_width=1,
            line_color='#F7F7F7',
            symbol=marker_symbol,
            opacity=0.7),
        name=str('GMM Cluster ' + str(i+1)), legendgroup=2),
        row=2, col=1)

for i, j in enumerate(plot_db['DB Cluster'].unique()):
    marker_symbol = 'circle'
    if i == 1:
        marker_symbol = 'diamond'
    fig.add_trace(go.Scatter3d(x=plot_db[plot_db['DB Cluster'] == j]['Spend'],
        y=plot_db[plot_db['DB Cluster'] == j]['Age'],
        z=plot_db[plot_db['DB Cluster'] == j]['Income'],
        mode='markers+lines', marker=dict(
            size=7,
            color=colors[i+1],
            line_width=1,
            line_color='#F7F7F7',
            symbol=marker_symbol,
            opacity=0.8),
        name=str('DB Cluster ' + str(j)), legendgroup=3),
        row=3, col=1)

```

```

fig.update_traces(hovertemplate='Customer Spending Score:  % $\{x\}$ <br>Income:
$% $\{z\}$ <br>Age: % $\{y\}$ ')
fig.update_layout(title="Customer Segments based on Income, Spending, and Age",
  paper_bgcolor='rgb(229, 236, 246)',
  title_font_size=22,
  template=temp, height=1800, legend_tracegroupgap = 500,
  scene=dict(aspectmode='cube',
    xaxis = dict(title='Spending Score',
      backgroundcolor="#F3F3F3",
      gridcolor="white",
      showbackground=True,
      zerolinecolor="white"),
    yaxis = dict(title='Age, in years',
      backgroundcolor="#E4E4E4",
      gridcolor="white",
      showbackground=True,
      zerolinecolor="white"),
    zaxis = dict(title='Income, $',
      ticksuffix='k',
      backgroundcolor="#F6F6F6",
      gridcolor="white",
      showbackground=True,
      zerolinecolor="white")),
  scene2=dict(aspectmode='cube',
    xaxis = dict(title='Spending Score',
      backgroundcolor="#F3F3F3",
      gridcolor="white",
      showbackground=True,
      zerolinecolor="white"),
    yaxis = dict(title='Age, in years',
      backgroundcolor="#E4E4E4",
      gridcolor="white",
      showbackground=True,
      zerolinecolor="white"),
    zaxis = dict(title='Income, $',
      ticksuffix='k',
      backgroundcolor="#F6F6F6",
      gridcolor="white",
      showbackground=True,
      zerolinecolor="white")),
  scene3=dict(aspectmode='cube',
    xaxis = dict(title='Spending Score',
      backgroundcolor="#F3F3F3",
      gridcolor="white",
      showbackground=True,
      zerolinecolor="white"),
    yaxis = dict(title='Age, in years',
      backgroundcolor="#E4E4E4",

```



```

        gridcolor="white",
        showbackground=True,
        zerolinecolor="white"),
    zaxis = dict(title='Income, $',
        ticksuffix='k',
        backgroundcolor="#F6F6F6",
        gridcolor="white",
        showbackground=True,
        zerolinecolor="white"))
    )
fig.show()

```

RFM Imputation:

```

df['Frequency'] =
df['NumWebPurchases']+df['NumCatalogPurchases']+df['NumStorePurchases']

df['Monetary'] =
df['MntWines']+df['MntFruits']+df['MntMeatProducts']+df['MntFishProducts']+df['MntSweet
Products']+df['MntGoldProds']

#Data distribution
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
init_notebook_mode(connected=True)

#plotting rfm distribution before scaling
fig = px.box(pd.melt(df_rfm), x='variable', y='value',
    title='<b>RFM Data Distribution Before Scaling</b>',
    color='variable', color_discrete_sequence=colors_rfm,
    boxmode='overlay', points='all')

fig.update_layout(showlegend=False,paper_bgcolor='rgb(229, 236, 246)',
    title_font_size=22)

fig.show()

# RFM Mapping
# plotting rfm
fig = px.scatter_3d(df_rfm_scaled, x='Recency', y='Frequency', z='Monetary',
    title='<b>RFM Mapping</b>',
    opacity=0.5, color='Monetary', color_continuous_scale='electric')

fig.update_traces(marker=dict(size=5))

fig.update_layout(paper_bgcolor='rgb(229, 236, 246)', title_font_size=22)

fig.show()

```

Determine k:

```
#plotting elbow method
fig = px.line(x=K, y=inertias,
              title='<b>Optimal Number of Clusters by Elbow Method</b>',
              color_discrete_sequence=['rgb(5, 60, 94)'])

fig.update_traces(line_width=4)

fig.update_layout(paper_bgcolor='rgb(229, 236, 246)',title_font_size=22,
                  xaxis_title='k (No. of Clusters)', yaxis_title='Inertia')
fig.add_vline(x=4, line_width=3, line_dash='dash', line_color='rgb(183, 9, 76)')

fig.add_annotation(x=4.1, y=1800, text='<i>optimal k=4</i>', font_size=16,
                  showarrow=True, ax=60, ay=-30, arrowhead=2, arrowsize=1,
                  arrowwidth=2)

fig.add_shape(type='circle', xref='x', yref='y',
              x0=3.9, y0=1650, x1=4.1,y1=1914,
              line_color='rgb(183, 9, 76)')

fig.show()

#plotting silhouette score
fig = px.line(x=K, y=silhouette,
              title='<b>Optimal Number of Clusters by Silhouette Score</b>',
              color_discrete_sequence=['rgb(5, 60, 94)'])

fig.update_traces(line_width=4)

fig.update_layout(paper_bgcolor='rgb(229, 236, 246)',title_font_size=22,
                  xaxis_title='k (No. of Clusters)',
                  yaxis_title='Silhouette Score')

fig.add_vline(x=2, line_width=3, line_dash='dash', line_color='rgb(183, 9, 76)')

fig.add_annotation(x=2.1, y=0.44, text='<i>optimal k=2</i>', font_size=16,
                  showarrow=True, ax=60, ay=-30, arrowhead=2, arrowsize=1,
                  arrowwidth=2)

fig.add_shape(type='circle', xref='x', yref='y',
              x0=1.9, y0=0.432, x1=2.1,y1=0.442,
              line_color='rgb(183, 9, 76)')

fig.show()
```

RFM plot:

```
df["Clusters"] = pd.to_numeric(df["Clusters"])

fig = px.scatter_3d(df, x='Recency', y='Monetary', z='Frequency',
                   color='Clusters', category_orders=dict(Clusters=[0,1,2,3]),
                   title='<b>Cluster Results</b>', opacity=0.5,
                   color_continuous_scale=colors_cluster,
                   labels={'Clusters': 'Cluster'})

fig.update_traces(marker=dict(size=6, opacity=0.6))

fig.update_layout(showlegend=True, paper_bgcolor='rgb(229, 236, 246)',
                  title_font_size=22, legend_title='Cluster')

fig.show()

#plotting income & monetary by clusters
#one customer with income of 666666 is excluded because it's obscuring the pattern
fig = px.scatter(df[df['Income']<500000], x='Monetary', y='Income',
                 color='Clusters',
                 color_continuous_scale='electric',
                 title='<b>Clusters by Income & Monetary</b>',
                 opacity=0.5)

fig.update_traces(marker=dict(size=11, opacity=0.85, line=dict(width=1, color='#F7F7F7')),
                  text=df['Clusters'])

fig.update_layout(paper_bgcolor='rgb(229, 236, 246)',title_font_size=22,
                  legend_title='Clusters', xaxis_title='Monetary', yaxis_title='Income')

fig.show()
```

Cluster Analysis (kmeans, GMM, DBSCAN):

```
#Visualize all the feature based on spending
df["Marital_Status"] = df["Marital_Status"].astype("category")
df["Education"] = df["Education"].astype("category")

Personal = ["Age", "Education", "Marital_Status", "NumWebVisitsMonth", "Familysize"]

for i in Personal:
    fig = px.scatter(data_frame=df, x=i, y="Spend", color="Cluster", marginal_y="violin",
                    marginal_x="box",
                    color_discrete_sequence=px.colors.qualitative.Plotly)
    fig.update_layout(title=i.capitalize() + " vs Spending",
                      xaxis_title=i.capitalize(),
                      yaxis_title="Total Expense",
                      paper_bgcolor='rgb(229, 236, 246)',
                      plot_bgcolor='rgb(229, 236, 246)',
```

```

width=800,
height=600,
template="simple_white")
fig.show()

```

Cluster Analysis (RFM):

```

#Visualize all the feature based on spending
df["Marital_Status"] = df["Marital_Status"].astype("category")
df["Education"] = df["Education"].astype("category")

Personal = ["Age", "Income", "Education", "Marital_Status", "NumWebVisitsMonth",
"Familysize", 'Days_Enrolled']

for i in Personal:
    fig = px.scatter(data_frame=df, x=i, y="Spend", color="Clusters", marginal_y="violin",
marginal_x="box",
                    color_discrete_sequence=colors_cluster,
category_orders=dict(Clusters=[0,1,2,3]))
    fig.update_layout(title=i.capitalize() + " vs Spending",
                    xaxis_title=i.capitalize(),
                    paper_bgcolor='rgb(229, 236, 246)',
                    plot_bgcolor='rgb(229, 236, 246)',
                    template="simple_white")

fig.show()

```

GUI Source code:

```

import sys
import numpy as np
import pandas as pd
import matplotlib
import matplotlib.lines
import matplotlib.pyplot as plt
from matplotlib.figure import Figure
from matplotlib.backends.qt_compat import QtWidgets as QC
from matplotlib.backends.backend_qt5agg import FigureCanvasQTAgg
from matplotlib.backends.backend_qt5agg import NavigationToolbar2QT as
NavigationToolbar
from mpl_toolkits.mplot3d import Axes3D
from sklearn.cluster import MiniBatchKMeans
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import make_pipeline
from PyQt5.QtWidgets import QWidget, QLabel, QLineEdit, QPushButton, QGroupBox,
QFormLayout, QVBoxLayout, QGridLayout, QHBoxLayout, QSizePolicy, QTableWidgetItem,
QAbstractItemView, QTableWidgetItem, QDialog, QApplication, QMainWindow,
QMessageBox
from PyQt5.QtGui import QIcon, QPalette, QColor

```

```

from PyQt5.QtCore import QPropertyAnimation, QRect, Qt
from PyQt5.QtGui import QPainter, QBrush, QColor, QPen

data = pd.read_csv('marketing_campaign.csv',delimiter='\t')
numeric_cols = ['Year_Birth', 'Income', 'Kidhome', 'Teenhome', 'Recency', 'MntWines',
'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts', 'MntGoldProds',
'NumDealsPurchases', 'NumWebPurchases', 'NumCatalogPurchases', 'NumStorePurchases',
'NumWebVisitsMonth']
data_numeric = data[numeric_cols]
X = StandardScaler().fit_transform(data_numeric)

class CustomerSegmentation(QWidget):
    def __init__(self):
        super().__init__()
        self.data = pd.read_csv('marketing_campaign.csv',delimiter='\t')

        self.setWindowTitle('Customer Segmentation')
        self.setGeometry(100, 100, 800, 600)

        header = QLabel('RFM')
        header.setStyleSheet('font-size: 36px; font-weight: bold; color: #fff;')
        subheader = QLabel('Customer Segmentation')
        subheader.setStyleSheet('font-size: 18px; color: #666; margin-bottom: 20px;')

        layout = QGridLayout()
        layout.setContentsMargins(20, 20, 20, 20)
        layout.setSpacing(10)

        layout.addWidget(header, 0, 0, 1, 2, Qt.AlignCenter)
        layout.addWidget(subheader, 1, 0, 1, 2, Qt.AlignCenter)

        recency_label = QLabel("Recency:")
        recency_icon = QIcon('calendar.png')
        self.recency_input = QLineEdit()
        self.recency_input.setPlaceholderText("Days Since Last Purchase")
        self.recency_input.setStyleSheet('font-size: 14px; color: #666;')
        recency_label.setPixmap(recency_icon.pixmap(24, 24))
        layout.addWidget(recency_label, 2, 0)
        layout.addWidget(self.recency_input, 2, 1)

        frequency_label = QLabel("Frequency:")
        frequency_icon = QIcon('grocery-cart.png')
        self.frequency_input = QLineEdit()
        self.frequency_input.setPlaceholderText("Number of Purchases")
        self.frequency_input.setStyleSheet('font-size: 14px; color: #666;')
        frequency_label.setPixmap(frequency_icon.pixmap(24, 24))
        layout.addWidget(frequency_label, 3, 0)
        layout.addWidget(self.frequency_input, 3, 1)

```

```

monetary_label = QLabel("Monetary:")
monetary_icon = QIcon('investment.png')
self.monetary_input = QLineEdit()
self.monetary_input.setPlaceholderText("Total Amount Spent")
self.monetary_input.setStyleSheet('font-size: 14px; color: #666;')
monetary_label.setPixmap(monetary_icon.pixmap(24, 24))
layout.addWidget(monetary_label, 4, 0)
layout.addWidget(self.monetary_input, 4, 1)

button_layout = QHBoxLayout()

self.add_button = QPushButton('Add')
self.add_button.setIcon(QIcon('add-user.png'))
self.add_button.setToolTip('Click to add a new customer')
self.add_button.setStyleSheet('font-size: 14px; color: #fff; background-color: #3b9cff;
border: none; padding: 10px 20px;')
self.add_button.setFixedSize(120, 50)
self.add_button.clicked.connect(self.add_customer)
button_layout.addWidget(self.add_button)

self.del_button = QPushButton('Del')
self.del_button.setIcon(QIcon('delete-user.png'))
self.del_button.setToolTip('Click to delete the selected customer')
self.del_button.setStyleSheet('font-size: 14px; color: #fff; background-color: #3b9cff;
border: none; padding: 10px 20px;')
self.del_button.setFixedSize(120, 50)
self.del_button.clicked.connect(self.del_customer)
button_layout.addWidget(self.del_button)

self.clear_button = QPushButton('Clear')
self.clear_button.setIcon(QIcon('clear-format.png'))
self.clear_button.setToolTip('Click to clear the input fields')
self.clear_button.setStyleSheet('font-size: 14px; color: #fff; background-color: #3b9cff;
border: none; padding: 10px 20px;')
self.clear_button.setFixedSize(120, 50)
self.clear_button.clicked.connect(self.clear_inputs)
button_layout.addWidget(self.clear_button)

self.predict_button = QPushButton('Predict')
self.predict_button.setIcon(QIcon('play.png'))
self.predict_button.setToolTip('Click to predict the customer segment')
self.predict_button.setStyleSheet('font-size: 14px; color: #fff; background-color:
#3b9cff; border: none; padding: 10px 20px;')
self.predict_button.setFixedSize(120, 50)
self.predict_button.clicked.connect(self.predict)
button_layout.addWidget(self.predict_button)

self.plot_button = QPushButton('3D Plot')
self.plot_button.setIcon(QIcon('3d-modeling.png'))

```

```

self.plot_button.setToolTip('Click to plot the customer segments in 3D')
self.plot_button.setStyleSheet('font-size: 14px; color: #fff; background-color: #3b9cff;
border: none; padding: 10px 20px;')
self.plot_button.setFixedSize(120, 50)
self.plot_button.clicked.connect(self.show_plot)
button_layout.addWidget(self.plot_button)

layout.addLayout(button_layout, 5, 0, 1, 2, Qt.AlignCenter)

table_button_layout = QHBoxLayout()

table_button_layout.addLayout(button_layout)

table_button_layout = QHBoxLayout()
self.table = QTableWidgetItem()
columns_to_drop = ['Year_Birth', 'Education', 'Marital_Status', 'Income', 'Kidhome',
'Teenhome', 'Dt_Customer', 'MntWines',
'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts', 'MntGoldProds',
'NumWebPurchases', 'NumCatalogPurchases', 'NumStorePurchases', 'NumDealsPurchases',
'NumWebVisitsMonth', 'AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3',
'AcceptedCmp4', 'AcceptedCmp5', 'Complain', 'Response', 'Z_CostContact', 'Z_Revenue']

self.data.drop(columns=columns_to_drop, inplace=True)

self.table.setRowCount(len(self.data))
self.table.setColumnCount(len(self.data.columns))

# Set the header labels of the table to match the column names of your data
self.table.setHorizontalHeaderLabels(list(self.data.columns))

self.table.setSelectionBehavior(QAbstractItemView.SelectRows)
self.table.setSelectionMode(QAbstractItemView.SingleSelection)
self.table.setEditTriggers(QAbstractItemView.NoEditTriggers)
self.table.cellClicked.connect(self.select_customer)
self.table.setStyleSheet('font-size: 14px;')
self.table.setFixedSize(600, 400)

table_button_layout.addWidget(self.table)

for i in range(len(self.data)):
    row_items = []
    for col_name in ['ID', 'Recency', 'Frequency', 'Monetary']:
        col_index = self.data.columns.get_loc(col_name)
        row_items.append(int(float(self.data.iloc[i, col_index])))

    self.table.insertRow(i)
    for j, item in enumerate(row_items):
        self.table.setItem(i, j, QTableWidgetItem(str(item)))

```

```

layout.addWidget(self.table, 6, 0, 1, 2, Qt.AlignCenter)
table_button_layout.addWidget(self.table)

result_layout = QHBoxLayout()
self.result_label = QLabel("Result:")
self.result_label.setStyleSheet('font-size: 14px; color: #fff;')
result_layout.addWidget(self.result_label)
layout.addLayout(result_layout, 8, 0, 1, 2)
self.customer = np.array([[0, 0, 0]])
self.setLayout(layout)
recency_label.setBuddy(self.recency_input)
frequency_label.setBuddy(self.frequency_input)
monetary_label.setBuddy(self.monetary_input)

app.setStyle('Fusion')
palette = QPalette()
palette.setColor(QPalette.Window, QColor(53, 53, 53))
palette.setColor(QPalette.WindowText, Qt.white)
palette.setColor(QPalette.Base, QColor(25, 25, 25))
palette.setColor(QPalette.AlternateBase, QColor(53, 53, 53))
palette.setColor(QPalette.ToolTipBase, Qt.white)
palette.setColor(QPalette.ToolTipText, Qt.white)
palette.setColor(QPalette.Text, Qt.white)
palette.setColor(QPalette.Button, QColor(53, 53, 53))
palette.setColor(QPalette.ButtonText, Qt.white)
palette.setColor(QPalette.BrightText, Qt.red)
palette.setColor(QPalette.Link, QColor(42, 130, 218))
palette.setColor(QPalette.Highlight, QColor(42, 130, 218))
palette.setColor(QPalette.HighlightedText, Qt.black)
app.setPalette(palette)

def add_customer(self):
    try:
        recency = int(self.recency_input.text())
        frequency = int(self.frequency_input.text())
        monetary = int(self.monetary_input.text())
    except ValueError:
        QMessageBox.warning(self, 'Invalid Input', 'Please enter valid numeric values for
recency, frequency, and monetary.')
        return

    id = int(self.data['ID'].max()) + 1

    new_customer = pd.DataFrame({
        'ID': [id],
        'Recency': [recency],
        'Frequency': [frequency],
        'Monetary': [monetary]
    })

```



```

row_num = 0
for i in range(len(self.data)):
    if self.data.loc[i, 'ID'] > id:
        break
    row_num += 1

self.table.insertRow(row_num)
for i, value in enumerate([id, recency, frequency, monetary]):
    self.table.setItem(row_num, i, QTableWidgetItem(str(value)))
self.data = pd.concat([self.data.iloc[:row_num],
self.data.iloc[row_num:]], ignore_index=True)

    new_customer,

    QMessageBox.information(self, 'Success', 'New customer added to the dataset.')
    self.recency_input.clear()
    self.frequency_input.clear()
    self.monetary_input.clear()

def del_customer(self):
    # Get the selected row index
    row_index = self.table.currentRow()
    if row_index >= 0:
        self.table.removeRow(row_index)
    else:
        QMessageBox.warning(self, 'Error', 'No customer selected.')

def select_customer(self, row, col):
    recency = self.table.item(row, 1).text()
    frequency = self.table.item(row, 2).text()
    monetary = self.table.item(row, 3).text()
    self.recency_input.setText(str(recency))
    self.frequency_input.setText(str(frequency))
    self.monetary_input.setText(str(monetary))

def predict(self):
    data['Frequency'] = data['NumWebPurchases'] + data['NumCatalogPurchases'] +
data['NumStorePurchases']
    data['Monetary'] = data['MntWines'] + data['MntFruits'] + data['MntMeatProducts'] +
data['MntFishProducts'] + data['MntSweetProducts'] + data['MntGoldProds']

    recency_str = self.recency_input.text().strip()
    frequency_str = self.frequency_input.text().strip()
    monetary_str = self.monetary_input.text().strip()

    if not recency_str or not frequency_str or not monetary_str:
        self.result_label.setText("Please provide values for all inputs")
        return

try:

```

```

    recency = int(float(recency_str))
    frequency = int(float(frequency_str))
    monetary = int(float(monetary_str))
except ValueError:
    self.result_label.setText("Please provide numeric values for inputs")
    return

X = data[['Recency', 'Frequency', 'Monetary']]
self.kmeans = MiniBatchKMeans(n_clusters=4, batch_size=100, random_state=42)
self.kmeans.partial_fit(X)
self.customer = np.array([[recency, frequency, monetary]])
cluster_label = self.kmeans.predict(self.customer.reshape(1, -1))[0]

customer_segments = {
    0: "Low-Spending Active Customers",
    1: "Mid-Spending Active Customers",
    2: "High-Spending Active Customers",
    3: "Churned or Inactive Customers"
}
segment = customer_segments.get(cluster_label)
self.result_label.setText("Cluster {}: {}".format(cluster_label, segment))

def plot(self):
    X = data[['Recency', 'Frequency', 'Monetary']]
    kmeans = MiniBatchKMeans(n_clusters=4, batch_size=100, random_state=42)
    kmeans.partial_fit(X)
    fig = Figure()
    ax = fig.add_subplot(111, projection='3d')
    colors = ['#3C1642', '#086375', '#1DD3B0', '#AFFC41']
    legend_elements = [
        matplotlib.lines.Line2D([0], [0], marker='o', color='w', label='Low-Spending Active',
markerfacecolor=colors[0], markersize=10),
        matplotlib.lines.Line2D([0], [0], marker='o', color='w', label='Mid-Spending Active',
markerfacecolor=colors[1], markersize=10),
        matplotlib.lines.Line2D([0], [0], marker='o', color='w', label='High-Spending Active',
markerfacecolor=colors[2], markersize=10),
        matplotlib.lines.Line2D([0], [0], marker='o', color='w', label='Churned or Inactive',
markerfacecolor=colors[3], markersize=10),
    ]
    ax.legend(handles=legend_elements, loc='upper right')

    for i in range(4):
        cluster_points = X[kmeans.labels_ == i]
        ax.scatter(cluster_points['Recency'], cluster_points['Frequency'],
cluster_points['Monetary'],
c=colors[i], label=f'Cluster {i}', s=50, alpha=0.5)

    ax.scatter(self.customer[:, 0], self.customer[:, 1], self.customer[:, 2], c='red', marker='x',
s=100, linewidths=2, alpha=0.8)

```

```

ax.set_xlabel('Recency')
ax.set_ylabel('Frequency')
ax.set_zlabel('Monetary')
canvas = FigureCanvasQTAgg(fig)
toolbar = NavigationToolbar(canvas, self, coordinates=True)
plot_window = QMainWindow()
plot_window.setWindowTitle('Customer Segmentation Plot')
plot_window_layout = QVBoxLayout()
plot_window_layout.addWidget(canvas)
plot_window_layout.addWidget(toolbar)
central_widget = QWidget()
central_widget.setLayout(plot_window_layout)
plot_window.setCentralWidget(central_widget)
plot_window.setGeometry(100, 100, 800, 600)
self.plot_window = plot_window

def show_plot(self):
    self.plot()
    self.plot_window.show()

def clear_inputs(self):
    self.recency_input.clear()
    self.frequency_input.clear()
    self.monetary_input.clear()

if __name__ == '__main__':
    app = QApplication(sys.argv)
    segmentation = CustomerSegmentation()
    segmentation.show()
    sys.exit(app.exec ())

```

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.: 1
Student Name & ID: Toh Wei Xuan 19ACB03568	
Supervisor: Dr Lim Jia QI	
Project Title: Customer Segmentation on Clustering Algorithms	

1. WORK DONE

Check for the previous work.

2. WORK TO BE DONE

Copy and paste chapters 1 and 2 to new fyp2 template.

3. PROBLEMS ENCOUNTERED

Previous source code got some bugs and error

4. SELF EVALUATION OF THE PROGRESS

Not bad Progress done until the visualization



Supervisor's signature



Student's signature

Trimester, Year: Y3S3	Study week no.: 2
Student Name & ID: Toh Wei Xuan 19ACB03568	
Supervisor: Dr Lim Jia QI	
Project Title: Customer Segmentation on Clustering Algorithms	

<p>1. WORK DONE Learnt how to Determine the k value.</p>
<p>2. WORK TO BE DONE Encounter all the clustering algorithms.</p>
<p>3. PROBLEMS ENCOUNTERED Various methods to find the optimal k.</p>
<p>4. SELF EVALUATION OF THE PROGRESS Still fine in start Clustering implementation</p>



Supervisor's signature



Student's signature

Trimester, Year: Y3S3	Study week no.: 3
Student Name & ID: Toh Wei Xuan 19ACB03568	
Supervisor: Dr Lim Jia QI	
Project Title: Customer Segmentation on Clustering Algorithms	

1. WORK DONE

Optimal $k = 4$
 Implement clustering algorithms k-means.

2. WORK TO BE DONE

Implement GMM and DBSCAN

3. PROBLEMS ENCOUNTERED

Different k value encountered 2 and 4 when used different metrics.

4. SELF EVALUATION OF THE PROGRESS

Still fine



 Supervisor's signature



 Student's signature

Trimester, Year: Y3S3	Study week no.: 4
Student Name & ID: Toh Wei Xuan 19ACB03568	
Supervisor: Dr Lim Jia QI	
Project Title: Customer Segmentation on Clustering Algorithms	

<p>1. WORK DONE Implemented k-means, GMM and DBSCAN</p>
<p>2. WORK TO BE DONE Performs summary comparison of performance metrics to determine k</p>
<p>3. PROBLEMS ENCOUNTERED Epsilon of DBSCAN is difficult to determine.</p>
<p>4. SELF EVALUATION OF THE PROGRESS Still fine</p>



Supervisor's signature



Student's signature

Trimester, Year: Y3S3	Study week no.: 5
Student Name & ID: Toh Wei Xuan 19ACB03568	
Supervisor: Dr Lim Jia QI	
Project Title: Customer Segmentation on Clustering Algorithms	

1. WORK DONE

Pivot table of summary metrics was implemented, optimal k was 2 and perform the three clustering algorithms except DBSCAN.

2. WORK TO BE DONE

Use GridSearchCV() to determine the epsilon of DBSCAN

3. PROBLEMS ENCOUNTERED

Hyperparameter of DBSCAN difficult to determine.

4. SELF EVALUATION OF THE PROGRESS

Mad on the progress since is stucked



Supervisor's signature



Student's signature

Trimester, Year: Y3S3	Study week no.: 6
Student Name & ID: Toh Wei Xuan 19ACB03568	
Supervisor: Dr Lim Jia QI	
Project Title: Customer Segmentation on Clustering Algorithms	

1. WORK DONE

Three algorithms, k-means, GMM and DBSCAN successfully implemented and plot the result in 2D and 3D form. The comparison between the clustering algorithms.

2. WORK TO BE DONE

learn new approach/ algorithm, RFM.

3. PROBLEMS ENCOUNTERED

Difficult to categorize the group of customers if two cluster only

4. SELF EVALUATION OF THE PROGRESS

Still fine



Supervisor's signature



Student's signature

Trimester, Year: Y3S3	Study week no.: 7
Student Name & ID: Toh Wei Xuan 19ACB03568	
Supervisor: Dr Lim Jia QI	
Project Title: Customer Segmentation on Clustering Algorithms	

<p>1. WORK DONE Distribution of RFM, RFM modelling and plotting</p>
<p>2. WORK TO BE DONE Cluster Analysis to categorize the clusters groups.</p>
<p>3. PROBLEMS ENCOUNTERED Got two k values, 2 and 4 and confirm to use k=4 due to its was behavioral based clustering and easier to identify.</p>
<p>4. SELF EVALUATION OF THE PROGRESS Still ok</p>



Supervisor's signature



Student's signature

Trimester, Year: Y3S3	Study week no.: 8
Student Name & ID: Toh Wei Xuan 19ACB03568	
Supervisor: Dr Lim Jia QI	
Project Title: Customer Segmentation on Clustering Algorithms	

<p>1. WORK DONE Cluster Analysis and RFM modelling and categorizing.</p>
<p>2. WORK TO BE DONE GUI development using Incremental k-means.</p>
<p>3. PROBLEMS ENCOUNTERED Naming of the categorized customer groups.</p>
<p>4. SELF EVALUATION OF THE PROGRESS Still fine.</p>



Supervisor's signature



Student's signature

Trimester, Year: Y3S3	Study week no.: 9
Student Name & ID: Toh Wei Xuan 19ACB03568	
Supervisor: Dr Lim Jia QI	
Project Title: Customer Segmentation on Clustering Algorithms	

1. WORK DONE

GUI using MiniBatch K-means was developed and got the function of input integer value recency, frequency and monetary.

2. WORK TO BE DONE

Add CRUD function on the GUI.

3. PROBLEMS ENCOUNTERED

Incremental k-means module cannot implement due to the scikit learn version problem of my pc and I replaced it to use MiniBatch k-means.

4. SELF EVALUATION OF THE PROGRESS

Still fine



Supervisor's signature



Student's signature

Trimester, Year: Y3S3	Study week no.: 10
Student Name & ID: Toh Wei Xuan 19ACB03568	
Supervisor: Dr Lim Jia Qi	
Project Title: Customer Segmentation on Clustering Algorithms	

1. WORK DONE

User friendly GUI developed.

2. WORK TO BE DONE

Report writing from chapter 3.

3. PROBLEMS ENCOUNTERED

Report organization not sure

4. SELF EVALUATION OF THE PROGRESS

Glad due to almost done



Supervisor's signature

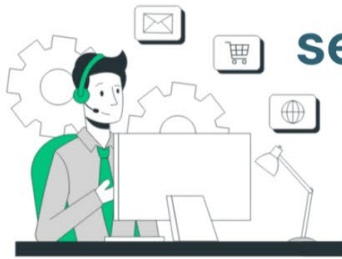


Student's signature

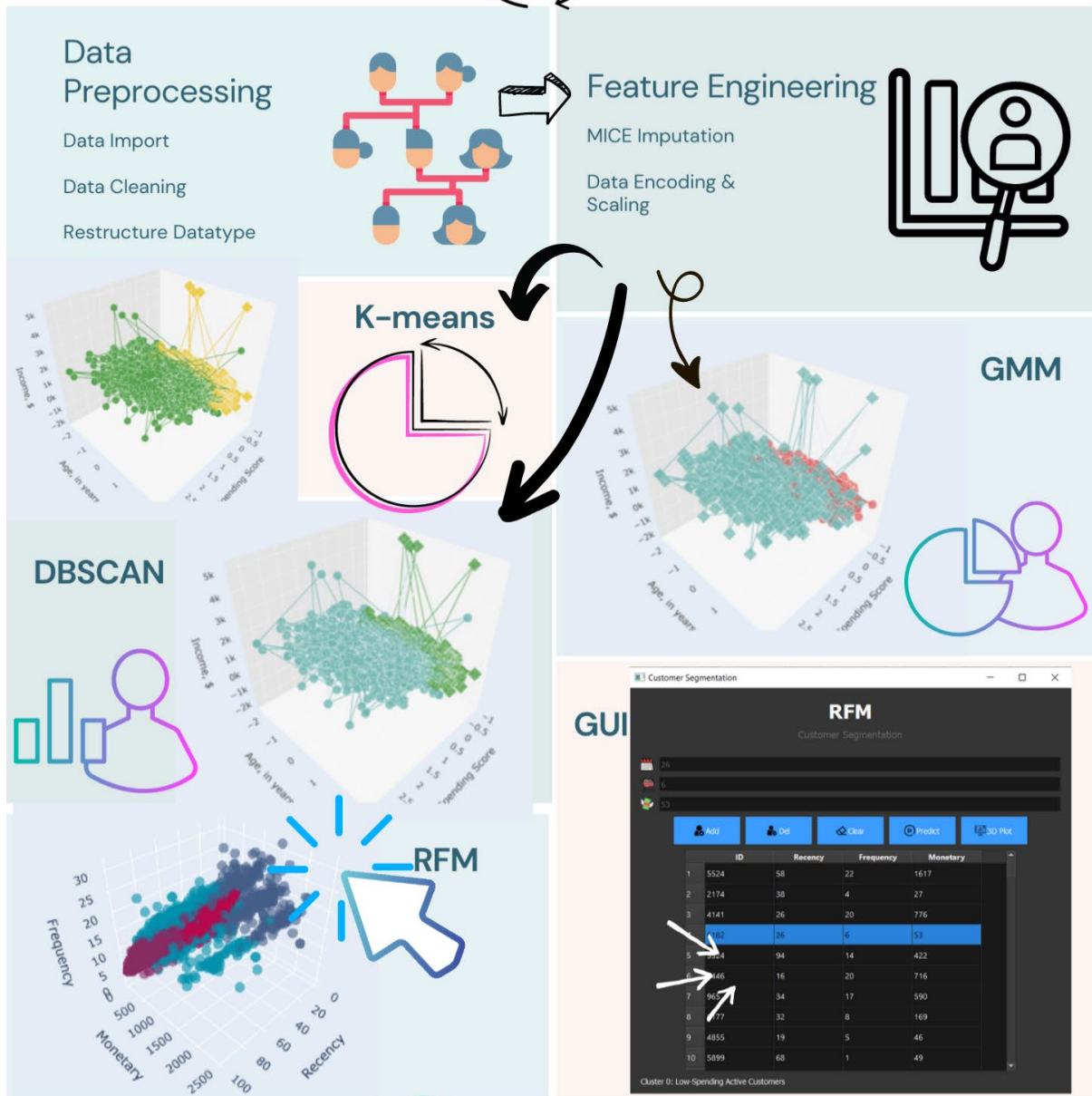
POSTER



Customer segmentation



Unsupervised Clustering Algorithms



PLAGIARISM CHECK RESULT

FYP2

ORIGINALITY REPORT

16%	8%	6%	12%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Middle East College of Information Technology Student Paper	1%
2	medium.com Internet Source	1%
3	Submitted to University of Liverpool Student Paper	1%
4	Submitted to Monash University Student Paper	1%
5	www.clusteranalysis4marketing.com Internet Source	1%
6	Submitted to University of Reading Student Paper	<1%
7	Partitional Clustering Algorithms, 2015. Publication	<1%
8	www.handlebar-online.com Internet Source	<1%
9	etda.libraries.psu.edu Internet Source	<1%

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1 of 1



FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	TOH WEI XUAN
ID Number(s)	19ACB03568
Programme / Course	CS
Title of Final Year Project	CUSTOMER SEGMENTATION ON CLUSTERING ALGORITHMS

Similarity	Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)
Overall similarity index: <u>16</u> % Similarity by source Internet Sources: <u>8</u> % Publications: <u>6</u> % Student Papers: <u>12</u> %	
Number of individual sources listed of more than 3% similarity: <u>0</u>	
Parameters of originality required and limits approved by UTAR are as Follows: (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor

Name: Lim Jia Qi

Date: 28/04/2023

Signature of Co-Supervisor

Name:

Date:



UNIVERSITI TUNKU ABDUL RAHMAN

**FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY
(KAMPAR CAMPUS)**

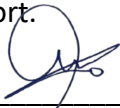
CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	19ACB03568
Student Name	TOH WEI XUAN
Supervisor Name	DR. LIM JIA QI

TICK (√)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
	Front Plastic Cover (for hardcopy)
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
√	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.



(TOH WEI XUAN)

Date: 25/4/2023