Data Science and Economics
Department of Economics, Management and Quantitative Methods
Department of Computer Science "Giovanni degli Antoni"
Università degli Studi di Milano

Statistical Learning (Unsupervised):

Uncovering Customer Patterns: A Clustering Approach to Online Retail Data

Soudabeh Masoudisoltani - 961506

September 2024

Table of Contents:

Chapter 1: Abstract

The online retail dataset from Kaggle is vastly used for a variety of unsupervised learning tasks, including market basket analysis, anomaly detection, and consumer segmentation..

The data was cleaned and features related to Recency, Frequency, and Monetary Value were engineered to capture key aspects of customer interactions.
We applied K-Means, Hierarchical, and DBSCAN clustering algorithms to both the original and PCA-transformed data.
While K-Means and Hierarchical clustering effectively helped identify clusters in dense data regions, they had challenges in low-density and outlier-prone areas. DBSCAN struggled with this dataset, detecting less clusters and did not provide meaningful segmentation.
PCA reduced dimensionality and led to fewer clusters but did not improve clustering results. The findings highlight the strengths and limitations of each clustering method, providing insights into their applicability based on data density and distribution characteristics.
This analysis underscores the importance of selecting appropriate clustering techniques and dimensionality reduction methods to achieve accurate and actionable customer segmentation.

Link to Public Repository containing R notebook:

link to Repository

Due to the dataset being large, I separately share the link to the dataset:

Online retail dataset

Chapter 2: Problem definition

In this study, our objective is to explore customer purchasing behavior.
Today learning about the behavior of customers and how it has evolved is very crucial for different businesses and their decision making strategies.
To make effective planning in e-commerce, the ability to segment customers based on their shopping patterns is very fundamental.
Companies can tailor their marketing strategies by categorizing customers into groups. They can take advantage of the results also for personalizing the shopping experience, enhancing customer loyalty and ultimately growing their revenue.
Given the importance of the findings of this study, we will perform a customer segmentation analysis using unsupervised learning techniques on the public available dataset from Kaggle.
We will be looking for underlying meaningful patterns to finally be able to find natural groupings among customers.
Some research questions that will be answered along the way are:

- Based on purchasing behavior, what are the distinct customer segments?
- Characterization of the segments in terms of their purchasing patterns, frequency, and contribution to revenue.
- Insights that can be drawn from the identified segments that could allow for informed business strategies in marketing, inventory management, and customer relationship management.

Chapter 3: Data and Preprocessing

3.1. Data

For this unsupervised task we have analyzed the available online retail dataset from Kaggle.

This is a real retail transaction dataset of two years (from 01/12/2009 until 09/12/2011)

Many of the customers are wholesalers.

This dataset consists of 541909 observations of 8 variables.

3.1.1. Attributes' information

1. InvoiceNo:

Invoice number which is a char/string variable.. A 6-digit unique identifier assigned to each transaction. If this code starts with the letter 'c', it indicates a cancellation.

2. StockCode:

Product code which is string. A 5-digit unique identifier assigned to each distinct product.

3. Description:

Product name, Nominal.

4. Quantity:

The quantities of each product per transaction, this is an integer variable.

5. InvoiceDate:

Invoice date and time. Numeric. The day and time when a transaction was generated.

6. UnitPrice:

Unit price. Numeric. Product price per unit in sterling

7. CustomerID:

Customer number. Nominal. A 5-digit unique identifier assigned to each customer. This column could contain missing values.

8. Country:

Country name, String. The name of the country where a customer resides

3.2. Pre-processing

To clean data:

- To assess the completeness and accuracy of the dataset, the number of missing values has been checked and is 135080 along with 5225 duplicated values for CustomerID. Rows with duplicates and null values have been dropped.
- 'InvoiceDate' has been converted into POSIXct datetime object format.
- 'Country' and 'Customer ID' have been converted into factors to make possible the representation and handling of categorical variables. Transforming these columns into factors will improve their suitability for use in statistical models and allow them to utilize them in categorical studies like segmentation.
- Due to the data being very large, we have chosen a 10% random sample with minimum 10000 and maximum 'actual number of rows'
- Created a new feature called 'TotalPrice' which is acquired by multiplying 'Unit price' by the 'Quantity'. This feature represents the total value of each transaction. which is valuable for analyzing sales performance, identifying high-value transactions, and conducting customer and revenue analysis.
- Key metrics for every client were combined to generate a new dataset called 'customer_data' containing 'Recency', 'Frequency' and 'MonetaryValue' useful for client segmentation, targeted marketing, and improving customer interaction tactics.

3.3. Transformation and Data visualization

This section helps in uncovering patterns, trends, and insights that are critical for any further analysis or modeling.

From the dataset we have derived three new features as below:

1. Recency: number of days between a customer's first and last transaction. This feature shows how recently the customer has had an interaction with the store.
2. Frequency: number of unique transaction ids, this feature indicates how often a customer has made a purchase.

3. MonetaryValue: Sum of total value of all purchases made by each customer, in other words, Sum of all the money spent by each customer.

3.3.1. Data visualization

Figure 3.1 and Figure 3.2. Below shows the distribution and transformed log of variables in terms of Recency, Frequency and Monetary value.
Several modifications were applied to the attributes Recency, Frequency, and MonetaryValue in order to address skewness and stabilize variance:
To manage a large range of data and stabilize variance, log transformation (log1p()) was utilized. For characteristics with moderate skewness, the square root transformation was used, and for features with negative values, the cube root transformation. Plotting the altered features' histograms allowed one to see how distributed they were. These plots, which are stored as picture files, help to spot patterns or anomalies by showing how each alteration impacts the distribution of the data.

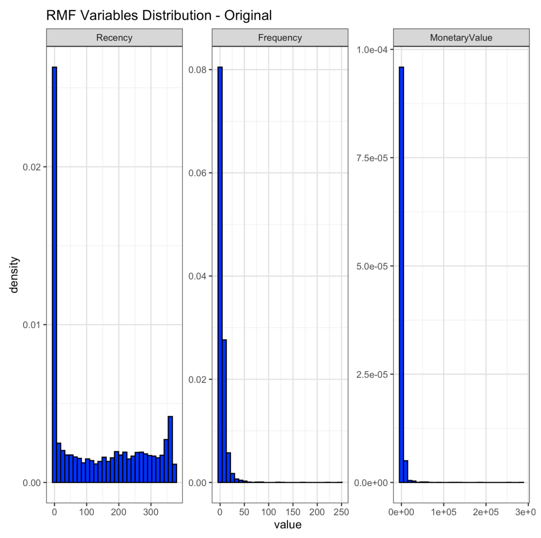


Figure 3.1. Variable distribution

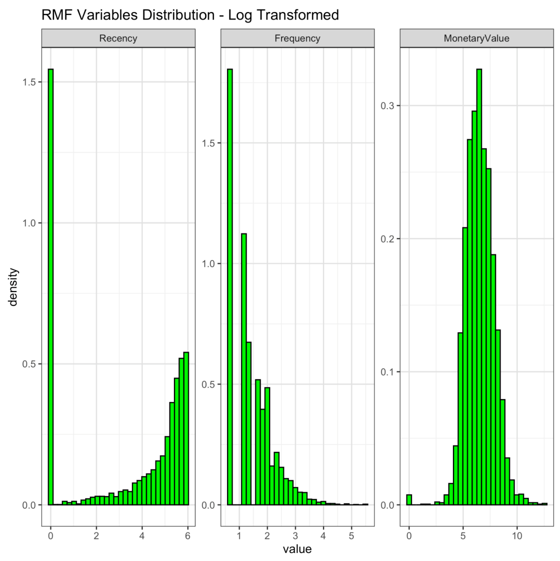


Figure 3.2. Transformed log of Variable

The original distribution plot helps as reference to allow comparing the effects of the transformations.

This plot shows the raw distributions before transformations. It highlights skewness, outliers, and extreme values. Comparing this with the transformed plots will show how the transformations have altered the data distribution, resulting in more normality and reduced skewness.

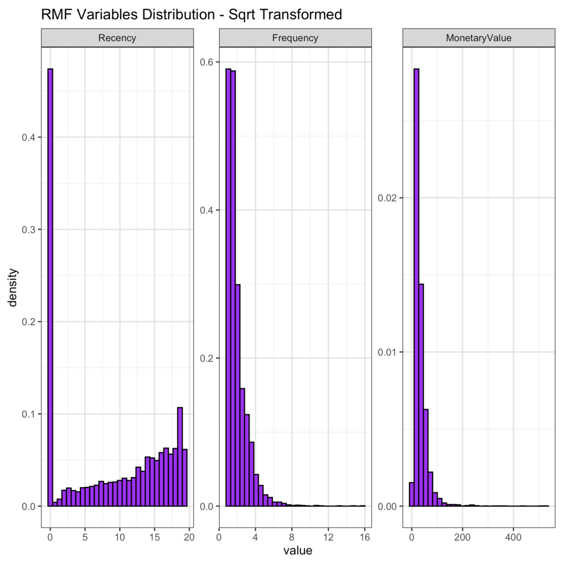


Figure 3.3. Square root transformed distribution

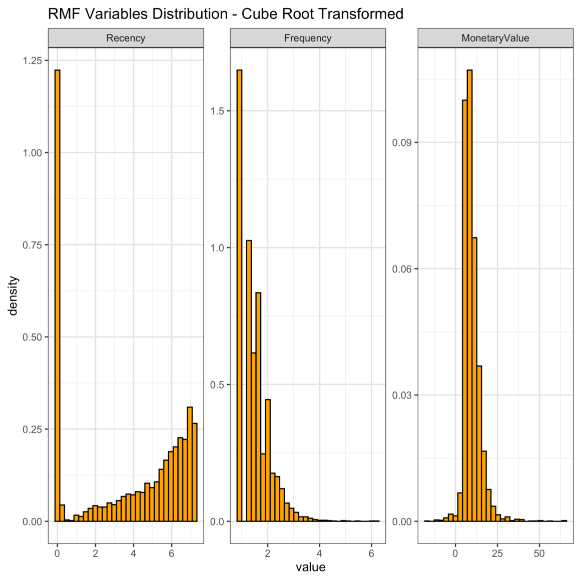


Figure 3.3. Cubic root transformed distribution

Log Transformation:

increases the visibility of patterns in data with high variability by reducing skewness and compressing large values.

Square Root Transformation:

For moderate ranges of values, this transformation moderates skewness and offers a balanced view without making drastic alterations.

Cube Root Transformation:

Provides a balanced distribution by managing both positive and negative values and moderate skewness.

Chapter 4: Unsupervised Models implementation

Firstly, we have randomly selected 10% from the datasets because for very large datasets such as the one we are working with here, setting the sample size at 100,000 rows helps manage computational resources and speeds up analysis and if the dataset has fewer than 100,000 rows, using the entire dataset ensures that no data is excluded from the analysis.
So to ensure a manageable sample size and optimize computational efficiency, the sample size was set at 100,000 rows. This allows for a sufficiently large sample from larger datasets while using the entire dataset when fewer than 100,000 rows are available. ensuring both efficiency and completeness in the analysis.
Here we have set the sample size to be the smaller value between 100,000 and the total number of rows in the dataset.
Secondly, to ensure that all features contribute equally to the analysis, the ‘Recency’, ‘Frequency’ and ‘Monetary value’ columns in the sampled dataset were standardized.
Through this technique, each feature is scaled to have a standard deviation of one and is centered around zero. For distance-based analysis techniques, standardizing the data is crucial to prevent features with disparate scales from unduly influencing the outcomes.

In this study, three unsupervised algorithms are utilized after having found the optimal number of clusters for this dataset which is crucial for meaningful results:

- K-means clustering
- DBScan clustering
- Hierarchical clustering

4.1. Choosing the optimal number of clusters

In clustering analysis, calculating out the ideal number of clusters is essential.
It assists you in determining the optimal number of clusters to employ, which has a big impact on how well the clustering results may be interpreted. There exist multiple techniques for ascertaining the ideal quantity of clusters.
We have performed the following methods:

4.1.1. Elbow method

The Elbow method helps evaluate the number of clusters we need by their performance on how well the clusters fit the data. This is the steps taken by elbow method:

- For a chosen clustering technique (K-means, for example) we run it for varying numbers of clusters (K=1, 2, 3,...).
- The "within-cluster sum of squares" (WCSS), which quantifies how near the data points are to their cluster centers, is calculated for each number of clusters.
- The WCSS is plotted against the total number of clusters. As shown by figure 4.1. Plots typically take the shape of an arm, with the reduction in WCSS beginning to lessen at the "elbow" point. This argument proposes striking a balance between having an excessive number of clusters (overfitting) and too few clusters (not reflecting the structure of the data).

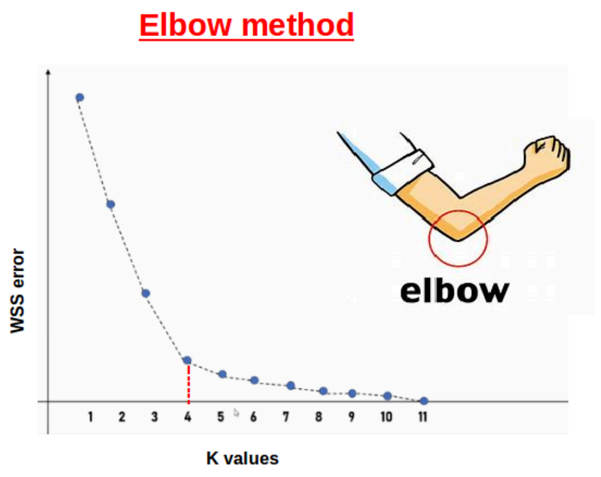


Figure 4.1. Elbow method example

The objective of using this method is that without overcomplicating the model, it offers a visual means of determining how many clusters best reflect the underlying patterns in the data.

4.1.1.1. Elbow method implementation

We calculated the total within-cluster sum of squares (WCSS) for a range of cluster sizes from 1 to 15 in order to establish the ideal number of clusters for our k-means clustering.
The Elbow Method was visualized by plotting the results.
The graphic indicates that as the number of clusters rises, the WCSS decreases. The 'elbow' of the curve, which is the point at which adding additional clusters results in a small loss in WCSS, is the ideal number of clusters.
The elbow method is implemented on our dataset, taking 1 to 15 clusters into consideration, and the outcome is presented in figure 4.2. Below

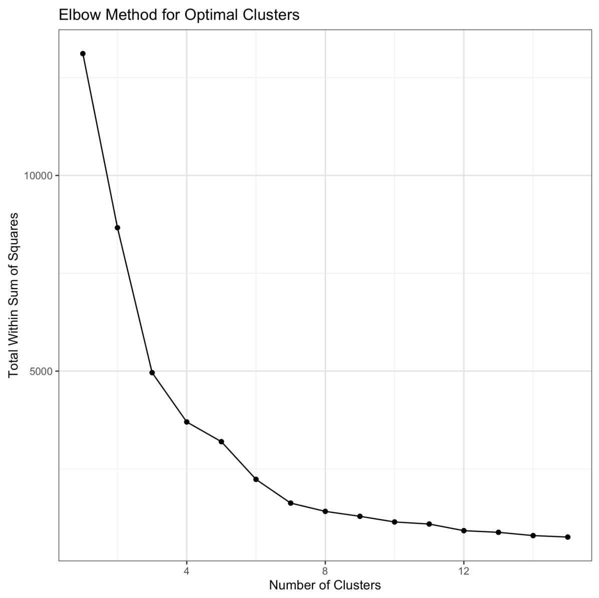


Figure 4.2. Elbow method for clusters

Looking for a point on the plot where the rate of decrease in WCSS slows down. This point is called the "elbow" and represents a good balance between the number of clusters and the WCSS. Adding more clusters beyond this point results in only marginal improvements in WCSS.

According to the results of the Elbow method, K = 4 is the number of clusters to be considered.

4.1.2. Silhouette Method

Similarly to the Elbow method, The Silhouette Method determines the ideal number of clusters by evaluating how well-separated the clusters are and it works by the steps below:

- The Silhouette score calculates the degree to which a data point resembles its own cluster in relation to other clusters. There is a range of -1 to 1.
- The point is poorly matched to nearby clusters and well-matched to its own cluster when the score is around +1. When a point has a score of almost zero, it is either on or very near the decision border separating two adjacent clusters. A low score suggests that the point may have been inadvertently added to the incorrect cluster.
- The number of clusters with the highest average Silhouette score is the ideal number of clusters. This is determined by calculating the average Silhouette score for various cluster sizes.

The purpose of this method is that it offers a numerical indicator of how suitable the clustering is. The clusters are more internally coherent and well-separated when the average Silhouette score is higher.

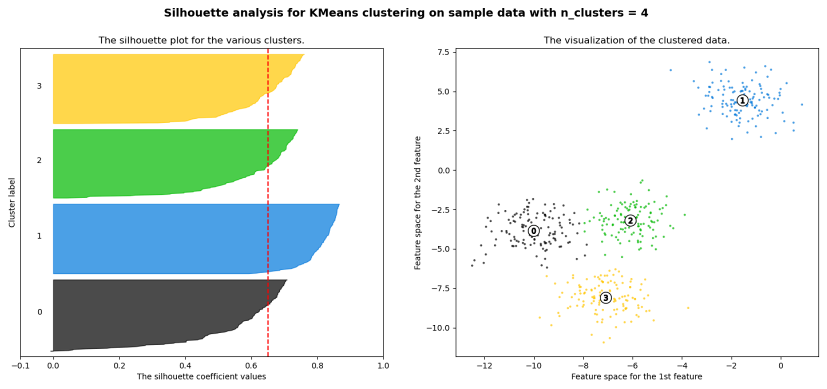


Figure 4.2. Silhouette method example

4.1.2.1. Silhouette method implementation

To find the ideal number of clusters for the k-means clustering analysis, the Silhouette Method was utilized. The average silhouette width for every clustering configuration was plotted after silhouette widths for cluster numbers ranging from 2 to 15 were calculated. The peak in the average silhouette width, which represents the most well-defined and unique clustering solution, indicates the ideal number of clusters. The plot offers a visual aid for determining how many clusters will optimize the distance between them.

The Silhouette method is implemented on our dataset, taking 2 to 15 clusters into consideration, and the outcome is presented in figure 4.3. Below

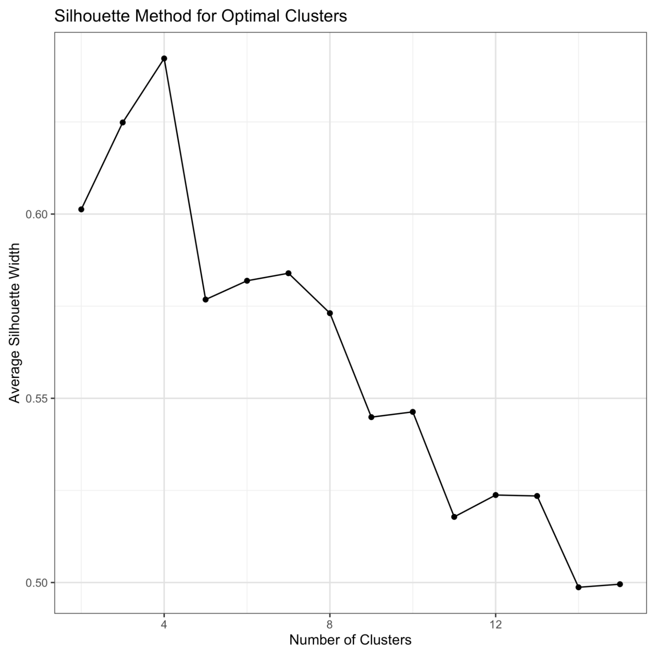


Figure 4.3. Silhouette method

The plot demonstrates the number of clusters that give the highest average silhouette width. This peak suggests the optimal number of clusters, providing the most distinct and well-separated clustering solution.

According to the results of the Silhouette method, K = 4 is the number of clusters to be considered.

4.2. K-means clustering

This is an unsupervised learning algorithm where the data is not labeled. Kmeans divides the objects into clusters that have similarities and are different from objects of other clusters.
This algorithm works by taking into consideration the number of clusters assigned to it and randomly assigning a center for each cluster. Data points are assigned to a cluster which has a closest center. It then calculates the average of all points in the cluster and moves the cluster center closest to its location. The steps mentioned above are repeated until there won't be significant movements in the cluster centers and labels won't change anymore.
The ultimate goal is to group data points in clusters where they share similar characteristics.

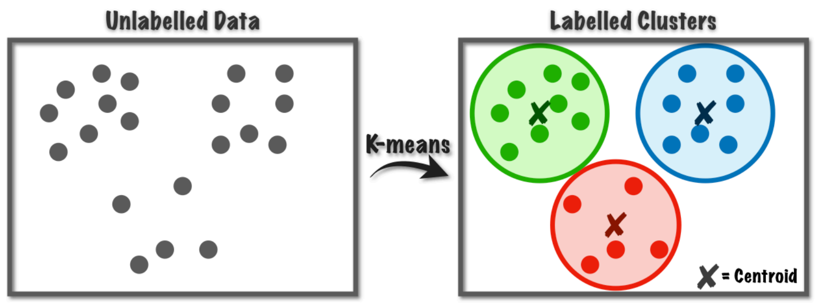


Figure 4.4. K-means clustering example

4.2.1. K-means clustering implementation

K-Means Clustering: Applied with the optimal number of clusters identified from the silhouette analysis. The clustering result segments the data into distinct groups based on customer behaviors.
Cluster Summaries: Summary statistics of each cluster were generated to understand the distribution of customers in terms of Recency, Frequency, and Monetary Value.
Visualizations: Scatter plots of Recency vs. Frequency illustrate the clustering results, and the cluster centers plot shows the central points of each cluster, helping to interpret the clustering configuration.

4.3. Hierarchical clustering

Data points can be grouped into clusters using the hierarchical clustering approach, which groups them based on similarity in a step-by-step (hierarchical) manner. This is how it operates:

- Begin by treating each point as a separate cluster: Each data point is initially thought to be a separate cluster.
- Combine Nearest Clusters: Combine the two clusters that are most similar to one another and nearest to one another to form a single cluster.
- Again: Once every point has been merged into a single, large cluster, keep merging the nearest clusters together.

The ultimate product is a dendrogram, which is a tree-like structure that displays the sequence in which the points were combined. Depending on the degree of similarity you want each cluster's points to have, you can chop the tree at different levels to create varying numbers of clusters.

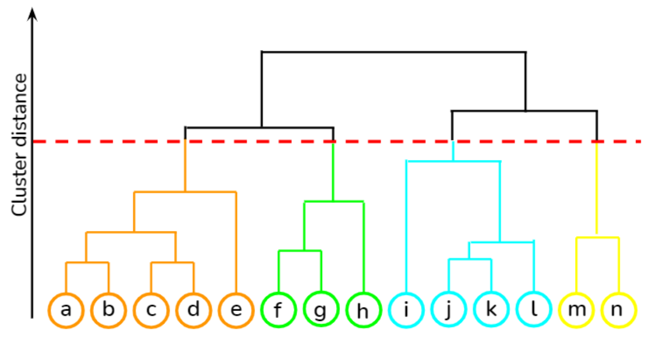


Figure 4.5. Hierarchical clustering example

4.3.1. Hierarchical clustering implementation

Hierarchical Clustering: A hierarchical clustering tree is created by applying Ward's approach. After determining the ideal number of clusters, the tree was sliced to create these clusters. All of the sampled data's observations were categorized into one of the hierarchical groups.
Cluster assignment: All of the sampled data's observations were categorized into one of the hierarchical groups.

Visualizations: The distribution of customers among clusters is shown by scatter plots of recency vs frequency. Understanding the clustering quality and segment characteristics can be gained by contrasting these visuals with those from alternative clustering techniques (K-Means, DBSCAN).

4.4. DBScan clustering

A clustering technique called DBSCAN (Density-Based Spatial Clustering of Applications with Noise) clusters data points according to their density. This is how it operates:

- Determine Core Points: Count the number of additional points that are within a given distance (referred to as the "epsilon" distance) for each point. When a point has more neighbors than the required minimum, it is referred to as a "core point."
- Create Clusters: Neighboring core points are combined together to form clusters. Points are added to a cluster if they are located within an epsilon distance of a core point.
- Managing Outliers "Noise" or "outlier" points are those that don't belong in any cluster since they don't have enough neighbors or aren't near any core points.

Unlike approaches that require you to define the number of clusters in advance, DBSCAN is good at handling outliers and identifying clusters of various forms.

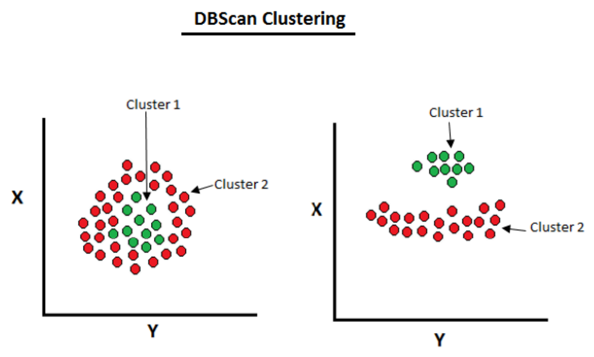


Figure 4.6. DBScan example

4.4.1. Hierarchical clustering implementation

DBSCAN Clustering: Applied with ‘eps = 0.5’ specifying the maximum distance between two samples for one to be considered as in the neighborhood of the other. This is a crucial parameter that determines the density threshold.

And min ‘Pts = 5’ which indicates the minimum number of points needed to form a dense region.

Points that are in a dense region (i.e., have at least min Pts neighbors within eps) are considered core points to identify clusters based on density.

This method does not require specifying the number of clusters beforehand and can detect outliers.

Cluster Assignments: Each observation in the samples data was assigned to a cluster, with some points potentially classified as noise.

Visualizations: Scatter plots of Recency vs. Frequency illustrate how the data is grouped by DBSCAN, including clusters and outliers.

Chapter 5: Analyzing the clusters and results

After having explained and formed the models that are to be used in this project in the previous chapter, this chapter will look at the results of each model and present the final cluster in each case. But first we will apply PCA on each of the three algorithms

5.1. PCA

A method for reducing data dimensionality while retaining as much variability (information) as feasible is principal component analysis, or PCA. It is frequently used in data analysis and machine learning to enhance algorithm performance, simplify datasets, and make them easier to display. PCA works by the steps below:

- Standardize the Data
- Compute the Covariance Matrix
- Compute the Eigenvectors and Eigenvalues
- Sort Eigenvalues and Select Principal Components
- Transform the Data

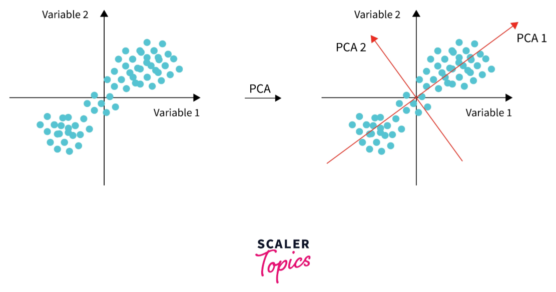


Figure 4.7. PCA example

There are several reasons to use PCA:

1. Reduction of Dimensionality: Principal component analysis (PCA) lowers the number of characteristics, or dimensions, in the data, facilitating analysis and visualization, particularly with high-dimensional datasets.Noise reduction: PCA can assist in removing noise or less significant information from data by concentrating on the major components that capture the greatest variance.

2. Enhanced Algorithm Performance: In machine learning, employing fewer, more illuminating characteristics can lower the chance of overfitting and enhance algorithm performance.
3. Visualization: By distilling the data to its top two or three principle components, principal component analysis (PCA) reduces complicated, high-dimensional data to a 2D or 3D representation.

5.2. K-means clustering plot with vs without PCA

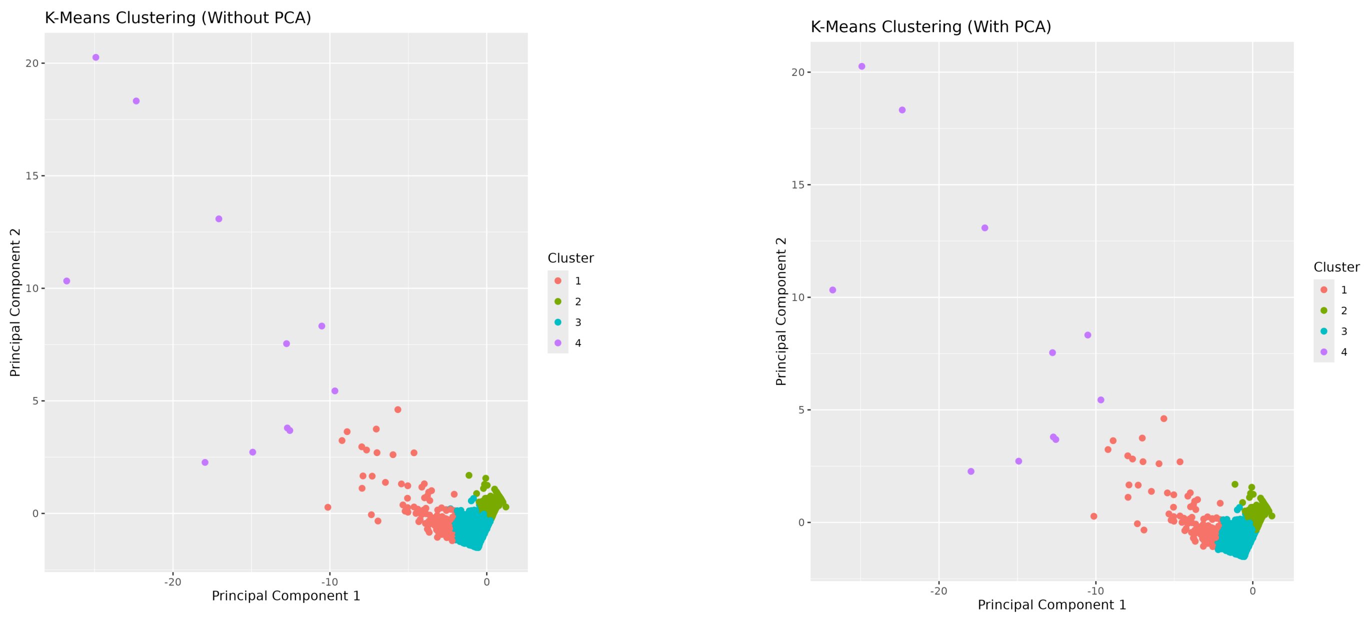
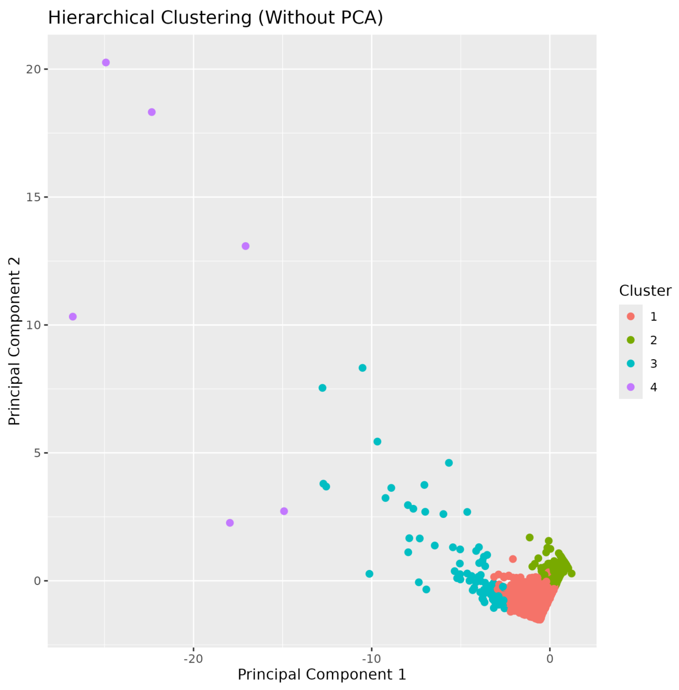


Figure 4.8. K-means clusters with and without PCA

Distinct consumer segments are shown by the summaries and visualizations, offering insights into various customer profiles. Personalized consumer interaction initiatives and tailored marketing tactics can be informed by these insights.

5.3. Hierarchical clustering plot with vs without PCA



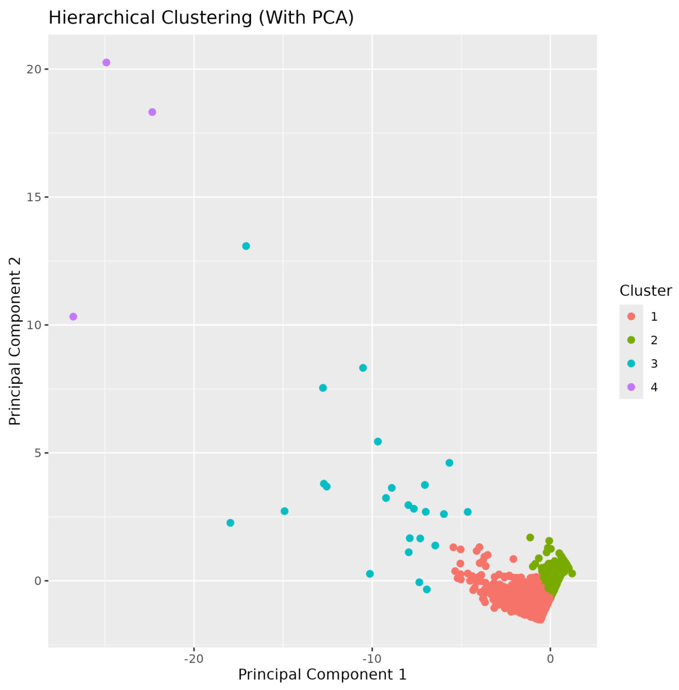


Figure 4.9. Hierarchical clusters with and without PCA

5.4. DBScan clustering plot with vs without PCA

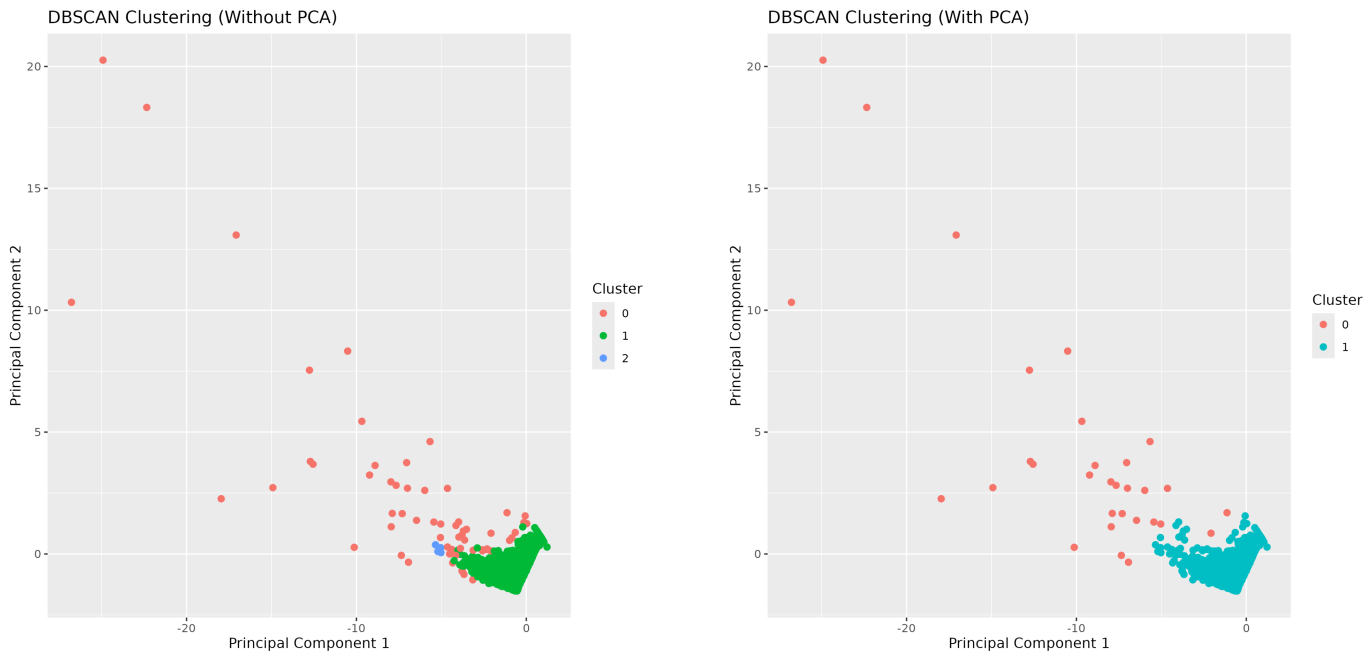


Figure 4.10. DBScan clusters with and without PCA

Cluster Identification: By using a density-based methodology, DBSCAN generates clusters according to local densities.

It can find clusters that other approaches would overlook, of different sizes and forms. The places where the data points do not fulfill the density requirements for clustering are indicated by the noise points.

Comparison: Understanding various clustering perspectives and validating the robustness of the results are aided by comparing DBSCAN results with K-Means and Hierarchical clustering.

Chapter 6: Conclusion

In this project, various clustering algorithms were applied to segment the data from online retail stores.

The approach involved:

- Data Preparation and Feature Engineering
- Data Transformation and Visualization
- Sampling and Scaling
- Clustering Analysis with and without PCA

While applying clustering, we decided to perform once clustering with PCA and another time without it:

PCA:

K-Means Clustering: Based on the initial feature set, the visualizations displayed clusters with distinct groupings and possible overlaps.

Hierarchical Clustering: A dendrogram illustrating the merging of clusters was produced based on the results, which showed that clusters were established based on hierarchical linkages among data points.

DBSCAN Clustering: By highlighting areas of different densities, the density-based method successfully detected clusters and outliers.

Without PCA:

K-Means Clustering: Applying K-Means to PCA-transformed data shows how dimensionality reduction can simplify the clustering process and highlight distinct clusters in a lower-dimensional space.

Hierarchical Clustering: Using principal component analysis, clustering in PCA space revealed hierarchical links in a smaller feature set and offered insights into how clusters are formed.

Applying DBSCAN to PCA data demonstrated how density-based clustering recognizes noise and clusters in the principal component space while adjusting to the lower dimensions.

Overall, In areas with a high data density, both K-Means and Hierarchical clustering demonstrated comparable performance. Both approaches successfully found clusters in these regions where data points are densely packed.
issues with Low-Density Regions:
Hierarchical Clustering: Usually more responsive to distant and low-density data points. In sparse areas, this method's clustering based on the proximity of surrounding points may produce deceptive results. Therefore, distant values may either break up the general clustering structure or establish their own clusters.
K-Means Clustering: Because its goal is to minimize the sum of squared distances from the cluster centroids, it is more vulnerable to distortion by outliers. Results may become less accurate as a result of outliers' large effects on the cluster centers.
DBSCAN Clustering: DBSCAN performed poorly and found fewer clusters in this dataset. Its density-based method works better with datasets that have distinct fluctuations in density, which the current data may not have.
Impact of PCA (Principal Component Analysis):
By using PCA, the data's dimensionality was decreased, which predicted fewer clusters. In this instance, PCA did not improve the clustering outcomes, though. For this specific dataset, the clustering performance was not enhanced by the reduction in dimensionality.

6.1. Key findings

1. High-Density Data: K-Means and Hierarchical clustering techniques both produced comparable outcomes in high-density data regions by successfully identifying clusters in these locations.
2. Low-Density Challenges: Because hierarchical clustering is sensitive to proximity, it may provide erroneous groupings in low-density or sparse locations. Outliers can also affect K-Means, causing results and cluster centroids to be distorted.
3. DBSCAN Limitations: DBSCAN performed poorly on this dataset and found fewer clusters, suggesting that it might not be appropriate for datasets lacking distinct density-based clusters.

4. Impact of PCA: Using PCA decreased dimensionality and predicted fewer clusters, but it had no positive effect on clustering performance in this dataset, indicating that different responses to PCA.
5. Suggestions: DBSCAN is the best for datasets with discrete density variations, K-Means performs well for dense data but is susceptible to outliers, and hierarchical clustering is less appropriate for datasets with vast variations or sparse distributions. While PCA can make clustering simpler, results may not always be improved.