UPC UB URV

MASTER IN ARTIFICIAL INTELIGENCE

Project 1: Clustering Exercise

Introduction to Machine Learning

Professor:
Dr. Maria Salamó
Course: 2023-2024.
Laboratory group 12L

Students:
Wafaa Guendouz
Alam Lopez
Laura Roldán
Mario Rosas
Abel Briones

October 23, 2023

Project 1

Clustering Exercise

Table of Contents

1 Introduction

Nowadays AI is based on data-driven decision-making, the field of machine learning continues to evolve, offering powerful tools to make sense of large datasets in order to produce models applicable to the real world. Clustering, a fundamental technique in machine learning, consists grouping similar data points together, commonly in pattern recognition, data exploration, and problem-solving across various domains. The objective of this report is to delve into the analysis and evaluation of different clustering algorithms by applying them to several datasets.

Algorithms explored include DBSCAN, BIRCH, K-Means, K-Modes, K-Prototypes and Fuzzy C-Means. The analysis covers a wide spectrum of data types, from numerical to categorical and mixed attribute data sets.

Throughout this report, we explore the performance of these clustering algorithms, evaluating their ability to reveal hidden structures within the data, and we analyze their sensitivity to different metrics and parameters. The primary aim is to gain a comprehensive understanding of how each algorithm operates and to determine their suitability for diverse data types and scenarios.

2 Datasets

The selected datasets for this task, namely **"Vowel," "Waveform,"** and **"Kr-vs-Kp,"** exhibit diverse data structures, making them invaluable for analytical exploration. Each dataset showcases unique characteristics, providing a comprehensive understanding of various scenarios in data clustering.

	#Classes	**#Samples**	**#Numeric features**	**#Categorical features**
Waveform	3	5000	40	0
Kr-vs-Kp	2	3196	0	36
Vowel	11	990	10	3

Table 1: Overview of Selected Datasets

These datasets were carefully chosen due to their diversity. Firstly, they represent a variety of data types: numerical, categorical, and mixed. Secondly, they differ in terms of data size, ranging from large datasets to smaller ones. Furthermore, the number of classes varies, with some datasets having multiple classes while others have only a few. Additionally, the datasets vary in the number of features they encompass, showcasing a wide spectrum from datasets with numerous features to those with a more limited set. Finally, all of them are well-balanced.

2.1 Dataset 1: Waveform

This first dataset is **Waveform**, a collection of simulated time series signals Breiman und Stone [1988].

Dataset information:

- **Number of classes:** 3

- **Number of samples:** 5000
- **Number of features:** 40 (numerical)
- **Missing values:** 0

The name of the features goes from x1 to x40, as mentioned, all are numerical. Moreover, an important point about this dataset is that the latter 10 attributes are all noise attributes with mean 0 and variance 1. This is visualized clearly in the correlation matrix.

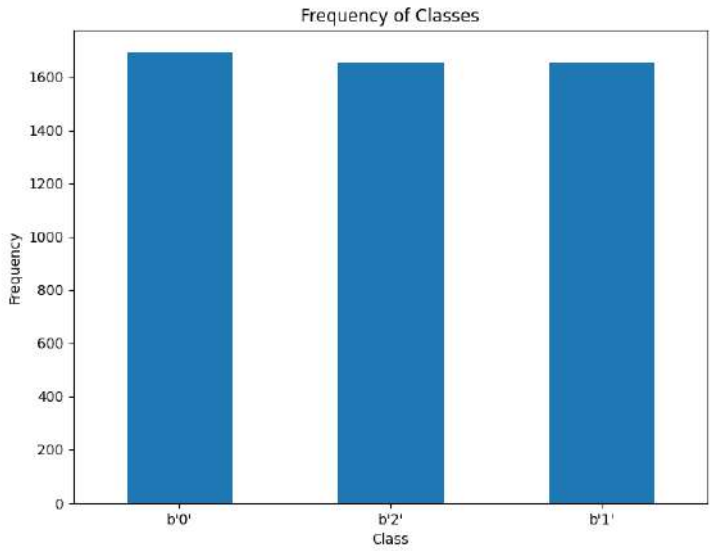

Figure 1: Samples for each class

As shown in the figure 1, all three classes have a similar number of samples, meaning that this dataset is balanced.

A comprehensive analysis of the correlation matrix 2 reveals three distinctive correlation patterns among the features:

- **Positive Correlation (Orange Area):** This area indicates a strong positive correlation among certain features. For instance, an increase in the value of x7 corresponds to an increase in x5.
- **Negative Correlation (Dark Blue Area):** This area indicates a strong negative correlation among certain features. For instance, an increase in the value of x15 corresponds to a decrease in x8.
- **No Correlation (Light Blue Area):** Some features, such as x35 and x10, demonstrate no significant association.

Furthermore, figure 3 provides more information about the features that have a correlation among them. The scatter plots show relationships between pairs of variables, where each point is colored based on the groundtruth class. The diagonal shows the three distributions of a feature for each class.

When features like x1 and x4 have consistent values across classes, it complicates the clustering analysis. The similarity in these features among classes blurs the distinctions, making it challenging to identify clear patterns.

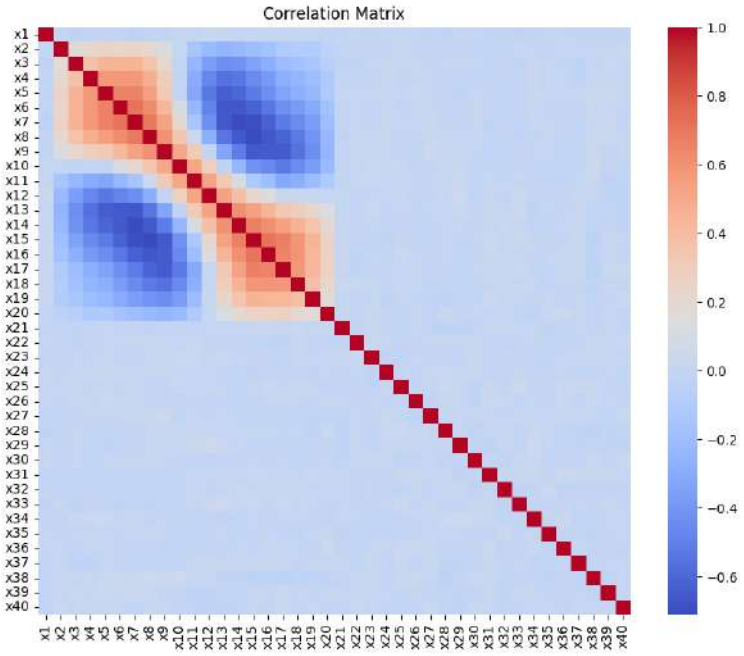

Figure 2: Correlation matrix

2.2 Datset 2: Kr-vs-Kp

Kr-vs-Kp Shapiro [1989] is a chess endgame scenario dataset. The position is described using a sequence of 36 attribute values representing the board. The features include various aspects such as the positions of the black king and rook, the presence of pawns, and other contextual information. The game is in the King+Rook versus King+Pawn on a7 (KRKPA7) configuration, with the pawn on a7 just one square away from queening. It is White's turn to move.

Dataset information:

- **Number of classes:** 2
- **Number of samples:** 3196
- **Number of features:** 36 (categorical)
- **Missing values:** 0

The features are categorical: ['bkblk', 'bknwy', 'bkon8', 'bkona', 'bkspr', 'bkxbq', 'bkxcr', 'bkxwp', 'blxwp', 'bxqsq', 'cntxt', 'dsopp', 'dwipd', 'hdchk', 'katri', 'mulch', 'qxmsq', 'r2ar8', 'reskd', 'reskr', 'rimmx', 'rkxwp', 'rxmsq', 'simpl', 'skach', 'skewr', 'skrxp', 'spcop', 'stlmt', 'thrsk', 'wkcti', 'wkna8', 'wknck', 'wkovl', 'wkpos', 'wtoeg']

Most of them are denoted by binary values 't' or 'f', except for a few: 'dwipd' ('g', 'l'), 'katri' ('b', 'n', 'w'), and 'wtoeg' ('n', 't', 'f'). These features capture specific conditions on the chessboard.

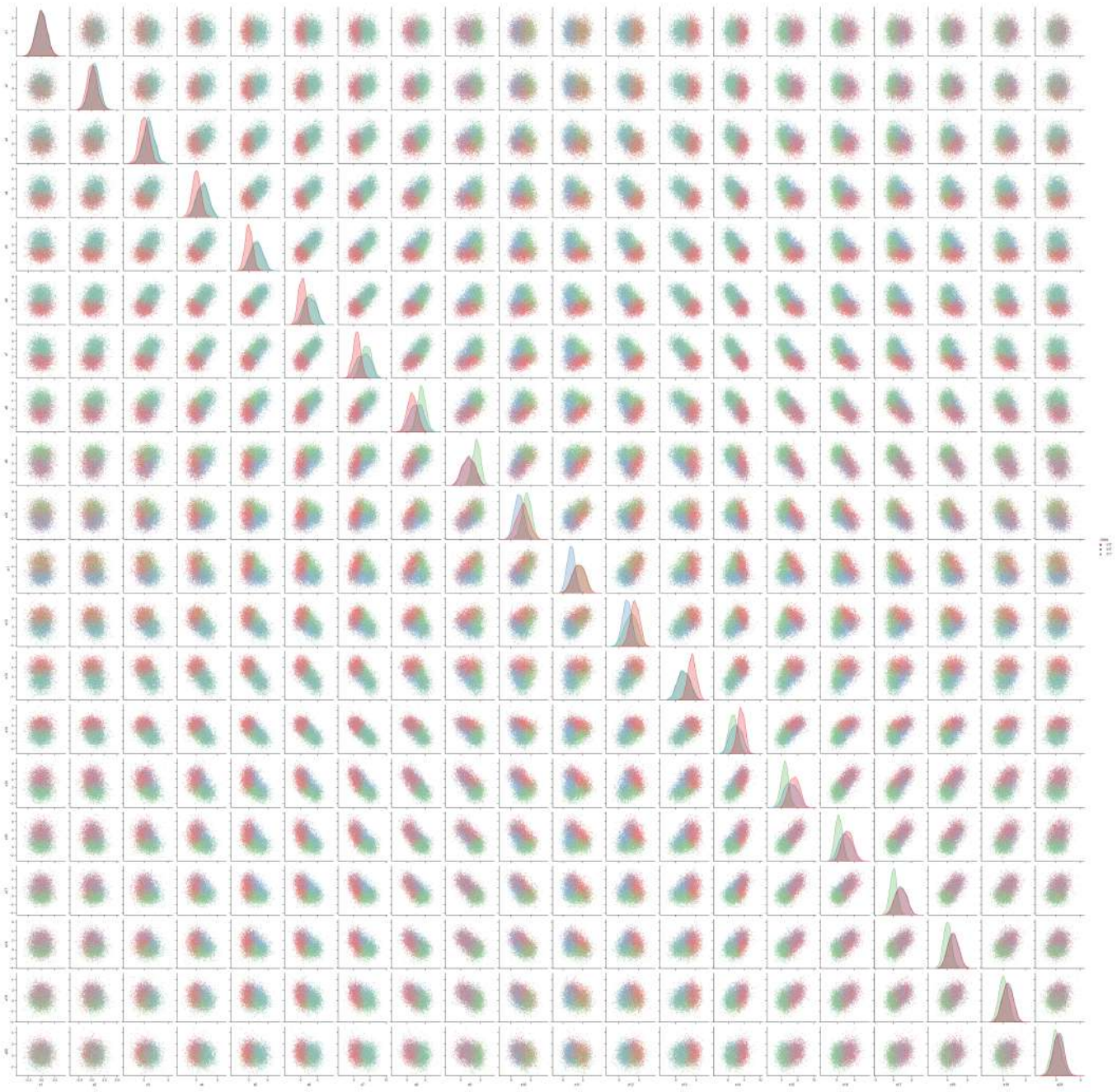

Figure 3: Relation between pair of features

The objective of the game is to predict the outcome based on these features. The possible classes are "won" (indicating White can win) and "nowin" (indicating White cannot win).

As shown in figure 4, all two classes have a similar number of samples, meaning that this dataset is balanced.

In this dataset, all features hold equal importance and are directly correlated. Each feature in an instance explains a specific aspect of the board position, making them essential for understanding the chess endgame scenario.

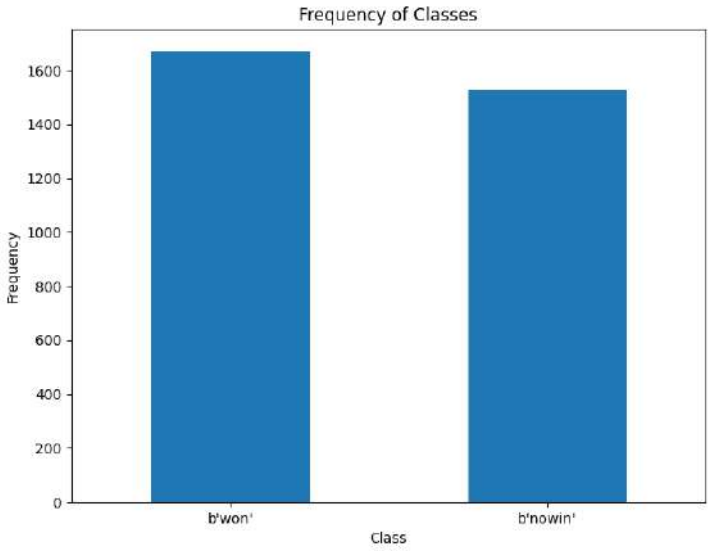

Figure 4: Samples for each class

2.3 Dataset 3: Vowel

Vowel Recognition Speaker [1] independent recognition of the eleven steady state vowels of British English using a specified training set of LPC derived log area ratios.

Dataset information:

- **Number of classes:** 11
- **Number of samples:** 990
- **Number of features:** 10 (numerical), 3 (categorical)
- **Missing values:** 0

The dataset comprises 11 different vowel sounds: hid, hId, hEd, hAd, hYd, had, hOd, hod, hUd, hud, hed. It includes 10 numerical features representing speech signals, spoken by fifteen different speakers.

Moreover, "Train_or_Test" is a binary categorical feature indicating whether the data is part of the training or testing set. "Speaker" consists of a list of fifteen names representing male and female speakers, and "Sex" is a binary categorical feature indicating the gender of the speaker. For visualization proposes, it is mapped the feature "Sex": "Male" is 1, "Female" is 0. The other two categorical data are not relevant. It will be explained in more detail in the section about preprocessing data.

As shown in figure 5, all eleven classes have the same amount of samples, meaning that this dataset is balanced.

The correlation matrix depicted in Figure 6 illustrates that each feature in the dataset is correlated with one another.

In Figure 7, the relationships between pairs of features are depicted. Observing these pairs reveals notable patterns in the data. For instance, Feature_9 presents challenges in recognizing distinct clusters for each class due to its distribution

[1] https://hastie.su.domains/ElemStatLearn/datasets/vowel.info.txt

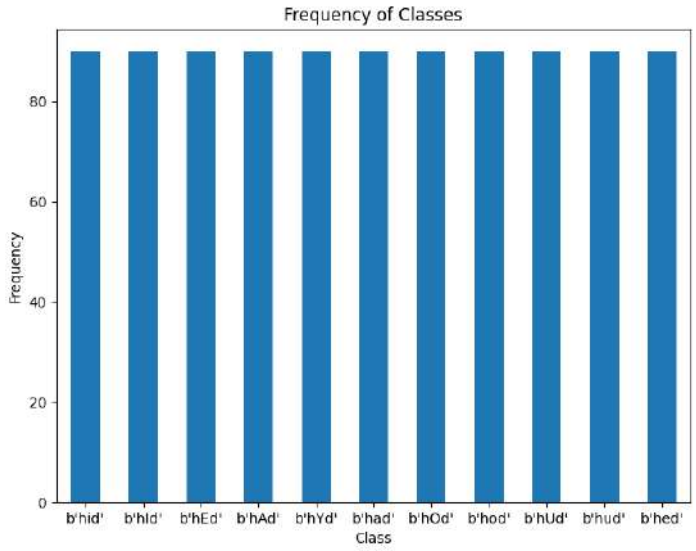

Figure 5: Samples for each class

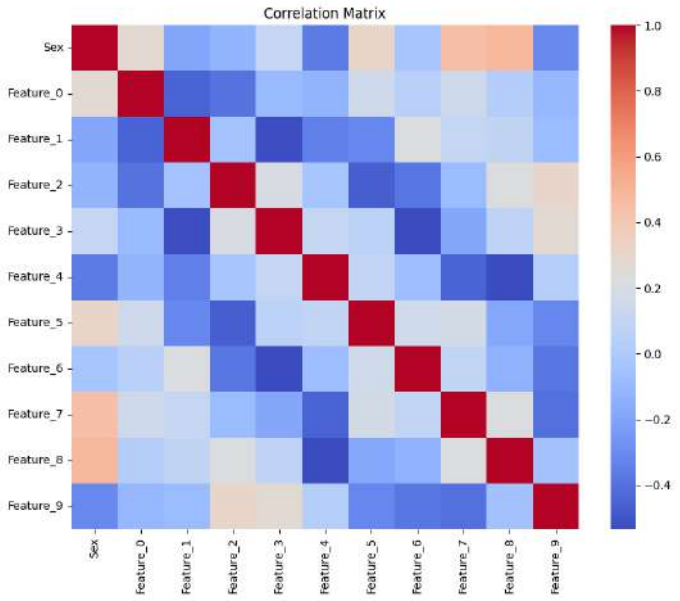

Figure 6: Correlation matrix

Figure 7: Relation between pair of features

characteristics. In contrast, Feature_0 exhibits a clearer and more discernible visualization, making it easier to identify patterns associated with different classes.

A common trend observed is the similarity in numerical feature values across the dataset. This similarity poses a challenge for clustering tasks, as the subtle differences that define different classes become harder to distinguish.

3 Data Preprocessing

Preparing the data for clustering is a key step in the process. It involves a series of operations to ensure that the data is ready for applying clustering algorithms effectively. Different validations on the nature of the raw data and the standards expected from it at the end of the preprocessing must be properly performed to make sure that the data is well adapted to our clustering algorithms. Moreover, standardization and label encoding processes are performed as necessary to prepare the data for clustering as stated in García u. a. [2016].

3.1 Process of data manipulation and techniques

1. **Data import.** The process begins by importing the raw data from ARFF files. The data is read using a library built-in function, and the metadata is extracted for reference. Data are allocated in a DataFrame structure to facilitate their manipulation for any necessary transformations.

2. **Data type handling.** The dataset may contain a mix of numerical and categorical attributes. A string decode function is applied to ensure that object data types are appropriately decoded.

3. **Labels separation.** The dataset's labels are separated from the features.

4. **Standarization.** The data standardization process is applied to prepare the features for clustering. The choice of standardization method, whether numerical, categorical, or mixed, is determined by the *method* parameter. For both numerical and categorical features, standardization includes imputing missing values with the mean or 'most frequent' strategy according to the data type, scaling is also performed with options to include mean and standard deviation in the process. For categorical features, options for one-hot encoding or ordinal encoding are available, depending on the setting of an specific parameter. For the handling of numerical characteristics with algorithms adapted for categorical variables we use a specific function to generate the discretization of the values according to the range contained in the data.

5. **Label encoding.** Labels are encoded numerically for clustering purposes. A mapping between original labels and their numeric counterparts is created.

6. **Statistics.** General statistics of the dataset, both before and after preprocessing, can be obtained using the statistics method. This includes information about the number of classes, features, instances, and various statistics for each feature.

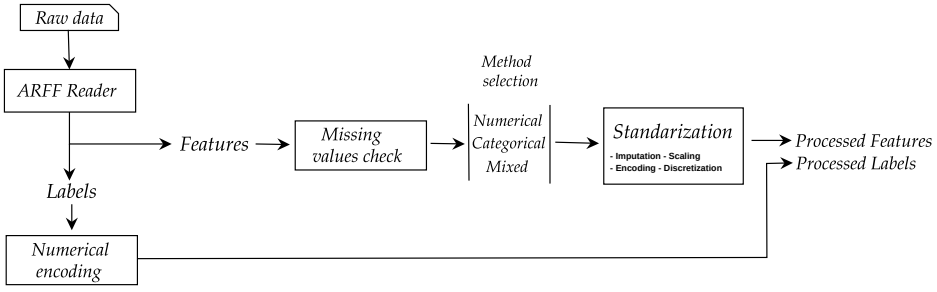

Figure 8: Data processing pipeline

As shown in the figure 8, the raw data is imported and separated into features and labels. The dataset is analyzed for missing values. Depending on the data type (numerical, categorical, or mixed), different preprocessing steps are applied. For numerical features, missing values are imputed with the mean, and scaling is performed. For categorical features, one-hot encoding or ordinal encoding is applied, as specified. The labels are encoded numerically. Lastly the trasnformed dta is returned.

4 Implementation of Clustering Algorithms

4.1 Implementations using Scikit-learn

4.1.1 DBSCAN

DBSCAN, or Density-Based Spatial Clustering of Applications with Noise, is a clustering algorithm known for its ability to handle irregularly shaped clusters and robustness against outliers. Unlike traditional methods, DBSCAN does not require the user to specify the number of clusters in advance, making it suitable for cases where the optimal cluster count is uncertain. It is less sensitive to initialization conditions and is relatively fast, making it efficient for processing large datasets in real-time applications.

However, DBSCAN has limitations. It struggles with datasets containing categorical features and clusters of varying density. The algorithm relies on distance metrics and is sensitive to the scale of variables, often requiring data rescaling. Additionally, it faces challenges in high-dimensional data.

The code performs DBSCAN clustering on a dataset, testing various combinations of parameters (epsilon, minimum samples, similarity metric, and distance algorithm). It evaluates clustering quality using the evaluation metrics chosen. The best parameters for each metric are the ones selected to analyze the results.

4.1.2 BIRCH

Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) is a clustering algorithm designed for large datasets. It offers several advantages. Firstly, BIRCH is highly effective in handling noise within datasets, making it robust for real-world applications where data imperfections are common. Secondly, it excels in producing high-quality clusters and sub-clusters, ensuring accurate patterns are extracted. Moreover, it is often regarded as superior to the DBSCAN clustering algorithm in various aspects.

However, BIRCH has limitations. One major drawback is its restriction to processing only metric attributes, excluding categorical attributes that can't be represented in Euclidean space. This limitation poses challenges, particularly when dealing with diverse data types.

The code performs DBSCAN clustering on a dataset. It iterates through various combinations of BIRCH parameters (threshold and branching factor). It evaluates clustering quality using the evaluation metrics chosen. The best parameters for each metric are the ones selected to analyze the results.

4.2 Implementations with our own code

4.2.1 K-Means + K-Modes

KMeans is a partition-based clustering algorithm as stated in Steinbach u. a. [2005], designed to partition n observations into k clusters, where each observation belongs to the cluster with the nearest mean (or centroid). The primary objective

of KMeans is to minimize the within-cluster sum of squares. Our K-means implementation follows these steps:

Algorithm 1 K-Means

Initialize k centroids randomly from the data points.

while $iter < max\_iter$ ***or*** $Y_t = Y_{t-1}$ **do**

- For each data point, compute its distance to each centroid.
- Assign each data point to the cluster of its closest centroid.
- Compute the new centroid (mean) for each cluster based on the assigned data points.
- If centroids do not change significantly or another convergence criterion is met, EXIT loop.

end

K-Modes algorithm is similar to the K-Means algorithm but is designed to handle categorical data. Unlike KMeans, KModes minimizes the dissimilarity (calculated using Hamming distance) within clusters. KModes uses "modes" as cluster centroids instead of "means" because means aren't meaningful for categorical data. A "mode" in this context is the most common value for each feature in a cluster.

Algorithm 2 KModes

Data: Dataset D of size $m \times n$, Number of clusters k, Maximum iterations `max_iter`

Result: Cluster labels for each data point, Modes of clusters

Initialization:

Randomly select k data points from D as initial cluster centers: $C = \{c_1, c_2, \ldots, c_k\}$.

Set `iteration` $= 0$.

while *iteration* $<$ `max_iter` ***and*** *changes in cluster assignments* **do**

- **for** *each data point* d_i *in* D **do**
  - **for** $j = 1$ *to* k **do**
    - Compute dissimilarity $s_{ij} = \text{HammingDistance}(d_i, c_j)$.
  - **end**
  - Assign d_i to cluster c_j such that s_{ij} is minimum.
- **end**
- **for** *each cluster center* c_j **do**
  - Update c_j as the mode of all data points assigned to cluster c_j.
- **end**
- Set `iteration` $=$ `iteration` $+ 1$.

end

return *Cluster labels, Modes* C.

4.2.2 K-Prototypes

The K-Prototypes algorithm (Huang [1998]) is a hybrid clustering method designed to handle data sets with a mixture of numerical and categorical attributes. It combines the strengths of K-Means for numerical data with an innovative approach based on K-Modes to measure the dissimilarity of categorical data. K-Prototypes provides a complete solution for clustering diverse data, allowing the discovery of hidden patterns in datasets where attributes have multiple data types.

Key technical characteristics of the K-Prototypes algorithm according to Aprilliant [2023] include :

- Utilize a distance metric that accommodate mixed data types, enhancing cluster quality.
- Balance the contributions of numerical and categorical features during clustering.

Algorithm 3 K-Prototypes

Data: $X \leftarrow$ Data points (num & cat) $K \leftarrow$ number of clusters
$\gamma \leftarrow$ categorical weight $max\_iter \leftarrow$ maximum number of iterations
Result: $y\_pred \leftarrow$ predicted labels

$C = X[random(k)]$ /* Randomly select K data points as initial cluster centroids */
$Y_t = argMin(dist(X,C))$ /* Assign each data point to the random clusters */
$t = 0$
while $t < max\_iter$ ***or*** $Y_t = Y_{t-1}$ **do**
  $t = t + 1$
  for $c \in C$ **do**
    $c = mean(X[c\_num]) + mode(X[c\_cat])$ /* Update num & cat feats of each clusters */
  end
  for $x \in X$ **do**
    $dist = sdistN(x,C) + \gamma \cdot distC(x,C)$ /* Compute num & cat distance features to clusters */
    $Y_t[x] = argMin(dist)$ /* Assign the data point to the nearest cluster */
  end
end
$y\_pred = Y_t$

4.2.3 Fuzzy C Means

Fuzzy Clustering algorithms, instead of the previous ones (K-means, K-prototypes, etc.) are soft cluster algorithms. This means that, instead of having a single strict cluster for each element, it computes a grade of membership for each of the instances to every cluster. This is something to take into consideration since the time to compute will be higher because it outputs a membership matrix U that has a size n by c, where n is the number of instances in the dataset and c is the number of clusters.

Within the soft clustering algorithms we selected the Fuzzy C-Means algorithm (FCM) (Cannon u. a. [1986]) that is very similar in the implementation to the K-means. In fact, the FCM would be equal to K-means if the fuzzy parameter is set to 1: m=1.

Algorithm 4 Fuzzy C Means

Initialize a random membership matrix U with size $n \times c$.

for $i = 0$; $i < max\_iters$; $i++$ **do**

Calculate the centers of the c clusters, given the U matrix.

Compute the distance from each cluster center, to all the data points.

With all the distances, calculate the new membership matrix U

If the distance between the new matrix U and the previous one does not change significantly (below an epsilon) or another convergence criterion is met, EXIT loop.

end

5 Evaluation Metrics

To evaluate the performance of our clustering algorithms we used a group of evaluation metrics. Depending on the availability of true labels and the objective of clustering, different metrics can be applied. Broadly, these metrics are classified into two categories: External and Internal as explained in Palacio-Niño und Berzal [2019].

5.1 External Evaluation Metrics

These metrics require the knowledge of the true labels of the data points. They evaluate how closely the clusters formed by the algorithm match the true labels:

- **Accuracy:** Measures the data points correctly clustered after aligning the predicted clusters to the true labels.
- **Adjusted Rand Index (ARI):** Compares the similarity between true and predicted labels, adjusted for chance. An ARI of 1 indicates perfect clustering.
- **Homogeneity:** Measures if each cluster contains only members of a single class.
- **Completeness:** Measures if all members of a given class are assigned to the same cluster.
- **V-Measure:** Harmonic mean of homogeneity and completeness.
- **Adjusted Mutual Information (AMI):** Measures clustering quality using mutual information between true and predicted labels, adjusted for chance.

5.2 Internal Evaluation Metrics

These metrics do not require true labels and evaluate the quality of clusters based on the data itself:

- **Silhouette Score:** Measures how close each point in one cluster is to the points in the neighboring clusters. A high value indicates well-separated clusters.
- **The Davies-Bouldin:** it provides a measure of the average similarity between each cluster and its most similar cluster, where "similarity" is a ratio of within-cluster distances to between-cluster distances. Therefore, clusters which are farther apart and less dispersed will result in a better (lower) Davies-Bouldin index.

6 Results

In this section, we present the outcomes of our exploration into various clustering algorithms. From foundational methods like K-Means to advanced techniques such as DBSCAN and K-Prototypes, our analysis spans diverse datasets. We'll provide a concise overview of the results, offering insights into the performance and adaptability of these algorithms, enhancing their practical utility in real-world applications.

6.1 Dataset 1: Waveform

6.1.1 DBSCAN

The DBSCAN algorithm was applied to first dataset consisting of simulated time series signals. Despite exhaustive parameter tuning and rigorous analysis, the clustering results were found to be unsatisfactory.

	Homogeneity	**Completeness**	**V_measure**	**ARI**	**AMI**	**Silhouette**
eps	5	5	5	5	5	7
min_samples	3	40	40	10	40	3
metric	euclidean	euclidean	euclidean	euclidean	euclidean	euclidean
algorithm	auto	auto	auto	auto	auto	auto
#Clusters	30	1	1	4	1	1
SCORE	**0.014**	**0.178**	**0.020**	**0.001**	**0.019**	**0.181**

Table 2: Results for Waveform using DBSCAN

The clustering performance, as measured by the score metrics, remained notably low. The sensitivity of DBSCAN to the choice of parameters, particularly 'eps' and 'min_samples', was evident. Variations in these parameters led to drastic changes in the number of clusters identified, making it challenging to establish a stable and meaningful clustering structure.

Given these challenges, it is apparent that the dataset's inherent complexity, involving the noisy features, posed difficulties for DBSCAN to accurately capture distinct clusters. Added the fact that this first dataset works with high-dimensional data. Just a reminder, the half of features added were noise.

6.1.2 BIRCH

The clustering results using BIRCH are as follows:

	Homogeneity	**Completeness**	**V_measure**	**ARI**	**AMI**	**Silhouette**
Threshold	0.1	0.1	0.1	0.1	0.1	0.1
Branching Factor	10	10	10	10	10	10
#Clusters	3	3	3	3	3	3
SCORE	**0.349**	**0.350**	**0.349**	**0.286**	**0.349**	**0.085**

Table 3: Results for Waveform using BIRCH

Compared to the results obtained from DBSCAN, BIRCH exhibited a significantly improved clustering performance. The algorithm successfully identified three distinct clusters in the data, as evidenced by all the metric validations, making it a consistent result and indicating strong clustering quality.

The BIRCH algorithm, leveraging its hierarchical clustering approach and adaptive nature, managed to capture inherent patterns in the simulated time series signals effectively. The chosen threshold and branching factor values, along with the number of clusters identified, collectively contributed to the improved clustering quality.

This outcome suggests that BIRCH, with its ability to handle large datasets and adapt to varying cluster densities, proved to be a suitable choice for this specific dataset.

6.1.3 K-Means + K-Modes

Upon initial application of the K-Means and K-Modes algorithms on the waveform dataset, a few key challenges emerged. The waveform dataset, primarily numerical, required some preprocessing tweaks to transform it suitably for the K-Modes algorithm, which is designed for categorical data. From the results, it was evident that K-Means with k=7 gave the best accuracy. However, silhouette scores were not particularly high across various k values, suggesting potential overlap between some clusters.

In the case of K-Modes, the best accuracy was achieved when n_clusters=3. Yet, with the highest Davies-Bouldin index value observed for this configuration, it indicates that this cluster might not be as distinct as one would hope.

Group	accuracy	homogeneity	completeness	v_measure	adjusted_rand	adjusted_mutual_info	silhouette	davies
'k': 7	0.761	0.434	0.46	0.447	0.42	0.447	0.055	3.394
'k': 5	0.732	0.405	0.45	0.426	0.393	0.426	0.053	3.476
'k': 2	0.626	0.294	0.466	0.361	0.349	0.36	0.131	2.402
'k': 3	0.521	0.173	0.292	0.217	0.162	0.217	0.125	2.39

Table 4: Results for Waveform using K-means

Group	accuracy	homogeneity	completeness	v_measure	adjusted_rand	adjusted_mutual_info	silhouette	davies
'n_clusters': 3	0.603	0.209	0.215	0.212	0.224	0.212	0.051	5.996
'n_clusters': 2	0.582	0.198	0.314	0.243	0.253	0.243	0.116	2.625
'n_clusters': 5	0.547	0.131	0.147	0.139	0.150	0.138	0.035	6.587
'n_clusters': 7	0.545	0.138	0.218	0.169	0.184	0.168	0.100	2.845

Table 5: Results for Waveform using K-modes

6.1.4 K-Prototypes

The results of K-Prototypes for clustering the Waveform dataset using K-Prototypes are consistent with the K-means results. K-Prototypes, designed for datasets that combine numerical and categorical attributes, but with this dataset it just applies the numerical procesing part. The results indicate variations with different values of K, and the highest

accuracy of 0.761 is achieved when K is set to 7. This suggests that the algorithm identifies seven distinct clusters, which may be an over-segmentation compared to the true three underlying patterns.

Homogeneity and completeness measures reveal that a K value of 7 results in relatively internally consistent and comprehensive clusters. While the V-Measure balances these aspects and also peaks at K=7, it's worth noting that this may lead to a larger number of clusters than the actual three clusters present in the data.

Adjusted Rand Index and Adjusted Mutual Information show varying degrees of agreement between the true three clusters and the predicted clusters, with K=7 showing the highest level of agreement. The silhouette score favors lower values of K (specifically 2) for better cluster separation, while the Davies-Bouldin score suggests that K=7 leads to more compact and well-separated clusters. In conclusion, the results indicate that K-Prototypes, while performing well, tends to produce more clusters than the actual three present in the Waveform dataset.

Group	accuracy	homogeneity	completeness	v_measure	adjusted_rand	adjusted_mutual_info	silhouette	davies
'k': 7	0.761	0.434	0.46	0.447	0.42	0.447	0.055	3.394
'k': 5	0.732	0.405	0.45	0.426	0.393	0.426	0.053	3.476
'k': 2	0.626	0.294	0.466	0.361	0.349	0.36	0.131	2.402
'k': 3	0.521	0.173	0.292	0.217	0.162	0.217	0.125	2.39

Table 6: Results for Waveform using K-Prototypes

6.1.5 Fuzzy C Means

As we can see in the table below, we computed the Fuzzy C Means algorithm increasing the number of clusters from 2 to 7, and the results were similar. In this case, all the metrics of the performance were below the ones obtained with same number of clusters in K-means, K-modes or K-prototypes.

Group	accuracy	homogeneity	completeness	v_measure	adjusted_rand	adjusted_mutual_info	silhouette	davies
C:2	0.625	0.289	0.458	0.355	0.346	0.354	0.131	2.403
C:3	0.625	0.29	0.459	0.355	0.347	0.355	0.131	2.403
C:5	0.625	0.289	0.458	0.355	0.346	0.354	0.131	2.403
C:7	0.625	0.289	0.458	0.355	0.346	0.354	0.131	2.403

Table 7: Results for Waveform using Fuzzy C Means

6.2 Dataset 2: Kr-vs-Kp

6.2.1 DBSCAN

These are the results after performing DBSCAN to the second dataset:

	Homogeneity	**Completeness**	**V_measure**	**ARI**	**AMI**	**Silhouette**
eps	3	10	3	5	3	10
min_samples	3	15	3	10	3	40
metric	euclidean	euclidean	euclidean	euclidean	euclidean	euclidean
algorithm	auto	auto	auto	auto	auto	auto
#Clusters	296	3	296	12	296	1
SCORE	**0.411**	**0.092**	**0.121**	**0.035**	**0.100**	**0.462**

Table 8: Results for Kr-vs-Kp using DBSCAN

However, despite the meticulous parameter tuning and rigorous analysis, the clustering outcomes remained challenging to interpret effectively. Each evaluation metric adjusts the clustering differently, indicating the difficulty in capturing distinct clusters within the dataset.

Given the nature of this dataset makes it more challenging for this kind of algorithm. It is a categorical dataset, converted during the preprocessing to a binary dataset. DBSCAN relies heavily on the notion of distance between data points. For numerical data, calculating distances is straightforward, but for binary features can only take on two values (0 or 1), making the notion of proximity less intuitive.

A small epsilon might result in very few points being considered neighbors, leading to fragmented and sparse clusters. Conversely, a large epsilon could result in most points being neighbors, potentially merging everything into a single cluster. This is what causes the differences in the number of clusters among the metrics.

6.2.2 BIRCH

Clustering using BIRCH in the second dataset. These are the results:

	Homogeneity	**Completeness**	**V_measure**	**ARI**	**AMI**	**Silhouette**
Threshold	0.8	0.8	0.8	0.8	0.8	0.9
Branching Factor	40	30	30	40	30	10
#Clusters	3	3	3	3	3	3
SCORE	**0.043**	**0.028**	**0.034**	**0.286**	**0.033**	**0.103**

Table 9: Results for BIRCH using Kr-vs-Kp

The algorithm consistently identified 3 clusters in the dataset. The clustering results displayed a moderate alignment with the true labels, evident from the Adjusted Rand Index score of 0.286. While the clustering wasn't entirely random, it fell short in terms of optimal purity, completeness, and distinctiveness. The clusters lacked homogeneity, meaning they weren't predominantly composed of a single class, and incompleteness, indicating they failed to encompass all items of specific classes.

6.2.3 K-Means + K-Modes

The K-means clustering results indicate that k=7 generally offers superior performance in terms of accuracy, homogeneity, completeness, and several other metrics, although k=2excels in silhouette score and k=5k=5 yields the lowest Davies-Bouldin Index, suggesting minimal cluster overlap.

group	accuracy	homogeneity	completeness	v_measure	adjusted_rand	adjusted_mutual_info	silhouette	davies
'k': 7	0.645	0.074	0.095	0.083	0.083	0.083	0.087	3.264
'k': 2	0.600	0.028	0.030	0.029	0.040	0.029	0.092	3.056
'k': 5	0.577	0.021	0.032	0.025	0.022	0.025	0.058	2.706
'k': 3	0.565	0.011	0.012	0.011	0.016	0.011	0.053	3.516

Table 10: Results for Kr-vs-Kp using K-means

In the K-modes clustering on categorical data, n_clusters=7 offers overall robust performance, but n_clusters=2 excels in intracluster similarity and exhibits the least cluster overlap.

Group	accuracy	homogeneity	completeness	v_measure	adjusted_rand	adjusted_mutual_info	silhouette	davies
'n_clusters': 7	0.595	0.025	0.027	0.026	0.036	0.026	0.053	4.113
'n_clusters': 2	0.590	0.026	0.034	0.029	0.031	0.029	0.104	2.657
'n_clusters': 5	0.577	0.016	0.019	0.018	0.023	0.017	0.082	3.134
'n_clusters': 3	0.522	0.000	1.000	0.000	0.000	0.000	N/A	N/A

Table 11: Results for Kr-vs-Kp using K-modes

6.2.4 K-Prototypes

The results obtained from applying K-Prototypes to the Kr-vs-Kp dataset we can observe that the choice of the number of clusters (K) plays a critical role in shaping the clustering quality metrics. The highest accuracy of 0.638 is achieved when K is set to 5, which may indicate a potential over-segmentation as it exceeds the actual number of clusters.

However, the homogeneity and completeness metrics remain relatively low across all K values, implying that the clusters may not be highly internally consistent or comprehensive. This suggests that the Kr-vs-Kp dataset may not exhibit well-defined clusters, making it challenging to uncover highly homogeneous and complete clusters.

The adjusted Rand Index and adjusted mutual (values ranging from 0.041 to 0.076) information metrics indicate that K-Prototypes achieves a reasonable level of agreement with the true two classes in the dataset. Silhouette with K=3 showing the best score, suggesting better cluster separation. The Davies-Bouldin score also highlights that K=3 results in clusters that are more compact and well-separated. This highlights the challenge of correctly identifying clusters when the true number is substantially lower.

group	accuracy	homogeneity	completeness	v_measure	adjusted_rand	adjusted_mutual_info	silhouette	davies
'k': 5	0.638	0.058	0.058	0.058	0.076	0.058	0.068	3.678
'k': 7	0.634	0.052	0.056	0.054	0.071	0.054	0.059	4.083
'k': 2	0.602	0.029	0.031	0.030	0.041	0.030	0.091	3.068
'k': 3	0.602	0.034	0.044	0.038	0.041	0.038	0.103	2.670

Table 12: Results for Kr-vs-Kp using K-prototype

6.2.5 Fuzzy C Means

The implementation of the FCM algorithm in this dataset with two distinct classes, yielded mixed results in terms of clustering performance. While the accuracy scores ranged from 0.522 to 0.544 (increasing at the same time that the number of clusters), showcasing a moderate degree of class separation, the homogeneity, completeness, and other performance indicators remained relatively low, suggesting limited agreement between the obtained clusters and true class labels.

The silhouette scores indicated the presence of overlapping clusters, particularly in the C:3 case. These findings underscore the importance of further refining the Fuzzy C Means parameters or considering alternative clustering techniques, like the previous ones implemented when dealing with datasets of this nature to enhance cluster quality and classification accuracy.

Group	**accuracy**	**homogeneity**	**completeness**	**v_measure**	**adjusted_rand**	**adjusted_mutual_info**	**silhouette**	**davies**
C:2	0.522	0	1	0	0	0		
C:3	0.529	0.002	0.015	0.004	0.001	0.003	-0.006	3.064
C:5	0.532	0.002	0.007	0.003	0.002	0.003	0.012	3.139
C:7	0.544	0.005	0.01	0.007	0.006	0.007	0.046	3.14

6.3 Dataset 3: Vowel

6.3.1 DBSCAN

This are the results of applying DBSCAN in the last dataset:

	Homogeneity	**Completeness**	**V_measure**	**ARI**	**AMI**	**Silhouette**
eps	3	3	3	3	3	5
min_samples	30	30	30	30	30	3
metric	euclidean	euclidean	euclidean	euclidean	euclidean	euclidean
algorithm	auto	auto	auto	auto	auto	auto
#Clusters	14	14	14	14	14	15
SCORE	**0.073**	**0.069**	**0.071**	**0.033**	**0.042**	**0.468**

Table 13: Results for Vowel using DBSCAN

This is the dataset with the lowest dimensional data and one of the best performances of DBSCAN. Nevertheless, it is not ideal.

The clusters seem to be well-separated (indicated by the high Silhouette score), meaning that the items within each cluster are more similar to each other than to those in other clusters. However, the clustering does not align well with the true class labels, as indicated by the low ARI and AMI scores. This suggests that the clusters do not accurately represent the underlying class structure of the data.

Overall, given the fact that this is a mixed dataset and DBSCAN does not perform correctly, this result is very promising.

6.3.2 BIRCH

Results after applying BIRCH to the third dataset:

	Homogeneity	**Completeness**	**V_measure**	**ARI**	**AMI**	**Silhouette**
Threshold	0.9	0.9	0.9	0.9	0.9	0.6
Branching Factor	30	30	30	30	30	50
#Clusters	3	3	3	3	3	3
SCORE	**0.162**	**0.362**	**0.224**	**0.105**	**0.219**	**0.155**

Table 14: Results for Vowel using BIRCH

In the analysis of the mixed dataset incorporating both numerical and categorical features, the BIRCH clustering algorithm consistently revealed three stable clusters. These results showcase the stability and reliability of BIRCH in partitioning the data. The clusters displayed improved internal cohesion and completeness, suggesting a more balanced representation of class members within the clusters.

6.3.3 K-Means + K-Modes

For the K-means clustering on the vowel dataset, k=15 presents the strongest overall performance in accuracy and most metrics, but k=11 boasts the best completeness and cluster distinction, while k=9 shines in intracluster similarity

group	accuracy	homogeneity	completeness	v_measure	adjusted_rand	adjusted_mutual_info	silhouette	davies
'k': 15	0.369	0.358	0.388	0.372	0.176	0.359	0.065	2.522
'k': 13	0.320	0.313	0.350	0.331	0.171	0.318	0.061	2.579
'k': 11	0.301	0.329	0.401	0.362	0.170	0.351	0.120	2.147
'k': 9	0.295	0.284	0.399	0.332	0.171	0.323	0.127	2.286
'k': 7	0.275	0.243	0.397	0.301	0.151	0.294	0.117	2.241
'k': 3	0.159	0.085	0.199	0.120	0.033	0.114	0.151	1.986

Table 15: Results for Vowel using Kmeans

6.3.4 K-Medoids or K-Prototypes

The results from the application of K-Prototypes to the Vowel dataset, a unique dataset characterized by both numerical and categorical features and featuring 11 distinct vowel classes, offer intriguing conclusions into the algorithm's performance. Surprisingly, the highest accuracy of 0.235 is attained with K set to 15. It appears that K-Prototypes segments the dataset into a larger number of clusters than the actual 11 vowel classes, suggesting a potentially more granular interpretation of the data.

The V-Measure, which balance homogeneity and completeness, remains modest. This indicates that the Vowel dataset may present challenges in forming clusters that accurately capture the nuances of the 11 vowel classes.

Moreover, the silhouette score raises concerns as it falls into negative territory for several K values, including the optimal K=15. This suggests that the clusters exhibit considerable overlap and are not well-separated. In summary, the results underscore the intricate nature of the Vowel dataset, the influence of the chosen K, and the challenges in achieving high-quality clustering results in a dataset with 11 distinct classes.

group	accuracy	homogeneity	completeness	v_measure	adjusted_rand	adjusted_mutual_info	silhouette	davies
'k': 15	0.235	0.179	0.238	0.204	0.090	0.192	-0.065	4.128
'k': 13	0.222	0.184	0.208	0.196	0.085	0.181	0.028	3.208
'k': 11	0.213	0.152	0.230	0.183	0.082	0.173	-0.064	2.649
'k': 9	0.183	0.121	0.154	0.136	0.055	0.123	0.081	2.978
'k': 7	0.165	0.096	0.146	0.116	0.055	0.106	0.151	3.129
'k': 3	0.091	0.000	1.000	0.000	0.000	0.000	N/A	N/A

Table 16: Results for Vowel using K-prototypes

6.3.5 Fuzzy C Means

We can see, that in this dataset with five real classes, regardless of the number of clusters, the accuracy scores remained relatively low, ranging from 0.131 to 0.140. The homogeneity, completeness, v-measure, and adjusted Rand Index values indicated only some agreement between the obtained clusters and the true class labels. Silhouette scores suggested limited separation between clusters.

Group	**accuracy**	**homogeneity**	**completeness**	**v_measure**	**adjusted_rand**	**adjusted_mutual_info**	**silhouette**	**davies**
C:2	0.131	0.016	0.057	0.025	0.011	0.022	0.152	2.295
C:3	0.14	0.023	0.075	0.036	0.014	0.029	0.055	2.625
C:5	0.135	0.023	0.075	0.035	0.011	0.028	0.069	2.177
C:7	0.134	0.02	0.071	0.032	0.012	0.025	0.081	2.271

Table 17: Results for Vowel using Fuzzy C Means

7 Conclusions

After applying these clustering algorithms to the different datasets, we reached some conclusions. In general, there is no perfect algorithm that will solve all of the different real-world scenarios that we are going to face, just by inputting the data and waiting for the results.

Therefore, it is important to know there is a variety of cluster algorithms, how are they designed, what are their specific drawbacks and "strengths", how they behave depending on the type of dataset we are analyzing, and also what is the specific goal that we want to achieve with this clustering process. Maybe for some particular cases, the accuracy of the algorithm is the most important performance metric, and in other cases is not as relevant. This way we can have a wider "toolkit" and understand the best option or "tools" that we can try to use in solving a particular problem.

The choice of the best clustering algorithm depends on the nature of the data. For datasets primarily consisting of categorical data, K-Modes or K-Prototypes should be favorable although this is not entirely consistent with our results. For numerical data, K-Means could be the best choice as shown in the results of Waveform dataset. When dealing with mixed data, BIRCH stands out for its adaptability and stability. The suitability of an algorithm is closely linked to the data's characteristics.

Now, specifically speaking we can get some conclusions for each of the algorithms we reviewed in this task:

In clustering high-dimensional data with DBSCAN, we faced challenges like poorly defined clusters and difficulty in determining the right cluster count. Notably, adjusting epsilon and minimum samples parameters was crucial for improving results, particularly evident in the last cluster. Surprisingly, changes in distance metric and algorithm choice did not significantly impact outcomes.

BIRCH consistently showcased stability and reliability in clustering across diverse datasets. Its adaptability was evident in capturing patterns and handling mixed features. Despite varying complexities, BIRCH generated balanced, cohesive clusters.

The selection of the best K value, especially in the context of K-Means and similar algorithms, was based on rigorous testing without using Silhouette or K-Means++, that would have provided valuable insights about it. The optimal K value was determined by evaluating the stability and cohesion of clusters. The specific K value that consistently generated the most balanced and cohesive clusters across diverse datasets was considered the best.

FCM allows for soft clustering, meaning it assigns data points to multiple clusters with varying degrees of membership, making it suitable for scenarios where data points may belong to more than one group. Also, is relatively robust to noise and outliers, as it considers the uncertainty associated with data point membership in clusters. However, it can be computationally expensive, particularly with large datasets or when the number of clusters is high, and it also requires specifying the number of clusters (C) a priori, which can be a challenging task in real-world applications.

In conclusion, our exploration of various clustering algorithms, including DBSCAN, BIRCH, k-Means, K-Modes, K-Prototypes, and Fuzzy Clustering, has revealed their varied strengths and suitability for different datasets. This report underscores the importance of understanding the specific characteristics of both the data and the algorithms in clustering analysis. Each algorithm offers unique insights, and the choice should be aligned with the dataset's features and objectives. Most of the times rigorous parameter tuning is vital for ensuring that clustering algorithms perform optimally and provide valuable information.

Bibliography

[Aprilliant 2023] APRILLIANT, Audhi: The k-prototype as Clustering Algorithm for Mixed Data Type (Categorical and Numerical). (2023), 1. – URL https://towardsdatascience.com/the-k-prototype-as-clustering-algorithm-for-mixed-data-type-categorical-and-numerical-fe7c50538ebb

[Breiman und Stone 1988] BREIMAN, L. ; STONE, C.J.: *Waveform Database Generator (Version 2)*. UCI Machine Learning Repository. 1988. – DOI: https://doi.org/10.24432/C56014

[Cannon u. a. 1986] CANNON, Robert L. ; DAVE, Jitendra V. ; BEZDEK, James C.: Efficient implementation of the fuzzy c-means clustering algorithms. In: *IEEE transactions on pattern analysis and machine intelligence* (1986), Nr. 2, S. 248–255

[García u. a. 2016] GARCÍA, Salvador ; RAMÍREZ-GALLEGO, Sergio ; LUENGO, Julián ; BENÍTEZ, José M. ; HERRERA, Francisco: Big data preprocessing: methods and prospects. In: *Big Data Analytics* 1 (2016), Nr. 1, S. 1–22

[Huang 1998] HUANG, Zhexue: Extensions to the k-means algorithm for clustering large data sets with categorical values. In: *Data mining and knowledge discovery* 2 (1998), Nr. 3, S. 283–304

[Palacio-Niño und Berzal 2019] PALACIO-NIÑO, Julio-Omar ; BERZAL, Fernando: Evaluation metrics for unsupervised learning algorithms. In: *arXiv preprint arXiv:1905.05667* (2019)

[Shapiro 1989] SHAPIRO, Alen: *Chess (King-Rook vs. King-Pawn)*. UCI Machine Learning Repository. 1989. – DOI: https://doi.org/10.24432/C5DK5C

[Steinbach u. a. 2005] STEINBACH, M ; KUMAR, V ; TAN, P: Cluster analysis: basic concepts and algorithms. In: *Introduction to data mining, 1st edn. Pearson Addison Wesley* (2005)