Data Mining For Customer Profiling and Sales Prediction

Bachelor Thesis

Author:
Adrian Predescu

Supervisors:
Dilek Dustegor
Huy Cuong Truong

University of Groningen
Netherlands
August 2024

Abstract

In the age of big data, organizations are frequently overwhelmed by the sheer volume of data they have to handle. The real challenge, however, is not in gathering the data, but in effectively using it. The process of converting raw data into structured, actionable insights is intricate and does not always yield successful results. In this Bachelor's Thesis, my goal is to investigate the use of data mining methods for customer profiling and sales predicting. This involves a deep dive into various data mining algorithms, including clustering algorithms like K-means, hierarchical clustering, and vector quantization. Moreover, I will evaluate their efficacy in interpreting customer behavior and predicting future sales patterns. This research will illustrate how these techniques can enable businesses to customize their strategies to satisfy customer demands and maximize sales. Additionally, it will add to the expanding knowledge base in the realm of data science and its practical uses in business intelligence. In essence, it aims to pave the way for harnessing data-driven insights in strategic decision-making.

Contents

List of Figures

List of Tables

1 Introduction

In the past years, the field of information technology has seen an exponential increase in the volume of data generated and processed [1]. Every day, our computer networks, the World Wide Web, and a variety of data storage devices are flooded with thousands of terabytes of data stemming from diverse sectors including business, society, science and engineering, medicine, and nearly all other facets of our daily lives [1]. This explosive growth of available data volume can be attributed to the digitization of our society and the fast development of powerful data collection and storage tools. Such tools allow businesses worldwide to generate vast data sets, including sales transactions, stock trading records, product descriptions, sales promotions, company profiles and performance, and customer feedback. For example, some of the largest retailers in the world, such as Amazon, handle tens of millions of transactions per week at thousands of branches around the world [2].

However, having enormous amounts of information does not mean you can easily understand and efficiently make use of it. Transforming the data into organized knowledge can be a cumbersome process that does not always succeed, especially when trying to foresee future events by drawing insights from past occurrences. One application of this idea is in the industry sector, where organizations want to maximize profits by predicting the buying actions of their customers. Therefore, the following question arises: Is there an efficient and reliable algorithm for anticipating sales based upon previous sales figures?

The motivation behind this question lies within the ever-evolving corporate landscape. It is the dream of every enterprise to know ahead of time what their clients will be buying, because it allows them to adapt their strategies, improve resource allocation, and enhance patron satisfaction. For example, the company taking part in my research wishes to estimate its sales in order to keep a small stock of prebuilt products to help in honoring larger orders with tight deadlines.

To answer this question, I will explore several topics related to computer science, such as data mining, machine learning, and clustering. Firstly, I start by presenting the current state-of-the-art data mining techniques in section 2. Then, I will describe the details of my approach and implementation in section 3. Afterwards, the experimental results and discussion (4th and

5th sections) of the research are outlined, defining the efficacy and potential applications of the proposed methods. Finally, the paper concludes in section 6 with a summary of the findings and suggestions for future research directions.

2 State Of The Art

The intriguing concept of data mining can be seen as a natural outcome of the development of information technology, which, since the 1960s, has evolved continuously and systematically from the early file processing systems to the sophisticated and capable database management systems [3]. The extensive volume of data contained within these databases has led researchers to devise a process for analysing it in order to find potentially useful relationships and patterns. Today, this process is commonly known as "data mining". Data mining incorporates many techniques from other domains such as statistics, machine learning, predictive analytics, visualization, and so on [4]. Its application-driven nature makes it particularly appealing to organizations looking to extract valuable insights from their data.

2.1 Customer Profiling

To achieve this goal, the idea of customer profiling or customer segmentation was introduced as a marketing tool that allows companies to build differentiated sales strategies. Customer profiling involves dividing customers into distinct, meaningful, and homogeneous subgroups based on various attributes and characteristics [6]. Creating the customer groups is performed through clustering, an unsupervised learning technique which can be broadly classified into four categories: centroid-based methods (K-Means Clustering) [5], connectivity-based methods (Agglomerative Hierarchical Clustering) [5], density-based methods (Density-based spatial clustering of applications with noise; DBSCAN for short) [5] and distribution-based methods (Gaussian Mixture Model) [5]. Each of these techniques uses several algorithms with different advantages and disadvantages that are presented in the subsequent chapters. Despite the multitude of available methods, the architecture for client segmentation shares two phases in common. First, in the data preparation phase, the data is collected and cleansed by removing any noise (the incomplete, missing or irrelevant records are removed from the set). In the second phase, cluster construction and profiling takes place [5]. Figure 1 provides a clearer visualization of the mentioned architecture.

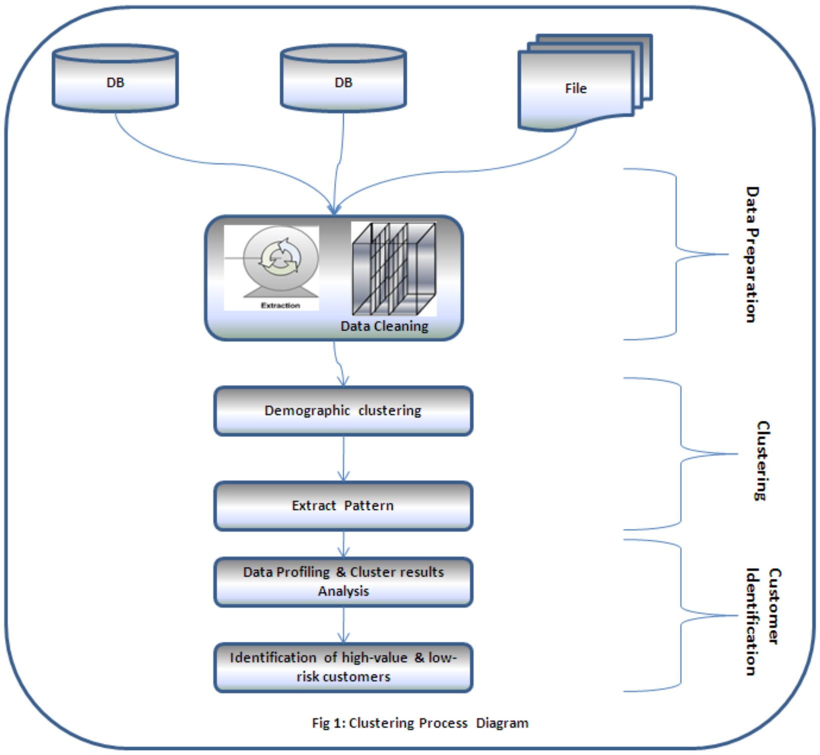
DB
DB
File
Extraction
Data Cleaning
Demographic clustering
Extract Pattern
Data Profiling & Cluster results Analysis
Identification of high-value & low-risk customers
Data Preparation
Clustering
Customer Identification
Fig 1: Clustering Process Diagram

Figure 1: Clustering Process Diagram [5]

In the next subsections, I will delve deeper into the specifics of feature selection and extraction, which are critical in handling high-dimensional data for effective clustering. I will also explore various distance and similarity measures, which play a crucial role in determining cluster assignments (grouping algorithms are distance-based). This will be followed by a detailed examination of the four different clustering methodologies, explaining how they work and their respective strengths and weaknesses.

2.1.1 Feature Selection And Extraction

In high-dimensional data, clustering algorithms can suffer from the curse of dimensionality—the phenomenon where the feature space becomes increasingly sparse as the number of dimensions grows relative to a fixed-size training set [11]. This sparsity complicates the visualization and interpretation of results, while also degrading the performance of the algorithms. To effectively mitigate these issues, two techniques, known as feature selection and extraction, become crucial.

Feature selection involves identifying and selecting the most relevant features from the original data set to enhance the predictive accuracy of the model. This process can be done manually, based on domain knowledge, or automatically using algorithms. Automatic methods are divided into three categories: filter methods, wrapper methods, and embedded methods. Filter methods rank features based on statistical measures and are generally faster and less computationally expensive. Wrapper methods use a predictive model to score feature subsets, while embedded methods perform feature selection during model training, making them more suited to specific algorithms [20].

In contrast, feature extraction involves creating new features from the original data set, often in a lower-dimensional space. Among the various procedures available for feature extraction, one particularly popular approach stands out: Principal Component Analysis, often abbreviated as PCA. This technique is widely used due to its effectiveness and simplicity. PCA is, at its core, a statistical technique that uses orthogonal transformation to convert a set of potentially correlated variables into a set of linearly uncorrelated variables, known as principal components [12]. This transformation ensures that the first principal component captures the maximum possible variance in the data, while each subsequent component captures the highest variance possible, subject to the condition that it is orthogonal to the preceding components, thereby forming an uncorrelated orthogonal basis set [12]. The input data to the transformation is usually standardized, since PCA is sensitive to the relative scaling of the original variables.

The main steps of the PCA algorithm can be summarized by the following [13]:

1. Standardize the data: Since PCA is affected by the scales of the variables, the data is standardized so that each variable contributes equally to the analysis.

2. Compute the covariance matrix: This matrix provides insights into how the variables in the data set relate to one another.

3. Calculate eigenvalues and eigenvectors: These are calculated from the covariance matrix. The eigenvectors (principal components) determine

the directions of the new feature space, and the eigenvalues determine their magnitude.

4. Sort eigenvalues and corresponding eigenvectors: The eigenvector with the highest eigenvalue is the first principal component that captures the most variance in the data. The eigenvector with the second highest eigenvalue is the second principal component, and so on.

5. Transform the original data set: The final step is to transform the original data set via the selected principal components to obtain a new data set of reduced dimensionality.

By incorporating feature selection and extraction methods into the clustering process, companies can enhance the customer segmentation process. These techniques simplify the data and help in identifying the most relevant features, which can contribute to more accurate and meaningful clusters. As a result, this improved clustering can support more targeted marketing strategies and potentially enhance customer satisfaction.

2.1.2 Distance And Similarity Measures

In the following sections, the terms "distance" or "distance function" are used when describing various algorithms. They refer to measurements of the distance (dissimilarity) or similarity between a pair of objects, an object and a cluster, or a pair of clusters based on their feature values or attributes [7]. The choice of distance or similarity measure depends on the nature of the data and the specific clustering algorithm being employed. For example, while common distance measures for continuous data include Euclidean distance and Manhattan distance, cosine similarity and Jaccard Index are often more appropriate when working with categorical data [8]. Euclidean distance, the most-widely used, is a measure that calculates the straight-line distance between two points in a space, its generalized formula being the following [8]:

$$d = \sqrt{\sum_{i=1}^{n} (x_{2i} - x_{1i})^2} \tag{1}$$

where n is the number of dimensions and x_{1i} and x_{2i} are the coordinates of the two points in the i-th dimension.

2.1.3 Centroid-Based Clustering

The fundamental concept of centroid-based clustering algorithms is to consider the centroid of data points as the center of the respective cluster. K-means and K-medoids are the two most well-known algorithms of this type, each with its own unique approach.

K-means operates on the principle of minimizing the variance within each cluster. It does this by continually updating the cluster's center, which is represented by the centroid of the data points. This centroid is the arithmetic mean of all the points in the cluster. The iterative calculations continue until a certain convergence criterion, such as no change in the cluster assignments or a maximum number of iterations, is satisfied [8].

On the other hand, K-medoids is an enhancement of K-means designed to handle discrete data and to be more robust to noise and outliers, but is computationally more expensive. Instead of using the mean of data points as the centroid (which may not correspond to an actual data point), K-medoids selects an existing data point from the cluster to represent it. This representative point, known as the medoid, is the data point that minimizes the sum of dissimilarities (e.g., distances, as explained in section 2.1.2) between itself and all other points in the cluster. In other words, it's the most centrally located point in the cluster according to the chosen distance measure [8].

Having discussed the K-medoids algorithm, we now turn our attention to the K-means algorithm, which will be used as a representative of centroid-based clustering. The main steps of the K-means algorithm are as follows [7]:

1. Initialize a K-partition randomly or based on some prior knowledge and then calculate the cluster prototype matrix $M = [m_1, ..., m_K]$, where each m_i represents the centroid (mean) of the i-th cluster.
2. Assign each object in the data set to the nearest cluster C_w by finding the minimum distance:
   $x_j \in C_w$ if $||x_j - m_w|| < ||x_j - m_i||$, for $j = 1, .., N$, $i \neq w$ and $i = 1, ..., K$.
3. Recompute the cluster prototype matrix based on the current partition by recalculating the centroids of the clusters.
4. Repeat steps 2 and 3 until there is no change for each cluster.

The general advantages and disadvantages of partition-based clustering (using K-means as reference) are the following [7]:
Advantages: relatively low time complexity (worst case is linear); works very well for compact and hyperspherical (well-separated) clusters.
Disadvantages: no efficient and universal method for identifying the initial partitions and the number of clusters K; the iteratively optimal procedure of K-means cannot guarantee convergence to a global optimum; sensitive to outliers and noise.

2.1.4 Density-Based Clustering

Density-based clustering algorithms, such as DBSCAN, OPTICS, and Mean-shift, operate on the principle that data in high-density regions of the data space belong to the same cluster [8]. DBSCAN, the most well-known among these, creates a new cluster from a data object (an individual data point within a data set) by absorbing all objects in its neighborhood, which must satisfy a user-specified density threshold. Therefore, it works on two key parameters: the radius of the neighborhood (often denoted as *eps*) and the minimum number of points required to form a dense region (often denoted as *MinPts*) [7].

The main steps of the DBSCAN partitioning algorithm can be outlined as follows [7]:

1. Choose an arbitrary data point P from the data set.
2. For each P, determine the neighborhood N(P) of P by finding all data points within distance *eps* from P. If $|N(P)| >= MinPts$, label P as a core point. If $|N(P)| < MinPts, |N(P)| \neq 0$, label P as a border point.
3. Form the clusters by performing:
    - Initialize an empty list to store clusters.
    - For each core point P that has not been visited:
        - Create a new cluster.
        - Perform a depth-first search (DFS) or similar traversal:
            - Add P to the current cluster.
            - For each neighboring point P' in N(P):

        - If P' is a core point and not yet visited, recursively add its neighbors to the cluster.
        - If P' is a border point, include P' in the current cluster.
    - Mark all points in the cluster as visited.

4. Label any point that remains unvisited after processing all clusters as a noise point.

The general advantages and disadvantages of density-based clustering (using DBSCAN as reference) are the following [8]:
Advantages: does not require a predefined number of clusters; robust to outliers and noise; suitable for data with arbitrary shape.
Disadvantages: cannot cluster data sets with large differences in densities well; very sensitive to its main parameters (*eps* and *MinPts*), which makes choosing them quite challenging.

2.1.5 Connectivity-Based Clustering

Connectivity-based clustering algorithms construct a hierarchical relationship among data in order to form clusters. Hierarchical clustering (HC) techniques organize data based on a proximity matrix, visualizing results through binary trees or dendrograms. In this representation, the root node signifies the entire data set, with each leaf node corresponding to a data object. Intermediate nodes reflect proximity between objects, and dendrogram height indicates the distances between object pairs or clusters. Clustering results are derived by cutting the dendrogram at different levels, offering valuable insights and visualization of potential clustering structures. This approach is particularly beneficial when real hierarchical relationships exist, such as in evolutionary studies of species diversity [7]. Moreover, HC methods are of two types: agglomerative (which start with individual data points and iteratively merge them into clusters) and divisive (which start with the entire data set and recursively split it into smaller clusters) [7].

The main steps of the agglomerative hierarchical clustering (AHC) algorithm are as follows [7]:

1. Start with N singleton clusters and calculate the proximity matrix for the N clusters.

2. Search the minimal distance:
   $D(C_i, C_j) = min(D(C_m, C_l)); 1 \leq m, l \leq N, m \neq l$

where $D(*, *)$ is the distance function, in the proximity matrix, and combine clusters C_i and C_j to form a new cluster.

3. Update the proximity matrix by computing the distances between the new cluster and the other clusters.

4. Repeat steps 2 and 3 until all objects are in the same cluster.

The general advantages and disadvantages of connectivity-based clustering (using AHC as reference) are the following [8]:
Advantages: having the dendrogram helps interpret relationships between groups; does not require a predefined number of clusters; does not impose assumptions about the shape or distribution of clusters.
Disadvantages: relatively high time complexity in general (worst case is squared); sensitive to noise and outliers; choosing the dendrogram cutting point can be difficult.

2.1.6 Distribution-Based Clustering

Distribution-based clustering, also known as probabilistic clustering, is a technique in machine learning that assumes data points are generated from a mixture of probability distributions [7]. It groups data points based on their probability distribution, often using Gaussian distributions (but can use t-distributions as well). The parameters of these distributions are estimated to identify clusters. Unlike centroid-based clustering, which relies on representative points, distribution-based clustering focuses on the underlying probability distributions. One commonly used approach to probabilistic clustering is to create a Gaussian Mixture Model (GMM) through Expectation-Maximization [9], which is used to find the parameters of the Gaussian distributions that best fit the data, where the ‘best fit’ is defined as the one that maximizes the likelihood of the data given the parameters.

The main steps of the GMM clustering algorithm are as follows [7]:

1. Choose initial values for the parameters (such as the means, covariances, and mixing coefficients for each Gaussian distribution). This can be done either randomly or by using methods like K-means clustering to provide an initial estimate.

2. Given the current parameters, calculate the probability that each data point belongs to each cluster or Gaussian distribution. This step utilizes Bayes’ theorem [10] and results in a ’responsibility’ matrix where

each entry represents the probability of a data point belonging to a specific cluster.

3. Update the parameters (means, covariances, and mixing coefficients) using the responsibility matrix calculated in step 2.

4. Evaluate whether the parameters have significantly changed since the last iteration. If there are substantial changes, return to step 2. If not, the algorithm has converged, and the current parameters are accepted as the final model parameters.

5. Finally, assign each data point to the cluster it is most likely to belong to based on the computed probabilities.

The general advantages and disadvantages of distribution-based clustering (using GMM as reference) are the following [8]:
Advantages: robust to outliers and noise; can accurately model data with overlapping clusters; provides probabilities for cluster assignments, aiding in uncertainty estimation.
Disadvantages: relatively high time complexity in general (worst case is squared); many parameters with strong influence on the clustering; assumes the data to be composed of distributions.

2.2 Cluster Validation Measures

In the field of cluster analysis, validation measures are crucial for assessing the quality of clustering results. Two commonly used validation measures are the Silhouette Score and the Davies-Bouldin Index.

The Silhouette Score is a metric that quantifies how similar an object is to its own cluster in comparison to other clusters. The score is bounded between -1 and 1, where a high value signifies that the object is well integrated into its own cluster and poorly integrated into neighboring clusters [17]. If the majority of objects have a high score, it suggests that the clustering configuration is suitable. Conversely, if many points have a low or negative score, it may indicate that the clustering configuration has an excess or deficiency of clusters, leading to the misclassification of objects. The formula for calculating the Silhouette Score for a single data point i is as follows [18]:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{2}$$

where:

- $a(i)$ is the average dissimilarity of i with all other data within the same cluster,
- $b(i)$ is the lowest average dissimilarity of i to any other cluster, of which i is not a member.

The Davies-Bouldin Index (DBI) also serves as a valuable metric for assessing the effectiveness of clustering algorithms. It does this by evaluating the compactness and separation of the resulting clusters. More specifically, the DBI quantifies the average similarity ratio of each cluster to its most similar counterpart [19]. Here, 'similarity' is defined as the ratio of the scatter within a cluster to the separation between clusters. A lower DBI value, which can range from 0 to ∞, signifies superior clustering performance. This is because it implies that the clusters are both compact and well-separated. Moreover, the index is calculated by taking the average of the worst-case similarity ratios for each cluster, this approach emphasizing the least desirable characteristics of the clustering configuration. The formula for the Davies-Bouldin Index is as follows [19]:

$$DBI = \frac{1}{n} \sum_{i=1}^{n} \max_{i \neq j} \left(\frac{\sigma_i + \sigma_j}{d_{ij}} \right) \tag{3}$$

where:

- n is the number of clusters,
- σ_i is the average distance of all elements in cluster i to the centroid of cluster i,
- d_{ij} is the distance between cluster centroids i and j.

2.3 Sales Prediction

After obtaining the customer aggregations and checking their quality, our attention shifts to the crucial aspect of sales prediction. The current body of literature offers a plethora of methods to predict the sales of products and services. These methods are typically either applied the raw data set itself or combined with the clusters derived from the data, thus enhancing the predictive power.

2.3.1 Qualitative and Quantitative Predicting

The techniques that are generally utilized in practice can be broadly categorized into two classes: qualitative and quantitative predicting [15].

Qualitative predicting leverages information gleaned from market surveys to anticipate future trends. On the other hand, quantitative predicting places emphasis on the analysis of numerical data. It involves studying the number of elements in development based on historical statistical data, thereby making it immune to subjective factors [16].

In contrast, quantitative predicting emphasizes the analysis of numerical data. It involves studying historical statistical data to predict future sales, making it less subjective and more data-driven. Quantitative methods include a range of statistical and machine learning techniques that analyze the relationships between different variables and sales outcomes [15].

2.3.2 Predictive Modelling Methods

Within the realm of quantitative predicting, various algorithms can be employed to construct predictive models, such as Linear Regression, Decision Trees, Random Forest and Gradient Boosting (a brief summary of each method is provided hereafter).

Linear Regression is used to model the relationship between one or more independent variables and a dependent variable. Simple linear regression involves a single predictor variable, while multiple linear regression involves two or more predictors. Multiple linear regression, in particular, is widely used due to its ability to account for the influence of multiple variables on sales, which is more common in practice [14]. However, it assumes a linear relationship and may not perform well with non-linear data.

Decision Trees are a type of supervised learning algorithm used for both classification and regression tasks. They work by splitting the data into subsets based on the value of input features, creating a tree-like model of decisions. Moreover, Decision Trees are easy to interpret and can handle non-linear relationships, but they can be prone to overfitting [24].

Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the average prediction of the individual trees. It improves the predictive performance and robustness of the model by reducing overfitting and increasing generalization. Also, Random Forests are effective for both classification and regression tasks and can handle large datasets with higher dimensionality, although with an increased computational cost [25].

Gradient Boosting is another ensemble technique that builds models sequentially, with each new model correcting the errors of the previous ones. It combines the predictions of multiple weak learners (typically decision trees) to create a strong predictive model. Gradient Boosting is highly effective for both classification and regression tasks, offering high accuracy and flexibility, but it can be computationally intensive and sensitive to overfitting if not properly tuned [26].

3 Methodology And Experimental Setup

The primary goal of this research is to explore and understand the usefulness of data mining when applied to a real-world data set with the purpose of predicting future sales through customer profiling. In order to achieve the primary goal, the research aims to provide a comprehensive comparison of four diverse clustering algorithms - K-means, DBSCAN, AHC, and GMM - in terms of their speed, quality of clusters, and their ability to handle outliers over the span of three experiments. Each experiment will correspond to a pair of features (see section 3.2.2) selected from a given sales data set (see section 3.1) and its cluster results will be used in building a predictive model (described in section 2.3.2). Furthermore, the research will be guided by the following questions:

“Can data mining techniques applied to customer profiling improve the accuracy of sales predictions? If so, which clustering algorithm provides the most accurate and efficient results?”.

In the subsequent sections, I will delve into the specifics of the data set, the implementation details of the experiments, and the technology stack used. This should provide a broad understanding of the methodology and experimental setup, setting the stage for the results and discussion.

3.1 Data Set

The conducted research is performed on a data set provided by an industrial partner from Lithuania that focuses on filter manufacturing. The set contains information about the sales of the company, describing the placed orders through 13 different features (columns in the table) and over 100000 entries (rows in the table). For a clearer understanding of the structure of the available data, table 1 provides an explanation for each feature.

Feature	**Explanation**
Order	ID of an order
Customer	ID of a customer
Order Date	date on which the order was placed
Order Need By	date on which the order needs to arrive at the customer
Order Ship By	date on which the order is ready for shipping
Order Amount	value of the order in euros
Line	ID of an item within the order it is part of
Line Type	type of an item of an order (part, service or set)
PartID	ID of a part
Part	name of a part
Line Qty	quantity of an item of an order
Line Ship By	date on which an item of an order is ready for shipping
Line Need By	date on which an item of an order needs to arrive at the customer

Table 1: Features of the sales data set

To gain a deeper understanding of the data set, I conducted an Exploratory Data Analysis (EDA) on the data set. EDA is a fundamental step in the data analysis process. It allows us to understand the data we are working with, discover patterns, spot anomalies, test assumptions, and check for any initial hypotheses [31]. Therefore, I checked the distribution of the continuous features (through the Kolmogorov-Smirnov test [23]) and found that the "Order Amount," for example, does not follow a normal distribution, which can influence the GMM results. I also examined the relationships between different features using methods such as correlation matrices (Pearson's correlation to be exact), which helped me identify any strong correlations between variables that could impact our clustering and prediction algorithms. For example, I found that the “Order Amount” and “Line Qty” columns had a correlation of 0.836 (closer to 1 is better). This high correlation indicates that both features provide similar information, making them suitable for linear regression in the predictive model.

3.2 Implementation

By combining the techniques mentioned in section 2, the research is conducted through the steps depicted in the following flowchart:

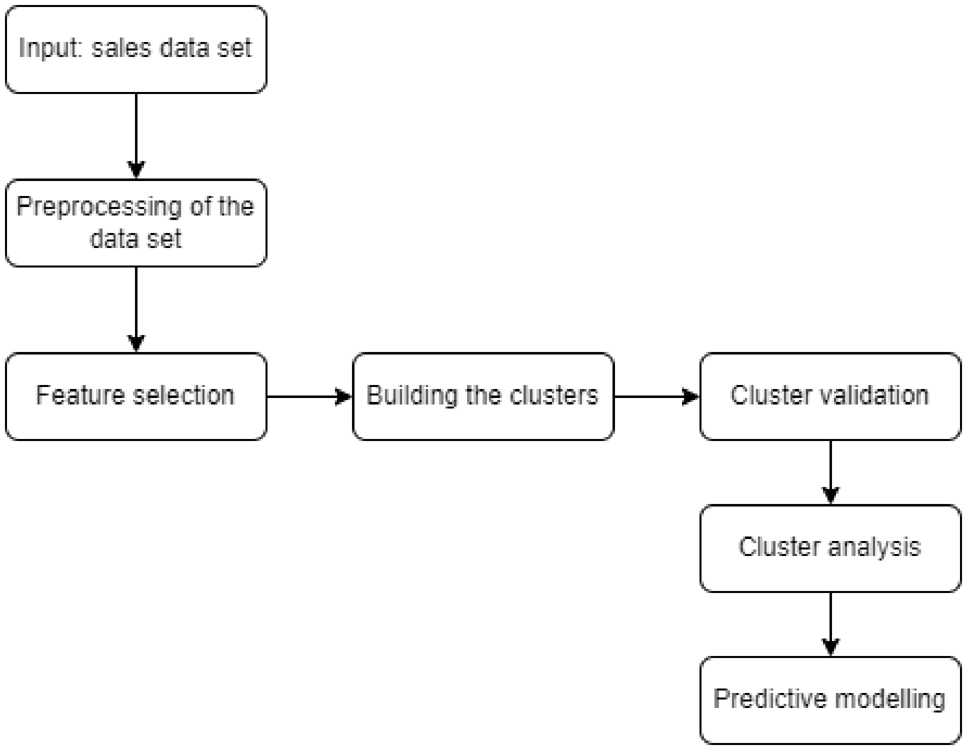

Figure 2: Flowchart of the implementation steps

3.2.1 Data Preprocessing

Before any analysis can be performed, the data set must be preprocessed to ensure its quality and relevance. This involves cleaning the data by removing any irrelevant entries, handling missing values, and converting data types where necessary. For instance, calendar dates may require conversion into a uniform format to facilitate subsequent processing, or numerical values may undergo normalization to fit within the range $[0, 1]$ or be subjected to standardization. Standardization is a preprocessing step in data analysis that helps to give different features the same weight during the clustering process. This is done to avoid any single feature from dominating the clustering due to its scale or units [21].

The initial data set is missing some values for the ‘order need by’ and ‘order ship by’ dates. While these features will not be used in our current analysis due to their clear low relevance in sales prediction (they are more related to logistics), it is important to consider potential scenarios where they could be significant. For instance, these dates might influence the order fulfillment efficiency, which could indirectly affect sales by altering the customer satisfaction. Ideally, if we were to use these features, one approach to handling the missing values would be to remove the corresponding orders to maintain the integrity of the data set. However, this is not the only solution. Alternatively, we could use imputation techniques to estimate the missing dates based on available data, or apply machine learning models that can handle missing values effectively [32]. Each method has its own advantages and trade-offs, and the choice would depend on the specific context and requirements of the analysis.

Moreover, there are only two outliers in the data set regarding the ‘order amount’, specifically the top two most expensive orders, as shown in figure 3. These outliers can influence the results of clustering algorithms that are sensitive to extreme values, such as K-means. Nevertheless, they also represent a natural part of the population we are analyzing, reflecting the reality that there can indeed be instances where someone spends significantly more than the average. Therefore, the top two orders by amount will not be removed in the performed research.

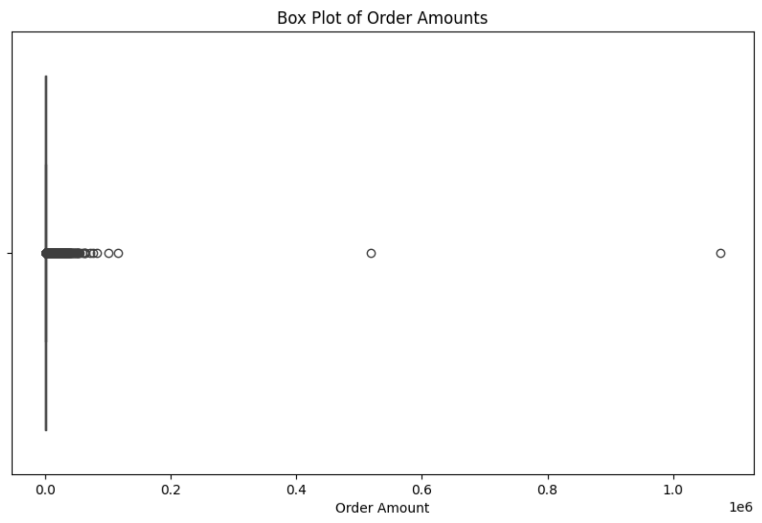

Figure 3: Box plot with the order amounts

3.2.2 Feature Selection

The next step is feature selection, which involves identifying the most pertinent features that will contribute to the effectiveness and relevance of the clustering. Features can be deemed relevant based on computational methods, such as their correlation with the target variable, or through domain knowledge and logical reasoning. For instance, the ID of an order might not be as significant as the quantity of a product ordered or the product's name. While visualizing high-dimensional data is complex, pairs of features are selected not only for visualization purposes but also to ensure meaningful and illustratable clusters.

The choice of feature selection over feature extraction is primarily due to the interpretability of the results. Feature selection maintains the original features, making the results more understandable and actionable. On the other hand, feature extraction, such as Principal Component Analysis (PCA), creates new composite features [13], which might be more difficult to interpret, especially in a business context like ours. Thus, we preserve a clear connection between the data and the real-world entities in the research.

In this study, three pairs of features were selected for three separate experiments to gain a comprehensive understanding of customer behaviors and trends. Since the data set contained no personal information about the customers, such as age, income, occupation, or company revenue (in the case of other businesses placing an order), the clusters will be more indirectly related to the customers. The chosen pairs are as follows:

- Frequency by quantity of a product over all orders: this pair can reveal the popularity of specific products across all orders, providing insights into what customers are most interested in.
- Total order amount by total product quantity per customer: this pair can help identify the spending habits of customers, showing whether they prefer to make frequent small purchases or fewer large ones.
- Order amount by the number of different products per order: this pair can indicate the diversity of products in a customer's order, which can be a sign of a customer's willingness to explore different items.

3.2.3 Building The Clusters

Once the relevant features have been selected, the clustering algorithm can be applied. There are various clustering algorithms available, such as K-means, Hierarchical Clustering, DBSCAN, and Gaussian Mixture Model, each with its own strengths and weaknesses which are explained in section 2. All four of these strategies are carried out on the sets of chosen features that were mentioned in the previous subsection, in order to assess and compare their performance and results. In addition, to make the results be as credible as possible, I test the algorithms beforehand on other data sets found on the internet that have the purpose of evaluating the correctness of an implementation.

3.2.4 Cluster Validation

After the clusters have been formed, their validity is assessed. This can involve either comparing the results with known customer profiles, if available, or applying statistical techniques to measure the quality of the clusters. In our situation, it's the latter option that is relevant, meaning that we have to use statistical measures such as the Silhouette Score and the Davies-Bouldin Index to assess the quality of our clusters, both of which have already been clarified in section 2.2. By using two metrics instead of a single one, we gain a more holistic view of the grouping results, allowing us to easily determine if the chosen number of clusters is ideal or further investigation is needed. Additionally, these quality scores help us select the best scatter plot for visualizing the clusters (the result of only one clustering algorithm is analysed per experiment), ensuring that the most representative and informative graph is used for analysis.

3.2.5 Cluster Analysis

Finally, the clusters are analyzed to derive insights about the customer profiles. This involves examining the characteristics of the points within each cluster of the graph chosen during the validation stage and identifying any common patterns or trends, which can be used to inform various business strategies. For instance, if we have a cluster where the majority of customers have a high frequency of orders with a large quantity of a specific part type, this could indicate a group of industrial clients who require these parts for their operations.

It is important to note that the results of cluster analysis are not absolute and should be interpreted with caution. Unlike classification, clustering is an unsupervised learning method that does not rely on predefined labels. Instead, it groups data based on similarity in characteristics, which can be subjective and dependent on the chosen parameters and distance measures. Therefore, while the clusters can provide valuable insights, they should not be viewed as definitive categories, since they serve only as a starting point for understanding the structure within the data.

3.2.6 Predictive Modelling

The final step in our implementation entails predictive modeling, where we utilize the insights gained from cluster analysis to predict future sales trends and behaviors. By insights from the cluster analysis, I refer to the cluster values assigned to each data point. For instance, in the first experiment, for a product, we can use its frequency and assigned cluster to predict its quantity. The assigned cluster provides additional context that can improve the accuracy of the predicted value. However, it is important to note that building a detailed predictive model is not strictly necessary. We can stop at the cluster analysis and still make some abstract predictions based on just the identified clusters, using our intuition. Given the varying correlation coefficients observed in the experiments (as discussed in section 3.1 during the Exploratory Data Analysis), if we choose to proceed with predictive modelling, we need to select appropriate models for predictive tasks:

- First Experiment (Pearson correlation coefficient: 0.338): Linear regression might not be the best choice due to the weak linear relationship. Instead, Random Forest Regression is used, a flexible model that can capture non-linear relationships and also handle complex data distributions.
- Second Experiment (Pearson correlation coefficient: 0.836): With a high correlation coefficient, linear regression is a suitable model due to the strong linear relationship between the variables.
- Third Experiment (Pearson correlation coefficient: 0.123): Given the very low correlation, models that can handle weak or complex relationships are needed. We choose Gradient Boosting, which builds an ensemble of weak prediction models, usually decision trees, to create a strong predictive model to capture more intricate patterns in the data.

The data (features used in the experiments) is then split into training and test sets, using the most common rule, i.e. 80%-20% split, to estimate model performance. Each model is trained on its respective training set and evaluated using the following metrics to assess accuracy and reliability:

- Mean Absolute Error (MAE): Measures the average magnitude of the errors in a set of predictions, without considering their direction. It provides a straightforward interpretation of the average error [27].

  $$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{4}$$

  where:

  - n is the number of observations
  - y_i is the actual value
  - $\hat{y}_i$ is the predicted value

- Mean Squared Error (MSE): Measures the average of the squares of the errors. It gives more weight to larger errors, making it useful for identifying significant deviations [27].

  $$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{5}$$

  where:

  - n is the number of observations
  - y_i is the actual value
  - $\hat{y}_i$ is the predicted value

- R-squared ($\mathbf{R^2}$): Represents the proportion of the variance in the dependent variable that is predictable from the independent variables. It indicates how well the model explains the variability of the outcome [27].

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{6}$$

  where:

  - y_i is the actual value
  - $\hat{y}_i$ is the predicted value
  - $\bar{y}$ is the mean of the actual values

Once the models are trained and validated, they can be deployed to make predictions on new data. These predictions can be used to inform business decisions, such as inventory management and marketing strategies. Moreover, by continuously monitoring the model performance and updating them with new data, we can maintain their accuracy and relevance over time.

3.3 Technology Stack

The research paper is written using *LaTeX* with the help of the online editor *Overleaf*. To ensure no progress is lost in case of unforeseen hardware/software malfunctions, all written source files of the required program are backed up on a remote *GitHub* repository, utilising the platform's CI/CD (continuous integration/delivery) capabilities. In terms of IDE, I chose to use *IntelliJ IDEA* with the *python* support plugin, since I am familiar with it and it helps streamline my coding process and catch errors early due to its intelligent features like code completion, refactoring, and quick navigation. Moreover, the data sets used in testing the algorithms are open source (can be found here [30]) and freely available for anyone to use and apart from a personal computer (laptop), no other hardware is required.

The experiments were set up using several *python* libraries, including *scikit-learn* (sklearn) for machine learning algorithms (used in the clustering and the predictive modelling), *pandas* for data manipulation and analysis, *seaborn* and *matplotlib* for data visualization, and *numpy* for any numerical computations.

4 Experimental Results

As mentioned in section 3.2.3, each experiment is run on four clustering algorithms. The provided graphs depict the obtained cluster configurations and the validation values for varying numbers of clusters. This was carried out to ascertain the optimal number of clusters for the K-means, GMM, and AHC algorithms. However, for the DBSCAN algorithm, the second graph is utilized to determine the optimal epsilon parameter instead (through the elbow method and k-nearest neighbor technique [22]). These visual representations serve as a crucial tool in understanding and fine-tuning the performance of each clustering method. Finally, a table containing the overall execution times and cluster quality ratings is provided per experiment, as well as a summary of the predictive model performance metrics.

4.1 Experiment 1

In this experiment, we are performing a clustering operation based on two key features: the frequency and quantity of every product across all orders. For example, one such product, T02057 to be more specific, has appeared in 504 different orders, with a total quantity of 40800 units ordered. In the context of our cluster graphs, these two figures serve as the x and y coordinates respectively.

The two features - frequency and quantity - are crucial in understanding the distribution and demand of the product. However, to ensure a fair and unbiased clustering operation, it is important that these features are standardized, since I do not consider one of them to be more significant than the other. Consequently, after the aggregation processes is completed, every data point, representing an individual product, is precisely allocated to its respective cluster, ensuring an objective representation of its characteristics.

4.1.1 K-Means

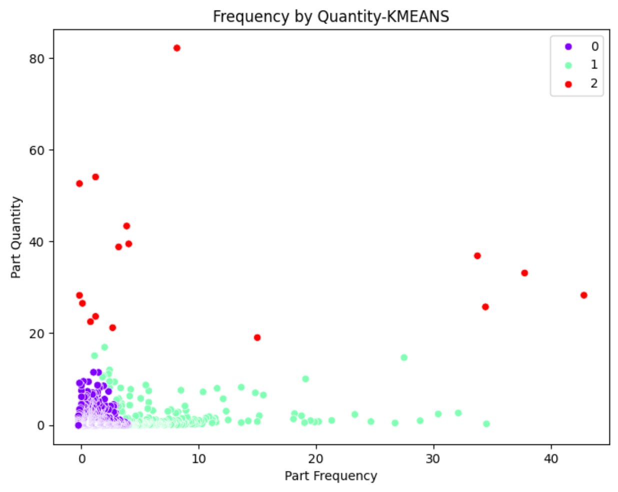

Figure 4: Exp 1: Clusters obtained through K-means

The optimal number of clusters was found by analysing the Silhouette Score and Davies-Bouldin Index figures below, giving the value 3 as the answer (Silhouette should be closer 1, while DBI closer to 0).

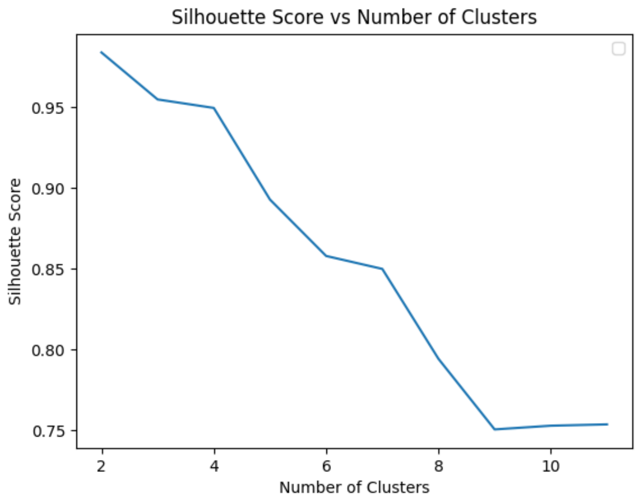

Figure 5: Exp 1: Silhouette score by number of clusters for K-means

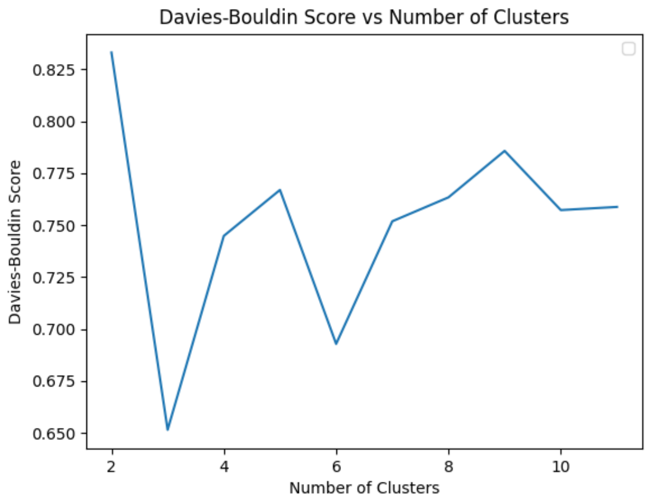

Figure 6: Exp 1: Davies-Bouldin Index by number of clusters for K-means

4.1.2 Gaussian Mixture Model

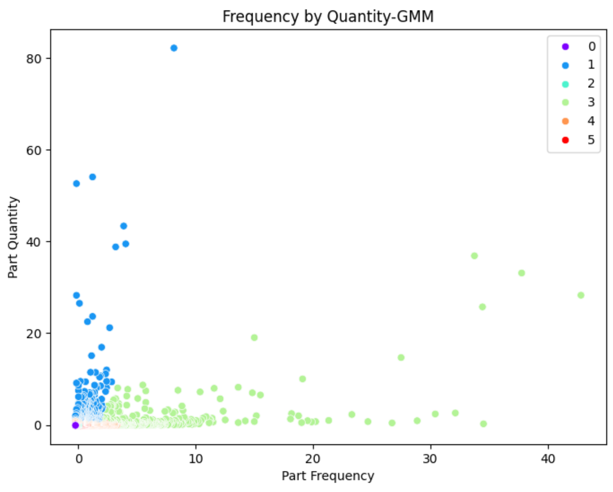

Figure 7: Exp 1: Clusters obtained through GMM

The optimal number of clusters was found by analysing the Silhouette Score and Davies-Bouldin Index figures below, giving the value 6 as the answer (Silhouette should be closer 1, while DBI closer to 0). Note that in the cluster visualization, cluster 5, represented in red, is located near the origin of the graph.

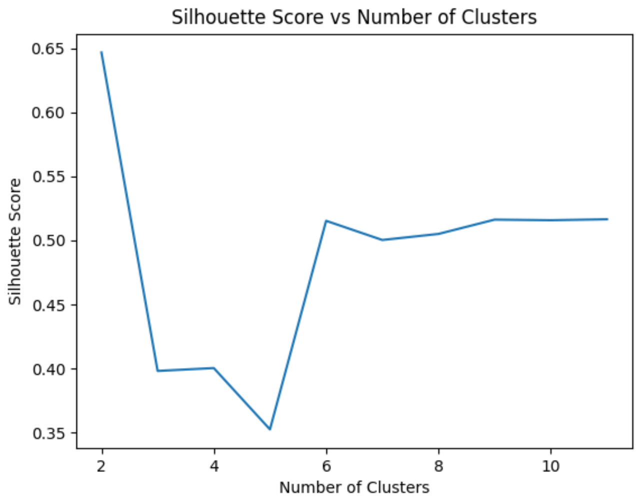

Figure 8: Exp 1: Silhouette score by number of clusters for GMM

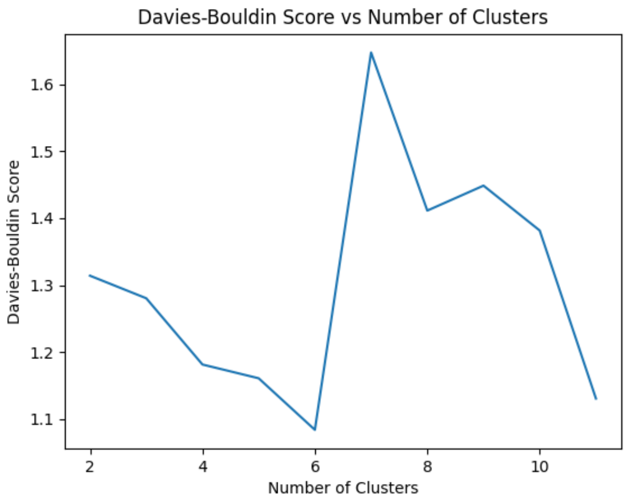

Figure 9: Exp 1: Davies-Bouldin Index by number of clusters for GMM

4.1.3 Agglomerative Hierarchical Clustering

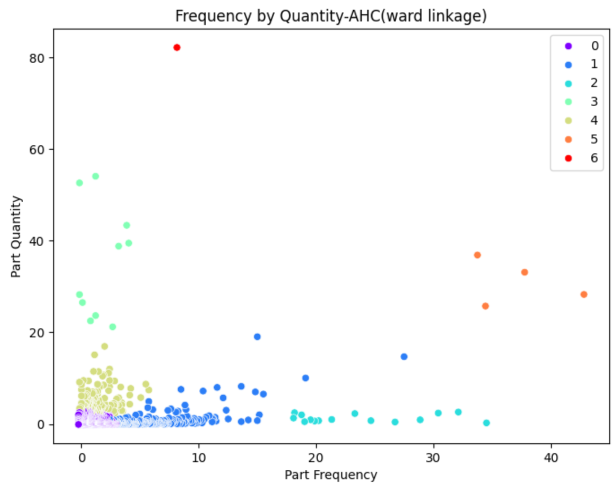

Figure 10: Exp 1: Clusters obtained through AHC

The optimal number of clusters was found by analysing the Silhouette Score and Davies-Bouldin Index figures below, giving the value 7 as the answer (Silhouette should be closer 1, while DBI closer to 0).

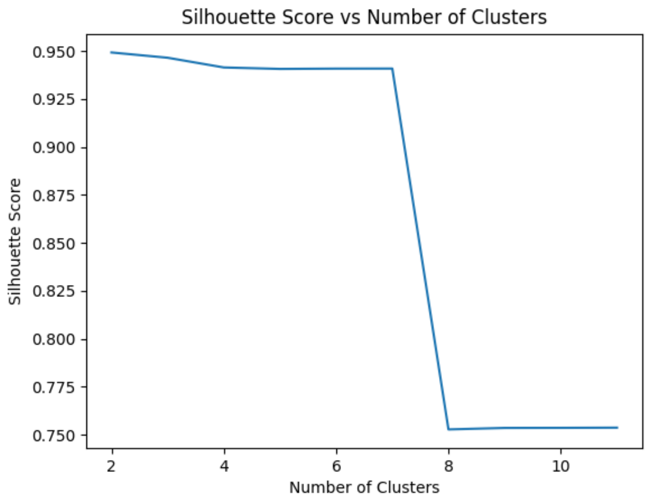

Figure 11: Exp 1: Silhouette score by number of clusters for AHC

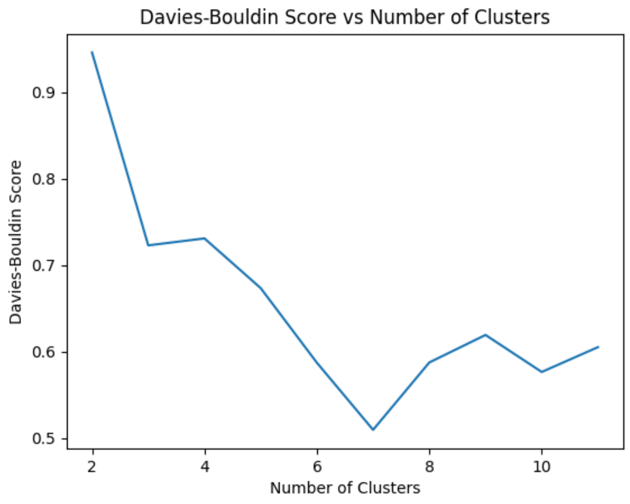

Figure 12: Exp 1: Davies-Bouldin Index by number of clusters for AHC

4.1.4 Density-Based Spatial Clustering of Applications With Noise

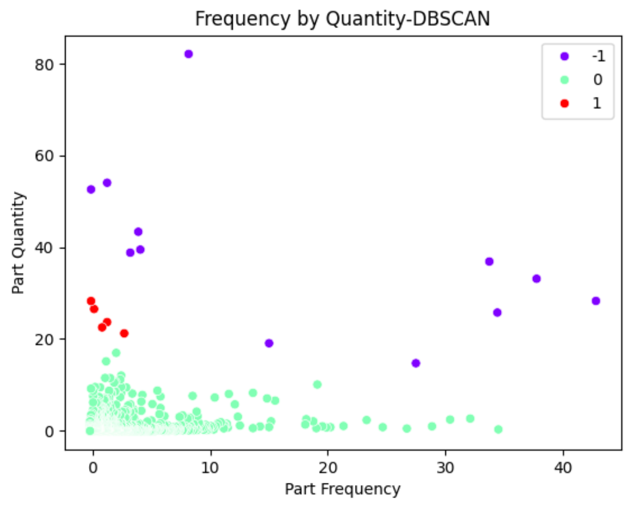

Figure 13: Exp 1: Clusters obtained through DBSCAN (-1 represents noise)

The optimal number of clusters was found automatically due to the nature of DBSCAN. The epsilon parameter required by the algorithm is 5.5, which given by the figure below.

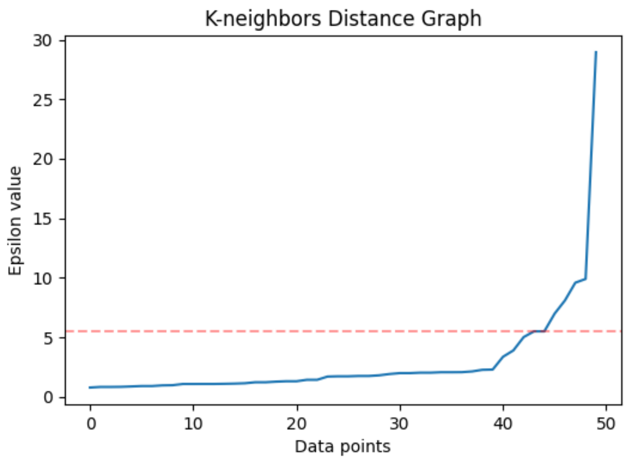

Figure 14: Exp 1: K-distance graph for determining optimal epsilon value

4.1.5 Overall Execution Times And Quality Scores

The table below summarizes the execution times and quality scores for different clustering algorithms: K-means, GMM, AHC and DBSCAN. These metrics help in comparing the algorithms and choosing one for the analysis step of the implementation. The scores are computed for the optimal number of clusters, which was determined for each algorithm in the subsections of 4.1.

	K-means	GMM	AHC	DBSCAN
Execution Time	0.31s	0.76s	9.50s	7.10s
Silhouette Score	0.95	0.53	0.94	0.98
Davies-Bouldin Index	0.65	1.08	0.52	0.12

Table 2: Exp 1: Execution times and quality scores for different algorithms

4.1.6 Predictive Model Performance Metrics

This table displays performance metrics for a predictive model that estimates the quantity of a product through Random Forest Regression. The model uses as predictors the frequency of a product and the cluster (from the DBSCAN result, since it had the best quality scores) it is part of, although other combinations of predictors would also be possible (for example, predicting frequency based on quantity and cluster).

Metric	Value
Mean Absolute Error	148.47
Mean Squared Error	995880.81
R-squared	0.64

Table 3: Exp 1: Accuracy scores of the predictive model

4.2 Experiment 2

In this experiment, we are conducting a clustering analysis on a data set representing the customer orders. Each data point in the cluster corresponds to a unique customer, while the two features we are considering for each customer are the total amount spent on orders and the total product quantity ordered. In addition, to guarantee that both features contribute equally to the clustering process, they are standardized before the analysis, ensuring that no single feature dominates the clustering due to its scale. Through this method, we aim to identify patterns and groupings among customers based on their purchasing behavior.

4.2.1 K-Means

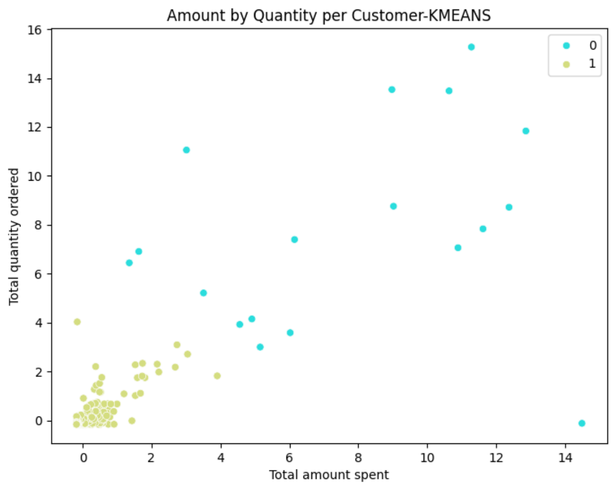

Figure 15: Exp 2: Clusters obtained through K-means

The optimal number of clusters was found by analysing the Silhouette Score and Davies-Bouldin Index figures below, giving the value 2 as the answer.

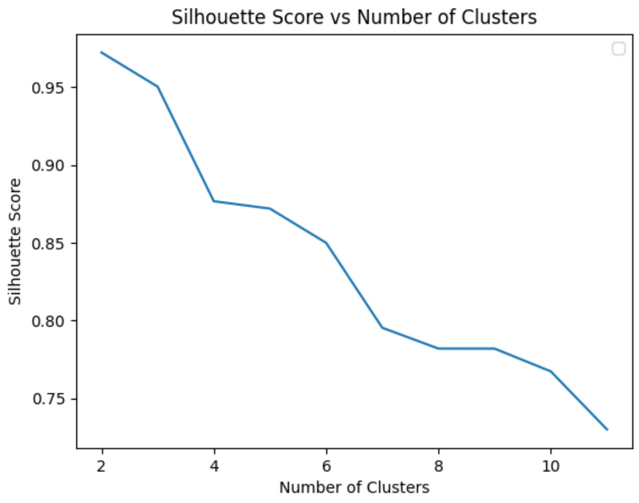

Figure 16: Exp 2: Silhouette score by number of clusters for K-means

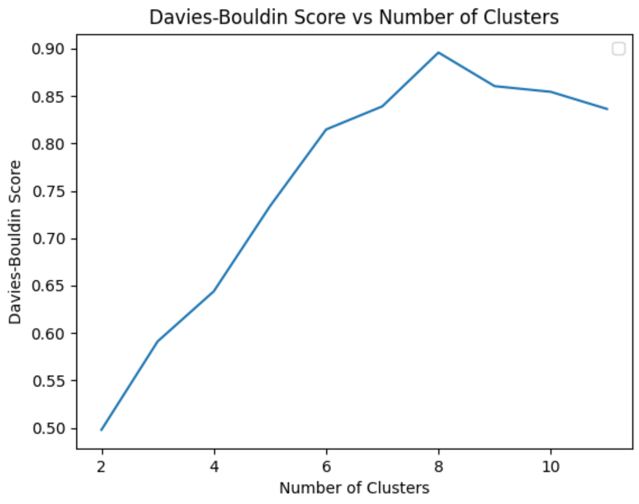

Figure 17: Exp 2: Davies-Bouldin Index by number of clusters for K-means

4.2.2 Gaussian Mixture Model

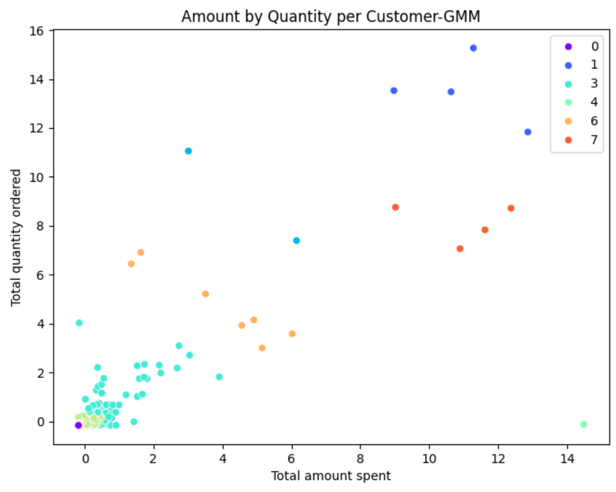

Figure 18: Exp 2: Clusters obtained through GMM

The optimal number of clusters was found by analysing the Silhouette Score and Davies-Bouldin Index figures below, giving the value 9 as the answer.

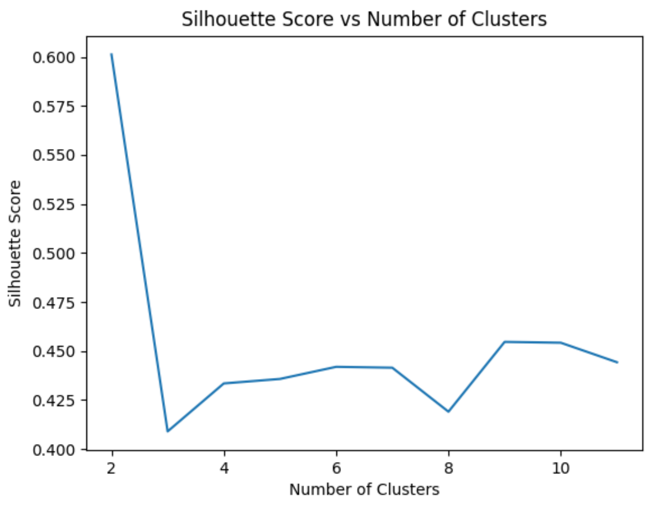

Figure 19: Exp 2: Silhouette score by number of clusters for GMM

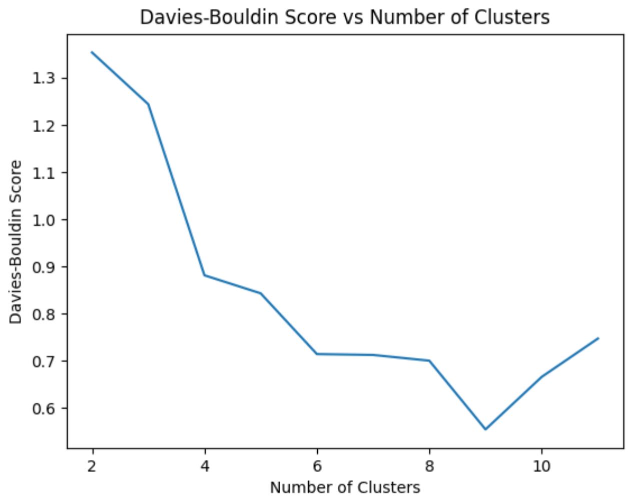

Figure 20: Exp 2: Davies-Bouldin Index by number of clusters for GMM

4.2.3 Agglomerative Hierarchical Clustering

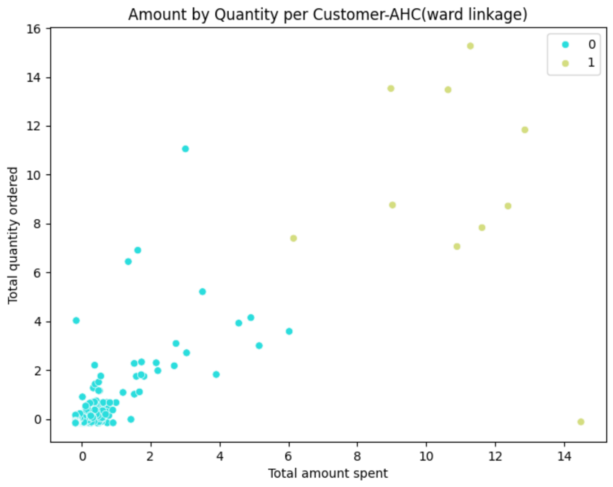

Figure 21: Exp 2: Clusters obtained through AHC

The optimal number of clusters was found by analysing the Silhouette Score and Davies-Bouldin Index figures below, giving the value 2 as the answer.

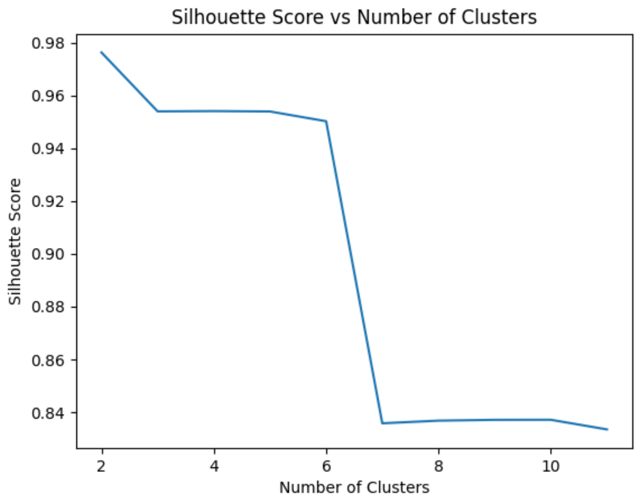

Figure 22: Exp 2: Silhouette score by number of clusters for AHC

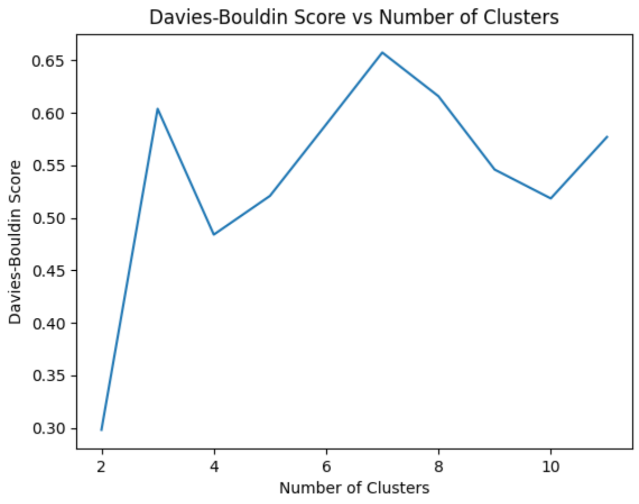

Figure 23: Exp 2: Davies-Bouldin Index by number of clusters for AHC

4.2.4 Density-Based Spatial Clustering of Applications With Noise

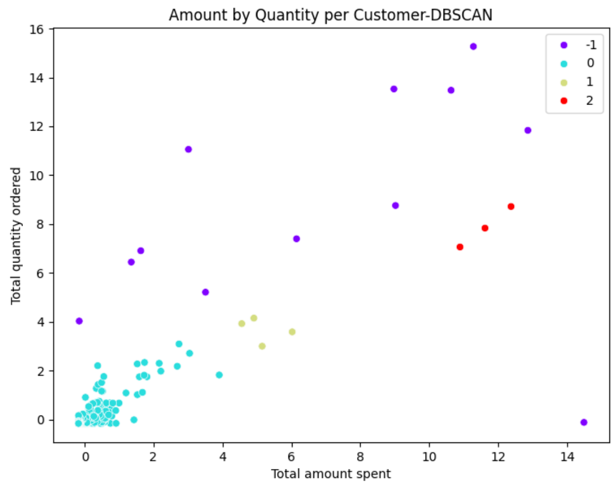

Figure 24: Exp 2: Clusters obtained through DBSCAN (-1 represents noise)

The optimal number of clusters was found automatically due to the nature of DBSCAN. The epsilon parameter required by the algorithm is 1.65, which given by the figure below.

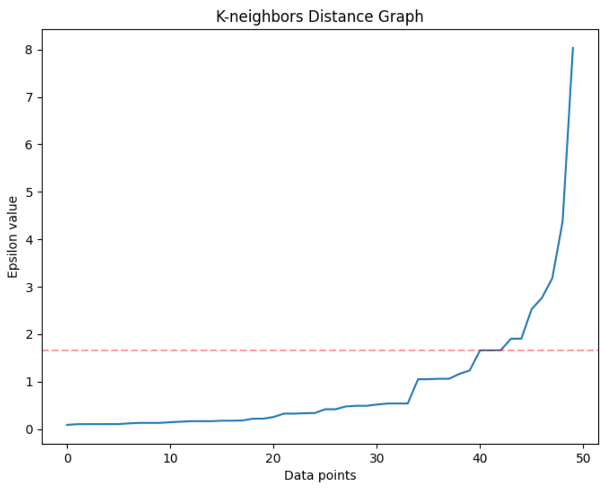

Figure 25: Exp 2: K-distance graph for determining optimal epsilon value

4.2.5 Overall Execution Times And Quality Scores

The table below summarizes the execution times and quality scores of the used clustering algorithms. These metrics help in comparing the algorithms and choosing one for the cluster analysis. The scores are computed for the optimal number of clusters, which was determined for each algorithm in the subsections of 4.2.

	K-means	GMM	AHC	DBSCAN
Execution Time	0.024s	0.20s	0.017s	0.019s
Silhouette Score	0.97	0.45	0.98	0.96
Davies-Bouldin Index	0.50	0.57	0.30	0.17

Table 4: Exp 2: Execution times and quality scores for different algorithms

4.2.6 Predictive Model Performance Metrics

This table displays performance metrics for a predictive model that estimates the total quantity of a product ordered by a customer through Linear Regression. The model uses as predictors the amount spent by a customer and the cluster (from the DBSCAN result, due to its high quality scores) the customer is part of, although other combinations of predictors would also be possible (for example, predicting amount spent based on quantity ordered and cluster). The high MSE value is due to the large scale of the order amount values, which results in larger squared errors. This is not necessarily indicative of poor model performance, but rather a reflection of the high values in the dataset.

Metric	Value
Mean Absolute Error	1093.56
Mean Squared Error	17111841.36
R-squared	0.95

Table 5: Exp 2: Accuracy scores of the predictive model

4.3 Experiment 3

In this analysis, we are focusing on a data set of individual orders, with each order represented as a unique data point in our cluster. We are examining two specific features for each order: the total amount of the order and the number of unique product types within that order. To ensure an equal contribution from both features during the clustering process, we have standardized them prior to the analysis, preventing any single feature from overpowering the clustering due to its scale. Through this method, we strive to unveil intricate patterns and establish meaningful classifications among orders, guided by the nuances of their monetary value and the diversity of products they encompass. Moreover, the top two orders by amount, which are outliers, are included during the grouping process but excluded from the cluster graphs themselves, as to improve the visibility of the final clusters (the only exception being the AHC result).

4.3.1 K-Means

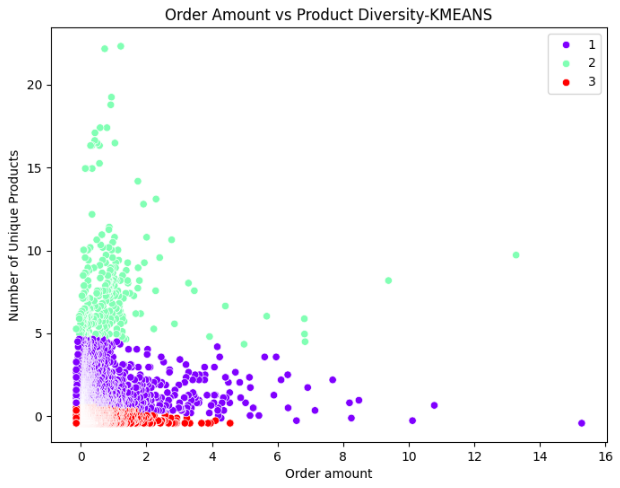

Figure 26: Exp 3: Clusters obtained through K-means

The optimal number of clusters was found by analysing the Silhouette Score and Davies-Bouldin Index figures below, giving the value 4 as the answer. The two outliers by order amount form their own cluster, with identifier 0. They have been removed from the figure to improve the scaling and make the other points more visible.

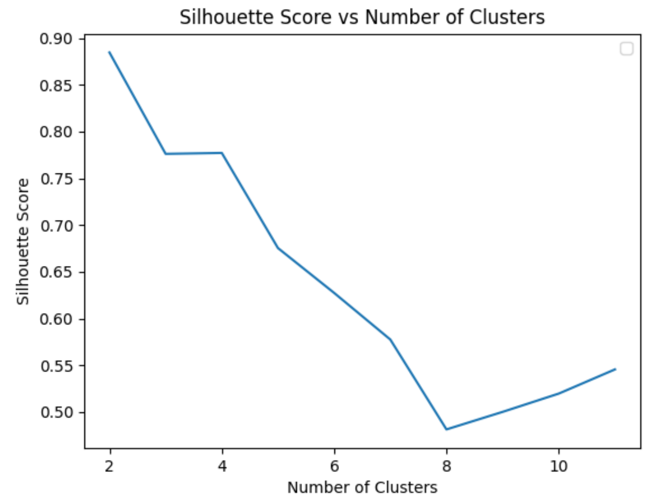

Figure 27: Exp 3: Silhouette score by number of clusters for K-means

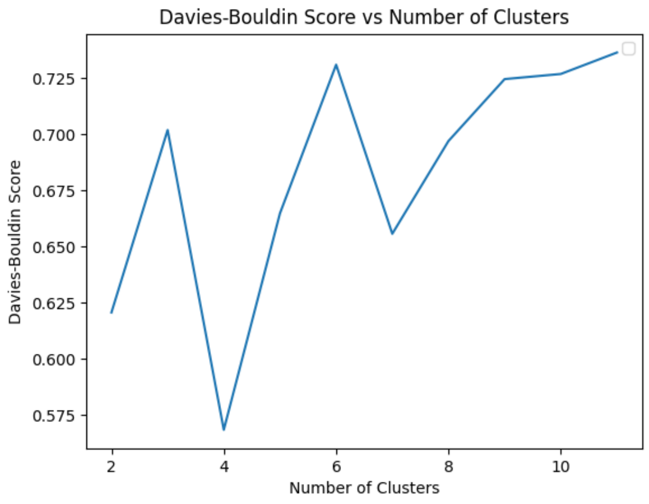

Figure 28: Exp 3: Davies-Bouldin Index by number of clusters for K-means

4.3.2 Gaussian Mixture Model

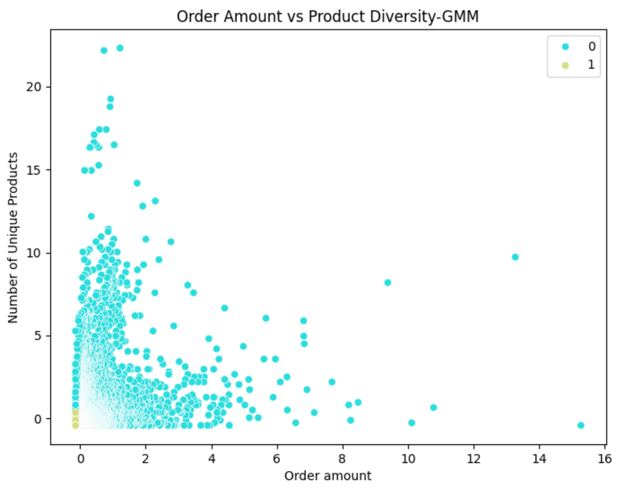

Figure 29: Exp 3: Clusters obtained through GMM

The optimal number of clusters was found by analysing the Silhouette Score and Davies-Bouldin Index figures below, giving the value 3 as the answer. The first outlier with the highest order amount forms its own cluster with identifier 2, while the second one was added to cluster 0. They have been removed from the figure to improve the scaling and make the other points more visible.

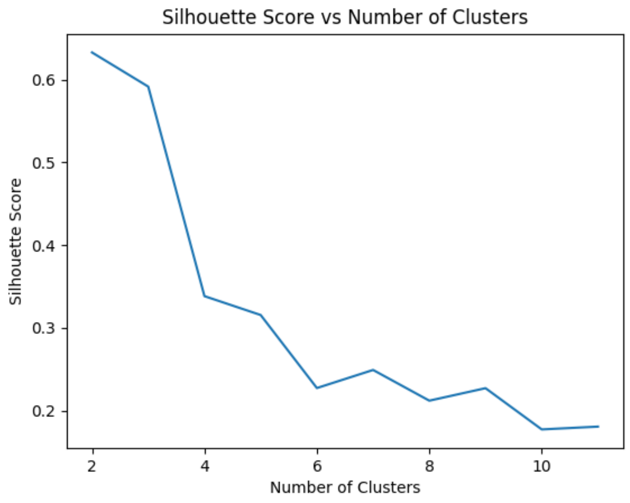

Figure 30: Exp 3: Silhouette score by number of clusters for GMM

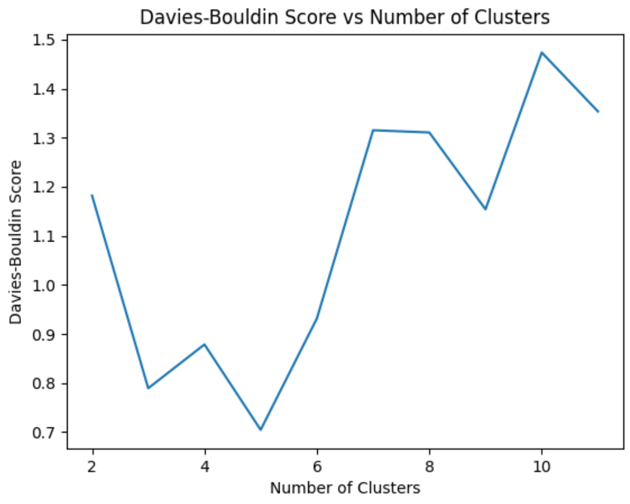

Figure 31: Exp 3: Davies-Bouldin Index by number of clusters for GMM

4.3.3 Agglomerative Hierarchical Clustering

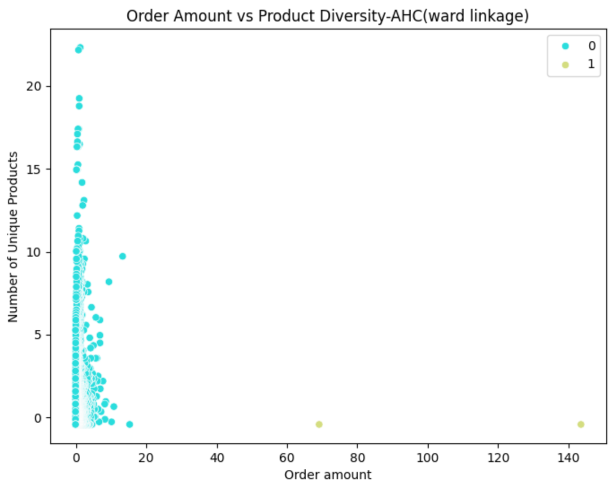

Figure 32: Exp 3: Clusters obtained through AHC

The optimal number of clusters was found by analysing the Silhouette Score and Davies-Bouldin Index figures below, giving the value 2 as the answer. The two outliers by order amount have not been from the figure in this case, since all of the other points form a single cluster, making the observability of the results not an issue.

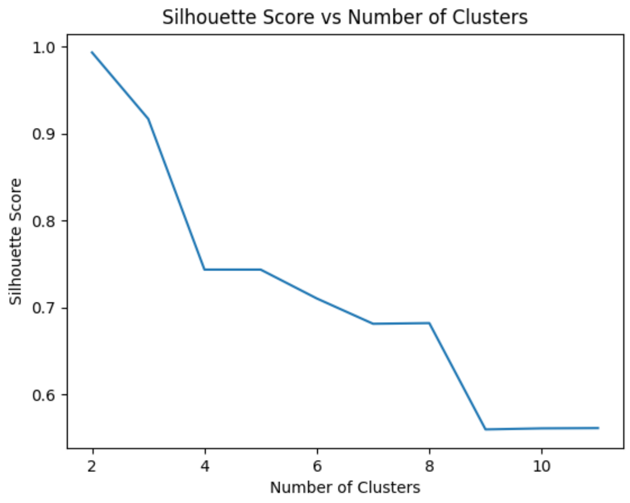

Figure 33: Exp 3: Silhouette score by number of clusters for AHC

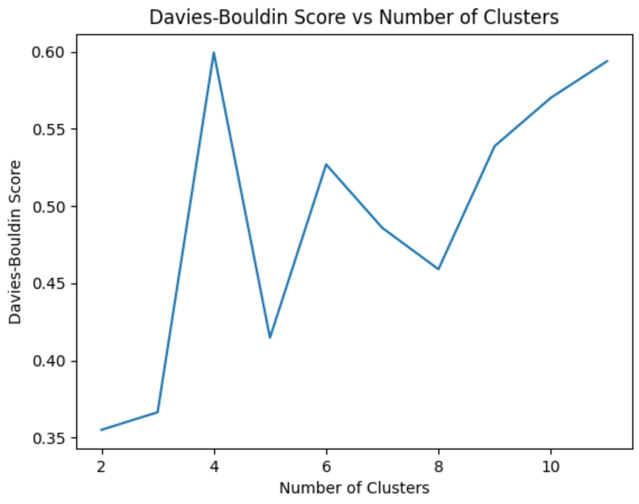

Figure 34: Exp 3: Davies-Bouldin Index by number of clusters for AHC

4.3.4 Density-Based Spatial Clustering of Applications With Noise

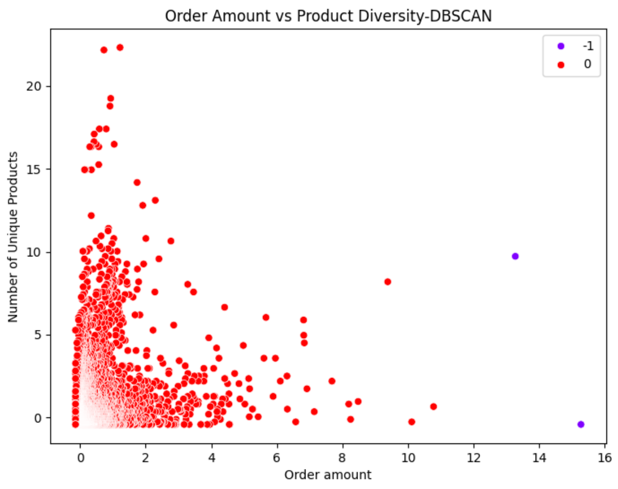

Figure 35: Exp 3: Clusters obtained through DBSCAN (-1 represents noise)

The optimal number of clusters was found automatically due to the nature of DBSCAN. The epsilon parameter required by the algorithm is 3.6, which given by the figure below. The two outliers by order amount are considered noise in this scenario, being part of the cluster with identifier -1. They have been removed from the figure to improve the scaling and make the other points more visible.

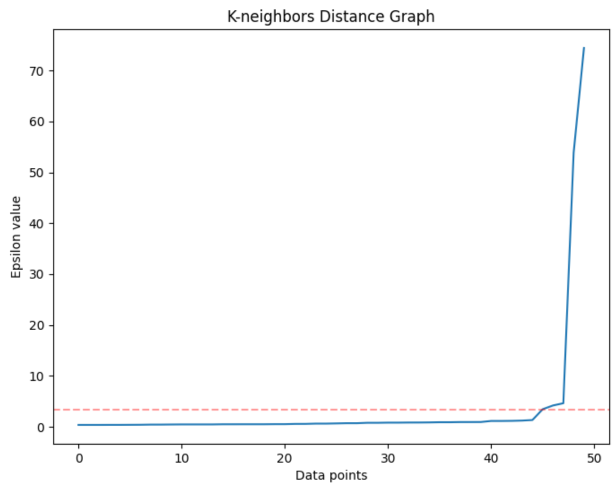

Figure 36: Exp 3: K-distance graph for determining optimal epsilon value

4.3.5 Overall Execution Times And Quality Scores

The table below outlines the execution times and quality scores for the clustering algorithms used in this experiment. These metrics are essential for comparing the performance of each algorithm and selecting the most suitable one for the cluster analysis. The scores are computed for the optimal number of clusters, as identified in the previous subsections of 4.3. Note that the "N/A" values for DBSCAN are due to the algorithm finding only a single cluster in this case.

	K-means	GMM	AHC	DBSCAN
Execution Time	0.33s	0.27s	10.00s	6.10s
Silhouette Score	0.77	0.59	0.99	N/A
Davies-Bouldin Index	0.57	0.80	0.36	N/A

Table 6: Exp 3: Execution times and quality scores for different algorithms

4.3.6 Predictive Model Performance Metrics

This table displays performance metrics for a predictive model that estimates the unique product count per order through Gradient Boosting Regression. The model uses as predictors the order amount and the cluster (from the AHC result, because it demonstrated the highest quality scores) the order is part of, although other combinations of predictors would also be possible (for example, order amount based on the number of unique products in the order and cluster).

Metric	Value
Mean Absolute Error	2.63
Mean Squared Error	28.40
R-squared	0.18

Table 7: Exp 3: Accuracy scores of the predictive model

5 Discussion

5.1 Algorithm Comparisons

The three of experiments we conducted, each honing in on distinct aspects of our data set, used four clustering algorithms: K-means, DBSCAN, AHC, and GMM. These experiments are important illustrations of the respective strengths and weaknesses inherent in each clustering algorithm. Conversely, the prediction algorithms will not be compared, because their performance is extremely dependent on the context of each experiment and the nature of the data involved.

K-means, known for its simplicity and speed, provided clear visualizations of the clusters in, overall, the lowest amount of time. However, since it assumes clusters to be convex (roughly spherical in shape) and isotropic (the spread of data points is the same in all directions within a cluster) [8], which may not always be the case for real-world data, it can result in unusual linear cluster shapes such as in figure 26. Moreover, due to some degree of randomness involved in the algorithm's initialization phase, multiple runs were required to get the average Silhouette Score and DBI, meaning those values are not the absolute truth. Also, despite being the fastest, K-means did not necessarily produce the highest quality clusters, reflected in the unimpressive Silhouette Score and DBI values when compared to those achieved by DBSCAN.

GMM demonstrated impressive speed, ranking as the second fastest algorithm. It trailed slightly behind K-means in the first and third experiments (it is important to note that the second experiment is not a reliable benchmark for the comparison of run times, given that its input data set was significantly smaller: almost thirty times less than the other two experiments). Nevertheless, GMM had the worst cluster quality scores, resulting in many overlapping clusters as in figures 7 and 29. This occurred because GMM assumes that the data points are generated from a mixture of several Gaussian distributions, which can lead to clusters overlapping when the actual data distribution does not fit this assumption well. Consequently, GMM struggled to delineate clear and distinct clusters, negatively impacting its performance on the provided real-world data. In addition, similar to K-means, GMM also involves some degree of randomness in the initialization phase, requiring multiple runs to obtain reliable results regarding the Silhouette Score and DBI.

AHC, while being the slowest among the four algorithms tested, excelled in producing high-quality clusters, securing the second highest quality scores after DBSCAN. The algorithm's hierarchical approach to clustering, where each data point starts in its own cluster and pairs of clusters are merged as one moves up the hierarchy, ensures a comprehensive exploration of the data structure. This methodical process, although computationally intensive, contributed to its superior performance in cluster quality as illustrated in figures 10 and 21. Unlike K-means and GMM, AHC does not involve any randomness in its initialization or execution phases. Consequently, it managed to generate consistent and reproducible results across multiple runs, with Silhouette Score and DBI values that are definitive and absolute.

DBSCAN emerged as the top performer in terms of cluster quality, achieving the highest Silhouette Score and DBI values among the four algorithms tested (to be noted that for the third experiment, there is only one cluster in figure 35 which would mean maximal scores, hence the "N/A" values in table 6). Although not the fastest, DBSCAN ranked third in terms of speed, ahead of AHC but trailing behind K-means and GMM. Moreover, just like AHC, DBSCAN has the advantage of a deterministic nature, i.e. it does not involve any randomness in its initialization or execution phases, meaning that the obtained clusters are the same no matter how many times the algorithm is run. Even though a distinguishing feature of DBSCAN is its ability to identify and treat outliers effectively, the algorithm can still classify normal data points as noise if they do not meet the minimum criteria for being part of a cluster based on a specified distance (epsilon) and minimum number of points. These noise points, illustrated in figures 13 and 24, can still be considered valid data points and might be regarded as forming their own "cluster" of outliers which can be further analysed. For example, in the context of our first experiment, the outlier cluster (figure 13) would actually be the most important one, since it represents the most popular products among customers.

5.2 Practical Analysis And Sales Prediction

By exploring the performance tables of each of the three experiments, we can find out the results of which clustering algorithm (i.e. its scatter plot) to choose for our sales analysis. Additionally, for the predictions, we can investigate the machine learning models that were built in order to estimate the relevant features per experiment.

5.2.1 First Experiment Predictions

In the case of the first experiment, DBSCAN displayed the best cluster quality scores (figure 13). Since the groupings represent what products the customers are most interested in, we are basically categorizing the products by overall popularity. Thus, the items in the purple cluster are most likely to sell or, at least, will have bigger orders, suggesting they are in high demand and have a strong market presence (the company should focus production on them). Conversely, the red cluster, while not as popular as the purple one, shows good potential due to its reasonable quantity, being able to maybe contribute to the total revenue in the future.

For more concrete predictions, we can use the model we built that approximates the product quantity using its frequency and cluster. The R^2 score of 0.64 indicates that approximately 64% of the variance in product quantity can be explained by this model. While this shows the model has some predictive power, there is still substantial room for improvement. The following table contains some prediction examples for a few products, showing both the estimated and actual quantities from the data set, while also illustrating the influence the cluster value has (the inputs to the model are only 'Frequency' and 'Cluster'):

Product ID	Frequency	Cluster	Estimated Quantity	Actual Quantity
T19969	341	0	1501	1442
T02057	504	-1	42280	40800
T00155	316	0	939	753

Table 8: Experiment 1 predictions

5.2.2 Second Experiment Predictions

For the second experiment, similar to the first, DBSCAN once again displayed the best cluster quality scores (figure 24). The clusters represent the customer purchasing behavior, providing valuable insights into which products are frequently bought together and by which type of customers. Focusing on the purple cluster customers' choices can guide the company towards potential best-sellers. These items are what is currently being purchased for quite substantial amounts of money, indicating a strong market preference. To capitalize on this trend, it would be beneficial for the company to concentrate production efforts on these particular products. Additionally, for products favored by light blue cluster customers, implementing discounts on large quantities or introducing loyalty programs could incentivize purchases. Such strategic moves could not only boost sales but also foster customer loyalty and satisfaction by rewarding their continued patronage.

For more precise predictions, we can use the Linear Regression model we built to estimate product quantities based on the amount spent by a customer and their cluster. The R^2 score of 0.95 means that 95% of the variance in product quantity per customer can be explained by this model, demonstrating its robustness and ability to give highly accurate predictions. The following table provides prediction examples for a few customers, showing both the estimated and actual quantities from the data set. This model can, of course, be used for any other customer as well (the inputs to the model are only 'Amount Spent' and 'Cluster'):

Customer ID	Amount Spent	Cluster	Estimated Quantity	Actual Quantity
1775	837	0	399	442
2490	5158	0	916	812
3075	751986	1	60017	66449

Table 9: Experiment 2 predictions

5.2.3 Third Experiment Predictions

Finally, int the third experiment, AHC emerged as the top performer (figure 32). The clusters were formed based on the order amount and the number of unique products per order. Although AHC had the best results, its scatter plot revealed one large cluster and a smaller one with only two orders in it, which essentially represent the outliers by order amount. The light blue cluster represents the majority of customers who make regular-priced orders, while the yellow cluster could defines a niche segment of customers who make exceptionally expensive orders. Despite the smaller size of this cluster, the high order amount indicates a significant contribution to the company's revenue, so it might be beneficial for the company to investigate this customer segment and tailor specific marketing strategies to cater to their unique purchasing behavior. Furthermore, the K-means graph (figure 26) could be consulted to better understand the distribution of the majority of customers.

Similar to the other experiments, reviewing the Gradient Boosting model we built to estimate unique products based on the order amount and its cluster will give more tangible predictions. The R^2 score of 0.18 means that only 18% of the variance in unique product count can be explained by this model, indicating a weak predictive power. This low R^2 score suggests that there are other significant factors affecting the said feature that are not captured by the model. The following table provides prediction examples for a few orders (the inputs to the model are only 'Order Amount' and 'Cluster'). The 'Order IDs' are not from the data set, since they will not repeat in the future, meaning we do not have actual values to compare the estimated ones with.

Order ID	Order Amount	Cluster	Estimated Unique Products
400000	100000	0	56
400001	9000	0	21
400002	300000	1	4

Table 10: Experiment 3 predictions

6 Conclusions And Future Work

This research aimed to address the following questions: "Can data mining techniques applied to customer profiling improve the accuracy of sales predictions? If so, which clustering algorithm provides the most accurate and efficient results?".

In pursuit of these questions, the results of the experiments underscore the importance of choosing the right clustering algorithm based on the specific characteristics and requirements of the data set. For instance, while K-means and GMM were found to be faster, they were limited by their assumptions and the quality of clusters they produced, meaning they are better suited for large data sets where speed is a priority and the data distribution aligns well with their premises. On the other hand, AHC and DBSCAN, despite being slower, excelled in producing high-quality clusters and handling outliers more effectively, making them suitable for smaller data sets.

Furthermore, the research highlights the potential value of outliers, which are often overlooked in data analysis. As demonstrated in the first experiment for DBSCAN, these outliers can represent important patterns in the data, such as the most popular products among customers, and can therefore provide valuable insights for businesses, which should think twice when deciding whether or not to remove them from the sales analysis.

In conclusion, my research demonstrates that customer profiling through data mining can be a reliable method for predicting future sales in real-world scenarios. However, the choice of clustering algorithm is critical and should be tailored to the specific characteristics and requirements of the data set, such as its size, distribution, and feature correlations. As we move forward, it is crucial to continue exploring and refining these methods to enhance their predictive capabilities and support more informed business decisions. Below are some example approaches for such future improvements.

Firstly, the performance of the clustering algorithms could be further enhanced by either fine-tuning their parameters or leveraging parallel computing techniques. Given the inherently iterative nature of these algorithms, parallelism could significantly reduce computation time, especially for large data sets.

Secondly, exploring additional clustering algorithms such as Spectral Clustering or OPTICS [28] could provide new insights and advantages. Ensemble methods that combine multiple clustering techniques might also be worth investigating. Additionally, we could consider evaluating a broader range of predictive algorithms beyond Linear Regression, Random Forest, and Gradient Boosting, including Support Vector Machines, Neural Networks, and XGBoost [29]. This could help in identifying the most effective approaches for accurate predicting.

Thirdly, the role of outliers in data analysis, which was highlighted in this research, deserves further exploration. More sophisticated methods for identifying and handling outliers could be developed, and the potential insights they can provide for businesses could be studied in more depth.

Lastly, the application of these clustering techniques could be extended beyond customer profiling to other areas of business, such as inventory management or marketing strategy. This would provide a more holistic view of the business and could lead to more informed and effective decision-making.

References

[1] Syed, A., Gillela, K., & Venugopal, C. (2013). The future revolution on big data. Future, 2(6), 2446-2451.

[2] Ukeni, C. S. (2015). Strategy Behind the Business Success of Amazon: A Case Study. Texila International Journal of Management, 1-16.

[3] Han, J., Kamber, M., & Pei, J. (2012). Data mining concepts and techniques third edition. University of Illinois at Urbana-Champaign Micheline Kamber Jian Pei Simon Fraser University.

[4] Papakyriakou, D., & Barbounakis, I. S. (2022). Data mining methods: a review. International Journal of Computer Application, 183(48), 5-19.

[5] Rajagopal, D. S. (2011). Customer data clustering using data mining technique. arXiv preprint arXiv:1112.2663.

[6] Ziafat, H., & Shakeri, M. (2014). Using data mining techniques in customer segmentation. Journal of Engineering Research and Applications, 4(9), 70-79.

[7] Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. IEEE Transactions on neural networks, 16(3), 645-678.

[8] Xu, D., & Tian, Y. (2015). A comprehensive survey of clustering algorithms. Annals of data science, 2, 165-193.

[9] McLachlan, G., & Krishnan, T. (1997). The EM Algorithm and Extensions. New York: Wiley.

[10] Berrar, D. (2019). Bayes' Theorem and Naive Bayes Classifier.

[11] Kuo, F. Y., & Sloan, I. H. (2005). Lifting the curse of dimensionality. Notices of the AMS, 52(11), 1320-1328.

[12] Li, D., & Liu, S. (2019). In Water quality monitoring and management.

[13] Lee, E. (2024, April 9). Secrets of PCA: A comprehensive guide to principal component analysis with Python and Colab. Retrieved from: https://drlee.io/secrets-of-pca-a-comprehensive-guide-to-principal-component-analysis-with-python-and-colab-6f7f3142e721.

[14] Rymarczyk, P., Gołąbek, P., Skrzypek-Ahmed, S., & Rzemieniak, M. (2021). Profiling and segmenting clients with the use of machine learning algorithms.

[15] Zhang, B., Tseng, M. L., Qi, L., Guo, Y., & Wang, C. H. (2023). A comparative online sales forecasting analysis: Data mining techniques. Computers & Industrial Engineering, 176, 108935.

[16] Choi, J. Y., & Lee, B. (2018). Combining LSTM Network Ensemble via Adaptive Weighting for Improved Time Series Forecasting. Mathematical Problems in Engineering, 2018, 8.

[17] Shahapure, K. R., & Nicholas, C. (2020, October). Cluster quality analysis using silhouette score. In 2020 IEEE 7th international conference on data science and advanced analytics (DSAA) (pp. 747-748). IEEE.

[18] Řezanková, H. A. N. A. (2018, September). Different approaches to the silhouette coefficient calculation in cluster evaluation. In 21st international scientific conference AMSE applications of mathematics and statistics in economics (pp. 1-10).

[19] Xiao, J., Lu, J., & Li, X. (2017). Davies Bouldin Index based hierarchical initialization K-means. Intelligent Data Analysis, 21(6), 1327-1338.

[20] Venkatesh, B., & Anuradha, J. (2019). A review of feature selection and its methods. Cybernetics and information technologies, 19(1), 3-26.

[21] Ali, P. J. M., Faraj, R. H., Koya, E., Ali, P. J. M., & Faraj, R. H. (2014). Data normalization and standardization: a technical report. Mach Learn Tech Rep, 1(1), 1-6.

[22] Larose, D. T., & Larose, C. D. (2014). k-nearest neighbor algorithm.

[23] Razali, N. M., & Wah, Y. B. (2011). Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. Journal of statistical modeling and analytics, 2(1), 21-33.

[24] Rokach, L., & Maimon, O. (2005). Decision trees. Data mining and knowledge discovery handbook, 165-192.

[25] Biau, G., & Scornet, E. (2016). A random forest guided tour. Test, 25, 197-227.

[26] Natekin, A., & Knoll, A. (2013). Gradient boosting machines, a tutorial. Frontiers in neurorobotics, 7, 21.

[27] Analytics Vidhya. (2020, August 15). MAE, MSE, RMSE, Coefficient of Determination, Adjusted R-Squared – Which Metric is Better? Medium.

[28] Ankerst, M., Breunig, M. M., Kriegel, H. P., & Sander, J. (1999). OPTICS: Ordering points to identify the clustering structure. ACM Sigmod record, 28(2), 49-60.

[29] Ramdani, F., & Furqon, M. T. (2022). The simplicity of XGBoost algorithm versus the complexity of Random Forest, Support Vector Machine, and Neural Networks algorithms in urban forest classification. F1000Research, 11, 1069.

[30] Stack Abuse. (n.d.). Stack Abuse. https://stackabuse.com/

[31] Chatfield, C. (1986). Exploratory data analysis. European journal of operational research, 23(1), 5-13.

[32] Papageorgiou, G., Grant, S. W., Takkenberg, J. J., & Mokhles, M. M. (2018). Statistical primer: how to deal with missing data in scientific research?. Interactive cardiovascular and thoracic surgery, 27(2), 153-158.