UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

ALGORITHM FOR INTERPRETABLE CLUSTERING USING DEMPSTER-SHAFER THEORY

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN COMPUTACIÓN

RICARDO MAURICIO VALDIVIA ORELLANA

PROFESOR GUÍA:
NELSON BALOIAN TATARYAN

MIEMBROS DE LA COMISIÓN:
FELIPE BRAVO MARQUEZ
JAZMINE MALDONADO FLORES

SANTIAGO DE CHILE
2024

Resumen

Algortimo para Clustering Interpretable usando la Teoría de Dempster-Shafer

El clustering es un método de aprendizaje no supervisado cuyo objetivo es identificar conjuntos de datos con características similares. La calidad de un modelo de clustering se mide a menudo por su validez en lugar de su precisión, utilizando indicadores como el Índice de Rand y el Coeficiente de Correlación. En los últimos años, ha surgido un interés creciente en crear modelos de clustering no solo válidos, sino también interpretables. La interpretabilidad se refiere a la capacidad del modelo para permitir que un usuario humano comprenda cómo y por qué el modelo llega a un resultado específico.

Los algoritmos de clustering actuales, como el K-means, son populares por su simplicidad y escalabilidad, pero a menudo son considerados como "cajas negras" debido a la falta de transparencia en sus resultados. Esto ha llevado a un enfoque creciente en la interpretación de los modelos de clustering y en el desarrollo de técnicas de explicación de modelos, como SHAP (SHapley Additive exPlanations), para proporcionar una interpretación clara de cómo se generan los resultados del clustering.

La solución propuesta en este proyecto es el desarrollo de un algoritmo de clustering que genera etiquetas para los datos y, utilizando el clasificador DS (Dempster-Shafer), produce reglas claras que aseguran la interpretabilidad para los usuarios. El desarrollo se realiza en dos etapas: la selección de etiquetas óptimas para el entrenamiento y la consolidación del algoritmo de clustering, incluyendo el entrenamiento y la predicción del clasificador DS para cada punto de datos.

El algoritmo DSClustering implementado logra una combinación efectiva de técnicas de clustering con interpretación mejorada a través de la generación automática de reglas categóricas y ajustes precisos en el proceso de entrenamiento del clasificador. El algoritmo se destaca por su capacidad para ofrecer resultados de clustering fiables y comprensibles, lo que mejora la transparencia y la confianza en la toma de decisiones basada en los datos. Esta combinación de validez y transparencia en los resultados de clustering representa un avance significativo en el campo del aprendizaje automático.

Abstract

Clustering is an unsupervised learning method aimed at identifying data sets with similar characteristics. The quality of a clustering model is often assessed by its validity rather than its accuracy, using measures such as the Rand Index and the Correlation Coefficient. Recently, there has been an increasing interest in creating not only valid but also interpretable clustering models. Interpretability refers to the model's ability to enable a human user to understand the how and why behind the model's specific outcomes.

Current clustering algorithms, like K-means, are favored for their simplicity and scalability, yet they are often viewed as"black boxe" due to their opaque results. This has led to a growing focus on understanding and interpreting clustering models, and in developing model explanation techniques, such as SHAP (SHapley Additive exPlanations), to provide a clear understanding of how clustering results are produced.

The proposed solution in this project involves the development of a clustering algorithm that generates labels for data and, using the DS (Dempster-Shafer) classifier, creates clear rules ensuring interpretability for users. The development occurs in two stages: selecting optimal labels for training and consolidating the clustering algorithm, including training and predicting with the DS classifier for each data point.

The implemented DSClustering algorithm achieves an effective combination of clustering techniques with enhanced interpretation through the automatic generation of categorical rules and precise adjustments in the training process of the classifier. The algorithm stands out for its ability to provide reliable and comprehensible clustering results, enhancing transparency and trust in data-driven decision-making. This blend of validity and transparency in clustering outcomes marks a significant advancement in the field of machine learning.

We are each a patchwork quilt of those who have loved us, those who have believed in our futures, those who showed us empathy and kindness or told us the truth even when it wasn't easy to hear. Those who told us we could do it when there was absolutely no proof of that.

- Taylor Swift

Acknowledgments

A mis incondicionales amig@s del colegio, Cata, Michi y Cristian, hemos compartido tantos años de risas, desafíos y triunfos. Cada recuerdo con ustedes es un tesoro que atesoro en mi corazón. Gracias por ser parte fundamental de mi historia.

Quiero expresar mi profundo agradecimiento a esas personas tan especiales que están el cielo e iluminan mi camino y que sé que me acompañan en cada jornada. Vanessa, eres una presencia constante en cada paso que doy, guiándome en esta travesía de la vida. Además, agradezco enormemente el regalo que me dejaste de una familia que siempre ha brindado su apoyo incondicional.

Quiero agradecer a Matí, aunque te uniste a mi vida más recientemente. Tu presencia ha marcado una diferencia significativa, y estoy agradecido por tenerte a mi lado. Gracias por cada palabra de cariño y ánimo, por cada meme o mensajito que siempre tienes en el momento más oportuno

A Belen que ha sido una fuente de alegría en estos años de U, nos acompañamos en momentos que podían parecer oscuros u que el destino no era el correcto. A Correa, con quien partí la especialidad y formó parte de todas esas noches hasta altas horas de la madrugada estudiando o jugando, pero siempre apoyándonos en los ramos, gracias también a Albani que se unió después con quien teníamos nuestras tardes de estudio y copuchas.

Whyatt, gracias por estar siempre allí para apoyarme. Tu confianza en mí ha sido un impulso constante, siempre dándome ánimo o acompañándome en tardes de juego para capear el tiempo

Santiago y Feña, ustedes son parte del grupo desde el principio de la universidad, y sin su amistad y compañía, este proceso no habría sido el mismo. Por esas tardes que siempre incluían alguna comida rápida y terminaban alegrando nuestros días, por esas arduas jornadas de estudio que terminaban en copuchas.

Por último quiero agradecer a mi querida mamá y a mi tía, ustedes son mis pilares. Su apoyo inquebrantable, contención y aliento, no solo en este viaje universitario, sino en toda mi vida, son invaluables. Gracias por ser mis faros en la tormenta.

Table of Content

List of Tables

List of Figures

Listings

Chapter 1

Introduction

1.1 Motivation

Due to the enormous amount of data generated from various sources, professionals from different fields, particularly in the field of data science, have created models that allow predicting unknown characteristics of a sample based on known features. Classification models are used to predict discrete features, while regression models are used to predict continuous values. These are known as supervised learning methods. Within unsupervised methods, there is clustering, whose objective is to identify datasets with similar characteristics.

When evaluating the quality of a model, the primary indicator considered is the accuracy of the machine learning model, measured as the difference between the predicted value and the previously unknown real value of a feature. For a classification model, the main performance indicators are accuracy, sensitivity, area under the ROC curve, and the $F1$ score [32]. For clustering models, there is a subtle difference because there is no real value to compare against. Therefore, their validity is evaluated instead of accuracy. The main indicators shift to the Dunn's Index [10], the Silhouette score[37] and the Correlation Coefficient [2].

In recent years, there has been an increased interest in creating highly accurate models that are also easy to understand in terms of the reasons for a certain prediction or grouping. In this context, the concept of interpretability emerges, referring to the model's ability to allow a human user to understand how and why the model arrives at a specific result. Despite the importance of interpretability, there is still a lack of agreement on its formal definition and how to measure it. Nevertheless, many authors agree that interpretability is crucial when users seek more information about the phenomenon generating the analyzed data [17], [27].

Both high performance and interpretability are desirable characteristics of a data science tool, although experience shows that high-performance methods generally have low interpretability, and vice versa. In fact, highly accurate regression methods such as gradient boosting and random forest are considered black-box algorithms due to their inherent lack of interpretability [16]. On the one hand, linear regression models and decision trees are recognized for their high inherent interpretability [12], however, linear models have the limitation of being unable to learn non-linear relationships between attributes, and decision trees

should have a small depth to be really interpretable; according to Peñafiel et al. (2020) [24], these characteristics make them less accurate.

In the work done by Peñafiel et al. (2020) [24], a classifier called Dempster-Shafer (DS classifier) has been created using Dempster-Shafer's theory of plausibility (Shafer, 1976) [29]. This model has proven to be equally effective as other black-box models in terms of accuracy while also having the advantage of being easily interpretable. Interpretability in this context refers to the model's ability to provide simple and understandable rules used for predicting outcomes through machine learning.

Due to the growing demand for clustering solutions that are both highly valid and interpretable, there is a need to continue developing methods that combine both features without seriously compromising the quality of any of them. Therefore, an interesting hypothesis from a research point of view is that it is feasible to generate a clustering algorithm that allows obtaining comparable results in terms of validity and interpretative ability compared to popular clustering algorithms such as k-means, DBSCAN, and agglomerative clustering but it is highly interpretable at the same time. This approach could help users better understand the clustering process and instill greater confidence in the results obtained.

The gap that this research aims to fill lies in the development of a clustering algorithm that provides a unique balance between validity and interpretability, a rare combination in current clustering methods. Unlike existing interpretable clustering algorithms, which often compromise validity, the proposed algorithm maintains high validity while providing simple rules for user understanding. It is also the only one that provides a measure of uncertainty for each cluster assignment. This dual approach increases user confidence and understanding of the underlying data patterns by addressing the need for reliable clustering solutions that are not only valid, but also transparent and understandable.

First, a standard clustering process is performed on the data, so that each of them is assigned to a cluster. Then, depending on the user's preference, the algorithm with the best silhouette score or the most frequent (most repeated) cluster for each data instance is chosen. Then, considering each cluster as a class, an interpretable classifier (DS classifier) is trained with this labelled data, which generates rules to assign the samples to a certain class, so that we have the rules of how the discovered clusters are formed.

Therefore, this work represents a contribution in the field of data science, as it offers a novel approach to clustering that combines the best of both worlds: the reliability of the best-known clustering methods and the clarity of interpretable models provided by the Dempster-Shafer classification method (known as DS classifier) developed by Peñafiel et al. (2020) [24], to obtain an interpretable clustering method.

1.2 Objectives

General Objective

The objective of this work is to create a clustering algorithm that balances the validity of clustering and model interpretation, and introduces at the same time a measure of uncertainty for each cluster assignment. This will improve the interpretation and reliability of the clustering process. Established metrics will be used to assess the performance of the method. The algorithm will be designed in order to maximize intra-cluster similarity and minimize inter-cluster similarity. It will also focus on interpretability, allowing users to understand the logic behind the clustering and to appreciate the level of certainty for the rules used in order to assign a sample to a cluster.

Specific Objectives

1. Establish a representative test dataset to evaluate the performance of the proposed clustering algorithm. This dataset should be large and diverse enough to cover different types of data and distributions, allowing the evaluation of the algorithm's ability to adapt to various situations and provide valid results. To achieve this goal, data must be carefully selected, and a cleaning and preprocessing process must ensure the quality and consistency of the dataset.

2. Define a method to adapt the Dempster-Shafer classifier to the clustering problem. Since this classifier has been primarily developed for classification problems rather than clustering, the peculiarities of the clustering problem must be considered, and ways to maximize validity in data clustering must be explored.

3. Develop an interpretability model specific to the suggested clustering algorithm. Interpretability is a key aspect in evaluating clustering algorithms, allowing an understanding of how data grouping is achieved and how much effort is required to understand the generated model. To achieve this goal, specific metrics must be defined to measure the model's complexity and its ability to explain clustering results.

4. Conduct comparative experiments between the proposed clustering algorithm and other reference clustering methods. In these experiments, both the validity of clustering and the interpretability of the model must be evaluated using the previously defined metrics. This will determine if the proposed clustering algorithm offers an adequate balance between validity and interpretability and how it compares to other reference clustering methods.

5. Identify areas of improvement for the suggested clustering algorithm and explore new possibilities to enhance its validity and interpretability. Based on the results obtained in comparative experiments and evaluation metrics, weaknesses of the algorithm must be identified, and ways to optimize its performance must be explored. This will ensure that the proposed clustering algorithm remains relevant and useful in the future.

1.3 Document Structure

This document consists of 7 chapters, along with a Table of Contents and indexes for Tables and Figures, to aid in understanding the document and the analysis of the obtained results.

In Chapter 1, which corresponds to the current chapter, motivations and objectives for the study of the area were presented, providing context to the problem.

In Chapter 2, the necessary background, concepts, and technologies are detailed to understand the problem and the solution.

In Chapter 3, the design is summarized, and the reasoning behind the decisions made is described. Additionally, the developed pipeline and clustering architecture are discussed.

In Chapter 4, label selection are discussed in detail.

In Chapter 5, the implementation of clustering is explained in detail.

In Chapter 6, specific experiments conducted, along with the data used in each, are indicated. Additionally, results are shown and explained using tables and graphs to compare the outcomes.

In Chapter 7, conclusions are drawn for the work done in this thesis. Accomplished objectives and those not fully achieved are described. In addition to this, possible options for future work based on the development of this thesis are suggested.

Chapter 2

State of the Art

This chapter presents relevant topics to understand the solution developed throughout this work and to comprehend the addressed problem.

Currently, interpretable clustering algorithms are a rapidly developing research area. Traditional clustering algorithms are often considered "black boxes" due to the lack of transparency in the results, leading to increased attention to the interpretation of clustering algorithms [19]. Furthermore, model interpretation has become an active research area to enhance the transparency and interpretability of machine learning models [21].

2.1 Frameworks to explain models

According to various authors, model explanation is an important area in the development of interpretable clustering methods, especially in medical applications, where understanding results is crucial for clinical decision-making. A promising solution is the use of model explanation techniques[33][6] to provide a clear interpretation of how clustering results are generated [19].

The authors propose a unified framework for interpreting predictions called SHAP[1] (SHapley Additive exPlanations). This framework can be used to explain the output of any machine learning model, including clustering models. The SHAP framework provides a way to attribute the prediction of an individual data point to its features, allowing a better understanding of how the model is making predictions. Therefore, SHAP can be used as a technique to explain clustering models and provide an interpretation of their results.

[1] `https://github.com/slundberg/shap`

2.2 Popular Clustering Methods

2.2.1 K-means Clustering

K-means clustering [5] is a data clustering method used to identify subgroups in the data, where data points in the same subgroup are very similar, while points in different subgroups are very different. It is an iterative algorithm that attempts to partition the dataset into K distinct, non-overlapping subgroups, where each data point belongs to a single group. The goal is to make intra-cluster data points as similar as possible while keeping groups as different (distant) as possible.

The process begins by randomly selecting 'k' points in the dataset as initial cluster centers or centroids, where a centroid is the central point or geometric mean of all points in a cluster. Each data point is then assigned to the nearest centroid based on a distance metric such as Euclidean distance, effectively partitioning the data set into k clusters. After this assignment, the centroids are recalculated as the average of all points in each cluster. This process of centroid assignment and recalibration is repeated iteratively until the centroids stabilize, i.e., their positions do not change significantly, or a specified number of iterations is reached. This results in a partitioning of the dataset where the variance within each cluster is minimized, ideally reflecting meaningful groupings in the data.

The K-means algorithm is considered one of the most used due to its simplicity. To apply the K-means algorithm, it is recommended to scale or standardize the data. Additionally, the elbow method can be used to select the optimal number of groups.

The K-means algorithm has several advantages[9], such as its simplicity, scalability, and guaranteed convergence. It is also easy to adapt to new examples and generalizes to groups of different shapes. However, it also has some disadvantages, such as the need to manually choose the number of groups and dependence on initial values. Additionally, it struggles with clustering data of different sizes and densities.

In general, the K-means algorithm is a useful technique for data exploration and identifying subgroups in the data. Its simplicity and scalability make it easy to implement, and it can adapt to a variety of situations. However, it's essential to consider its limitations and explore other clustering techniques if the data does not fit well with the algorithm's assumptions.

2.2.2 DBSCAN Clustering

The DBSCAN algorithm is a density-based clustering method used to examine spatial data[31]. DBSCAN can find arbitrary-shaped clusters without being affected by noise and can identify the densest part of data samples while ignoring less dense areas or noise. The DBSCAN algorithm is straightforward and defines clusters by estimating local density. The clustering process can be divided into four stages[7]:

1. For each observation,, look at the number of points within a maximum distance ε from it. This region is called the ε-neighborhood of the observation.

2. If an observation has at least a certain number of neighbors, including itself, it is considered a core observation. In this case, a high-density observation has been detected.
3. All observations in the ε-neighborhood of a core observation are considered part of the same cluster.
4. Repeat the process for all cluster observations until no more core observations are found.

DBSCAN can find clusters with arbitrary shapes and has a notion of noise, being robust in detecting outliers[36]. Additionally, it is the fastest clustering method but is only suitable when a very clear search distance can be used, and it works well with all potential clusters.

2.2.3 Agglomerative Clustering

Agglomerative clustering [11] is a cluster analysis method that aims to group clusters to form a new one or separate an existing one to give rise to two others. This method is based on the main idea that closer objects are more related than those far apart, connecting "objects" to form "groups" based on their distance [35].

In agglomerative clustering, you start with each object in its own cluster and merge the closest clusters until all objects belong to the same cluster[22]. This process is done iteratively, and the final result is a dendrogram that shows how the objects were grouped.

The effectiveness of agglomerative clustering depends largely on the choice of the distance measure and the method of cluster merging. Some of the most common merging methods are the single-linkage method, the complete-linkage method, and the average-linkage method.

2.3 Existing Interpretable Clusterings

In recent years, several interpretable clustering methods have been proposed, such as the Explanatory Hierarchical Clustering (EHC), which utilizes a traditional hierarchical clustering algorithm and provides a clear explanation of the resulting groups [3]. It proposes a modification to the hierarchical clustering algorithm to incorporate prior knowledge in the form of relative constraints.

It introduces a hierarchical algorithm that finds a clustering solution satisfying the given constraints. Relative constraints can be used to represent any hierarchical knowledge of the domain and are easy to represent [18]. A constrained clustering approach is suggested, enabling the incorporation of prior knowledge to support analysis and draw meaningful conclusions. This approach can be employed to validate clustering by taking an independent sample or many independent samples from the underlying population.

The challenge of using algorithms based on decision trees with reduced depth to build interpretable models is that, while these algorithms are easy to interpret, their validity may

decrease due to the lack of complexity in the rules defining the classification used to interpret the rules defined by clustering. This means that by reducing the depth of the tree, the amount of information that can be used for prediction is limited, potentially resulting in less valid clustering.

Furthermore, reducing the depth of the tree can lead to the loss of important information and the generation of simplified rules that do not adequately represent the data. This can result in a misinterpretation of the model's results and, ultimately, incorrect decision-making.

On the other hand, there is a type of interpretable clustering based on hyperplanes [14]. In this approach, multiple convex polyhedra are used to represent the clusters, and each polyhedron is defined by a set of hyperplanes. Each hyperplane divides the space into two parts, and points falling in the same part are assigned to the same cluster. The advantage of this approach is that convex polyhedra are easy to interpret as they can be visualized in the feature space. Additionally, the hyperplanes defining the polyhedra can be interpreted as logical rules describing the characteristics of points belonging to each cluster.

However, this approach also has some weaknesses. Firstly, choosing the number of polyhedra and hyperplanes can be challenging and may require manual adjustments. Additionally, convex polyhedra may not be able to represent clusters with complex or non-convex shapes. Finally, the approach can be computationally expensive, as it requires the optimization of multiple convex polyhedra.

2.4 Dempster Shafer Clustering

The Dempster-Shafer theory, also known as the theory of belief functions, is a general framework for reasoning with uncertainty, with applications in fields such as artificial intelligence, statistics, and computer science. Developed independently by Arthur P. Dempster and later expanded by Glenn Shafer, it offers an alternative to traditional probability theory by allowing for the combination of evidence from different sources and the representation of various degrees of belief [30].

In contrast to probability theory, which requires precise probabilities for each event, the Dempster-Shafer theory allows for "belief functions" that do not need to sum to one, enabling the representation of a range of belief and the expression of uncertainty. This is achieved through two functions: the belief function and the plausibility function. The belief function provides a measure of the total belief that supports a given set of outcomes, while the plausibility function measures how much belief could potentially be placed in the same set of outcomes if more evidence were available [38].

Clustering algorithms using Dempster-Shafer theory offer an intriguing approach to addressing the challenge of grouping data into coherent sets. Based on the evidence theory developed by Dempster and Shafer, these algorithms provide a unique perspective for analyzing complex data, where the precise assignment of elements to groups can be challenging due to ambiguity or overlap between data features.

2.4.1 Evidential Clustering of Proximity Data

The Evidential Clustering of Proximity Data [8] is a specific approach within clustering algorithms that leverages Dempster-Shafer theory. This method is based on calculating the proximity between objects to assign the degree of conflict generated between them. Proximity is determined using a distance measure, such as Euclidean or Mahalanobis, capturing the similarity or relationship between the features of the objects.

Once the proximity between objects is calculated, belief functions are employed to assign the degree of conflict between them. These functions allow for more flexible handling of uncertainty and lack of information compared to traditional clustering methods. By assigning degrees of conflict, overlap or ambiguity between objects can be captured, which is particularly useful in complex datasets where boundaries between groups may be fuzzy.

In summary, the Evidential Clustering of Proximity Data uses object proximity to calculate the degree of conflict and employs belief functions to assign this degree. This approach addresses uncertainty and overlap in the data, providing a more robust form of grouping that can capture subtle relationships between objects.

2.4.2 Evidential C-means and Constrained evidential C-means

The Evidential C-means and the Constrained Evidential C-means are clustering algorithms based on the fuzzy C-means method, incorporating Dempster-Shafer theory (DS) to achieve a semi-supervised approach and integrate auxiliary information. These algorithms are known as fuzzy and hard clustering, respectively.

The Evidential C-means [20] is a fuzzy variant of the C-means algorithm, where each object is assigned a degree of membership to each group, represented by a belief distribution. These membership degrees are calculated using Dempster-Shafer theory, enabling the capture of uncertainty associated with assigning objects to groups. By combining proximity information between objects and belief functions, a fuzzy partition reflecting uncertainty in object-group assignment is obtained.

On the other hand, the Constrained Evidential C-means [1] is a hard variant of the C-means algorithm, imposing an additional constraint on the assignment of objects to groups. This constraint is based on auxiliary information, such as labels or similarity constraints between objects. Dempster-Shafer theory is used to integrate this auxiliary information into the clustering process, allowing the generation of partitions more consistent with provided prior knowledge or constraints.

In summary, Evidential C-means and Constrained Evidential C-means are clustering algorithms based on the C-means method, utilizing Dempster-Shafer theory to make them semi-supervised and integrate auxiliary information. Evidential C-means focuses on generating fuzzy partitions, while Constrained Evidential C-means aims to generate partitions consistent with provided constraints or auxiliary information. These algorithms offer a robust form of clustering by capturing uncertainty and enabling the incorporation of prior knowledge.

Despite the advantages offered by DS-based algorithms in terms of handling uncertainty and integrating auxiliary information, it is essential to recognize that none of these algorithms directly address the interpretability issue or provide guidance on how to make them more interpretable. While the generality achieved by using different data sources is mentioned, additional strategies such as visualization techniques, relevant feature selection, and appropriate evaluation metrics must be explored to ensure that clustering results are understandable and useful in decision-making.

2.5 Analysis in Cluster Validation

2.5.1 Silhouette

Silhouette analysis is a method used to determine the quality of clustering in unsupervised learning. It provides a concise graphical representation of how well each object lies within its cluster, which is crucial for validating the consistency within clusters of data. The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation).

The Silhouette score for each data point is a number between -1 and +1. Here's how it's calculated:

- Cohesion (a): For each data point, the average distance between the point and all other points in the same cluster is calculated. This measures how close each point in a cluster is to the points in its own cluster.
- Separation (b): For the same data point, calculate the average distance from all points in the nearest cluster to which the point does not belong. This measures how far each point is from points in other clusters.
- Silhouette Value (s): The silhouette value for each point is then calculated using the formula:

$$s = \frac{b - a}{max(a, b)} \tag{2.1}$$

  - If s is close to +1, it indicates the data point is well clustered and far from other clusters.
  - If s is close to 0, it suggests the data point is on or very close to the decision boundary between two neighboring clusters.
  - If s is close to -1, it indicates that the data point has been assigned to the wrong cluster.

By averaging the silhouette scores of all the points, you can get an overall measure of how well the clusters are separated. High average silhouette values indicate well-separated and well-defined clusters, while low values indicate overlapping clusters. This analysis is often

used to determine the optimal number of clusters by comparing the average silhouette scores for different values of k in k-means clustering, or to validate the quality of clusters formed by any clustering algorithm. [26]

2.5.2 Dunn's index

Dunn's index is a metric used in clustering analysis to evaluate the quality of the clusters formed. It measures the ratio between the smallest distance between observations in different clusters to the largest intra-cluster distance. A higher Dunn's index indicates well-separated clusters.

Dunn's index is calculated as follows:

1. Inter-cluster Distance: First, determine the smallest distance between observations in different clusters. This could be the distance between the closest points in different clusters, or the distance between cluster centroids, depending on the variant of the index used.
2. Intra-cluster Distance: Next, find the largest distance between observations within the same cluster. This distance usually represents the diameter of the largest cluster.
3. Dunn's Index: The index is then the ratio of the smallest inter-cluster distance to the largest intra-cluster distance.

$$Dunn's\ Index = \frac{Smallest\ Inter - cluster\ Distance}{Largest\ Intra - cluster\ Distance} \tag{2.2}$$

A higher Dunn's Index indicates a better clustering solution, where clusters are compact (small intra-cluster distances) and well-separated (large inter-cluster distances).

It's particularly useful for identifying sets of clusters where the members of each cluster are close to each other (high intra-cluster similarity) and far from members of other clusters (high inter-cluster dissimilarity). The index is beneficial for determining the optimal number of clusters in a dataset, as it helps in identifying a clear separation between clusters.

2.6 Tradeoff Between Clustering Validity and Interpretability

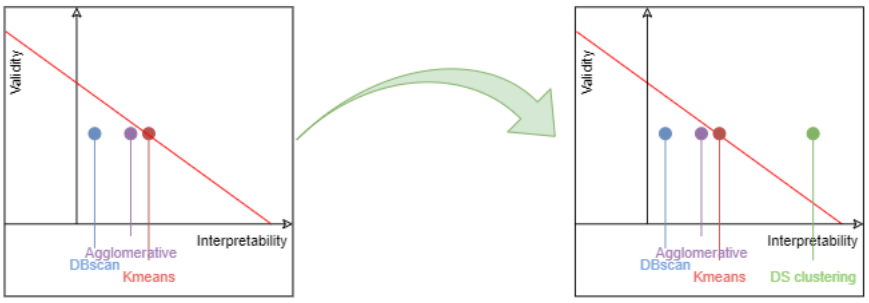

Figure 2.1: Interpretability goal for DS clustering

At present, interpretability in clustering algorithms remains a qualitative rather than a quantitative measure. Unlike validity, which can be measured by various statistical indices and scores, interpretability lacks a universally accepted metric for quantification. There have been efforts to quantify interpretability, such as Sandhya Saisubramanian's study "Balancing the Tradeoff Between Clustering Value and Interpretability" [28], but a definitive metric has yet to emerge. It's inherently subjective, often judged on how the clustering results can be understood and explained by human analysts. The goal in developing new clustering methods is to create a model that maintains validity levels similar to those of popular clustering techniques, where performance is context dependent, and to significantly improve interpretability, as indicated by the shift in the figure 2.1. This means that while maintaining comparable validity (which can vary depending on the context of the data), the new algorithm aims to make the clustering results more accessible and understandable to users, thereby facilitating better decision making and insights.

The interpretability of popular clustering algorithms is subject to personal judgement. The qualitative nature of interpretability in clustering stems from the fact that different algorithms present results in unique ways that may resonate differently with different observers. For example, K-means, with its straightforward centroid-based approach, is generally considered to be highly interpretable due to its simplicity and the ease with which one can visualise the partitioning of the data. Conversely, DBSCAN's density-based grouping can result in complex cluster shapes that may be less immediately clear, but offer deeper insights to those familiar with the domain-specific nuances of the data. In addition, agglomerative clustering provides a hierarchical view of data groupings that can be highly informative, but may require a more sophisticated level of analysis to fully appreciate. The interpretability of such a method is therefore highly individualistic, relying on the analyst's ability to decipher and communicate the meaning of the hierarchical structure.

In conclusion, although interpretable clustering algorithms are an area of ongoing research, there is a growing emphasis on developing methods that allow greater understanding and transparency in clustering results, especially in medical applications where interpretation is crucial.

Considering the data presented in the state of the art, emphasizing the importance of interpretability in clustering algorithms and recognizing the limitations in some existing methods, an opportunity for improvement in generating more understandable and transparent results is identified. With the aim of contributing to this opportunity, the proposal involves creating a new clustering algorithm that integrates elements of interpretation, simplicity, and effectiveness in data clustering. The goal is to overcome the limitations observed in other approaches, with interpretability being a central aspect in the formulation of the new algorithm.

Chapter 3

Description of the Solution

In this chapter, the problem that motivates this work and the proposed solution, along with the technologies used, are explained in a general manner.

3.1 Description

The challenge of finding new clustering algorithms lies in the quest for methods that not only generate valid clusters but are also inherently interpretable. Most advanced clustering algorithms tend to produce effective results but often lack a clear and understandable interpretation of the generated clusters. This issue is crucial, especially in applications where an intuitive understanding of the results is essential for informed decision-making. In practice, the lack of interpretability can constrain the adoption of these algorithms in environments where understanding the clusters is fundamental.

The current solution is based on significant prior work carried out by Sergio Peñafiel in his thesis entitled "Intepretable method for general classification using Dempster-Shafer theory". In this work, he developed a classification algorithm that overcomes the problems of interpretability associated with such algorithms without compromising precision.

By comparing this classifier with typical classifiers such as KNN or NB, it was found to have similar accuracy to classical classification methods. The results obtained in controlled scenarios were as expected, with the model often achieving perfect classifications. In all cases, the accuracy was not less than 10% of the best model compared.

In view of these results, the question arises: why not apply this algorithm, which has shown such promising results, to other purposes such as clustering?

3.2 Proposed Solution

The proposed solution involves the development of a clustering algorithm capable of generating labels for the provided data. Additionally, utilizing the DS classifier, the algorithm generates clear rules that ensure interpretability for users. It works on two simple stages. The fisrt one is called Label Selection and consists of applying a smart clustering technique (which will be detailed later) in order to find the clusters. Then it considers each cluster as a class and assigns to each sample a label indicating which class the sample belongs according to the clustering process. The second stage consists of training an interpretable classifier using the already labelled data in order to have an explanation about what are the criteria or rules that are used to assign one sample to a class (cluster). In this case we will use the DS classifier which produces rules for explaining the classification process, thus explaining what are the rules for assigning a sample to a cluster.

The diagram in the figure 3.1 graphically represents the process flow of the clustering method described above, that takes user input and data and produces labels (clusters) and rules.

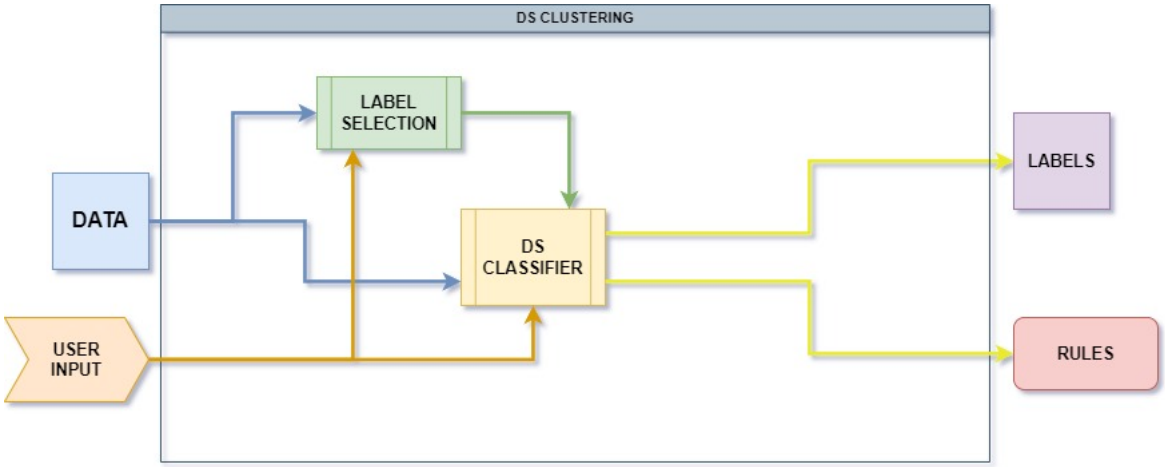

Figure 3.1: General DS clustering diagram

Below, each element is explained in detail.

- **Data**: This is the basic input to the system. It consists of the samples to be clustered.
- **User Input**: This input sets meta-parameters that influence the clustering process. It consists of 2 main parameters:
  - Most Voted Label: this can be set to True (default value) in order to make the process to use the most voted label strategy for assigning a cluster label to a sample and False in order to use the best label, which will be explained later.
  - Number of clusters: To set a specific number of clusters the process should find.
- **Label Selection**: This stage involves selecting appropriate labels for the data based on the data itself and user input. It includes clustering the data points with specific characteristics or identifiers using popular clustering techniques.

- **DS classifier**: This is the component where the classification algorithm based on Dempster-Shafer theory operates. It takes as input the data original data plus the information about to which cluster was it assigned in the previous step. This information is used to train the classification model, which will produce the rules for classifying a sample with already assigned and selected labels.
- **Labels**: These are the clusters assigned to each instance of the data by the DS clusterubg process.
- **Rules**: The decision rules used to reproduce the operation of the clustering model are derived from the DS classifier, based on Dempster-Shafer theory.

Now lets take a closer look at the Label Selection stage. This process takes the data to be clustered, the number of clusters and the label assignment policy defined by the user as input and runs three clustering algorithms independently: Kmeans, DBscan and Agglomerative. Each of the clustering process will assign one data sample to a cluster. The final label a sample gets is determined according to the user preferred policy declared at the beginning of the process and taken as an input by the process. There are two possible strategies to choose from: most voted label or best value.

The principal components of this stage, which are shown in figure 3.2 are:

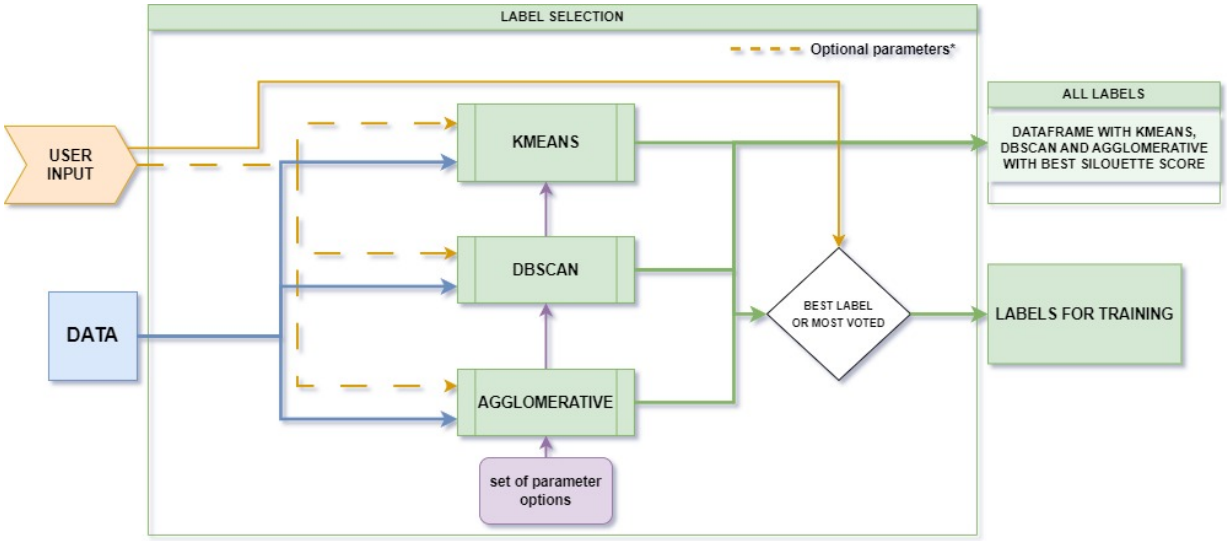

Figure 3.2: Label Selection Diagram

- **KMeans, DBSCAN, Agglomerative**: These are three different clustering algorithms. Each algorithm will be applied to the dataset to generate a set of labels. KMeans is a centroid-based clustering algorithm, DBSCAN is a density-based clustering algorithm, and Agglomerative is a hierarchical clustering algorithm.
- **Optional Parameters**: These are the optional parameters that can be set for each clustering algorithm to influence their behaviour, such as the number of clusters or the strategy to use to assign the labels to the data saples

- **Set of Parameter Options**: Refers to the different parameter settings that can be applied to the clustering algorithms to choose the best ones based on the silhouette score. The parameters consist of three sets of options. The first set determines the number of clusters for **Kmeans** and **Agglomerative** methods. The second set includes eps values, and the third set includes *min_samples* used to identify the optimal **DBscan**.
- **All Labes** is a Dataframe with KMeans, DBSCAN, and Agglomerative with Best Silhouette Scores: After the clustering algorithms have been run, a dataframe (a table-like structure used for handling data) is created which includes the results of all three clustering algorithms. The best silhouette score is used to determine the quality of the clustering.
- **Label Assignment Strategy** chooses the Best Label or Most Voted according to the input given by the user in order to determine the final label for each data sample. There are two implemented strategies and the user can chose which one to use. One strategy is to use the label assigned by the algorithm which obtained the best silhouette score considering all data samples (hence all data instances will receive the label assigned by that algorithm). The other one is to use the label receives the most votes (i.e., the label that is most frequently assigned to the data instance by the three algorithms).
- **Labels for Training**: These are the final selected labels that will be used for further training in a supervised learning context, i.e, the DSClassifier.

K-means, DBSCAN and Agglomerative Clustering have been chosen as popular clustering algorithms over others for several compelling reasons. Firstly, these methods are examples of different approaches to clustering, providing a broad view of the range of clustering techniques. K-means is known for its simplicity and efficiency, making it very suitable for a wide range of applications, especially when dealing with large datasets and the need for fast, generalisable clusters. DBSCAN is characterised by its ability to identify clusters of arbitrary shape and its robustness to outliers, overcoming the limitations of distance-based clustering methods such as K-means. Agglomerative Clustering, representing hierarchical clustering methods, provides a detailed insight into the data structure through its dendrogram, allowing for a more nuanced understanding of cluster relationships. Together, these algorithms cover a significant portion of clustering needs, from simple partitioning to complex hierarchical and density-based clustering, making them a comprehensive guide to many practical applications.

The second stage centers on training an inherent interpretable classifier, in this case the DS classifier, which will be able to predict with a certain accuracy to which class a data sample belongs. The accuracy will depend on how "good" the clusters are in terms of clearly differentiating them from each other. Subsequently, the presentation of the rules providing interpretability for the clustering algorithm will be produced by the algorithm, which adds the interpretability component to the developed model. Since the classifier is a supervised machine learning method. Therefore, it is essential to have a control group. For this purpose, it was decided to use the features chosen by the user as "best label" or "most vote", since it is expected that the best performing or most repeated features among the different methods will be the most suitable clusters for the data.

The process implemented by the DS classifier is described in figure 3.3, showing the main elements and steps.

- **Create Model**: This process uses the data and user input to create a DS classifier. This model incorporate the Dempster-Shafer theory to manage uncertainty and make predictions.
- **Categoric Rules**: These are rules derived from the data that determine how instances are classified. They are based on the attributes or features within the data, and in this case, on the labels obtained by the clustering algorithms.

  To achieve this, we use the 3 clusterings already obtained from popular clustering methods and provide them as input to the model. Since they are clusters, they are considered as categorical variables that contribute to the training of the classifier. It is important to emphasise that this process is independent of the clustering method used for training. Furthermore, since they are created variables, their rules are not included in the final result presented to the user. This means that they are only used to improve the performance of the classifier when attempting to predict the final cluster for each data instance, thereby improving the accuracy and effectiveness of the classifier.
- **Training**: This step involves training the DS classifier using a portion of the data. Training allows the model to learn from the data, adjust its parameters, and improve its categoric rules for making predictions.
- **Predict**: After the model has been trained, it can make predictions (assign a sample to a class/cluster) on the whole data
- **Labels**: These are the output from the classifier for each instance of the data after the prediction step. Labels are the classifier's conclusions about which cluster each instance belongs to.
- **Rules**: These may be the specific decision rules or logic that the classifier uses to assign labels to data instances. These rules are likely based on the categoric rules refined during training.

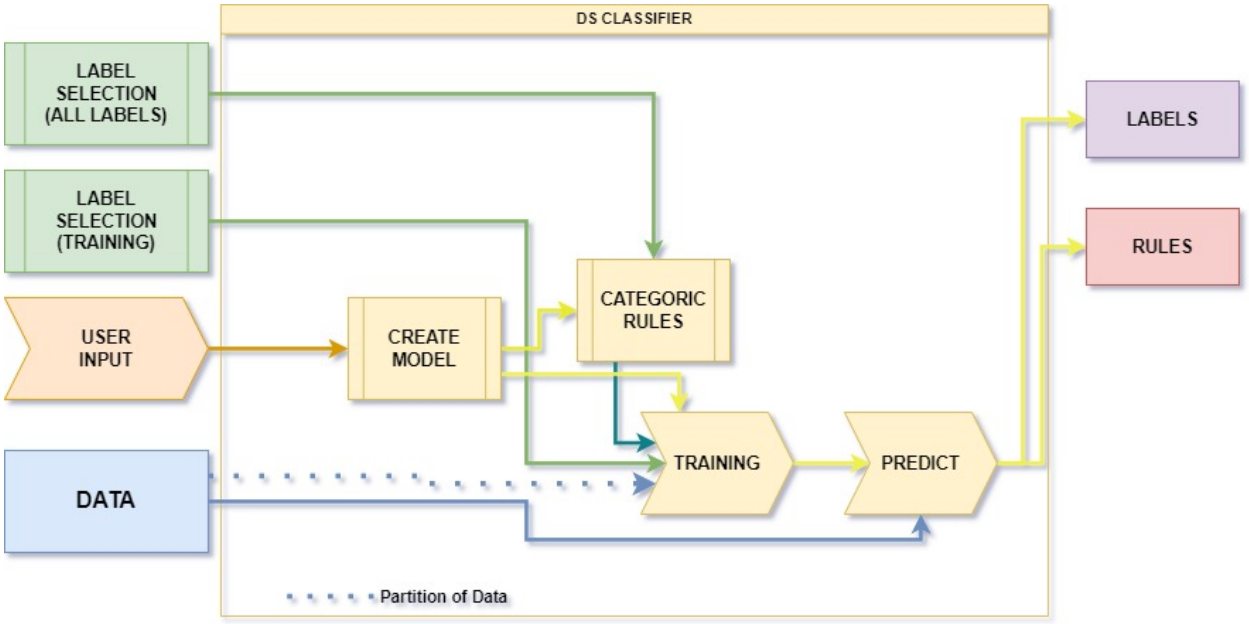

Figure 3.3: DS classifier usage diagram

It is important to note that all features of the DS classifier are applicable and inherited by DS clustering, more information can be found in the github documentation of the DS classifier library.[1]

Based on the concepts discussed in the previous chapter, the validity of the clustering algorithm's results will depend on the silhouette score. This metric will be used to measure the validity of the clusters formed by the algorithm. As the silhouette score is a measure of how well a data point has been assigned to its cluster, it allows for a quantitative assessment of the degree to which the clusters are distinct and well-separated.

The interpretability will be evaluated qualitatively, with the silhouette score providing a quantitative validation. Acknowledging the lack of a quantitative benchmark for interpretability, as previously discussed, the focus will shift to evaluating the clustering results based on human intuition and logic. This qualitative assessment will draw on domain expertise to determine the extent to which the clustering outcomes are meaningful, insightful, and align with the intuitive understanding of the data's structure and relationships.

[1] `https://github.com/Sergio-P/DSGD`

Chapter 4

Solution Implementation

In this chapter, a detailed description of the implementation of the interpretable clustering algorithm is provided.

4.1 Tools used

The tools to be used in the development of this work include:

- Python: is a high-level programming language widely used due to its ability to leverage a vast array of libraries. Specifically, it is the language in which the DS classifier is written. Throughout the development of this work, the following libraries were employed:
  - Pandas: Pandas is a powerful Python library designed for data analysis and manipulation. It provides flexible data structures, such as DataFrames, enabling efficient organization and analysis of tabular data.[1]
  - Numpy: Numpy is a fundamental library for numerical computing in Python, offering a set of functions and operations that facilitate efficient manipulation of arrays and multidimensional matrices.[2]
  - Scikit-learn (Sklearn): Sklearn is a machine learning library for Python that provides simple and efficient tools for predictive analysis and data mining. It includes a variety of supervised and unsupervised learning algorithms. [3]
  - Matplotlib: Matplotlib is a 2D visualization library for Python that allows the creation of high-quality graphs. It provides an interface for generating static plots, interactive graphs, and custom visualizations.[4]

[1] `https://pandas.pydata.org/`
[2] `https://numpy.org/`
[3] `https://scikit-learn.org/stable/`
[4] `https://matplotlib.org/`

– DSGD: DSGD corresponds to the library containing the DS classifier designed by Sergio Peñafiel. This classifier ensures inherent interpretability for our clustering algorithm.[5]

The successful implementation of the proposed solution is facilitated by the use of these tools, which form the essential technological infrastructure for processing data efficiently and presenting results derived from the clustering algorithm. Proper integration and precise handling of these tools are of vital importance to achieve the objectives outlined in this project.

To ensure the correctness of the code and compliance with PEP8 standards[6], the flake8, isort, trailing-whitespace, and mixed-line-ending hooks will be used. The development will take place in Google Colab[7] and will be subsequently saved on GitHub, where the repository name is *CDSGD*[8]. Finally, the solution will undergo thorough testing, primarily focusing on the validity of the assigned labels and the interpretability of the rules. Pandas and Matplotlib will be used for calculations, label management, and visualizing label assignments, comparing them with popular clustering algorithms.

For the implementation of this algorithm, the functions performed for the previously mentioned DSclassifier algorithm were taken into account.

Overall, the entire code has been developed using Python as the main programming language, which provides high flexibility and ease of use.

This chapter has a unique section that thoroughly addresses the functions for generating clusters using the classifier as a base.

4.2 DS clustering

This chapter section can be divided into four main subsections: Data Preparation, Automatic Generation of Categorical Rules, Classifier Fitting, and Label and Rule Retrieval.

4.2.1 Data Preparation

The initial step involves creating the DS clustering class, which accepts a dataframe containing the data for clustering. This leads to label selection as explained in the previous chapter, using the *"select_best_clustering()"* function to gather labels generated by popular clustering methods.

Depending on user preference, one can choose labels with the best silhouette score (DS clustering Best Labels) or those most frequently occurring for each data instance(DS clus-

[5] `https://github.com/Sergio-P/DSGD`
[6] `https://peps.python.org/pep-0008/`
[7] `https://colab.research.google.com`
[8] `https://github.com/ricardo-valdivia/CDSGD`

tering Most Voted Label). This selection is facilitated by the *"most_voted"* attribute, which defaults to false, prioritizing the best labels.

```
    ...
if not most_voted:
  best = selector.get_best_labels()
else:
  best = selector.get_most_voted()
    ...
```

Listing 4.1: The script decides whether to use the best labels from the selected clustering algorithm or the most voted label across different algorithms.

Upon obtaining the labels, we can initialize the classifier with the number of classes equal to the clusters defined by the clustering methods, which was given by the user as parameter, that would be defined as "DS clustering with number of clusters". This setup, paves the way for later generating automatic rules, drawing the expertise of popular clustering labels and the model fit.

4.2.2 Labels Selection

This section provides a comprehensive description of the initial phase of the project, which primarily involves the selection of labels intended for training the classifier. The overarching goal of this phase is to lay the foundation for generating clusters in the subsequent stages of the project, aiming to ensure optimal training conditions for the classifier.

The ClusteringSelector is a class created for identifying the optimal clustering algorithm for a given dataset. It compares various clustering methods such as K-Means, Agglomerative Clustering, and DBSCAN, and selects the most effective one based on performance metrics.

```
class ClusteringSelector:
        .
        .
        .
    def __init__(self, data, cluster=None):
        self.data = data
        self.best_algorithm = None
        self.best_params = [0]*5
        self.best_labels = None
        self.cluster = cluster
        self.kmeans_labels = None
        self.agglomerative_labels = None
        self.dbscan_labels = None
        .
        .
        .
```

Listing 4.2: This piece of code shows the initialization of the ClusteringSelector class, setting up necessary attributes for later use.

The selection of labels is a crucial part of the clustering process. Here is how ClusteringSelector manages it for the "best value" strategy:

- Initialization of Algorithms: It initializes multiple clustering algorithms with different parameters. This includes determining the number of clusters, linkage criteria, and other relevant parameters.
- Algorithm Evaluation: Each algorithm is applied to the dataset, and its performance is evaluated using metrics such as silhouette scores. The silhouette score is a measure of how similar an object is to its own cluster compared to other clusters.
- Label Selection: The algorithm with the highest performance score is selected. The labels generated by this algorithm are considered the best representation of the data clusters.

```
def select_best_clustering(self):
    # Test Parameters
    n_clusters_values = [2, 3, 4] if self.cluster is None else [self.cluster]
    # ... [code for initializing and evaluating clustering algorithms] ...
    # Comparing scores and selecting best algorithm
    if kmeans_score > best_score_kmeans:
        self.best_params[0] = n_clusters
        best_score_kmeans = kmeans_score
        self.kmeans_labels = kmeans_labels
    # ... [similar code for other algorithms] ...
```

Listing 4.3: This section shows how the class tests different clustering algorithms and parameters, then selects the best based on performance metrics.

For implementing the "Most Repeated Label" strategy the ClusteringSelector can determine the most repeated label across different algorithms. This process involves:

- Data Normalization or Standardization

  Prior to applying clustering algorithms, normalizing or standardizing the data is essential. This step is critical for the following reasons:

  - Uniformity of Scale: Ensures all features contribute equally to the analysis.
  - Accuracy Improvement: Reduces distortions caused by differences in feature scales.
- Label Comparison: Comparing the labels assigned to each instance by the different algorithms.
- Majority Voting: Selecting the label that appears most frequently for each data instance. This approach enhances the reliability of the clustering assignment.

```
def get_most_voted(self):
    labels_df = self.get_cluster_labels_df()
    row_modes = labels_df.mode(axis=1)
    # ... [code to determine the most repeated label] ...
    return np.array(row_modes[0])
```

Listing 4.4: This method calculates the most frequently occurring label for each data instance across different clustering results.

Using ClusteringSelector streamlines the process of selecting optimal clustering labels, ensuring that the best algorithm is chosen for a given dataset. The methodology of selecting the most repeated label and prior data normalization are vital steps contributing to the validity and reliability of clustering outcomes.

4.2.3 Automatic Generation of Categorical Rules

For the automatic generation of categorical rules, the "*generate_categorical_rules()*" function, inherited from the classifier's model, is employed.

This function receives a dataframe with labels from popular clustering methods, which contain categorical values (the cluster to which each instance belongs to). This approach is akin to integrating "expert information", leading to the creation of automatic rules. These rules enhance the classifier's performance, enabling it to yield more valid clusters.

This process not only leverages the intrinsic categorization from the clustering labels but also enriches the classifier's ability to discern and apply nuanced distinctions within the data.

4.2.4 Classifier Fitting

This is one of the most crucial parts of the entire algorithm as it handles the training or fitting of the DSClassifier model.

In this section, we utilize the *"train_test_split'* function from the scikit-learn library in Python, as its shown in the Listing 4.5. This function is typically used to divide the data into two subsets: one for training and the other for testing. In our case, we use the training data to carry out the training of the classifier, and then all the data is used to perform the clustering.

The function receives a dataframe containing all the user-provided data, along with the clusters concatenated from the label selection module discussed in the previous section.

The "*best*" variable is used as the target for real cluster values, which could be the best labels according to silhouette scores or the most repeated labels in each data instance, depending on the user's choice when initializing the class.

```
X_train, _, y_train, _ = train_test_split(self.df_with_labels.values,
                                                    self.best,
                                                    test_size=0.6,
                                                    random_state=42)
```

Listing 4.5: This script separates the data between training and testing using a data frame with the initial data and the generated labels, and using as an objective the best selection according to the user

It's important to note that we use a 60% test size. This means 40% of the data is used for training, and 60% is reserved for testing. A larger test size provides a substantial amount of data to evaluate the clustering model and helps avoid overfitting.

Overfitting occurs when a model closely fits the specific patterns of the training data, which can lead to poor performance on new data. By using a majority of the data for testing, the model's ability to generalize and perform on unseen data is rigorously assessed.

This approach is especially beneficial in clustering scenarios where validating a large and diverse set of data points is critical for determining the robustness and effectiveness of the clustering results.

4.2.5 Label and Rule Retrieval

The last and most critical part of the solution pertains to the model's assignment of clusters. This is achieved through a "*predict()*" method embedded within the classifier, which, upon receiving the full range of user-inputted data, deploys the classifier's training to accurately deliver the predicted clusters. This step signifies the completion of the primary phase or the initial goal of this algorithm's development, which is clustering.

```
def predict(self):
        _, _, _ = self.fit()
        self.y_pred = super().predict(self.df_with_labels.values)

        return self.y_pred
```

Listing 4.6: This script shows how predicts the cluster labels for given data points

A notable feature of this solution is its interpretability, which is a direct result of the designed interpretable classifier. This classifier offers a set of rules that play a pivotal role in the model's decision-making process when assigning clusters to each data instance. The interpretability of these rules is crucial as it mirrors the demonstrated interpretability in the classification algorithm. By providing a clear and understandable set of guidelines or conditions under which the model operates, users can comprehend why certain decisions are made, thereby enhancing trust and transparency in the model's functioning.

```
find_most_important_rules(self, classes=None, threshold=0.2)
print_most_important_rules(self, classes=None, threshold=0.2):
```

Listing 4.7: This script shows the two initializations for the functions that deliver and display rules

For a deeper dive into the specific functions, "*print_most_important_rules()*" displays the rules with the highest significance for the current model to the user. On the other hand, "*find_most_important_rules()*" extracts the most critical rules, which are fundamental in defining the model's behavior.

Moreover, the model allows for explicit explanations for particular data instances. This functionality permits a detailed analysis of the rules applied to each instance of data provided by the user, thus offering greater insight into how each cluster was assigned to each data point.

```
def predict_explain(self, x):
        pred, cls, rls, builder = super().predict_explain(x)
        builder = builder.replace("Class", "Cluster")
        rls = rls[~rls['rule'].str.contains('K-Means Labels|DBSCAN Labels|' +
                                            'Agglomerative Labels',
                                            case=False, regex=True)]
        return pred, cls, rls, builder
```

Listing 4.8: This script shows how it's generated the explanation for an instance of the data

In the broader scope of clustering evaluation, the "*metrics()*" method plays a crucial role. It is designed to retrieve specific metrics based on the clustering results. This function can accept the anticipated real values for the clusters and provide two types of statistics. If real values are incorporated, one can evaluate the precision of the underlying classification model.

```
...
if y is not None:
    y = ClusteringSelector.normalize_labels(self.y_pred, y)
    print("Information of DSClassifier")
    print("\nAccuracy: %.1f%%" % (accuracy_score(y, self.y_pred) * 100.))
    print("F1 Macro: %.3f" % (f1_score(y, self.y_pred, average="macro")))
    print("F1 Micro: %.3f" % (f1_score(y, self.y_pred, average="micro")))
    print("\nConfusion Matrix:")
    print(confusion_matrix(y, self.y_pred))
    print("-------------------")
    print("Clustering Metrics")
    rand_index = adjusted_rand_score(y, self.y_pred)
    pearson_corr, _ = pearsonr(self.y_pred, y)
    print("Rand_index: ", rand_index)
    print("Pearson: ", pearson_corr)
print("-----------------------------------------------")
print("Silhoutte:" , (silhouette_score(self.data, self.y_pred)
                      if len(set(self.y_pred)) > 2 else 0))
```

Listing 4.9: This snippet displays the statistics or metrics for both the base classifier and the clustering algorithm, as applicable.

The metrics for clustering evaluation include the Rand index and the Pearson correlation coefficient, which require real values for each cluster to compute. These metrics are significant as they offer a quantitative measure of the clustering performance. The Rand index measures the extent to which the clustering correctly identifies pairs of items as being in the same or different clusters, while the Pearson correlation coefficient assesses the linear correlation between the predicted and actual cluster assignments.

Independent of whether actual cluster values are provided, the silhouette score is also

given by this function. This score is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette score provides a concise and effective way to evaluate the quality of the clustering, as it encapsulates both the tightness of the clustering and the separation between different clusters. This holistic approach to evaluating clustering results makes the algorithm versatile and robust in various data scenarios.

Finally, a crucial aspect to consider is that when displaying the rules or explanations for each data instance, the categorical rules generated from the information provided by the labels created by popular clustering techniques are not shown. This is because these rules are used solely as an information base to enable the classification model to more accurately predict which cluster will be assigned to each piece of data.

Chapter 5

Experiments and results

To validate the utility of the new clustering algorithm, various tests were conducted to evaluate typical clustering metrics. Additionally, its results were compared with those of standard algorithms commonly used for this task.

5.1 Explanation of Experiments and Data Used

For conducting the experiments, different control datasets were created, which are described as follows:

1. Uniform: Represents a straight line with y=0, where data points with x<0 belong to cluster 0 and those with x>0 belong to cluster 1.
2. Rectangle Uniform: Consists of 2 rectangles of data contained within -1<x<1, where the first for values of y<-0.15 corresponds to cluster 0 and those with y>0.15 to cluster 1.
3. Gaussian Distribution: Comprises 2 normal distributions centered at [1.2, 0.2] and [0.2, 0.9], with their respective clusters being 0 and 1.
4. Gaussian Mix Distribution: A set of 3 normal distributions centered at [-0.8, 0.6], [0.6, -0.6], and [-1.2, -0.6], with their respective clusters being 0, 1, and 2.

The visualizations of these datasets are provided below.

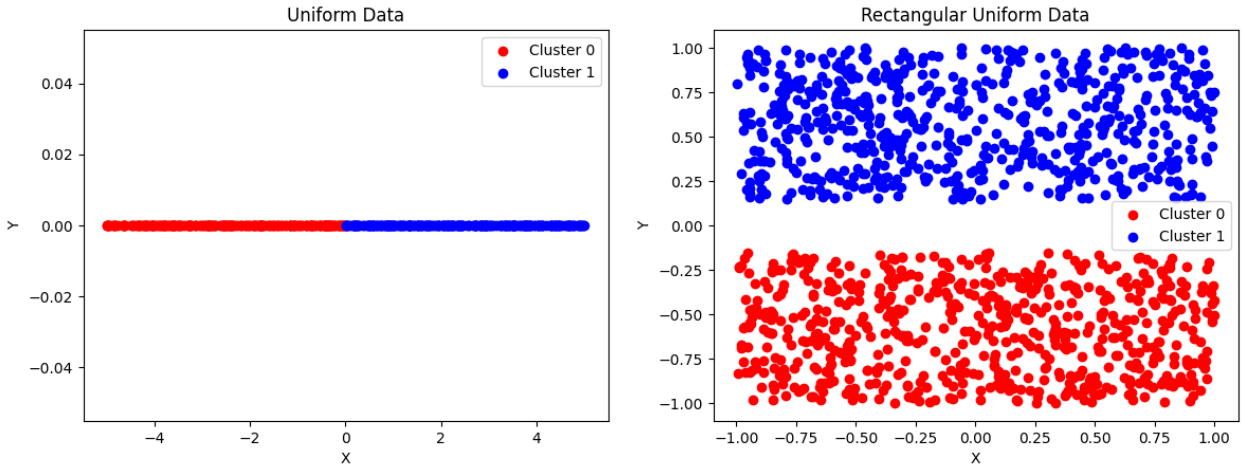

Figure 5.1: Left: Uniform line, Right: Two uniform rectangles

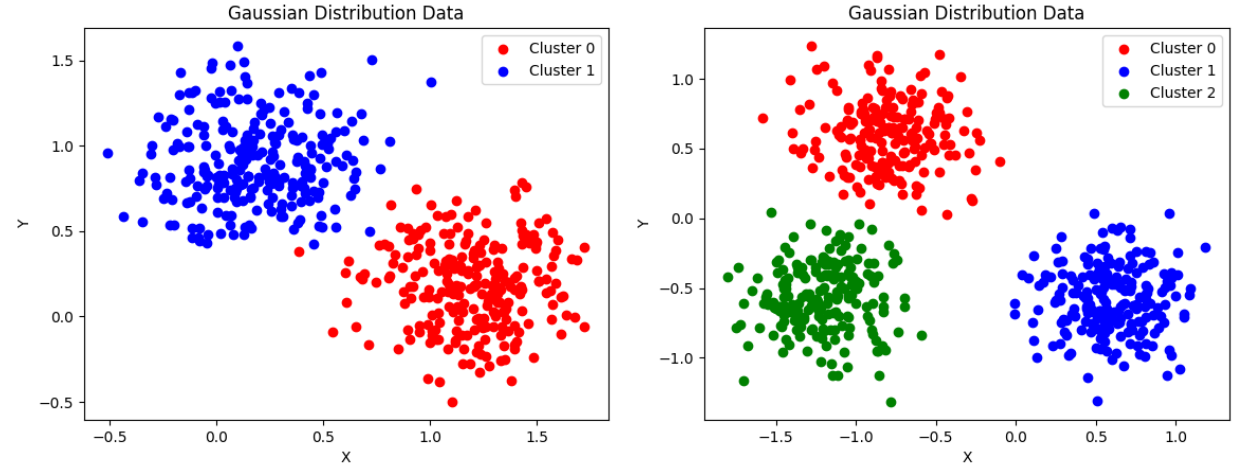

Figure 5.2: Left: Gaussian distribution of two clusters, Right: Gaussian distribution of three clusters

For the first and second sets, `random.uniform` from the NumPy library was used, and for the third and fourth, `random.normal` from the same library was employed.

Furthermore, popular datasets such as:

1. Iris: A classic dataset for pattern recognition, featuring measurements of iris flowers for species classification.
2. Wines: Consists of physicochemical properties of various wine samples, used for quality assessment and classification.
3. Breast Cancer: Contains features computed from digitized images of breast mass, used for cancer diagnosis.

were also utilized in the experiments.

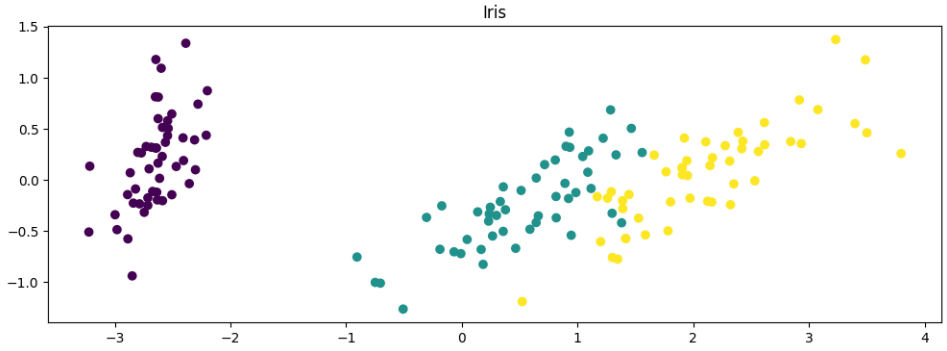

Figure 5.3: Iris dataset clusters

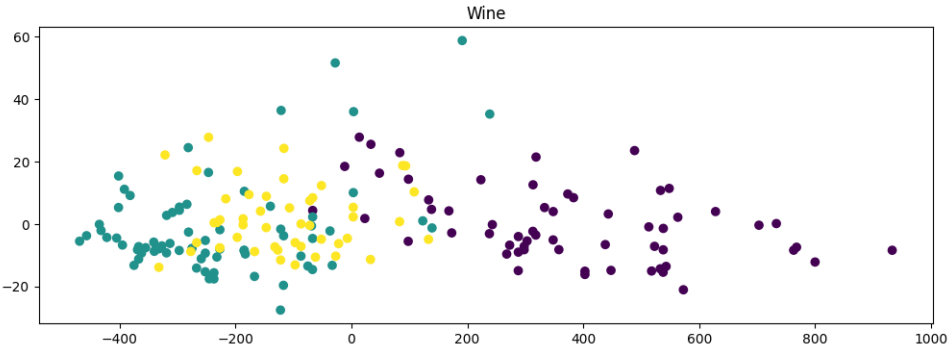

Figure 5.4: Wine dataset Clusters

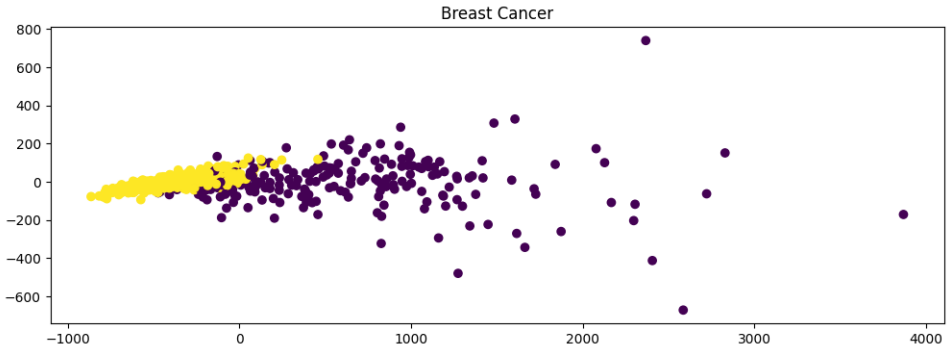

Figure 5.5: Breast Cancer dataset Clusters

And finally, the last datasets to be used, correspond to real datasets for which there is no specific clustering, these are:

1. Customer Behavior
2. Airline Passenger Satisfaction

With these datasets as input, various types of clustering algorithms were applied. Specifically, popular clustering algorithms such as KMeans, DBSCAN, and Agglomerative Clustering were utilized. Additionally, the newly developed DSclustering algorithm was tested under three potential scenarios in which a user might employ it, namely:

1. *DS clustering: Best Labels:* Utilizing the best-generated clusters.
2. *DS clustering: Most Voted label:* Using the most frequently repeated labels for each instance.
3. *DS clustering: Best Label with number of cluster:* Providing the expected number of clusters.

The results of applying these algorithms will be detailed in the following sections. To conclude the experiments, a comparison will be made among all the applied algorithms, and an analysis will be conducted on the rules generated by the DS clustering across the different datasets.

5.2 Results

5.2.1 Validation of the Clustering

To evaluate the results of the newly designed clustering algorithm, experiments were conducted assessing three important metrics relevant to clustering algorithms: the silhouette index, Pearson correlation coefficient, and Rand index. These results were compared with three popular clustering algorithms and across three potential usage scenarios of DS clustering for users. This aims to answer the following questions:

- How does the clustering algorithm perform in terms of the validity of its results?
- Do the results of this algorithm compare favorably with those of popular classes for certain data types?

One of the main hypotheses that emerge is that the DS clustering algorithm should yield results similar to those of popular clustering algorithms. This is because it uses these algorithms as a foundation, particularly their outcomes, as expert experience to feed the necessary rules for predicting the labels.

The results presented in Table 5.1 suggest that the DS clustering algorithm is expected to yield results similar to those of popular clustering algorithms.

Dataset	method	Rand Index	Silhouette	Pearson
Uniform line	KMeans	0.968192	0.6092	-0.984111
	DBSCAN	0.0	0.0	0.0
	Agglomerative	0.626526	0.6008	-0.807947
	DSC Best Labels	0.444714	0.574601	0.478302
	DSC Most Voted label	0.252542	0.323917	0.679125
	DSC with number of cluster	0.984032	0.6090	0.992022
Uniform Rectangle	KMeans	1.0	0.4396	1.0
	DBSCAN	0.0	0.0	0.0
	Agglomerative	1.0	0.4396	1.0
	DSC Best Labels	0.503001	0.495606	-0.065668
	DSC Most Voted label	0.503082	0.495662	-0.064969
	DSC with number of cluster	1.0	0.43963	-1.0
Gaussian Distribution	KMeans	0.984032	0.6435	-0.992000
	DBSCAN	0.0	0.0	0.0
	Agglomerative	0.984032	0.64351	-0.992000
	DSC Best Labels	0.968192	0.4653	0.984031
	DSC Most Voted label	0.948585	0.475981	0.961466
	DSC with number of cluster	0.984032	0.64352	0.992000
Gaussian mix Distribution	KMeans	1.0	0.684170	-0.5
	DBSCAN	0.570609	0.6048	9.11e-18
	Agglomerative	1.0	0.684170	0.5
	DSC Best Labels	0.869085	0.558766	0.219600
	DSC Most Voted label	0.869085	0.558766	0.219600
	DSC with number of cluster	0.570609	0.6048	9.11e-18
Iris	KMeans	0.730238	0.552819	0.224350
	DBSCAN	0.520619	0.486034	0.367712
	Agglomerative	0.731199	0.554324	0.205441
	DSC Best Labels	0.546422	0.509248	0.750098
	DSC Most Voted label	0.546422	0.509248	0.750098
	DSC with number of cluster	0.695636	0.478250	0.126230
Wine	KMeans	0.371114	0.571138	-0.029116
	DBSCAN	0.0	0.0	0.0
	Agglomerative	0.368402	0.564480	0.572525
	DSC Best Labels	0.313358	0.370674	-0.226526
	DSC Most Voted label	0.276761	0.324729	-0.252627
	DSC with number of cluster	0.304176	0.584262	-0.673827
Breast Cancer	KMeans	0.519788	0.6972	-0.697773
	DBSCAN	0.0	0.0	0.0
	Agglomerative	0.390288	0.6899	0.742352
	DSC Best Labels	0.0	0.0	0.0
	DSC Most Voted label	0.0	0.0	0.0
	DSC with number of cluster	0.0	0.0	0.0

Table 5.1: Comparison of metrics for different clustering methods on various datasets

Observations from the majority of the datasets in these experiments show that the results of the designed class algorithm are quite close to those of other algorithms. For example, specifically looking at the results for uniform rectangles, DS clustering for best labels and most voted label performs better in terms of the silhouette measure than popular clustering methods.

However, for other datasets, the results are lower in terms of various metrics, which is expected. The performance of class algorithms can vary depending on the types of data used.

For example, the wine dataset, although rich in chemical composition data, may be inherently unsuitable for clustering due to the complexity and subtle variances in its characteristics, characterised by measurements that are closely related to the intrinsic qualities of wine, which may not have clear boundaries or groupings. In addition, the overlap in chemical characteristics between different types of wine can blur the distinctions needed to form well-defined clusters. The high dimensionality of the data can also obscure underlying patterns, making it difficult for clustering algorithms to identify clear separations. As a result, the structure of the dataset may not lend itself to clean clustering, leading to unsatisfactory and ambiguous grouping results.

So K-means, DBSCAN and Agglomerative Clustering may give poor results on the Wine dataset for several reasons. Firstly, K-means assumes clusters to be approximately equal in size and spherical in shape, which may not match the distribution of the Wine dataset, which has inherently non-spherical and unevenly sized clusters. Secondly, DBSCAN relies on a density-based approach, which can struggle with high-dimensional data; the Wine dataset, with its complex and high-dimensional chemical composition data, may not have well-defined density clusters, making parameter tuning difficult. Thirdly, agglomerative clustering, a hierarchical method, can also stumble if the data contains noise and outliers, which are common in real-world datasets like Wine, where outliers can significantly distort the hierarchy of cluster formation. As our model is based on K-means, DBSCAN and Agglomerative Clustering, it inherits their limitations.

One of the reasons for "bad results" when using best labels, for example for "Breast Cancer dataset" is that DBSCAN can in some cases distort the silhouette score measurements. In the absence of additional information, the generation of a single cluster can be considered a good result, even if it is not the expected one. It is important to remember that clustering is an exploratory tool and the validity of any cluster found depends on the individual applying it.

The silhouette score, which measures how similar an object is to its own cluster compared to other clusters, can be misleading in such scenarios. If DBSCAN produces a single cluster or assigns many points as outliers, the silhouette score may not reflect a nuanced understanding of the structure of the data. This is particularly true when no external information is available to guide the interpretation of the results.

Furthermore, the interpretative nature of clustering underlines the subjective aspect of analysing the results. What constitutes a "good" clustering result can vary significantly between different users or applications. Some may value the detailed partitioning of data into many small clusters, while others may prefer a broader overview with fewer, larger

clusters. Therefore, the effectiveness and appropriateness of using best labels or relying heavily on metrics such as the silhouette score should be evaluated in the context of the specific goals and knowledge of the dataset at hand. So automated metrics and labels can provide valuable insights, they should not replace critical analysis and domain expertise when evaluating clustering results. The exploratory nature of clustering means that its success is determined not only by algorithmic outputs, but also by how well the results align with the user's objectives and understanding of the data.

With these results, we can try to answer the first questions of this subsection. The metrics obtained for some of the clusterings carried out are very valid and similar to other clustering methods, especially those carried out with "DS clustering: with number of clusters". It's important to point out that other methods can, in certain cases, perform quite similarly to popular clusterings, in terms of silhouette score, they are quite close. It's worth remembering that the main focus of this development is to achieve an algorithm that can be similar in validity to popular clusterings, but much more interpretable, which is what we'll be looking at in the next section.

In the context of the comparison, it is worth noting that DS clustering shows versatile performance on different datasets. This means that, like all other existing clustering algorithms, it can achieve good results and generate valid clusters if the dataset supports good clustering. In particular, its ability to perform well on silhouette scores indicates its effectiveness in identifying well-separated clusters, a crucial aspect in many practical applications.

5.2.2 Interpretability

To evaluate the interpretability gained through the use of rules provided by the classification model, three datasets will be presented. In each, DS Clustering is applied with a different type of usage (Best Label, Most Voted, and giving the number of clusters).

Uniform Rectangles

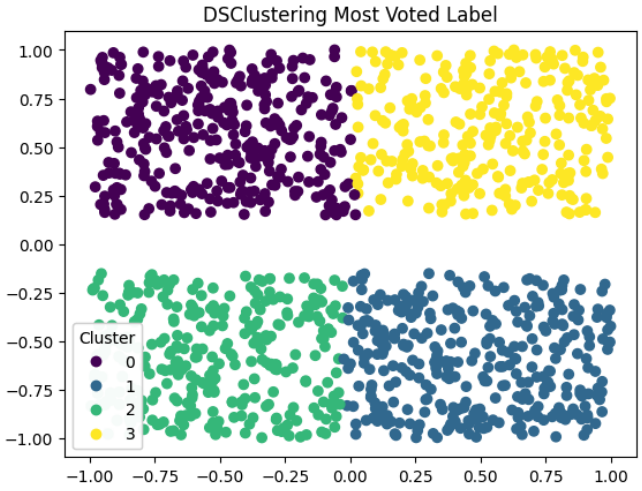

Figure 5.6: Clusters of Uniform Rectangles

Cluster	Rule	Importance	0	1	2	3	Unc
0	R16: 0.011 < y < 0.440	0.537	0.406	0.000	0.000	0.305	0.289
	R10: x < -0.410	0.436	0.291	0.000	0.364	0.000	0.345
	R28: Negative x - -0.025, y - 0.011	0.409	0.256	0.396	0.000	0.000	0.347
	R11: -0.410 < x < -0.025	0.400	0.264	0.000	0.343	0.000	0.394
	R17: y > 0.440	0.379	0.229	0.000	0.000	0.399	0.372
1	R28: Negative x - -0.025, y - 0.011	0.509	0.256	0.396	0.000	0.000	0.347
	R12: -0.025 < x < 0.360	0.420	0.091	0.248	0.000	0.375	0.286
	R13: x > 0.360	0.419	0.000	0.284	0.000	0.333	0.383
	R14: y < -0.417	0.410	0.000	0.290	0.290	0.000	0.421
	R15: -0.417 < y < 0.011	0.401	0.000	0.282	0.287	0.000	0.431
2	R10: x < -0.410	0.488	0.291	0.000	0.364	0.000	0.345
	R11: -0.410 < x < -0.025	0.456	0.264	0.000	0.343	0.000	0.394
	R14: y < -0.417	0.410	0.000	0.290	0.290	0.000	0.421
	R15: -0.417 < y < 0.011	0.404	0.000	0.282	0.287	0.000	0.431
	R27: Positive x - -0.025, y - 0.011	0.375	0.000	0.000	0.257	0.291	0.452
3	R12: -0.025 < x < 0.360	0.517	0.091	0.248	0.000	0.375	0.286
	R17: y > 0.440	0.500	0.229	0.000	0.000	0.399	0.372
	R16: 0.011 < y < 0.440	0.466	0.406	0.000	0.000	0.305	0.289
	R13: x > 0.360	0.454	0.000	0.284	0.000	0.333	0.383
	R27: Positive x - -0.025, y - 0.011	0.400	0.000	0.000	0.257	0.291	0.452

Table 5.2: Most Important Rules for Each Cluster of Uniform Rectangles

Observing the points in Figure 5.6 and comparing them with the rules in Table 5.2, we can see how the rules effectively separate the data for each cluster identified by the algorithm. This makes it easy to interpret that if a value falls within a certain range of X, it can be

readily assigned to a specific cluster efficiently. The key is to choose the cluster with the highest value in the table. This value represents a higher "probability" of belonging to that cluster.

An additional benefit of using this classifier is that it also provides the uncertainty of whether a certain value within the range defined by the presented rule belongs to one of the aforementioned clusters. Thus, the decision-making process of the cluster is highly detailed and transparent, making it easy for the user to identify which cluster to assign a specific data instance. For example, if we take a value located at 0.5 on the X-axis and 0.2 on the Y-axis, the given rules indicate that it belongs to cluster number 3, as its Y value is greater than 0.44 (rule number 17) and its X value is greater than 0.360 (rule number 13).

Gaussian Mix Distribution

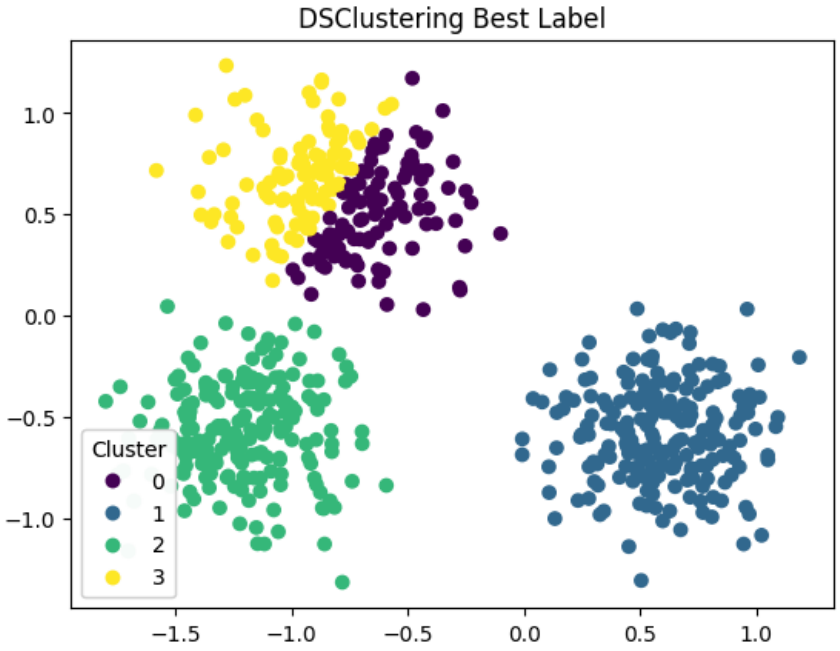

Figure 5.7: Clusters of Gaussian Mix Distribution

Cluster	Rule	Score	0	1	2	3	Unc
0	R16: y > 0.187	0.545	0.360	0.000	0.000	0.465	0.175
	R10: -1.010 < x < -0.468	0.532	0.321	0.000	0.373	0.186	0.119
	R27: Negative x - -0.468, y - -0.213	0.385	0.240	0.163	0.000	0.213	0.384
	R11: -0.468 < x < 0.074	0.348	0.348	0.000	0.000	0.000	0.652
	R15: -0.213 < y < 0.187	0.279	0.085	0.406	0.425	0.000	0.083
1	R15: -0.213 < y < 0.187	0.610	0.085	0.406	0.425	0.000	0.083
	R13: y < -0.612	0.516	0.000	0.365	0.362	0.000	0.272
	R14: -0.612 < y < -0.213	0.504	0.000	0.363	0.338	0.000	0.299
	R12: x > 0.074	0.402	0.000	0.402	0.000	0.000	0.598
	R27: Negative x - -0.468, y - -0.213	0.317	0.240	0.163	0.000	0.213	0.384
2	R15: -0.213 < y < 0.187	0.625	0.085	0.406	0.425	0.000	0.083
	R10: -1.010 < x < -0.468	0.573	0.321	0.000	0.373	0.186	0.119
	R13: y < -0.612	0.513	0.000	0.365	0.362	0.000	0.272
	R9: x < -1.010	0.506	0.000	0.000	0.358	0.357	0.285
	R14: -0.612 < y < -0.213	0.487	0.000	0.363	0.338	0.000	0.299
	R26: Positive x - -0.468, y - -0.213	0.267	0.043	0.063	0.220	0.000	0.675
3	R16: y > 0.187	0.620	0.360	0.000	0.000	0.465	0.175
	R9: x < -1.010	0.505	0.000	0.000	0.358	0.357	0.285
	R10: -1.010 < x < -0.468	0.405	0.321	0.000	0.373	0.186	0.119
	R27: Negative x - -0.468, y - -0.213	0.363	0.240	0.163	0.000	0.213	0.384

Table 5.3: Most Important Rules for Each Cluster of Gaussian Mix Distribution

Unlike the previous results, the following experiment considers data that are not separated by a specific straight line but rather form a mass or a density of data to which the clustering algorithm is applied. We can observe how the algorithm is still capable of detecting the existence of clusters, separating them, and generating rules to assign a cluster to each data instance.

The importance of this experiment is to highlight that the rules can become more complex. In this case, we can see in Table 5.3 how rule number 27 includes two parameters, to which a range is applied or a parameter is verified. This is necessary because, as we recently mentioned, the data are no longer delimited by straight lines. It's essential to establish a more consistent and robust rule when assigning a cluster. This gives us indications that the model works effectively, even as we introduce more complex data.

Wine Dataset

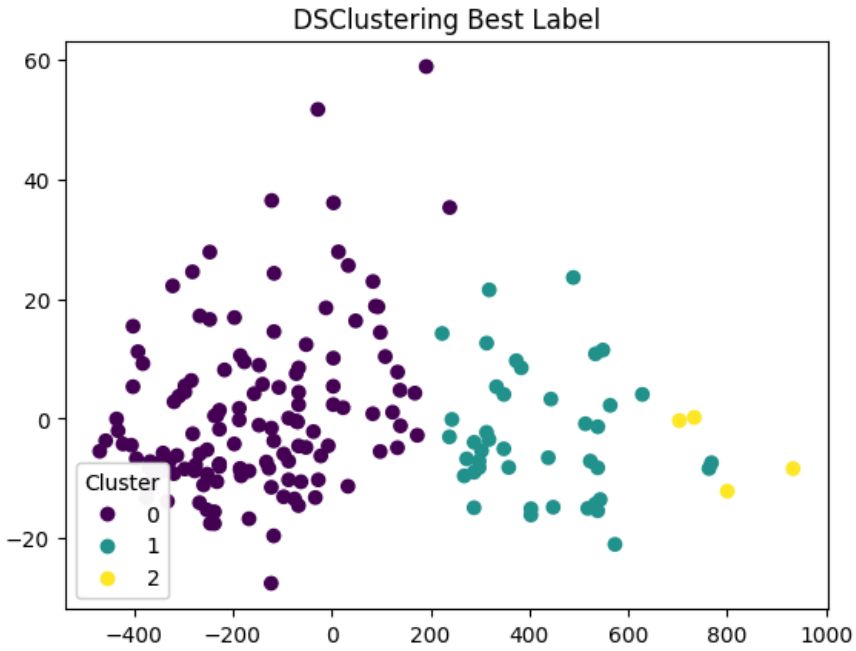

Figure 5.8: Clusters of Wine Dataset

Lastly, this experiment involves a dataset that is not synthetically generated for this process and generally does not yield as good results when applying clustering algorithms. It is noteworthy that the rules also consider values for any feature, how its shown in Table .1, not just for the X and Y axes.

An important aspect to highlight is that the validity of the clusters generated by this algorithm can still be improved. This means enhancing the metrics evaluated in Table 1, aiming not only to improve the obtained labels but also the generated rules. This approach would lead to a more robust and interpretable clustering algorithm.

5.3 Real World Datasets

5.3.1 Consumer Behavior Dataset

This subsection delves into a detailed experiment using a complete data set from Kaggle[1]. The data set, which reflects real-world purchasing habits, provides fertile ground to apply and compare various clustering techniques. This could be useful in the real world to discover different groups or patterns in consumer behavior, which can provide valuable information for market segmentation, targeted marketing, and understanding customer preferences. The

[1] `https://www.kaggle.com/datasets/zeesolver/consumer-behavior-and-shopping-habits-dataset/?select=shopping_behavior_updated.csv`

analysis will use, like the previous part, both traditional clustering methods and a novel approach, thus contributing to a broader understanding of the results of the new proposed clustering algorithm.

Data Preprocessing

The preprocessing stage is crucial in shaping the dataset for effective clustering analysis.

We removed columns such as

- *Customer ID*
- *Item Purchased*
- *Review Rating*
- *Subscription Status*
- *Discount Applied*
- *Promo Code Used*
- *Location*
- *Size*

These columns were deemed less relevant for clustering purposes due to their individual-specific nature or lack of aggregatable information.

To accommodate clustering algorithms, we applied *Label Encoding*[2], transforming categorical variables into a numerical format. This step was essential to ensure compatibility with the clustering algorithms, which predominantly operate on numerical data, thus setting the stage for a more nuanced and structured analysis.

Analysis and Results

The comparative analysis of the clustering algorithms focused primarily on their performance as indicated by the Silhouette Score. The scores are those shown in table 5.4.

The KMeans and DSClustering algorithm with a predefined number of clusters showed the highest silhouette scores, indicating strong cluster formation with well-defined separations. In contrast, DBSCAN's zero score suggests an inability to form meaningful clusters, possibly due to its sensitivity to parameter settings and data density. The DSClustering Best Label and Agglomerative Clustering scores, while lower than those of KMeans, still demonstrate reasonable cluster formation.

[2] `https://www.geeksforgeeks.org/ml-label-encoding-of-datasets-in-python/`

The remarkable performance of DSClustering with a number of clusters, achieving a Silhouette Score close to that of the widely recognized KMeans method, stands out not only for synthetic datasets but also in real-world scenarios. This proximity in scoring suggests once again that DSClustering is capable of generating clusters as valid and distinct as those produced by an established algorithm like KMeans. Furthermore, it indicates the robustness and effectiveness of DSClustering in handling complex, real-world data without predefined data labels or class categories. In the context of a real dataset, where predefined categorizations and the true grouping are unknown, the fact that DSClustering achieves results comparable to KMeans is a promising indication of its potential in practical applications such as consumer behavior analysis and market segmentation.

Dataset	Method	Silhouette
Consumer Behavior	KMeans	0.26837
	DBSCAN	0.0
	Agglomerative	0.24299
	DSC Best Labels	0.22165
	DSC Most Voted label	0.12827
	DSC with number of cluster	0.26830

Table 5.4: Comparison of metrics for different clustering methods on Consumer Behavior dataset

Next, we present the graphs generated by the application of the six clustering models analyzed. It's important to emphasize that in these visualizations, the 'Age' feature is used on the x-axis, while the y-axis represents a merged representation of the remaining data features. This approach was adopted to retain a logical interpretation of the groups formed, allowing us to observe how age correlates with other combined features. At the same time, this method enables us to condense the entire dataset into a comprehensible and visually interpretable format. Such a representation is crucial for understanding the clustering behavior and for drawing meaningful insights from the complex, multidimensional nature of the data.

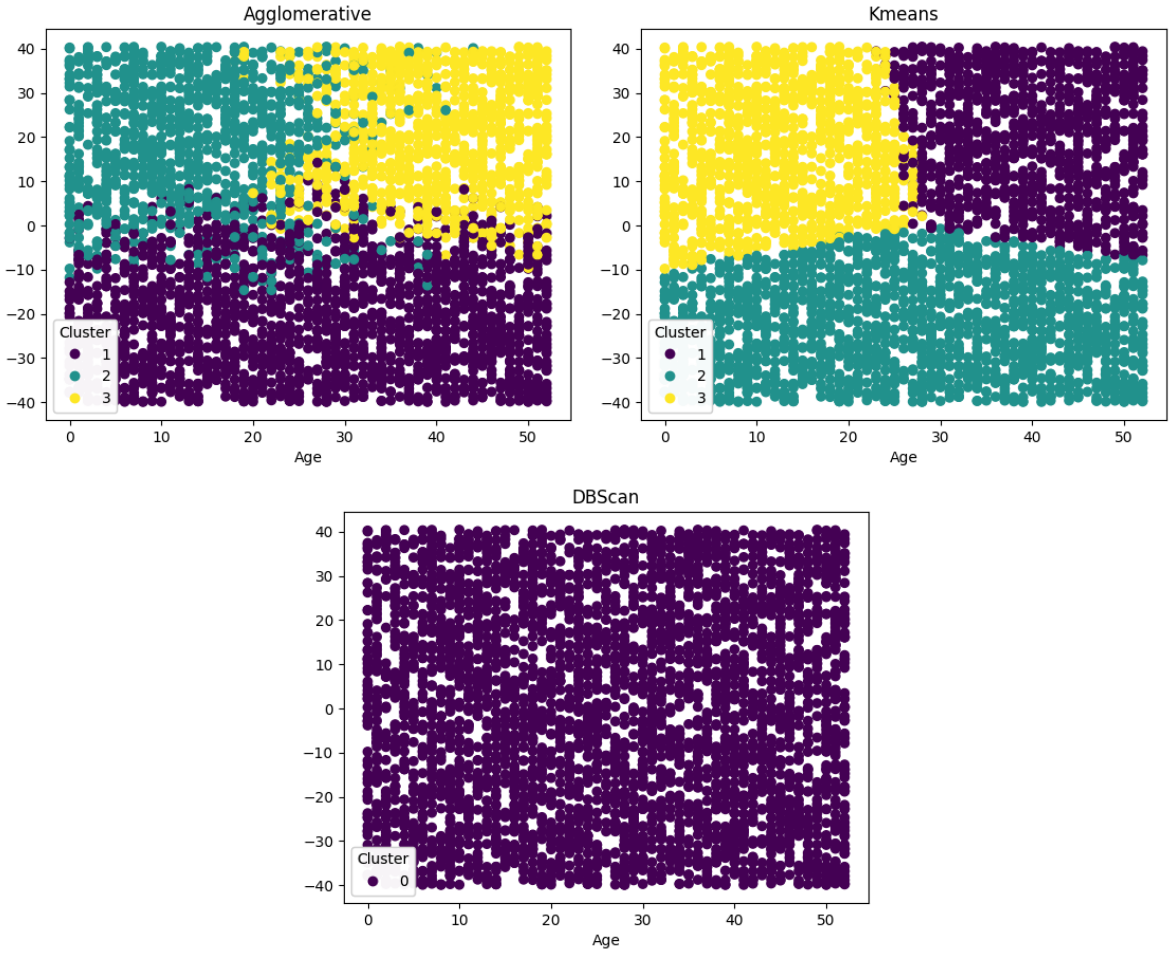

Figure 5.9: Left: Agglomerative Clustering, Right: Kmeans clustering, Below: DBScan Clustering

In the figure 5.9, we observe the clustering results from the popular methods, where both Agglomerative Clustering and KMeans reveal a fairly clear structure of three clusters. The delineation of these clusters is particularly more defined in the case of KMeans, indicating its effectiveness in segregating distinct groups within the dataset. However, for DBSCAN, due to the high density of the data, it fails to achieve a clear separation of clusters. This outcome highlights DBSCAN's sensitivity to data density and its limitations in distinguishing separate groups under such conditions. The contrast in these results underscores the importance of selecting an appropriate clustering algorithm that aligns with the specific characteristics and density of the dataset under analysis.

In Figure 5.10, we can observe the cluster formations as executed by the three models of the DSclustering algorithm. Across all three models, there are quite distinct patterns, but a structure of three clusters predominates, especially in the model where the number of clusters is predefined. This particular model was also the one we observed to have the best Silhouette Score result.

This graphical and empirical evidence showcases the usability of the DSclustering algorithm for clustering tasks in datasets that are neither controlled nor synthetic. The fact

that the DSclustering algorithm, especially the variant with a predefined number of clusters, closely mirrors the cluster structures identified by more established methods like KMeans, reinforces its validity. It demonstrates the algorithm's capability to discern meaningful patterns in real-world data, confirming its potential as a valuable tool for clustering in practical, complex data environments

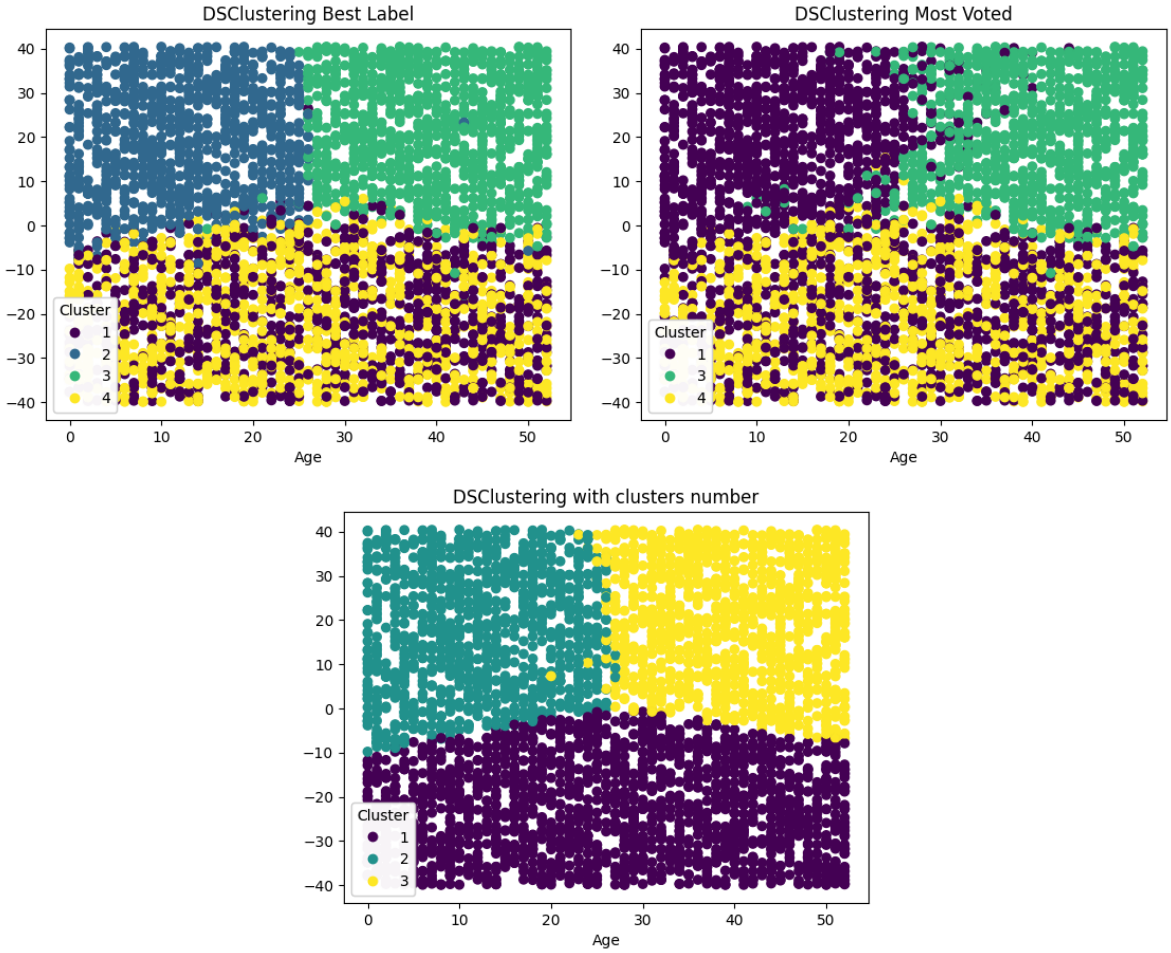

Figure 5.10: Left: DSClustering with Best Labels, Right: DSClustering with Most Voted Labels, Below: DSClustering with number of clusters

The analysis reveals critical insights into the effectiveness of different clustering methodologies applied to consumer behavior data. The superior performance of KMeans and DSClustering with a predefined number of clusters underscores their suitability for datasets with distinct, separable groups. The underperformance of DBSCAN highlights its limitations in handling diverse data densities and structures, as commonly found in consumer datasets. The moderate success of DSClustering Best Label and Agglomerative Clustering suggests their potential in specific contexts, though they may not be universally applicable.

DS clustering Rules

The key rules generated by the three models of DSClustering, which provide interpretability to the model, are presented in table 5.5, 5.6 y 5.7. It's crucial to remember that this aspect of interpretability was demonstrated by Sergio Peñafiel in his thesis and is now being applied in this analysis. The interpretability offered by these rules is a significant advancement in understanding the clustering process and its outcomes.

For instance, in the model where the number of clusters is predefined, the rules enable us to comprehend the clustering logic more clearly. It's important to note, however, that for the visualizations, the features are combined, which means that directly correlating these rules with the graphical representation is not straightforward. Nonetheless, we can observe the separation between ages for clusters 2 and 3, as indicated by the rules. This separation aligns with the visual clusters in the graphs, affirming the relevance of the rules in guiding our understanding of how the algorithm segments the data.

The ability of DSClustering to generate these interpretable rules is a testament to its utility in practical clustering applications. It not only facilitates the formation of distinct clusters but also provides a clear rationale behind each grouping. This dual capability of clustering and interpretability, as illustrated in Sergio Peñafiel's thesis, enhances the practical value of DSClustering in real-world data analysis, bridging the gap between complex algorithmic processes and actionable insights.

As we already seen, rules generated in the context of clustering analysis serve as guidelines for determining the cluster assignment for a specific data instance. They operate by evaluating each rule and considering, in this case, the probability that the instance belongs to a particular cluster or the uncertainty of it not belonging to that cluster. This probabilistic and uncertainty-based approach allows for a nuanced understanding of how each data point fits within the overall clustering structure.

N° C	Rule	Score	C 1	C 2	C 3	C 4	Unc
1	Purchase Amount (USD) < 23.892	0.675	0.474	0.000	0.000	0.487	0.039
	23.892 < Purchase Amount (USD) < 39.685	0.653	0.537	0.025	0.000	0.232	0.206
	15.940 < Age < 26.107	0.632	0.465	0.252	0.000	0.139	0.143
	Previous Purchases > 34.214	0.550	0.548	0.000	0.004	0.000	0.448
	26.107 < Age < 36.274	0.546	0.298	0.000	0.279	0.423	0.000
2	Purchase Amount (USD) > 55.477	0.676	0.000	0.467	0.512	0.000	0.021
	Age < 15.940	0.638	0.049	0.578	0.000	0.077	0.296
	24.450 < Previous Purchases < 34.214	0.595	0.150	0.400	0.331	0.003	0.116
	Shipping Type = 5	0.588	0.000	0.464	0.094	0.188	0.254
	39.685 < Purchase Amount (USD) < 55.477	0.529	0.010	0.351	0.437	0.000	0.203
3	Purchase Amount (USD) > 55.477	0.708	0.000	0.467	0.512	0.000	0.021
	14.686 < Previous Purchases < 24.450	0.684	0.000	0.233	0.503	0.197	0.068
	Category = 3	0.592	0.000	0.141	0.521	0.011	0.327
	39.685 < Purchase Amount (USD) < 55.477	0.590	0.010	0.351	0.437	0.000	0.203
	Age > 36.274	0.570	0.009	0.000	0.565	0.000	0.426
4	Previous Purchases < 14.686	0.720	0.000	0.117	0.212	0.574	0.097
	Purchase Amount (USD) < 23.892	0.684	0.474	0.000	0.000	0.487	0.039
	26.107 < Age < 36.274	0.650	0.298	0.000	0.279	0.423	0.000
	Color > 16.722	0.447	0.000	0.192	0.047	0.343	0.418
	Negative Previous Purchases - 24.450, Frequency of Purchases - 2.935	0.438	0.158	0.006	0.000	0.364	0.473

Table 5.5: Most Important Rules for Each Cluster Using DSClustering Best Labels

N° C	Rule	Score	C 1	C 2	C 3	C 4	Unc
1	Age < 15.940	0.551	0.543	0.000	0.000	0.016	0.441
	Previous Purchases > 34.214	0.499	0.495	0.006	0.001	0.000	0.498
	Purchase Amount (USD) > 55.477	0.433	0.256	0.041	0.435	0.000	0.268
	15.940 < Age < 26.107	0.359	0.227	0.039	0.014	0.288	0.431
	Negative Purchase Amount (USD) - 39.685, Previous Purchases - 24.450	0.338	0.332	0.013	0.000	0.000	0.655
3	Age > 36.274	0.687	0.000	0.048	0.611	0.113	0.228
	39.685 < Purchase Amount (USD) < 55.477	0.623	0.092	0.033	0.563	0.000	0.312
	14.686 < Previous Purchases < 24.450	0.573	0.000	0.018	0.435	0.301	0.246
	Purchase Amount (USD) > 55.477	0.564	0.256	0.041	0.435	0.000	0.268
	Positive Age - 26.107, Purchase Amount (USD) - 39.685	0.420	0.000	0.055	0.316	0.187	0.442
4	Previous Purchases < 14.686	0.686	0.000	0.030	0.201	0.580	0.189
	Purchase Amount (USD) < 23.892	0.632	0.096	0.000	0.000	0.585	0.319
	26.107 < Age < 36.274	0.505	0.058	0.000	0.085	0.439	0.418
	14.686 < Previous Purchases < 24.450	0.476	0.000	0.018	0.435	0.301	0.246
	Positive Purchase Amount (USD) - 39.685, Previous Purchases - 24.450	0.438	0.000	0.001	0.148	0.370	0.481

Table 5.6: Most Important Rules for Each Cluster Using DSClustering Most Voted Labels

As illustrated in Tables 5.5 and 5.6, each rule is accompanied by probabilities and uncertainties for cluster membership. This information is critical in deciphering the logic behind the cluster assignments. However, our focus will be primarily on analyzing Table 5.7, which presents the results of DSClustering with a predefined number of clusters. This particular model has demonstrated the best performance in terms of the Silhouette Score. Likewise, it is important to highlight how in table 5.6, there are no rules for cluster 2 and this is also reflected in the graph, since there is no point for these data instances that belong to that cluster.

The higher Silhouette Score of this model indicates that it has been more effective in distinguishing between clusters, creating more defined and well-separated groupings within the dataset.

The correspondence between the rule "26.107 < Age" and the division observed in clusters 2 and 3 in the graph is a significant aspect of our analysis. This observation demonstrates how rules generated by the DSClustering model can reflect discernible patterns within the visualized data.

In the figure 5.10, if we notice that the separation between clusters 2 and 3 indeed occurs around the age of 26.107 years, this validates the mentioned rule. This coincidence suggests that age is a determining factor in differentiating these two clusters. In other words, the rule "26.107 < Age" appears to be a good indicator for distinguishing between the groups represented by clusters 2 and 3.

N°C	Rule	Score	C 0	C 1	C 2	Unc
1	Purchase Amount (USD) < 23.892	0.503	0.503	0.000	0.000	0.497
	Age < 15.940	0.460	0.253	0.587	0.000	0.161
	23.892 < Purchase Amount (USD) < 39.685	0.453	0.453	0.000	0.000	0.547
	Payment Method = 0	0.278	0.159	0.000	0.326	0.514
	26.107 < Age < 36.274	0.252	0.111	0.000	0.464	0.425
2	39.685 < Purchase Amount (USD) < 55.477	0.772	0.000	0.597	0.401	0.002
	Age < 15.940	0.702	0.253	0.587	0.000	0.161
	Purchase Amount (USD) > 55.477	0.607	0.000	0.370	0.626	0.004
	15.940 < Age < 26.107	0.494	0.100	0.444	0.008	0.449
	Negative Age - 26.107, Purchase Amount (USD) - 39.685	0.404	0.086	0.363	0.000	0.551
3	Purchase Amount (USD) > 55.477	0.790	0.000	0.370	0.626	0.004
	39.685 < Purchase Amount (USD) < 55.477	0.633	0.000	0.597	0.401	0.002
	Age > 36.274	0.597	0.043	0.000	0.576	0.381
	26.107 < Age < 36.274	0.517	0.111	0.000	0.464	0.425
	Payment Method = 0	0.398	0.159	0.000	0.326	0.514

Table 5.7: Most Important Rules for Each Cluster Using DSClustering with number of cluster

The practical implication of this observation is twofold. On one hand, it underlines the usefulness of the rules generated by the DSClustering model in interpreting and validating the results of the clustering analysis. On the other hand, it highlights the importance of age as a distinctive factor in consumer behavior represented in the dataset.

In summary, the alignment between the generated rule and the distribution observed in the graph not only reinforces confidence in the interpretability of the clustering model but also provides valuable insights into the underlying characteristics that define consumer behavior patterns in the analyzed dataset.

5.3.2 Airline Passenger Satisfaction

This subsection delves into a detailed experiment using a complete data set from Kaggle[3]. The data set, This dataset contains an airline passenger satisfaction survey, provides another fertile ground to apply and compare clustering models. This can be used to identify patterns among the groups that are either satisfied or dissatisfied with traveling.

[3] `https://www.kaggle.com/datasets/teejmahal20/airline-passenger-satisfaction?select=train.csv`

Analysis and Results

The comparative analysis of the clustering algorithms focused primarily on their performance as indicated by the Silhouette Score. The scores are those shown in table 5.8.

The KMeans and DSClustering algorithm with a predefined number of clusters showed the highest silhouette scores, indicating strong cluster formation with well-defined separations. In contrast, DBSCAN's zero score suggests an inability to form meaningful clusters, possibly due to its sensitivity to parameter settings and data density. The DSClustering Best Label and most voted are 0.0, this could be for multiple reasons but primarly beacause DBSCAN's zero score.

This is one of the typical cases encountered in data exploration, where the user must explicitly indicate the number of clusters to achieve better results or performance. In clustering, as already mentioned, this requirement can stem from various factors, such as data density, the correlation among features, and it's an aspect that can be refined through the statistical analysis conducted in tandem with the clustering process.

Delving deeper, the necessity for specifying the number of clusters beforehand highlights the importance of understanding the data's inherent structure before applying clustering algorithms. Data density, for instance, affects how tightly or loosely grouped the data points are, influencing the optimal number of clusters. Similarly, the correlation between features can reveal natural groupings within the data, guiding the selection of an appropriate number of clusters.

Pre-clustering statistical analysis plays a crucial role in this context. Techniques such as principal component analysis (PCA) can reduce dimensionality and highlight the underlying patterns in the data, while correlation matrices can help identify relationships between variables. These analyses provide valuable insights into the data's structure, informing the choice of the number of clusters and enhancing the clustering algorithm's effectiveness.

Dataset	Method	Silhouette
Airline Passenger Satisfaction	KMeans	0.689
	DBSCAN	0.0
	Agglomerative	0.685
	DSC Best Labels	0.0
	DSC Most Voted label	0.0
	DSC with number of cluster	0.688

Table 5.8: Comparison of metrics for different clustering methods on Airline Passenger Satisfaction dataset

Next, we present the plots generated by applying the six clustering models analysed. It is important to emphasise that in these visualisations the 'Flight Distance' function is used on the x-axis, while the y-axis represents a merged representation of the remaining data features. This approach was adopted in order to maintain a logical interpretation of the groups formed, allowing us to observe how flight distance correlates with other combined features. This is

the first indication of how people can be grouped on the basis of their satisfaction with the trip.

Taking this further, using 'flight distance' on the x-axis and a composite of other characteristics on the y-axis provides a unique perspective on people's preferences. By analysing these clusters, we can begin to uncover patterns and trends that may not be immediately apparent when looking at data characteristics in isolation. For example, certain clusters may reveal that passengers travelling longer distances have different expectations or satisfaction levels than those on shorter flights. This nuanced understanding enables airlines and service providers to tailor their offerings more effectively, ensuring that customer satisfaction is optimised across different travel experiences. In addition, this clustering analysis can serve as a fundamental step in developing personalised marketing strategies and improving overall service quality, demonstrating the power of data-driven insights to improve customer loyalty and satisfaction.

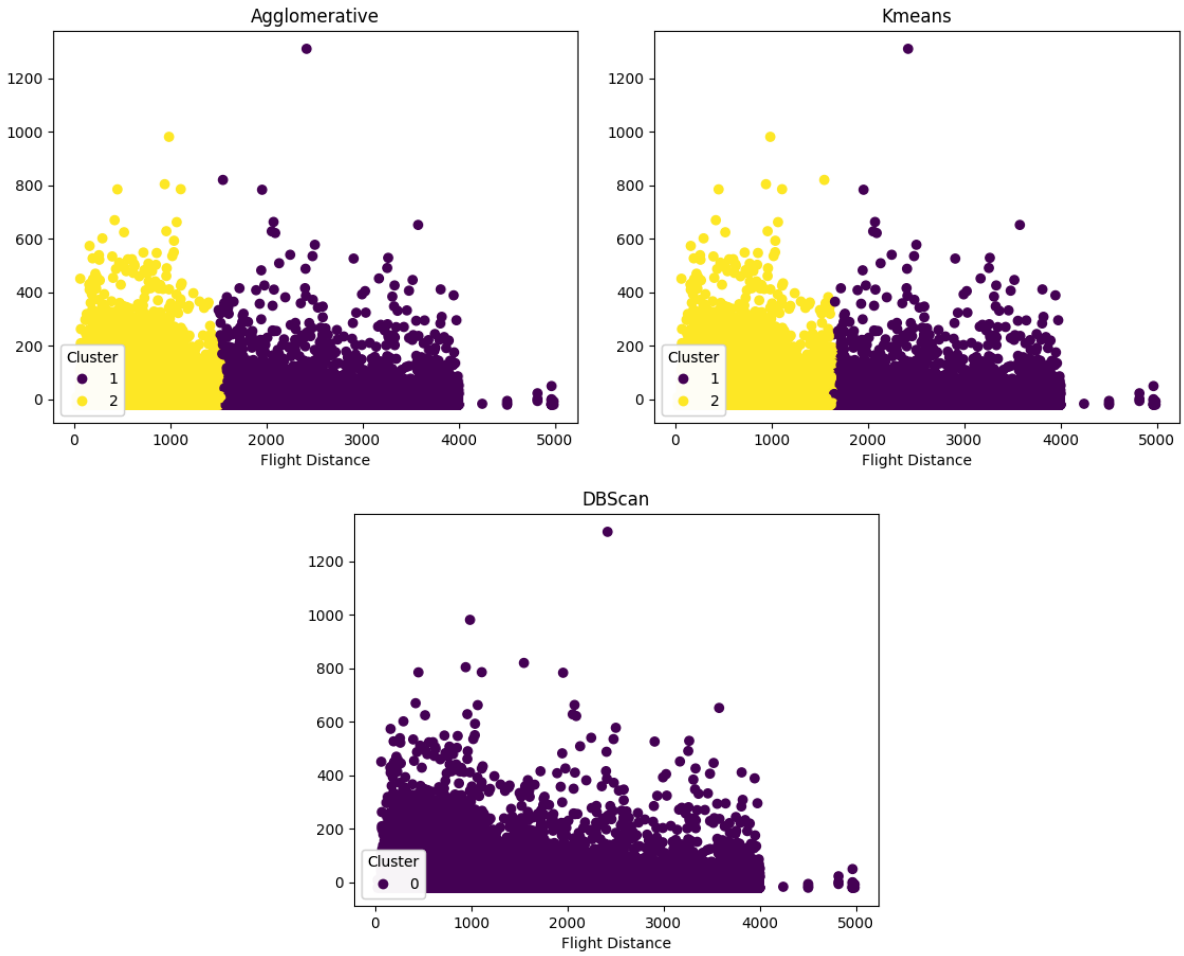

Figure 5.11: Left: Agglomerative Clustring, Right: Kmeans clustering, Below: DBScan Clsutering

In the figure 5.11, we observe the clustering results from the popular methods, where both Agglomerative Clustering and KMeans reveal a fairly clear structure of twoclusters. For the DBSCAN algorithm, the dense nature of the dataset hinders its ability to distinctly separate

the clusters. The variation in outcomes underscores the necessity of choosing a clustering algorithm that is well-suited to the dataset's unique attributes and density, emphasizing the critical role of algorithm selection in achieving meaningful clustering results.

In Figure 5.12, we can observe the cluster formations as executed by the three models of the DSclustering algorithm. Just the model where the number of cluster is predefined has distinct clusters, this particular model was also the one we observed to have the best Silhouette Score result.

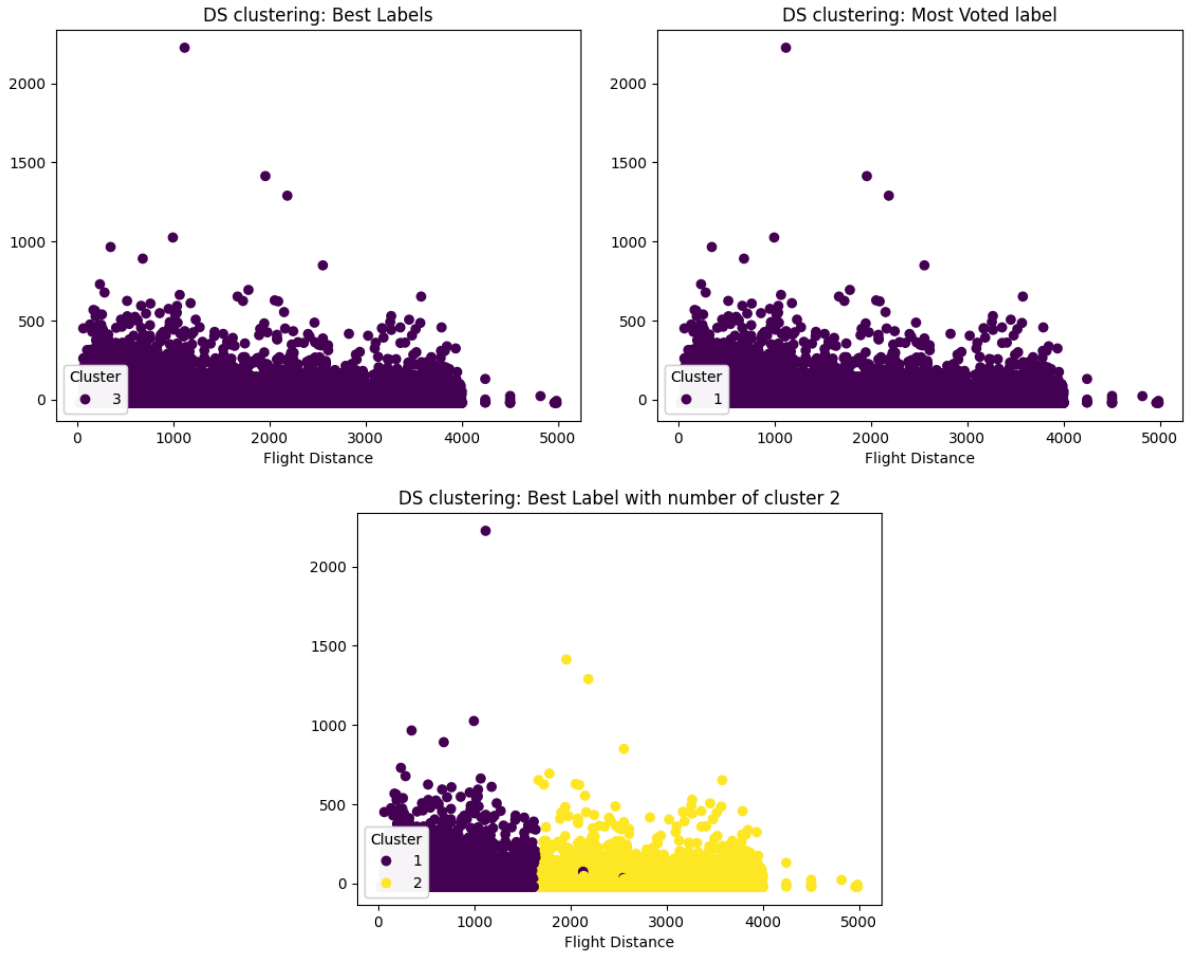

Figure 5.12: Left: DSClustering with Best Labels, Right: DSClustering with Most Voted Labels, Below: DSClustering with number of clusters

Here we can again observe the effectiveness of the models, while repeating the theme that KMeans and DS clustering with number of clusters have better performance, it is shown that Agglomerative and DS clustering Best Label and most voted, are not applicable to this particular dataset and have similar constraints and limitations as DBSCAN.

DS clustering Rules

Next we will present the rules generated by DS clustering: with number of clusters in figure 5.9, this given that it is the only one of the 3 models that generated a distinction of clusters.

N° C	Rule	Score	0	1	Unc
0	522.383 < Flight Distance < 1196.568	0.571	0.571	0	0.429
	Flight Distance < 522.383	0.499	0.499	0	0.501
	Positive Customer Type - 0.186, Flight Distance - 1196.568	0.322	0.322	0	0.678
	Customer Type = 1	0.313	0.313	0	0.687
	Negative Type of Travel - 0.306, Class - 0.591	0.295	0.286	0.018	0.696
1	Food and Drink = 0	0.624	0	0.624	0.376
	Flight Distance > 1870.752	0.608	0	0.608	0.392
	Negative Customer Type - 0.186, Flight Distance - 1196.568	0.345	0	0.345	0.655
	Negative Type of Travel - 0.306, Flight Distance - 1196.568	0.303	0	0.303	0.697
	On-board service = 0	0.280	0	0.280	0.720

Table 5.9: Rules generated for Airline Passenger Survey with DS clustering: with number of clusters

This observation regarding the Flight Distance rule as a significant factor in clustering by satisfaction levels sheds light on the nuanced relationship between travel logistics and passenger satisfaction. The dataset's goal to categorize individuals based on their satisfaction reveals an intriguing trend: as travel distance increases, the likelihood of a passenger rating their experience as satisfactory decreases, potentially due to the compounded effects of various service-related aspects like in-flight meals or customer service quality.

The specific rule range of 522.383 < Flight Distance < 1196.568 serves as a critical threshold, indicating a zone where passengers' satisfaction begins to waver more noticeably. This range could represent short to medium-haul flights, where passengers might have heightened expectations for comfort and service that are not fully met, affecting their overall satisfaction. It suggests that airlines need to pay particular attention to these flight segments, possibly by enhancing service offerings or improving the overall travel experience to mitigate the negative impact of longer flight durations on passenger satisfaction.

Moreover, this insight can guide airlines in tailoring their customer experience strategies, emphasizing the importance of understanding how different segments of travelers perceive their journey. For instance, providing additional amenities or improving service efficiency for flights within this critical range could significantly enhance passenger satisfaction. It also highlights the potential for airlines to adopt a more data-driven approach in service design, focusing on the specific needs and expectations of passengers across various flight distances to ensure a consistently satisfactory travel experience.

5.4 General Discussion

The DSClustering algorithm demonstrated consistent performance across a variety of datasets. It effectively leveraged the strengths of KMeans, DBSCAN, and Agglomerative Clustering, incorporating a unique rule-based approach that proved robust for various clustering tasks.

If we make a qualitative comparison with the SHAP model, although they share features, such as that rules are additive, and have a weight or importance, we should consider that SHAP is easier to interpret if and only if users are familiar with the importance of features in machine learning models and want to understand the contribution of each feature to specific predictions, but this requires a lot of prior knowledge about the data. The DS classifier employs Dempster-Shafer theory, which is useful in contexts where uncertainty and the combination of evidence from multiple sources are important.

The results produced by DS clustering and its rules can be more easily understood as they provide information on the uncertainty of the rules generated. Additionally, if there is prior knowledge, such as that required for SHAP, these data characteristics can be incorporated as expert experience to be considered during rule generation.

With all the results presented throughout this work, we can list the following conclusions obtained according to the results of the experiments:

- The algorithm's ability to achieve high silhouette scores across datasets indicates its effectiveness in identifying well-separated clusters, a critical aspect in many practical applications.
- The adaptability of DSClustering to different datasets, along with its versatility in handling both simple and complex data structures, was a notable strength. This adaptability was evident in its ability to generate meaningful and interpretable rules, even with complex datasets.
- A key advantage of the DSClustering algorithm was its interpretability. The use of rules provided clear guidelines for cluster assignment, enhancing transparency and user trust in the model.

Chapter 6

Conclusion

6.1 Work Performed

The project successfully developed the DS clustering algorithm, which incorporates the strengths of traditional clustering methods and enhances them with a rule-based approach.

The algorithm was tested on a range of datasets, from simple uniform lines to more complex structures like Gaussian mix distributions. It demonstrated strong performance across these varying datasets. One of the primary objectives, ensuring the interpretability of the clustering results, was achieved. The algorithm provided clear and understandable rules for each cluster assignment.

With respect to the specific objectives raised in Section 1.2, it can be said that were achieved almost in their entirety, in addition to this, the objectives served as a basis for step-by-step development of work, were the project adeptly adapted the DS classifier, initially designed for classification, to address clustering challenges, focusing on maximizing data clustering validity. Concurrently, a specialized interpretability model was developed for this new clustering algorithm, emphasizing the ease of understanding the data grouping process.

Through comparative experiments against established clustering methods, the project not only validated the effectiveness of the adapted classifier but also demonstrated its superior balance between clustering validity and model interpretability, underscoring its potential against traditional clustering techniques

Finally, it is important to highlight that DS clustering is an innovation in the field of data analysis for clustering, as it not only explains the clusters obtained for each data instance, but also provides probabilities for each feature. This is achieved with a similar level of validity to the popular clusterings mentioned above, allowing us to say that the main advantage of using this method is the inherent interpretability it offers. This interpretability not only facilitates our understanding of the clustering process, but also provides crucial information by quantifying the uncertainty associated with each clustering decision. This is a measure that is not available in other interpretable clustering methods.

6.2 Future Work

Despite having achieved the objectives, there are some details that can be perfected to improve both the accuracy and performance of the system, such as:

- Future work could focus on improving the precision of the clustering results further, enhancing the metrics evaluated in the initial experiments. And for example, addin dunn's score to the ClusteringSelector
- Testing the algorithm on larger and more diverse datasets would provide deeper insights into their scalability and robustness.
- Applying the DS clustering algorithm to real-world scenarios could provide practical insights and highlight areas for further refinement.
- Exploring more sophisticated methods for rule generation could improve the algorithm's interpretability, especially in complex data scenarios.
- Developing methods for the automated tuning of the algorithm's parameters could enhance its usability and effectiveness in different data environments.
- One of the foremost tasks for future work is to compare the current clustering models, particularly DS clustering, with the ICOT algorithm[34]. ICOT stands out for its use of interpretable trees in clustering, offering a unique approach to understanding data groupings. A comparative analysis with ICOT would provide a benchmark for evaluating the effectiveness of our models
- Another important aspect to address in future work is the time efficiency of the model, especially considering that DSClustering requires training a classifier. This process can be more time-consuming compared to other clustering algorithms, which may limit its applicability in scenarios where rapid data processing is crucial.

Bibliography

[1] V. Antoine, B. Quost, M.-H. Masson, and T. Denœux. Cecm: Constrained evidential c-means algorithm. *Computational Statistics Data Analysis*, 56(4):894–914, 2012.

[2] A. G. Asuero, A. Sayago, and A. G. González. The correlation coefficient: An overview. *Critical Reviews in Analytical Chemistry*, 36(1):41–59, 2006.

[3] J. Basak and R. Krishnapuram. Interpretable hierarchical clustering by constructing an unsupervised decision tree. *IEEE Transactions on Knowledge and Data Engineering*, 17(1):121–132, 2005.

[4] Dimitris Bertsimas, Agni Orfanoudaki, and Holly Wiberg. Interpretable clustering: an optimization approach. *Machine Learning*, pages 1–50, 2020.

[5] Ibrahim Dabbura. K-means clustering: Algorithm, applications, evaluation methods, and drawbacks. *Towards Data Science*, 2018.

[6] DataScience.eu. Interpretación de su modelo de aprendizaje profundo por shap, Jan 2024. Accedido por última vez: 04-03-2024.

[7] DataScientest. Machine learning clustering: el algoritmo dbscan, Feb 2024. Accedido por última vez: 04-03-2024.

[8] Thierry Denoeux. Evclus: Evidential clustering of proximity data. *IEEE Transactions on Systems, Man, and Cybernetics*, 25(5):803–813, 1995.

[9] Google Developers. Advantages and disadvantages of k-means and dbscan, Feb 2024. Accedido por última vez: 04-03-2024.

[10] J. C. Dunn. Well separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4(1):95–104, 1974.

[11] J. Gallardo. Cluster analysis, 2024. Accedido por última vez: 04-03-2024.

[12] Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. *IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pages 80–89, 2019.

[13] IBM. Análisis de clústeres de k-medias. `https://www.ibm.com/docs/es/SSLVMB_sub/statistics_mainhelp_ddita/spss/base/idh_quic.html`, n.d.

[14] Connor Lawless, Jayant Kalagnanam, Lam M Nguyen, Dzung Phan, and Chandra Reddy. Interpretable clustering via multi-polytope machines. *Thirty-Fifth AAAI Conference on Artificial Intelligence*, 2021.

[15] E. Leonguz. Métricas para la validación de clustering. `https://disi.unal.edu.co/~eleonguz/cursos/mda/presentaciones/validacion_Clustering.pdf`, 2015. [PDF].

[16] Yimou Li, David Turkington, and Alireza Yazdani. Beyond the black box: an intuitive approach to investment prediction with machine learning. *The Journal of Financial Data Science*, 2(1):61–75, 2020.

[17] Zachary Chase Lipton. The mythos of model interpretability. *CoRR*, abs/1606.03490, 2016.

[18] Eric Yi Liu, Zhaojun Zhang, and Wei Wang. Clustering with relative constraints. In *Knowledge Discovery and Data Mining*, 2011.

[19] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30:4765–4774, 2017.

[20] Marie-Hélène Masson and T. Denœux. Ecm: An evidential version of the fuzzy c-means algorithm. *Pattern Recognition*, 41(4):1384–1397, 2008.

[21] Christoph Molnar. Interpretable machine learning: A brief history, state-of-the-art and challenges. *arXiv preprint arXiv:1904.05146*, 2019.

[22] University of Valencia. AnÁlisis cluster. `https://www.uv.es/ceaces/multivari/cluster/CLUSTER2.htm`, n.d.

[23] Karl Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine Series 5*, 50(302):157–175, 1900.

[24] Sergio Peñafiel, Nelson Baloian, Horacio Sanson, and José A. Pino. Applying dempster–shafer theory for developing a flexible, accurate and interpretable classifier. *Expert Systems with Applications*, 148:113262, 2020.

[25] William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[26] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

[27] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[28] Sandhya Saisubramanian, Sainyam Galhotra, and Shlomo Zilberstein. Balancing the tradeoff between clustering value and interpretability. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, AIES '20, page 351–357, New York, NY, USA, 2020. Association for Computing Machinery.

[29] Glenn Shafer. A mathematical theory of evidence. *Princeton university press*, 1976.

[30] Glenn Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[31] StatDeveloper. Introducción al clustering dbscan, Feb 2024. Accedido por última vez: 04-03-2024.

[32] Alaa Tharwat. Classification assessment methods. *Applied Computing and Informatics*, 17(1):168–192, 2021.

[33] Unknown. A unified approach to interpreting model predictions, Feb 2024. Accedido por última vez: 04-03-2024.

[34] Holly Wiberg. Interpretable clustering via optimal trees. *arXiv*, Dec 2018.

[35] Wikipedia. Análisis de grupos. `https://es.wikipedia.org/wiki/An%C3%A1lisis_de_grupos`. Accedido por última vez: 04-03-2024.

[36] Wikipedia. Dbscan. `https://es.wikipedia.org/wiki/DBSCAN`. Accedido por última vez: 04-03-2024.

[37] Wikipedia. Silhouette (clustering) — wikipedia, the free encyclopedia, 2024. [Online; 21-02-2024].

[38] Ronald R. Yager, Janusz Kacprzyk, and Mario Fedrizzi, editors. *Advances in the Dempster-Shafer Theory of Evidence*. Wiley, 1994.

ANNEX

Experiments and results

Cluster	Rule	Importance	0	1	2	Unc
0	R57: 751.451 < proline < 966.533	0.351	0.351	0.000	0.000	0.649
	R8: 12.403 < alcohol < 13.005	0.288	0.288	0.000	0.000	0.712
	R16: 2.191 < ash < 2.368	0.273	0.273	0.000	0.000	0.727
	R157: Negative alcalinity_of_ash - 19.530, nonflavanoid_phenols - 0.373	0.238	0.238	0.000	0.000	0.762
	R242: Positive nonflavanoid_phenols - 0.373, proline - 751.451	0.237	0.226	0.023	0.000	0.751
1	R11: malic_acid < 1.685	0.479	0.081	0.440	0.000	0.479
	R29: 2.299 < total_phenols < 2.764	0.351	0.000	0.351	0.000	0.649
	R58: proline > 966.533	0.338	0.000	0.286	0.114	0.601
	R18: ash > 2.545	0.333	0.002	0.332	0.000	0.666
	R34: flavanoids > 2.753	0.324	0.000	0.263	0.138	0.600
2	R12: 1.685 < malic_acid < 2.403	0.651	0.000	0.000	0.651	0.349
	R50: hue > 1.104	0.520	0.000	0.104	0.470	0.426
	R53: 2.544 < od280/od315_of_diluted_wines < 3.035	0.477	0.000	0.066	0.446	0.488
	R17: 2.368 < ash < 2.545	0.468	0.105	0.000	0.419	0.476
	R19: alcalinity_of_ash < 17.105	0.458	0.000	0.090	0.415	0.495

Table .1: Most Important Rules for Each Cluster of Wine Dataset