Fuzzy Optim. Model. J. 2026; 7(1) : 92-112

Fuzzy Optimization and Modeling Journal (FOMJ)

https://doi.org/10.57647/fomj.2026.0701.06

Original Research

Enhancing Data Cleaning through the Extraction and Expansion of Relaxed Functional Dependencies

Mona Kardehi Moghadam[1], Seyyed Javad Seyyed Mahdavi Chabok[2,*], Reza Sheibani[2], Reza Ghaemi[3]

[1]*Department of Computer Engineering, Ne.C, Islamic Azad University, Neyshabur, Iran*
[2]*Department of Computer Engineering, Ma.C, Islamic Azad University, Mashhad, Iran*
[3]*Department of Computer Engineering, Qu.C, Islamic Azad University, Quchan, Iran*

[*]**Corresponding author:** sj.mahdavi@iau.ac.ir

Article History

Received:
28 October 2025

Revised:
26 February 2026

Accepted:
26 March 2026

Published in Issue:
30 March 2026

Abstract:

Data cleaning is pivotal for ensuring high-quality datasets in machine learning and data analytics, yet traditional methods often rely on manual thresholding, which is subjective and inefficient. This paper proposes RFD-Sugeno-KMeans-Clean, a data cleaning framework that integrates a Sugeno fuzzy inference system for automatic thresholding to extract Relaxed Functional Dependencies (RFDs). Then, by combining K-means++ clustering, it extracts the most effective pattern functional dependencies for data cleaning from complex datasets. In the second phase, a two-step SQL method is employed to detect and clean the data. The method also utilizes Principal Component Analysis (PCA), Identifying and Repairing Multiple Violations (Spline), Median, and Adaptive Neuro-Fuzzy Inference System (ANFIS) to enhance outlier detection across diverse datasets. The effectiveness of the proposed method has been validated using a dataset from the UCI database. The results demonstrate that the proposed method achieves a significant reduction in the number of data points and tuples, with a decrease of approximately 99.5% in most cases, while ensuring stability and reliability in responses. The proposed method's ability to extract optimal patterns in data cleaning demonstrates higher stability and reliability in responses compared to earlier methods, making it a viable solution for addressing data cleaning challenges in various domains.

Keywords: Data cleaning; Relaxed functional dependencies(RFD); K-means++ clustering; Sugeno fuzzy inference system; Automatic thresholding; Two-step SQL method; Multiple violations; Outlier detection; ANFIS

Cite this article: Kardehi Moghadam M, Seyyed Mahdavi Chabok SJ, Sheibani R, Ghaemi R. Enhancing Data Cleaning through the Extraction and Expansion of Relaxed Functional Dependencies. Fuzzy Optim. Model. J. 2026;7(1): 92-112. https://doi.org/fomj.2026.0701.06

1. Introduction

In today's data-driven world, the quality of information plays a critical role in the success of organizations across various sectors. As entities increasingly rely on data for decision-making, the presence of dirty, erroneous, and inconsistent data can lead to significant economic repercussions and irreparable financial losses. Such problems with data quality not only undermine credibility but also erode user trust, making it imperative to address these challenges effectively.

Data cleaning is recognized as a complex and resource-intensive process, often consuming between 30% and 80% of the total development time within a database project [1, 2]. This substantial investment underscores the need for more efficient data cleaning methodologies.

Data cleaning has been a vibrant research area, which can generally be classified into three main categories:

- Deep learning and machine learning-based methods [2, 3, 4, 5, 6, 7, 8, 9, 10, 11].
- Relaxed and Functional dependencies-based methods [1, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22,

23, 24, 25, 26, 27, 28].

- Fuzzy logic-based methods [29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43].

Below, in sections 1.1 to 1.2, we review recent advancements relevant to outlier detection, Relaxed Functional Dependencies (RFDs), and fuzzy systems, focusing on their strengths and limitations.

1.1 Background

- Deep learning and machine learning-based methods (Outlier Detection Methods)

Recent outlier detection methods leverage machine learning and statistical techniques. The Isolation Forest method, refined by Hariri et al., uses random partitioning for efficient anomaly detection but struggles with non-uniform noise patterns [2]. DBSCAN, revisited by Khan et al., identifies outliers as low-density points but requires careful parameter tuning, which limits its adaptability [3]. Deep learning-based approaches, such as those proposed by Pang et al., combine autoencoders with anomaly scoring but demand large training datasets and lack interpretability [4]. These methods highlight the need for adaptive, parameter-free solutions that can effectively address the limitations of existing approaches.

Yasin et al. examine the need for effective data cleaning processes to ensure the accuracy and reliability of datasets in machine learning and big data analytics. Traditional manual cleaning methods are often inefficient and error-prone. This study explores automated machine learning-based techniques that can improve data quality and reduce data preparation time. The complexity of big data and the variety of errors in the data can make the cleaning process challenging. Furthermore, there is a need to fine-tune algorithms to detect and correct different types of errors [5].

- Relaxed functional dependencies and functional dependencies-based methods

Caruccio et al. proposed an RFD discovery algorithm using lattice-based search, but it relies on fixed thresholds, reducing flexibility The introduction of the RFD detection algorithm called Difference Matrix and ε-threshold (DiMε) is also significant. Based on a network structure search space, DiMε demonstrates its impact on both tuples and rate comparisons. However, a persistent challenge identified in this study is the reliance on a user-defined threshold, which can hinder efficiency and introduce variability in results. Addressing this issue is crucial for enhancing the reliability of RFD detection methods [1]. Qahtan et al. introduced Pattern Functional Dependencies (PFDs) for data cleaning, improving on RFDs by incorporating contextual patterns, yet manual thresholding remains a bottleneck [12]. These studies underscore the need for automated thresholding in RFD extraction.

In addition, the exploration of Conditional Functional Dependencies (CFD) [14] compares CFDs with traditional FDs, highlighting the increased complexity in detecting constraint violations. This complexity necessitates the adoption of SQL-based techniques to identify inconsistencies as CFD violations, showcasing the evolving nature of data cleaning practices and the need for adaptable solutions.

Research [15] has explored the challenges of modifying incompatible databases with FD constraints, revealing that solvable FD patterns are characterized by left-hand-side chains. However, this improvement is accompanied by high computational complexity.

As an important step in improving the efficiency and effectiveness of functional dependency proofing in relational databases, Barlag et al. introduce the concept of Local Consistency. They provide a set of better and generalized rules for FDs to reduce the challenges of computational complexity and accuracy of dependency detection [28].

Significant advancements have been made in the area of Matching Dependencies (MDs), particularly with the introduction of the Hybrid Matching Dependency (HyMD) algorithm [22]. This algorithm efficiently discovers all minimal MDs, even in large datasets exceeding 3 GB. Such advancements contribute to the ongoing development of effective data cleaning methodologies that can handle the growing volume and complexity of data.

One of the most notable contributions in the realm of Functional Dependency (FD) is presented in research [26], which introduces Pattern Functional Dependency (PFD) for data cleaning. This approach enables the discovery of valid and useful PFDs from dirty data, facilitating the identification of data errors and improving overall data quality.

The concept of Local Consistency in this context means a partial or local examination of the rules and constraints of the data, rather than trying to prove dependencies globally in the entire database. This involves analyzing and testing the constraints on smaller subsets of the data, such as subsets of rows or columns. Instead of a complete and costly examination of the entire data, local consistency rules and algorithms are used to detect the presence or absence of functional dependencies more quickly and with less computational cost [28].

- Fuzzy logic in data cleaning

Fuzzy logic excels in handling uncertainty, making it a suitable approach for data cleaning. Sugeno fuzzy inference systems, as described by Li et al. [30] offer computational efficiency and adaptability through linear output functions [29]. Hudec applied fuzzy functional dependencies for knowledge discovery, utilizing linguistic interpretations to enhance interpretability [30]. Recent work by Khedher et al. [32] demonstrated the effectiveness of Takagi-Sugeno models for real-time process identification, highlighting their potential for adaptive systems. However, integrating Sugeno systems with Relative Functional Dependencies (RFDs) for data cleaning

https://doi.org/10.57647/fomj.2026.0701.06

remains a significant gap in current research.The Adaptive Neuro-Fuzzy Inference System (ANFIS), which combines fuzzy logic and neural networks, has been used for system modeling, but its potential for outlier detection remains underexplored [33].

Liu et al. proposed a framework called ADAR (Adaptive Dynamic Attribute and Rule) designed for managing rules and attributes in fuzzy inference systems on high-dimensional data. This framework dynamically simplifies complex fuzzy models by utilizing adaptive weighting mechanisms and automatic growth and pruning strategies, without compromising their performance or interpretability. The main challenges of this approach are managing and selecting important features in high-dimensional data, as well as maintaining the interpretability of the models [43].

Table 1 summarizes significant contributions in the field, highlighting methodologies and key findings from recent research articles focused on data cleaning based on functional dependencies.

Despite these advancements, current methods face several challenges:

1. Manual thresholding in RFD-based cleaning limits scalability and introduces uncertainty and instability in their responses.

2. Traditional outlier detection methods struggle with high computational complexity.

3. Fuzzy logic applications in data cleaning are limited to specific domains.

1.2 Research contributions

To address these challenges and extract optimal patterns, this paper utilizes the RFD-Sugeno-KMeans Combined Method to address these gaps:

- First, it introduces a Sugeno fuzzy system for automatic thresholding.

- Then, it integrates with K-means++ to improve clustering, aiming to minimize calculations and reduce the number of tuples during pattern extraction while improving the quality of results.

- Finally, it uses the Two-step SQL technique for the data cleaning process, providing a robust, scalable, and interpretable data cleaning solution. This approach aims to reduce dimensionality (the number of features), minimize tuples, and extract optimal clusters, ultimately providing responses with minimal error and improved response times. By implementing these methods, the objective is to enhance the accuracy of extracting optimal patterns through effective data cleaning, thereby achieving responses with reduced error rates and higher efficiency. This comprehensive approach not only addresses existing challenges but also sets the stage for future advancements in data quality improvement.

This paper is organized into several sections. Section 2 related work, while section 3 examines the proposed method in detail. Section 4 provides an illustrative example of the steps involved in the proposed method. In section 5, a comprehensive overview of the research procedure and the overall structure of the proposed method is presented. Section 6 analyzes the proposed method, and section 7 includes charts based on the results obtained, comparing and evaluating the proposed method against previous approaches. Finally, section **??** concludes with a final analysis and suggestions for future research directions, emphasizing the ongoing need for innovation in data cleaning methodologies.

2. Related work

2.1 Motivation and background

FDs have been instrumental in analyzing databases and assessing design quality. Over the past few decades, they have been employed for various purposes, including query optimization and data cleaning. However, while FDs effectively describe and illustrate data sources, they struggle when it comes to comparing large or multimedia datasets [44].

To overcome this limitation, we introduce Relaxed Functional Dependencies (RFDs). RFDs are a type of approximate functional dependency that can efficiently identify and retrieve frequently used and similar data

Table 1. Summary of recent articles on data cleaning based on functional dependencies.

Reference	Target	Suggested Method	Disadvantages	Advantages
[1]	Discover RFD	RFD Detection Algorithm	Reliance on user-defined thresholds	Effective RFD detection with DiMε
[12]	Error Detection	Patterns Functional Dependency	Potential inaccuracies	Valid PFDs for detecting errors
[15]	Data Cleaning	Conditional Functional Dependencies	Increased complexity in detection	Introduction of CFDs for improved data cleaning
[16]	Database Modification	Modifying Incompatible Databases	Complexity of FD patterns	Insights into computational complexity
[23]	Discover minimal MDs	Hybrid Matching Dependency Algorithm	Limited to minimal MDs	Efficient processing of large datasets

https://doi.org/10.57647/fomj.2026.0701.06

within extensive or multimedia datasets. They achieve this by utilizing defined features and thresholds, thereby enhancing the data cleaning process.

In this section, we will first review the definition of FDs, which will help us establish a comprehensive definition for RFDs.

2.2 Formal definition of functional dependencies (FDs)

Consider a relational database schema R defined over a set of attributes (R), which is derived from the union of the attributes of the relation schemas that comprise R. Given an instance r of R, an attribute $A \in \text{attr}(R)$, and a tuple $t \in r$, we denote the projection of t onto A as $t[A]$. Similarly, for a set X of attributes in (R), $t[X]$ denotes the projection of t onto X [1].

An FD over R is expressed as $X \to Y$ (where X implies Y), with $X, Y \subseteq \text{attr}(R)$. This means that for a given instance r of R, the FD $X \to Y$ holds in r if and only if, for every pair of tuples (t_1, t_2) in r, whenever $t_1[X] = t_2[X]$, it follows that $t_1[Y] = t_2[Y]$. Here, X and Y represent the Left-Hand Side (LHS) and Right-Hand Side (RHS) of the FD, respectively.

The constraint defined by an FD can also be represented using the concept of tuple partitioning. Specifically, given a set of attributes X, the tuple partition of an instance r on X, denoted as π_X (r), is a collection of disjoint equivalence classes of tuples from r, each containing a unique combination of values for the attribute set X. The union of these equivalence classes corresponds to the relation r.

The equivalence class of a tuple $t \in r$ with respect to X is denoted as $[t]_X$ and is defined as $[t]_X = \{u \in r | t[A] = u[A]$ for all $A \in X\}$. A partition π is said to refine another partition π' if every equivalence class in π is a subset of some equivalence class in π'. It can be shown that an FD $X \to Y$ holds on r if and only if π_X (r) refines π_Y (r).

2.3 Formal definition of relaxed functional dependencies (RFDs)

In the definition of FDs, the projections of two tuples over a subset of attributes are compared using the equality function. This comparison is one of the two aspects that have been modified to define RFDs. RFDs allow for tuple comparisons based on constraints rather than strict equality.

A constraint can be understood as a predicate that evaluates whether the distance or similarity between two values of an attribute falls within predefined intervals. Informally, an RFD can be described as a functional dependency that relaxes the traditional tuple comparison by incorporating constraints on the distance or similarity between attribute values. Additionally, RFDs may also relax the extent of the dependency by employing a coverage measure, which indicates the minimum number or percentage of tuples on which the RFD must hold.

Furthermore, RFDs can include conditions that restrict their applicability domain, allowing for more flexible data relationships. This flexibility makes RFDs particularly useful in scenarios where exact matches are not feasible or desirable, such as in large or multimedia datasets.

2.4 Mathematical formulation of RFDs

Definition 1: RFD Consider a relational database schema R and a relation schema $R = (A_1, \ldots, A_m)$ of R. An RFD ϕ on R is denoted by equations (1),(2) [1]:

$$\mathbb{D}_c : X_{\Phi_1} \xrightarrow{\Psi \leq \varepsilon} X_{\Phi_2} \tag{1}$$

where:

$$\mathbb{D}_c = \{t \in \text{dom}(R) \mid \bigwedge_{i=1}^{m} c_i(t[A_i])\} \tag{2}$$

Here, $c = (c_1 \cdot c_2 \cdot c_3 \cdot \ldots \cdot c_m)$ consists of predicates on dom (A_i) that filter the tuples on which φ applies;

• **Attributes:**

$X = (X_1 \cdot X_2 \cdot X_3 \cdot \ldots X_h)$ and $Y = (Y_1 \cdot Y_2 \cdot Y_3 \cdot \ldots Y_k)$ where $X \cdot Y \subseteq \text{attr}(R)$ and $X \cap Y = \phi$.

• **Constraints:**

is denoted by Eq. (3):

$$\Phi_1 = \bigwedge_{X_i \in X} \varphi_i[X_i](\Phi_2 = \bigwedge_{Y_j \in Y} \varphi_j[Y_j].resp) \tag{3}$$

where $\phi_i(\phi_j \cdot resp)$ is a conjunction of predicates on $X_i(Y_j, resp)$ with $i = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot h$ (and $j = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot k, resp \cdot$).

For any pair of tuples $(t_1 \cdot t_2) \in$ dom (R), the constraint $\Phi_1(\Phi_2 \cdot resp)$ is true if $t_1[X_i]$ and $t_2[X_i](t_1[Y_i]$ and $t_2[Y_i] \cdot resp \cdot)$ satisfy the constraint $\phi_i(\phi_j \cdot resp \cdot)$ for all $\forall i \in [1 \cdot h]$ (and $j \in [1 \cdot k], resp \cdot$) [1].

• **Coverage measure:**

Ψ is a coverage measure defined on dom (R), quantifying the number of tuples violating or satisfying ϕ. It can be defined as a function: Ψ :dom $(X) \times dom(Y) \to \mathbb{R}^+$ where dom (X) is the Cartesian product of the domains of attributes composing X [1].

• **Threshold:**

ε is a threshold indicating the upper bound (or lower bound in the case where the comparison operator is $\geq$) for the result of the coverage measure.

Satisfaction Condition

Given $r \subseteq D_c$ as a relation instance on R, r satisfies the RFD ϕ, denoted by $r| = \phi$, if and only if:

$\forall t_1 \cdot t_2 \in r$, if Φ_1 is true, then almost always Φ_2 is true.

Here, "almost always" is expressed by the constraint $\Psi \leq \varepsilon$.

This formalization provides a flexible framework for modeling approximate dependencies in real-world datasets, enabling the detection of near-duplicates, inconsistencies, or semantically similar records in large and heterogeneous databases [1].

3. Proposed method

The proposed RFD-Sugeno-K-means framework combines K-means++ clustering, the Two-step SQL technique, and a Sugeno fuzzy inference system to perform

https://doi.org/10.57647/fomj.2026.0701.06

data cleaning via automatic thresholding for RFD extraction. The method consists of key steps: data preprocessing, clustering, fuzzy-based thresholding, and outlier removal. Figure 1 illustrates the flowchart of the proposed method.

The flowchart of the proposed method is outlined as follows:

• **Data preprocessing:**
rocessing is a critical initial step in any data analysis project, serving as the foundation for ensuring the quality and reliability of the results. This phase involves a systematic approach to collecting, cleaning, and organizing data from various sources to enhance its robustness and applicability. The preprocessing steps encompass several key activities, including feature selection, tuple comparison, data cleaning, and outlier handling. Overall, this structured preprocessing approach aims to enhance data integrity and prepare a clean, reliable dataset for subsequent dependency discovery and pattern extraction, ultimately contributing to more accurate and meaningful insights.

To enhance the clarity, transparency, and reproducibility of the proposed RFD-Sugeno-KMeans framework, Table 2 provides a unified summary of all mathe-

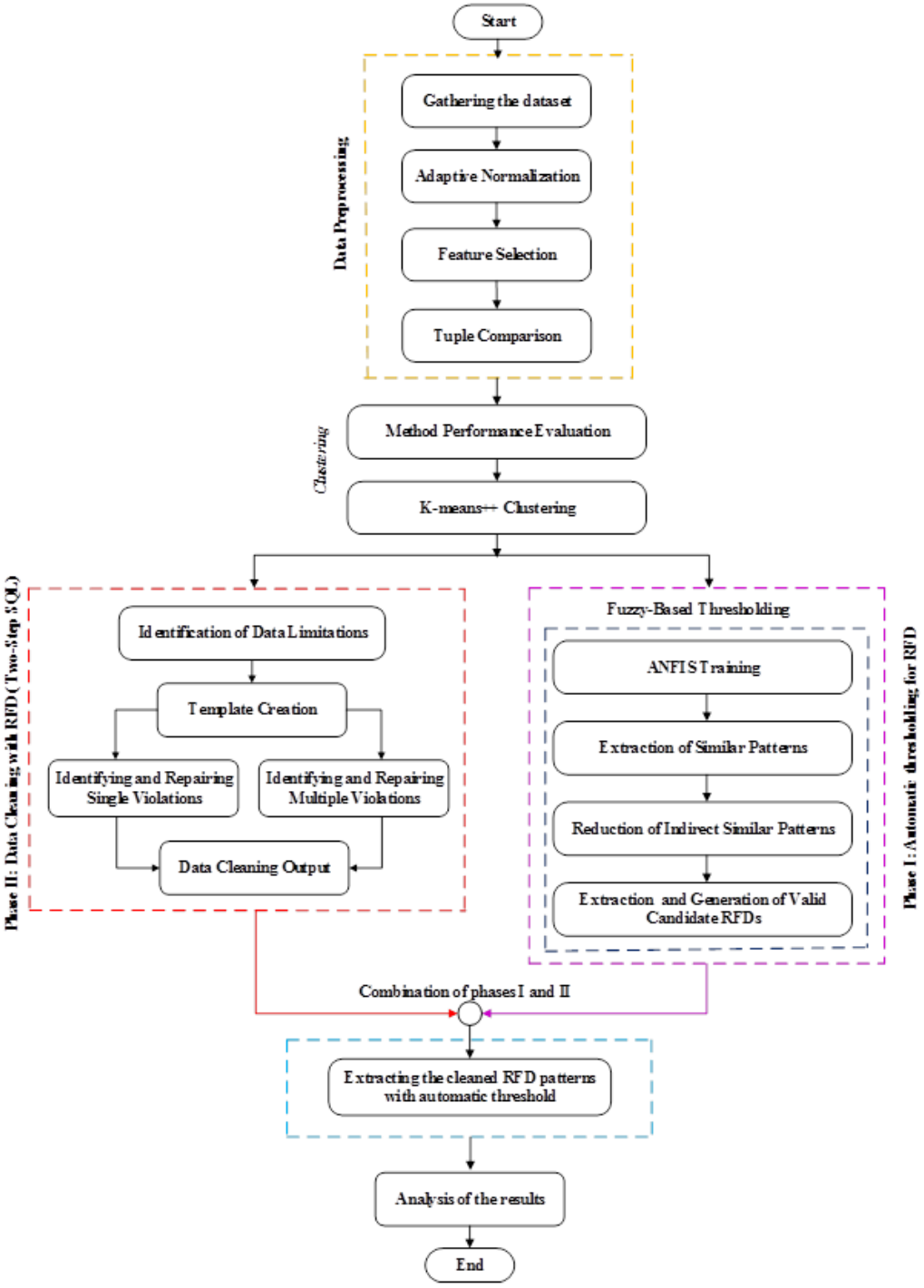

Figure 1. General steps of the proposed method.

https://doi.org/10.57647/fomj.2026.0701.06

Table 2. Mathematical and algorithmic parameter settings used in the proposed method.

Category	Symbol / Parameter	Description	Value / Setting	How obtained / Role
Preprocessing	Missing values	Missing value handling	Nearest neighbor imputation	Prevents tuple removal
	MAD	Median absolute deviation	Equation (5)	Outlier detection metric
	Outliers	Outlier repair method	Median + Spline interpolation	Data cleaning
Normalization	-	Feature scaling	Min-Max [0,1]	Preprocessing step
Dimensionality Reduction	PCA	Variance retained	99%	Feature selection threshold
Distance Metric	$\Delta(A)$	Distance function for attribute A	Chebyshev distance	Fixed metric for comparisons
RFD Structure	C_X, C_Y	LHS and RHS constraints	Automatically generated	RFD extraction
	$MX\Delta$	Difference matrix	-	Constructed from $\Delta(A)$
	$SX(t)$	Similar pattern of tuple t	-	Defined by threshold θ
	I_X	Similar pattern subsets	-	Pruned similarity sets
Clustering	K	Number of clusters	Calinski–Harabasz optimum	Cluster validation
	Initialization	k-means initialization	K-means++	Improves stability
	Replicates	Number of runs	5	Avoids local minima
Fuzzy Inference	Θ	Automatic RFD threshold	95^{th} percentile of distances	Sugeno FIS + ANFIS
	E	Similarity tolerance	Data-driven	Automatic thresholding
RFD Measures	Γ	Coverage measure	Relative frequency	RFD satisfaction criterion
	T	Coverage threshold	Dataset-dependent	Experimentally tuned
Statistics	σ_i	Std. deviation of cluster i	Cluster-dependent	Computed from cluster data
	mean_cheb_i	Mean Chebyshev distance in cluster i	Cluster-dependent	Derived from distance matrix
Evaluation	Runs	Number of independent runs	15	Stability assessment

matical, statistical, dependency-related, and algorithmic parameters used throughout the method. Specifically, this table reports the definitions, evaluation settings, and functional roles of the RFD-related parameters (e.g., $\Delta(A), C_X, C_Y, MX\Delta, SX(t), I_X, \theta, \varepsilon, \gamma$, and τ), cluster-level statistical descriptors (σ_i and mean_cheb_i), as well as the preprocessing, clustering, fuzzy inference, and evaluation settings employed in the experimental study. This unified presentation ensures that both the theoretical formulation and the practical implementation of the proposed method can be precisely understood and fully reproduced.

▷ **Gathering the dataset:** The study begins with data collection from multiple sources, including benchmark datasets such as Iris, MS, thyroid, breast cancer, diabetes, hepatitis, and lymphography, all sourced from the UCI Machine Learning Repository. This diversity not only enriches the analysis but also provides a comprehensive view of the data landscape [10, 11, 15, 25].

▷ **Adaptive normalization:** This stage involves cleaning the data by removing outliers and normalizing the data using the Adaptive Normalization method: For each feature (f), skewness is computed. If ($|skewness(f)| > 1$), z-score normalization is applied: $x'_f = (x_f - \mu_f)/\sigma_f$ where μ_f is the mean and σ_f is the standard deviation. Otherwise, Min-Max normalization is used, which is denoted by Eq. (4):

$$Z_i = \frac{X_i - \min(x_i)}{\max(x_i) - \min(x_i)} \tag{4}$$

This normalization ensures that all features contribute equally to the analysis by scaling them to a common range.

▷ **Feature selection:** PCA is employed to reduce dimensionality while preserving 95% of the total variance, transforming the original dataset to

https://doi.org/10.57647/fomj.2026.0701.06

$X_{proc} = \mathbb{R}^{n \times k}$ where $k \leq m$. This reduction mitigates noise, alleviates the curse of dimensionality, and focuses computational effort on the most informative feature directions. The 95th percentile threshold for Chebyshev distances is grounded in statistical hypothesis testing, corresponding to a significance level of $\alpha = 0.05$. This establishes a controlled Type I error rate where only 5% of in-distribution points are expected to be falsely identified as outliers under the null hypothesis. This convention aligns with standards in anomaly detection literature and ensures the threshold represents the minimum distance value that, with probability $(1-\alpha)$, is exceeded by no more than 5% of the nominal data distribution.

▷ **Tuple comparison:** A distance matrix is created to facilitate comparisons between data tuples. This matrix helps in identifying similarities and differences among data points, which is crucial for subsequent analysis.

• **Performance evaluation and clustering:**

▷ **Method performance evaluation:** Various methods are evaluated, with a focus on the RFD-Sugeno-Kmeans method, which has shown superior performance in extracting optimal patterns from different datasets.

▷ **K-means++ clustering:** K-means++ is used to partition X_{proc} into (k) clusters, where (k) is set based on the number of classes (e.g., 3 for Iris). The algorithm employs:

1. City block Distance: Robust to outliers compared to Euclidean distance.
2. Silhouette Optimization: The best clustering is selected from 10 runs based on the highest average Silhouette score, ensuring robust partitioning.

Phase I: Automatic thresholding for RFD

In this phase, the emphasis shifts to pattern extraction and optimization. This phase builds upon Pre-processed database, utilizing advanced methods to determine optimal thresholds for pattern recognition. The goal is to enhance the model's ability to identify relevant patterns while maintaining stability and accuracy in results.

▷ **Fuzzy-based thresholding:** Sugeno fuzzy inference is employed to generate an automatic threshold, enhancing decision-making capabilities based on fuzzy logic principles. in such a way that A Sugeno fuzzy inference system, trained via ANFIS, determines the threshold θ for RFD extraction:

▷ **ANFIS training**

1. **Input features:** For each cluster (i), compute:
   o σ_i: Mean standard deviation of features.
   o mean_cheb$_i$: Mean Chebyshev distance to the cluster center.
2. **ANFIS:** 80% of samples are used for training, 20% for validation, with 5 FCM clusters and 100 epochs. The hybrid optimization method (least squares and backpropagation) ensures convergence.
3. **Output:** Threshold θ_i, set as the 95th percentile of Chebyshev distances.

▷ **Extraction of similar patterns:** Automatic thresholds are applied to extract features, determining the degree of similarity among data points and validating candidate RFDs.

▷ **Reduction of indirect similar patterns:** The subset of similar patterns is refined by eliminating redundant entries, which streamlines the dataset and focuses on meaningful comparisons.

▷ **Extraction and generation of valid candidate RFDs:** For each sample (x_j) in cluster (i), if the Chebyshev distance exceeds θ_j, it is marked as an outlier. This process captures approximate dependencies while tolerating minor variations. The output of this phase is the generation of valid candidate RFDs, representing the most relevant patterns identified during the analysis.

Phase II: Data cleaning with RFD (two-step SQL)

In this phase, the focus is on preparing the dataset for analysis by addressing data quality issues. This involves identifying and rectifying defects, removing irrelevant features, removing samples identified as outliers are and ensuring that the data is clean and reliable for further processing which finally results in the cleaned data set (X_{clean}). The steps in this phase are designed to enhance the integrity of the data and results in accurate results.

• **Identification of data limitations:** This step aims to uncover any defects, Outlier or limitations in the data that could affect the analysis. Techniques like PCA, Spline, and Median methods are used to pinpoint these issues effectively.

▷ **Template creation:** A structured template is developed using a two-step SQL approach, which organizes the data in a way that facilitates easier analysis and processing.

▷ **Identifying and repairing single violations:** Here, the focus is on eliminating insignificant features through PCA, allowing for the extraction of important variables. This is visualized using a pebble chart to represent the impact and accuracy of features.

▷ **Identifying and repairing multiple violations:** This step addresses outliers in the data using Spline and Median methods, ensuring that the dataset remains robust and reliable.

▷ **Data cleaning output:** The results from this phase produce a cleaned and refined dataset, which results in generation the cleaned data set (X_{clean}).

https://doi.org/10.57647/fomj.2026.0701.06

Combination of phases I and II:

▷ **Extracting the cleaned RFD patterns with automatic threshold:** The final output is a combination of phases 1 and 2, which results in is the generation of cleaned up valid candidate RFDs.

The proposed method employs a structured approach that begins with the extraction of valid patterns and culminates with data cleaning, ensuring data integrity and optimal results. the pattern extraction phase utilizes advanced techniques, including automatic thresholds, to optimize pattern recognition. An accuracy range of 99% is maintained during feature selection, ensuring that only the most relevant features are utilized. Following this, the data cleaning phase focuses on identifying and rectifying defects, irrelevant features, and outliers to enhance the dataset's quality. Eight datasets from the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets) have been carefully chosen, as shown in Table 3, considering their balanced and imbalanced nature as well as class distribution equality. This comprehensive methodology not only strengthens data integrity but also facilitates thorough analysis, making it a significant contribution to the field of data analysis.

The proposed method emphasizes the importance of various variables, such as the number of clusters, features, tuples, and the percentage reduction in tuples, for effective pattern extraction. Once the data is collected, selected properties are organized into an X vector, represented as $X = \{X_1, X_2, X_3, \ldots, X_n\}$ signifies a distinct property. Due to significant fluctuations in these attributes and their lack of proportionality, preprocessing operations are essential Initially, to optimize the method for extracting patterns and ensuring stability and high reliability in results, the outcomes are evaluated using three combined methods across eight different scenarios. In the second phase, the data cleaning process utilizes a two-step SQL method, as outlined in Table 4. This structured approach ensures that the analysis is both thorough and effective, reinforcing the overall integrity of the findings.

The pseudocode of the proposed method is presented as the Algorithm1.

4. Providing an example to demonstrate the steps of the proposed method:

In this section, by presenting an example, the phases of the proposed method have been studied. Table 5 indicates the dataset used in this example.

4.1 Data cleaning with the two-step SQL method

In this phase, twelve methods were evaluated for finding, identifying, and correcting outliers. Notably, 70% of the reviewed articles reported the best results using the spline and median methods. Therefore, this research focuses on these two methods for addressing multiple violations (deletion or correction of outliers).

- **Median:** In the median method, outliers exceeding three times the scaled Median Absolute Deviation (MAD) are replaced using equation (5):

$$MAD = \frac{-1}{\sqrt{2} \times erfcinv(\frac{3}{2})} * \\ median(|A - median(A)|) \tag{5}$$

- **Spline:** The Spline method employs piecewise cubic spline interpolation to generate new points within a known set, effectively filling in gaps and correcting outliers in the dataset.

Table 3. Specifications of the studied datasets from the UCI site.

Nature of data	The name of the data set	Number of records	Number of features	The number of class categories	Volume (kb)
Balance	Iris	151	4	3	13
	Merge	400	34	2	59
	Thyroid	7200	21	3	696
Imbalanced	Breast-cancer	700	10	2	42
	Heart	270	13	2	24
	Diabetes	768	8	2	50
	Hepatitis	153	19	2	54
	Lymphography	149	18	4	18

Table 4. Pre-processing methods used for data-cleaning.

Pre-processing Methods	Steps
Two-step SQL method	Identification and repair of unit defect using PCA method
	Identification and repair of multiple defects using Spline and Median methods
Normalization	Min Max

https://doi.org/10.57647/fomj.2026.0701.06

Algorithm 1. Proposed RFD-based Data Cleaning Method.

Input: Raw dataset D Output: Cleaned dataset X_{clean}, Valid candidate RFDs

// Phase 1: Data Preprocessing
1. Load datasets from UCI Repository
2. for each feature f in D do:
    if $|skewness(f)| > 1$:
        apply z-score normalization
    else:
        apply Min-Max normalization
3. Apply PCA to retain 95% variance $\rightarrow D_{reduced}$
4. Construct distance matrix for tuple comparison

// Phase 2: Automatic RFD Thresholding
5. Perform K-means++ clustering:
   - Use city block distance
   - Select optimal clusters via silhouette score (10 runs)
6. Train ANFIS fuzzy system:
   - Inputs: σ_i (mean std), mean_cheb$_i$ (mean Chebyshev distance)
   - Output: $\theta_i \leftarrow 95^{\text{th}}$ percentile of Chebyshev distances
7. Extract similar patterns using θ_i thresholds
8. Generate valid candidate RFDs

// Phase 3: Data Cleaning
9. Identify data limitations using PCA, Spline, Median methods
10. Apply two-step SQL approach:
   - Repair single violations (PCA feature elimination)
   - Repair multiple violations (Spline/Median outlier handling)
11. Produce cleaned dataset X_{clean}

// Final Integration
12. Combine Phase 2 and 3 outputs
13. return X_{clean} and valid candidate RFDs

After addressing outliers, data normalization is performed using the max-min normalization method, which confines the data within a specific range, typically between zero and one, simplifying the analysis of datasets. Following normalization, the PCA method is applied to examine unit violations (removal of unimportant or ineffective features) [45, 46]. The normalized variables serve as input for PCA, and the results are illustrated using a pebble chart, displaying the accuracy rate based on the number of features in a specified range, demonstrating the reduction in dimensions and the percentage of tuple reduction.

For instance, in the breast cancer dataset, the first feature's effect is 60%, while the second feature contributes 40%. Combining these features yields an accuracy rate increase of 2.99%.

Table 5. A sample dataset.

Tuple number	Height	Weight	Shoe size	Skin color
1	175	70	40	A
2	175	75	39	C
3	175	69	40	C
4	176	71	40	A
5	210	115	54	B
6	150	73	30	A
7	170	40	39	B

In this example, Table 6 assumes that shoe sizes range between 37 and 41, heights range between 169 and 178, widths range between 62 and 81, and skin color is categorized into three groups: white (A), black (B), and red (C). This structured approach illustrates the effective-

https://doi.org/10.57647/fomj.2026.0701.06

ness of the proposed method in handling data cleaning, normalization, and pattern extraction, ensuring reliable results.

4.1.1 Identify and repair multiple violations

In this stage, we employ the Spline and Median methods to scrutinize the input data to discover, pinpoint, and rectify erroneous entries. As a result, we've identified and mitigated several anomalies by either eliminating or correcting incorrect data represented as outliers.

According to Table 6, applying the above two methods allows us to identify and rectify multiple violations in rows 5, 6, and 7, where outliers are visible. Table 7 presents the corrections made to the outlier data.

Table 6. Case study data.

Tuple number	Height	Weight	Shoe size	Skin color
1	175	70	40	A
2	175	75	39	C
3	175	69	40	C
4	176	71	40	A
5	210	115	54	B
6	150	73	30	A
7	170	40	39	B

Table 7. Modified data using two methods, Spline and Median.

Tuple number	Height	Weight	Shoe size	Skin color
1	175	70	40	A
2	175	75	39	C
3	175	69	40	C
4	176	71	40	A
5	178	81	41	B
6	169	73	37	A
7	170	62	39	B

4.1.2 Min-max normalization

After preprocessing, the data should be in an equal range with respect to each other, for example, all in the same range such as zero to one. Additionally, repeated records should be summarized, which is referred to as data normalization. This method is often known as Feature Scaling, where the equation (2) is used to normalize each variable and repeated records.

According to Table 6, the columns for Height, Weight, and Shoe Size have data ranges that are significantly different. By applying the normalization method and the equation above, all data are transformed into the range between zero and one, as illustrated in Table 8.

This normalization process ensures that all features contribute equally to the analysis, facilitating more accurate results in subsequent analyses.

4.1.3 Identify and repair unit violations

At this stage, unimportant or irrelevant features (features that do not have significant importance or role in achieving the desired goal) are removed from the dataset. Therefore, in the example of Table 6, which aims to determine the shoe size of individuals, the skin color feature is considered unimportant and is removed. As a result, after cleaning the data, Table 9 is obtained.

Table 8. Normalized data using equation (6).

Tuple number	Height	Weight	Shoe size	Skin color
1	0.66	0.42	0.75	A
2	0.66	0.68	0.5	C
3	0.66	0.36	0.75	C
4	0.77	0.47	0.75	A
5	1	1	1	B
6	0	0.57	0	A
7	0.11	0	0.5	B

Table 9. Purified data set with two-step SQL method.

Tuple number	Height	Weight	Shoe size
1	175	70	40
2	175	75	39
3	175	69	40
4	176	71	40
5	178	81	41
6	169	73	37
7	170	62	39

4.2 DiMε detection algorithm

The DiMε discovery algorithm presented in this section starts from automatic thresholds and performs a level-by-level generation of candidate dependencies. It employs a two-step SQL method to detect and clean the data. In the subsequent step, the algorithm compares the results of three combined methods: RFD-KMeans, RFD-Sugeno, and RFD-Sugeno-KMeans. Within the scope of eight distinct scenarios, we assess which of these methods excels in deriving the optimal model for multimedia data across various datasets, taking into account both equality and inequality of class sets, as well as the consistency of the results. The generation of candidate RFDs is achieved through a unique network constructed with consideration for all potential feature combinations. Specifically, given a relational schema $R = \{A_1 \cdot A_2 \cdot A_3 \cdot \ldots \cdot A_n\}$, its feature grid consists of an empty set at level zero, singular sets (one for each attribute) at level one, pair sets (one for each possible combination of two features) at level two, and continues in this manner until the last level, level m, which contains a singular set comprising all features of R. This structured approach ensures a comprehensive exploration of the feature space, facilitating the identification of the most effective dependencies for model optimization.

Following the generation process of column-based methods highlighted in Fig. 2, the proposed algorithm first generates attribute sets X at Level l and then formulates all possible RFDs $XA \rightarrow A$, were A in X, to be successively validated. Compared to similar algorithms for Functional Dependency (FD) discovery, the

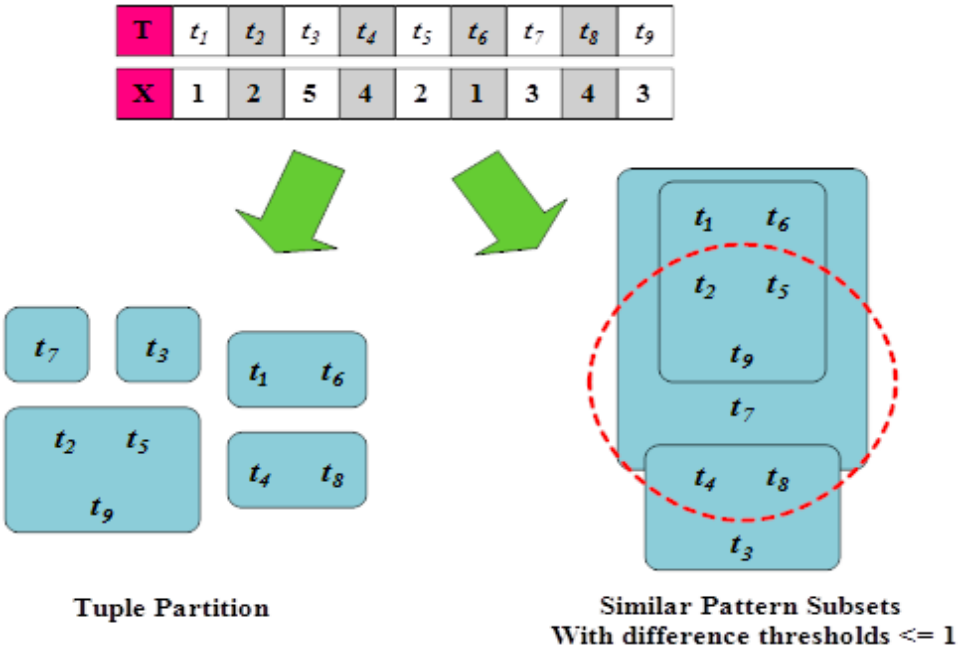

Figure 2. An example of multiple partitions and subsets of the same pattern.

proposed algorithm utilizes their candidate RFD generation strategies but introduces new pruning strategies to ensure the generation of minimal candidate RFDs. Additionally, it employs a validation technique that addresses similarity subsets and considers the possibility that an RFD might hold for a subset of tuples. Handling tuple similarities rather than strict equality adds significant complexity, as similarity subsets hinder the ability to leverage properties of disjoint partitions for RFD validation purposes. Since the Candidate Generation phase aligns with that of column-based FD discovery methods, we will concentrate on the remaining two phases of the proposed algorithm: validation and pruning, with a particular emphasis on automatic thresholding.

4.2.1 Candidate RFD validation

When candidate RFDs are generated by partitioning feature sets from network levels, their validation necessitates verifying that two tuples are identical on the feature in the RHS. This condition is applicable to a significant portion of the database whenever the tuples are similar in the LHS cases. The validation process involves generating similar subsets of tuples and utilizing these subsets to validate the candidate RFDs. This approach ensures that the candidate RFDs are accurately assessed for their validity based on the underlying similarities within the data. Figure 3 shows an example of multiple partitions and subsets of the same pattern.

4.2.2 Create similar subsets

Unlike tuple partitions commonly found in many functional dependency discovery algorithms, similarity subsets of tuples can also intersect, as illustrated in Fig. 4. To clarify the process of generating similarity subsets, we will introduce the concept of a difference matrix.

Definition 1 (Difference matrix of an attribute): Let r be an instance of a relation schema R, A an attribute of R, and δ a distance function defined on the domain of A. The difference matrix for A is a matrix M_A whose entry (i, j) contains the distance value $\delta(t_i[A] \cdot t_j[A])$ of the projections of tuples t_i and t_j on A.

Example 1: Let us consider the sample dataset in Table 8. Since it has all numerical attributes, we use the absolute difference, denoted with *abs*, as a distance

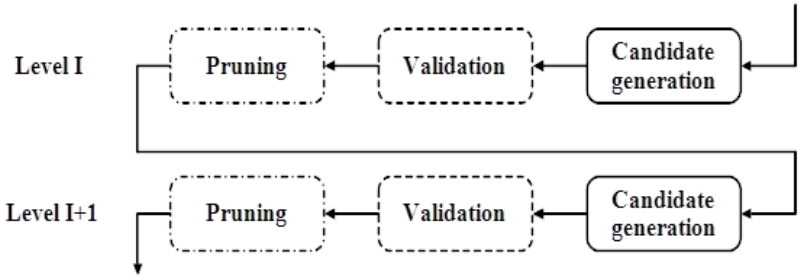

Figure 3. The main stage of column detection methods.

https://doi.org/10.57647/fomj.2026.0701.06

	1	2	3	4	5	6	7
1	0	0	0	1	3	6	5
2	0	0	0	1	3	6	5
3	0	0	0	1	3	6	5
4	1	1	1	0	2	7	6
5	3	3	3	2	0	9	8
6	6	6	6	7	9	0	1
7	5	5	5	6	8	1	0

(a) Difference Matrix $M_{Heightabs}$

	1	2	3	4	5	6	7
1	0	5	1	1	11	3	8
2	5	0	6	4	6	2	13
3	1	6	0	2	12	4	7
4	1	4	2	0	10	2	9
5	11	6	12	10	0	8	19
6	3	2	4	2	8	0	11
7	8	13	7	9	19	11	0

(b) Difference Matrix $M_{Weightabs}$

	1	2	3	4	5	6	7
1	0	1	0	0	1	3	1
2	1	0	1	1	2	2	0
3	0	1	0	0	1	3	1
4	0	1	0	0	1	3	1
5	1	2	1	1	0	4	2
6	3	2	3	3	4	0	2
7	1	0	1	1	2	2	0

(c) Difference Matrix $M_{Shoe\ Sizeabs}$

	1	2	3	4	5	6	7
1	0,0	0,5	0,1	1,1	3,11	6,6	5,8
2	0,5	0,0	0,6	1,4	3,6	6,2	5,13
3	0,1	6,6	0,0	1,2	3,12	6,4	5,7
4	1,1	1,4	1,2	0,0	2,10	7,2	6,9
5	3,11	3,6	3,12	2,10	0,0	9,8	8,19
6	6,3	6,2	6,4	7,2	9,8	0,0	1,11
7	5,8	5,13	5,7	6,9	8,19	1,11	0,0

(d) Difference Matrix of $M_{Heightabs.}$ $M_{Weightabs}$

Figure 4. Distance matrices for the dataset are shown in Table 8.

function to construct the following three difference matrices: $M_{Heightabs}$, $M_{Weightabs}$, and $M_{Shoe\ Sizeabs}$, as shown in Fig. 4a-c.

The definition of difference matrix can be easily generalized to attribute sets, in which an entry will contain an n-tuple of distance values, one for each attribute in the set [43].

Definition 2 (Difference matrix of an attribute set): Let r be an instance of a relation schema R, $X = \{A_1, A_2, A_3, \ldots, A_n\}$ an attribute set of R, and $\Delta = (\delta_1, \delta_2, \delta_3, \ldots, \delta_n)$ a sequence of distance functions defined on the domains of $A_1 \cdot A_2 \cdot A_3 \cdot \ldots \cdot A_n$, respectively. A difference matrix for X is a matrix $M_{X\Delta}$ whose entry (i, j) contains the n-tuple $(d_1, d_2, d_3, \ldots, d_n)$, where $d_k = \delta_k(t_i[A_k] \cdot t_j[A_k])$, and t_i, t_j are tuples of r [43].

The difference matrix for an attribute set X can be derived from the difference matrices for the single attributes composing it, by concatenating elements with the same coordinates. As an example, for the dataset of Table 8, the difference matrix $M_{Heightabs}$ and Weigh tabs, shown in Fig. 4d, is built from $M_{Heightabs}$ and $M_{Weightabs}$, of Fig. 4a, and b [1].

We exploit the notion of difference matrix to generate the *similar pattern* of a tuple t or a matrix row, which represents sets of tuples or matrix columns satisfying a given constraint with respect to t. Successively, we group the tuples sharing the same similar patterns into similar pattern subsets [1].

Definition 3 (Similar pattern): Let r be an instance of a relation schema R. $X = \{A_1, A_2, A_3, \ldots, A_n\}$ an attribute set of R, $M_{X\Delta}$ a difference matrix for X, and $\varphi = (\varphi_1, \varphi_2, \varphi_3, \ldots, \varphi_n)$ a sequence of constraints on the values of $M_{X\Delta}$, A similar pattern of tuple t_i of $M_{X\Delta}$, denoted as τ_x^{ti}, is the sequence $(j_1, j_2, j_3, \ldots, j_h)$ with $h \leq |r|$ and $M[i, j_k] = (d_1, d_2, d_3, \ldots, d_n)$, where d_q satisfies the constraints $\varphi_q \forall q \in [1, n]$ and $\forall k \in [1, h]$ [1].

Definition 4 (subsets of the same pattern): Let r be an instance of relation scheme R, $X = \{A_1, A_2, A_3, \ldots, A_n\}$ be a set of features of R, $M_{X\Delta}$ a distance matrix for X, and $\varphi = (\varphi_1, \varphi_2, \varphi_3, \ldots, \varphi_n)$ a sequence of constraints on the values of $M_{X\Delta}$. A subset of the pattern similar to S_X for X is defined as $S_X = \{j_1, j_2, j_3, \ldots, j_h\}\{i_1, i_2, i_3, \ldots, i_k\}$ with $1 \leq k \leq h \leq |r|$, $\tau_x^{ip} = (j_1, j_2, j_3, \ldots, j_h) \forall p \in [1, k]$ and $\tau_x^{iv} X \neq (j_1, j_2, j_3, \ldots, j_h) \forall v \notin [1, k]$.The set of different similar pattern subsets for X is denoted as I_X [1].

4.2.3 Automatic thresholding to create similar subsets

By utilizing automatic thresholding, we can minimize errors in responses and effectively extract relevant patterns. This study integrates several methodologies, specifically RFD-KMeans, RFD-Sugeno, and RFD-Sugeno-KMeans, to achieve this goal.

To determine the automatic threshold limit, the RFD method inputs a distance matrix with equal dimensions to the data into the Sugeno fuzzy system. This fuzzy system then transforms the distance matrix into multiple clusters, yielding, for example, three optimal clusters that offer the best accuracy among the n clusters generated. The outputs are subsequently revisited and reclustered using the K-means clustering algorithm, with the number of newly formed clusters defining the automatic threshold.

The results indicate that employing the RFD-Sugeno-KMeans method produces the most favorable threshold, resulting in precise answers and stability in identifying the optimal pattern with the fewest tuples. The Sugeno fuzzy inference system automatically determines the RFD's threshold limit through fuzzy logic and subsequently applies the K-means clustering algorithm to achieve the most optimal clustering.

To illustrate the operation of this system, we examined the breast cancer dataset, which consists of eight inputs and one output (as shown in Fig. 5). However, for the purposes of this discussion, we will focus on its three-

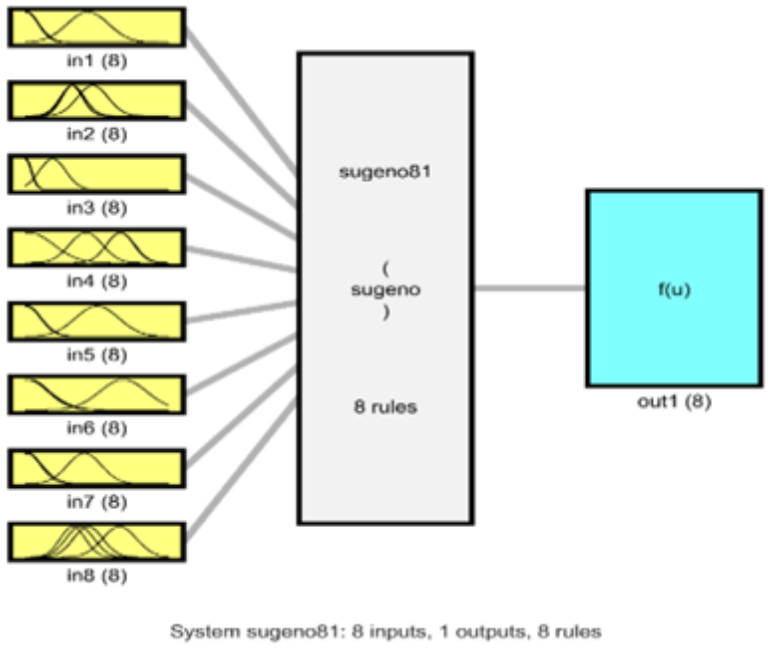

Figure 5. A view of the proposed fuzzy system.

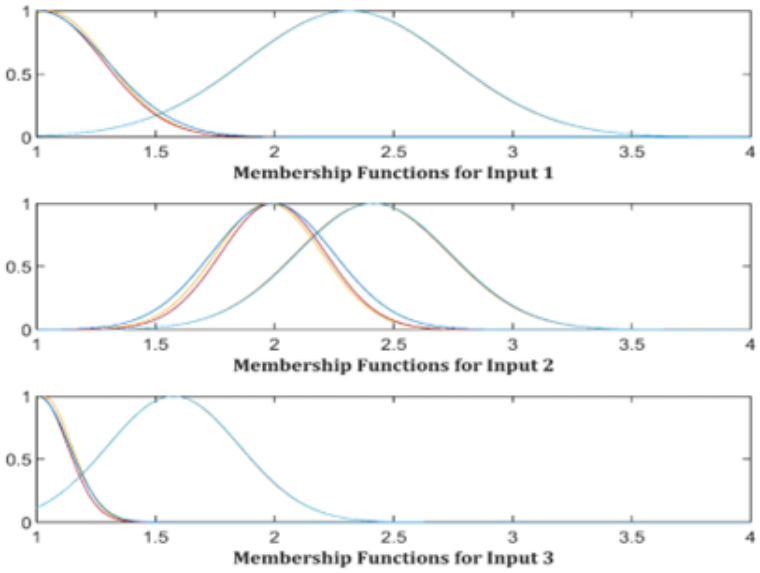

Figure 6. Diagram of three fuzzy input systems from the breast cancer dataset.

input model (as depicted in Fig. 6).

The diagram below (Fig. 7) illustrates the relationship between the first and second inputs and the output in the fuzzy system. This three-dimensional diagram represents various states, providing an example of their interactions.

In the fuzzy system described above, the rules are established automatically. The k-means clustering algorithm is then utilized to optimize the membership functions and rules of this fuzzy system.

Example 2: Consider the dataset presented in Table 8, where boundaries are set based on the absolute value function, taking into account the threshold limit (the number of clusters with the best accuracy obtained from re-clustering with k-means). In this scenario, the comparison operator ≤ 1 is designated as the automatic threshold for both the height and shoe size attributes, while a threshold of 10 is set for the weight attribute. Consequently, Fig. 8 and the following sets of similar pattern subsets are generated:

$-\ \mathrm{I}_{\mathrm{Height}} = \{\{1,2,3,4\}_{\{1,2,3,4\}}, \{5\}_{\{5\}}, \{6,7\}_{\{6,7\}}\};$

$-\ \mathrm{I}_{\mathrm{weight}} = \{\{1,2,3,4,6,7\}_{\{1,3\}}, \{1,2,3,4,5,6\}_{\{2,6\}}, \{1,2,3,4,5,6,7\}_{\{4\}}, \{2,4,5,6\}_{\{5\}}, \{1,3,4,7\}_{\{7\}}\};$

$-\ \mathrm{I}_{\mathrm{shoe\ size}} = \{\{1,2,3,4,5,7\}_{\{1,3,4\}}, \{1,2,3,4,7\}_{\{2,7\}}, \{1,3,4,5\}_{\{5\}}, \{6\}_{\{6\}}\};$

These subsets are highlighted in Fig. 8, with each subset represented in a different color. To validate candidate

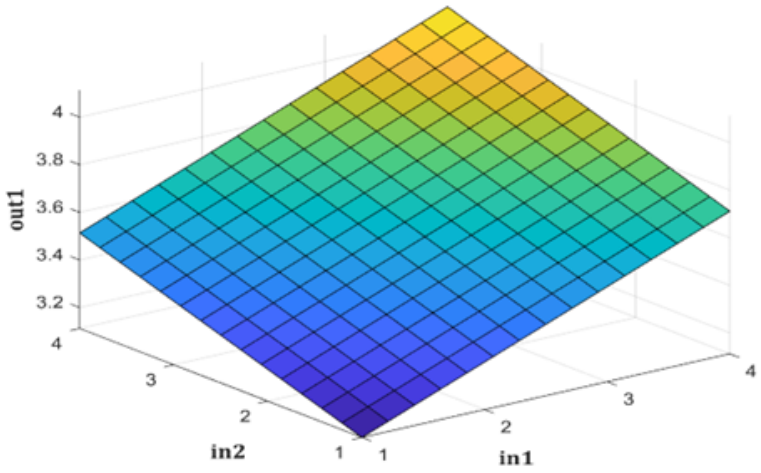

Figure 7. Diagram of the relationship between first and second input with output in fuzzy system.

RFDs, the proposed method reduces the number of similar pattern subsets for the attributes of the LHS by eliminating singletons. Singletons correspond to the diagonal entries of the matrix, which trivially compare each tuple with itself. This process leads to the definition of the set of stripped similar pattern subsets.

Definition 5 (Set of stripped similar pattern subsets): Let r be an instance of a relation schema R, $X = \{A_1, A_2, A_3, \ldots, A_n\}$ an attribute set of R, and I_X a set of similar pattern subsets of X. The set of stripped similar pattern subsets I_X of X is defined as formula (6):

$$\{S_X = I_X | S_X = \{j_1, j_2, j_3, \ldots, j_h\}\{i_1, i_2, i_3, \ldots, i_h\} \text{ and } h > 1\}. \tag{6}$$

5. Results and analysis

To verify the validity of the proposed method, simulations were conducted using the MATLAB programming language within the MATLAB R2021b environment on

	1	2	3	4	5	6	7
1	0	0	0	1	3	6	5
2	0	0	0	1	3	6	5
3	0	0	0	1	3	6	5
4	1	1	1	0	2	7	6
5	3	3	3	2	0	9	8
6	6	6	6	7	9	0	1
7	5	5	5	6	8	1	0

(a) Similar pattern subsets of Height.

	1	2	3	4	5	6	7
1	0	5	1	1	11	3	8
2	5	0	6	4	6	2	13
3	1	6	0	2	12	4	7
4	1	4	2	0	10	2	9
5	11	6	12	10	0	8	19
6	3	2	4	2	8	0	11
7	8	13	7	9	19	11	0

(b) Similar pattern subsets of Weight.

	1	2	3	4	5	6	7
1	0	1	0	0	1	3	1
2	1	0	1	1	2	2	0
3	0	1	0	0	1	3	1
4	0	1	0	0	1	3	1
5	1	2	1	1	0	4	2
6	3	2	3	3	4	0	2
7	1	0	1	1	2	2	0

(c) Similar pattern subsets of Shoe size.

Figure 8. Color distance matrices highlighting similar pattern subsets.

https://doi.org/10.57647/fomj.2026.0701.06

a PC equipped with an i5 CPU, eight gigabytes of RAM, and a one-gigabyte graphics card.

The datasets utilized for this method was sourced from the UCI Machine Learning Repository, comprising diverse medical data related to various individuals. One of the fields in this database pertains to the nature of the data type in terms of class equality (whether the classes are equal or unequal).

Our examination included eight scenarios, each based on three key considerations: feature selection using PCA, normalization, and the correction or removal of outlier data (multiple violations). Each of these methods was applied to the existing datasets.

As outlined in Table 10, it can be noted that in Scenario 8, by implementing normalization and optimal feature selection for the dataset, stability in the results was observed. Specifically, in 15 independent repetitions conducted for each dataset in Scenario 8, all 15 repetitions yielded consistent results. This consistency was maintained regardless of the nature of the data type concerning class equality (whether the classes were equal or unequal). This indicates that the reliability of the responses obtained from Scenario 8 is significantly higher.

Table 10. Scenarios that have been explored.

Scenario	Correction of outliers	Normalization	Feature selection
1	✓	✓	✓
2	✓	-	-
3	✓	✓	-
4	-	-	✓
5	-	-	-
6	-	✓	✓
7	-	✓	-
8	✓	-	✓

In the subsequent stage of the study, we examined a combination of several methods (RFD-KMeans, RFD-Sugeno, and RFD-Sugeno-KMeans). It was determined that the RFD-Sugeno-KMeans method outperformed the others, achieving the best results across various datasets (whether the classes were equal or unequal) while also demonstrating greater consistency in the outcomes.

In contrast, the results from other scenarios clearly indicate that the removal or modification of outlier data leads to drastic changes in the outcomes. This variability complicates the determination of the best result for each scenario, which is undesirable. In any evaluation or presentation of an algorithm, it is essential that the stability of the results and the reliability of the responses are clear and understandable to the user.

To address this, the nearest neighbor method was employed in the preprocessing phase to replace missing data across different datasets. This approach mitigates the risk of removing an entire dataset due to the loss of a single data point, which could result in the loss of important information from the original dataset.

6. Evaluation of the results

Pareto charts have become widely recognized as one of the most popular tools for quality control worldwide, allowing users to easily identify the most significant problems. These charts combine line and bar graphs, with the longest bars positioned on the left side representing the most critical issues. The primary advantage of the Pareto chart structure is that it enables quick identification of areas to focus on. From left to right, the bars are arranged from largest to smallest, while the line at the top indicates the overall share or percentage.

Typically, a Pareto chart features different categories, each associated with specific numerical values. This allows for data analysis in terms of the frequency of events, which is usually measured in terms of cost, quantity, or time.

To provide a simple and popular definition of Pareto analysis, we can state that: Pareto analysis suggests that 80% of your problems arise from just 20% of the factors. In other words, by concentrating on just 20% of the causes, you can resolve 80% of your problems.

The graphs have been analyzed using both balanced and imbalanced datasets to evaluate the results. For instance, the findings have been demonstrated using the Iris dataset (balanced) and the breast cancer dataset (imbalanced).

Fig. 9 illustrates the variance in the percentage of data reduction. The ideal scenario is positioned on the right side of the Pareto diagram. Notably, the eighth scenario demonstrated the best performance in both the Iris dataset (balanced) and the breast cancer dataset (imbalanced). The placement of the red dot on the graph indicates the quality of each scenario, with a lower position signifying poorer performance.

Building upon the previous study on the percentage of data reduction, figure 10 presents the number of remaining tuples for each of the eight scenarios under investigation. In the Pareto diagram, the scenario with the best performance is positioned at the rightmost point. In this case, the seventh scenario demonstrated the best performance in both the Iris dataset (balanced) and the breast cancer dataset (imbalanced). The lowest position of the red dot indicates poorer performance for that scenario.

Now, we will discuss which method exhibited the best performance among the eight scenarios studied. The results for these different scenarios are presented in Tables 11 and 12, based on the nature of the data from the two sample datasets: Iris (balanced) and Breast Cancer (imbalanced).

By identifying scenario seven as the most favorable option, we can proceed to determine the best method among the three presented choices. The results obtained are illustrated in Fig. 11.

Referring to the aforementioned tables, Fig. 11 presents the average variance of the percentage of data reduction and the number of tuples. Notably, it demonstrates that the RFD-Sugeno-KMeans method exhibits superior performance, yielding the best results for both

https://doi.org/10.57647/fomj.2026.0701.06

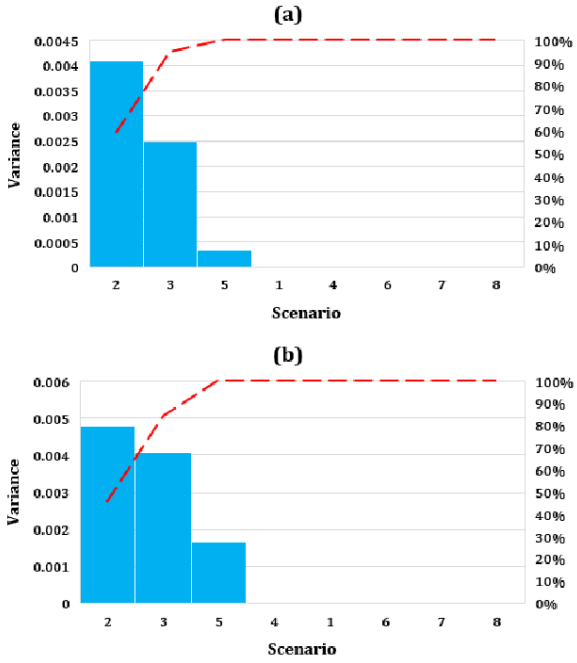

Figure 9. Pareto diagram of the variance value of data reduction percentage in (a) breast cancer (b) Iris data set in eight scenarios.

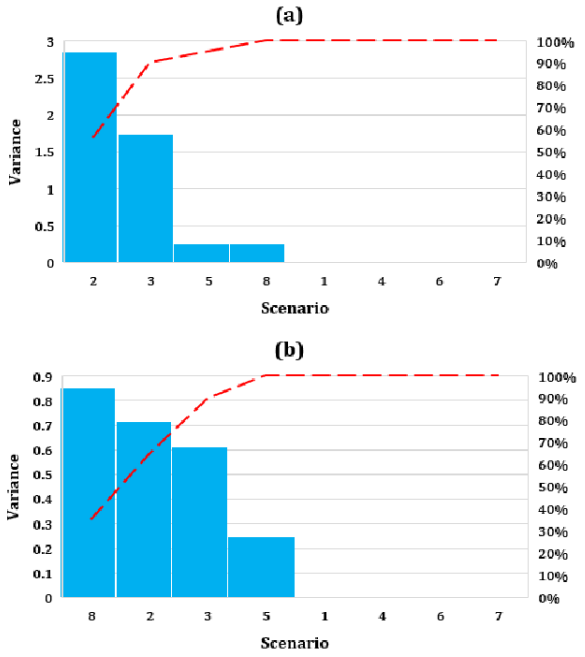

Figure 10. Pareto diagram of the variance value of the number of remaining tuples in the (a) breast cancer (b) Iris data set in eight scenarios.

the Iris (balanced) and the breast cancer (imbalanced) datasets.

In this study, we conducted a comparative evaluation of the previous methodologies of FD and RFD, as detailed in [43]. The objective was to examine the efficacy of the DiMε algorithm in detecting RFDs using an automatic threshold. To achieve this, we explored three combined methods involving RFDs: RFD-KMeans, RFD-Sugeno, and RFD-Sugeno-KMeans.

For the RFD-Sugeno-KMeans method, we determined the optimal number of clusters by applying a clustering approach to the results obtained from Sugeno fuzzy inference, which we then utilized for automatic thresholding. After a data cleansing process, we extracted the most desirable pattern, characterized by the smallest number of tuples.

To facilitate comparison, we employed diverse datasets, including the Iris dataset (balanced), the breast cancer dataset (imbalanced), and the lymphographydataset (imbalanced). The results of these comparisons are illustrated in Table 13 and figure 12.

Table 11. The amount of standard deviation of the decrease of data's in two imbalanced and balance sample datasets.

Data	Method	Scenarios								statistical measures		
		1	2	3	4	5	6	7	8	Mean	SD	Improvement (%)
Breast cancer	A	0	0.004	0.002	-	0.0003	0	0	0	0.0009	0.00155	
	B	0.001	0.001	0.001	-	1.110	1.110	1.110	1.111	0.63486	0.59292	
	C	0.001	0	0	-	0	2.220	0	0	0.31729	0.83902	-97.35
Iris	A	0	0.004	0.004	2.220	0.001	0	0.0056	0.017	0.28145	0.78331	
	B	3.330	0.003	0.001	0.002	0.001	3.330	0.002	0.013	0.83535	1.5398	
	C	3.330	2.220	2.220	0.002	2.220	0.001	2.220	2.225	1.80475	1.17693	-23.54

* Method: (A) RFD-KMeans, (B) RFD-Sugeno, (C) RFD-Sugeno-KMeans

Table 12. The amount of standard deviation of the decrease of in the number of tuples in two imbalanced and balance sample datasets.

Data	Method	Scenarios							
		1	2	3	4	5	6	7	8
Breast cancer	A	0	2.856	1.738	-	0.249	0	0.249	0
	B	0.805	0.8	0.869	-	0	0	0	0.708
	C	0.816	0	0	-	0	0	0	0
Iris	A	0	0.718	0.611	0	0.249	0	0.853	0.705
	B	0	0.498	0.249	0.679	0.249	0	0.4	0.248
	C	0	0	0	0.4	0	0.249	0	0

* Method: (A) RFD-KMeans, (B) RFD-Sugeno, (C) RFD-Sugeno-KMeans

https://doi.org/10.57647/fomj.2026.0701.06

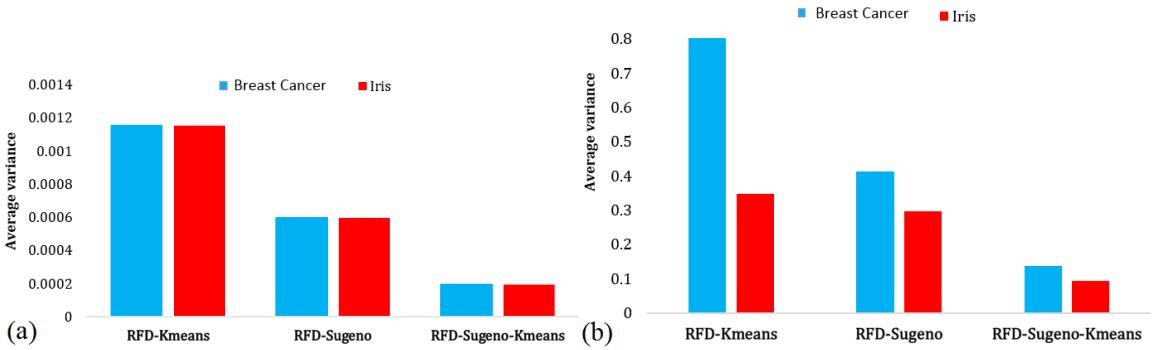

Figure 11. (a) The average variance of percentage reduction average, (b) The variance of the number of tuples in the three proposed methods.

Table 13. Comparison of the proposed method with previous methods (FD) and (RFD).

Nature of data	The name of the data set	FD[43]	RFD[43]	RFD-KMeans	RFD-Sugeno	RFD-Sugeno-KMeans
Balance	Iris	8	5	4	4	3
Imbalance	Breast cancer	46	10	8	3	2
	Lymphography	2730	1882	1315	1110	986

The analysis of the results indicates that our proposed methodologies produced fewer tuples compared to the previous methods. This significant reduction in duplicate data has led to a decrease in computational complexity relative to the approaches described in [43]. Therefore, we can confidently assert that the RFD-Sugeno-KMeans method is effective in extracting an optimal pattern characterized by the least number of tuples and minimal computational complexity, ultimately delivering superior outcomes.

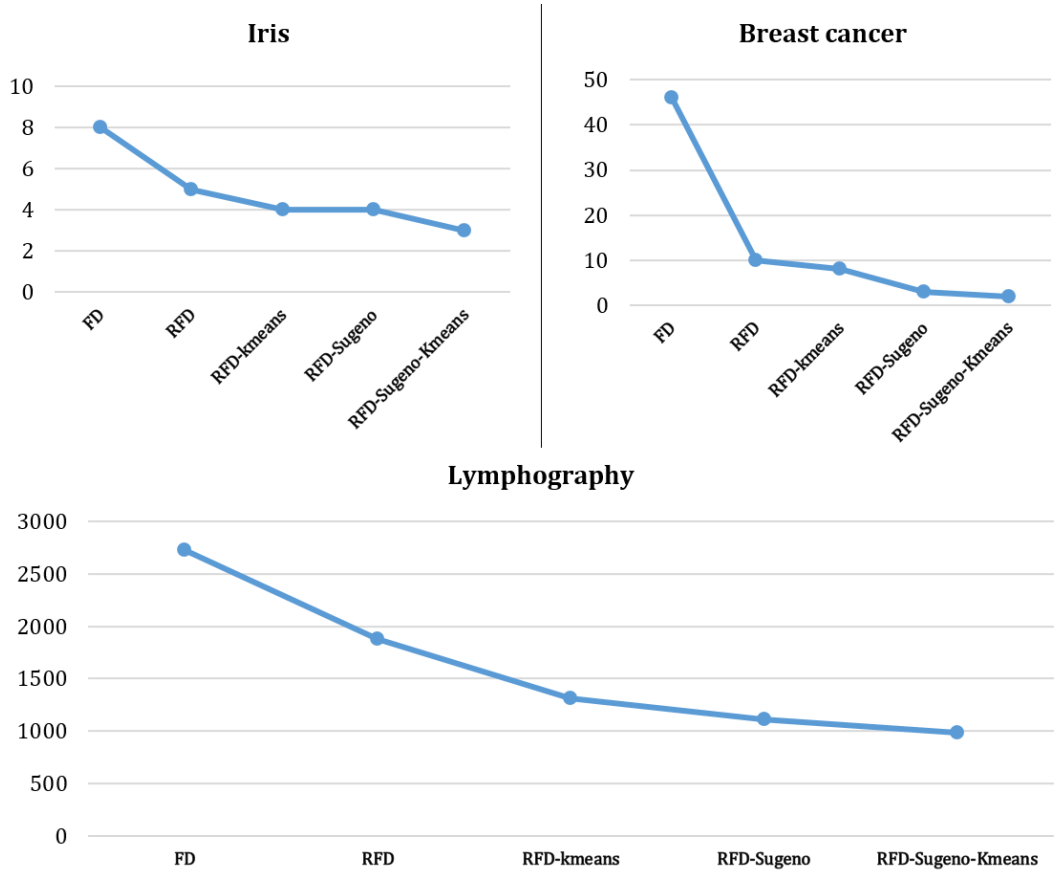

Figure 12. Comparing the proposed methodologies with previous methods [1], focusing on tuple count.

https://doi.org/10.57647/fomj.2026.0701.06

7. Conclusion and future works

Data cleaning is a critical area of research that has garnered significant attention from scholars, with many striving to enhance the quality, accuracy, and precision of data through the development and refinement of their models. Given the importance of data cleaning in contemporary society, it is essential to present methods that yield optimal results and patterns in this field.

In this study, we introduced three combined methods RFD-KMeans, RFD-Sugeno, and RFD-Sugeno-KMeans across eight scenarios, each designed to provide the best clustering for extracting optimal patterns with the minimum number of tuples. We employed two techniques, Spline and Median, to identify and repair multiple violations, while the PCA algorithm was utilized for identifying and addressing single violations, alongside normalization using the Min-Max method.

The results indicate that scenario 6, combined with the RFD-Sugeno-KMeans method, achieved the best performance across both balanced and unbalanced datasets. This method demonstrated superior effectiveness in delivering optimal responses across different datasets, regardless of class distribution, and exhibited stability in the results.

Furthermore, considering the significance of providing a reliable algorithm in any study, the stability of results and the clarity of responses for users are paramount. To address this, we employed the nearest neighbor method in the preprocessing phase to replace missing data, as deleting records due to missing values could lead to the loss of essential information from the original dataset.

The proposed method was applied to datasets related to various diseases sourced from the UCI repository and was subsequently compared with the previous structures of FD and RFD in extracting optimal patterns. The findings reveal that our proposed method, with its ability to extract optimal patterns in data cleaning, demonstrates higher stability and reliability in responses compared to earlier methods. In summary the main findings are:

- PCA Method: The PCA technique was utilized to identify and repair unit violations by reducing dimensionality, resulting in high accuracy and reliability, thereby increasing the percentage of data cleaning.
- Spline and Median Methods: The application of these methods for identifying and repairing multiple violations significantly enhanced the percentage of data cleaning, contributing to stability and reliability in responses.
- Optimal Performance: The best performance was observed in scenario seven, which achieved the minimum number of clusters through the RFD-Sugeno-KMeans method. This scenario facilitated the extraction of the best pattern by determining the optimal automatic threshold. In a comparative evaluation of results across 15 runs for each method, this approach yielded the best outcomes with the least percentage of data and tuple reduction, while ensuring reliability and stability in responses.
- Significant Reduction: The reduction in the number of data points and tuples compared to previous methods was substantial, with a decrease of approximately 99.5% in most cases across different scenarios.

Future research directions may include the following:

- Hybrid Models: Combining fuzzy systems with deep learning for improved feature extraction in high- dimensional datasets.
- Streaming Data: Extending RFD-Sugeno-KMeans-Clean to handle data streams using online ANFIS training.
- Parallelization: Optimizing K-means++ and ANFIS for distributed computing to enhance scalability.

Authors contributions
All authors contributed to the conceptualization and methodology of the study, as well as to validation and supervision. Data collection, simulation, and analysis were carried out by Mona Kardehi Moghadam, who was also responsible for technical investigation, software, data curation, and visualization. The first draft of the manuscript was written by Mona Kardehi Moghadam. Subsequent writing, editing, reviewing, and revising were performed through iterative feedback from all authors. All authors have read, reviewed, and approved the final manuscript.

Availability of data and materials
The authors declare that the data supporting the findings of this study are available within the paper.

Conflict of interests
The authors assert that they do not have any identifiable conflicting financial interests or personal relationships that might be perceived to influence the work presented in this paper.

References

1. Caruccio L, Deufemia V, and Polese G. Mining relaxed functional dependencies from data. Data Min. Knowl. Discov 2020; 34(2):443–77. DOI: 10.1007/s10618-019-00667-7
2. Hariri S, Kind MC, and Brunner RJ. Extended isolation forest. IEEE Trans. Knowl. Data Eng 2020; 33(4):1479–89. DOI: 10.1109/TKDE.2019.2947676
3. Khan A, Khan MA, and Khan MA. Adaptive DBSCAN with intelligent parameter tuning. Appl. Intell 2022; 52(3):2456–70
4. Pang G, Shen C, Cao L, and Hengel AV. Deep learning for anomaly detection: A review. ACM Comput. Surv 2021; 54(2):1–38. DOI: 10.1145/3439950

5. Yasin HM and Khorsheed AK. Automated data cleaning in large databases using machine learning methods. Asian J. Res. Comput. Sci 2025; 18(5):364–86. DOI: 10.9734/ajrcos/2025/v18i5661
6. Beil D and Theissler A. Cluster-clean-label: An interactive machine learning approach for labeling high-dimensional data. Proc. 13th Int. Symp. Visual Information Communication and Interaction 2020 :1–8. DOI: 10.1145/3430036.3430060
7. Wang Q et al. Deep Q-network-based feature selection for multisourced data cleaning. IEEE Internet Things J 2020; 8(21):16153–64. DOI: 10.1109/JIOT.2020.3016297
8. Côté PO, Nikanjam A, Ahmed N, Humeniuk D, and Khomh F. Data cleaning and machine learning: A systematic literature review. Autom. Softw. Eng 2024; 31(2):54. DOI: 10.1007/s10515-024-00453-w
9. Ponzio F, Deodato G, Macii E, Di-Cataldo S, and Ficarra E. Exploiting 'uncertain' deep networks for data cleaning in digital pathology. 2020 IEEE 17th Int. Symp. Biomedical Imaging (ISBI) 2020 :1139–43. DOI: 10.1109/ISBI45749.2020.9098605
10. Li P et al. CleanML: A benchmark for joint data cleaning and machine learning [experiments and analysis]. arXiv preprint arXiv:1904 2019 :09483
11. Kokkonen H. Effects of data cleaning on machine learning model performance. Bachelor's thesis 2019
12. Qahtan A, Tang N, Ouzzani M, Cao Y, and Stonebraker M. Pattern functional dependencies for data cleaning. Proc. VLDB Endowment 2020; 13(5):684–97. DOI: 10.14778/3377369.3377377
13. Caruccio L, Deufemia V, and Polese G. Lattice-based discovery of hybrid relaxed functional dependencies. CEUR Workshop Proceedings 2020. DOI: 10.1007/s10618-019-00667-7
14. Prokoshyna N et al. Combining quantitative and logical data cleaning. Proc. VLDB Endowment 2015; 9(4):300–11. DOI: 10.14778/2856318.2856325
15. Bohannon P et al. Conditional functional dependencies for data cleaning. 2007 IEEE 23rd Int. Conf. Data Engineering 2006. DOI: 10.1109/ICDE.2007.367920
16. Livshits E, Kimelfeld B, and Wijsen J. Counting subset repairs with functional dependencies. J. Comput. Syst. Sci 2021; 117:154–64. DOI: 10.1016/j.jcss.2020.10.001
17. Hakawati MR et al. Data cleaning model for XML datasets using conditional dependencies. Eur. J. Electr. Eng. Comput. Sci 2020; 4(1):1–5. DOI: 10.24018/ejece.2020.4.1.163
18. Martinez-Mosquera D et al. Data cleaning technique for security logs based on Fellegi-Sunter theory. 10th SIGSAND/PLAIS Eurosymposium 2017 2017 :3–12. DOI: 10.1007/978-3-319-66996-0_1
19. Du Y et al. Discovering context-aware conditional functional dependencies. Front. Comput. Sci 2017; 11:688–701. DOI: 10.1007/s11704-016-5265-4
20. Fan W et al. Discovering graph functional dependencies. ACM Trans. Database Syst. (TODS) 2020; 45(3):1–42. DOI: 10.1145/3397198
21. Caruccio L et al. Discovering relaxed functional dependencies based on multi-attribute dominance. IEEE Trans. Knowl. Data Eng 2020; 33(9):3212–28. DOI: 10.1109/TKDE.2020.2967722
22. Schirmer P et al. DynFD: Functional dependency discovery in dynamic datasets. EDBT 2019
23. Schirmer P et al. Efficient discovery of matching dependencies. ACM Trans. Database Syst. (TODS) 2020; 45(3):1–33. DOI: 10.1145/3392778
24. Salem R and Abdo A. Fixing rules for data cleaning based on conditional functional dependency. Future Comput. Inform. J 2016; 1(1-2):1026. DOI: 10.1016/j.fcij.2017.03.002
25. Caruccio L, Deufemia V, and Polese G. A genetic algorithm to discover relaxed functional dependencies from data. SEBD 2017
26. Vucetic M, Hudec M, and Vujošević M. A new method for computing fuzzy functional dependencies in relational database systems. Expert Syst. Appl 2013; 40(7):2738–45. DOI: 10.1016/j.eswa.2012.11.019
27. Kumar R, Kavitha R, and Chadrasekaran RM. Attribute correction-data cleaning using association rule and clustering methods. Int. J. Data Min. Knowl. Manage. Process 2011; 1(2):22–32. DOI: 10.5121/ijdkp.2011.1202
28. Barlag T, Hannula M, Kontinen J, Pardal N, and Virtema J. Local consistency and axioms of functional dependence. arXiv preprint arXiv:2505 2025 :11057
29. Li Y and Li D. Photovoltaic abnormal data cleaning based on fuzzy clustering-quartile algorithm. 2023 IEEE 6th Int. Conf. Industrial Cyber-Physical Systems (ICPS) 2023 :1–5. DOI: 10.1109/ICPS58381.2023.10128026
30. Vučetić M, Hudec M, and Božilović B. Fuzzy functional dependencies and linguistic interpretations employed in knowledge discovery tasks from relational databases. Eng. Appl. Artif. Intell 2020; 88:103395. DOI: 10.1016/j.engappai.2019.103395
31. Pani AK and Mohanta HK. Online monitoring of cement clinker quality using multivariate statistics and Takagi-Sugeno fuzzy-inference technique. Control Eng. Pract 2016; 57:1–7. DOI: 10.1016/j.conengprac.2016.08.011

32. Khedher A, Othman KB, and Benrejeb M. Active fault tolerant control (FTC) design for Takagi-Sugeno fuzzy systems with weighting functions depending on the FTC. Int. J. Comput. Sci. Issues 2011; 8(3)
33. Gunasekaran J, Sevvel P, Solomon IJ, and Roy JV. Optimization of FSW parameters using SA algorithm and ANFIS-based models to maximize mechanical properties of AZ80A Mg alloy joints. J. Mater. Eng. Perform 2024 :1–20. DOI: 10.1007/s11665-024-10062-z
34. Pruengkarn R, Wong KW, and Fung CC. Data cleaning using complementary fuzzy support vector machine technique. Neural Information Processing: 23rd Int. Conf., ICONIP 2016 2016 :160–7. DOI: 10.1007/978-3-319-46672-9_19
35. Hudec M, Vučetić M, and Vujošević M. Synergy of linguistic summaries and fuzzy functional dependencies for mining knowledge in the data. 2014 18th Int. Conf. System Theory, Control and Computing (ICSTCC) 2014. DOI: 10.1109/ICSTCC.2014.6982438
36. Fatima A, Nazir N, and Khan MG. Data cleaning in data warehouse: A survey of data pre-processing techniques and tools. Int. J. Inf. Technol. Comput. Sci 2017; 9(3):50–61. DOI: 10.5815/ijitcs.2017.03.06
37. Caruccio L, Deufemia V, and Polese G. On the discovery of relaxed functional dependencies. Proc. 20th Int. Database Engineering & Applications Symp 2016. DOI: 10.1145/2938503.2938519
38. Vaz R et al. Automated big-O analysis of algorithms. 2017 Int. Conf. Nascent Technologies in Engineering (ICNTE) 2017. DOI: 10.1109/ICNTE.2017.7947882
39. Ježková L, Cordero P, and Enciso M. Fuzzy functional dependencies: A comparative survey. Fuzzy Sets Syst 2017; 317:88–120. DOI: 10.1016/j.fss.2016.06.019
40. Skjerve TA et al. Using density and fuzzy clustering for data cleaning and segmental description of livestock data. J. Agric. Biol. Environ. Stat 2024 :1–6. DOI: 10.1007/s13253-024-00622-0
41. Chu X and Ilyas IF. Qualitative data cleaning. Proc. VLDB Endowment 2016; 9(13):1605–8. DOI: 10.14778/3007263.3007320
42. Sun R et al. Research on multi-source heterogeneous data cleaning technology based on integrating neural network with fuzzy rules for renewable energy accommodation. 2020 IEEE 4th Conf. Energy Internet and Energy System Integration (EI2) 2020 :3024–7. DOI: 10.1109/EI250167.2020.9346757
43. Liu K, Ma J, and Lai EM. A dynamic fuzzy rule and attribute management framework for fuzzy inference systems in high-dimensional data. arXiv preprint arXiv:2504 2025 :19148. DOI: 10.2139/ssrn.5279538
44. Ding X et al. Efficient relaxed functional dependency discovery with minimal set cover. 2024 IEEE 40th Int. Conf. Data Engineering (ICDE) 2024 :3519–31. DOI: 10.1109/ICDE60146.2024.00271
45. Caruccio L, Cirillo S, Iuliano G, Polese G, and Stanzione R. RYAN: A tool for explaining and visually analyzing the evolution of relaxed functional dependencies. 2024 IEEE Int. Conf. Big Data (BigData) 2024 :1249–54. DOI: 10.1109/BigData62323.2024.10826143
46. Fan W. Extending dependencies with conditions for data cleaning. 2008 8th IEEE Int. Conf. Computer and Information Technology 2008
47. Rahul K and Banyal R. Data cleaning mechanism for big data and cloud computing. 2019 6th Int. Conf. Computing for Sustainable Global Development (INDIACOM) 2019
48. Ezugwu AE et al. A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects. Eng. Appl. Artif. Intell 2022; 110:104743. DOI: 10.1016/j.engappai.2022.104743
49. Kriegel HP, Schubert E, and Zimek A. The (black) art of runtime evaluation: Are we comparing algorithms or implementations? Knowl. Inf. Syst 2017; 52:341–78. DOI: 10.1007/s10115-016-1004-2
50. Hartigan JA and Wong MA. Algorithm AS 136: A k-means clustering algorithm. J. R. Stat. Soc. Ser. C (Appl. Stat.) 1979; 28(1):100–8. DOI: 10.2307/2346830
51. Mackay DJ. Information Theory, Inference and Learning Algorithms. Cambridge, U.K.: Cambridge Univ. Press 2003
52. Repository UML. Available from: https://archive.ics.uci.edu/datase.ts
53. Huhtala Y, Karkkainen J, Porkka P, and Toivonen H. TANE: An efficient algorithm for discovering functional and approximate dependencies. Comput. J 1999; 42(2):100–11. DOI: 10.1093/comjnl/42.2.100
54. Caruccio L, Cirillo S, Deufemia V, Polese G, and Stanzione R. REQUIRED: A tool to relax queries through relaxed functional dependencies. EDBT 2023 :823–6
55. Breve B, Caruccio L, Deufemia V, and Polese G. RENUVER: A missing value imputation algorithm based on relaxed functional dependencies. EDBT 2022 :1–52

56. Breve B, Caruccio L, Cirillo S, Deufemia V, and Polese G. Indibits: Incremental discovery of relaxed functional dependencies using bitwise similarity. 2023 IEEE 39th Int. Conf. Data Engineering (ICDE) 2023 :1393–405. DOI: 10.1109/ICDE55515.2023.00111

57. R. Garg MS nd and Mishra P. MapReduce-based parallel data cleaning algorithm in web usage mining. Int. J. Comput. Sci. Appl 2017; 14(2). DOI: 10.4018/IJITWE.2018040102

58. Zhang W, Wang D, and Tan X. Robust class-specific autoencoder for data cleaning and classification in the presence of label noise. Neural Process. Lett 2019; 50:1845–60. DOI: 10.1007/s11063-018-9963-9

59. Fanani L and Priandani ND. Data cleaning and prototyping using k-means to enhance classification accuracy. Int. J. Appl. Eng. Res 2017; 12(15):5242–7

60. Bertossi L, Kolahi S, and Lakshmanan LV. Data cleaning and query answering with matching dependencies and matching functions. Proc. 14th Int. Conf. Database Theory 2011. DOI: 10.1145/1938551.1938585

61. Ansari Z, Azeem MF, Babu AV, and Ahmed W. A fuzzy clustering-based approach for mining usage profiles from web log data. arXiv preprint arXiv:1509 2015 :00693