POLITECNICO DI TORINO

Corso di Laurea Magistrale Ingegneria Gestionale

Master's Degree Thesis

Comparative Study of Customer Segmentation Strategies Based on Business Analytics

Supervisors

Prof. Eliana PASTOR

Prof. Vicenc FERNANDEZ ALARCON

Candidate

Davide AIMAR

2024-2025

Summary

This thesis explores a comparative analysis of customer segmentation strategies supported by advanced analytical methodologies. It focuses on two foundational frameworks: **Recency, Frequency, Monetary (RFM)** and **Customer Lifetime Value (CLV)**, which respectively capture short-term transactional behaviors and long-term economic contributions. These metrics are subsequently analyzed through five clustering algorithms: *K-Means, Hierarchical Clustering, DBSCAN, Gaussian Mixture Models (GMM), and Fuzzy C-Means.*

The study utilizes the UK E-Commerce data set from the UCI repository, which undergoes meticulous preprocessing and normalization to ensure robust and consistent input for the clustering models. The evaluation framework leverages two internal validation metrics—**Silhouette Score** and **Calinski–Harabasz Index**—to provide complementary perspectives on local density separation and global variance partitioning.

Experimental results reveal that **DBSCAN** consistently outperforms other methods in identifying dense microclusters, often representing high-value or niche customers. In contrast, **K-Means** and **Hierarchical Clustering** exhibit stronger performance in generating broader global partitions. While **Fuzzy C-Means** achieves moderate results by accommodating overlapping segment boundaries through soft membership, **GMM** struggles with the non-Gaussian characteristics of the RFM and CLV datasets.

The findings underscore that no single approach universally outperforms the others. Instead, the selection of metrics and clustering algorithms should be strategically aligned with business goals, such as identifying anomalies or performing large-scale segmentation. This study provides actionable insights for businesses aiming to enhance marketing strategies, optimize resource allocation, and strengthen **Customer Relationship Management (CRM)** through data-driven segmentation approaches.

Table of Contents

List of Tables

List of Figures

List of abbreviations / Glossary

RFM (Recency, Frequency, Monetary): A framework for segmenting customers based on how recently they purchased (Recency), how frequently they buy (Frequency), and how much they spend (Monetary), providing a score for each dimension.

CLV (Customer Lifetime Value): A metric estimating the total revenue a customer is expected to generate throughout their relationship with a business.

Elbow Method: A graphical approach to determine the optimal number of clusters by identifying the point where additional clusters yield minimal improvement in variance reduction.

Silhouette Score: Evaluates clustering quality by measuring how similar a data point is to others in its cluster versus points in other clusters; scores range from -1 (poor) to 1 (excellent).

CH Index: A metric that compares between-cluster dispersion with within-cluster dispersion; higher values indicate well-defined and distinct clusters.

DBSCAN: A clustering algorithm that identifies clusters as dense areas of data points, detecting arbitrarily shaped clusters and outliers based on neighborhood radius (`eps`) and minimum points (`MinPts`).

K-Means: A centroid-based clustering algorithm that partitions data into k predefined clusters by minimizing variance within clusters, assuming spherical cluster shapes.

Hierarchical Clustering: A method that creates nested clusters visualized in a dendrogram, allowing flexibility in choosing the number of clusters without prior specification.

Noise Points (DBSCAN): Points that DBSCAN labels as outliers due to insufficient density in their neighborhood, often representing anomalies.

k-NN Distance Plot: A diagnostic tool to identify the optimal `eps` parameter for DBSCAN, using a sharp increase in k-nearest neighbor distances as the threshold.

Gaussian Mixture Model (GMM): A probabilistic clustering technique that assumes data points come from a mixture of Gaussian distributions, providing flexibility in cluster shapes and sizes.

Fuzzy C-Means (FCM): A clustering approach that assigns data points to multiple clusters with degrees of membership, ideal for identifying overlapping or transitional customer segments.

Recency Mean (R Mean): The average number of days since the last purchase among customers in a cluster, reflecting their recent engagement level.

Frequency Mean (F Mean): The average number of transactions made by customers in a cluster, indicating their purchase frequency.

Monetary Mean (M Mean): The average total revenue generated by customers in a cluster, representing their financial contribution.

Average Transaction Value: The average monetary value of a customer's transactions, calculated as TotalRevenue divided by Frequency, indicating spending patterns.

Lifespan: The time span of a customer's active relationship with a business, measured from their first to their most recent transaction.

Hard Assignments: Assigns each data point exclusively to a single cluster, as used in algorithms like K-Means and DBSCAN.

Soft Assignments: Allows data points to have varying degrees of membership in multiple clusters, as implemented in FCM and similar methods.

Centroid-Based Clustering: Partitions data by optimizing the placement of centroids to minimize variance within clusters, commonly used in K-Means.

Density-Based Clustering: Identifies clusters as dense regions in the data space and separates them from sparser areas, as seen in DBSCAN, which can also detect outliers.

BIC (Bayesian Information Criterion): A statistical tool to select the best number of clusters in probabilistic models like GMM, balancing model fit with complexity.

Customer Segmentation: Divides a customer base into smaller, behaviorally or demographically similar groups for tailored marketing and improved resource allocation.

Outliers: Data points that deviate significantly from the majority, often representing anomalies, errors, or unique high-value customers.

Dendrogram: A hierarchical, tree-like diagram visualizing nested clusters, helping determine the most appropriate number of clusters for analysis.

Chapter 1

Introduction

In today's busy market, companies trying to build steady growth increasingly depend on **customer segmentation** to improve their position. *Segmentation* splits a customer base into smaller groups with similar behavior or economic value. By targeting these groups exactly, organizations can improve *marketing efficiency*, *resource allocation*, along with *customer loyalty*.

This thesis explains the complex idea of segmentation using two main analysis tools: the **RFM** (*Recency, Frequency, Monetary*) model and **CLV** (*Customer Lifetime Value*). *RFM* looks at recent buying habits—*when* a purchase happens, *how often* a customer buys, as well as *how much* each purchase earns. *CLV* adds time, examining a customer's profit over a long period. Both views matter, work well together, along with give a full idea of *customer relationships*.

To use these views well, the study tests five main clustering methods—**K-Means**, **Hierarchical Clustering**, **DBSCAN**, **Gaussian Mixture Models (GMM)**, and **Fuzzy C-Means**—each method has its own basic theory and useful features. The main goal is to *compare and judge* how well these methods split customers using both **RFM** and **CLV** datasets. For this purpose, careful *data cleaning* and *scaling* keep the data accurate; two measures, the **Silhouette Score** and the **Calinski–Harabasz Index**, check quality locally; they also check it globally.

This thesis focuses on how *useful* the segments are: how fast an algorithm with a metric framework can guide tactical or strategic choices, such as finding *best customers*, spotting *inactive buyers*, or setting *marketing budgets*. The work adds to *academic research* on data-based segmentation and offers simple advice for companies that seek to improve their *marketing mix* or *customer management*.

Chapter 2

Literature Review

Wendell R. Smith's groundbreaking paper *"Product Differentiation and Market Segmentation as Alternative Marketing Strategies"* in the Journal of Marketing in 1956, [1] delineated the history of customer segmentation. Smith was an advocate of the idea of consumer heterogeneity and the practice of changing customer groups conformed with certain common characteristics. His view highlighted the role of this kind of strategic alignment of products and services with specific market segments. Today is considered the precursors of the modern marketing paradigm. In the years that followed,segmentation was adopted as an automatic part of **Customer Relationship Management (CRM)** processes and turned into a basic resource for targeted marketing, allocation of resources, and financial gain through the retention of loyal customers. Following studies on customer segmentation such as Thorsten Teichert's study *"Customer Segmentation Revisited: The Case of the Airline Industry"* demonstrated the segmentation implementation even in the most dynamic sectors while pointing it out as the importance in different business environments. [2]

Segmentation is not only a theoretical concept but has actual impacts. A prime example is the airline industry where it is possible to segment firms into business travelers who are frequently flying and occasional leisure passengers. The result of this differentiation is the effect on the loyalty program, airfare structure, or the promotional campaign [2]. By researching their clients, businesses are able to correctly target their actions to increase customer satisfaction and revenue.

2.1 RFM as a Baseline Segmentation Tool

The **RFM** model is a fundamental instrument often employed for the purpose of customer segmentation. The model assesses customer behavior by analyzing the time passed since the last purchase of the customer (Recency), the frequency of purchases (Frequency) and the monetary value of the transaction (Monetary). With these factors under consideration, businesses can divide customers into categories such as high-frequency, loyal buyers like or rare, low-value buyers [3]. The research paper written by K. H. Chung and M. Chen titled "RFM Analysis: A Balancing Act Between Business Intelligence and Marketing Intelligence" is the proof of the fact

that the model is simple, yet very effective [3].

The RFM model, despite its simplicity and flexibility, is mainly characterized by marked strengths. It offers a fast perspective on customer value with minor calculations and is thereby, preferable for firms that are technically lacking. The most significant aspect of this framework is the fact that it is static: the instance is not the one capturing customer behavior changes of time, by way of example, declining interest or acquiring patterns. This predicate is, of course, dynamic so it cannot be static: CLV analysis is no exception here. RFM and CLV are however different in that the former only keeps track of the current behavior of the customer while the latter can do that plus making a projection of the future behavior of the customer as well thus giving businesses the knowledge and time to come up with long term plans.

RFM is a frequently employed tool for customer analysis, but it is not necessarily the best method to show a customer's full economic impact. For instance, a customer who buys items moderately in a short time frame might present a different trend if looked over a longer duration. RFM does not take into account profit margins which have an effect on purchasing trends. Addressing these problems, researchers including Mahboubeh Khajvand et al. in their paper "Estimating Customer Lifetime Value Based on RFM Analysis of Customer Purchase Behavior: Case Study" corrected it by combining RFM with advanced metrics like **Customer Lifetime Value (CLV)** [4].

2.2 Incorporating Customer Lifetime Value (CLV)

CLV measures the total value a customer is expected to deliver to a business during their relationship. Compared to RFM, which looks at past data, CLV includes profit estimates as well as retention rates to give a view of the future [5]. The paper "Valuing Customers" by S. Gupta et al. shows that CLV helps plan by predicting long term profit [5].

CLV shows customer value by joining behavior, money or time factors. By calculating today's worth of future cash flows per customer, businesses can support high value customers while using resources well. This approach makes marketing reactive as well as predictive to meet long term goals.

Khajvand et al. showed in their study that combining CLV weights with RFM scores produces segments that identify customers who bring high value over time, even if recent activity is average [4]. J. Villanueva and D. M. Hanssens in "Customer Equity: Measurement, Management next to Research Opportunities" noted that adding social influence data in CLV models helps because customers with strong referrals lift growth [6]. Lemmens and Croux in "Bagging and Boosting Classification Trees to Predict Churn" point out that tools such as regression or decision trees help CLV predict future behavior better [7].

2.3 Clustering Algorithms for Segmentation

While metrics like RFM and CLV define the *dimensions* of segmentation, clustering algorithms group customers based on these metrics. Over the past decades, **unsupervised learning** has become a practical solution for revealing hidden patterns in data without pre-labeled categories.

2.3.1 K-Means Clustering

K-Means is by far the most popular used clustering algorithms. It consist in dividing a dataset into smaller clusters by minimizing the intra-cluster variance. M. K. Pakhira et al.'s paper "Validity Index for Crisp and Fuzzy Clusters" demonstrates its computational efficiency and applicability to large datasets [8]. However, K-Means assumes spherical cluster shapes and requires a predefined number of clusters (k), which may lead to suboptimal performance if the true data structure is non-spherical or unknown [9].

D. T. Pham et al. in "Selection of K in K-Means Clustering" introduced methods for determine the optimal number of clusters, trying to mitigate one of K-Means' significant limitations [9]. Nevertheless the simplicity of the algorithm still makes it ideal for initial exploratory analysis in segmentation studies.

K-Means is particularly effective for high-volume retail data, where computational efficiency is paramount. However, it struggles with datasets that contain noise or clusters of varying density. To address these limitations, hybrid approaches that combine K-Means with density-based methods like DBSCAN are increasingly being explored.

2.3.2 Hierarchical Clustering

Hierarchical clustering constructs a tree-like structure of clusters and is thus mainly appropriate for those datasets, which require nested groupings. The vast potential of the method in discovering macro- and micro-segmentations is discussed by A. D. Fallis in his article "Hierarchical Clustering Approaches for Large Scale Data" [10]. The researchers use linkage methods like *Ward's method* to minimize within-cluster variance at every step [11]. Hierarchical clustering presents the visual insight of the clustering structures through the dendrograms, which facilitate the selection of optimal cut points.

The multilevel segment structure revealing feature is one of the key advantages of hierarchical clustering. On the other hand, its computational complexity prevents it from being applied on a very large scale.

2.3.3 DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a method specifically designed for detecting various types of clusters and isolating the outlying data. The original method was developed by M. Ester et al. and was elaborated

in the paper "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise", which indicated that it could be adapted for use in different contexts [12]. Unlike traditional clustering algorithms, DBSCAN focuses on two essential values; Pre-defining cluster numbers are not required. The parameters are epsilon (ϵ) and the minimum number of points (*MinPts*).

For example, it can isolate high-value outliers such as corporate clients or bulk buyers, which might be misclassified in centroid-based methods. However, its performance depends heavily on parameter tuning, which requires expertise.

2.3.4 Gaussian Mixture Models (GMM)

Gaussian Mixture Models assume that data arises from a mixture of Gaussian distributions. J. Han et al. in "Data Mining: Concepts and Techniques" detailed GMM's flexibility, which allows clusters to take on various shapes [13]. GMM uses the *Expectation-Maximization* algorithm to iteratively refine cluster parameters, making it suitable for datasets with overlapping or non-spherical distributions. However, its reliance on Gaussian assumptions may limit its effectiveness for highly skewed or multi-modal data.

GMM is particularly suited for datasets where clusters exhibit significant overlap, such as customer groups with similar spending patterns but different product preferences. Its probabilistic framework allows for soft clustering, assigning each customer a likelihood of belonging to multiple clusters.

2.3.5 Fuzzy C-Means

Fuzzy C-Means extends traditional clustering by assigning membership degrees to each cluster, accommodating blurry segment boundaries. J. C. Bezdek's book "Pattern Recognition with Fuzzy Objective Function Algorithms" introduced this concept, emphasizing its relevance for datasets where customers exhibit overlapping behaviors [14]. While advantageous for capturing subtle differences, Fuzzy C-Means requires careful tuning of parameters like the fuzzifier, which can complicate its application.

The flexibility of Fuzzy C-Means makes it particularly valuable for customer segmentation in industries with diverse product offerings. For instance, in e-commerce, customers might exhibit characteristics of both "bargain hunters" and "premium buyers." By allowing partial membership, Fuzzy C-Means provides insights into hybrid customer profiles, enabling more personalized marketing strategies.

2.4 Leveraging CLV and RFM: Clustering

Empirical studies advocate combining **CLV metrics** with advanced clustering algorithms for a multidimensional understanding of customer behavior. Khajvand et al.'s study demonstrated how *CLV-weighted RFM* scores enhance segmentation by identifying high-lifetime-value customers with sporadic activity [4]. This aligns with

Gupta et al.'s concept of **persistence models**, where potential future value guides marketing strategies [5].

DBSCAN's ability to isolate outliers has proven effective for identifying "sleepers" or occasional high spenders who might otherwise be overlooked by centroid-based methods [15]. Real-time clustering approaches are gaining traction, adapting dynamically to updated transactions, returns, or seasonal variations [13].

Integrating CLV and RFM with clustering algorithms provides a holistic view of customer behavior. For example, a company might use Fuzzy C-Means to identify hybrid profiles, combining this insight with CLV metrics to prioritize high-value segments. Similarly, DBSCAN can uncover hidden patterns in noisy datasets, while GMM offers probabilistic insights into overlapping customer behaviors.

2.5 Conclusion

The surveyed literature marks a transition in the methods of customer segmentation from the traditional techniques that are based on *RFM* to multisided techniques incorporating *CLV*. Traditional models like *RFM* are able to classify customers by the recent transactions, frequency, and monetary value very well, however, the static property that they have does not allow them to see the long-term behavioral trends. On the contrary, the *CLV* method presents the outlook through the integration of the profit and retention estimates. The algorithms are further clustered into groups and this makes the segmentation strategy more effective. The methods of K-means, Hierarchical Clustering, DBSCAN, Gaussian Mixture Models, and Fuzzy C-Means each have their strengths, but also their limitations.

As such, K-Means provides the ability of fast computation for big data sets, while DBSCAN, in addition to dense difficult-to-spot clusters, is able to recognize and separate outliers by not needing a predetermined cluster number. Meanwhile, it is through Gaussian Mixture Models and Fuzzy C-Means that the statistical analysis capabilities are increased based on the probabilistic and soft clustering frameworks, which are particularly interesting for cases where data has both overlapping or non-spherical forms. These conclusions form a strong theoretical basis for the empirical study that follows.

Chapter 3

Methodology

The methodology of the thesis follows a structured, sequential approach to process and analyze the dataset for customer segmentation using advanced clustering techniques [1]. Below is an outline of the methodology as illustrated in the figure 3.1

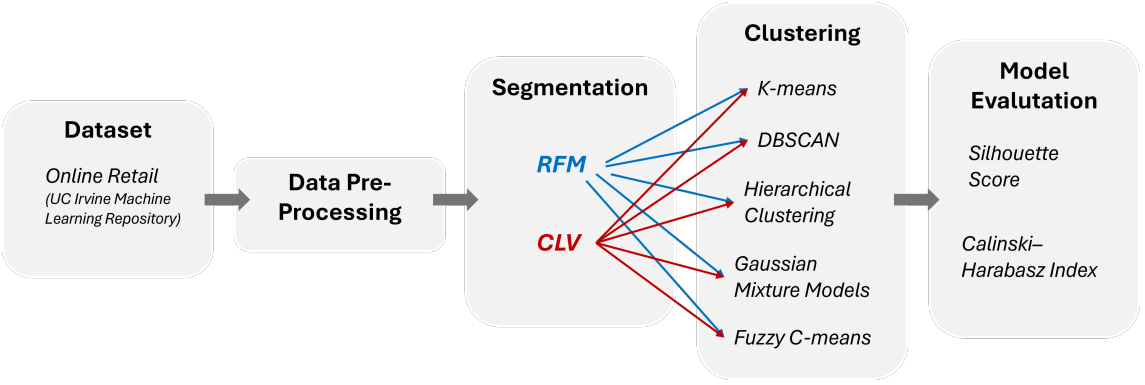


Figure 3.1: The structure of the methodology

- **Dataset**:
  - The analysis begins with a dataset sourced from the UCI Machine Learning Repository, containing transactional data from an online retailer.
- **Preprocessing**:
  - The dataset undergoes preprocessing to clean and refine the data.
- **Segmentation**:
  - The refined data is segmented using two approaches:
    - **RFM (Recency, Frequency, Monetary) segmentation**: Assigning scores to customers based on their purchase behavior [3].
    - **CLV (Customer Lifetime Value) calculation**: Estimating the long-term value of each customer [4].

- **Clustering Models**:
  - Both RFM and CLV segmentations are independently subjected to clustering using five models
    1. **K-means**
    2. **Hierarchical clustering**
    3. **DBSCAN**
    4. **Gaussian Mixture Models (GMM)**
    5. **Fuzzy C-means**
- **Model Evaluation**:
  - Each clustering solution is evaluated using two metrics:
    - **Silhouette Score**: Measures cluster cohesion and separation.
    - **Calinski-Harabasz Index**: Assesses the ratio of between-cluster to within-cluster variance [8].
  - The evaluation compares the performance of clustering models for both RFM and CLV segmentations.

3.1 Data Collection

The dataset utilized in this thesis was sourced from the UCI Machine Learning Repository. It contains transnational communication records of a UK-based non-store online retailer and duration of December 1, 2010, to December 9, 2011.

Dataset Description: The dataset consists of 541,909 transactions. It includes eight attributes which could be used for various types of analysis together with time series data on retail operations and customer demographics.

- **InvoiceNo**: Unique identifier of the transaction; if prefixed with 'C', shows it is a cancellation.
- **StockCode**: The code is unique for each item.
- **Description**: Description of the item in a text format.
- **Quantity**: The amount of products sold in each transaction.
- **InvoiceDate**: Each transaction's timestamp (e.g., "12/1/2010 8:26").
- **UnitPrice**: The price of one product unit.
- **CustomerID**: Identifier for each customer uniquely.
- **Country**: The customer's country.

3.1.1 Preprocessing and Data Integrity:

To ensure the data was reliable for the analysis the following preprocessing steps were followed:

- **Data Cleaning**: Duplicate records were filtered out to ensure the uniqueness which is important for analysis. Cancellations were removed to keep the data in the exact order, but they were identified as such by their InvoiceNo prefixes.
- **Date Parsing**: The 'InvoiceDate' field was converted from a string format to a date-time object for the time-series analyses.
- **Error Handling**: Entries with missing CustomerID or with unrealistic transaction values (e.g., negative prices or quantities) were carefully checked and appropriately handled.

Finally, a new column named "TotalPrice" was included in the dataset. It was calculated by multiplying the number of items by their unit price. This step was essential for calculating the Monetary Value for each transaction.

The dataset shrank after the preprocessing was done to 392692 records distributed in nine columns (original eight and a new "TotalPrice"). Thus, the data set is now ready for the further segmentation.

3.2 Segmentation Methodology

3.2.1 RFM Framework

The Recency, Frequency, Monetary (RFM) model is a widely used customer segmentation technique that is commonly used in database marketing and retail analytics. In this model, customers are evaluated by means of a score based on three specific criteria:

- **Recency (R):** Recency is a metric that measures how recent was the purchase performed the customer. A lower recency value means that the customer bought more recently in the store or business, which means that he/she is more engaged and therefore he/she has more chances for repeat purchases. Recency is calculated as the difference of days between the current date and the customer's last purchase date:

$$R = \text{Current Date} - \text{Last Purchase Date}$$

- **Frequency (F):** Frequency refers to how many times a customer purchases an item over a specific period. Retailers frequently count the total number of transactions made by each customer during a certain time frame to calculate their loyalty and engagement. Many times, trust and satisfaction are indicated by the number of interactions that a customer has had with the brand:

$$F = \text{Total number of transactions}$$

- **Monetary (M):** This metric provides the figure of how much money a customer has spent over a certain range of time. Increased values of money are a parameter of more customer reach to the company. It is common for businesses to utilize this aspect of the customers to find out who their 'high spenders' are, and plan for the future by predicting their revenue:

$$M = \text{TotalPrice} = \text{Unit Price} \times \text{Quantity}$$

The enforcement of the RFM model in the company will allow for making the marketing strategies more differentiated and for also personalizing the communications to each group of customers. This not only focuses on the most lucrative segments, therefore, optimizing the marketing process but also makes the customers satisfied.

3.2.2 RFM Score Calculation and Segmentation

For the segmentation of the customer dataset to be successful, all RFM metrics were quantified and scored based on quintiles. Each customer was given a score from 1 to 5 for each parameter, where a score of 5 indicated the best 20% of behavior (e.g., purchases that were made the most recently, highest frequency, and the highest spending).

- The customers were ordered based on every parameter and each order was divided into five equal sections (quintiles). The quintiles helped in the allocation of the scores that would be assigned to each RFM parameter.
- The Recency scores were reversed, where newer customers scored higher (i.e., a customer having a purchase of the most recent date gets a score of 5).
- Frequency and Monetary values were summarized normally; thus, the higher rates the higher point values were assigned.

With the help of this methodology, it is possible to single out the groups that are different in their transactional habits and find suitable paths for targeted marketing.

Defining Customer Segments

Based on the calculated RFM scores, customers were classified into six segments. This classification helps in tailoring marketing efforts according to the specific characteristics of each segment:

Thus, regarding the prior table, we formulate the zoning as follows

- **Whales**: Among the high-caliber consumers that are granted a lot of attention, they have purchased recently, have a high transaction frequency, and have spent considerably much. Hence, they are the kind who will need particular retention strategies and exclusive promotions.
- **Active Stars**: They are close to whales in actions however; they are a bit less intense than whales in behaviors yet, they are key figures as long as they spend and purchase frequently.

Table 3.1: Customer Segmentation Based on RFM Scores

R_score	F_score	M_score	Segment
5	5	5	Whales
≥ 4	≥ 3	≥ 4	Active Stars
≥ 4	≥ 4	≤ 3	Loyal Regulars
≤ 3	≥ 3	≥ 4	Sleeping Giants
≥ 4	≤ 3	≤ 5	New & Occasional Buyers
	Other cases		Lost Clients

- **Loyal Regulars**: They are consistent and have recently bought products with moderate spending, thus, they create a stable income and can be attracted with promotions to increase their spending.
- **Sleeping Giants**: They are previously good customers that now are not buying, however, they are seen as the potential revenue source if they are summoned back accurately.
- **New & Occasional Buyers**: This group is comprised of clients with recent purchases but not frequent ones, but rather rare tops and downs, and they have a chance of becoming the regulars through the adjustment of retainment.
- **Lost Clients**: These customers are the least involved and have historically spent low amounts, consequently, they are sometimes the last ones to prioritize unless specific re-engagement strategies are feasible.

3.2.3 Customer Lifetime Value (CLV)

The concept of Customer Lifetime Value (CLV) is another way of segmenting customers. It assists in determining how much a company should spend on the maintenance of relationships with existing customers and in acquiring new ones [5].

CLV is the overall profit that a corporation would realize by a given customer during their business relationship. It is calculated as the profit margin of the different transactions, retention rate and discount rate are taken into account for the time value of money. With the help of CLV, companies should be able to optimize their marketing resources on customers that are expected to yield the most lifetime value.

Methodology for Calculating CLV

The approach used in calculating CLV in this particular project is a set series of steps that are followed. For detailed descriptions, the dataset used in this study was described in previous chapters, and the cleaned and preprocessed ones have been used. Here, we highlight each of the steps that are involved in the calculation:

Calculation of Key Metrics Three major metrics are being calculated for every client:

- **Total Revenue:** It is the total addition of prices namely `TotalPrice` which is a sum for each customer's all transactions that gives the cumulative value of money the customer has spent. Simply, the total is the multiplication of the `Quantity` and `UnitPrice` for each item and the addition of all of them.
- **Frequency:** Here this index estimates the customer loyalty and purchasing frequency as the number of unique invoices per customer that reveals how frequently the customer is transacting.
- **Average Transaction Value:** Generally, each customer is meant to have an average by which the `TotalPrice` for each transaction is totally divided for attaining this metric. Thus, this metric provides a useful insight into the customer for the total transaction money.

Lifetime Calculation The lifespan of the customer relationship is calculated by determining the number of days between the first and last purchase dates. This metric provides a temporal dimension to the monetary and frequency values, offering a better view of the customer's engagement over time.

Computation of CLV The CLV is then calculated using the formula:

$$CLV = \text{Frequency} \times \text{Average Transaction Value} \times \text{Expected Lifespan}$$

Where:

- **Frequency** is a number of transactions (as calculated earlier).
- **Average Transaction Value** is mean spending per transaction.
- **Expected Lifespan** is an estimated duration of the customer relationship in months. This duration can be adjusted based on historical data or industry averages.

This formula incorporates both behavioral (Frequency, Average Transaction Value) and temporal (Expected Lifespan) elements to give a holistic estimate of the customers total potential value to the business.

The Role of CLV in Customer Segmentation

Having calculated the CLV, this metric now serves as the fundament for the further segmentation analysis. It will enable the business to divide its customers into groups on the basis of value, which will be the basis for the more precise and efficient marketing strategies. For instance, the company's high-CLV customers can be specifically selected to access premium facilities, while the company can launch strategies to promote a higher CLV for those customers that score lower.

The calculated CLV not only evaluates the previous and present value of customers but also helps to assure the future customer transactions and profitability and therefore, are guides in management and marketing of strategic decision. Implicitly

these calculations of CLV prove the segmentation and targeting to be economically grounded thus maximizing the ROI (Return On Investment) in customer relationships.

3.3 Clustering Algorithms

3.3.1 K-Means Clustering

K-means clustering is a trending method of unsupervised machine learning by which similar data points are grouped into a predetermined number of clusters denoted by k. This method has become a special application of customer segmentation: through customer segmentation, distributors can benefit by realizing the fragmentary structure of their customer base and enable the tailoring of strategies like marketing [9].

Algorithm Overview

K-means is the algorithm of arranging n observations to k clusters where every observation belongs to the cluster with the nearest mean, which acts as the prototype of the cluster. First, k centroids are randomly selected from the data set, which are, in turn, assigned each data point to the closest centroid based on the Euclidean distance and recalculated the centroids as the mean of all points in the cluster. The iteration of this process continues with points being reassigned and centroids being upd ated until the centroids remain unchanged showing a slight or no movement at all showing convergence.

Mathematical Formulation

The purpose of K-means is the minimization of the Gaussian clustering error or the within-cluster sum of squares (WCSS), which is the sum of the squared Euclidean distances from each point to its centroid expressed mathematically as:

$$\text{Minimize} \quad WCSS = \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2$$

where μ_i is the mean of the points in S_i, and k is the sum of clusters.

Choosing the Number of Clusters

The 'Elbow Method' is the most common approach to the selection of the optimal number of clusters [9]. It encircles the range of the k and runs K-means clustering for all of them thereby capturing the trend across WCSS for each value of k. The best value of k usually is at the point of WCSS leveling where the curve has an elbow shape. A cut-off point is a right term for this method since it displays visually the point after which further increasing clusters has no significant influence on within-cluster variation. The Elbow Method is especially useful in cases of plateau WCSS when the decrease is marginal, hinting at the possibility of overfitting due to higher cluster numbers without much gain in outlier separation.

Algorithm Steps

1. **Initialization**: Choose k centroids at random from the dataset.
2. **Assignment**: Each point of data is assigned to the nearest centroid according to the Euclidean distance.
3. **Update**: Centroids are calculated as the mean of the points in each cluster.
4. **Repeat**: Assignment and update create steps until centroid changes are less than a threshold or non-existent.

Implementation in R

In R, K-means clustering is realized using the *stats* pack which complements the kmeans() function that implements this clustering result directly. This function stands out for its flexibility since it allows specifying the cluster number, the maximum number of iterations, and start options on random to improve the robustness and accuracy of the clustering results.

3.3.2 Clustering Algorithms: Hierarchical Clustering

Hierarchical clustering is a branch of cluster analysis that aims to create a hierarchy of clusters. In this research, we particularly consider agglomerative hierarchical clustering, which is a bottom-up scheme in which each observation is made its own cluster, and then pairs of clusters are successively merged as one ascends the hierarchy. This approach is used mainly due to its stability in uncovering the natural subdivisions of a data set, e.g., customer segments in the RFM data [10].

Agglomerative Hierarchical Clustering Overview

Agglomerative hierarchical clustering works from the data point to the cluster and is the most common approach to this type of clustering. Pairing the two metrics employed to measure distances - Ward's method as the criterion of linkage - with the Euclidean distance will be the best choice in this research due to its general stability and simple parameters compared to other benchmarks.

Algorithm Overview

The algorithm for hierarchical clustering that is implemented in this study first calculates a distance matrix that shows the distance between each pair of readings in the normalized RFM space taking into account the Euclidean distance. This matrix is important as the basis for establishing which clusters are the nearest and should thus be combined in the iterative process.

The clustering process through the following steps:

1. **Initialization**: Start with n clusters (each containing one data point).

2. **Merge Clusters**: During iterations, merge the pair of clusters that produce the smallest increase in the total within-cluster variance according to Ward's method.
3. **Update Distance Matrix**: Upon merging two clusters, the distances between the new cluster and others should be updated.
4. **Termination**: The process should be repeated until all data points are merged into a single cluster.

Ward's Method Explained

Ward's method which serves as the linkage criterion in the hierarchical clustering scheme is an exemplary way to create spherical clusters that are compact. Different from other linkage methods that deal with distances between the closest or furthest points (single and complete linkage, respectively), Ward's method eliminates the total within-cluster variance at every step of clustering.

At every point of agglomeration, the Ward's criteria would compare each pair of clusters with the potential variance if they were merged. The closest to no variance increase is taken for merging. The approach is mathematically described by the following formula:

$$\Delta \text{SS}(k, l) = \frac{n_k \times n_l}{n_k + n_l} \times \|\mu_k - \mu_l\|^2$$

Where n_k and n_l are the sizes of the clusters k and l respectively, and μ_k and μ_l are the mean vectors of the clusters. The formula shows the increase in the total sum of squared deviations from the mean (SS) when two clusters are combined.

The algorithm described in Joe H. Ward, Jr.'s paper, "Hierarchical Grouping to Optimize an Objective Function," is most useful in statistical data analysis. It pursues the goal of minimizing variance, so it is claimed to be more objective than threshold methods that set distance cutoffs subjectively. [16]

Implementation in R

Hierarchical Cluster in R was implemented with the `hclust` function which is one of the functions in the standard statistical toolset of the R programming environment. The `dist` function was utilized to calculate the initial distance matrix using the Euclidean technique, and Ward's method was chosen through the `method` argument of the `hclust` function. This would mean that the clustering process would be optimized within each group which is imperative for the meaningful formation of customer segments.

3.3.3 DBSCAN Clustering

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is an eminent clustering algorithm that denotes clusters as dense points isolated by areas of low

density. It is much acclaimed for its ability to cope with noise and to recognize outliers thus it is fitting for the datasets that contain irregular patterns and the number of clusters is unknown [12].

Algorithm Overview

DBSCAN is a technique based on the core point concept which grows clusters from them. It does not need the specifications of the number of clusters upfront but relies on two inputs: the epsilon radius and the minimum number of points for the creation of a cluster (MinPts). Reasons like this flexibility of the method to shape the clusters in any desired form and robustness against noise and outliers stand behind the increased utility in a wide range of applications [17].

Core Concepts of DBSCAN

The DBSCAN algorithm derives from two major parameters:

- **Epsilon (ϵ)**: This is the radius of the neighborhoods surrounding each point. the points that are inside the ϵ neighborhood of a core point are directly density reachable.
- **Minimum Points (MinPts)**: This is the minimum number of points necessary to make a dense area. A point is designated as a core point if its ϵ-neighborhood contains more than MinPts.

Algorithm Steps

1. **Parameter Selection**: The determination of reasonable values for parameters, ϵ and MinPts, should be done using domain knowledge or heuristic methods, such as the k-distance graph.
2. **Core Points Identification**: Each point in the dataset is labeled as the core point if the point has, at least, MinPts points in its ϵ-neighborhood.
3. **Cluster Expansion**: Start the process by a core point and then include all the directly density-reachable points to the cluster, which expand to the points that are density-connected to the points.
4. **Point Classification**: All the points in the dataset are assigned with a label that is either core point, border point, or noise. Therefore, the points, which do not match the core or border criteria, are considered as noise and they do not belong to any cluster since they are in low-density areas.

Parameter Estimation

This assessment was carried out based on the dataset characteristics:

- **Epsilon (ϵ)**: The k-distance plot method was used that displays the distance to the k-th nearest neighbor in a plot. The ϵ value was determined at the 'elbow' point from the graph, that is, the inflection point, which provides the best trade-off between coverage and accuracy.

- **Minimum Points (MinPts)**: In this case, the dimensional nature of the dataset was the cause of the result; the usual MinPts was typically greater than twice the number of data dimensions. For this dataset, MinPts started at a value of 5 to find a few meaningful clusters while excluding noise.

DBSCAN in Action

The application of DBSCAN in the current research was carried out by the `dbscan` package in R which provides a simple method with great flexibility being offered in parameter adjustments of this algorithm. After determining the optimal ϵ via the elbow method from the k-distance plot, DBSCAN was utilized to segregate the normalized RFM data into several clusters and identify outliers.

3.3.4 Gaussian Mixture Models

Gaussian Mixture Models (GMM) is the cutting-edge theoretical approach to the probabilistic division of data, which is based on the concept that data result from a combination of normally distributed random variables. This method is extremely effective for analyzing data sets including overlapping clusters and non-spherical distributions, such as customer segmentation where the characteristic behaviors and might not be perfectly visible [11].

Algorithm Framework

Every GMM cluster is a Gaussian distribution that is identified by its mean (center), covariance (spread and orientation), and mixture weight in the form of the proportion of that cluster in the entire dataset. The main procedures of GMM are:

1. **Initialization**: The means, covariances, and mixture weights of the Gaussian components are initially set.
2. **Expectation Step (E-step)**: The current parameters are used to compute the probability of each data point belonging to all the clusters.
3. **Maximization Step (M-step)**: To make the computation of data likelihood at these probabilities maximum, parameters are modified.
4. **Convergence Check**: E-step and M-step steps are reiterated until the small changes in the parameters become below a threshold that was defined earlier.

The Mathematical Formulation

In fact, maximization takes the following expression to be:

$$L(\theta) = \prod_{i=1}^{n} \sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(x_i | \mu_k, \Sigma_k)$$

where:

- π_k is the proportion of the k-th mixture component,

- μ_k and Σ_k are the mean and covariance matrix of the k-th Gaussian component,
- $\mathcal{N}$ is a short reference of a probability density function from the Gaussian distribution.

Using the Bayesian Information Criterion (BIC) for Model Selection

In the quest for the best-fit model and the right number of clusters, the Bayesian Information Criterion (BIC) is of extreme utility. The computation formula is stated as:

$$BIC = \ln(n) \times k - 2 \times \ln(\hat{L})$$

where:

- n is the number of data points,
- k is the number of parameters used in the model,
- $\hat{L}$ is the likelihood that the model of the data has been maximized.

It is a commonly known fact that the identification of the model is significantly supported through the use of BIC which penalizes the model complexity while rewarding its fitting goodness. Its aim overall is to detect the model that is neither very complex for the data nor is it fitted poorly as it becomes complex.

BIC in GMM: The Key Role:

- For each GMM setting, the BIC value is calculated in a unique way, where the number of clusters and the covariance type are the key choices.
- The GMM setting which leads to the overall lowest BIC is the one that is selected because it has demonstrated the best fitting of the data and at the same time it has avoided the adding complexity to the model.

Implementation of GMM in R programming

The functioning of GMM in R is possible through the facilitating of the *mclust* package, whose model-based clustering, classification, and density estimation tools act as a toolkit. Its construction is made easy and it is completely automated; hence, the user does not require any prior knowledge for selecting the optimum model and necessary cluster count through BIC, proving the importance of the *mclust* package in data analytics for facing difficult clustering situations.

3.3.5 Fuzzy C-means Clustering

Fuzzy C-means (FCM) is a *soft* clustering algorithm that generalizes k-means providing more flexibility to the method [14]. Whereas *hard* clustering methods (e.g., k-means) attribute each point to a single cluster, FCM can determine the membership of a point to each of the clusters as a grade between 0 and 1. Soft assignment is

particularly useful in situations where cluster boundaries are not well defined or when some data points are naturally being mixed.

Algorithm Overview

FCM has the objective to identify with the following:

1. Cluster centroids $\{v_1, v_2, \ldots, v_c\}$, where c is the total number of clusters.
2. A membership matrix U, with each element U_{ij} representing the degree of the i-th data point to the j-th cluster.

The algorithm goes through the iterations as listed:

1. **Initialization**: Pick c random cluster centroids (could be random or based on some method), and set the initial membership matrix U such that U_{ij} is the temporary membership of point x_i to cluster j.
2. **Membership Update**: For all the point x_i, and for each circle v_j, renew the membership values with the following formula:

$$U_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\|x_i - v_j\|}{\|x_i - v_k\|} \right)^{\frac{2}{m-1}}}.$$

3. **Centroid Update**: Calculate each centroid v_j based on the new memberships:

$$v_j = \frac{\sum_{i=1}^{n} U_{ij}^m \, x_i}{\sum_{i=1}^{n} U_{ij}^m}.$$

4. **Convergence Check**: The procedures of membership and centroid updating are repeated until the changes in the membership matrix or the centroids are less than some preset limit, or until a maximum number of iterations is reached.

Mathematical Formulation

The main target of FCM is to minimize the following cost function:

$$J_m(U, V) = \sum_{i=1}^{n} \sum_{j=1}^{c} U_{ij}^m \, \|x_i - v_j\|^2,$$

where:

- x_i is the i-th data point,
- v_j is the centroid of the j-th cluster,
- U_{ij} is the membership degree of x_i in cluster j,

- m (> 1) is the fuzziness parameter controlling how *soft* the clustering boundaries become.

Fuzziness Parameter and Its Range

A major element in FCM is m (fuzziness parameter). Many applications set it empirically between 1.5 and 3:

- **Lower Bound ($m \approx 1$)**: As m approaches 1, it acts as if the algorithm is a hard clustering method, with memberships close to 0 and 1. This reduces the level of softness that the FCM aims to provide.

- **Upper Bound ($m > 3$)**: Larger values of m will diffuse memberships to the highest level, thus overlapping cluster differences, and interpreting them puzzlingly.

This is the range the suggested setting is working in. Specifically, $m = 2$ is considered as the most balanced choice, which is used most often. This value usually provides enough *softness* to capture overlaps in the clusters, and yet does not cause the memberships to be too widely distributed. In our case, we opted for the choice of $m = 2$ due to its reasonable trade-off: we wanted partial membership in several segments but still clear enough cluster centers to interpret easily.

Interpreting Results

In fuzzy clustering, each customer (or data point) is given a vector of membership values that states the likelihood of cluster membership. Some practical approaches could be:

- **Using Fuzzy Memberships Directly**: Analyze the soft memberships to explore nuanced or hybrid customer profiles. This can reveal hidden overlaps where customers exhibit characteristics of multiple segments.

- **Hard Clustering Conversion**: For simpler comparison with other clustering approaches (e.g., k-means), one can convert fuzzy memberships to the cluster by assigning each point to the cluster with the highest membership. This does not keep as much detail, but, the subsequent analyses and visualizations are simpler.

With the partial memberships being supported, FCM is further enlightening in the area of customer behavior and finding the intersection between the methods that the two generally-accepted methods never brought out. This could be notably beneficial in instances where RFM segmentation is going on since customers on account and frequency used normally both point to boundaries that cumulatively state customer clustering in some archetypes.

Implementation in R

R Programming Language Implementation

In case you want to perform Fuzzy C-means clustering in R, the `e1071` package is a smart choice. This package is available through `cmeans()` function which users are able to set a wide range of parameters for example: fuzzification exponent (`m`), the number of cluster centers (`centers`), the maximum number of iterations (`iter.max`), and so on. As a result, it automatically computes the centroids and then the membership matrix, where each data point has a partial membership to multiple clusters.

3.4 Analytical Tools and Environment

The analytical research carried out in this thesis has been implemented in R programming language, which has been designed primarily for graphics and statistical computing. The version that has been specifically used was **R version 4.3.1 (2023-06-16 ucrt)**, on a **Windows 11** operating system with an **x86_64-w64-mingw32/x64 (64-bit)** architecture. The analysis was performed with RStudio, which is an integrated development environment for the R language. The specific version used was **2024.09.1+394 "Cranberry Hibiscus"**.

Packages and Libraries

Core packages represent foundational packages that although are used in combination with others to assist data analysis, they stand out because of their unique functionalities like visualization, data manipulation, and clustering analysis. The URLs leading to their R Archive Network (CRAN) page are mentioned here.

- **dplyr** (version 1.1.3): This is a data manipulation grammar. It yields a uniform collection of verbs to address the vast majority of data issues.

  URL: `https://CRAN.R-project.org/package=dplyr`

- **lubridate** (version 1.9.2): The package enhances the R preferences of date and time manipulation by providing easy access to time-series data.

  URL: `https://CRAN.R-project.org/package=lubridate`

- **ggplot2** (version 3.5.1): It is the system of declarative graphics, -based on the grammar of graphics, therefore it is flexible and can be layered.

  URL: `https://CRAN.R-project.org/package=ggplot2`

- **plotly** (version 4.10.4): This package is an interface to the Plotly JavaScript graphing library and enables the creation of interactive web-based data visualizations.

  URL: `https://CRAN.R-project.org/package=plotly`

- **cluster** (version 2.1.4): This package comprises of cluster analysis methods including agglomerative hierarchical clustering which is the primary tool needed for customer segmentation.

  URL: `https://CRAN.R-project.org/package=cluster`

- **dbscan** (version 1.2-0): This is a faster implementation of the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm.

  URL: `https://CRAN.R-project.org/package=dbscan`

- **mclust** (version 6.1.1): The package contains the application of finite Gaussian mixture models for model-based clustering and enables automatic selection of the model using BIC.

  URL: `https://CRAN.R-project.org/package=mclust`

- **e1071** (version 1.7-13): It contains fuzzy clustering (`cmeans`) along with some additional machine learning tools like Support Vector Machines.

  URL: `https://CRAN.R-project.org/package=e1071`

- **reshape2** (version 1.4.4): A reshaping and melting of data frames are quite simple; this package is necessary for the pre-analysis or visualization data transformation task.

  URL: `https://CRAN.R-project.org/package=reshape2`

- **fpc** (version 2.2-13): The package includes cluster methods and validation tools. It is a complement for DBSCAN with diagnostic and plotting functions.

  URL: `https://CRAN.R-project.org/package=fpc`

- **ggdendro** (version 0.1.23): It is the one which offers a tool to create easily interpreted dendrograms of hierarchical clustering outputs.

  URL: `https://CRAN.R-project.org/package=ggdendro`

- **fmsb** (version 0.7.6): Adding radar charts has now become much easier; it visualizes the segmentation results from the Customer Lifetime Value (CLV) study.

  URL: `https://CRAN.R-project.org/package=fmsb`

Applications, tools, and knowledge libraries such as the aforementioned ones win a tough battle against the data market. The right version of the tools, which is maintained by the website URLs, increases the correctness and repeatability of the whole research.

3.5 Model Evaluation

Two internal validation metrics, namely the **Silhouette Score** and the **Calinski–Harabasz Index (CH)**, were employed in the evaluation of the clustering solutions for both the RFM and CLV datasets. These metrics were chosen as the

best ones to materialize the compactness of the clusters and their separation, thereby evaluating data more clearly.

3.5.1 Silhouette Score

The Silhouette Score provides the measure of how well the clustering solution is done by suggesting the degree to which the data point treatment is proper through points in its cluster (*cohesion*) and that treatment (point) through the closest other cluster (*separation*). This is a metric that holds great significance in terms of verifying the internal structure of the clusters and also detecting possible errors in the assigned cluster. [13].

Mathematical Definition: For a specific point i, the Silhouette Score $S(i)$ is represented as:

$$S(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))},$$

where:

- $a(i)$: The average distance between i and all other points in the same cluster (intra-cluster distance).
- $b(i)$: The average distance between i and all points in the nearest neighboring cluster (inter-cluster distance).

The complete Silhouette Score is calculated using the mean of $S(i)$ for all points:

$$S = \frac{1}{n} \sum_{i=1}^{n} S(i),$$

where n stands for the total number of points.

Interpretation:

- $S(i) \approx 1$: The point is properly clustered.
- $S(i) \approx 0$: The point is located on the border between the clusters.
- $S(i) < 0$: The point might be in the wrong cluster.

Special Handling:

- **DBSCAN:** Points that are identified as noise (-1) were omitted from the process as they do not belong to any valid clusters.
- **Fuzzy C-means:** The fuzzy memberships are converted to hard cluster assignments by picking the cluster with the highest membership value ($\arg\max$).

3.5.2 Calinski–Harabasz Index

Calinski–Harabasz Index is the method used to assess the clusters by calculating the ratio of the between-cluster variance and the within-cluster variance. It incentivizes proper clustering solutions, where the clusters are separated well and they are compact in size.

Mathematical Definition: The CH index is stated as follows:

$$CH = \frac{\text{Between-cluster variance}/(k-1)}{\text{Within-cluster variance}/(n-k)},$$

where:

- k: The number of clusters.
- n: The total number of points.
- **Between-cluster variance**:

$$B = \sum_{j=1}^{k} n_j \|\mu_j - \mu\|^2,$$

  where n_j is the size of cluster j, μ_j is the centroid of cluster j, and μ is the overall mean of the dataset.

- **Within-cluster variance**:

$$W = \sum_{j=1}^{k} \sum_{x \in C_j} \|x - \mu_j\|^2,$$

  where C_j denotes the points in cluster j.

Interpretation:

- Higher CH values denote better clustering layouts, with the clusters being more condensed and exhibiting greater disparity.

Special Handling:

- **DBSCAN:** In the case where the algorithm only detects one cluster or labels every point as noise, the CH index would not be existent.
- **Fuzzy C-means:** Like Silhouette Score, the fuzzy memberships were converted to hard assignments prior to calculating the CH index.

Chapter 4

Results

4.1 RFM Segmentation & Clustering

In this subsection, the results of the RFM segmentation and subsequent clustering will be discussed.

4.1.1 RFM Segmentation Results

Segment Distribution

Using the segmentation method, the total number of customers can be reduced to the number of those that have some undefined type of problem or a very actively engaged person. These segments have been determined on the basis of the recency, frequency, and monetary value obtained from the scores of the RFM.

The chartists of the customers spread between the segments are demonstrated in figure 4.1. The results depict the following important things on customer behavior:

- **Lost Clients** form 47.12% of the customer base, which is the largest segment, that is mainly characterized by the longest recency and the lowest indicators of frequency and monetary, therefore, the group suggests a minimal recent engagement and low revenue contributions.
- **Active Stars** consist of 15.65% of the customers, having a couple of parameters that suggest a positive outcome, such as moderate recency and frequency as well as a higher monetary value; consequently, they are a subject for investment.
- **Whales** together account for 8.02% of the customer base but are remarkable for their high activity with a frequency rate of 18.2 transactions and monetary amount averaging €11,222; consequently, they are the most valuable and major contributor group.
- The leftover groups like **Loyal Regulars** (4.03%) and **New & Occasional Buyers** (12.26%) depict the trends of stabilizing or occupying the market.

- **Sleeping Giants** (12.91%) continuously contribute partly, but on average, they are slower with their purchases where periods of inactivity are longer.

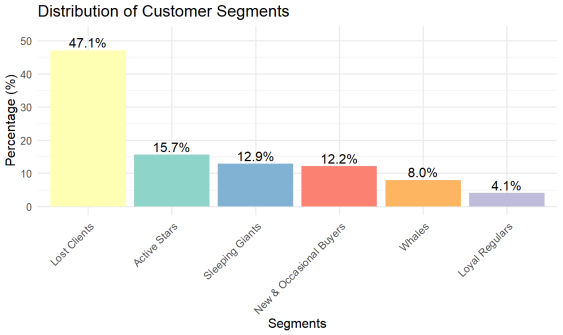


Figure 4.1: Distribution of RFM Segments Analysis

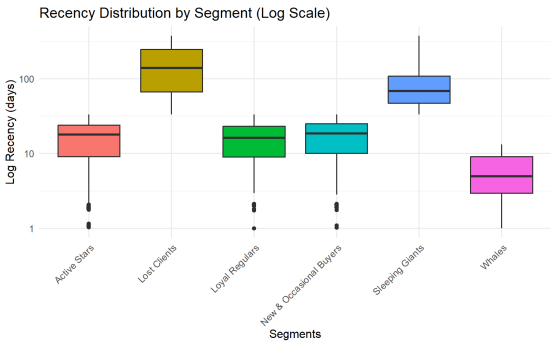
Segment Statistics

The central statistics for each segment can be found in the Table 4.1. The assessment of recency, frequency as well as monetary metrics provides a fuller outlook to the behavior of the customers:

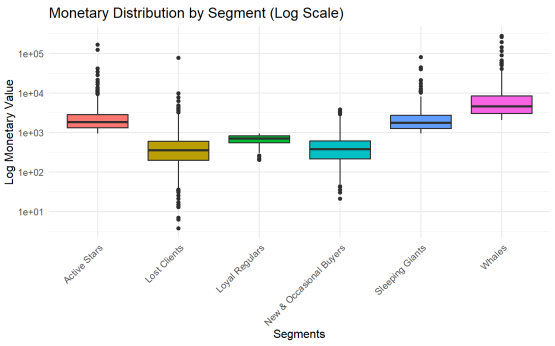
- **Whales**, who have the lowest *RecencyMean* (5.93 days) are the ones who have the most recent activity and therefore are the most loyal to the brand.
- On the other hand, **Lost Clients** with the longest *RecencyMean* (161 days) and the lowest amount spent are the ones that might need different customer acquisition strategies or should be taken out of the upcoming campaigns.
- **Active Stars** hold a *MonetaryMean* of €3,156 and a moderate number of transactions (6.52), so they can be ranked as a steady but still developing group.
- **Sleeping Giants** showed a certain interest with their not low *FrequencyMean* of 5.15 transactions, but the fact that their *MonetaryMean* was €2,656, had to be mentioned.

Segment	R Mean	R Median	F Mean	F Median	M Mean	M Median	Size
Active Stars	17.10	17.80	6.52	5	3156	2500	679
Lost Clients	161.00	138.00	1.60	1	495	450	2044
Loyal Regulars	15.90	16.00	4.06	4	671	520	175
New & Occasional	17.60	18.50	1.66	2	464	390	532
Sleeping Giants	88.20	68.00	5.15	4	2656	2400	560
Whales	5.93	4.94	18.20	13	11222	9500	348

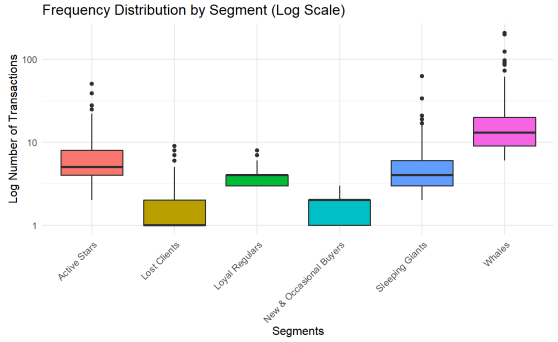
Table 4.1: Segment Statistics for RFM Analysis



(a) RFM log R



(b) RFM log M



(c) RFM log F

Figure 4.2: Logarithmic boxplots of RFM indicators

4.1.2 Clustering Gains on RFM Segmentation

The livestock doling out clustering analysis on the RFM data was performed through five different scales such as K-Means, DBSCAN, Hierarchical Clustering, Gaussian Mixture Model (GMM), and Fuzzy C-Means (FCM). The methods themselves were determined through previously proven data patterns and the practical implications of the clustering on customer segmentation. The clustering parameters têm been set by the particular evaluation techniques like the Elbow Method, the k-NN Distance Plot, and the BIC. This section will cover a thorough report on the clustering results; namely the statistical summaries, the visualizations and the logical reasons for parameter selection.

K-Means Clustering Results

K-Means clustering phase was exclusively executed to categorize customers into unique segments by normal RFM data. The optimal numerals for clusters were achieved through elbow method.

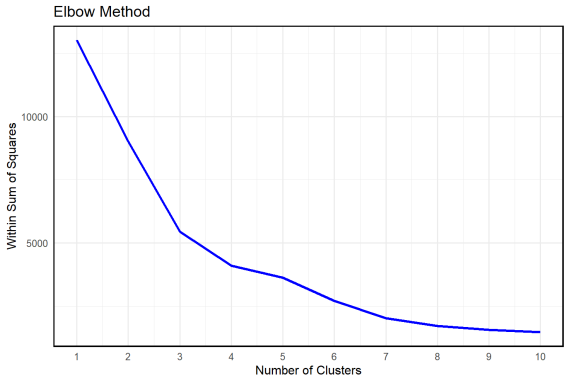


Figure 4.3: Elbow method graph

Elbow Method for Optimal k: The Elbow Method graph in Figure 4.3 shows that the best number of clusters is $k = 4$. At this moment, the difference between j 'WSS Inside' is huge than k=3 but the rate of decrease is almost - 'dead flat' at k=4 or beyond. Thus, this value is a good combination of WSS minimization and acceptance/modeling complexities balancing.

Clust	R Mean	R Median	F Mean	F Median	M Mean	M Median	Size
1	16.0	5.97	22.1	19	12510	7931	209
2	7.66	2.03	82.5	63	127338	117380	13
3	249.0	244.0	1.55	1	478	310	1061
4	44.5	33.0	3.66	3	1353	826	3055

Table 4.2: Cluster Statistics for K-Means Clustering.

Visualization: This information indicates that the customers are divided into different clusters where the Recency and Frequency dimensions are used in the K Means 2D graph shown in Figure 4.4. The high value of Cluster 2 is expressed by the size of the points which represent the monetary value, that is the smaller points indicate lower money spent.

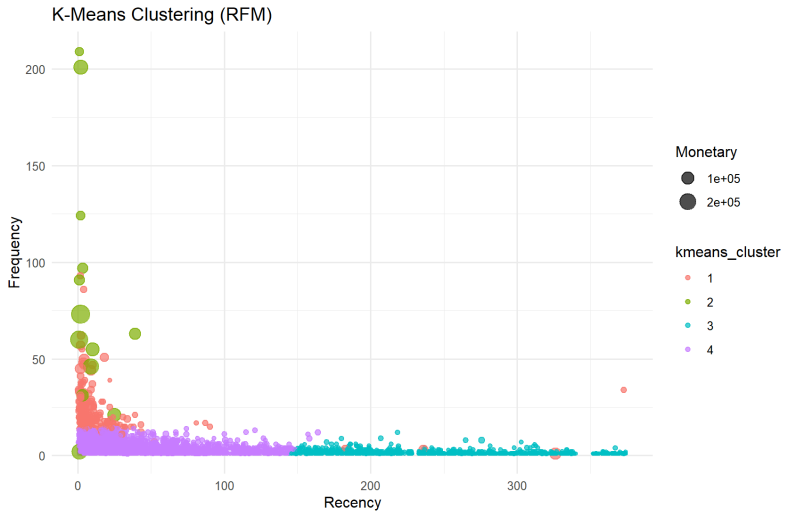


Figure 4.4: K Means 2D graph

Insights:

- Cluster 2 comprises high-frequency customers with recent transactions and high monetary contributions, which in turn makes it the prime retention target. The total monetary value across all clusters is estimated to be **8,910,557€**. Out of the total amount, **1,655,394€**, which is **18.58%** of the total, comes from Cluster 2. This indicates that in this cluster, that is, only 13 customers out of a total spent a considerable amount of money due to their good behavior.
- Cluster 1 and cluster 4 are odd, in recency profiles, are moderate-frequency and monetary customers, respectively.
- Cluster 3 which is the biggest is made of the customers who are low-frequency and low-value thus underlining the necessity of special approaches to motivate them to move to the higher-value groups.

DBSCAN Clustering Results

The DBSCAN algorithm was applied to normalized RFM data in search of non-convex shaped and denser clusters. In contrast to K-Means where predefining clusters is needed, this DBSCAN operates without a defined model and hence is perfect for noise point detection. Nevertheless, it requires a lot of parameter settings particularly of `eps` (epsilon) and `MinPts`.

Parameters for DBSCAN: The k-NN distance plot is displayed in Figure 4.5 and was used to compute the optimal value for the `eps` parameter. A potential point where the I-NN distance from a neighboring nearest neighbor is stable almost and then ascends fastly may be found where the eps parameter is value around `eps = 0.45`. In this situation, the `MinPts` was set to 5 (as stated in the previously pointed out Methodology).

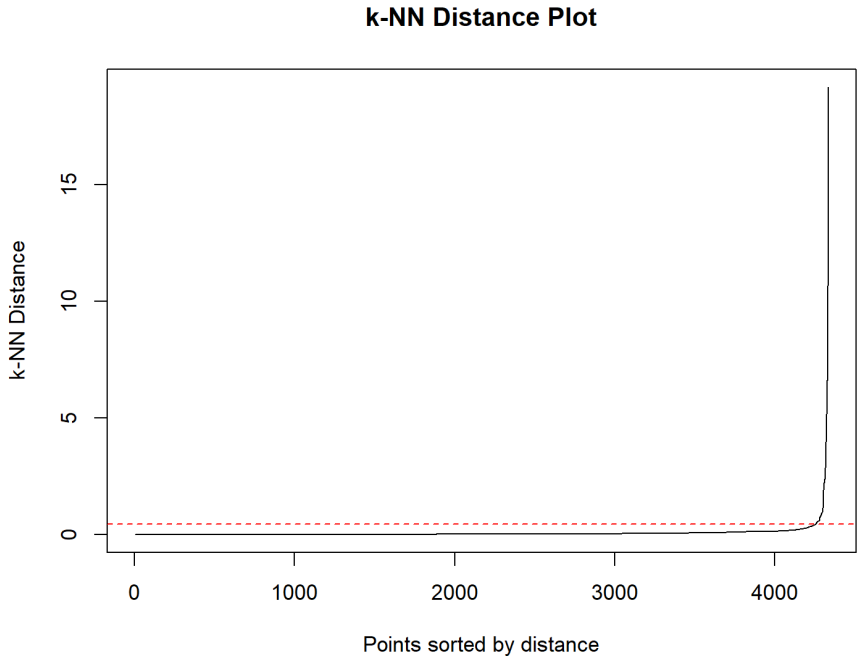


Figure 4.5: k-NN Distance Plot for DBSCAN Parameter Selection (`eps = 0.45`).

Cluster Statistics: DBSCAN, the clustering method, categorised the data into three groups, one of which was marked as noise cluster (`Cluster 0`). The table 4.3 provides a complete description of all the clusters.

Cluster	R Mean	R Median	F Mean	F Median	M Mean	M Median	Size
0(Noise)	51.1	8.98	39.5	31	48452	30358	62
1	93.8	52.0	3.73	2	1376	659	4272
2	3.27	3.05	38	38	7413	7065	4

Table 4.3: Cluster Statistics for DBSCAN Clustering.

Visualization: The figure 4.6 visualizes the DBSCAN clustering results.

Insights:

- **Cluster 0 (Noise):** Quite unexpectedly, this noise group is made up of 62 clients with a very high monetary value (*Monetary Mean = 48452€*). These clients are most probably outliers or very high-value clients according to the criterion defined by dense RFM space of DBSCAN. This situation contributes to the concerns regarding the algorithm's performance on these critical data points since, in classifying a noise these important customers, the algorithm undermines the segmentation's strategic value.

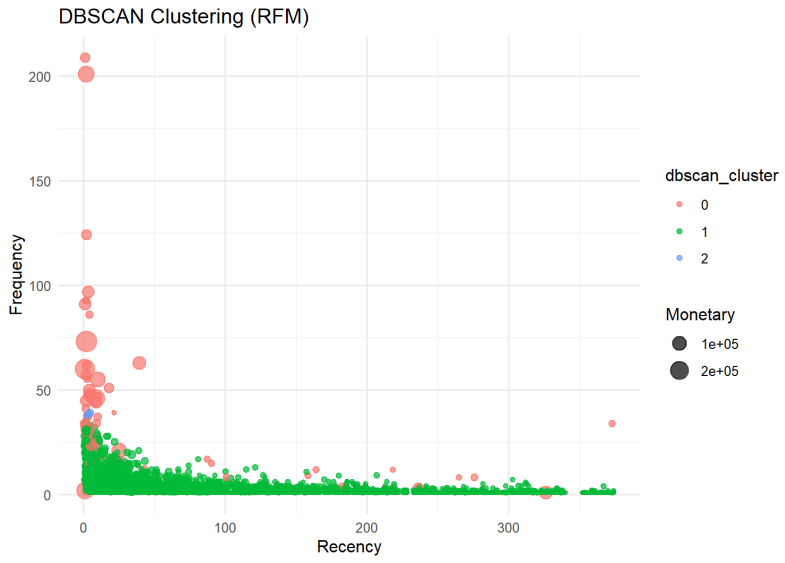


Figure 4.6: DBSCAN Clustering (RFM)

- **Cluster 1:** This cluster has a large proportion of customers (*Group Size = 4272*) with low frequencies and low monetary values. This group contains low-value, infrequent customers, and this is consistent with the expectations for this section of the dataset.
- **Cluster 2:** Cluster 2 is particularly remarkable because it is composed of only 4 customers who show an exceptionally high frequency (*Frequency Mean = 38*) and only moderate monetary contributions (*Monetary Mean = 7413€*). This could mean that they are a small number of loyal customers that buy often but spend less. Alternatively, these customers could also behave peculiarly when they buy products with low-value, high-frequency transactions.

Although DBSCAN is adaptive to the finding of the clusters in different forms and densities, it shows limits in the RFM dataset's usage. The indication of high-value clients as noise implicates that the parameters chosen are insufficient to portray the data structure properly. Moreover, the small isolated cluster that appeared (Cluster 2) made us consider the algorithm's sensitivity to sparse regions and outliers.

Hierarchical Clustering Results

The normalized RFM dataset is used for hierarchical clustering with the application of the Ward.D2 method. This method's objective is to minimize the total variance within each cluster to achieve the formation of compact, well-separated clusters. Unlike clustering methods, hierarchical clustering builds a dendrogram, which is a visual representation of the data's nested structure.

Number of Clusters: The number of clusters was determined manually based on the dendrogram interpretation (Figure 4.7). A cutoff of seven clusters was chosen based on the distribution and spacing of the branches in the dendrogram. The

cutoff chosen was a good balance between interpretability and granularity. The red rectangles in Figure 4.7 represent the separation of data into seven clusters.

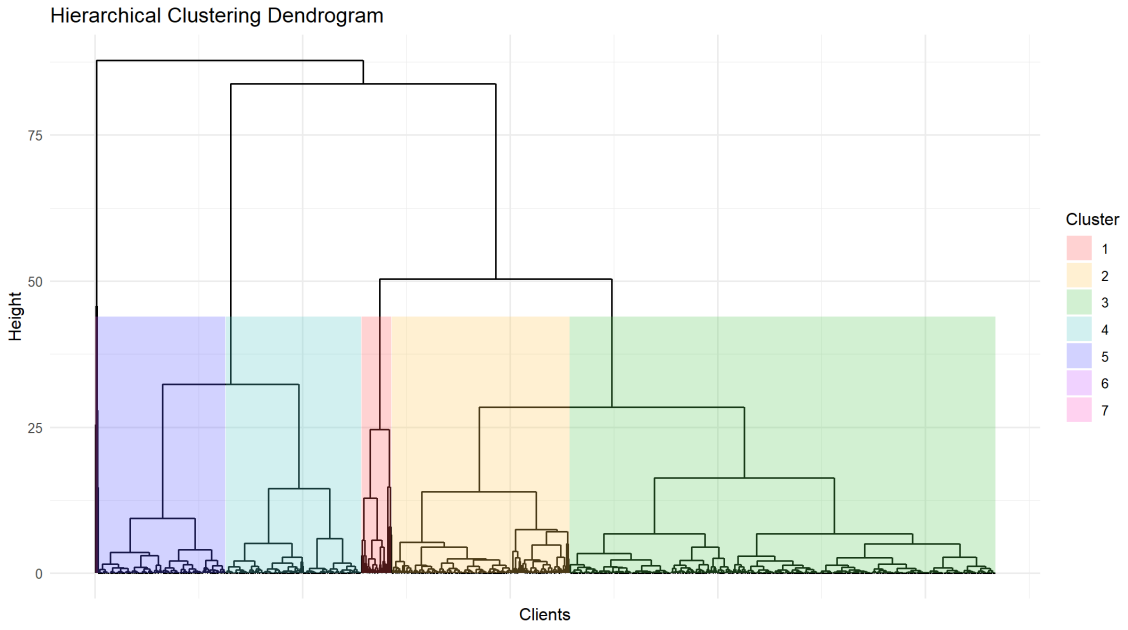
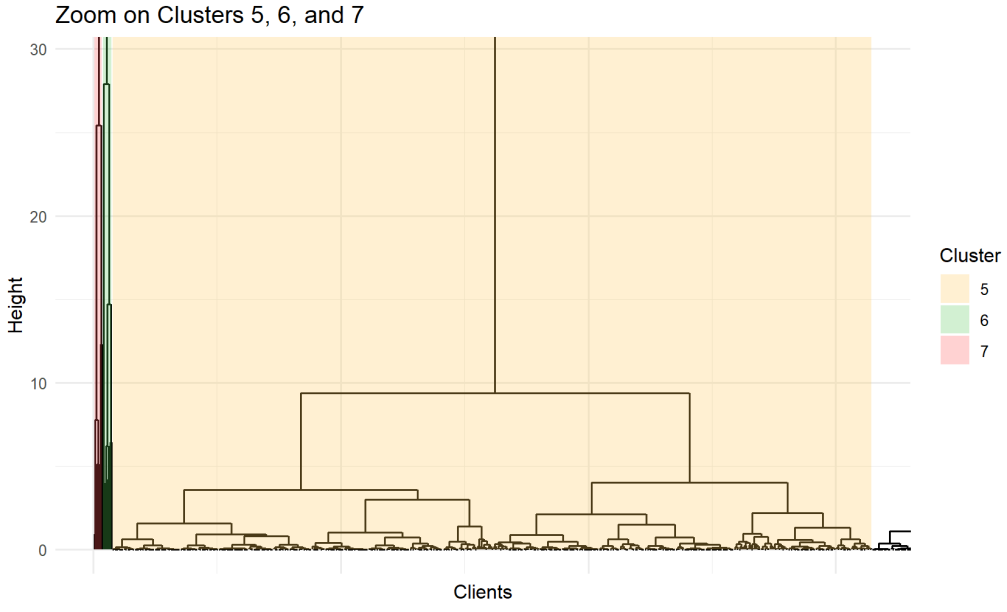


Figure 4.7: Hierarchical dendrogram

The visibility of clusters 6 and 7 is limited in the full dendrogram. A zoomed-in view of clusters 5, 6, and 7, shown in Figure 4.8, provides a clearer view.



!h

Figure 4.8: Cluster 5,6,7 zoomed in (left side of the previous dendogram)

Cluster Statistics: The hierarchical clustering results are summarized in Table 4.4.

Clust	R Mean	R Median	F Mean	F Median	M Mean	M Median	Size
1	12.6	4.95	24.4	21	14514	8277	146
2	23.5	16.9	8.08	7	3174	2473	857
3	44.5	39.1	2.34	2	851	601	2052
4	291.0	284.0	1.35	1	439	297	654
5	163.0	163.0	1.83	1	587	376	614
6	11.3	6.04	43.9	50.5	164658	146694	8
7	2.12	1.87	129.0	97.0	51640	40992	7

Table 4.4: Cluster Statistics for Hierarchical Clustering.

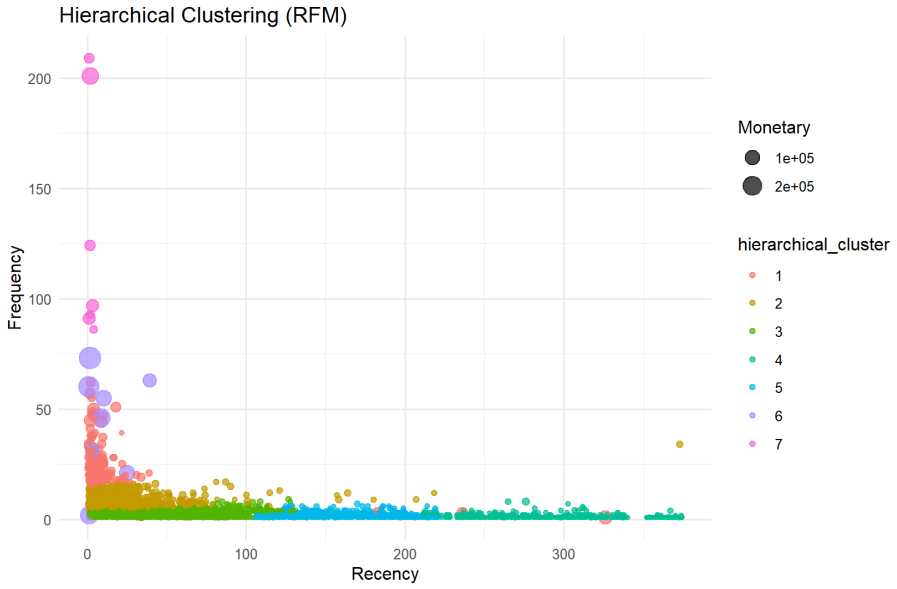


Figure 4.9: Hierarchical Clustering on RFM Data

Cluster Insights:

- **Cluster 6 and 7:** These clusters are the biggest contributors in monetary terms. Cluster 6 is marked with the highest monetary mean of 164,658 € and eight purchases. However, Cluster 7 displaying a high monetary mean of 51,640€ has only seven customers.
- **Clusters 1 and 2:** This cluster records medium-value customers. Cluster 1 contains high frequency and also, mainly recent activity while the one presenting moderate frequency and monetary values is Cluster 2.
- **Clusters 3, 4, and 5:** These clusters characterize low or dormant customers. At first glance, it looks like Cluster 4 is the most interesting, as it contains the customers with the least recent activity (*Recency Mean = 291 days*) and they should be the segment to apply re-engagement strategies.

Hierarchical clustering was effective in the grouping of customers that showed different

patterns in behavior. The problem was solved through the manual selection of the seven clusters, which provided granularity and reliability. Dendrogram's subjective interpretation is the prime concern in this method that may lead to biases. The segmentation of clustering that can be beneficial in personalizing the approaches to customers, particularly in those segments with the highest value like Clusters 6 and 7.

Gaussian Mixture Model (GMM) Clustering Results

Gaussian Mixure Modle (GMM) was the normailzed RFM dataset implemented by its probalistic approach The Gaussian Mixture Model(GMM was the method applied to the normalized RFM dataset) for the identification of clusters.GMM, which is different from k-means and DBSCAN, is based on the idea that the data arises from a mixture of certain types of Gaussian distributions. Therefore, it allows for displaying different cluster shapes and densities independently.

Cluster Statistics: GMM identified nine clusters in the RFM data, each characterized by distinct behavioral patterns. The statistics for each cluster are summarized in Table 4.5.

Clust	R Mean	R Median	F Mean	F Median	M Mean	M Median	Size
1	15.8	15.2	4.93	4	1217	1107	748
2	47.7	44.9	1.40	1	379	336	1113
3	176.0	176.0	1.34	1	314	290	406
4	20.0	12.1	15.0	14	8134	6902	240
5	23.0	19.9	8.35	8	3275	3191	399
6	281.0	278.0	1.15	1	295	255	424
7	108.0	86.0	3.53	3	1359	1214	838
8	367.0	367.0	1.00	1	236	208	108
9	65.2	11.4	36.3	21.5	46833	30358	62

Table 4.5: Cluster Statistics for GMM Clustering.

Insights:

- **Cluster 9:** The standout cluster, contributing significantly to the monetary value with an average monetary mean of 46833€ and a group size of just 62 customers. These high-value customers could be prioritized for retention and upselling strategies.
- **Clusters 4 and 5:** These clusters represent moderately high-value customers with notable transactional frequency. Cluster 4, for instance, has an average frequency of 15.0 and a monetary mean of 8134€, indicating engaged and valuable customers.

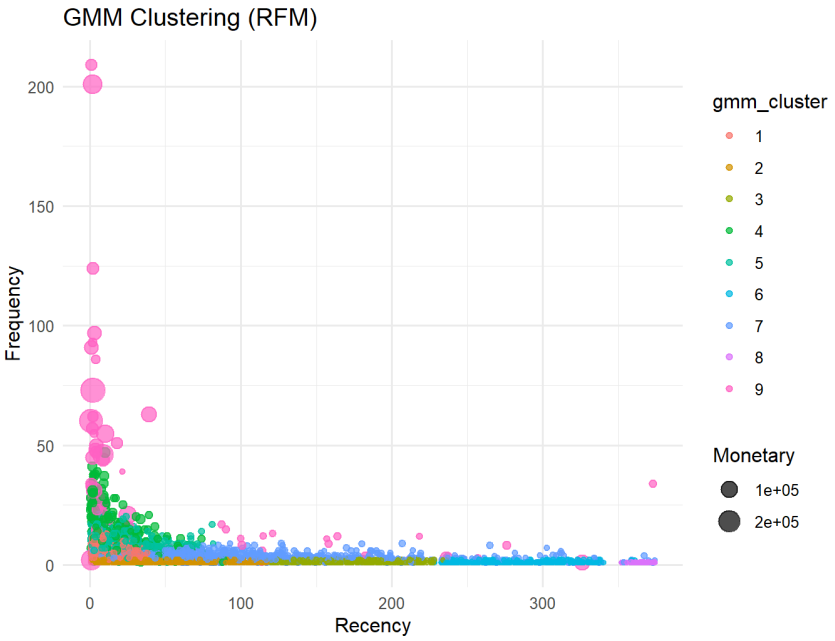


Figure 4.10: GMM clustering results (RFM)

- **Clusters 1, 7, and 2:** These clusters encompass medium to low-value customers. Cluster 7 shows a slightly higher frequency compared to clusters 1 and 2, but their monetary contributions remain relatively modest.

- **Clusters 3, 6, and 8:** These represent low-value segments with sparse participation and minimal monetary contributions. Cluster 8 stands out due to its very high recency value (*Recency Mean = 367 days*), suggesting customers who have been inactive for a long period. Cluster 8 probably is grouping whose that can be consider as "Lost Client"

The probabilistic nature of the GMM allowed for the identification of nuanced patterns in customer behavior. However, the high number of clusters (9) raises questions about the interpretability of results and potential overfitting. While clusters like 9 and 8 provide clear idea in what they are clustering, other clusters, such as 2 - 3 and 6 , may require additional analysis to confirm their relevance. GMM demonstrates robust flexibility, but further validation against other clustering techniques or business objectives is essential to refine these insights.

Fuzzy C-Means Clustering

Fuzzy C-Means (FCM) clustering does not simply leverage the advantages of traditional methods (like k means) but also brings in additional ones through the use of the membership of data points to multiple clusters with different levels of density. The advantages of this algorithm become particularly evident in case of ambiguous edges that do not distinctly separate the clusters.

Optimal number of Clusters The use of the k value derived from the Elbow Method ($k = 4$) in K-Means clustering to be used in Fuzzy C-Means clustering is influenced by the similarities between the two techniques. Like K-Means, Fuzzy C-Means is also a centroid-based clustering technique since it divides a dataset into k clusters through the minimization of distances from the data points to the individual cluster centroids. This shared approach makes the k value, which is determined by the Elbow plot in K-Means, a good and acceptable choice for the Fuzzy C-Means application.

Fuzzy Membership Visualization In order to demonstrate the distinctness of Fuzzy C-Means clustering, we chose a heatmap as our visualization method. Fig. 4.11 is the one which included this heatmap that visualizes these memberships for individual customers across clusters. To avoid the visualization muddle which is normally caused by the large dataset, we used a random sample of customers. This not only helped us ensure clarity and interpretability but also and most importantly the randomness highlighted the issue of membership ambiguity due to Fuzzy C-Means clustering. In fact, Fuzzy C-Means does not group customers in one cluster that is characteristic of Hard C-Means, but rather, they have several degrees of membership across different clusters. The discerned clusters are shown on the heatmap dependent on the colors which darken in intensity with a higher degree of membership, thus providing an indication of the overlaps and fuzziness between the clusters with the colors.

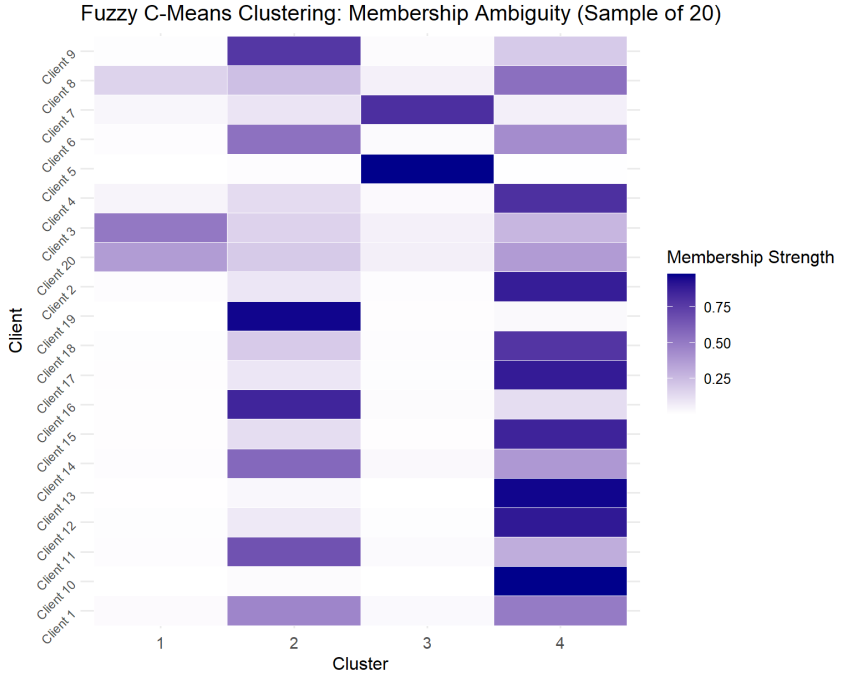


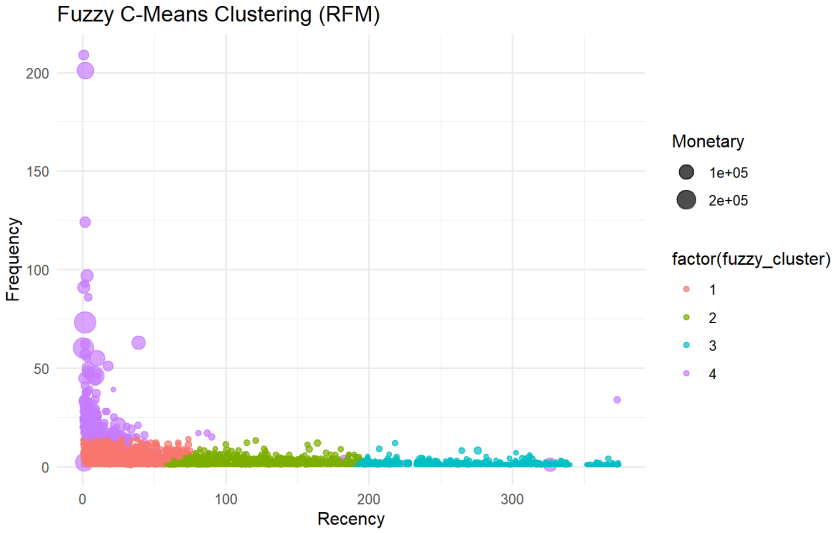
Figure 4.11: Heatmap of Fuzzy Memberships (Sampled Data)

Cluster Characteristics The table 4.6 shown summarizes the characteristics of the four clusters identified by FCM.

Clust	R Mean	R Median	F Mean	F Median	M Mean	M Median	Size
1	25.9	23.0	4.12	3	1508	981	2204
2	112.0	104.0	2.22	2	807	512	1109
3	14.4	5.97	25.1	19	18336	7937	234
4	276.0	269.0	1.40	1	509	302	791

Table 4.6: Cluster Statistics for Fuzzy C-Means Clustering.

The figure 4.12 visualizes the FCM results. Each point is assigned to the cluster with the highest membership value (hard assignment).



!h

Figure 4.12: Fuzzy C-Means Clustering (Hard Assignments)

Insights:

- **Cluster 1:** This is the biggest group of customers consisting of relatively average Recency and low Frequency and Monetary values, thus, customers are moderately engaged.
- **Cluster 2:** Less frequent and less recent buyers with low Monetary contributions make up this cluster, which possibly both represent disengaged or dormant customers.
- **Cluster 3:** The existence of high-value customers who perform transactions frequently and have a low Recency fits the description of highly engaged and valuable clients.
- **Cluster 4:** Excessive absenteeism characterizes customer engagement this cluster possesses with extremely low Frequency and Monetary values, usually indicating a long time to inactive clients.

RFM segmentation and clustering processes covered end here. Moving on, the next section will be dedicated to CLV-based segmentation that allows implementation of the same clustering methods. The explanation of the procedures used to obtain the results has been comprehensively provided in the previous subsection; hence, the focus will now be on the interpretation and comments on the results.

4.2 CLV Segmentation & Clustering

4.2.1 CLV Segmentation

This section is a transition from RFM-based methodology to a focus on **Customer Lifetime Value (CLV)**. We aim at getting a more holistic assessment of every customer's long-term value by including factors like *TotalRevenue*, *Frequency*, *Average Transaction Value*, *Lifespan*, and a derived *CLV* metric. The following subsections detail the clustering analysis performed on these CLV-driven features, providing insights into customer segments that can guide strategic marketing, retention, and acquisition efforts.

4.2.2 Clustering Results on CLV Segmentation

K-Means Clustering Results

The K-Means algorithm was applied to the normalized dataset which had *TotalRevenue*, *Frequency*, *Average Transaction Value*, *Lifespan*, and *CLV*. The best number of clusters k was determined using the **Elbow Method** as in the RFM analysis.

Elbow Method for Optimal k: Figure 4.13 illustrates the Elbow Method, where the Within Sum of Squares (WSS) was plotted against the number of clusters. It is noticeable a "elbow" at $k = 4$ that indicates that adding more clusters is inefficient, causing just marginal improvements in variance reduction. Hence, the selected optimal solution was $k = 4$.

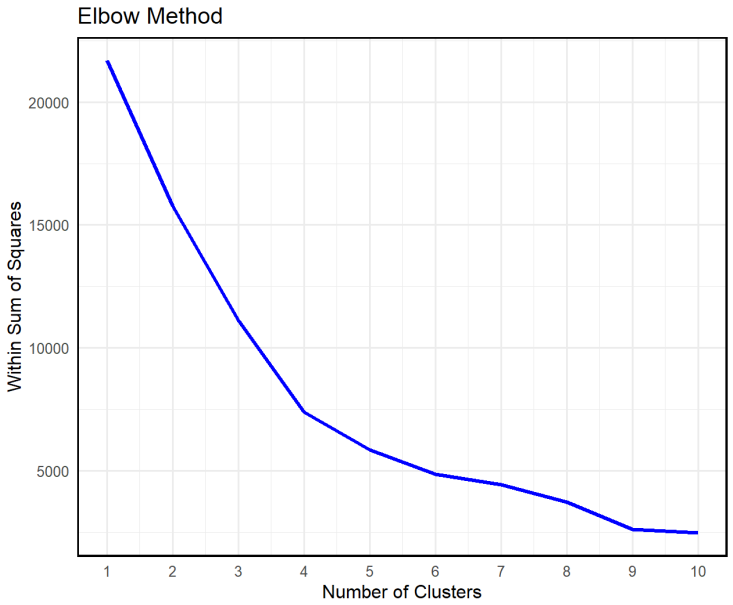


Figure 4.13: Elbow Method for K-Means (CLV Data)

Cluster Statistics: The K-Means algorithm thus partitions the customer base into four clusters, each exhibiting distinct behaviors in terms of revenue generation, purchase frequency, and lifetime value. Table 4.7 provides an overview of the key metrics for each cluster.

kC	TR M	F M	CLV M	AT M	LS M	Size
1	2915	7.08	70,932	33.3	274	1753
2	122,828	1.50	11,512,288	66,671	102	2
3	84,711	72.30	3,396,779	192	363	23
4	628	1.74	2,369	39.2	30.9	2560

Table 4.7: K-Means Cluster Statistics for CLV-Based Analysis

Legend:
kC K-Means Clustering
TR Total Revenue (€)
F Frequency
CLV Customer Lifetime Value
AT Average Transaction Value
LS Lifespan (days)
M Mean

Visualization: Figure 4.14 presents a 2D scatter plot of the clusters in terms of *Frequency* and *TotalRevenue*, with point size reflecting the *CLV* magnitude. Additionally, Figure 4.15 shows the radar chart of K-Means. A radar chart is a graphical method used to visualize multivariate data. Each variable is represented by an axis starting from the center, and the data points are plotted on these axes to form a polygon. In this context, radar charts display the normalized averages of key metrics (e.g., Total Revenue, Frequency, CLV) for each cluster, offering a compact

view of their unique characteristics.

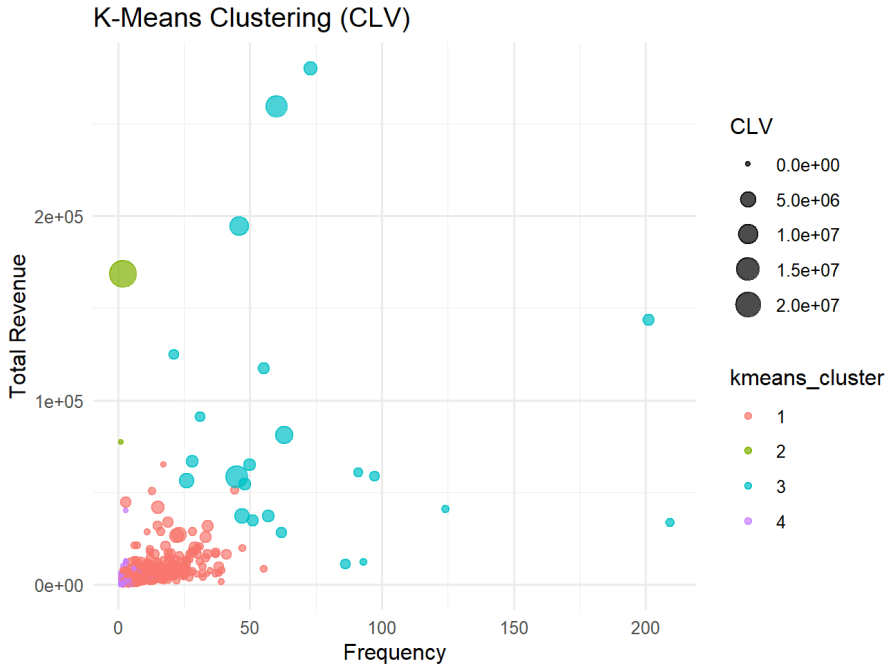


Figure 4.14: K-Means Clustering on CLV Data (2D Projection)

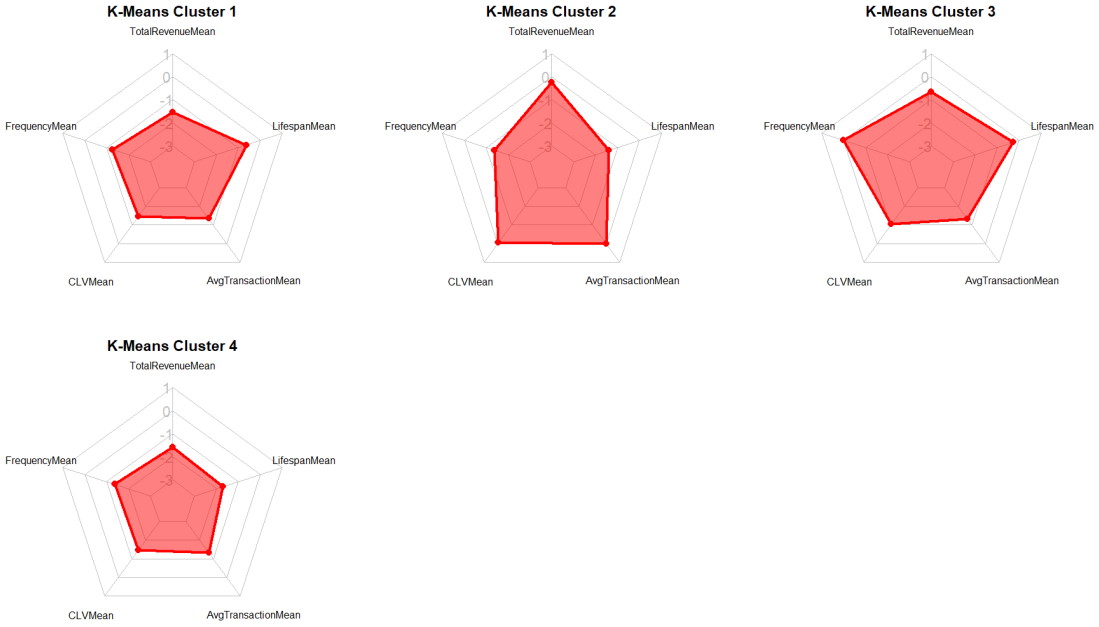


Figure 4.15: K-Means Radar Chart

Insights:

- **Cluster 2** (only 2 customers) exhibits extremely high *TotalRevenue* and *AvgTransactionValue*, resulting in the largest *CLV* by far. These represent "ultra-premium" clients, where personalized retention and upselling strategies can be highly impactful.

- **Cluster 3** stands out for its elevated *Frequency* (over 70 purchases on average) and a high *CLV*, suggesting a loyal and active customer group. Fostering loyalty programs and subscription models could further cement these relationships.

- **Cluster 1** contains moderately active customers with a notable lifetime value, though significantly lower than Cluster 2 or 3. Targeted campaigns can aim to increase their purchase frequency or upsell to boost their *AvgTransactionValue*.

- **Cluster 4** holds the largest volume of customers (2,560), but with low *TotalRevenue*, *CLV*, and *Frequency*. It could encompass one-time or infrequent buyers. Re-engagement or cross-selling strategies may help convert a portion of this large group into higher-value segments.

Having outlined the CLV-based K-Means clustering, the following subsections will compare these findings with alternative clustering methods (Hierarchical, DBSCAN, GMM, and Fuzzy C-Means) to further validate or refine the segmentation strategy.

Hierarchical Clustering Results

The Hierarchical Clustering approach was applied to the same CLV-based dataset using Ward's minimum variance method (`ward.D2`). This algorithm recursively merges clusters to minimize the total within-cluster variance, producing a dendrogram that illustrates the nested structure of the data.

Determination of the Number of Clusters: The cutoff height was selected through a visual analysis of the dendrogram (Figure 4.16), thus resulting in $\boldsymbol{k = 4}$ clusters. The rectangular boundaries drawn on the dendrogram confirm the selection, by balancing the ease of interpretation and the detail level of the clusters.

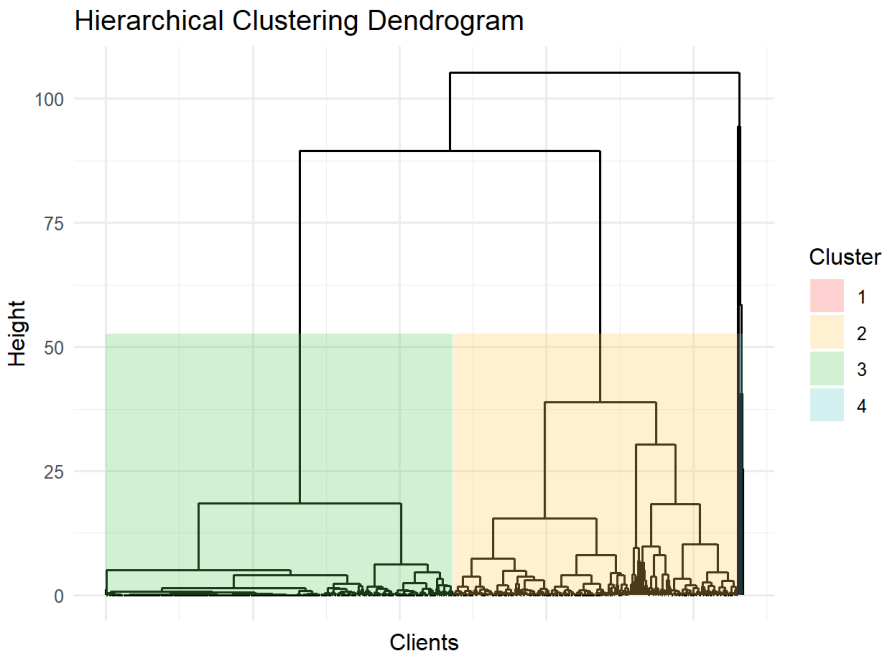


Figure 4.16: Hierarchical Clustering Dendrogram (CLV Data)

Cluster Statistics (Means): Table 4.8 provides an overview of each cluster's mean values for the key variables. The abbreviations are explained in the legend below the table.

hC	TR M	F M	CLV M	AT M	LS M	Size
1	122,828	1.50	11,512,288	66,671	102	2
2	2,586	6.63	58,597	29.7	261	1,941
3	571	1.63	1,637	34.5	21.6	2,364
4	74,050	58.2	2,933,135	772	329	31

Table 4.8: Hierarchical Clustering (CLV) – Mean Values.

Legend: **hC** Hierarchical Cluster

Cluster Statistics (Standard Deviations): To further assess variability within each cluster, Table 4.9 shows the corresponding standard deviations for the same metrics.

hC	TR SD	F SD	CLV SD	AT SD	LS SD
1	64,551	0.71	16,280,833	14,868	145
2	3,304	5.80	137,274	75.4	72.7
3	747	1.07	7,167	129	36.4
4	65,367	48.8	3,529,243	2,458	91.2

Table 4.9: Hierarchical Clustering (CLV) – Standard Deviations.

Visualization: Figure 4.17 shows a 2D projection of the four hierarchical clusters (color-coded by cluster membership), plotted against *Frequency* and *TotalRevenue*, with the point size reflecting the *CLV*. Additionally, Figure 4.18 shows the radar charts of the Hierarchical Clustering

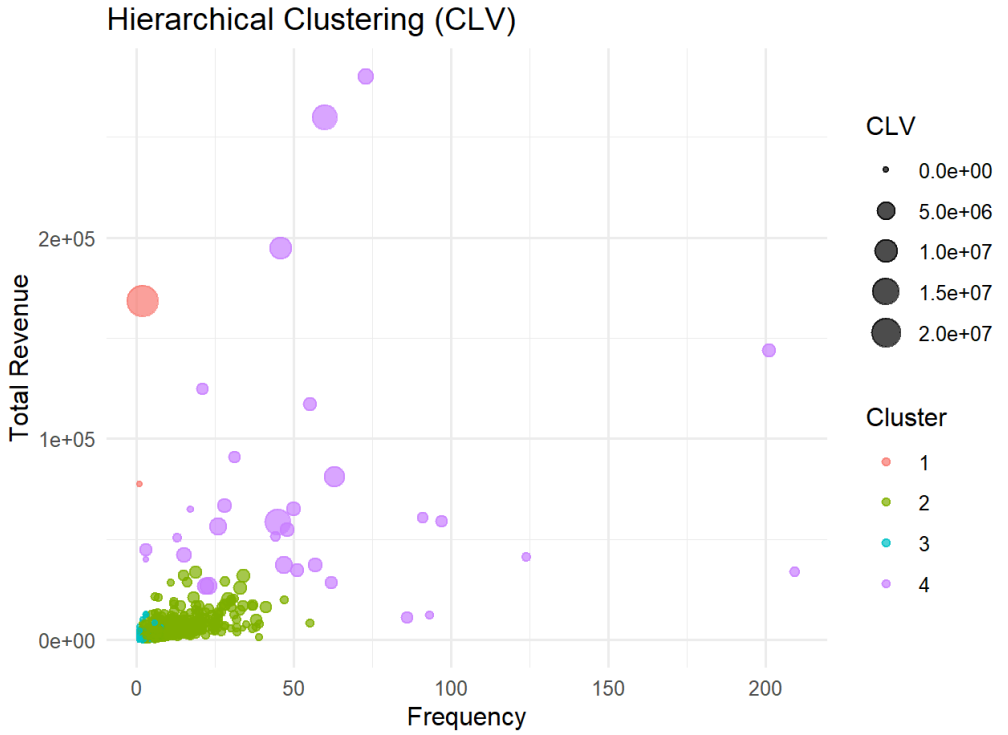


Figure 4.17: Hierarchical Clustering on CLV Data (2D Projection)

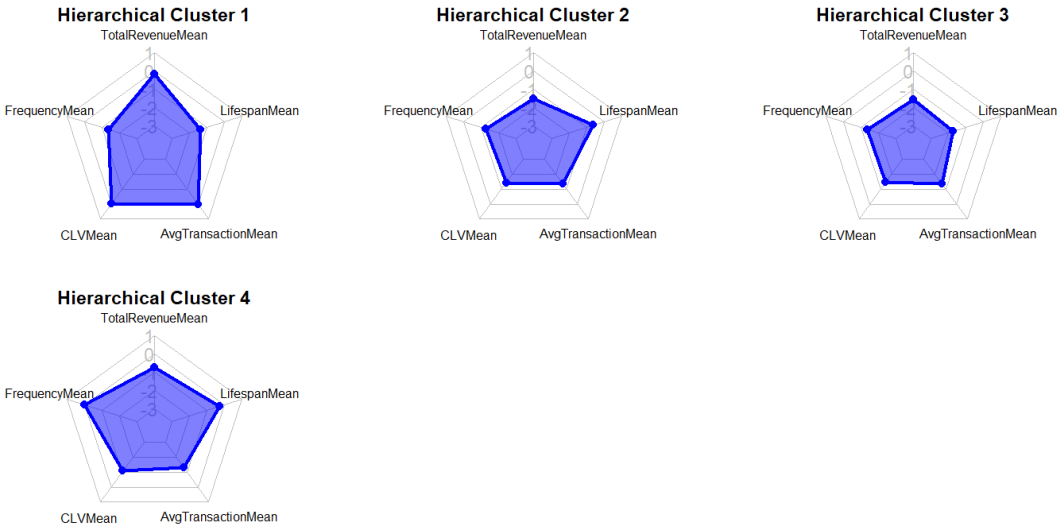


Figure 4.18: hC Radar Charts

Insights:

- **hC 1:** With only *2* customers, it shows extremely high *TotalRevenue* and *CLV*, similar to the "ultra-premium" cluster identified in K-Means. Retention and bespoke marketing can amplify the value of these elite clients.
- **hC 2:** The largest cluster (1,941 customers) with moderate revenue and frequency. Although their *CLV* is relatively modest, an upsell or cross-sell approach could unlock further potential.
- **hC 3:** A large group of low-value and infrequent buyers. This segment may be harder to convert, but re-engagement campaigns or targeted promotions might reactivate some portion.
- **hC 4:** A niche but very active cluster ($F\ Mean \approx 58.2$). Despite generating substantial revenue ($\approx$ 74,050), they remain far behind the top-tier cluster in terms of *CLV*. Strengthening loyalty programs could increase their average transaction value.

These findings mirror certain patterns seen in the K-Means segmentation, albeit with variations in cluster size and boundaries. The manual selection of four clusters introduces a degree of subjectivity, yet the hierarchical approach offers a more intuitive view of segment cohesion and separation via the dendrogram.

DBSCAN Clustering Results

The **DBSCAN** algorithm takes a density-based approach, identifying dense regions in the CLV feature space while designating low-density points as noise. Unlike centroid-based methods, DBSCAN automatically determines the number of clusters based on the parameters `eps` and `MinPts`.

Parameter Selection: A *k-NN distance plot* was generated to guide the choice of `eps`. After observing in the figure 4.19 a sharp increase in the distance values near `eps = 0.6`, this threshold was adopted, with `MinPts` set to 5. The algorithm yielded **2** main clusters in the context of CLV data.

Cluster Statistics (Means): Table 4.10 provides the mean values for each cluster, employing the same notation used previously.

dbC	TR M	F M	CLV M	AT M	LS M	Size
0	39,506	35.2	1,746,547	2,184	296	81
1	1,342	3.68	21,162	28.1	128	4,257

Table 4.10: DBSCAN Clustering (CLV) – Mean Values.

Legend: **dbC** DBSCAN Cluster

Cluster Statistics (Standard Deviations): Table 4.11 reports the standard deviations for each metric, indicating the dispersion within each cluster.

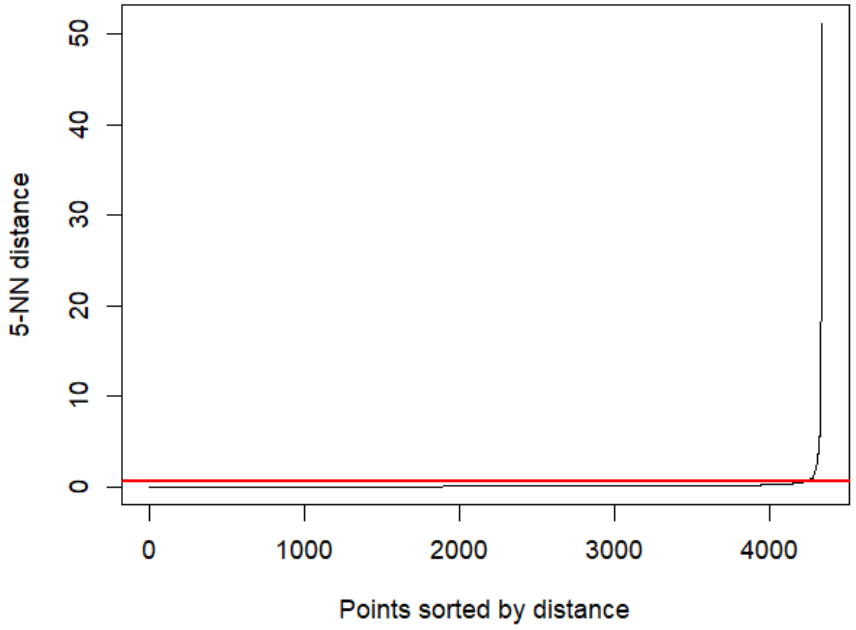


Figure 4.19: k-NN Distance Plot for DBSCAN Parameters Selection (`eps = 0.6`)

dbC	TR SD	F SD	CLV SD	AT SD	LS SD
0	52,305	36.9	3,448,736	10,585	113
1	1,928	4.03	56,020	53.4	131

Table 4.11: DBSCAN Clustering (CLV) – Standard Deviations.

Visualization: The 2D plot in Figure 4.20 illustrates the clusters in terms of *Frequency* (x-axis) and *TotalRevenue* (y-axis), with the size of each point corresponding to its *CLV*.

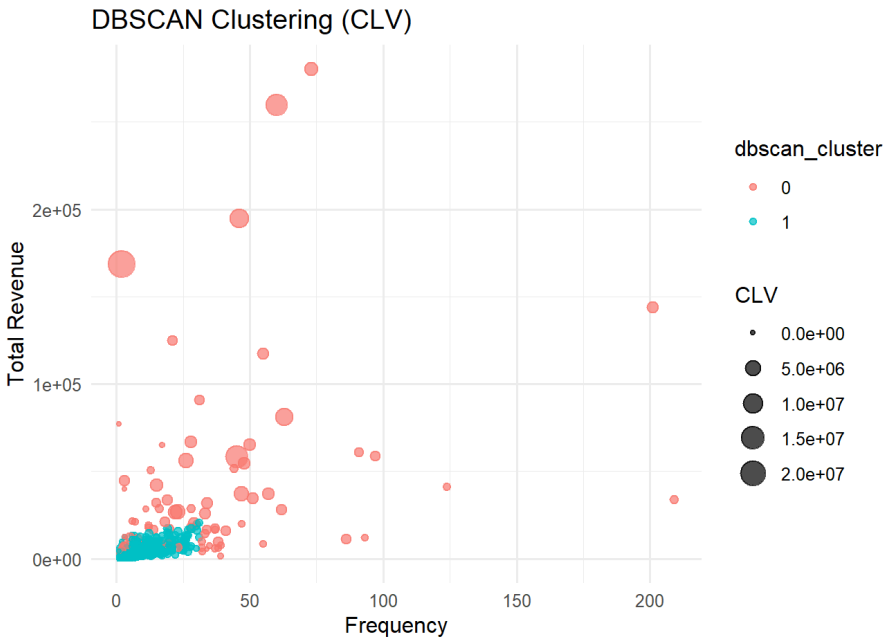


Figure 4.20: DBSCAN Clustering on CLV Data (2D Projection)

Insights:

- **Cluster 0**: Comprises only 81 customers but shows significantly higher *Frequency*, *TotalRevenue*, and *CLV* on average. The standard deviations (**TR SD** and **CLV SD** in particular) are notably large, indicating a broad range of spending patterns within this high-value segment.
- **Cluster 1**: The vast majority of customers (over 4,000), marked by modest revenue and low frequency. Although their *CLV* remains comparatively small, targeted marketing campaigns could potentially lift a subset of these customers into higher-value brackets.
- **Noise Points**: DBSCAN typically designates isolated or less dense regions as noise. In this dataset, however, most customers fall into one of the two clusters, suggesting that the chosen parameters effectively captured the primary data structure.

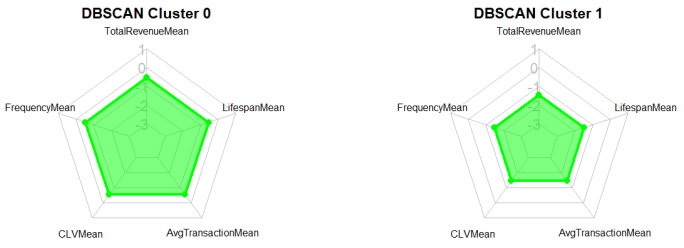


Figure 4.21: DBSCAN Radar Charts

In summary, DBSCAN identifies a small but highly valuable group of buyers (Cluster 0) alongside a large, lower-value segment (Cluster 1). The elevated **SD** metrics in Cluster 0 indicate diverse spending habits, meriting a closer look at sub-segmentation or personalized offers for high-spending individuals.

Gaussian Mixture Model (GMM) Clustering Results

The **Gaussian Mixture Model (GMM)** is a statistical method based on probabilities which relies on the fact that each data point may come from a mixture of Gaussian distributions. This inherent flexibility of GMM allows it to cluster the data in different shapes and densities, which in many cases can lead to more accurate segmentations than the solely distance-based methods.

Model Fitting: In this analysis, the `Mclust` package was utilized to automatically determine the optimal number of components through the Bayesian Information Criterion (BIC). The resulting best-fit model identified **6** clusters in the CLV-based feature space.

Cluster Statistics (Means): Table 4.12 presents mean values for each of the six GMM clusters, using the abbreviated notation described in earlier sections.

gC	TR M	F M	CLV M	AT M	LS M	Size
1	304	1.00	0	19.6	0	1,393
2	35,999	26.1	1,750,038	2,513	208	75
3	643	2.00	3,721	17.6	104	730
4	5,705	10.5	156,058	102	217	333
5	1,211	4.60	12,095	13.4	204	1,083
6	2,911	7.23	44,922	27.1	252	724

Table 4.12: GMM Clustering (CLV) – Mean Values.

Legend: **gC** GMM Cluster

Cluster Statistics (Standard Deviations): Table 4.13 displays the standard deviations, indicating how dispersed each metric is within every cluster. Notably, *gC 2* has a high **CLV SD**, reflecting a broad range of purchase patterns among these otherwise high-value customers.

gC	TR SD	F SD	CLV SD	AT SD	LS SD
1	224	0	0	14.9	0
2	56,132	39.6	3,606,250	10,957	158
3	512	0	3,673	8.88	85.9
4	7,045	11.8	183,436	79.3	145
5	723	2.43	9,001	6.28	97.5
6	1,711	4.44	32,664	13.0	102

Table 4.13: GMM Clustering (CLV) – Standard Deviations.

Visualization: Figure 4.22 illustrates the GMM clustering in a 2D plane (*Frequency* vs. *TotalRevenue*).

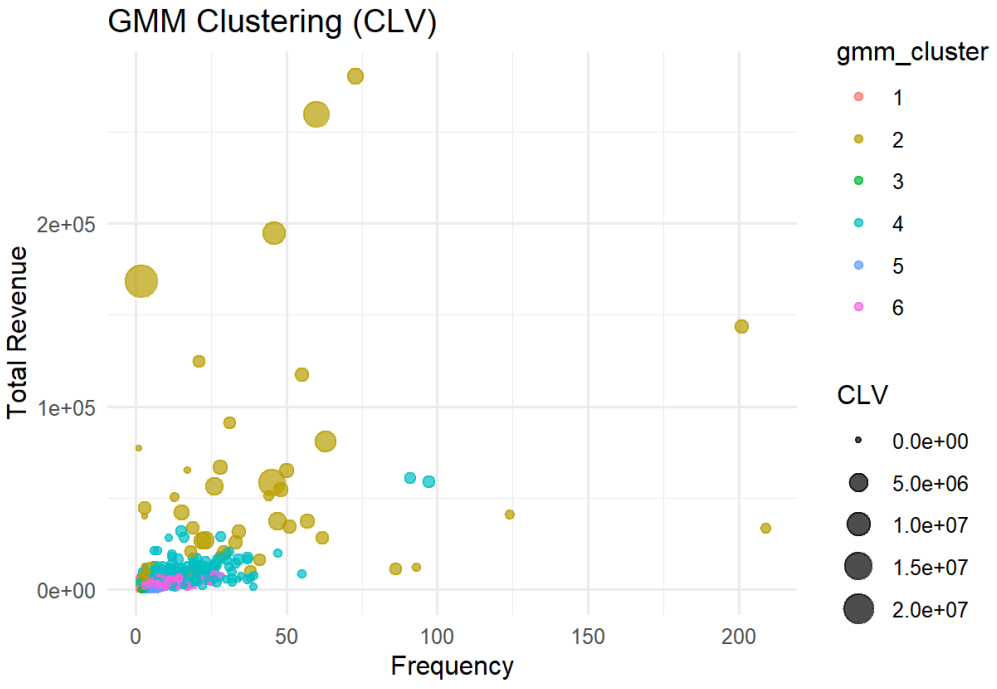


Figure 4.22: GMM Clustering on CLV Data (2D Projection)

Figure 4.23: GMM Radar Char

Insights:

- **gC 1:** A large, almost trivial cluster (1,393 customers) with minimal *TotalRevenue* and *CLV*—likely representing buyers who made only one low-value purchase.
- **gC 2:** Features a modest group of high-frequency, high-revenue customers. The **CLV SD** is very large, indicating significant variability even among these top spenders.
- **gC 4:** A moderately sized cluster with notable revenue and purchase frequency, though still dwarfed by gC 2 in terms of *CLV*.
- **gC 6:** Occupies a middle ground, with moderate *Frequency*, *TotalRevenue*, and *CLV*.

In general, GMM exhibits various levels of customer value and customer behavior, looking at it is more detailed than simple two or four-cluster alternatives. But the larger number of clusters might be an impediment to decision making hence it is necessary to match segmentation granularity with business goals.

Fuzzy C-Means (FCM) Clustering Results

Fuzzy C-Means (FCM) is a clustering technique in which each point belongs to multiple clusters called the degrees of the membership. This method is quite revealing when the customer baselines are not clearly drawn thus, it allows the segments to flow between each other.

Optimal Number of Clusters: Consistent with the K-Means analysis, we set $\boldsymbol{k = 4}$ for FCM. This choice stems from the *Elbow Method* determination and the conceptual similarity between K-Means and FCM in partitioning data around k centroids.

Cluster Statistics (Means): Table 4.14 summarizes the mean values for each FCM cluster, following the abbreviations used throughout this section.

fC	TR M	F M	CLV M	AT M	LS M	Size
1	602	1.66	1,878	40.2	23.2	2,404
2	10,827	19.7	316,841	292	335	333
3	169,464	70.0	10,311,524	8,381	337	7
4	1,677	4.70	30,963	27.6	249	1,594

Table 4.14: Fuzzy C-Means Clustering (CLV) – Mean Values.

Legend: **fC** Fuzzy C-Means Cluster

Cluster Statistics (Standard Deviations): Table 4.15 shows the standard deviations. Notably, *fC 3* exhibits extremely high **TR SD** and **CLV SD**, indicating that even within this very small but ultra-premium cluster, there is still substantial variation in spending patterns.

fC	TR SD	F SD	CLV SD	AT SD	LS SD
1	1,134	1.13	7,916	300	38.4
2	15,068	17.0	590,130	4,234	49.1
3	83,485	62.1	7,211,349	21,069	59.7
4	1,386	2.31	65,330	78.4	66.3

Table 4.15: Fuzzy C-Means Clustering (CLV) – Standard Deviations.

Visualization: Because each point belongs to multiple clusters with different membership degrees, a single 2D plot may not fully convey the "fuzziness." Nevertheless, Figure 4.24 plots customers based on their *hard assignments* (i.e., the cluster to which they have the highest membership).

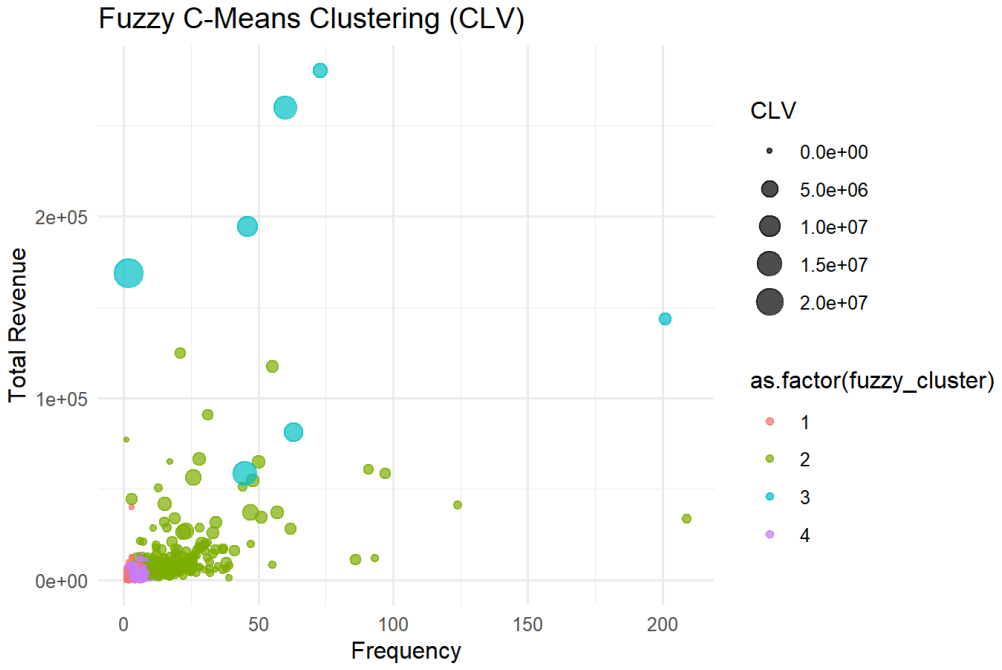


Figure 4.24: Fuzzy C-Means Clustering on CLV Data (Hard Assignments)

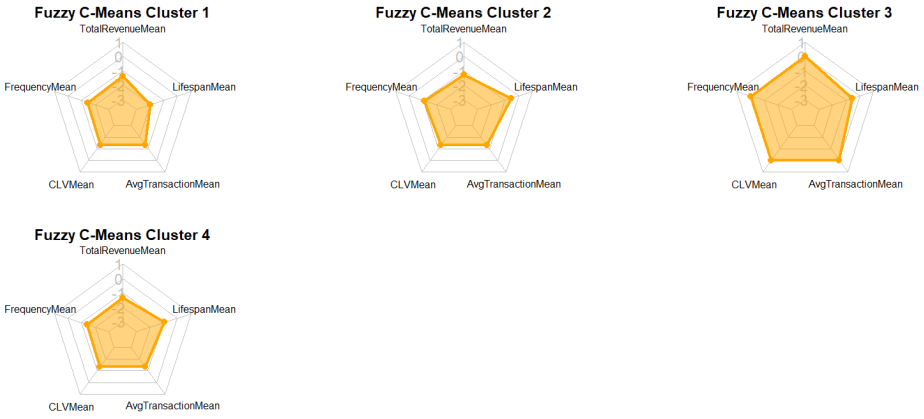


Figure 4.25: fC Radar Charts

Insights:

- **fC 3:** Contains only **7 customers**, each with extremely high *CLV* and *Average Transaction Value*. Although its size is minuscule, the **TR SD** and **CLV SD** reveal that this elite group can still be quite heterogeneous in its purchasing behavior.
- **fC 2:** Represents a moderately sized segment (333 customers) with robust frequency and total revenue, indicating recurring and sizable purchases over a substantial lifespan.
- **fC 1 & 4:** Cover the majority of the customer base (with a combined size of

nearly 4,000), displaying relatively modest spending habits and lower frequencies. However, membership degrees in FCM could identify "bridge" customers on the cusp of higher-value segments.

FCM offers a more nuanced view for understanding how customers transition between segments. Such granularity can be particularly valuable for designing tailored promotions or loyalty programs, where partial affinities to premium clusters may signal a high potential for up-selling or cross-selling interventions.

4.3 Models Evalutation

After deriving clusters from five different algorithms (*K-Means, Hierarchical, DBSCAN, GMM, Fuzzy C-Means*) for both the **RFM** and **CLV** data, we evaluated the solutions using two internal validation metrics:

- **Silhouette Score**
- **Calinski–Harabasz (CH) Index**

Each metric offers a distinct perspective on cluster quality, with Silhouette focusing on cohesion and separation at the data-point level, and the CH index measuring the ratio of between-cluster to within-cluster variance.

4.3.1 Evaluation on RFM Data

Tables 4.16 and 4.17 present the Silhouette scores and Calinski–Harabasz indices, respectively, for the RFM-based clustering solutions.

Method	**Silhouette Score**
K-Means	0.6161
DBSCAN	0.6904
Hierarchical	0.4329
GMM	0.1583
Fuzzy C-Means	0.4167

Table 4.16: Silhouette Scores for RFM Clustering

Method	**Calinski–Harabasz Index**
K-Means	902.28
DBSCAN	562.38
Hierarchical	502.99
GMM	272.46
Fuzzy C-Means	601.39

Table 4.17: Calinski–Harabasz Indices for RFM Clustering

Observations for RFM:

- **DBSCAN** achieves the highest Silhouette score (0.6904), suggesting that at a local (density) level, it forms cohesive and well-separated clusters for the RFM data.
- **K-Means** presents the highest Calinski–Harabasz index (≈ 902.28), indicating strong global separation among clusters relative to their internal variance.
- **GMM** shows relatively poor performance on both metrics, implying that the RFM distribution may not align well with Gaussian assumptions.
- **Fuzzy C-Means** and **Hierarchical** yield intermediate results. Their Silhouette and CH scores suggest moderately coherent clusters, yet not as distinctly separated as with K-Means or DBSCAN.
- The discrepancy between Silhouette and CH for DBSCAN vs. K-Means illustrates how density-based methods can excel in local separation (Silhouette), while centroid-based approaches might yield better global variance ratios (CH).

4.3.2 Evaluation on CLV Data

Tables 4.18 and 4.19 show the Silhouette scores and Calinski–Harabasz indices for the CLV-based clustering methods.

Method	Silhouette Score
K-Means	0.6226
Hierarchical	0.6212
DBSCAN	0.8372
GMM	0.2012
Fuzzy C-Means	0.6173

Table 4.18: Silhouette Scores for CLV Clustering

Method	Calinski–Harabasz Index
K-Means	989.23
Hierarchical	994.80
DBSCAN	889.47
GMM	260.48
Fuzzy C-Means	833.93

Table 4.19: Calinski–Harabasz Indices for CLV Clustering

Observations for CLV:

- **DBSCAN** attains the highest Silhouette score (0.8372), indicating that it identifies highly cohesive and well-separated dense regions in the CLV feature

space.

- **Hierarchical** slightly outperforms K-Means on the CH index (994.80 vs. 989.23), suggesting that it offers marginally better global separation for these five-dimensional features (*TotalRevenue, Frequency, AverageTransactionValue, Lifespan, CLV*).
- **GMM** again shows lower performance on both metrics (0.2012 Silhouette, 260.48 CH), implying that a purely Gaussian mixture approach may not capture the irregular patterns of CLV distributions.
- **Fuzzy C-Means** and **K-Means** demonstrate comparably moderate Silhouette scores (around 0.62), but with K-Means performing better on the CH index. This dynamic again underscores how certain algorithms may excel in local cohesion yet differ in global variance partitioning.

Summary of Model Evaluation

	RFM Clustering		CLV Clustering	
Method	**S Score**	**CH Index**	**S Score**	**CH Index**
K-Means	0.6161	902.28	0.6226	989.23
DBSCAN	0.6904	562.38	0.8372	889.47
Hierarchical	0.4329	502.99	0.6212	994.80
GMM	0.1583	272.46	0.2012	260.48
Fuzzy C-Means	0.4167	601.39	0.6173	833.93

Table 4.20: Clustering Evaluation for RFM and CLV Data

- **DBSCAN** has consistently proved its ability to achieve high Silhouette scores in both RFM and CLV datasets, indicating well-defined clusters in terms of local density. However, its CH scores, though decent, fall short of those of K-Means or Hierarchical in some cases.
- **K-Means** and **Hierarchical** often dominate the **Calinski–Harabasz** index, suggesting they produce broader inter-cluster separation relative to intra-cluster variance. Hierarchical clustering shows a slight edge over K-Means for the CLV data's CH index.
- **GMM** has consistently performed below the rest of the techniques in both metrics, because of issues related to the choice of the features (RFM or CLV) that are distributed non-Gaussian.
- **Fuzzy C-Means** performs moderately in both metrics, especially for CLV. Overall it seems a worse version of the K means.

In conclusion, the choice of clustering method depends heavily on whether the primary goal is *local cohesion* vs. *global separation*. For local, density-driven segmentation (as indicated by Silhouette), DBSCAN frequently emerges as the best candidate.

For maximizing between-cluster variance (as indicated by the Calinski–Harabasz index), K-Means or Hierarchical may be preferred. The next chapter discusses these findings in detail and outlines potential use-cases and future directions for applying the various clustering methodologies.

Chapter 5

Conclusions

Overall, this thesis has comparatively analyzed customer segmentation strategies based on the **RFM (Recency, Frequency, Monetary)** and **CLV (Customer Lifetime Value)** models, integrating them with five distinct clustering algorithms: *K-Means*, *Hierarchical Clustering*, *DBSCAN*, *Gaussian Mixture Models (GMM)* and *Fuzzy C-Means (FCM)*. The aim was to thoroughly investigate how different segmentation techniques can provide useful indications on both a tactical and strategic basis, highlighting their respective strengths and weaknesses. The results obtained provide significant considerations both from a methodological and a practical-applicative point of view.

1. Summary of Objectives and Context

The thesis is placed in the broader panorama of *Customer Relationship Management* (CRM) and *data-driven marketing*, where customer segmentation plays a central role in optimizing resources and maximizing the return on investment of campaigns. In particular, the use of the **RFM** model allows to interpret purchasing behaviors in terms of temporal proximity (*Recency*), transaction frequency (*Frequency*) and monetary value (*Monetary*). Although extremely widespread for its interpretative simplicity, the RFM model has some limitations when it comes to evaluating the future economic potential of a customer. For this reason, the concept of **CLV** has also been included, which considers estimated future purchases, the duration of the relationship with the customer (*lifespan*) and other parameters that can give a long-term view on the economic value that can be generated.

Algorithmically, each of the five clustering methods offers a different perspective:

- *K-Means* and *Fuzzy C-Means* use a centroid-based approach, useful for obtaining compact groups, and differ in membership (hard vs. fuzzy).
- *Hierarchical Clustering* (particularly with Ward.D2) builds a hierarchy of clusters, which is useful when exploring the nested structure of data.
- *DBSCAN* identifies regions of high density by separating them from areas of

lower density (classified as *noise*), without the need to fix the number of clusters a priori.

- *GMM* adds a probabilistic approach, assuming that the data comes from a mixture of Gaussian distributions, each with its own mean and variance.

The starting data set, coming from the *UCI Machine Learning Repository*, was first cleaned and pre-treated (including removal of duplicates, management of outliers and date conversion). For the RFM part, three fundamental indicators were calculated (*Recency*, *Frequency*, *Monetary*), while for the CLV part, *TotalRevenue*, *Average Transaction Value*, *Frequency*, *Lifespan* and the estimate of *CLV* itself were introduced. Once these sets of variables were obtained, we proceeded with the application and comparison of the clustering algorithms, also evaluated through the internal metrics **Silhouette Score** and **Calinski–Harabasz Index**.

2. Key Findings in RFM Analysis

2.1 Default RFM Segments

A first analysis divided customers into six segments **(Whales, Active Stars, Loyal Regulars, New & Occasional Buyers, Sleeping Giants, Lost Clients)** based on the RFM scores calculated with the quintile method. This "manual" categorization indicated that almost half of the customers (over 47%) fall into the *Lost Clients*, i.e. customers who do not show recent or frequent purchases, with an overall modest spending value. On the other hand, a small group of **Whales** (just over 8%) generates a spending volume that is enormously higher than the average, placing itself as a top priority segment for *retention* or *cross-selling* strategies.

2.2 Comparison of Clustering Algorithms (RFM)

- **K-Means**: It highlighted 4 clusters (determined by Elbow Method). One of these (Cluster 2) includes very few customers (13) with high monetary contribution ($\sim$18.5% of the total), confirming the existence of an elite group with very high economic value.

- **DBSCAN**: It identified 3 clusters, one of which is classified as *noise* (Cluster 0) and contains customers with an even more out of scale spending profile. This highlights a potential limit of DBSCAN, which tends to isolate the “extreme” points and classify them as noise if the ϵ and *MinPts* parameters are not calibrated very carefully.

- **Hierarchical Clustering**: With a cut to 7 clusters, it allowed to identify in a granular way segments of customers with high frequency and monetary value, including some segments of minimum size but very high value ($M \approx$ 164,658£ in one case). The hierarchical approach is very transparent, allowing to visualize how the groupings are formed through a dendrogram, although the choice of the cut point remains subjective.

- **GMM**: It revealed 9 clusters for the RFM dataset, a rather high number. Here too, a small group of customers (Cluster 9) with an extremely high average monetary value (over 46,000 €) and a series of "intermediate" clusters with various compositions emerge. This shows the ability of GMM to capture nuances, but raises doubts about the interpretability of segmentations that are too fragmented.
- **Fuzzy C-Means (FCM)**: By setting 4 clusters (in line with K-Means), it distributed the customers in a less clear-cut way. Thanks to the fuzzy membership, some customers are positioned between multiple groups, reflecting that in real situations the behavioral boundaries are not always clear. However, the interpretation of the "partial memberships" requires a greater analytical effort.

2.3 Application Interpretations

The RFM analysis shows that "high-spending" customers are often few but decisive for the turnover. This suggests that the RFM segmentation—with its immediacy of calculation—is excellent for tactical *snapshots*, such as planning seasonal campaigns or identifying inactive clusters (*Lost Clients*) to recover. However, if the decision horizon extends to long-term considerations (e.g. estimate of future revenues or the probability of customer "survival"), RFM risks losing its effectiveness because it does not incorporate the prospective temporal dynamics.

3. Main Evidence in CLV Analysis

3.1 Various Definitions of Long-Term Value

The second part of the thesis focused on the **CLV**, calculated by integrating *total spent*, *frequency*, *average value per transaction* and *lifespan* of the customer, to obtain a metric that summarizes the potential return over a prolonged time period. This perspective better intercepts marketing strategies oriented to the balance between maintaining the "best customers" and acquiring new high-potential ones.

3.2 Comparison of Clustering Algorithms (CLV)

- **K-Means**: With $k = 4$, it was highlighted how a very small cluster (only 2 customers) has an *extraordinary* average CLV, higher than 11 million euros. A second cluster (23 customers) shows very high frequency and robust CLV, while the majority falls into low or medium value clusters.
- **Hierarchical Clustering**: It provided 4 clusters, one of which is again occupied by a few *ultra-premium* customers, and another by customers with relatively high frequency. The hierarchical approach confirms the presence of large masses of low-value customers and elite minorities with extreme parameters.
- **DBSCAN**: Detected only 2 main clusters, separating a group of 81 "top spenders" (Cluster 0) from the rest (Cluster 1, over 4000 low CLV customers).

This clear separation, while easy to interpret, neglects the presence of possible substructures within the high-value group, as suggested by GMM or the dendrogram.

- **GMM**: Estimates 6 clusters via BIC-based *mclust*. Different gradations of high-value customers emerge, including those who make a single, very expensive purchase and those who have very high frequencies, with varied behaviors. The risk, as always, is over-segmentation, which could be too granular for lean marketing plans.
- **Fuzzy C-Means**: Set on 4 clusters, it reiterated the existence of an *exceptional* micro-segment (fC 3 with just 7 customers but average CLV $>$ 10 million) and a medium-high segment (fC 2) with about 333 customers. Most of the population remained in low-value clusters (fC 1 and 4), highlighting a large portion of customers that, in an *upselling* perspective, could be encouraged to grow.

4. Comparative Considerations: RFM vs. CLV

A central element of this work consists in the comparison between the segmentation based on **RFM** and that based on **CLV**. Both models identify *few* customers with very high economic value and *many* customers with low spending. However:

1. **Time horizon**:
   - RFM favors the current behavior. If a customer with high past purchases stops buying for a few months, in the RFM *Recency* will get worse, and that individual could move from a "high-spending" cluster to a less desirable one.
   - CLV instead estimates the future propensity, possibly recognizing high spending margins if historically the frequency has been high and the analysis time window suggests a probability of repurchase.
2. **Strategic**:
   - RFM lends itself to short-medium term *direct marketing* (for example, how to launch a Christmas promotion or a retargeting action on recently inactive customers).
   - CLV is more connected to strategic decisions: defining acquisition budgets, predictively evaluating the effectiveness of investments in *retention*, justifying extreme customization for the "top tiers".
3. **Computation Complexity**:
   - RFM is simpler and easily adoptable by companies with basic IT infrastructures.
   - CLV calculation requires forecasts or hypotheses on future behavior and requires models or *assumptions* (e.g. discount rates, estimated retention rate), making the procedure more complex.

4. **Segmentation Stability**:
    - RFM scores can fluctuate rapidly when the time dimension varies (a customer considered *recent* today may not be so in a few weeks).
    - CLV offers a more stable picture, at least as long as the calculation assumptions remain valid. However, if purchasing patterns change dramatically, CLV estimates should also be reevaluated.

5. Clustering Algorithm Selection and Differentiated Approaches

No algorithm has proven to be unquestionably "best" overall: its quality depends on the business goals, the shape of the data, and whether well-separated clusters or fluid clusters are preferred. The analyses conducted in this thesis show that applying multiple clustering methods provides a comprehensive framework for customer segmentation. Each approach contributes unique insights, extending beyond technical descriptions.

- **K-Means**: A fast and computationally efficient method for segmenting customers into distinct, compact groups. It supports broad marketing strategies and resource allocation but rigidly assigns customers to clusters, potentially overlooking subtle behavioral differences. It is most effective when the number of clusters is well-defined and the data lacks strong outliers.
- **Hierarchical Clustering**: Provides a multi-level view of customer segmentation, revealing both broad categories and nested subgroups. This structure is valuable for analyzing variations in customer loyalty and engagement. While dendrograms enhance interpretability, determining the optimal cut remains subjective.
- **DBSCAN**: Excels at detecting dense micro-clusters and identifying outliers, which may represent niche segments or anomalies. Its density-based approach uncovers non-linear clusters that centroid-based methods might miss, making it particularly useful for re-engagement strategies. However, improper calibration of ϵ and *MinPts* may lead to misclassification of valuable customers as noise.
- **Gaussian Mixture Models (GMM)**: Uses a probabilistic framework, assigning each customer a likelihood of belonging to multiple clusters. This makes it effective for capturing overlapping customer behaviors and supporting dynamic marketing strategies. However, it risks over-segmenting the data, particularly if the distribution does not follow Gaussian assumptions.
- **Fuzzy C-Means (FCM)**: Similar to GMM, this approach assigns degrees of membership rather than hard classifications, reflecting the continuum of customer behaviors. This flexibility is valuable when customers exhibit affinities for multiple segments (e.g., purchasing both premium and standard products). However, interpreting fuzzy memberships requires additional effort compared to traditional clustering methods.

The combined use of these clustering methods allows for a balanced strategy, leveraging the clarity of hard clustering with the adaptability of soft clustering. This integrated perspective enhances customer segmentation, optimizes resource allocation, and supports adaptive marketing strategies that align with the evolving nature of customer interactions. By capturing both broad trends and subtle behavioral nuances, these methods contribute to more informed, data-driven decision-making in CRM and marketing.

6. Managerial Implications

The distinctions between RFM and CLV, coupled with the varying characteristics of clustering algorithms, provide numerous insights for developing operational strategies:

1. **Customer Portfolio Management**
    - Identify *Whales* or *High-CLV* segments as priorities for exclusive campaigns, such as personalized discounts, early access to products, and dedicated customer service.
    - For larger low-value clusters (*Lost Clients* in RFM or clusters of "one-timers" in CLV), it is advisable to implement cost-effective actions. Strategies may include mass actions (e.g., generic email marketing) or selective recovery efforts (e.g., targeted discounts, loyalty packages), as investing in expensive strategies for these segments does not yield significant added value.
2. **Resource Optimization**
    - Allocate marketing resources (budget, time, contacts) proportionally to the potential value of each cluster. Specifically, CLV can justify greater investments in retaining top customers, where a high return is anticipated in the long term.
3. **Loyalty and Cross-Selling Programs**
    - Encourage segments with high *Frequency* but moderate unit spending to purchase higher-margin products through cross-selling (offering related or complementary products) and up-selling (encouraging the purchase of more expensive items) initiatives.
    - Target segments with low *Recency* but a history of substantial spending (*Sleeping Giants*) with personalized "win-back campaigns" (strategies aimed at re-engaging inactive customers).
4. **Competition and Offer Analysis**
    - If a company observes that customers within a specific cluster are migrating to competitors, corrective measures can be adopted, such as improving service quality, reducing delivery times, or launching targeted promotions.
    - RFM and CLV metrics can serve as internal benchmarks to monitor how customer distributions across segments evolve over time.

5. **Data-Driven Culture**
    - Implement internal *dashboards* that allow managers to explore RFM and CLV metrics in real time (or near real time).
    - The synergy between the two perspectives (short- to medium-term RFM and long-term CLV) reinforces a corporate culture based on data and predictive analysis, rather than solely on instinctive decision-making.

7. Limitations and Future Perspectives

Although the research has provided important insights, there are some aspects to consider for possible future developments:

1. **Data Quality and Updating**
    - The analyzed dataset, although robust, reflects a specific period of time (about a year of transactions). In real contexts, the data should be continuously updated, and the clustering models "recursively" recalibrated.
    - The possible presence of seasonality (Christmas period, Black Friday, etc.) could alter the values of *Recency* and *Frequency* significantly.
2. **Relevance of Additional Variables**
    - Both RFM and CLV ignore dimensions such as product category, net profitability (which would require considering costs and margins), demographics (age, location) or behavioral variables (feedback, reviews, return rate). Integrating additional data sources could refine the segmentation, but make the calculation more complex.
3. **Clustering Parameters**
    - DBSCAN, for example, is extremely sensitive to the choice of ϵ. The same is true for GMM, which can return a variable number of clusters based on the BIC. Greater methodological robustness could include a more sophisticated *model selection* (cross-evaluation of multiple metrics and comparisons with simulated data).
4. **Advanced Machine Learning Applications**
    - If segmentation is combined with predictive models (e.g. *churn forecasting* or *propensity scoring*), even more targeted results can be obtained. Methods such as *deep clustering* could be tested in contexts with large amounts of unstructured data (clickstream, navigation logs).
5. **External Validation**
    - In this work, internal validation metrics were used (*Silhouette*, *Calinski–Harabasz*). It would be useful to integrate an external validation, measuring the actual impact of each cluster on real business metrics (e.g. redemption rate of campaigns, upgrade rate to premium plans, etc.).

8. Overall Conclusion

Ultimately, this thesis has shown how, in a marketing and CRM context, the choice between **RFM** and **CLV** and the selection of a particular clustering algorithm cannot be reduced to universal rules valid for all. On the contrary, it is necessary to consider:

- *The nature of the data*: A dataset with many outliers and a highly skewed distribution can benefit from the most flexible clustering models (DBSCAN, GMM), as long as parameters that distort its interpretation are avoided (for example, erroneously defining valuable customers as "noise").
- *The time horizon and strategy*: If marketing actions aim to recover inactive customers of the last few months, RFM is an immediate indicator. If, on the other hand, it is a question of investing in long-term loyalty programs, CLV becomes central.
- *The desired granularity*: K-Means and FCM allow a fixed number of clusters, while DBSCAN, GMM and Hierarchical can potentially create more or less segments, proving useful or dispersive depending on the case.
- *The ease of interpretation*: The adoption of a certain method should always take into account the possibility of communicating the results clearly to business decision-makers. An excessive fragmentation into poorly distinguishable clusters risks confusing managers rather than helping them.

This work contributes to the literature on *customer analytics* by demonstrating how different clustering methods can lead to partly divergent interpretations, especially in the presence of highly heterogeneous data and with few individuals generating the majority of the revenue. However, the "complementary" nature of RFM and CLV metrics suggests that the ideal choice often consists in **combine both** perspectives. For example, a company could define primary clusters using RFM (quick to update and interpret), and then *prioritize* customers with the highest CLVs within each cluster.

From an operational perspective, the results obtained provide a practical *framework* for those within the organization who want to identify, describe and retain the best customers, without neglecting conversion opportunities among the "“average” groups or recovery of dormant groups. The road to truly personalized marketing passes through the continuous evolution of these models: **iterating** the segmentation with updated data, inserting **new variables** that provide additional levels of depth (net profitability, purchase preferences, channels used, social media interactions) and **comparing** the hypotheses with tangible economic results.

In summary, the thesis reinforces the idea that there is no "“one size fits all” for customer segmentation. Each business context and each marketing objective require tailor-made analyses, both in terms of the definition of metrics (RFM vs. CLV, or a mix of both) and the choice of the clustering algorithm. In the long run, a **hybrid** and **iterative** approach appears to be the most solid solution, where the results of one tool (e.g. RFM) are enriched and validated by the perspectives of another (CLV), and where multiple clustering methods are compared to capture the patterns that

best reflect the market structure and business needs. This approach allows to acquire an integrated vision and to draw increasingly effective *data-driven* decisions in the current competitive context.

Appendix

.1 RFM analysis, clustering code and evalutation

```r
library(dplyr)
library(lubridate)
library(ggplot2)
library(plotly)
library(cluster)
library(dbscan)
library(mclust)
library(ggdendro)
library(e1071)
library(reshape2)

########## read data ##########
data <- read.csv("data/data.csv")
data <- data %>%
  mutate(
    InvoiceDate = as.POSIXct(InvoiceDate, format = "%m/%d/%Y %H:%M"),
    TotalPrice = Quantity * UnitPrice
  ) %>%
  filter(!is.na(CustomerID), Quantity > 0, UnitPrice > 0)

### rfm metrics ##########
reference_date <- max(data$InvoiceDate) + days(1)
RFM <- data %>%
  group_by(CustomerID) %>%
  summarise(
    Recency = as.numeric(difftime(reference_date, max(InvoiceDate),
    units = "days")),
    Frequency = n_distinct(InvoiceNo),
    Monetary = sum(TotalPrice, na.rm = TRUE)
  ) %>%
  mutate(
    R_score = ntile(desc(Recency), 5),
    F_score = ntile(Frequency, 5),
    M_score = ntile(Monetary, 5),
    RFM_score = paste0(R_score, F_score, M_score)
  )

########## rfm segments #######
RFM <- RFM %>%
```

```
  mutate(
    segments = case_when(
      R_score == 5 & F_score == 5 & M_score == 5 ~ "Whales",
      R_score >= 4 & F_score >= 3 & M_score >= 4 ~ "Active Stars",
      R_score >= 4 & F_score >= 4 & M_score <= 3 ~ "Loyal Regulars",
      R_score <= 3 & F_score >= 3 & M_score >= 4 ~ "Sleeping Giants",
      R_score >= 4 & F_score <= 3 & M_score <= 5 ~ "New & Occasional
    Buyers",
      TRUE ~ "Lost Clients"
    )
  )

##### rfm segment distribution ##########
segment_distribution <- prop.table(table(RFM$segments)) * 100
segment_distribution <- as.data.frame(segment_distribution)
colnames(segment_distribution) <- c("Segment", "Percentage")

ggplot(segment_distribution, aes(x = reorder(Segment, -Percentage), y =
    Percentage, fill = Segment)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = sprintf("%.1f%%", Percentage)), vjust = -0.5,
    size = 3.5) +
  scale_fill_brewer(palette = "Set3") +
  labs(title = "rfm segment distribution", x = "segment", y = "
    percentage") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1), legend.
    position = "none")

########## rfm 3d plot ##########
plot_ly(
  data = RFM,
  x = ~Recency,
  y = ~Frequency,
  z = ~Monetary,
  color = ~segments,
  type = "scatter3d",
  mode = "markers"
) %>%
  layout(
    title = "rfm segmentation (log scale)",
    scene = list(
      xaxis = list(title = 'recency', type = 'log'),
      yaxis = list(title = 'frequency', type = 'log'),
      zaxis = list(title = 'monetary', type = 'log')
    )
  )

###### rfm stats #####
segment_stats <- RFM %>%
  group_by(segments) %>%
  summarise(
    RecencyMean = mean(Recency),
    RecencyMedian = median(Recency),
    FrequencyMean = mean(Frequency),
    FrequencyMedian = median(Frequency),
```

```
    MonetaryMean = mean(Monetary),
    MonetaryMedian = median(Monetary),
    GroupSize = n()
  )
print(segment_stats)

###### normalization #######
RFM_normalized <- RFM %>%
  mutate(
    Recency = scale(Recency),
    Frequency = scale(Frequency),
    Monetary = scale(Monetary)
  ) %>%
  select(Recency, Frequency, Monetary)

########## elbow method ########
set.seed(333)
wss <- sapply(1:10, function(k) {
  kmeans(RFM_normalized, centers = k, nstart = 10)$tot.withinss
})
ggplot(data.frame(k = 1:10, wss = wss), aes(x = k, y = wss)) +
  geom_line(size = 1, color = "blue") +
  scale_x_continuous(breaks = 1:10) +
  labs(title = "elbow method", x = "number of clusters", y = "within
    sum of squares") +
  theme_minimal() +
  theme(panel.border = element_rect(color = "black", fill = NA, size =
    1))

elbow_values <- data.frame(k = 1:10, wss = wss)
print(elbow_values)

########## k-means #########
k_optimal <- 4
kmeans_result <- kmeans(RFM_normalized, centers = k_optimal, nstart =
    10)
RFM$kmeans_cluster <- as.factor(kmeans_result$cluster)

ggplot(RFM, aes(x = Recency, y = Frequency, size = Monetary, color =
    kmeans_cluster)) +
  geom_point(alpha = 0.7) +
  labs(title = "k-means clustering (rfm)", x = "recency", y = "
    frequency") +
  theme_minimal()

plot_ly(
  data = RFM,
  x = ~Recency,
  y = ~Frequency,
  z = ~Monetary,
  color = ~kmeans_cluster,
  type = "scatter3d",
  mode = "markers",
  marker = list(size = 4)
) %>%
  layout(
```

```
    title = "k-means clustering (rfm) log scale",
    scene = list(
      xaxis = list(title = 'recency', type = 'log'),
      yaxis = list(title = 'frequency', type = 'log'),
      zaxis = list(title = 'monetary', type = 'log')
    )
  )

cluster_statistics <- function(data, cluster_column) {
  data %>%
    group_by(!!sym(cluster_column)) %>%
    summarise(
      RecencyMean = mean(Recency),
      RecencyMedian = median(Recency),
      FrequencyMean = mean(Frequency),
      FrequencyMedian = median(Frequency),
      MonetaryMean = mean(Monetary),
      MonetaryMedian = median(Monetary),
      GroupSize = n()
    )
}
kmeans_stats <- cluster_statistics(RFM, "kmeans_cluster")
print(kmeans_stats)

########## dbscan ######
kNNdistplot <- function(data, k) {
  distances <- kNNdist(data, k = k)
  distances <- sort(distances)
  plot(distances, type = "l", main = "k-NN distance plot",
       xlab = "points sorted by distance", ylab = "k-NN distance")
  abline(h = 0.45, col = "red", lty = 2)
}
kNNdistplot(RFM_normalized, k = 5)

dbscan_eps <- 0.45
dbscan_minpts <- 5
dbscan_model <- fpc::dbscan(RFM_normalized, eps = dbscan_eps, MinPts =
    dbscan_minpts)
RFM$dbscan_cluster <- as.factor(dbscan_model$cluster)

ggplot(RFM, aes(x = Recency, y = Frequency, size = Monetary, color =
    dbscan_cluster)) +
  geom_point(alpha = 0.7) +
  labs(title = "dbscan clustering (rfm)", x = "recency", y = "frequency
    ") +
  theme_minimal()

plot_ly(
  data = RFM,
  x = ~Recency,
  y = ~Frequency,
  z = ~Monetary,
  color = ~dbscan_cluster,
  type = "scatter3d",
  mode = "markers"
) %>%
```

```
  layout(
    title = "dbscan clustering (log scale)",
    scene = list(
      xaxis = list(title = 'recency', type = 'log'),
      yaxis = list(title = 'frequency', type = 'log'),
      zaxis = list(title = 'monetary', type = 'log')
    )
  )

dbscan_stats <- cluster_statistics(RFM, "dbscan_cluster")
print(dbscan_stats)

########## gmm ##########
set.seed(333)
gmm_result <- Mclust(RFM_normalized)
RFM$gmm_cluster <- as.factor(gmm_result$classification)

plot(gmm_result, what = "BIC")

ggplot(RFM, aes(x = Recency, y = Frequency, size = Monetary, color = gmm_cluster)) +
  geom_point(alpha = 0.7) +
  labs(title = "gmm clustering (rfm)", x = "recency", y = "frequency") +
  theme_minimal()

plot_ly(
  data = RFM,
  x = ~Recency,
  y = ~Frequency,
  z = ~Monetary,
  color = ~gmm_cluster,
  type = "scatter3d",
  mode = "markers",
  marker = list(size = 4)
) %>%
  layout(
    title = "gmm clustering (log scale)",
    scene = list(
      xaxis = list(title = 'recency', type = 'log'),
      yaxis = list(title = 'frequency', type = 'log'),
      zaxis = list(title = 'monetary', type = 'log')
    )
  )

gmm_stats <- cluster_statistics(RFM, "gmm_cluster")
print(gmm_stats)

########## hierarchical ##########
dist_matrix <- dist(RFM_normalized)
hc_model <- hclust(dist_matrix, method = "ward.D2")
RFM$hierarchical_cluster <- as.factor(cutree(hc_model, k = 7))

plot(hc_model, labels = FALSE, main = "hierarchical clustering dendrogram",
     xlab = "", sub = "ward method")
```

```
247 rect.hclust(hc_model, k = 7, border = "red")
248
249 dendro <- as.dendrogram(hc_model)
250 dendro_data <- dendro_data(dendro)
251
252 k <- 7
253 cluster_assignments <- cutree(hc_model, k)
254 labels_df <- data.frame(label = rownames(RFM_normalized), cluster =
        cluster_assignments, stringsAsFactors = FALSE)
255 labels_df <- merge(labels_df, dendro_data$labels, by = "label")
256
257 rect_data <- aggregate(x ~ cluster, data = labels_df, FUN = function(x)
        c(min(x), max(x)))
258 rect_data <- do.call(rbind, lapply(1:nrow(rect_data), function(i) {
259   data.frame(
260     cluster = rect_data$cluster[i],
261     xmin = rect_data$x[i, 1],
262     xmax = rect_data$x[i, 2],
263     ymin = 0,
264     ymax = max(dendro_data$segments$yend) / 2
265   )
266 }))
267
268 ggplot() +
269   geom_segment(data = dendro_data$segments,
270                aes(x = x, y = y, xend = xend, yend = yend)) +
271   geom_rect(data = rect_data,
272             aes(xmin = xmin, xmax = xmax, ymin = ymin, ymax = ymax,
        fill = as.factor(cluster)),
273             alpha = 0.3) +
274   scale_fill_manual(name = "cluster",
275                     values = c("#FF6666", "#FFCC66", "#66CC66", "#66
        CCCC", "#6666FF", "#CC66FF", "#FF66CC")) +
276   labs(title = "hierarchical clustering dendrogram", x = "clients", y =
        "height") +
277   theme_minimal() +
278   theme(axis.text.x = element_blank(), axis.ticks.x = element_blank(),
        legend.position = "right")
279
280 # zoom on some clusters
281 zoom_xmin <- min(rect_data$xmin[rect_data$cluster %in% c(5, 6, 7)])
282 zoom_xmax <- max(rect_data$xmax[rect_data$cluster %in% c(5, 6, 7)])
283 zoom_ymax <- max(dendro_data$segments$yend)
284
285 ggplot() +
286   geom_segment(data = dendro_data$segments,
287                aes(x = x, y = y, xend = xend, yend = yend)) +
288   geom_rect(data = rect_data[rect_data$cluster %in% c(5, 6, 7), ],
289             aes(xmin = xmin, xmax = xmax, ymin = ymin, ymax = ymax,
        fill = as.factor(cluster)),
290             alpha = 0.3) +
291   scale_fill_manual(name = "cluster",
292                     values = c("#FFCC66", "#66CC66", "#FF6666")) +
293   coord_cartesian(xlim = c(zoom_xmin, zoom_xmax), ylim = c(0, zoom_ymax
        / 3)) +
```

```
294   labs(title = "zoom on clusters 5, 6, and 7", x = "clients", y = "
        height") +
295   theme_minimal() +
296   theme(axis.text.x = element_blank(), axis.ticks.x = element_blank(),
        legend.position = "right")
297
298 ggplot(RFM, aes(x = Recency, y = Frequency, size = Monetary, color =
      hierarchical_cluster)) +
299   geom_point(alpha = 0.7) +
300   labs(title = "hierarchical clustering (rfm)", x = "recency", y = "
        frequency") +
301   theme_minimal()
302
303 plot_ly(
304   data = RFM,
305   x = ~Recency,
306   y = ~Frequency,
307   z = ~Monetary,
308   color = ~hierarchical_cluster,
309   type = "scatter3d",
310   mode = "markers",
311   marker = list(size = 4)
312 ) %>%
313   layout(
314     title = "hierarchical clustering (log scale)",
315     scene = list(
316       xaxis = list(title = 'recency', type = 'log'),
317       yaxis = list(title = 'frequency', type = 'log'),
318       zaxis = list(title = 'monetary', type = 'log')
319     )
320   )
321
322 hierarchical_stats <- cluster_statistics(RFM, "hierarchical_cluster")
323 print(hierarchical_stats)
324
325 ########## fuzzy c-means ##########
326 set.seed(333)
327 fcm_result <- cmeans(RFM_normalized, centers = k_optimal, m = 2, iter.
      max = 100, verbose = FALSE)
328 RFM$fuzzy_cluster <- apply(fcm_result$membership, 1, which.max)
329
330 ggplot(RFM, aes(x = Recency, y = Frequency, size = Monetary, color =
      factor(fuzzy_cluster))) +
331   geom_point(alpha = 0.7) +
332   labs(title = "fuzzy c-means clustering (rfm)", x = "recency", y = "
        frequency") +
333   theme_minimal()
334
335 sample_indices <- sample(1:nrow(RFM_normalized), 20)
336 sample_membership <- fcm_result$membership[sample_indices, ]
337 membership_df <- as.data.frame(sample_membership)
338 membership_df$Client <- paste("client", 1:nrow(membership_df))
339 membership_long <- melt(membership_df, id.vars = "Client",
340                         variable.name = "Cluster", value.name = "
      Membership")
341
```

```
ggplot(membership_long, aes(x = Cluster, y = Client, fill = Membership)) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "lightblue", high = "darkblue", name = "membership") +
  labs(title = "fuzzy c-means membership (sample)", x = "cluster", y = "client") +
  theme_minimal() +
  theme(axis.text.x = element_text(size = 10),
        axis.text.y = element_text(size = 8, angle = 45, hjust = 1),
        axis.ticks = element_blank())

plot_ly(
  data = RFM,
  x = ~Recency,
  y = ~Frequency,
  z = ~Monetary,
  color = ~factor(fuzzy_cluster),
  type = "scatter3d",
  mode = "markers"
) %>%
  layout(
    title = "fuzzy c-means clustering (log scale)",
    scene = list(
      xaxis = list(title = 'recency', type = 'log'),
      yaxis = list(title = 'frequency', type = 'log'),
      zaxis = list(title = 'monetary', type = 'log')
    )
  )

fuzzy_stats <- cluster_statistics(RFM, "fuzzy_cluster")
print(fuzzy_stats)

########## evaluation (silhouette & ch index) ##########
dist_matrix <- dist(RFM_normalized)

silhouette_kmeans <- silhouette(as.numeric(RFM$kmeans_cluster), dist_matrix)
silhouette_dbscan <- silhouette(as.numeric(RFM$dbscan_cluster), dist_matrix)
silhouette_hierarchical <- silhouette(as.numeric(RFM$hierarchical_cluster), dist_matrix)
silhouette_gmm <- silhouette(as.numeric(RFM$gmm_cluster), dist_matrix)
silhouette_fuzzy <- silhouette(as.numeric(RFM$fuzzy_cluster), dist_matrix)

cat("silhouette scores:\n")
cat("k-means: ", mean(silhouette_kmeans[, 3]), "\n")
cat("dbscan: ", mean(silhouette_dbscan[, 3], na.rm = TRUE), "\n")
cat("hierarchical: ", mean(silhouette_hierarchical[, 3]), "\n")
cat("gmm: ", mean(silhouette_gmm[, 3]), "\n")
cat("fuzzy c-means: ", mean(silhouette_fuzzy[, 3]), "\n")

calinski_harabasz <- function(data, cluster_vector) {
  k <- length(unique(cluster_vector))
  if (k < 2) return(NA)
```

```
391   data_mat <- as.matrix(data)
392   n <- nrow(data_mat)
393   overall_mean <- colMeans(data_mat)
394   W <- 0
395   B <- 0
396   for (cl in unique(cluster_vector)) {
397     cl_indices <- which(cluster_vector == cl)
398     cl_data <- data_mat[cl_indices, , drop = FALSE]
399     cl_mean <- colMeans(cl_data)
400     W <- W + sum(rowSums((cl_data - cl_mean)^2))
401     B <- B + nrow(cl_data) * sum((cl_mean - overall_mean)^2)
402   }
403   if (W == 0 || k == 1 || (n - k) == 0) return(NA)
404   (B / (k - 1)) / (W / (n - k))
405 }
406
407 ch_kmeans <- calinski_harabasz(RFM_normalized, RFM$kmeans_cluster)
408 ch_dbscan <- calinski_harabasz(RFM_normalized, RFM$dbscan_cluster)
409 ch_hierarchical <- calinski_harabasz(RFM_normalized, RFM$hierarchical_
      cluster)
410 ch_gmm <- calinski_harabasz(RFM_normalized, RFM$gmm_cluster)
411 ch_fuzzy <- calinski_harabasz(RFM_normalized, RFM$fuzzy_cluster)
412
413 ch_results <- data.frame(
414   method = c("k-means", "dbscan", "hierarchical", "gmm", "fuzzy c-means
      "),
415   calinski_harabasz = c(ch_kmeans, ch_dbscan, ch_hierarchical, ch_gmm,
      ch_fuzzy)
416 )
417 print(ch_results)
```

.1.1 RFM 3D visualization

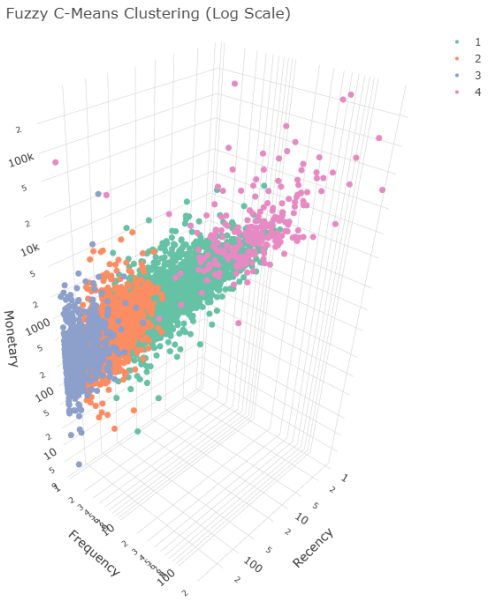


Figure 1: RFM K-Means Clustering (LOG)

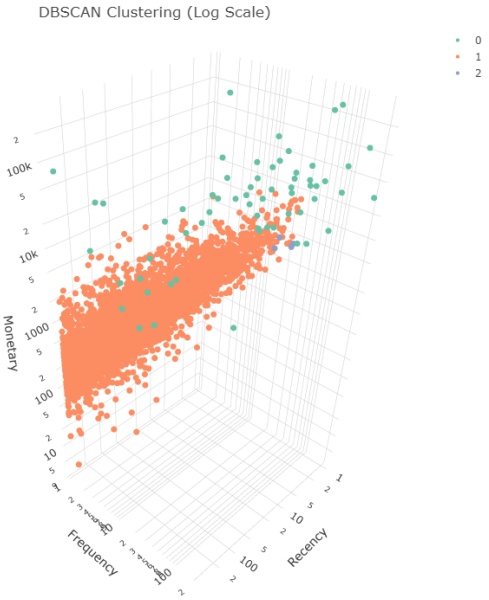


Figure 2: RFM DBSCAN Clustering (LOG)

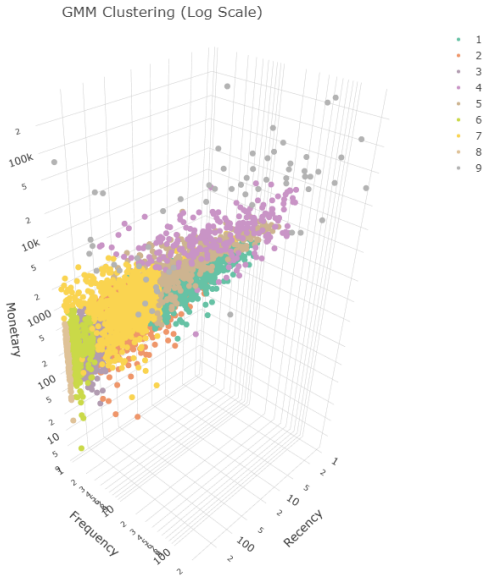


Figure 3: RFM GMM Clustering (LOG)

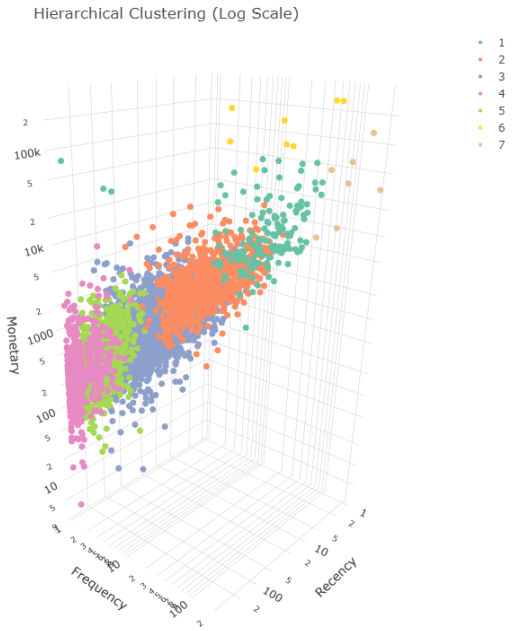


Figure 4: RFM Hierarchical Clustering (LOG)

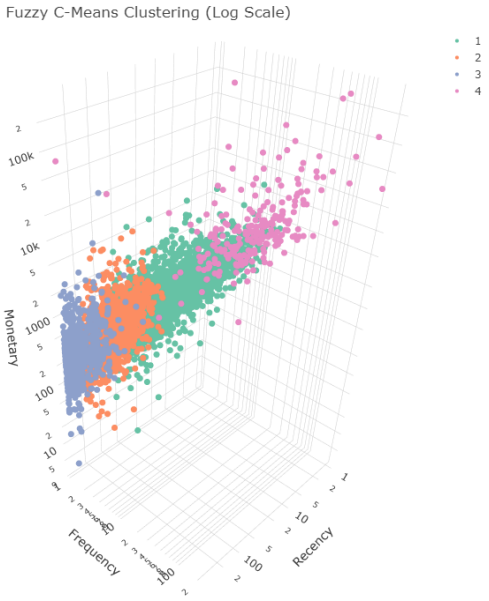


Figure 5: RFM Fuzzy Clustering (LOG)

.2 CLV segmentation,clustering and model evaluation

```r
########### load libraries ###########
library(e1071)
library(dplyr)
library(lubridate)
library(ggplot2)
library(cluster)
library(dbscan)
library(mclust)
library(reshape2)
library(ggdendro)
library(fmsb)
########### load and preprocess data ##
data <- read.csv("data/data.csv")
data <- data %>%
  mutate(InvoiceDate = as.Date(InvoiceDate, format = "%m/%d/%Y")) %>%
  mutate(TotalPrice = Quantity * UnitPrice) %>%
  filter(!is.na(CustomerID), Quantity > 0, UnitPrice > 0) %>%
  group_by(CustomerID) %>%
  summarise(
    TotalRevenue = sum(TotalPrice),
    Frequency = n_distinct(InvoiceNo),
    AvgTransactionValue = mean(TotalPrice),
    Lifespan = as.numeric(difftime(max(InvoiceDate), min(InvoiceDate),
    units = "days"))
  ) %>%
```

```r
  ungroup()

########## calculate clv ##########
data <- data %>%
  mutate(CLV = Frequency * AvgTransactionValue * Lifespan)

########## normalize data ##########
data_normalized <- data %>%
  select(TotalRevenue, Frequency, AvgTransactionValue, Lifespan, CLV) %>%
  scale() %>%
  as.data.frame()
rownames(data_normalized) <- data$CustomerID

########## define statistics functions ##########
calculate_statistics <- function(data, cluster_column) {
  data %>%
    group_by(!!sym(cluster_column)) %>%
    summarise(
      TotRevenueMean = mean(TotalRevenue),
      FMean = mean(Frequency),
      CLVMean = mean(CLV),
      AvgTMean = mean(AvgTransactionValue),
      LifespanMean = mean(Lifespan),
      TotSD = sd(TotalRevenue),
      FSD = sd(Frequency),
      CLVSD = sd(CLV),
      AvgTSD = sd(AvgTransactionValue),
      LifespanSD = sd(Lifespan),
      Size = n()
    )
}

##### elbow method ########
set.seed(123)
wss <- sapply(1:10, function(k) {
  kmeans(data_normalized, centers = k, nstart = 10)$tot.withinss
})
ggplot(data.frame(k = 1:10, wss = wss), aes(x = k, y = wss)) +
  geom_line(size = 1, color = "blue") +
  scale_x_continuous(breaks = 1:10) +
  labs(title = "elbow method", x = "number of clusters", y = "within sum of squares") +
  theme_minimal() +
  theme(
    panel.border = element_rect(color = "black", fill = NA, size = 1)
  )

k_values <- 1:10

#print elbow values
elbow_values <- data.frame(k = k_values, wss = wss)
print(elbow_values)

########## k-means ##########
k_optimal <- 4
```

```
79
80 kmeans_result <- kmeans(data_normalized, centers = k_optimal, nstart =
       10)
81 data$kmeans_cluster <- as.factor(kmeans_result$cluster)
82
83 ########## visualize k-means ##########
84 ggplot(data, aes(x = Frequency, y = TotalRevenue, color = kmeans_
       cluster, size = CLV)) +
85   geom_point(alpha = 0.7) +
86   labs(title = "k-means clustering (clv)", x = "frequency", y = "total
       revenue", size = "clv") +
87   theme_minimal()
88
89
90 ########## k-means statistics ##########
91 kmeans_stats <- calculate_statistics(data, "kmeans_cluster")
92 print(kmeans_stats)
93
94 ########## hierarchical ##########
95 dist_matrix <- dist(data_normalized)
96 hc_model <- hclust(dist_matrix, method = "ward.D2")
97 data$hierarchical_cluster <- cutree(hc_model, k = k_optimal)
98
99 ########## dendrogram ##########
100 plot(hc_model, main = "hierarchical clustering dendrogram",
101      xlab = "index", ylab = "distance")
102 rect.hclust(hc_model, k = k_optimal, border = "red")
103
104 ########## enhanced dendrogram ##########
105 dendro <- as.dendrogram(hc_model)
106 dendro_data <- dendro_data(dendro)
107
108 cluster_assignments <- cutree(hc_model, k = k_optimal)
109 labels_df <- data.frame(label = rownames(data_normalized),
110                         cluster = cluster_assignments,
111                         stringsAsFactors = FALSE)
112 labels_df <- merge(labels_df, dendro_data$labels, by = "label")
113
114 rect_data <- aggregate(x ~ cluster, data = labels_df,
115                        FUN = function(x) c(min(x), max(x)))
116
117 rect_data <- do.call(rbind, lapply(1:nrow(rect_data), function(i) {
118   data.frame(
119     cluster = rect_data$cluster[i],
120     xmin = rect_data$x[i, 1],
121     xmax = rect_data$x[i, 2],
122     ymin = 0,
123     ymax = max(dendro_data$segments$yend) / 2
124   )
125 }))
126
127 ggplot() +
128   geom_segment(data = dendro_data$segments,
129                aes(x = x, y = y, xend = xend, yend = yend)) +
130   geom_rect(data = rect_data,
```

```
131             aes(xmin = xmin, xmax = xmax, ymin = ymin, ymax = ymax,
    fill = as.factor(cluster)),
132             alpha = 0.3) +
133   scale_fill_manual(name = "cluster",
134                     values = c("#FF6666", "#FFCC66", "#66CC66", "#66
    CCCC")) +
135   labs(title = "hierarchical clustering dendrogram", x = "clients", y =
     "height") +
136   theme_minimal() +
137   theme(axis.text.x = element_blank(), axis.ticks.x = element_blank(),
    legend.position = "right")
138
139 ########## visualize hierarchical ##########
140 ggplot(data, aes(x = Frequency, y = TotalRevenue, color = as.factor(
    hierarchical_cluster), size = CLV)) +
141   geom_point(alpha = 0.7) +
142   labs(title = "hierarchical clustering (clv)", x = "frequency", y = "
    total revenue", size = "clv", color = "cluster") +
143   theme_minimal()
144
145
146 ########## hierarchical statistics ##########
147 hierarchical_stats <- calculate_statistics(data, "hierarchical_cluster"
    )
148 print(hierarchical_stats)
149
150 ###### dbscan ####
151 kNNdistplot <- function(data, k) {
152   distances <- kNNdist(data, k = k)
153   distances <- sort(distances)
154   plot(distances, type = "l", main = "k-NN distance plot",
155        xlab = "points sorted by distance", ylab = "k-NN distance")
156 }
157
158 kNNdistplot(data_normalized, k = 5)
159 abline(h = 0.6, col = "red", lwd = 2)
160 dbscan_eps <- 0.6
161 dbscan_result <- dbscan(data_normalized, eps = dbscan_eps, MinPts = 5)
162 data$dbscan_cluster <- as.factor(dbscan_result$cluster)
163
164 ########## visualize dbscan ##########
165 ggplot(data, aes(x = Frequency, y = TotalRevenue, color = dbscan_
    cluster, size = CLV)) +
166   geom_point(alpha = 0.7) +
167   labs(title = "dbscan clustering (clv)", x = "frequency", y = "total
    revenue", size = "clv") +
168   theme_minimal()
169
170
171 ###### dbscan statistics ##########
172 dbscan_stats <- calculate_statistics(data, "dbscan_cluster")
173 print(dbscan_stats)
174
175 ########## gmm ##########
176 gmm_result <- Mclust(data_normalized)
177 data$gmm_cluster <- as.factor(gmm_result$classification)
```

```
178
179 ##### visualize gmm ##########
180 ggplot(data, aes(x = Frequency, y = TotalRevenue, color = gmm_cluster,
        size = CLV)) +
181   geom_point(alpha = 0.7) +
182   labs(title = "gmm clustering (clv)", x = "frequency", y = "total
        revenue", size = "clv") +
183   theme_minimal()
184
185 ########## gmm statistics ##########
186 gmm_stats <- calculate_statistics(data, "gmm_cluster")
187 print(gmm_stats)
188
189 ########## fuzzy c-means ##########
190 fcm_result <- cmeans(data_normalized, centers = k_optimal, m = 2, iter.
        max = 100)
191 data$fuzzy_cluster <- apply(fcm_result$membership, 1, which.max)
192
193 ########## visualize fuzzy c-means ##########
194 ggplot(data, aes(x = Frequency, y = TotalRevenue, color = as.factor(
        fuzzy_cluster), size = CLV)) +
195   geom_point(alpha = 0.7) +
196   labs(title = "fuzzy c-means clustering (clv)", x = "frequency", y = "
        total revenue", size = "clv") +
197   theme_minimal()
198
199
200 ########## fuzzy c-means statistics ##########
201 fuzzy_stats <- calculate_statistics(data, "fuzzy_cluster")
202 print(fuzzy_stats)
203
204 ########## evaluation (silhouette & ch index) ######
205 dist_matrix <- dist(data_normalized)
206
207 silhouette_kmeans <- silhouette(as.numeric(data$kmeans_cluster), dist_
        matrix)
208 silhouette_hierarchical <- silhouette(as.numeric(data$hierarchical_
        cluster), dist_matrix)
209 silhouette_dbscan <- silhouette(as.numeric(data$dbscan_cluster), dist_
        matrix)
210 silhouette_gmm <- silhouette(as.numeric(data$gmm_cluster), dist_matrix)
211 silhouette_fuzzy <- silhouette(as.numeric(data$fuzzy_cluster), dist_
        matrix)
212
213 cat("silhouette scores:\n")
214 cat("k-means: ", mean(silhouette_kmeans[, 3]), "\n")
215 cat("hierarchical: ", mean(silhouette_hierarchical[, 3]), "\n")
216 cat("dbscan: ", mean(silhouette_dbscan[, 3], na.rm = TRUE), "\n")
217 cat("gmm: ", mean(silhouette_gmm[, 3]), "\n")
218 cat("fuzzy c-means: ", mean(silhouette_fuzzy[, 3]), "\n")
219
220 calinski_harabasz <- function(data, cluster_vector) {
221   k <- length(unique(cluster_vector))
222   if (k < 2) return(NA)
223   data_mat <- as.matrix(data)
224   n <- nrow(data_mat)
```

```r
  overall_mean <- colMeans(data_mat)
  W <- 0
  B <- 0
  for (cl in unique(cluster_vector)) {
    cl_indices <- which(cluster_vector == cl)
    cl_data <- data_mat[cl_indices, , drop = FALSE]
    cl_mean <- colMeans(cl_data)
    W <- W + sum(rowSums((cl_data - cl_mean)^2))
    B <- B + nrow(cl_data) * sum((cl_mean - overall_mean)^2)
  }
  if (W == 0 || k < 2 || (n - k) == 0) return(NA)
  (B / (k - 1)) / (W / (n - k))
}

ch_kmeans <- calinski_harabasz(data_normalized, data$kmeans_cluster)
ch_hierarchical <- calinski_harabasz(data_normalized, data$hierarchical_cluster)
ch_dbscan <- calinski_harabasz(data_normalized, data$dbscan_cluster)
ch_gmm <- calinski_harabasz(data_normalized, data$gmm_cluster)
ch_fuzzy <- calinski_harabasz(data_normalized, data$fuzzy_cluster)

ch_results <- data.frame(
  method = c("k-means", "hierarchical", "dbscan", "gmm", "fuzzy c-means"),
  calinski_harabasz = c(ch_kmeans, ch_hierarchical, ch_dbscan, ch_gmm, ch_fuzzy)
)
print(ch_results)

#######radar charts###########

# function to create radar charts for each cluster
create_radar_charts <- function(data, cluster_column, color, title_prefix) {
  # calculate cluster means only for the 5 main variables
  cluster_means <- data %>%
    group_by(!!sym(cluster_column)) %>%
    summarise(
      TotalRevenueMean = mean(TotalRevenue),
      FrequencyMean = mean(Frequency),
      CLVMean = mean(CLV),
      AvgTransactionMean = mean(AvgTransactionValue),
      LifespanMean = mean(Lifespan)
    )

  cluster_means_normalized <- as.data.frame(scale(cluster_means[, -1]))
  colnames(cluster_means_normalized) <- colnames(cluster_means)[-1]
  cluster_means_normalized <- rbind(
    rep(3, ncol(cluster_means_normalized)),  # max values for scaling
    rep(-3, ncol(cluster_means_normalized)), # min values for scaling
    cluster_means_normalized
  )

  # add labels
  cluster_means_normalized$Cluster <- c("Max", "Min", as.character(
    cluster_means[[cluster_column]]))
```

```
276
277     cluster_ids <- unique(cluster_means[[cluster_column]])
278   par(mfrow = c(2, 3), mar = c(1, 1, 1, 1))  # adjust grid size for up
      to 6 clusters
279   for (i in seq_along(cluster_ids)) {
280     cluster_data <- cluster_means_normalized[cluster_means_normalized$
      Cluster == as.character(cluster_ids[i]), ]
281     radarchart(
282       cluster_means_normalized[c(1, 2, which(cluster_means_normalized$
      Cluster == as.character(cluster_ids[i]))), -ncol(cluster_means_
      normalized)],
283       axistype = 1,
284       pcol = color,
285       pfcol = alpha(color, 0.5),
286       plwd = 2,
287       cglcol = "grey",
288       cglty = 1,
289       axislabcol = "grey",
290       caxislabels = seq(-3, 3, 1),
291       cglwd = 0.5,
292       vlcex = 0.7
293     )
294     title(main = paste(title_prefix, cluster_ids[i]), cex.main = 1)
295   }
296 }
297
298 #all the radar charts
299 create_radar_charts(data, "hierarchical_cluster", "blue", "Hierarchical
      Cluster")
300 create_radar_charts(data, "kmeans_cluster", "red", "K-Means Cluster")
301 create_radar_charts(data, "dbscan_cluster", "green", "DBSCAN Cluster")
302 create_radar_charts(data, "gmm_cluster", "purple", "GMM Cluster")
303 create_radar_charts(data, "fuzzy_cluster", "orange", "Fuzzy C-Means
      Cluster")
```

Bibliography

[1] Wendell R. Smith. «Product Differentiation and Market Segmentation as Alternative Marketing Strategies». In: *Journal of Marketing* 21.1 (1956), pp. 3–8 (cit. on pp. 2, 7).

[2] Thorsten Teichert. «Customer Segmentation Revisited: The Case of the Airline Industry». In: *Journal of Air Transport Management* 14.6 (2008), pp. 329–336 (cit. on p. 2).

[3] K. H. Chung and M. Chen. «RFM Analysis: A Balancing Act Between Business Intelligence and Marketing Intelligence». In: *Industrial Management & Data Systems* 116.2 (2016), pp. 20–33 (cit. on pp. 2, 3, 7).

[4] Mahboubeh Khajvand, Kiyana Zolfaghar, M. Ashrafi, and S. Alizadeh. «Estimating Customer Lifetime Value Based on RFM Analysis of Customer Purchase Behavior: Case Study». In: *Procedia Computer Science* 3 (2011), pp. 57–63 (cit. on pp. 3, 5, 7).

[5] S. Gupta, D. R. Lehmann, and J. A. Stuart. «Valuing Customers». In: *Journal of Marketing Research* 41.1 (2006), pp. 7–18 (cit. on pp. 3, 6, 11).

[6] J. Villanueva and D. M. Hanssens. «Customer Equity: Measurement, Management and Research Opportunities». In: *Foundations and Trends in Marketing* 1.1 (2007), pp. 1–95 (cit. on p. 3).

[7] A. Lemmens and C. Croux. «Bagging and Boosting Classification Trees to Predict Churn». In: *Journal of Marketing Research* 43.2 (2006), pp. 276–286 (cit. on p. 3).

[8] M. K. Pakhira, S. Bandyopadhyay, and U. Maulik. «Validity Index for Crisp and Fuzzy Clusters». In: *Pattern Recognition* 37.3 (2004), pp. 487–501 (cit. on pp. 4, 8).

[9] D. T. Pham, S. S. Dimov, and C. D. Nguyen. «Selection of K in K-Means Clustering». In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 219.1 (2005), pp. 103–119 (cit. on pp. 4, 13).

[10] A. D. Fallis. «Hierarchical Clustering Approaches for Large Scale Data». In: *Journal of Big Data Analysis* 2.1 (2013), pp. 35–41 (cit. on pp. 4, 14).

[11] B. S. Everitt, S. Landau, M. Leese, and D. Stahl. *Cluster Analysis*. 5th. Wiley, 2011 (cit. on pp. 4, 17).

[12] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. «A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise». In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. AAAI Press, 1996, pp. 226–231 (cit. on pp. 5, 16).

[13] J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques*. 3rd. Morgan Kaufmann, 2011 (cit. on pp. 5, 6, 23).

[14] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Springer Science & Business Media, 1981 (cit. on pp. 5, 18).

[15] T. Christy, Noor Raihani B., and Gunawan D. D. «Enhancing RFM Analysis with Weighting». In: *Proceedings of the International Conference on Computing and Informatics* 7 (2018), pp. 84–91 (cit. on p. 6).

[16] Jr. Ward Joe H. «Hierarchical Grouping to Optimize an Objective Function». In: *Journal of the American Statistical Association* 58.301 (1963), pp. 236–244 (cit. on p. 15).

[17] E. Schubert, J. Sander, M. Ester, H.-P. Kriegel, and X. Xu. «DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN». In: *ACM Transactions on Database Systems* 42.3 (2017), pp. 1–21 (cit. on p. 16).