Cluster Categorical Data

Minghao Zhai, Mingze Li, Peng Xu, Hongbo Pang

November 2023

Clustering has always been a broad topic in the artificial intelligence area. One of the well-known algorithms when dealing with clustering is k-means which was invented by MacQueen 1967. However, data types are different, a serious issue has been brought up quickly: numerical data clustering methods may not be suitable for categorical data. This method is ineffective for categorical data as mean and distance in Euclidean space cannot be implemented directly to categorical data. K-modes, on the other hand, were invented by Huang(1997) who improved the k-means algorithm and proposed a k-modes algorithm in order to decrease the processing time of large categorical data. By replacing means with modes, using a frequency-based approach to define cluster centers, it is perfectly suitable for categorical variables and also employs a dissimilarity measure appropriate for categorical attributes, facilitating the clustering of non-numeric data. In this paper, we will talk about four different algorithms starting with a basic algorithm K-mode to more advanced version DBSCAN.

1 K-mode

1.1 Mathematical Background and Procedure of K-Mode

Suppose we have n observations and each observation contains m attributes. Now, we want to cluster these observations into k clusters. We first define our dissimilarity function:

$$d(X, Y) = \sum_{j=1}^{m} I(x_j, y_j)$$

Function d consumes two observations X and Y, and count the number of mismatches between two observations. Function I is the different indicator, the outcome is one when$x_j =$

y_j and zero when they are different.

Next, we can directly go for total cost functions:

$$\min_{} P(X, Q) = \sum_{r}^{k} \sum_{i=1}^{n} d(x_i, q_i)$$

The input Q represent centroids for each cluster $Q = [Q_1, .., Q_k]$. Centroid means it has the least mismatches within the cluster. Notice that q_i does not necessarily need to be an element of X. The full procedure of K-mode is:

Step 1: Randomly select $\mathbf{K}$ observations from X as centroids for each cluster

Step 2: We then calculate the mismatch between every data point and every centroid.

Step 3: Allocate rest n-k data into k clusters by their number of mismatches and use mode to calculate a new centroid for each cluster. Notice, at this stage, the new centroid generated may be different from any data point in the current cluster

Step 4: Calculate the mismatch between every data point and every centroid

Step 5: Allocate n data into k clusters by their number of mismatches and use mode to calculate a new centroid for each cluster.

Step 6: repeat steps 4 and 5 until the clusters and centroids become stable

1.2 Advantage and Disadvantages

The k-modes algorithm stands out for its efficiency, simplicity, flexibility, and utility across various applications. It efficiently handles large datasets with categorical attributes, making it particularly valuable in scenarios where traditional numerical algorithms falter.

However, the first step is to select k centroids randomly and then we start clustering data based on these centroids. This means that the algorithm although would become stable eventually, the outcome still significantly depends on our initial selections, especially when we happen to select outliers as centroid. This directly leads to the first disadvantage: **Not guarantee for global optimal solution**.

Another difficulty is that it is hard to determine the optimal number of clusters: $\mathbf{K}$ by the nature of unsupervised learning. In real life we are not aware of how many clusters

should have in the data set. Therefore, by choosing different $\mathbf{K}$, the performance of K-mode algorithm will be various accordingly. K-mode also cannot accept missing values. (All information about K-mode is based on the (Huang 1997))

2 PAM with Gower's distance

Intuitively, the difference between red and yellow is the same as between red and blue. But the difference between very satisfied and satisfied should be smaller than very satisfied to dissatisfied. Transform ordered categorical variables(like severity, age group, and satisfaction) to numbers, then apply Gower's distance would be a nice option. John C. Gower introduced it in his paper "A General Coefficient of Similarity and Some of Its Properties" on the Biometrics journal in 1971. Gower's distance can also handle missing values. Modified Gower's distance has a very strong capability to handle outliers and noises.

Partitioning Around Medoids (PAM) is a clustering algorithm that developed from k-means. To help understand, view it as "K-median". The aim is to partition the original dataset into k parts(clusters). The basic PAM algorithm is fully described in Chapter 2 of Kaufman and Rousseeuw(1990). PAM uses k real data points in the actual dataset as medoids for corresponding clusters, and it can accept dissimilarity matrixes as input.

2.1 Mathematical Background and Procedure

Assume there is a dataset X with n observations and m attributes, and the observations in X are $\{X_1, X_2, ..., X_n\}$. X will have n rows: $X_i = \{x_{i1},\, x_{i2},\, x_{i3},\, x_{i4}, ...,\, x_{im}\}$. Let the a^{th} column of X be $x_{.a}$. A dissimilarity matrix of X is a lower triangle matrix with $n-1$ rows and columns. The n^{th} column l^{th} row entry indicates the dissimilarity between observation X_n and X_{l+1}. Notice that here we need $n \leq l$.

For categorical variables, Gower's distance uses the same one as k-mode. For numeric variables, Gower's distance is a scaled to $0 \backsim 1$ absolute difference. The Gower's distance between two observation vectors is the weighted average of Gower's distance between each variable of those two vectors. If there is a missing value in any variable for either observation,

that variable is ignored. In formula,

$$d(x_{ia}, x_{ja}) = \begin{cases} f_a I(x_j, y_j) & \text{if categorical} \\ f_a \frac{|x_{ia}-x_{ja}|}{\max(x_{\cdot a})-\min(x_{\cdot a})} & \text{if numerical} \end{cases}$$

Define not missing indicator f_a, if at least one of x_{ia},x_{ja} is missing, $f_a = 0$, otherwise $f_a = 1$.

$$d(X_i, X_j) = \sum_{a=1}^{p} \frac{d(x_{ia}, x_{ja})}{\sum_{b=1}^{p} f_b}$$

Here i, j are integers in $[1, n], d(X_i, X_j) = d(X_j, X_i)$. 1 indicates complete different and 0 means completely same. If $\sum_{b=1}^{p} f_b = 0$, $d(X_i, X_j)$ would be a missing value. Repeat this method to fill the Gower's distance dissimilarity matrix.

Some modified Gower's distance may be also useful. Some variables may be considered more important while some variables may close to a "noise". We can set important variables to have a larger weight. Define weight vector $W = \{w_1, w_2, ..., w_m\}$, $0 \leq w_a \leq 1 \, \forall a$, $\sum_{a=q}^{m} w_a = 1$,

$$d_1(X_i, X_j) = \sum_{a=1}^{p} \frac{w_a d(x_{ia},\, x_{ja})}{\sum_{b=1}^{p} w_b f_b}.$$

When there are some outliers in numerical variables, the weight of that variable may become unexpectedly low. Consider use a smaller denominator in d, and set all differences larger than 1 to be 1. Similarly, We can also apply a window, and all differences smaller than that window become zero. Thus for numerical variables, modified Gower's distance has:

$$d_2(x_{ia}, x_{ja}) = \begin{cases} 0 & \text{if numerical and } |x_{ia} - x_{ja}| \leq window \\ f_a & \text{if numerical and } |x_{ia} - x_{ja}| \geq range \\ f_a \frac{|x_{ia}-x_{ja}|}{range} & \text{if numerical and neither in window or out of range} \end{cases}$$

For the choice of range, consider the interquartile range(IQR) and interdecile range(IDR) for every observation. Using the empirical distribution of the observation in that column, IQR ranges from 25% to 75% quantile and IDR uses 10% to 90%. The window could be the difference of the kth nearest neighbor of the observation in that column.

PAM can then use the dissimilarity matrix of X to cluster:

1. Randomly select k data points as initial medoids.

2. Assign all non-medoid points in X to the nearest medoid. The medoid and all points assigned to that medoid are in the same cluster.

3. For each cluster, select a new medoid. A medoid is a point with the smallest sum of the dissimilarity between all other points in that cluster.

4. Repeat steps 2 and 3 until the medoids and clusters are not changed.

2.2 Discussions

Gower's distance has some capability to handle mixed data, missing data, outliers, and different weighting. This makes it capable of dealing with a variety of applications. However, Gower's distance is very compute intensive, and a dissimilarity matrix requires a huge storage room when facing a large dataset($\frac{n(n-1)}{2}$ numbers). Gower's distance does not provide optimal numbers for ordered categorical variables and optimal weighting vector, although the natural number and the all 1 weighting vector are already efficient enough in some cases. Too many missing values may still cause problems as PAM can not process dissimilarity matrixes with missing values.

PAM shows better robustness than k-means when facing outliers and fewer uncertainties due to randomized initial points(Not solved completely). Outliers or misclassifications will affect PAM's cluster center less than the k-means' cluster center(Figure 1). Since PAM does not need to calculate dissimilarity, the amount of calculations is reduced significantly. Similar to k-mode, PAM cannot find optimal k either, and it is also based on ellipsoid distribution assumptions(may be partially solved by adding a weight vector in Gower's distance).

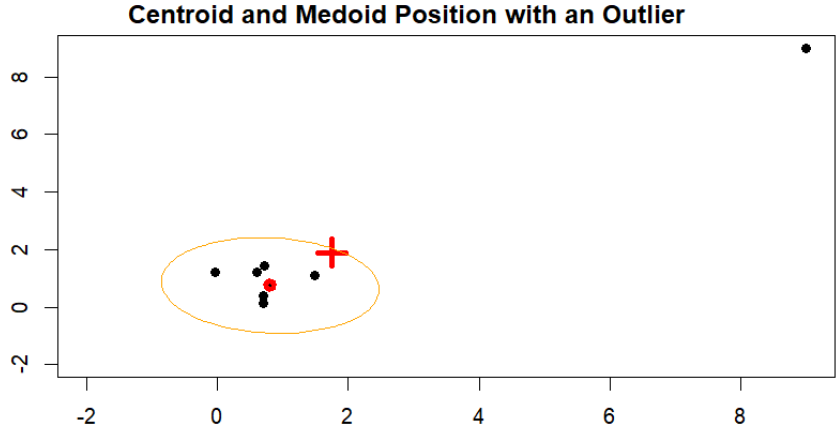

Figure 1: The centroid is almost on the edge of the cluster(red cross), while the medoid is not affected

3 Fuzzy C-means (FCM)

Pioneered by Dunn and further refined by Bezdek, FCM is a versatile clustering method that allows data points to belong to multiple cluster centers concurrently, enhancing clustering flexibility. FCM considers a weighted sum of distances and membership degrees in its objective function, enabling nuanced handling of complex datasets. Emerges a pivotal tool with fuzzy membership, showcasing adaptability to intricate datasets, allowing mixed input and missing values.

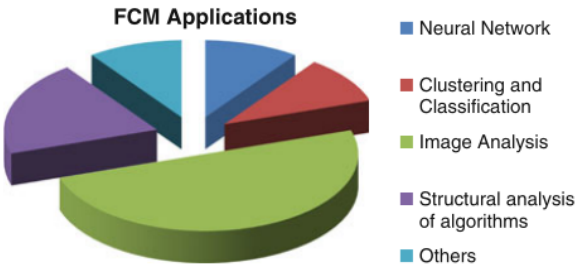

Figure 2: How FCM been applied in proportion.

3.1 algorithm principle

Fuzzy C-means is based on fuzzy set theory, partition original datatset into clusters, and also determine the membership degree of data points to these clusters. each data point belongs to all clusters with a certain degree of membership. Membership degree is a numerical value between 0 and 1, indicating the degree of a data point belonging to a certain cluster.

The sum of membership degrees is 1 for each observation. The aim is to minimize an objective function that measures the distance between each point and its cluster center. The total number of clusters is c, and the degree of membership of data point x_i to the cluster center v_i is u_{ij}. Note that v_i is not necessarily an observation in X. The fuzzification parameter is m which is decided externally, controls the degree of fuzziness of the membership, and is usually set to be a number between 1 and 2. when $m = 1$, this algorithm is K-mode. The dissimilarity measure is the same as k-modes between data points and the cluster center. The objective function is

$$J(U, V) = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{ij}^{m} I(x_j, y_j)$$

Note $\sum_{j=1}^{C} u_{ij} = 1$. Now need to optimize both U and V to minimize the objective function.

One method is to optimize partially and iteratively. These two elements are interdependent. The degree of membership could obtained according to the distance to centers. The position of the cluster centers can be determined by the mean of the data points weighted on their membership degrees, which means all data points matter. The updated formulas for the membership degree and cluster center are as follows:

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{I(x_j, v_j)}{I(x_j, v_j)} \right)^{\frac{2}{m-1}}}, \quad v_j = \frac{\sum_{i=1}^{N} u_{ij}^m x_i}{\sum_{i=1}^{N} u_{ij}^m}$$

When an observation is the same as the center, $u_{ij} = 1$ automatically, and membership for this observation to all other clusters is 0. Repeatedly updating the membership degrees and cluster centers, gradually adjusting the membership degree of each data point and the position of each cluster center, the FCM algorithm eventually meets some convergence conditions like the movement of the objective function is under a predetermined threshold. This process ensures that the clustering results can reflect the distribution and belonging of the data points as accurately as possible.

The execution process of the Fuzzy C-means algorithm is:

1. Randomly select c data points as initial centers.
2. Based on the current cluster centers, calculate the membership degree of each data point for each cluster.
3. Recalculates the centers of each cluster use the membership degree.
4. Repeat steps 2 and 3 until some conditions. Like interaction number or movement of objective function over the threshold.

3.2 Discussions

FCM introduces the groundbreaking concept of fuzzy membership, offering unparalleled flexibility in clustering analysis. This allows each data point to simultaneously belong to multiple cluster centers, making FCM adept for overlapping data distributions. FCM enables the distribution of data points across various cluster centers based on probability rather than strict categorization. This proves beneficial for analyzing intricate data with multiple features, accurately capturing diversity and complexity.

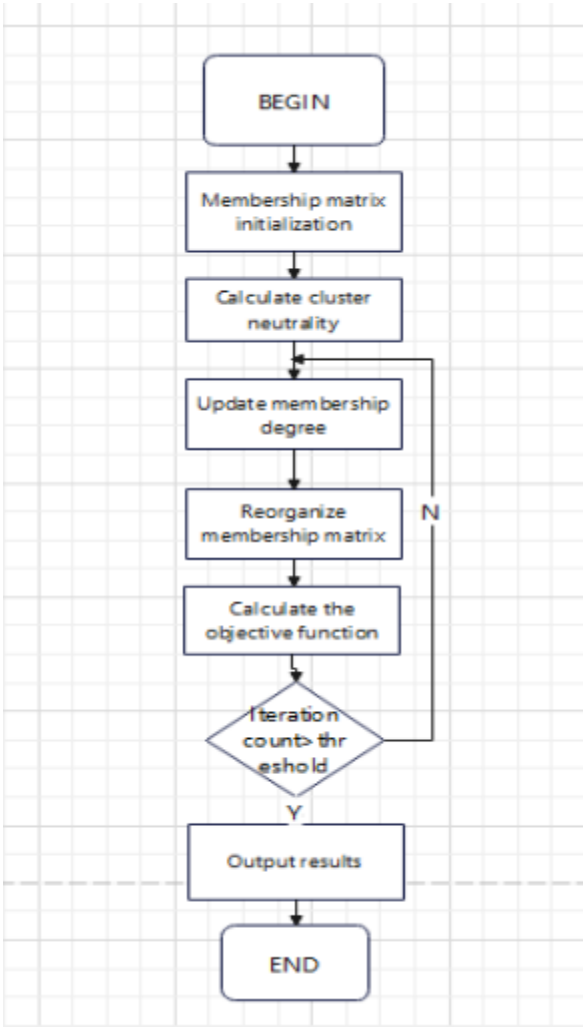

Figure 3: procedure of FCM

An additional significant advantage lies in FCM is dynamically adjusts cluster centers and memberships. This not only enhances overall clustering quality but also allows FCM to comprehensively express interrelationships between data points, particularly in scenarios challenging for traditional hard clustering methods.

In practical applications, FCM demonstrates wide applicability and effectiveness, excelling in image processing for tasks like segmentation, pattern recognition for multimodal data, and bioinformatics for analyzing gene expression. FCM's flexibility extends to integration with other technologies like neural networks and optimization algorithms, offering novel solutions for complex problems and expanding its application scope in data analysis and pattern recognition. Overall, FCM stands out as a flexible and effective algorithm with demonstrated success across diverse fields, underscoring its broad potential and practical value in data analysis and pattern recognition.

FCM also confronts challenges. The algorithm's need to calculate distances between

each data point and all cluster centers, results in heightened computational complexity, particularly in large datasets with numerous clusters. This poses a substantial burden on the algorithm's efficiency. Additionally, FCM's reliance on distance calculations makes it more susceptible to noise and outliers, which can distort clustering results and compromise the accuracy of the clustering center.

FCM also has similar shortages as previous algorithms. It mandates a predetermined number of clusters Sensitivity to the selection of initial clustering centers and may converge to a local optimal solution instead of the global optimal solution. Especially in cases of complex data distribution or multiple potential clustering structures.

The fuzziness parameter (m) significantly influences clustering outcomes, and improper selection may lead to results that are overly vague or clear. Addressing these challenges and limitations is crucial in practical applications to ensure that the FCM algorithm realizes its unique advantages in fuzzy clustering, delivering high-quality and reliable clustering results.

3.3 Application

For pure numerical data, consider using the caret package to handle missing values in the dataset, with a focus on employing the K-Nearest Neighbors Imputation method to fill in the missing values. Utilize the preProcess function, and set the method parameter to 'knnImpute,' indicating that the preProcess function will use the KNN algorithm to estimate missing values. Next, we apply this preprocessing step to the dataset with the predict function to implement the preprocessing step, resulting in the imputed dataset.

For mixed datasets, consider the fanny function with the "Manhattan" metric. It applies a modified fuzzy objective function the most similar to the one we proposed. The fanny function can accept some missing values. It can also accept a dissimilarity matrix or membership matrix. But since it involves the recalculation of centroids, inputting the original data usually works better.

Different from other cluster algorithms. FCM can also produce a membership matrix, that contains the degree of membership every observation to every cluster. The cluster that has the largest membership would become the hard cluster number of that observation. But if an observation has a membership of more than 0.1 to some cluster, then the probability of belonging to that cluster cannot be ignored.

4 DBSCAN

Density-based Spatial Clustering of Application (DBSCAN) is a popular clustering algorithm by Martin Ester et al. introduced in early 1996 (Ester, 1996). As its name suggests, the DBSCAN clustering algorithm is specially designed to find clusters of arbitrary shapes and sizes based on the density of the data point while effectively handling the noise in the data. Compared with other unsupervised learning clustering methods, such as K-means, DBSCAN does not require specifying the number of clusters in advance. It finds the clusters by categorizing the data points into core, border, and noise points, with the user-defined parameters Minimal Points and Epsilon. Figure 4 illustrates the different clusters identified by DBSCAN and K-Means.

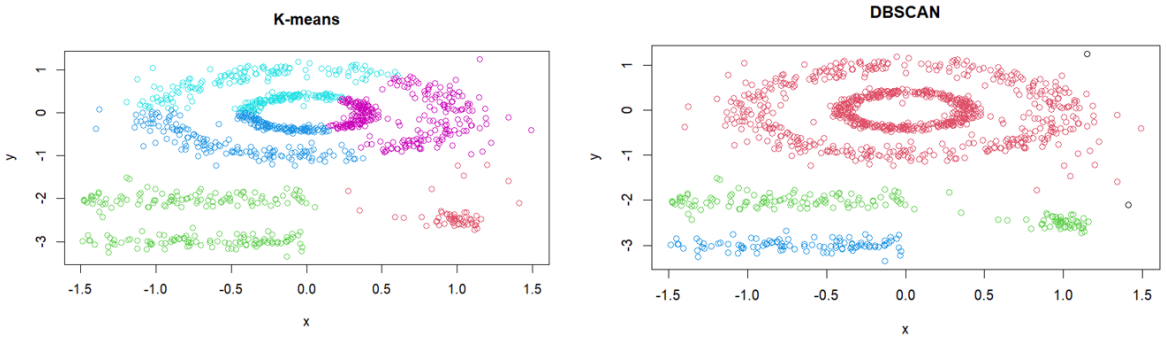

Figure 4: Difference between DBSCAN and K-Means

In DBSCAN, the core points have at least MinPts (minimal points) within a radius of eps (epsilon), and the border points are points within the eps radius of a core point. By fine-tuning these parameters, DBSCAN can effectively identify clusters as areas of high density separated by areas of low density. Moreover, one of the critical strengths of DBSCAN is its robustness to outliers, as it naturally separates noise from clusters. Points not belonging to any cluster due to sparse neighbors are directly categorized as noise. Compared to other methods like K-means, where every point is forced into a cluster, DBSCAN avoids misinterpreting the noise data due to direct noise identification. However, the quality of the DBSCAN clusters heavily depends on the selected eps and MinPts parameters. Inappropriate values can lead to poor clustering performance. The user is expected to observe the KNN distance plot and manually test throughout different eps to get the optimal clusters. Extensions like HDBSCAN and OPTICS were developed to overcome such limitations of DBSCAN, especially its sensitivity to the parameters and challenges in handling varying density clusters.

Algorithms

4.1 DBSCAN Algorithm

In the context of DBSCAN (Density-Based Spatial Clustering of Applications with Noise), the concepts of ”reachability” and ”connectivity” are fundamental to understanding how the algorithm identifies clusters within a dataset. These concepts are related to how the algorithm determines whether points are part of a cluster based on their density. Figure 5 (Sharma 2020) illustrates three critical states of two points in DBSCAN.

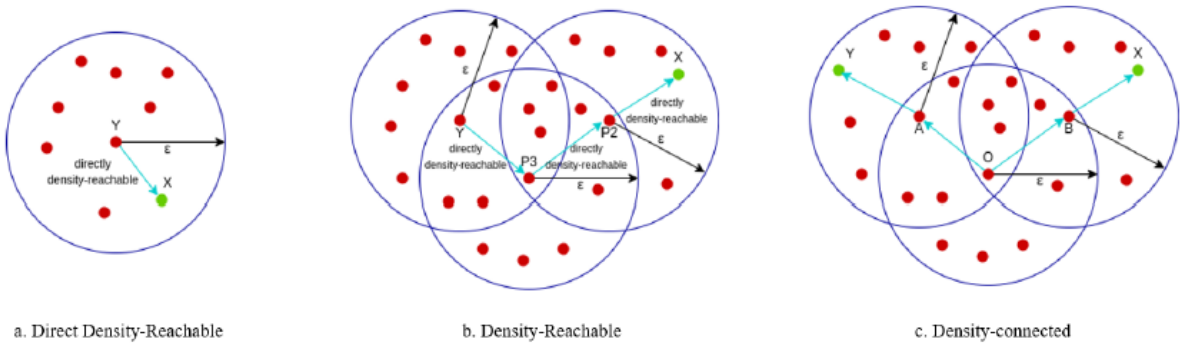

Figure 5: Points relation in DNSCAN

Reachability in DBSCAN refers to the minimum distance needed to consider one point directly density-reachable from another. In Figure 2a, X is directly density-reachable from Y w.r.t eps since the distance between two points is smaller than eps, and Y is the core point to its cluster. In Figure 2b, X is density-reachable from Y w.r.t eps and minPts because there is a china of points (P3, P2), and P2 is directly density-reachable from X. In Figure 2c, X is density-connected to Y w.r.t eps and minPts because there is a point PO and both X and Y are density-reachable from PO w.r.t eps and minPts.

Algorithm 1 (Ester, 1996) describes the essential ExpandCluster function of DBSCAN expanding a given cluster by iteratively adding density-reachable points. The function first determines if the seed point is a core point, requiring a minimum number of neighbours (MinPts) within a specified radius (eps). If the seed point qualifies as a core point, the function then iteratively adds all points within this radius to the cluster. This expansion process includes the immediate neighbours and any other points that are density-reachable from these neighbours, effectively chaining through the dataset. The function continues this process until no more points can be added, either due to them being already classi-

fied, not meeting the core point criteria, or falling outside the Eps neighbourhood. Through ExpandCluster, DBSCAN identifies and grows clusters based on local density, effectively differentiating between dense clusters and sparser noise in the dataset.

ALGORITHM 1. EXPANDCLUSTER IN DBSCAN [3]

```
ExpandCluster (SetOfPoints, Point, ClId, Eps, MinPts) : Boolean
  seeds : =SetOfPoints. regionQuery (Point, Eps ) ;
  IF seeds.size<MinPts THEN // no core point
    SetOfPoint. changeCl Id (Point, NOISE) ;
    RETURN False;
  ELSE // all points in seeds are density-reachable from Point
    SetOfpoints. changeClIds ( seeds, C1 Id);
    seeds .delete (Point);
    WHILE seeds <> Empty DO
      currentP := seeds.first()
      result := setofPoints.regionQuery(currentP, Eps)
      IF result.size >= MinPts THEN
        FOR i FROM 1 TO result.size DO
          resultP := result.get(i)
          IF resultP. ClId IN (UNCLASSIFIED, NOISE} THEN
            IF resultP.ClId = UNCLASSIFIED THEN
              seeds.append (resultP)
            END IF;
              SetOfPoints. changeClId ( resultP, ClId)
          END IF; // UNCLASSIFIED or NOISE
        END FOR;
      END IF;
    END WHILE;
END IF END;
```

5 Application

The famous soybean disease data(Michalski 1988) is said to be ”one of the standard test data sets used in the machine learning community”(Z Huang 1997). We use it because we can view all of the attributes as categorical variables, have the correct clustering answer, and it is public. Our shortened Soybean dataset involved 69 observations with 4 diseases. The no-missing value version has 46 observations. The full dataset has 10 charcoal-rot, diaporthe stem canker, and rhizoctonia root rot, and 39 diaporthe stem canker observations. No-missing value version only has 16 diaporthe stem canker observations and all others are the same.

The k-mode produced the perfect result most of the time. However, some unlucky starting points may produce a bad result(Figure 2). In PAM with original Gower’s distance, we set

date, plant-stand, precip, temp, hail, crop-hist, area-damaged, seed-tme, leaves, leaf-malf, and lodging to be natural number variables. It always produces perfect results when dealing with the no-missing version, while the full version always has one mismatch for phytophthora rot(Figure 2). We found that if set the weight ratio to less or equal to 0.2 for the last variable, and all others 1, then the result would always be perfect.

	1	2	3	4
charcoal-rot	0	5	5	0
diaporthe-stem-canker	0	0	0	10
phytophthora-rot	16	0	0	0
rhizoctonia-root-rot	10	0	0	0

	1	2	3	4
charcoal-rot	0	10	0	0
diaporthe-stem-canker	10	0	0	0
phytophthora-rot	0	0	1	38
rhizoctonia-root-rot	0	0	10	0

Figure 6: Left: An unlucky result from k-mode. Right: Always one mismatch for PAM with original Gower's distance

Now move to the result of FCM. Analysis of accuracy revealed that clusters 1, 2, and 3 exhibited perfect accuracy, while cluster 3 had a lower accuracy(16 out of 39 missed), indicating some degree of missed classifications. But when reviewing the membership degrees, we found that although the algorithm misclassified some observations into cluster 3, all of those have cluster 4 as the second largest memberships. In addition, these elements' membership in cluster 4 is larger than 0.1, meaning the probability of belonging to cluster 4 cannot be ignored. Other clusters, however, almost have no second-largest membership larger than 0.1. So we believe that this algorithm produced a reasonable result.

The scatter plots visually demonstrated that most classifications converged on the diagonal, but there were instances of misclassifications outside the diagonal, suggesting room for improvement in the FCM algorithm. Overall, the study provides insights into the classification performance of the FCM algorithm on the soybean dataset.

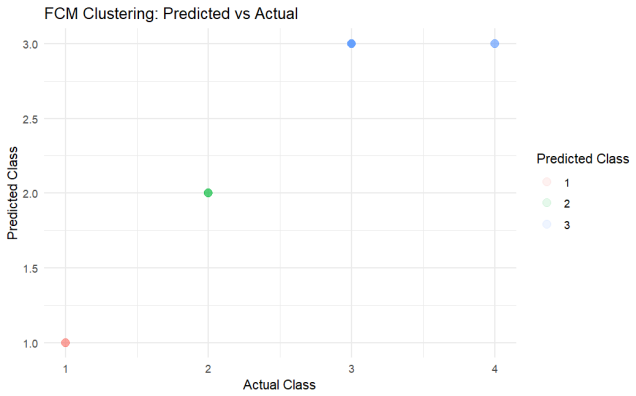

Figure 7

To conduct DBSCAN on the condensed soybean dataset, it's essential to estimate two critical parameters: eps and MinPts accurately. Due to DBSCAN's sensitivity to these parameters, their optimal selection is vital for efficient clustering. Initially, we set MinPts to three,

considering the dataset's sparsity, high dimensionality, and limited size. Post data normalization and outlier removal, we examined the KNN distance plot to determine the ideal eps. As Figure 8 illustrates, the plot's elbow point signifies a notable increment in the nearest neighbor distance, suggesting that points beyond this are likely part of different clusters.

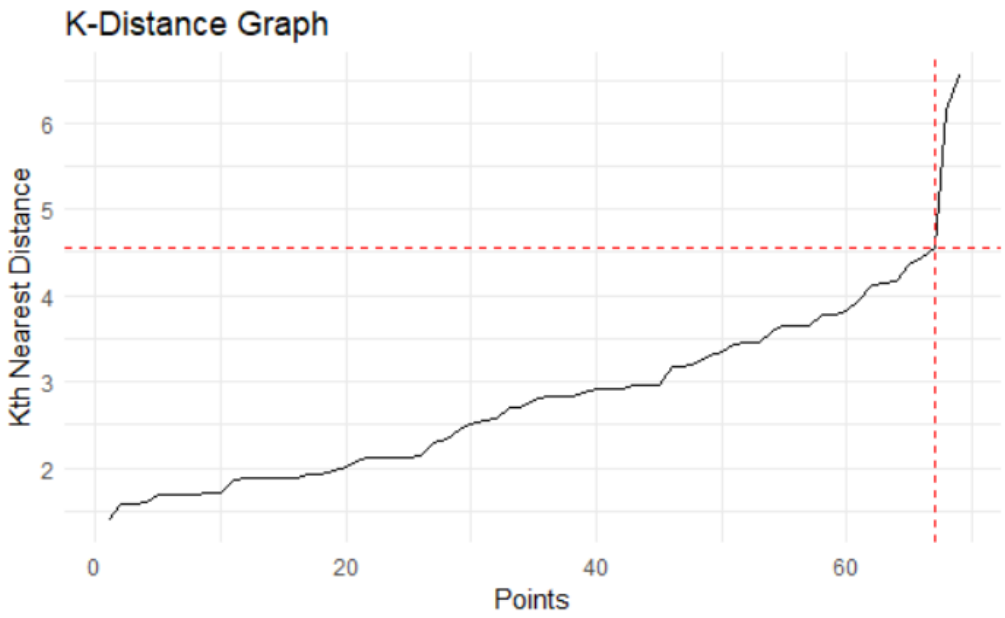

Figure 8: KNN Distance Plot for reduced soybean data

Upon configuring the eps and MinPts, we executed clustering using DBSCAN, HDBSCAN, and OPTICS. HDBSCAN enhances DBSCAN into a hierarchical model, replacing eps with a min cluster size parameter for more intuitive clustering. OPTICS builds on DBSCAN's foundation, introducing reachability distance for added flexibility. These methodologies aim to mitigate the impact of eps sensitivity. Figure 9 displays a comparative analysis of clustering outcomes from DBSCAN, HDBSCAN, and OPTICS.

```
                      0  1  2  3  4
  charcoal-rot        2  0  8  0  0
  diaporthe-stem-canker 2  8  0  0  0
  phytophthora-rot    10  0  0  0 29
  rhizoctonia-root-rot  2  0  0  8  0
[1] "DBSCAN"
```

```
                      0  1  2  3  4
  charcoal-rot        0 10  0  0  0
  diaporthe-stem-canker 0  0  0  0 10
  phytophthora-rot    11  0  0 28  0
  rhizoctonia-root-rot  2  0  8  0  0
[1] "HDBSCAN"
```

```
                      0  1  2  3  4
  charcoal-rot        2  0  0  0  8
  diaporthe-stem-canker 2  8  0  0  0
  phytophthora-rot    10  0 29  0  0
  rhizoctonia-root-rot  2  0  0  8  0
[1] "OPTICS "
```

Figure 9: Left: DBSCAN result. Center:HDBSCAN, Left: OPTICS

6 Reference

1. Abhishek S. 2020. How Does DBSCAN Clustering Work? Understanding the Basics. Retrieved fromhttps://www.analyticsvidhya.com/blog/2020/09/how-dbscan-clustering-works/

2. Bhat, Aruna. Kmedoids clustering using partitioning around medoids for performing face recognition. ***International Journal of Soft Computing, Mathematics and Control*** 3.3 (2014): 1-12. Available online: https://d1wqtxts1xzle7.cloudfront.net/55431775/paper_2-libre.pdf?1514977028=&response-content-disposition=inline%3B+filename%3DK_MEDOIDS_CL

3. D'Orazio, Marcello. Distances with mixed type variables some modified Gower's coefficients. ***arXiv preprint arXiv***, 2101.02481 (2021).
Available online: https://arxiv.org/ftp/arxiv/papers/2101/2101.02481.pdf

4. Gao Y, Wang Z, Xie J, et al.A new robust fuzzy c-means clustering method based on adaptive elastic distance[J]. ***Knowledge-Based Systems***,237(2022).
Available online: https://Improving_fuzzy_C-means_clustering_algorithm_based.pdf

5. Gower, John C. A general coefficient of similarity and some of its properties. ***Biometrics*** (1971): 857-871.
Available online:https://www.jstor.org/stable/pdf/2528823.pdf?casa_token¯ZyDFl9D5iYAAAAA

6. Huang, Z. A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining. ***Dmkd***, 3.8 (1997): 34-39
Available online: https://www.diag.uniroma1.it/ sassano/STAGE/Fast_Clustering.pdf

7. Kumar N, Kumar H. A fuzzy clustering technique for enhancing the convergence performance by using improved Fuzzy c-means and Particle Swarm Optimization algorithms[J]. ***Data & Knowledge Engineering***, 140 (2022).
Available online: https://doi.org/10.1016/j.datak.2022.102050.

8. Martin E, Hans-Peter K, Jörg S, and Xiaowei X. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. ***In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining***
Available online: https://dl.acm.org/doi/10.5555/3001460.3001507

9. Michalski,R.S. and Chilausky,R.L.. Soybean (Large). ***UCI Machine Learning Repository.*** (1988) Available online: https://doi.org/10.24432/C5JG6Z

10. Wu C, Zhang X.A self-learning iterative weighted possibilistic fuzzy c-means clustering via adaptive fusion[J]. ***Expert Systems with Applications***, 209(2022).
Available online: https://doi.org/10.1016/j.eswa.2022.118280.