Journal of Data Science and Modeling, Vol. 2, No. 2, 217-244, June 2024

Research Manuscript

Fuzzy K-Nearest Neighbor in Classification and Regression

Zahra Behdani*, Majid Darehmiraki
Department of Mathematics, Behbahan Khatam Alanbia University of Technology, Khouzestan, Iran.

Recieved: 14/12/2024 Accepted: 29/05/2025

Abstract:
The Fuzzy K-Nearest Neighbour (FKNN) method is a classification approach that integrates fuzzy theories with the K-Nearest Neighbour classifier. The algorithm computes the degree of membership for a given dataset within each class and then chooses the class with the highest degree of membership as the assigned classification outcome. This algorithm has several applications in regression problems. When the mathematical model of the data is not known, this method can be used to estimate and approximate the value of the response variable. This paper introduces a method, which incorporates a parametric distance measure to empower decision-makers to make precise selections across several levels. Furthermore, we provide an analysis of the algorithm's strengths and shortcomings, as well as a comprehensive explanation of the distinctions between the closest neighbour approach in tasks of classification and regression. Finally, to further elucidate the principles, we present a range of examples that demonstrate the application of closest neighbour algorithms in the classification and regression of fuzzy numbers.
Keywords: Fuzzy, K-Nearest Neighbour, Distance, Classification, Regression.
Mathematics Subject Classification (2010): 62H30, 68T10.

*Corresponding Author: behdani@bkatu.ac.ir

1. Introduction

The K-Nearest Neighbours (KNN) algorithm, a non-parametric approach, is a widely used machine learning tool for classification, clustering, and regression applications. It belongs to the category of lazy learning algorithms, which eliminate the need for offline training. The K-Nearest Neighbours method performs a direct search over all of the training instances during the classification stage for a particular testing example. We accomplish this task by calculating the distances between the testing example and all of the training data sources. The objective of this search is to identify the closest neighbours of the testing case in order to get a classification result. This algorithm ascertains the class affiliation of an unlabelled sample by considering the class affiliations of the K-labelled samples that are in closest proximity to it. Various studies have explored the choice of K, proposing various variants of KNN. However, no variant has proven to outperform all others. Some proposed variants ensure that the K nearest neighbours are close to the unlabelled sample and find K along the way. These variants have been tested and compared to the standard KNN in theoretical scenarios and for indoor localization based on ion-mobility spectrometry fingerprints, achieving higher classification accuracy while maintaining the same computational demand Müller (2023). Another variation of KNN uses the Nk interaction graph to determine the K Nearest Neighbours, allowing for the formation of clusters with arbitrary shapes. The original KNN has been compared to two new algorithms based on the Nk interaction graph in tests with datasets that have different properties Yacoub and *et al.* (2022). Additionally, researchers propose a novel classifier, Power Muirhead Mean K-Nearest Neighbours (PMM-KNN), to address challenges with outliers, small datasets, and unbalanced datasets. PMM-KNN calculates the local means of every class using the Power Muirhead Mean operator and has outperformed three state-of-the-art classification methods in experiments with five well-known datasets De Castro and Tinos (2022). Bian and colleagues Bian and *et al.* (2020) proposed a novel classification methodology based on the fuzzy K-nearest neighbours (FKNN) algorithm, referred to as the fuzzy KNN method with adaptive closest neighbours. The primary objective of this methodology was to ascertain a distinct optimal value of K for every testing sample. The cited work in Maillo et al. Maillo and *et al.* (2019) proposed both the global approximate hybrid spill tree FKNN and the local hybrid spill tree FKNN. Both approximate approaches aim to improve classification efficiency while maintaining accuracy and fidelity.
The integration of fuzzy sets with the KNN algorithm is motivated by the need to handle real-world data characterized by ambiguity, noise, and overlapping class boundaries. Traditional KNN relies on crisp classifications, which may fail to capture the inherent uncertainty in data where samples do not distinctly belong to

a single class. Fuzzy KNN (FKNN) addresses this limitation by assigning partial membership degrees to neighbours, thereby refining the decision-making process. This approach is particularly advantageous in scenarios such as medical diagnostics, financial forecasting, and sensor data analysis, where imprecision is prevalent. By incorporating fuzzy theory, FKNN enhances robustness and interpretability while mitigating the sensitivity to outliers and imbalanced datasets.

Zheng et al.'s Zheng and *et al.* (2023) study integrates the fuzzy K-nearest neighbour technique with the enhanced Sparrow Search Algorithm to derive numerical outcomes and evaluate their predictive precision. In their study, Wongkhuenkaew et al. (Wongkhuenkaew and *et al.* (2023)) outlined the use of the fuzzy K-nearest neighbour (FKNN) method for feature classification in the dental fluorosis classification system. In their study, Memis et al. (Memis and *et al.* (2022)) introduced a novel method known as FPFS-kNN (Fuzzy Parameterised Fuzzy Soft kNN). This algorithm demonstrates superior performance compared to previous kNN-based algorithms, achieving higher accuracy estimates in 24 out of 35 datasets. Kumbure et al. Kumbure and *et al.* (2020) developed the mean-based K closest neighbours approach. This technique uses the Bonferroni mean statistic. Biswas et al. proposed a dynamic fuzzy K-nearest neighbour classifier that is independent of parameters and incorporates feature weighting relevant to each class. The challenges associated with determining an optimum K-value and a set of feature optimal weights, which are contingent upon the class, were reformulated as optimization issues with a single objective in their resolution. To address the optimisation problem, the researchers used Differential Evolution Biswas and *et al.* (2018). Zhai et al. Zhai and *et al.* (2021) proposed a concise framework for fuzzy K-nearest neighbour optimisation. This approach entails iteratively transferring instances from the training set T to the beginning instance set S, after selecting pertinent examples from T. The authors explicitly partitioned their approach into three distinct phases. To begin, rather than using the variable T, the algorithm employs the function S to identify the K-nearest neighbours and thereafter calculates the fuzzy membership degrees of these selected KNN. The algorithm then uses the fuzzy K-nearest neighbours (KNN) approach to compute the fuzzy membership degrees of variable x. The quantified information entropy of x ultimately determines the selection of an instance. The writers Patel and Thakur came up with a way to fix the problem of uneven data and pick the right test instance memberships by combining an adaptive K-nearest neighbour technique with fuzzy KNN (Patel and Thakur (2019)). In Li et al. Li and *et al.* (2020, 2023) publication, the authors of the study presented two classification methods: a fuzzy granule K-nearest neighbour and a boosted fuzzy granule K-nearest neighbour. They developed the techniques in this study using fuzzy granulation data.

The majority class, including its nearest neighbours, decides the classification of a new sample using the K-Nearest Neighbours (KNN) method, which is based on the principle of majority voting. A distinct imbalance in data collection necessitates consideration of the potential limitations associated with the majority voting principle. In particular, this methodology demonstrates a preference for classifying novel samples by considering the prevalence of the class or classes that have a significant sample count. The primary factor contributing to this dominance is the higher prevalence of these classes among the K closest neighbours Ramadhani and *et al.* (2023). Using local means generated from the classes present among the nearest neighbours is one plausible approach to address this constraint. The determination of class assignment is contingent upon the local mean vector in close proximity, rather than being only reliant on the number of nearest neighbours. This methodology ensures that classes with a larger number of samples do not unduly influence classes with a smaller number of instances. Two factors influence the FKNN algorithm's classification results: the dominant class and the proximity of the unclassified sample to its nearest neighbours. It is important to note that differences in distance measurements can make it challenging to accurately judge how similar two samples are, which also affects the process of classifying.

Nearest neighbour (NN) algorithms, particularly in the context of regression, are foundational techniques in the field of machine learning and statistics. The primary idea behind the NN approach is to leverage the proximity of data points to make predictions about unknown instances. In a standard nearest neighbour regression, the algorithm identifies the 'K' closest training examples to a query point and aggregates their outputs, commonly by averaging, to predict the target value for that instance Cover and Hart (1967); Altman (1992). The efficiency and simplicity of KNN methods have contributed to their widespread use, especially in high-dimensional spaces where they can adapt without requiring a priori assumptions about the data distribution.

In contrast, fuzzy nearest neighbour (FNN) techniques expand upon the traditional nearest neighbour methodology by incorporating the concept of partial membership and uncertainty. Instead of treating the neighbours of a data point as having equal weighting, fuzzy approaches assign degrees of membership to neighbours based on their proximity to the query instance. This allows for a more nuanced aggregation of outputs, accommodating the inherent uncertainty in real-world data (Keller and *et al.* (1985); Atkinson and Tatnall (1998)). FNN is particularly beneficial in scenarios where the boundaries between classes are not well-defined, providing a more flexible framework for regression tasks.

The integration of fuzziness into the nearest neighbour paradigm serves to enhance prediction accuracy, especially in datasets characterized by noise and overlapping

class distributions. This dual approach of standard NN and its fuzzy counterpart is pivotal in various applications, including pattern recognition, time series forecasting, and decision-making, positioning these methods as essential tools in contemporary data analysis frameworks Michalski and *et al.* (1983).
Nearest neighbour regression is extensively used for predictive modeling in various domains, including finance, marketing, and biology. Its ability to model non-linear relationships makes it a popular choice in regression analysis (Cleveland (1979)). In fields such as environmental science and geography, Wackernagel Wackernagel (2003) used KNN regression to predict phenomena based on spatially distributed data points, facilitating better spatial decision-making. NN techniques are employed in recommendation systems where the goal is to predict user preferences based on the behaviors of similar users or items Linden and *et al.* (2003). Fuzzy nearest neighbour regression is utilized in contexts where data is imprecise or uncertain. Keller et al. Keller and *et al.* (1996) applied FKNN in medical diagnostics, where symptoms and test results may not have clear delineations.
Similar to K-NN classification (or any other prediction method), K-NN regression has both advantages and disadvantages. Here is a list of some:

- **Advantages:** The method is simple and easy to understand. It does not rely heavily on assumptions about the data's structure and is effective in handling non-linear connections, meaning it can handle relationships that are not straight lines.

- **Limitations:** The model's performance decreases significantly as the training data size increases. Additionally, the model may not work well when there are a large number of predictors, and it may not accurately predict values that are outside the range of values used in the training data.

Despite the advancements in FKNN, a critical gap remains in the adaptability of distance measures to diverse data structures and decision-making contexts. Existing FKNN methods often rely on fixed distance metrics (e.g., Euclidean or Diamond distance), which may not optimally capture the variability in fuzzy data representations or decision-maker preferences. Our study addresses this limitation by introducing a parametric distance measure that allows dynamic adjustment based on the problem's requirements. This innovation empowers decision-makers to fine-tune the similarity assessment at different granularity levels, enhancing flexibility in both classification and regression tasks. By bridging this gap, our approach improves the interpretability and precision of FKNN, particularly for fuzzy numbers with non-uniform spreads or asymmetric shapes (e.g., LR-type or trapezoidal fuzzy numbers).
In this article, the nearest neighbour algorithm is used in classification and re-

gression of fuzzy numbers. For this purpose, we have used different metrics; the results show that according to the nature of this algorithm, the results are constant for all these metrics. The rest of the article is as follows: First, in the second chapter, we introduce the general concepts of the article, such as fuzzy sets and fuzzy numbers. In the third chapter, we describe the nearest neighbour algorithm. We also explain the application of this algorithm in classification and regression. In the final part, we show the concepts presented in the article numerically and practically by mentioning numerical examples.

2. Preliminaries of Fuzzy Arithmetic

A fuzzy number is a mathematical abstraction that broadens the notion of a conventional number by including elements of ambiguity or imprecision. The domains of fuzzy analysis and interval analysis often employ this technique, effectively using it as a supportive premise for interval analysis. Fuzzy numbers have found use in several domains, including artificial intelligence techniques such as neural networks and quantum neural networks, with the aim of enhancing computational efficiency and precision in handling intricate and unpredictable datasets. Fuzzy equations and option pricing issues have been effectively addressed via their utilization, offering a meticulously organized synopsis and paving the way for novel prospects within the realm of fuzzy mathematics. Fuzzy numbers also have the ability to generalize to include a partially ordered set, such as a lattice, making them useful in fields such as cognitive mapping and expert evaluations.

Definition 2.1. *A fuzzy number M is a convex normal set of real numbers $\mathcal{R}$ such that:*

1. *There is only one $x_0 \in \mathcal{R}$ for which $\mu_M(x_0) = 1$.*
2. *The function μ_M is a continuous linear piecewise function.*

Definition 2.2. *A fuzzy number M is called of type LR if there are functions L (for the left side) and R (for the right side), and numbers $\alpha > 0$ and $\beta > 0$ such that*

$$\mu_M(x) = \begin{cases} L\left(\frac{m-x}{\alpha}\right), & x \leq m, \\ R\left(\frac{x-m}{\beta}\right), & x \geq m. \end{cases}$$

Here, m is called the middle and is a real number, and α and β are the left and right widths, respectively. Symbolically, M is represented as $(m, \alpha, \beta)_{LR}$, and L and R are called reference functions. Various functions are used as L and R, the

most famous of which are:

$$L(x) = \max\{0, (1-x)^p\}, \quad L(x) = e^{-x}, \quad L(x) = e^{-x^2}.$$

These functions can also be used as $R(x)$.

As algebraic operators are defined on ordinary numbers, these operations on fuzzy numbers LR can also be defined as follows:

Definition 2.3. *If* M *and* N *are two fuzzy numbers in the form of* $M = (m, \alpha, \beta)_{LR}$ *and* $N = (n, \gamma, \delta)_{LR}$ *then*

1. $(m, \alpha, \beta)_{LR} \oplus (n, \gamma, \delta)_{LR} = (m + n, \alpha + \gamma, \beta + \delta)_{LR}$.
2. $(m, \alpha, \beta)_{LR} \Theta (n, \gamma, \delta)_{LR} = (m - n, \alpha - \delta, \beta - \gamma)_{LR}$.
3. $$\lambda \otimes (m, \alpha, \beta)_{LR} = \begin{cases} (\lambda m, \lambda\alpha, \lambda\beta)_{LR}, & \lambda > 0, \\ (\lambda m, -\lambda\beta, -\lambda\alpha)_{LR}, & \lambda < 0. \end{cases}$$

2.1 Fuzzy number

Fuzzy numbers are a generalized form of real and ordinary numbers that include a range of possible values instead of referring to a specific value. The membership degree refers to the weight of each possible value, which ranges from 0 to 1. Fuzzy numbers are a special type of fuzzy set. Therefore, by understanding the concept of fuzzy sets, you can easily learn fuzzy numbers. In classical logic, every number is a definite value, but in fuzzy logic, every number is an approximate value. A fuzzy number is a fuzzy set with the following triple conditions:

- It is normal.
- It is convex.
- Its support set is limited.

People have proposed and used a wide variety of fuzzy numbers with different names and properties. But an important principle in applying fuzzy theory is its computational efficiency. Working with different fuzzy values presents many challenges. To solve this problem, Dubois and Prade Dubois and Prade (1990) introduced "right and left" fuzzy numbers known as LR numbers. Later, triangular and trapezoidal fuzzy numbers were introduced, which have high computational efficiency.

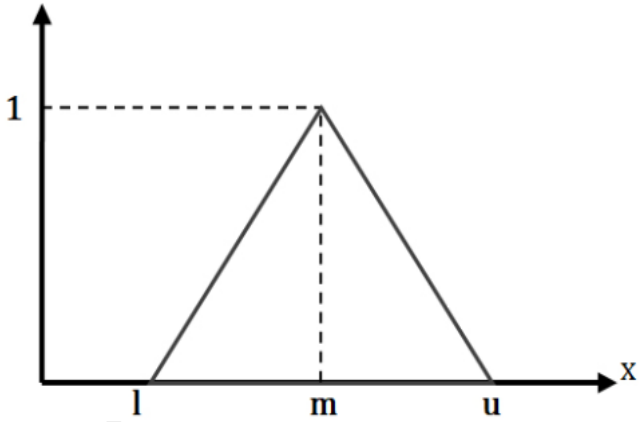


Figure 1: An example of a triangular fuzzy number

Three real numbers, $F = (l, m, u)$, represent a triangular fuzzy number (TFN). The upper limit indicated by u is the maximum value that the fuzzy number F can take. The lower limit indicated by l is the minimum value that the fuzzy number F can take. m is the most probable value of a fuzzy number. The degree of fuzzy membership, or the membership function of a triangular fuzzy number, can be expressed as follows:

$$\mu_F(x) = \begin{cases} \frac{x-l}{m-l}, & l \leq x \leq m, \\ \frac{u-x}{u-m}, & m \leq x \leq u. \end{cases}$$

In geometric space, the triangular fuzzy number $F = (l, m, u)$ is displayed as Figure 1.

According to the membership function of triangular numbers, it is clear that if x is between l and m, then the larger it is, the larger its membership degree will be, until the degree of membership is equal to one for $x = m$. If x is between m and u, the larger it is, the smaller the degree of membership will be, and at $x = u$, the degree of membership will be zero. $F = (l, m_1, m_2, u)$ represents a trapezoidal fuzzy number. The trapezoidal fuzzy number membership function is defined as follows:

$$\mu_F(x) = \begin{cases} \frac{x-l}{m_1-l}, & l \leq x \leq m_1, \\ 1, & m_1 \leq x \leq m_2, \\ \frac{u-x}{u-m_2}, & m_2 \leq x \leq u. \end{cases}$$

Figure 2 shows a geometric representation of a trapezoidal fuzzy number.

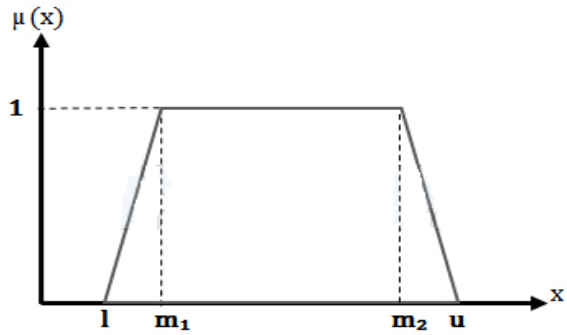


Figure 2: A sample trapezoidal fuzzy number

2.2 Comparison of Triangular and Trapezoidal Fuzzy Numbers in FKNN

Triangular fuzzy numbers (TFNs) and trapezoidal fuzzy numbers (TrFNs) are widely used in FKNN due to their computational efficiency and interpretability, but they differ in flexibility and applicability:

1. Representation:
    - A TFN $F = (l, m, u)$ encodes a single "most probable" value (m) with linear membership decay toward the lower (l) and upper (u) bounds. This simplicity makes TFNs suitable for symmetric, unimodal uncertainty.
    - A TrFN $F = (l, m_1, m_2, u)$ generalizes TFNs with a plateau of maximum membership between m_1 and m_2, accommodating intervals of equal plausibility (e.g., expert-defined ranges).
2. Relevance in FKNN:
    - TFNs are ideal for problems where uncertainty peaks at a single point (e.g., sensor readings with Gaussian-like noise). Their compact form simplifies distance computations (e.g., Equation 3.1).
    - TrFNs better model scenarios with imprecise intervals (e.g., "approximately 5 to 7 units") or multi-source consensus (e.g., aggregated expert opinions). The plateau allows FKNN to weigh neighbors more uniformly across the interval.
3. Trade-offs:

- TFNs require fewer parameters (3 vs. 4 for TrFNs), reducing computational overhead in large datasets.
- TrFNs offer richer representations but may introduce redundancy if the plateau is negligible (i.e., $m_1 \approx m_2$).

In FKNN, the choice between TFNs and TrFNs hinges on data characteristics: TFNs suffice for sharp, unimodal uncertainty, while TrFNs excel in capturing graded or multi-modal ambiguity (see Examples 5.1 and 5.2).

3. Fuzzy KNN

Nearest Neighbor Search, also known as proximity search, similarity search, or nearest point search, is an optimization problem for finding the closest points in metric spaces. The problem is as follows: The set S contains a number of points in a metric space such as M and a point $q \in M$ is given. The goal is to find the closest point in S to q. Often, the space M is a d-dimensional Euclidean space, and we measure the distance between points using the Euclidean distance, Manhattan distance, or other metric distances. K-nearest neighbor search returns the K nearest neighbors to the query point. Predictive analysis typically uses this method to estimate or categorize a point based on the consensus of its neighbors.

To use the fuzzy KNN method for LR-type fuzzy numbers, it is necessary to have an appropriate distance measure. Consider two LR-type fuzzy numbers, $A = (a, l_a, r_a)$ and $B = (b, l_b, r_b)$, where a and b represent their modal points, l_a and l_b represent their left spreads, and r_a and r_b represent their right spreads. The diamond distance can be defined in the following way:

$$D_0^2(A, B) = (a - b)^2 + (l_a - l_b)^2 + (r_a - r_b)^2, \tag{3.1}$$

The following is an example of a distance measurement that Yang and Ko Yang and *et al.* (1996) suggested between A and B:

$$D_1^2(A, B) = 3(a - b)^2 + L_0^2(l_a - l_b)^2 + R_0^2(r_a - r_b)^2 + 2(a - b) \times \Big(R_0(r_a - r_b) - L_0(l_a - l_b)\Big), \tag{3.2}$$

where $L_0 = \int_0^1 L^{-1}(w)\, dw$ and $R_0 = \int_0^1 R^{-1}(w)\, dw$. Diamond and Körner Diamond and Korner (1997) defined another distance measure between A and B as follows:

$$D_2^2(A, B) = (a - b)^2 + L_2(l_a - l_b)^2 + R_2(r_a - r_b)^2 + 2(a - b) \times \Big(R_1(r_a - r_b) - L_1(l_a - l_b)\Big), \tag{3.3}$$

Algorithm 1: Classification by KNN algorithm

1. Data loading
2. Determine the value of K, which is the number of nearest neighbors.
3. For each data sample:
   - Calculate the distance between the new data sample and the existing data samples.
   - Enter the distance and index of each sample into a list.
4. Sort the entire list based on the distance of the data samples, from the smallest to the largest distance.
5. The first K samples in the sorted list are selected as the K nearest neighbors.
6. Check the label of these K samples.
7. If it is a regression problem, the average of the labels of these K data samples will be the label of the new data sample.
8. If it is a classification problem, the new sample will have the same label as the majority of the K neighbors.

where $L_1 = \frac{1}{2}\int_0^1 |L^{-1}(w)|\,dw$, $R_1 = \frac{1}{2}\int_0^1 |R^{-1}(w)|\,dw$, $L_2 = \frac{1}{2}\int_0^1 |L^{-1}(w)|^2\,dw$. Darehmiraki Darehmiraki (2019) proposed the following parametric distance between fuzzy numbers A and B:

$$D_3^2(A,B) = \int_0^\alpha [(a - l_a L^{-1}(w)) - (b - l_b L^{-1}(w))]^2 dw + \int_0^\alpha [(a + r_a R^{-1}(w)) - (b + r_b R^{-1}(w))]^2 dw \tag{3.4}$$

For two trapezoidal fuzzy numbers $A = (a_l, a_m, a_u, a_r)$ and $B = (b_l, b_m, b_u, b_r)$, the Diamond distance is defined as:

$$D^2(A,B) = (a_l - b_l)^2 + 0.5(a_m - b_m)^2 + 0.5(a_u - b_u)^2 + (a_r - b_r)^2, \tag{3.5}$$

The KNN algorithm heavily relies on the distance function, and as various distance functions generate neighborhoods with varying shapes, they significantly influence the selection of a sample class. Therefore, it is crucial to test various distance functions and select the most effective one.

Algorithm 2: K-nearest Neighbors Regression

1. Select K, the number of neighbors.
2. Calculate the Euclidean distance to the K nearest neighbors.
3. Take the K nearest neighbors as per the calculated Euclidean distance.
4. Count the number of data points in each category among these K neighbors.
5. The new data point is assigned to the category with the maximum number of neighbors.

3.1 Methodology

The KNN algorithm heavily depends on the chosen distance function. Since different distance functions result in neighborhoods with varying shapes, they can significantly impact the classification of samples. Therefore, it is essential to test multiple distance functions and select the one that yields the best performance.

Algorithm 3: Classification by KNN algorithm

1. Determine the parameter k, the number of nearest neighbors.
2. Calculate the distance between the input sample and all training samples.
3. Sort training samples based on distance and select k nearest neighbors.
4. Assign the class that has the majority in the nearest neighbors as an estimate for the class of the input sample.

To further elucidate the FKNN process, Figure 3 provides a flowchart summarizing the key steps for both classification and regression tasks. The algorithm begins by computing fuzzy distances between the test instance and training data, selects the K-nearest neighbors, and branches based on the task type. For classification, it aggregates fuzzy memberships, while regression employs weighted averaging of neighbor outputs. This visual guide complements the pseudocode by highlighting decision points and task-specific operations.

Algorithm 4: Pseudocode for Fuzzy K-Nearest Neighbor Algorithm

Input : Training data D_{train}, test instance x_{test}, K, $d(\cdot, \cdot)$

Output: Predicted label or value

for *each* $(x_i, y_i) \in D_{train}$ **do**
  Compute $d(x_{\text{test}}, x_i)$;
  Store (d_i, y_i) in S;
end for
Sort S by d_i;
$N_K \leftarrow$ top K entries from S;
if *task is classification* **then**
  Assign x_{test} to majority class in N_K;
else
  Predict $\hat{y} = \frac{1}{Z} \sum_{i=1}^{K} \frac{y_i}{1+d_i}$;
end if

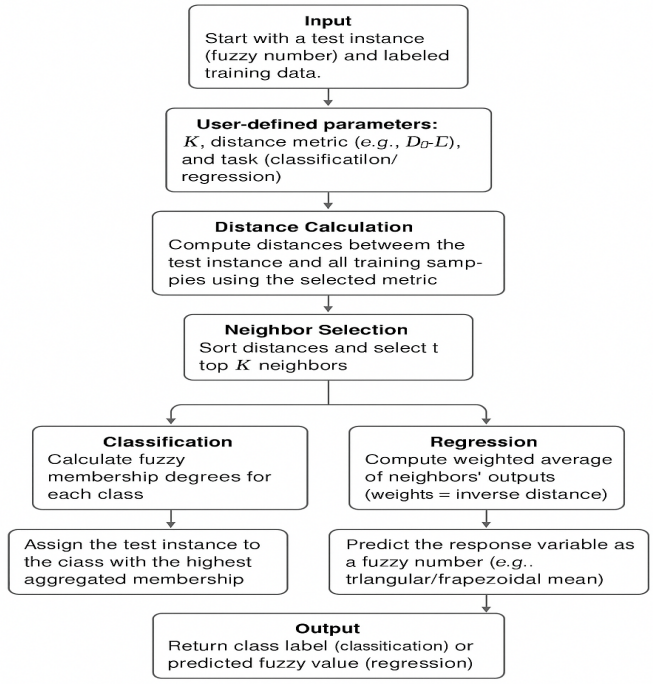


Figure 3: Flowchart of Fuzzy K-Nearest Neighbor (FKNN) for Classification and Regression

4. Computational Complexity of FKNN

The FKNN algorithm involves the following key steps, each contributing to its computational cost:

- **Operation**: Calculate distances between the test instance and all training instances using a fuzzy distance metric (e.g., Diamond distance, Yang-Ko distance).

- **Complexity**:
  For N training instances and d-dimensional features:

  1. Non-fuzzy data: $O(N \cdot d)$ (standard KNN).
  2. Fuzzy data: Each dimension is represented by parameters (e.g., triangular: (l, m, u)). Distance metrics like Equation 3.1–3.5 require computations over these parameters.
  3. Worst-case: $O(N \cdot d \cdot p)$, where p is the number of parameters per fuzzy number (e.g., $p = 3$ for triangular).

For neighbor selection:

- **Operation**: Sort distances to find K-nearest neighbors.

- **Complexity**: $O(N \log N)$ (due to sorting).

Membership Aggregation (Classification) or Weighted Averaging (Regression) Operation:

- Classification: Compute fuzzy membership degrees for each class (Equation 3.6).

- Regression: Weighted average of neighbors' fuzzy outputs.

- Complexity: $O(K \cdot c)$ for classification (where c is the number of classes) or $O(K \cdot p)$ for regression (aggregating fuzzy numbers).

For total complexity, we have

$$O(N \cdot d \cdot p) + O(N \log N) + O(K \cdot c \text{ or } p)$$

Dominated by distance computation ($O(N \cdot d \cdot p)$) and sorting ($O(N \log N)$).

4.1 K-nearest Neighbors Regression

Similar to classification, regression is a predictive problem where we aim to apply past knowledge to forecast future observations. Nevertheless, when it comes to regression, the objective is to make predictions about numerical values rather than categorical values. People often use the term "response variable" to refer to the variable they want to predict. We are able to produce predictions in regression by using a K-nearest neighbors-based strategy, which is quite similar to the approach we use in classification. By averaging nearby data, the KNN regression method (a non-parametric approach) simply approximates the association between independent variables and the continuous result. Either the analyst has to know how big the neighborhood is, or they can use cross-validation to find out what size neighborhood produces the best mean-squared error.

KNN Regression is characterized by its ease of implementation and comprehensibility, although it may provide computational challenges, particularly for large datasets, due to the need to compute distances between the new data point and all preexisting data points. Moreover, the selection of an optimal value for K and a suitable distance measure may significantly influence the accuracy of the forecasts. Cross-validation and hyperparameter tweaking may assist in selecting appropriate values for K and the distance measure. In the field of regression analysis, the K-Nearest Neighbors algorithm is often known as "KNN Regression" or "K-Nearest Neighbors Regression". Here's a concise explanation of how KNN regression works:

- Data Collection: The first dataset consists of input characteristics and corresponding goal values. In regression tasks, the target values are continuous and serve as the desired output that you want to forecast.

- Calculating the Optimal Number of Neighbors (K): To produce predictions, you must choose the value of K, which represents the number of closest neighbors to consider. You can adjust this hyperparameter based on the specific attributes of your data. A low value of K (e.g., 1 or 3) may result in forecasts that are noisy, while a high value of K can result in predictions that are excessively smoothed.

- KNN uses a distance metric, such as the Euclidean distance, to quantify the similarity between data points. Depending on the characteristics of your data, you may use various distance measures.

- Prediction: To produce a prediction for a new input data point, the KNN algorithm computes the distance between this point and all other data points in the dataset. Next, it chooses the K data points that have the shortest distances.

- Regression Prediction: Regression determines the projected value of a new data point by averaging the target values of its K closest neighbors. This might be a basic calculation of the average. Another option is to use the weighted average of this data, which translates into the following calculations:

$$\hat{Y} = \sum_{i=1}^{k} (w_i Y_i)$$

where $w_i = \dfrac{\dfrac{1}{Dis_i}}{\sum_{i=1}^{k} \dfrac{1}{Dis_i}}$

In this article, we use the following criteria to assess the goodness of fit and the model's error rate:

$$\begin{aligned} S =& \frac{1}{n} \sum_{i=1}^{n} \frac{\int \min\{\hat{w}_i(t), \tilde{\hat{w}}_i(t)\} dt}{\int \max\{\hat{w}_i(t), \tilde{\hat{w}}_i(t)\} dt}, \\ E_1 =& \frac{1}{n} \sum_{i=1}^{n} \int |\hat{w}_i(t) - \tilde{\hat{w}}_i(t)| dt, \\ E_2 =& \frac{1}{n} \sum_{i=1}^{n} \frac{\int |\hat{w}_i(t) - \tilde{\hat{w}}_i(t)| dt}{\int \hat{w}_i(t) dt}. \end{aligned} \tag{4.6}$$

S is a similarity measure, and the closeness of this measure to one indicates a high similarity between the fitted data and the real data. The other two criteria are measures of error, and the smaller they are, the better the model.

4.2 Optimal K Value

For better results, it is essential to choose the right value of K, a process known as parameter tuning. To choose the value of K, we calculate the square root, or radical, of the total number of data points in the dataset. We always choose an odd value of K to prevent confusion between the two classes.

5. Numerical example

In this section, we present two numerical examples and one practical example to demonstrate the effectiveness of the proposed method. We examine two sets of fuzzy numbers, each with 5 and 6 members, in the two numerical examples. In order to check the effectiveness of the proposed nearest neighbour algorithm, we first cluster the data using one of the HFCM or MVFCM Yang and *et al.* (2006); Hathaway and *et al.* (1996) methods or the parametric method of reference

Farnam and Darehmiraki (2023). Then we use the proposed KNN method to determine the data class in question.

Example 5.1. *We consider a dataset with five triangular fuzzy numbers. These data and their classification into two groups by methods $MVFCM$ and $HFCM$ are recorded in Table 1.*

Table 1: Triangular fuzzy numbers in Example 5.1 and their classification in two ways: $MVFCM$ and $HFCM$.

	MVFCM		H FCM	
Fuzzy data				
	u_{1k}	u_{2k}	u_{1k}	u_{2k}
$A = (-1, 0, 1)$	0.6121	0.3879	0.9666	0.0334
$B = (-3, 0, 3)$	0.7022	0.2978	0.6620	0.3380
$C = (-5, 0, 5)$	0.8337	0.1663	0.1744	0.8256
$D = (-1, 2, 5)$	0.3092	0.6908	0.3153	0.6847
$E = (-1, 4, 9)$	0.2132	0.7868	0.1234	0.8766

Suppose we have new fuzzy data $F = (-1, 1, 2)$ (Figure 4 shows a picture of the data in Table 1 and the new data $F = (-1, 1, 2)$), and we want to know to which of the clusters this new data belongs. To achieve this goal, we must first calculate the distance between this new data and all existing data. These distances, calculated using each of the relationships 3.2, 3.3, and 3.4, are recorded in Table 2. Once we have calculated the distances for all data points, we arrange them in order, identifying the k nearest neighbors with the shortest distances. As is clear in Table 2, although the values of the distances calculated by different methods are different, their rankings are the same for all these methods. Based on the closest distance to point F, the data in Table 1 are arranged as follows:

$$A,\ B,\ D,\ C,\ and\, E$$

As can be seen, if $k = 2$, the closest points to point F are A and B, and these two points belong to the first group. Among the 2 nearest neighbors, both belong to class 1; therefore, F belongs to class 1.

As shown in Table 2, although the size of the distance between each point and point $F = (-1, 1, 2)$ changes with different metrics and their magnitudes are not the same across all formulas, the order of these distances is the same. In other words, if we rank them from smallest to largest, the rank of each point does not change with each metric, and the rank of each point is the same for all these metrics.

Table 2: Distance of the new data ($F = (-1, 1, 2)$) to any of the existing data using different metrics in Example 5.1

Fuzzy data	Diamond	D_1	D_2	$D_3(\alpha = 0.25)$	$D_3(\alpha = 0.5)$	$D_3(\alpha = 0.75)$	$D_3(\alpha = 1)$
$A = (-1, 0, 1)$	1.4142	1.5	0.8165	0.5052	0.736	0.9437	1.1547
$B = (-3, 0, 3)$	2.4495	1.803	1.1547	1.0129	1.307	1.4843	1.633
$C = (-5, 0, 5)$	5.099	2.87	2.1602	2.2091	2.7462	2.9607	3.055
$D = (-1, 2, 5)$	2.4495	2.29	1.5275	1.3788	1.7911	2.0194	2.1602
$E = (-1, 4, 9)$	5.8309	6.02	3.8297	3.26	4.31	4.9513	5.4160

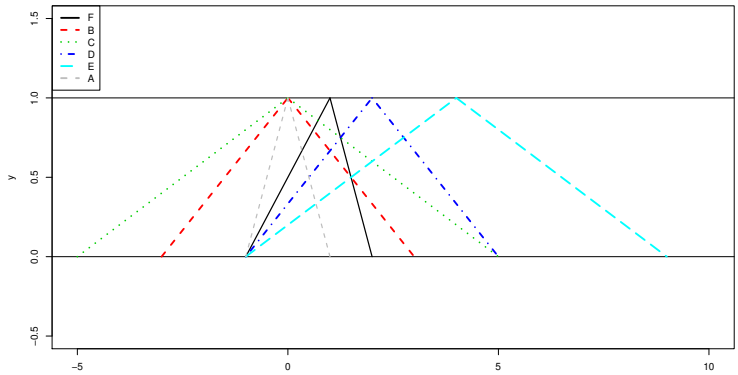


Figure 4: Triangular fuzzy numbers in Table 1 and $F = (-1, 1, 2)$ fuzzy number in Example 5.1

Example 5.2. *In this example, we use six trapezoidal fuzzy numbers. These fuzzy numbers and their classification in two methods, MVFCM and HFCM, are presented in Table 3.*

In this example, we would like to place point $G = (0.5, 1.5, 2.5, 5.5)$ *in one of these two groups using the k nearest neighbor method. Considering that the ranking of the data is the same based on all the metrics, we only calculated the distances to point G using the Diamond metric, and they are recorded in Table 4. An image of these fuzzy numbers is shown in Figure 5, where you can see how these numbers are arranged in the figure.*

Now, if we sort the data in Table 3 based on the closest distance to the fuzzy number $G = (0.5, 1.5, 2.5, 5.5)$*, we can arrange them as follows:*

$$E,\ D,\ A,\ B,\ F,\ C$$

Given that $k = 3$*, the closest points to the data G are E, D, and A, of which E and D belong to group 2 and A to group 1. Therefore, based on the three nearest neighbors, we assign G to group 2.*

Example 5.3. *This example is a practical example of clustering cars based on three attributes: price, comfort, and safety Coppi and et al. (2012). Considering*

Table 3: Trapezoidal fuzzy numbers in Example 5.2 and their classification into two groups using methods $MVFCM$ and $HFCM$.

	MVFCM		H FCM	
Fuzzy data	u_{1k}	u_{2k}	u_{1k}	u_{2k}
$A = (-1, 1, 1, 3)$	0.973	0.0266	0.8060	0.1940
$B = (-3, 1, 1, 5)$	0.9721	0.0279	0.9405	0.0595
$C = (-3, 0, 0, 3)$	0.9690	0.0310	0.9603	0.0397
$D = (2, 3, 5, 6)$	0.0231	0.9769	0.0527	0.9473
$E = (1, 3, 5, 7)$	0.0078	0.9922	0.0076	0.9924
$F = (-2, 2, 4, 8)$	0.1134	0.8866	0.5392	0.4608

Table 4: Distance between the data in Table 3 and point $G = (0.5, 1.5, 2.5, 5.5)$ in Example 5.2 using the Diamond metric.

Fuzzy data	*Diamond*
$A = (-1, 1, 1, 3)$	3.1225
$B = (-3, 1, 1, 5)$	3.7081
$C = (-3, 0, 0, 3)$	4.7697
$D = (2, 3, 5, 6)$	2.5981
$E = (1, 3, 5, 7)$	2.5981
$F = (-2, 2, 4, 8)$	3.7081

that the price of a car is a quantitative parameter, we display it with a definite number, but we use trapezoidal fuzzy numbers to display the other two parameters. Table 5 shows the value of each of these three characteristics for different cars. According to the data in Table 6, the Aston Martin, Mitsubishi Lancer, Tira Ativa, Toyota Tercel, and Nissan Micro cars belong to cluster 1, and the rest belong to cluster 2.

Now suppose we have the Fiat 128 car with the following specifications (Fiat 128 Price $= (-3.64, 0, 0, 3.64)$*, Convenience* $= (-16, -1, 1, 13)$ *and Safety* $= (-11.5, 1, 3, 15.5)$*). We would like to place this car with these features in one of the groups in Table 6. Similar to before, we first calculated the distances for each of the data in Table 5 and recorded them in Table 7. Now, if we arrange these data, we will see that the Toyota Tercel, Nissan Micro, Tira Ativa, and Mitsubishi Lancer have the least distance with the Fiat 128, respectively. On the other hand, these data are in groups 1, 1, 1 and 2, so we put the Fiat 128 car in group 1.*

Example 5.4. *Table 8 is an example where the inputs and outputs are all TriFN, and the example data is taken from Table 3 in Sakawa (1992) Sakawa (1992).*

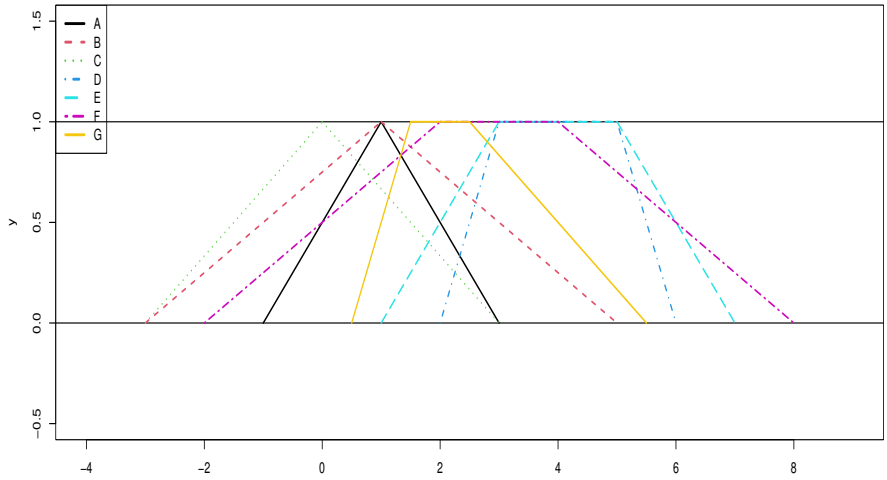


Figure 5: Trapezoidal fuzzy numbers in Table 3 and $G = (0.5, 1.5, 2.5, 5.5)$ fuzzy number in Example 5.2

Table 5: Data on the characteristics of 10 cars

Type of car	Price	Convenience	Safety
Aston Martin	$(-9.63, 0, 0, 9.63)$	$(-9, 1, 3, 13)$	$(-7.5, 1.5, 4.5, 13.5)$
Mitsubishi Lancer	$(-9.51, 0, 0, 9.51)$	$(-5, 1, 3, 9)$	$(-4.5, 1.5, 4.5, 10.5)$
Mitsubishi Galant	$(-8.71, 0, 0, 8.71)$	$(-15, -1, 1, 11)$	$(-6.67, -1.5, 1.5, 6.62)$
Tira Ativa	$(-9.46, 0, 0, 9.46)$	$(-5, 1, 3, 9)$	$(-4.5, 1.5, 4.5, 10.5)$
M2000	$(-6.64, 0, 0, 6.64)$	$(-7, 1, 3, 11)$	$(-7.5, 1.5, 4.5, 13.5)$
Toyota Tercel	$(-8.45, 0, 0, 8.45)$	$(-4, 2, 2, 4)$	$(-4.5, 1.5, 4.5, 10.5)$
Toyota Corolla	$(-3.74, 0, 0, 3.74)$	$(-15, -1, 1, 11)$	$(-10.5, 1.5, 4.5, 16.5)$
Toyota Prius	$(-9.72, 0, 0, 9.72)$	$(-9, 1, 3, 13)$	$(-6.67, -1.5, 1.5, 6.62)$
Nissan Laurel Altima	$(-9.69, 0, 0, 9.69)$	$(-7, 1, 3, 11)$	$(-10.5, 1.5, 4.5, 16.5)$
Nissan Micro	$(-9.39, 0, 0, 9.39)$	$(-4, 2, 2, 4)$	$(-2.5, 3, 3, 8)$

Table 10 shows the estimated values using the K-nearest neighbor method (with different values of K). This method estimates the value of the dependent variable Y for the independent variable X. Figure 1 also shows a picture of observed values and estimated values for different values of K.

Now suppose we want to predict the value of Y for $X_1 = (6, 1)_T$. For this purpose, we first calculate the distance of point X_1 from all available points. These values are recorded in Table 10. As you can see, points 3, 4, 5, and 2 are the closest to point X. Therefore, using these values and $k = 4$, the value of $\hat{Y}$ is equal to $(7, 0.75)$. It is also possible to use the weighted average of these data, resulting in an estimate of $(7.14, 0.82)$. If we apply the least squares method to fit the regression

Table 6: Classification of the data in Table 5 by Farnam and Darehmiraki (Farnam and Darehmiraki , 2023) method

Type of Car	Cluster 1 ($\alpha = 0.3$)	Cluster 2 ($\alpha = 0.3$)
Aston Martin	0.5425	0.4575
Mitsubishi Lancer	0.6749	0.3251
Mitsubishi Galant	0.3325	0.6675
Tira Ativa	0.7059	0.2941
M2000	0.4595	0.5405
Toyota Tercel	0.6893	0.3137
Toyota Corolla	0.3485	0.6515
Toyota Prius	0.3335	0.6665
Nissan Laurel Altima	0.2646	0.7354
Nissan Micro	0.6266	0.3734

Table 7: Distance between Fiat 128 and cars in Table 5

Type of Car	Price	Convenience	Safety	Distance
Aston Martin	8.4711	21.7543	4.6098	23.7962
Mitsubishi Lancer	8.3014	17.5285	8.6747	21.2465
Mitsubishi Galant	7.17	17.6983	10.3166	21.7042
Tira Ativa	8.2307	17.5285	8.6747	21.2189
M2000	4.2426	20.6700	4.6098	21.5986
Toyota Tercel	6.8024	14.8745	8.6747	18.5141
Toyota Corolla	0.1414	21.5232	1.8028	21.5991
Toyota Prius	8.5981	20.9740	10.3167	24.9053
Nissan Laurel Altima	8.556	23.3934	1.8028	24.9741
Nissan Micro	8.1317	13.8293	11.8004	19.9154

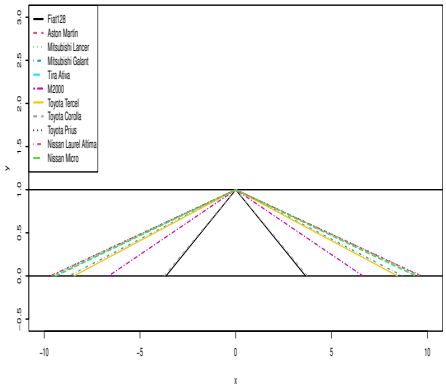
Table 8: Sample data and the estimated value in 5.4.

No.	X	Y
1	$(2; 0.5)_T$	$(4.5, 0.5)_T$
2	$(3.5; 0.5)_T$	$(5.5, 0.5)_T$
3	$(5.5, 1)_T$	$(7.5, 1)_T$
4	$(7, 0.5)_T$	$(6.5, 0.5)_T$
5	$(8.5, 0.5)_T$	$(8.5, 1)_T$
6	$(10.5, 1)_T$	$(8, 1)_T$
7	$(11, 0.5)_T$	$(10.5, 0.5)_T$
8	$(12.5, 0.5)_T$	$(9.5, 0.5)_T$

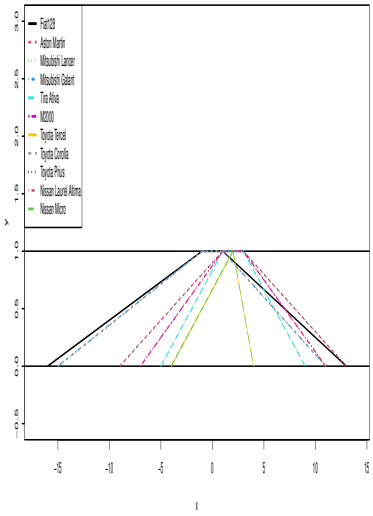
model, we will obtain the following model:

$$y = (3.85, 0.17)_T + (0.49, 0.83)_T * x.$$

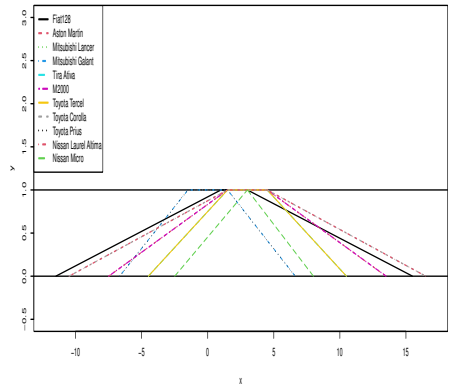
Using this line, the fitted values have been calculated and recorded in Table 12.



(a) Price characteristic membership function diagram for the data in Table 5.



(b) Convenience characteristic membership function diagram for the data in Table 5.



(c) Safety characteristic membership function diagram for the data in Table 5.

Figure 6: The membership function of the different characteristics of the data machines in Table 5.

The results show that in this particular case, the least squares method performs better than the nearest neighbor method. In this example, the results showed that

Table 9: The fitted values of Table 8 using the K-nearest neighbor method for different values of K

		$K=3$		$K=5$		$K=7$	
No.	X	Ordinary	Weighted	Ordinary	Weighted	Ordinary	Weighted
1	$(2;0.5)_T$	$(6.5,0.67)_T$	$(6.17,0.62)_T$	$(7.2,0.8)_T$	$(8,0.71)_T$	$(7.01,0.67)_T$	$(6.15,0.59)_T$
2	$(3.5;0.5)_T$	$(6.17,0.67)_T$	$(5.92,0.67)_T$	$(7,0.8)_T$	$(7.86,0.71)_T$	$(6.81,0.71)_T$	$(6,0.68)_T$
3	$(5.5,1)_T$	$(6.83,0.67)_T$	$(6.61,0.61)_T$	$(6.6,0.7)_T$	$(7.57,0.64)_T$	$(6.96,0.62)_T$	$(6.57,0.58)_T$
4	$(7,0.5)_T$	$(7.17,0.83)_T$	$(7.56,0.91)_T$	$(8,0.8)_T$	$(7.71,0.71)_T$	$(7.8,0.82)_T$	$(7.89,0.91)_T$
5	$(8.5,0.5)_T$	$(8.33,0.67)_T$	$(8,0.66)_T$	$(8.4,0.7)_T$	$(7.43,0.64)_T$	$(7.67,0.66)_T$	$(7.62,0.66)_T$
6	$(10.5,1)_T$	$(9.5,0.67)_T$	$(9.95,0.59)_T$	$(8.5,0.7)_T$	$(9.49,0.61)_T$	$(7.5,0.64)_T$	$(9.16,0.6)_T$
7	$(9,0.83)_T$	$(8.67,0.83)_T$	$(8.43,0.88)_T$	$(8,0.8)_T$	$(7.14,0.71)_T$	$(8,0.83)_T$	$(8.15,0.93)_T$
8	$(12.5,0.5)_T$	$(9,0.83)_T$	$(9.27,0.76)_T$	$(8.2,0.8)_T$	$(7.28,0.71)_T$	$(8.43,0.73)_T$	$(9.23,0.7)_T$

Table 10: The distance of point X from the data in Table 8.

No.	X	D_1	D_2
1	$(2;0.5)_T$	6.9372	4.0104
2	$(3.5;0.5)_T$	4.3445	2.5166
3	$(5.5,1)_T$	0.8660	0.5
4	$(7,0.5)_T$	1.7678	1.0408
5	$(8.5,0.5)_T$	4.3445	2.5166
6	$(10.5,1)_T$	7.7942	4.5
7	$(11,0.5)_T$	8.6674	5.008
8	$(12.5,0.5)_T$	11.2638	6.5064

Table 11: Actual values and fitted values using the least squares method for the data in Example 5.4.

No.	X	Y	$\hat{Y}$
1	$(2;0.5)_T$	$(4.5,0.5)_T$	$(4.83,0.58)_T$
2	$(3.5;0.5)_T$	$(5.5,0.5)_T$	$(5.56,0.57)_T$
3	$(5.5,1)_T$	$(7.5,1)_T$	$(6.54,0.99)_T$
4	$(7,0.5)_T$	$(6.5,0.5)_T$	$(7.28,0.57)_T$
5	$(8.5,0.5)_T$	$(8.5,1)_T$	$(8.01,0.57)_T$
6	$(10.5,1)_T$	$(8,1)_T$	$(8.99,0.99)_T$
7	$(11,0.5)_T$	$(10.5,0.5)_T$	$(9.24,0.57)_T$
8	$(12.5,0.5)_T$	$(9.5,0.5)_T$	$(9.97,0.57)_T$

the least squares method performs better in the linear mode. We would like to highlight the K-NN regression algorithm's ability to function well with non-linear connections, making it effective in situations where the relationship is not a straight line. The algorithm uses the closest neighbors to make predictions about values. In reality, the algorithm makes very few assumptions about the structure of the data

Table 12: Some goodness-of-fit criteria for the two methods: nearest neighbor and least squares for the data in Example 5.4.

	Least Squares	$K=3$		$K=5$		$K=7$	
	Least Squares	Ordinary	Weighted	Ordinary	Weighted	Ordinary	Weighted
E_1	0.1025	0.29	0.31	0.3	0.27	0.2925	0.3087
E_2	0.1764	0.3972	0.4401	0.3951	0.4054	0.4179	0.4402
S	0.8628	0.6637	0.6485	0.6531	0.6764	0.6584	0.6530

it needs to function properly.

The K-NN regression method, much like the K-NN classification algorithm (or any other prediction algorithm for that matter), boasts both strengths and flaws. Some of them are listed below:

There are a number of strengths associated with the K-nearest neighbors regression approach, including its simplicity and intuitiveness, its need for minimal assumptions on the appearance of the data, and its ability to operate well with non-linear relationships (that is, relationships that are not linear). The K-nearest neighbors regression has some drawbacks. It can become increasingly sluggish as the size of the training data increases; it may not perform well with a high number of predictors; and it may not predict well outside the range of values supplied in the training data. In regression, using the nearest neighbor is very useful, especially when the regression model is unclear or there are outliers. All of these represent potential risks associated with the prediction process.

Example 5.5. *In order to better understand the capabilities of the proposed method, in this example, Table 13 shows a comparison between the algorithm proposed in this paper and the algorithm proposed in the paper Biswas and et al. (2018) for 3 datasets.*

Table 13: Comparison between the proposed method and the PIFW method Biswas and *et al.* (2018).

Dataset	PIFW	Proposed Method
Iris	95.56	97.78
Breast Cancer	96.14	97.08
Ionosphere	94.34	84.91

5.1 Comparative Analysis of Distance Metrics in FKNN

To evaluate how different distance metrics influence FKNN's performance, we compare the metrics defined in Section 3 (D_0, D_1, D_2, and D_3) using the datasets from Examples 5.1 (TFNs) and 5.2 (TrFNs). Table 14 summarizes the results, focusing on:

1. Rank Consistency: All metrics preserve neighbor rankings (as noted in Examples 5.1–5.2), ensuring stable classification.

2. Computational Cost: Parametric metrics (e.g., D_3) incur higher overhead due to integral calculations but offer tunable precision.

3. Sensitivity to Spread: Metrics like D_1 (Yang-Ko) penalize spread discrepancies more aggressively than the Diamond distance (D_0).

Table 14: Performance of FKNN under different distance metrics.

Metric	Type	Rank Consistency	Sensitivity to Spread	Best Use Case
Diamond (D_0)	Non-parametric	High	Low	Symmetric TFNs/TrFNs
Yang-Ko (D_1)	Non-parametric	High	High	Asymmetric LR-type numbers
Korner (D_2)	Non-parametric	High	Moderate	Noisy or overlapping classes
Parametric (D_3)	Tunable	High	Adjustable (α)	Customizable decision levels

6. Conclusion

Fuzzy k-nearest neighbor has several advantages and disadvantages. One advantage is that FKNN takes advantage of the fuzzy dominance relation between instances, which helps in constructing monotonic classifiers. This allows FKNN to decrease the disturbance caused by noisy data and improve the selection range of the nearest neighbors. Another benefit is that FKNN can effectively handle datasets with class imbalances and outliers.
In this paper, we tried to improve the efficiency of the fuzzy nearest neighbor algorithm by presenting a parametric distance measure and giving decision-makers the power to solve their desired problem at the level of decision they want.
The fuzzy k-nearest neighbor algorithm is a classification approach that leverages the fuzzy dominance connection between instances. We use this method to develop monotonic classifiers, accounting for the degrees of fuzzy dominance connection between pairs of cases, especially those instances that are not comparable to each other. The method aims to minimize noise data disruption, which could potentially affect the selection range of the k-nearest neighbors. Modifying the thresholds of the fuzzy dominance relation degrees accomplishes this. We will investigate regression trees, splines, and general local regression techniques with fuzzy data using KNN in the course of our future studies.

References

Altman, N. S. (1992). An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *The American Statistician*, **46(3)**, 175–185.

Atkinson, P. M., and Tatnall, A. R. (1998). Introduction to Neural Networks for End-Users. *International Journal of Geographical Information Science*, **12(3)**, 275–287.

Bian, Z., Vong, C. M., Wong, P. K., and Wang, S. (2020). Fuzzy KNN Method with Adaptive Nearest Neighbors. *IEEE Transactions on Cybernetics*, **52(6)**, 5380–5393.

Biswas, N., Chakraborty, S., Mullick, S. S., and Das, S. (2018). A Parameter Independent Fuzzy Weighted K-Nearest Neighbor Classifier. *Pattern Recognition Letters*, **101**, 80–87.

Cleveland, W. S. (1979). Robust Locally Weighted Regression and Smoothing Scatterplots. *Journal of the American Statistical Association*, **74(368)**, 829–836.

Cover, T. M., and Hart, P. E. (1967). Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, **13(1)**, 21–27.

Coppi, R., D'Urso, P., and Giordani, P. (2012). Fuzzy and Possibilistic Clustering for Fuzzy Data. *Computational Statistics & Data Analysis*, **56(4)**, 915–927.

De Castro, G. F., and Tinós, R. (2022, November). K-Nearest Neighbors Based on the Nk Interaction Graph. *In Anais do XIX Encontro Nacional de Inteligência Artificial e Computacional*, 694–703.

Dubois, D., and Prade, H. (1990). Rough Fuzzy Sets and Fuzzy Rough Sets. *International Journal of General Systems*, **17(2–3)**, 191–209.

Diamond, P., and Körner, R. (1997). Extended Fuzzy Linear Models and Least Squares Estimates. *Computers & Mathematics with Applications*, **33(9)**, 15–32.

Darehmiraki, M. (2019). A Novel Parametric Ranking Method for Intuitionistic Fuzzy Numbers. *Iranian Journal of Fuzzy Systems*, **16(1)**, 129–143.

Farnam, M., and Darehmiraki, M. (2023). A Parametric Method for Fuzzy Data Clustering. *Fuzzy Systems and Its Applications*, **5(2)**, 93–119.

Hathaway, R. J., Bezdek, J. C., and Pedrycz, W. (1996). A Parametric Model for Fusing Heterogeneous Fuzzy Data. *IEEE Transactions on Fuzzy Systems*, **4(3)**, 270–281.

Keller, J. M., Givens, J. R., and Grey, W. M. (1985). A Fuzzy K-Nearest Neighbor Algorithm. *IEEE Transactions on Systems, Man, and Cybernetics*, **15(4)**, 580–585.

Keller, J. M., Pettenati, M. C., andGivens, J. R. (1996). A Fuzzy K-Nearest Neighbor Algorithm for Regression. *IEEE Transactions on Fuzzy Systems*, **4(4)**, 430–440.

Kumbure, M. M., Luukka, P., and Collan, M. (2020). A New Fuzzy K-Nearest Neighbor Classifier Based on the Bonferroni Mean. *Pattern Recognition Letters*, **140**, 172–178.

Linden, G., Smith, B., and York, J. (2003). Amazon.com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing*, **7(1)**, 76–80.

Li, W., Chen, Y., and Song, Y. (2020). Boosted K-Nearest Neighbor Classifiers Based on Fuzzy Granules. *Knowledge-Based Systems*, **195**, 105606.

Li, Y., He, X., and Liu, X. (2023). Fuzzy Multiple Linear Least Squares Regression Analysis. *Fuzzy Sets and Systems*, **459**, 118–143.

Maillo, J., García, S., Luengo, J., Herrera, F., and Triguero, I. (2019). Fast and Scalable Approaches to Accelerate the Fuzzy K-Nearest Neighbors Classifier for Big Data. *IEEE Transactions on Fuzzy Systems*, **28(5)**, 874–886.

Michalski, R. S., Carbonell, J. G., and Mitchell, T. M. (1983). Machine Learning: An Artificial Intelligence Approach. Morgan Kaufmann Publishers.

Müller, P. (2023). Flexible K Nearest Neighbors Classifier: Derivation and Application for Ion-mobility Spectrometry-Based Indoor Localization. arXiv preprint arXiv:2304.10151.

Memis, S., Enginoglu, S., and Erkan, U. (2022). Fuzzy Parameterized Fuzzy Soft K-Nearest Neighbor Classifier. *Neurocomputing*, **500**, 351–378.

Patel, H., and Thakur, G. S. (2019). An Improved Fuzzy K-Nearest Neighbor Algorithm for Imbalanced Data Using Adaptive Approach. *IETE Journal of Research*, **65(6)**, 780–789.

Ramadhani, F., Satria, A., and Sari, I. P. (2023). Implementasi Metode Fuzzy K-Nearest Neighbor dalam Klasifikasi Penyakit Demam Berdarah. *Hello World Journal Ilmu Komputer*, **2(2)**, 58–62.

Sakawa, M., (1992). Multiobjective Fuzzy Linear Regression Analysis for Fuzzy Input–Output Data. *Fuzzy Sets and Systems*, **47**, 173–181.

Wackernagel, H. (2003). *Multivariate Geostatistics: An Introduction with Applications*, Springer.

Wongkhuenkaew, R., Auephanwiriyakul, S., Theera-Umpon, N., Teeyapan, K., and Yeesarapat, U. (2023). Fuzzy K-Nearest Neighbor Based Dental Fluorosis Classification Using Multi-Prototype Unsupervised Possibilistic Fuzzy Clustering via Cuckoo Search Algorithm. *International Journal of Environmental Research and Public Health*, **20(4)**, 3394.

Yacoub, M. H., Ismail, S. M., Said, L. A., Madian, A. H., and Radwan, A. G. (2022). *Generic Hardware Realization of K Nearest Neighbors on FPGA*. In 2022 International Conference on Microelectronics (ICM) (pp. 169–172). IEEE.

Yang, M. S., Hung, W. L., and Cheng, F. C. (2006). Mixed-Variable Fuzzy Clustering Approach to Part Family and Machine Cell Formation for GT Applications. *International Journal of Production Economics*, **103(1)**, 185–198.

Yang, M. S., and Ko, C. H. (1996). On a Class of Fuzzy C-Numbers Clustering Procedures for Fuzzy Data. *Fuzzy Sets and Systems*, **84(1)**, 49–60.

Zhai, J., Qi, J., and Zhang, S. (2021). An Instance Selection Algorithm for Fuzzy K-Nearest Neighbor. *Journal of Intelligent & Fuzzy Systems*, **40(1)**, 521–533.

Zheng, C., Mohd Kasihmuddin, M. S., Gao, Y., Chen, J., and Mansor, M. A. (2023). A Modified Fuzzy K-Nearest Neighbor Using the Improved Sparrow Search Algorithm for Two-Classes and Multi-Classes Datasets. In Proceedings of the 2023 2nd Asia Conference on Algorithms, Computing and Machine Learning (pp. 222–227).