ISSN 2007-9737

# Compensatory Fuzzy Logic Genetic Algorithm for Classification Problems: A Case Study

José Fernando Padrón-Tristán[1], Laura Cruz-Reyes[1], Rafael Alejandro Espín-Andrade[2],
Carlos Eric Llorente-Peralta[1,*], Fausto Antonio Balderas-Jaramillo[1], Jessica González-San-Martín[1]

[1] Tecnológico Nacional de México,
Instituto Tecnológico de Ciudad Madero,
Division of Graduate Studies and Research,
Ciudad Madero, Tamaulipas,
Mexico

[2] Universidad Autónoma de Coahuila,
Unidad Torreón, Facultad de Contaduría y Administración,
Dirección de Posgrado,
Torreón, Coahuila,
Mexico

jose.pt@cdmadero.tecnm.mx

**Abstract.** This article presents an approach for creating fuzzy predicates using genetic algorithms. The proposed method incorporates an internal genetic algorithm to optimize the membership functions of the linguistic variables involved in the discovered predicates, taking advantage of statistical data for the initialization of the population and taboo and weighted roulettes for the construction of the predicates. The generation of fuzzy predicates is based on the implication and equivalence operators, as well as on deductive structures, such as modus ponens. Furthermore, the evaluation of predicates on data sets is based on For All and Exists quantifier operators, which also guide the search for the best predicates according to their truth values. Furthermore, the popular Iris database is used as a case study to demonstrate the effectiveness and applicability of this approach.

**Keywords.** Compensatory fuzzy logic, genetic algorithms, fuzzy inference, fuzzy interpretability, iris problem.

## 1 Introduction

Fuzzy logic has proven to be a tool for modeling uncertainty and imprecision in complex systems. However, manually creating fuzzy predicates can be laborious and error-prone. In this context, genetic algorithms (GA) have emerged as a technique for optimization and finding solutions to complex problems.

Optimisation is another field where Fuzzy logic stands out because it helps decision-makers to solve optimisation problems considering the uncertainty that commonly occurs in application domains [1].

In this paper, an approach for creating fuzzy predicates using GA is presented, which includes an internal genetic algorithm to optimize the membership functions of linguistic states. Our method is based on the use of statistical data for the initialization of the population and the use of taboo and weighted roulettes for the construction of fuzzy predicates.

Furthermore, the implication and equivalence operators are implemented, as well as the modus ponens deductive structure to compare the accuracy and precision in the classification of the generated fuzzy predicates. Likewise, the parameters of the membership functions that define the optimized linguistic states that are included in the constructed predicates are analyzed. To evaluate and validate this approach, we used the Iris database, which contains trait

doi: 10.13053/CyS-28-3-5183

measurements of different flower species and which are separated into three different types of this family, as a case study. The experimental results show that the proposed method is capable of generating efficient and accurate fuzzy predicates for flower species classification in the Iris database.

In summary, this article presents a contribution to the field of compensatory fuzzy logic and genetic algorithms, providing techniques for the automated creation of fuzzy predicates with applications in a wide range of real-world problems.

## 2 Fuzzy Logic

Fuzzy logic is a formal system of logic that deals with propositions that can have truth-values, which are intermediate values between false and true, representing degrees of truth. In contrast to classical binary logic, where the result of evaluating a proposition is absolute, whether true or false, fuzzy logic allows for the representation and reasoning about uncertainty and imprecision in data [2].

Fuzzy logic (FL) is based on the idea that the way human thinking is constructed is not through numbers, but rather through linguistic labels. Linguistic terms are inherently less precise than numerical data, but they express knowledge in terms more accessible to human understanding [3, 4].

Vagueness and uncertainty can be considered using the fuzzy set theory proposed by L. Zade [5]. The fundamental concept in fuzzy set theory is the concept of the membership function [6]:

Let $M$ be a set, let $x$ be an element of $M$, then a fuzzy subset A of M is defined as a set of ordered pairs $\{x, \mu A(x)\}, \forall x \in M\ \mu A(x)$ which is a membership function characteristic that takes its values n a well-ordered set $E$, which indicates the degree or level of membership of an element $x$ to a subset $A$.

### 2.1 Compensatory Fuzzy Logic

Compensatory fuzzy logic (CFL) is an approach to multivalent logic different from the axiomatic norm (conjunction) and conorm (disjunction) approaches, which define functions of operations on fuzzy sets. The CFL has characteristics that allow it to be a support for decision-making [7].

The CFL is made up of a quatrain of continuous operators: conjunction (c), disjunction (d), strict fuzzy order (o), and negation (n), where:

$$\begin{gathered} c[0,1]^n \rightarrow [0,1] \\ d[0,1]^n \rightarrow [0,1] \\ n[0,1]^n \rightarrow [0,1] \\ o[0,1]^n \rightarrow [0,1] \end{gathered}$$

These operators satisfy the group of axioms for the FL, to which those of compensation and veto are added [7].

The compensation axiom states that, for the particular case of two components, the fact that the value of the operator is between the minimum and the maximum can be interpreted as the second value compensating the value of the first in the veracity of the conjunction. The idea is generalized to the case of $n$ components [8].

The veto axiom grants any basic predicate of conjunction the ability to veto, that is, the ability to prevent any form of compensation when its value is equal to zero [8].

CFL satisfies the idea that under certain terms compensation is permissible, which is an interesting approach to modeling human decision-making.

### 2.2 Fuzzy Predicates

According to [6], a fuzzy predicate is a function $G(z_1, z_2, \dots, z_n)$, defined in fuzzy variables $z_1, z_2, \dots, z_n$ whose range of values is a statement whose truth is estimated by values of the interval (0, 1). This true value is obtained by the composition of fuzzy variables (known as individual fuzzy predicates) and fuzzy logic operators.

### 2.3 Quantifier Operators

The most common way to deal with precision in the inference of fuzzy systems is error-based [9, 10]. To calculate the precision of fuzzy predicates, the universal quantifier ***For All*** is used:

$$\forall_{x,y \in X,Y} = e^{-\sum \ln[P(x,y)]}. \tag{1}$$

Another quantifier used is Exists, which indicates that in a set of data, there are instances for which the evaluated predicate is fulfilled:

$$\exists_{x,y \in X,Y} = e^{-\sum ln[1-P(x,y)]}. \qquad (2)$$

In this work, all operator is used to guide the search for the best predicates discovered.

## 3 Fuzzy Inference Systems

Through inference, observations of the world are used to discover unobserved facts or to identify causal effects from the data collected. In a broad sense, inference ranges from implication to the operational mental process that allows reaching a conclusion based on certain information [11].

From formal logic, inference has been understood as "the passage from one set of propositions to another; the first set can be called the class of premises and the second the class of conclusions." This definition presents inference as a syllogistic structure, that is, a three-level logical entity: two premises and a conclusion. Thus, inference not only reveals unobserved facts, but also follows a logical structure that facilitates the derivation of conclusions from given premises [12].

### 3.1 Mamdani Fuzzy Inference System

The Mamdani fuzzy inference system (FIS), introduced by Ebrahim Mamdani in 1975, stands as one of the earliest and most prevalent systems in fuzzy logic. It serves as a technique for framing control issues using fuzzy logic principles, emulating human decision-making processes. This model is renowned for its simplicity and its ability to intuitively encapsulate expert Knowledge [13].

This model is characterized by its simplicity and the intuitive way it represents expert knowledge. Stages of the fuzzy inference system Mamdami based are (figure 1):

1. Fuzzification: crisp input variables are converted into fuzzy sets through a MF. For each input is obtained how much belongs to a fuzzy set.
2. Rule Evaluation: The system uses a set of if-then rules that describe how input values translate into outputs. These rules are expressed in terms of fuzzy logic.

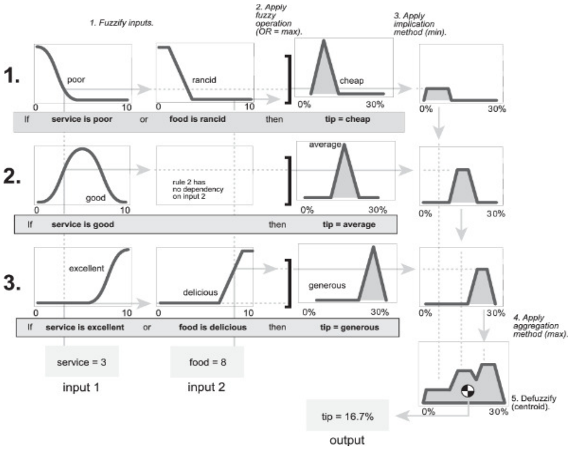
1. Fuzzify inputs.
2. Apply fuzzy operation (OR = max).
3. Apply implication method (min).
1.
poor
0
10
rancid
0
10
cheap
0%
30%
0%
30%
If service is poor or food is rancid then tip = cheap
2.
good
0
10
rule 2 has no dependency on input 2
average
0%
30%
0%
30%
If service is good then tip = average
3.
excellent
0
10
delicious
0
10
generous
0%
30%
0%
30%
4. Apply aggregation method (max).
If service is excellent or food is delicious then tip = generous
service = 3
input 1
food = 8
input 2
5. Defuzzify (centroid).
0%
30%
tip = 16.7%
output

**Fig.1.** Stages of the Mamdani’s FIS [14]

3. Aggregation of Rule Outputs: Rules are evaluated and their fuzzy outputs are combined into a single fuzzy output set.
4. Defuzzification: The fuzzy output is converted back into a crisp value. Common defuzzification methods include the centroid, weighted average, and maximum height methods.

### 3.2 Takagi-Sugeno FIS

The Sugeno fuzzy inference model, also referred to as the Takagi-Sugeno-Kang (TSK) model, was developed by Takagi and Sugeno in 1985. In contrast to the Mamdani model, which produces fuzzy set outputs for its rules, the Sugeno model generates outputs that are linear functions of the inputs [15].

Stages of the TSK model are listed next (figure 2):

1. **Fuzzification**: Similar to the Mamdani model, crisp inputs are converted into fuzzy sets using membership functions.
2. **Rule Evaluation**: The if-then rules in the Sugeno model are of the form: "If x is A and y is B, then z = f(x, y)," where f(x,y) is a polynomial function.
3. **Aggregation of Rule Outputs**: The rule outputs are crisp functions (linear or constant) and are aggregated through a weighting process.

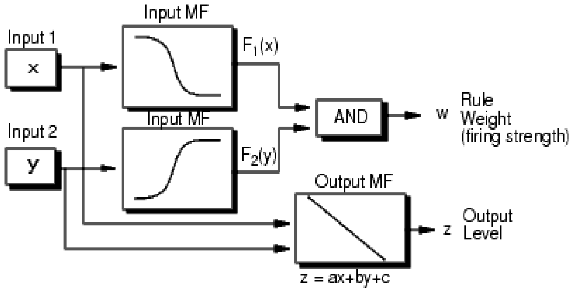

**Fig. 2.** TSK model [14]

4 **Defuzzification**: Instead of defuzzifying a fuzzy set, the Sugeno model produces a crisp output directly by the weighted combination of the rule outputs.

### 3.3 Comparison of Models Mamdani with TSK

Table 1 shows a comparison of the Mamdani model with the Sugeno model:

Analyzing the advantages and disadvantages that appear in both models, it was decided to work with the Mamdani model, since it presents the following characteristics [13, 5]:

1 **Intuitive rule representation:** The rules in the Mamdani model are formulated in natural language, such as "if x is A and y is B then z is C". This structure closely mirrors human reasoning, making the system more understandable for domain experts who may not have a background in mathematics.

2 **Richness of output expression:** The Mamdani model produces outputs as fuzzy sets, enabling a more nuanced and expressive depiction of uncertainty and partial truths.

3 **Ease of rules definition and maintenance:** Since the rules are based on linguistic terms that directly correspond to expert knowledge and real-world observations, the system is easier to update and expand.

4 **Suitable for complex systems with multiple inputs and outputs:** This feature makes the model ideal for applications where the relationships between variables are difficult to quantify.

## 4 Classification Task

In this work binary classification is performed, which means that the GA generates a predicate that determines if an instance in the dataset belongs to a category or not.

If the instance belongs to the searched category, then is called a positive instance, otherwise it is considered a negative case.

GA discovers the best predicates according to their truth value and is executed once per class in the dataset. If the dataset contains $m$ classes, then the GA is performs $m$ times the discovery of predicates to find the best ones corresponding to each class. To do this, the corresponding $m$ symbolic predicates are created as part of the GA configuration, which are listed as follows:

$$(IMP\ p_1(x)\ class_1)$$
$$(IMP\ p_2(x)\ class_2)$$
$$\vdots$$
$$(IMP\ p_m(x)m\ class_m).$$

For each instance, the classification task takes the best predicate according to its truth value to extract the premise or the generated sub-predicates $p_j(x)$ to be evaluated and compared with the others.

The $p_j(x)$ with the maximum truth value, is selected to get the class obtained for the instance.

The accuracy for the classification task is given for [16]:

$$Acc = \frac{tp+tn}{tp+fp+tn+fn}, \tag{3}$$

where:

$tp$: (true positive) positive instances correctly classified.

$tn$: (true negative) negative instances correctly classified.

$fp$: (false positive) instances incorrectly classified as positives.

$fn$: (false negatives) instances incorrectly classified as negatives.

## 5 State of the Art

Data classification is a widely researched area in pattern recognition. It involves dividing the data space, known as pattern space, into different classes. Among various applications, there are: image processing, classification and segmentation; voice recognition and restoration; signal processing and, among others, financial data analysis; human action recognized in video [17]; facial geometry identification [18].

The design of a classifier is based on a set of data divided into categories, which comprises data vectors along with their expected labels, known as labeled data [19].

A well-designed classifier must have a property known as generalization, which is to accurately classify new and unseen data with a low error rate [19]. To this end, several methods have been used for several years, including those based on distance, statistical methods, neural networks, and fuzzy logic (FL) [20].

FL was introduced by Lofti A. Zadeh [5, 21], which uses membership functions to model knowledge. Membership functions quantify the degree to which a feature of data satisfies an attribute, with truth values ranging between 0 and 1 in FL.

Data acquisition has improved and accelerated through technological advances in recent decades. Likewise, improvements in hardware have been used to propose more computationally intensive algorithms for data processing. But despite the successes obtained in pattern recognition, approaches are required that facilitate the discovery of knowledge and allow the analysis of the extensive information contained in the data [22].

An approach that has gained strength in recent years is the automation of the definition of membership functions (MF) and the generation of predicates from a data set [23, 24] This approach allows the analysis of MFs and predicates eliminating the need for expert knowledge in the use of FL.

Unlike the traditional way, in which an FL model was defined based on expert knowledge, it is now derived automatically from the data, this being a crucial aspect. According to Zadeh [25], MFs and predicates encapsulate the meanings of expressions in natural language, making their interpretation closely linked to knowledge. Preserving the semantic meaning of natural language expressions to achieve interpretable descriptions is essential; Otherwise, pattern recognition-based methods could supplant FL-based models.

**Table 1.** Comparison of models Mamdani and TSK

| Attribute | Mamdani | Sugeno |
|---|---|---|
| Efficient | Less efficient | More efficient |
| Computational Efficiency | Intense | Reduced computational cost |
| Precision | Limited precision | Higher precision |
| Intuitiveness | Easy to understand; accessible to domain experts | More difficult to understand; for experts with mathematical training |
| Flexibility | Can handle systems with multiple inputs and outputs | Less flexible; to define the output functions can be difficult |

On the other hand, the use of the CFL presents a series of advantages such as the following [7]:

Compensation in fuzzy predicate evaluations: If a value does not completely meet one criterion, but largely meets another, these values can compensate for each other, providing a more balanced and realistic approach in modeling complex systems.

Adaptability: can better adjust to the nuances of the relationships between variables.

More informed decisions: Provides a stronger basis for decision making, especially in situations where decision criteria are conflicting or are not equally important.

Flexibility: reduce the inflexibility of decisions based solely on strict individual criteria.

In addition to these advantages, in this work the generalized membership function [10] (GMF, section 6.2) was used, which is a flexible function, with which any linguistic state can be represented, since it can take a sigmoidal, sigmoidal form.

negative, Gaussian or concave, based on its parameters.

In this work, techniques were also explored to control the length and consistency of fuzzy predicates and types of predicates for the classification of instances in a data set were also explored, such as those that use the implication operator, the equivalence and also, the deductive structure Modus Ponens, complementing it with the existential quantifier.

## 6 Proposal of Solution

The objective of this work is to prove that a genetic algorithm that uses CFL provides a solid methodology that allows for solving classification problems through fuzzy predicates.

To achieve this objective there is a dataset that contains a register of 150 instances of iris flower and measurements of sepal and petal of the flower. This dataset and a set of parameters of configuration are the input of a genetic algorithm, which implements a series of techniques to generate predicates that classify what type of iris is a flower according to its attributes.

1 The dataset and a configuration set are the input of the GA, including a symbolic predicate that establishes what logical operator of equivalence or implication or a deductive structure is going to be used.
2 The GA creates fuzzy predicates according to the input optimizing the parameters of the membership functions.
3 The GA outputs a list of the best predicates according to their truth value. GA is executed for each class in the dataset (setosa, virginica, and versicolor) and, for each symbolic predicate (implication and equivalence operators and deductive structure modus ponens).
4 Sub-predicates are extracted from generated fuzzy predicates to evaluate each instance in the dataset.
5 For each instance, the truth value of the sub-predicates corresponds to the implication, equivalence, and deductive structure modus ponens. Through the highest, the class for the instance is selected.
6 The accuracy is calculated according to equation (3) and compared with the other proposals to get the highest.

## 7 Genetic Algorithm

The proposed genetic algorithm (GA) includes techniques to maximize the truth-value of results, such as:

1 Multiple generators of sub-predicates with independent search configuration in the creation of chromosomes.
2 Linguistic variables are defined by the generalized membership function.
3 Initial population based on statistical data.
4 Parameters optimization through an intern genetic algorithm.
5 Taboo roulette to avoid inconsistent predicates.
6 Weighted roulette to control predicate length.
7 Calculation of universal and existential quantifiers in fuzzy predicates.
8 Use of correct deductive structures as an objective function.

The genetic algorithm is given for the next steps (Algorithm 1):

The initial population is generated based on statistical data in the dataset, and then according to the total generations set in the configuration, the genetic operators: selection, crossover, mutation, optimization, and reduction of the population of chromosomes are performed.

In the generation of initial population and selection, the taboo and probabilistic roulettes are used.

The optimization of chromosomes is the generation of the best parameters of GMF of linguistic variables and it is performed through an internal GA.

**Algorithm 1. Genetic Algorithm**

```
Input:
        Search Algorithm configurations
        D: Dataset
Output:
        R: Set of best fuzzy predicates based on
truth-value
```

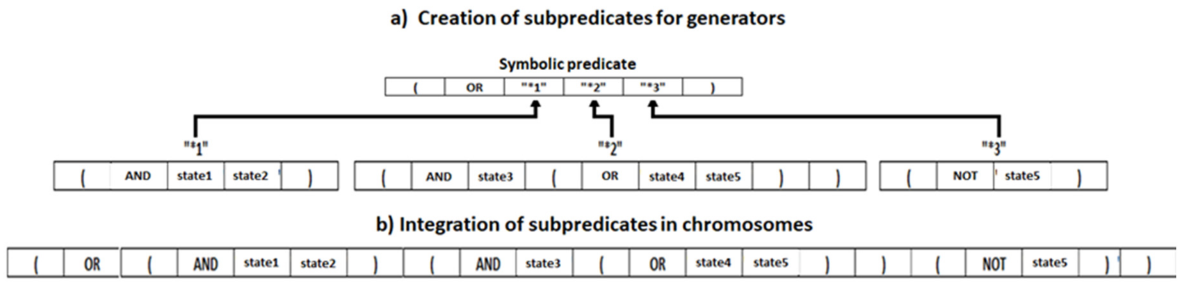
a) Creation of subpredicates for generators
Symbolic predicate
( OR "*1" "*2" "*3" )
"*1"
"*2"
"*3"
( AND state1 state2 )
( AND state3 ( OR state4 state5 ) )
( NOT state5 )
b) Integration of subpredicates in chromosomes
( OR ( AND state1 state2 ) ( AND state3 ( OR state4 state5 ) ) ( NOT state5 ) )

**Fig. 3.** Integration of sub-predicates in chromosome

```
R=Generate Initial Population
For i=1 to total Generations
        parents=selection(R)
        R=crossover(R,parents)
        R=mutation(R)
        R=optimize(R)
        R=reduce(R)
R=sort(R)
return R
```

The generation of initial population (step 1) is done through the creation of chromosomes and the initialization of the parameters of the MF defining the linguistic variables that are included (sections 6.1 - 6.3). This operation also uses the taboo roulette to avoid the generation of inconsistent sub-predicates (Section 6.8).

The selection operator (step 3) generates the parents matrix, which contains $n$ rows and 2 columns to form pairs of chromosomes that are chosen to interchange genes. Selection operator works as follows:

1 The first parent is the best chromosome available according to its truth value.
2 The second parent is chosen through the weighted roulette, which provides a greater probability of being selected to the chromosomes with the best truth value.
3 This operations are repeated to complete the required $n$ pairs of chromosomes.
4 The crossover, mutation and optimization operators are explained in sections 6.4 - 6.6 respectively.
5 The population of chromosomes is reduced by a percentage according to the GA configuration.
6 After calculating the total number of generations, the population is ordered according to the truth value of the chromosomes.

## 7.1 Chromosomes

In the GA, which is a search algorithm, chromosomes are a representation of fuzzy predicates, which are evaluated over the dataset to calculate the truth value.

The GA is configured through a list of parameters to generate fuzzy predicates that include (figure 3):

1 A symbolic predicate that defines the structure of the searched predicates, i. e.:

$$(OR“*1”“*2”“*3”).$$

2 Depth defines how many levels maximum is in the predicate.
3 Logical operators included.
4 Linguistic variables, among others.

Symbolic predicates may include generators of sub-predicates, which are represented by stars and a consecutive number. These generators have an independent configuration with their parameters for the search algorithm (table 2).

According to their configuration, generators create sub-predicates which include a set of parameters (Figure 3). 1=Generator, 2=Linguistic variables, 3=depth, 4=Logical operators.

Algorithm 2 shows how are created the subchromosomes, a fragment of a chromosome corresponding to a generator:

**Algorithm 2. Create subChromosome**

```
Input:
        $v$: List of linguistic variables
        $o$: List of logical operators
        $c$: Search configuration
Output:
        $s = \{\emptyset\}$: subchromosome
$n = roulette.getValue$(depth)  //  return the amount of levels for the subredicate in [0, depth]
$if\ n = 0$
        $index = tabooRoulette(v)$
        $s = s \cup v_{index}$
        $return\ s$
$index = roulette.getValue(\text{o})$
$operator = \text{o}_{\text{index}}$
$s = s \cup \{(", \text{operator}. ")\}$
$for\ level = 1\ to\ total\ n$
        $for(index = 1\ to\ size\ of\ s$
                currentLevel = $searchInCurrentLevel(index, level, s)$
                $If\ currentLevel = true$
                        $s_{index} = insertOperand(s, index, v, tabooRoulette)$
$return\ s$
```

The method searchInCurrentLevel verifies that element i corresponds to the current depth level within the subpredicate and returns true if so and false otherwise.

The insertOperando method looks for the index element in s and inserts an operator or a linguistic variable, which are selected randomly.

If an operator is selected, then:

$$s = \{s_1, s_{index}\} \cup \{(\text{",operator, ")"}\} \cup \{s_{index+1}, s_n\},$$

where operator is selected randomly.

If a linguistic variable is selected, then:

$$s = \{s\_1, s\_index\ \} \cup v\_i \cup \{s\_(index + 1), s\_n\},$$

where

$$i = tabooRoulette(v).$$

## 7.2 Linguistic Variables

The linguistic states are presented as linguistic variables initially, in which the parameters of the membership functions (MF) are going to be optimized to maximize the truth value of the predicate. The MF used to define the linguistic variables is the generalized membership function GMF [10]:

**Table 2.** Configuration of the search algorithm

| 1 | 2 | 3 | 4 |
|---|---|---|---|
| *1 | state1, state2, state3, state 5 | 1 | AND |
| *2 | state3, state4, state5 | 2 | AND, OR |
| *3 | state1, state3, state5 | 1 | NOT |

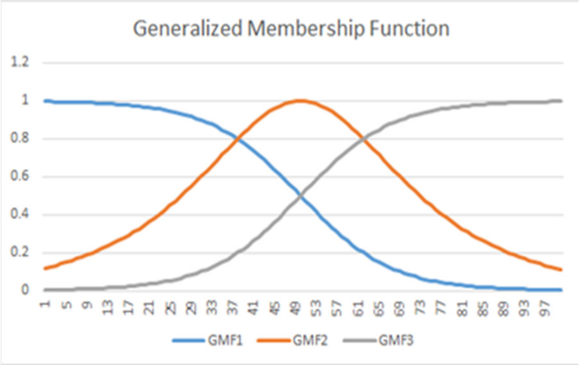

**Fig. 4**. GMF with parameters m=0 (GMF1), m=0.5 (GMF2) and, m=1 (GMF3)

$$GMF = \frac{sigm(x,\gamma,\alpha)^m[1-sigm(x,\gamma,\alpha)]^{1-m}}{M}. \quad (4)$$

With $sigm(x, \gamma, \alpha) = \frac{1}{1+e^{-\alpha(x-\gamma)}}$.

Here $M = m^m(1-m)^{1-m}$ and

1 $\gamma$ defines the center of the graphic of the function,

2 $\alpha$ defines the breadth of the function,

3 $m$ modifies the form of the function.

The advantage of the GMF over other membership functions is that it has the flexibility to change the shape of the function and can be used to model different linguistic variables, unlike other functions such as triangular or trapezoidal that require other types of functions to model different linguistic states (figure 4).

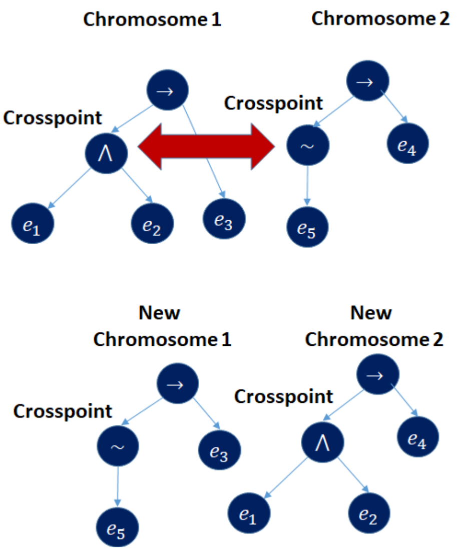

**Fig. 5.** Crossover operation

### 7.3 Membership Function's Parameters Initialization

In the discovery of fuzzy predicates, the first data is separated into two parts by attributes. The first part includes the attributes of the condition, meaning all of them will be used to create sub-predicates that correspond to the attribute of decision, which is the second part of attributes in the dataset.

For each attribute in the dataset, it is calculated the values of the parameters through percentiles and according to the number of linguistic variables associated with each attribute.

For each variable, the gamma parameter is calculated according to the percentile that corresponds to it, for example, if there are three linguistic variables for an attribute, the corresponding percentiles are 25, 50 and 75 and the gamma values are those that correspond to these percentiles.

The dataset is normalized, and all attributes of the condition and all instances take values in the range [0, 1]. This normalization allows the parameters of the GMF $\gamma$ and $m$ to take values also in [0, 1] and $\alpha$ is standardized in [10, 50].

According to the linguistic variable $i$ defined for the attribute of condition $k$, algorithm 3 shows the definition of each linguistic variable *s*:

**Algorithm 3. Definition of GMF**

```
Input
    $D$: Dataset normalized
Output
    $E$: Set of linguistic variables
    Let $A$ be the set of attributes in $D$
    For each $a \in A$
        Let $n_k$ the number of linguistic variables for $a$
        For $i = 1 \ to \ n_k$
            $e_{label} = a_{name} + i$
            $e_{attribute} = a_{name}$
            $e_{MF} = "GMF"$
            $e_{\gamma} = noise\left(calcPercentile\left(a, \frac{100i}{n_k+1}\right)\right)$
            $e_{\alpha} = noise(35\ rand(0,1) + 15)$
            $e_m = noise\left(\frac{m_i - m_1}{m_{n_k} - m_1}\right)$
            $E = E \cup e$
        return $E$
```

The method noise injects random variations in range [-0.05,0.05] to the computed parameters.

### 7.4 Crossover Operator

In the crossover operator, the chromosomes selected as parents generate copies of themselves and then the created chromosomes exchange information. On each chromosome created, a crossover point is selected, which can point to a linguistic variable or a complete sub-predicate (Figure 5).

The linguistic variable or the corresponding sub-predicate is taken from each chromosome and is integrated into the other at the crossing point that corresponds to each one.

It is validated that the new chromosomes comply with the restrictions of the GA search algorithm configuration; if not, the operation is repeated.

If they comply with the configuration, the truth value of the altered chromosomes is calculated and they are integrated into the population.

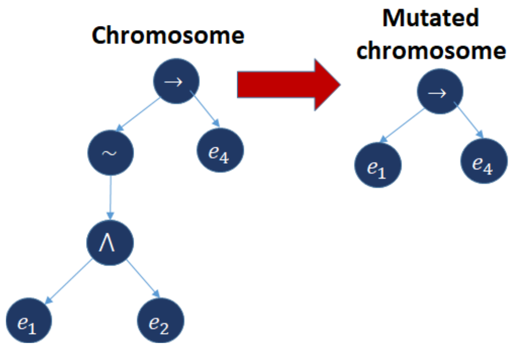

**Fig. 6.** Mutation operation

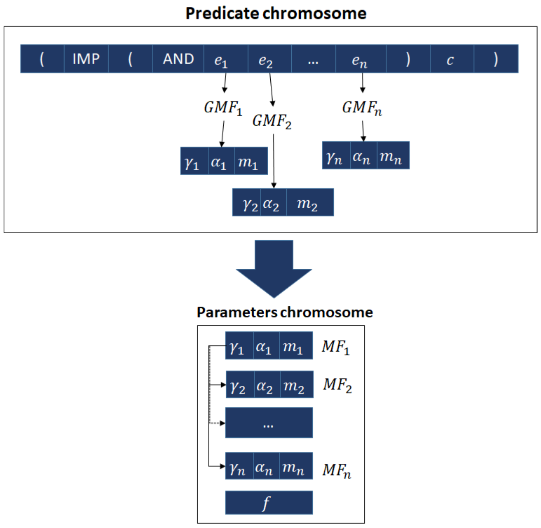
Predicate chromosome
(
IMP
(
AND
$e_1$
$e_2$
...
$e_n$
)
$c$
)
$GMF_1$
$GMF_2$
$GMF_n$
$\gamma_1$ $\alpha_1$ $m_1$
$\gamma_2$ $\alpha_2$ $m_2$
$\gamma_n$ $\alpha_n$ $m_n$
Parameters chromosome
$\gamma_1$ $\alpha_1$ $m_1$ $MF_1$
$\gamma_2$ $\alpha_2$ $m_2$ $MF_2$
...
$\gamma_n$ $\alpha_n$ $m_n$ $MF_n$
$f$

**Fig. 7.** Extraction of the parameters of the predicate to create the parameter's chromosome

### 7.5 Mutation Operator

In the mutation operator, a chromosome is selected to be altered. On that chromosome, a point is randomly selected at which a linguistic variable or a sub-predicate can be found (figure 6).

This variable or sub-predicate is extracted from the chromosome and is replaced by an optimized sub-predicate or variable.

It is validated that the new chromosome meets the restrictions of the search algorithm configuration and if it does not, this operation is repeated.

If the new chromosome meets these constraints, then its truth value is calculated and it is returned to the population.

### 7.6 Optimization of Parameters in Membership Functions

The optimization of parameters for GMF is done through an internal GA that first extracts the parameters to build a new population of chromosomes that work with $\alpha, \gamma,$ and $m$ related to the linguistic variables in the fuzzy predicates (Figure 7, steps 1 and 2 of algorithm 4). In the generation of the initial population, the parameters of the MF are injected with noise.

The algorithm 4 shows how genetic operators such as selection, crossover, and mutation are applied to the population of parameters' chromosomes for the number of generations set in the GA configuration.

In this algorithm, whenever a chromosome is altered, the parameters in it are substituted into the predicate to obtain the truth value of the chromosome.

The selection operator works in the same way as in algorithm 3.

In the crossover operator, the parent chromosomes create copies of themselves and in them, they go through parameter by parameter and exchange the values randomly. If exchanges could not be carried out, the operation is repeated.

In the mutation operator, the selected chromosome is traversed parameter by parameter to alter some of them by selecting them randomly.

Finally, the fuzzy predicate is returned with the set of parameters that maximize the truth value in it.

**Algorithm 4. Parameter Optimizer**

```
   Input
      $D$: Dataset normalized
      $p$: Fuzzy predicate
   Output
      $q$: Predicate with best set of parameters
 of MF according to their truth value
      1. $r = extractParameters(p)$
      2. $S = generateInitialPopulation(r)$
      3. For $i = 1$ $to$ $totalGenerations$
      4.     $parents = selection()$
      5.     $S = crossover(S, parents)$
      6.     $S = mutation(S)$
      7. $return\ q = \max_{truthvalue} S$
```

### 7.7 Weighted Roulette

Logical operators have different arity (Table 3). In the case of conjunctions and disjunctions, to avoid predicates with a high number of linguistic states that increase the complexity of their reading, a weighted roulette is proposed that assigns a greater probability of choosing a smaller number of linguistic states in the operation.

Given the following: If $r_i = q^{i-2}$ where $q = 0.5 < i < 1$ and $i = 2\ to\ n$ is the number of operands, then the cumulative probability that the selected operator has $i$ linguistic states as operands is given by:

$$p_i = \frac{r_i}{\sum_{i=2}^{n} r_i} + \begin{cases} 0 & if\ i = 2, \\ p_{i-1} & if\ i > 2, \end{cases} \tag{5}$$

with $\mathrm{i} = 2$ to n.

### 7.8 Taboo Roulette

Taboo roulette is implemented to avoid inconsistent predicates, which are those that in the logical operations contain duplicate linguistic variables in the case of implication, equivalence, or disjunction operators, for example:

(*IMP* **high chlorideshigh chlorides**)

(*OR*"***high alcohol***" "***high alcohol***"

"*low pH*")

In the case of conjunctions, inconsistent predicates, besides incorporating duplicate linguistic states, also can include contradictory linguistic states, that is, they correspond to a fuzzy partition and generate contrary meanings that result in low truth values. An example is the following:

("*AND*" "*low chlorides*" "*high chlorides*").

In disjunctions, to avoid the generation of contradictory predicates in the initial population, a taboo roulette is used, in which in the list of participating slots, once a linguistic state is selected, it is eliminated from the roulette and added to a taboo list. This will be done for each logical operation in the predicate.

**Table 3.** Operators' arity

| Operator | Arity |
|---|---|
| Negation (NOT) | 1 |
| Implication (IMP) | 2 |
| Equivalence (EQV) | 2 |
| Conjunction (AND) | 2 to n |
| Disjunction (OR) | 2 to n |

In conjunctions, when a linguistic state is selected, all states with the corresponding attribute are taken to the taboo list.

Once the selection of linguistic states in a sub-predicate is completed, the roulette wheel restarts, returning all states in the taboo list to the list of participating slots.

Each slot includes:

1. Name of the linguistic state
2. Attribute to which the linguistic state belongs in the data set.

### 7.9 Calculation of Universal and Existential Quantifiers in Fuzzy Predicates

In this genetic algorithm it was implemented the evaluation of quantifiers included in the fuzzy predicate, to calculate the truth value of the quantifiers, first, they are evaluated and the result obtained replaces the partial operation in the predicate, for example:

(*AND (EXIST "sepal-length")*
*(FORALL "sepal-width") petal-length*).

First, evaluate (*EXIST*"*sepal-length*"), then evaluates (*FORALL*"*sepal-width*") and the resulting predicate remains as follows:

(*AND* 0.5789 0.3724 "*petal-length").*

The resulting predicate can now be evaluated for the instances in the dataset.

### 7.10 Use of Correct Deductive Structures as the Objective Function

The implication operator has a combination that results in ineffective like the following:

$$False \to True \Longrightarrow True.$$

In CFL, the combination of:

$$low\ truth\ value \longrightarrow high\ truth\ value \\ \Longrightarrow\ high\ truth\ value.$$

Which means that a premise with a low truth value implies a conclusion with a high truth value, which is not correct, that is, you can have a predicate that evaluated can indicate that an instance corresponds to a result or a class to which is not related.

For this reason, it was implemented a correct deductive structure as an objective function such as modus ponens, which is combined with the existential quantifier. In this way, the objective function is defined like this:

$$\exists P \wedge (P \to Q).$$

The predicate notation in the algorithm is as follows:

$$\big(AND\,(EXIST\ {*1}")\ (IMP\ {*1}"\ "class")\big).$$

## 8 Experimentation

As a case study, the Iris set was used for experimentation. The data set contains 50 samples of each of the three Iris species (setosa, virginica, and versicolor). Four traits were measured for each sample: the length and width of the sepal and petal, in centimeters.

The data set with 150 records was divided into two parts, one for discovery and the other for inference, each part with 120 and 30 respectively.

The data set was used for the classification task, where a class was assigned for each type of Iris species and, this attribute is defined as the decision variable within the classification.

On the other hand, the length and width attributes of the sepal and petal are defined as the condition variables, that is, those by which it is possible to find which type of Iris each instance in the data set corresponds to.

The following configuration of the search algorithm was used for the discovery of fuzzy predicates (Table 4):

**Table 4.** Configuration of GA

| Parameter | Value |
|---|---|
| Máximum Results | 10 |
| Total generations | 10 |
| Minimum truth-value | 0.8 |
| Mutation percentage | 0.05 |
| Initial Population | 50 |
| Depth | 1 |
| Máximum Results | 10 |

The following are symbolic predicates:

1 The implication

$$(IMP\ "{*1}"\ \ "\mathrm{setosa}"),$$

2 The equivalence

$$(EQV\ "{*1}"\ \ "\mathrm{setosa}"),$$

3 The modus ponens deductive structure

$$(AND\ (EXIST\ "{*1}")(IMP\ "{*1}"\ \ "\mathrm{setosa}")).$$

In the same way, the symbolic predicates for the classes *virginica* and *versicolor*.

To calculate the accuracy of this work it was used k-cross-validation and to get the number of folds it was calculated by:

$$K = \ln(total\ instances) = \ln(150) = 5.01 = 5.$$

## 9 Results

Table 5 shows the best predicates obtained for each class using the implication operator, in the same way, tables 6 and 7 show the best predicates using equivalence and correct deductive structure, respectively.

Sub-predicates that act as a premise are extracted from each predicate and evaluated to get its truth value for each instance.

For equivalence, the sub-predicates extracted are:

1 $(AND\ \mathrm{petal\_widthpetal\_length}),$

2 $(AND\ \mathrm{sepal\_lengthpetal\_width}$
$"petal\_length"),$

3 $(AND\ \mathrm{sepal\_lengthpetal\_length}).$

**Table 5.** Best fuzzy predicates obtained for each class for implication operator

| Class | Predicate |
|---|---|
| Setosa | *(EQV (AND "petal_width" "petal_length") “setosa”)* |
| Virginica | *(EQV (AND "sepal_length" "petal_width" "petal_length") “virginica”)* |
| Versicolor | *(EQV (AND "sepal_length" "petal_length") “versicolor”)* |

**Table 6.** Best fuzzy predicates obtained for each class for implication operator

| Class | Predicate |
|---|---|
| Setosa | *(IMP (AND "petal_width" "petal_length") “setosa”)* |
| Virginica | *(IMP (AND "sepal_length" "petal_width" "petal_length") “virginica”)* |
| Versicolor | *(IMP (AND "sepal_length" "petal_length") “versicolor”)* |

**Table 7.** Best fuzzy predicates obtained for each class for correct deductive structure

| Class | Predicate |
|---|---|
| Setosa | *(AND (EXIST (AND "petal_length" "sepal_width") (IMP (AND "petal_length" "sepal_width") “setosa”))* |
| Virginica | *(AND (EXIST (AND "sepal_length" "sepal_width" "petal_length" "petal_length") (IMP (AND "sepal_length" "sepal_width" "petal_length" "petal_length") “virginica”))* |
| Versicolor | *(AND (EXIST (AND "sepal_length" "petal_width") (IMP (AND (EXIST (AND "sepal_length" "petal_width") “versicolor”))* |

**Table 8.** Fragment of results of classification task for equivalence operator

| RC | Se | Vi | Ve | OC | Cl |
|---|---|---|---|---|---|
| 1 | **0.998** | 0.086 | 0.000 | 1 | Hit |
| 3 | 0.000 | 0.287 | **0.622** | 3 | Hit |
| 3 | 0.000 | 0.142 | **0.912** | 3 | Hit |
| 2 | 0.000 | **0.635** | 0.079 | 2 | Hit |
| 2 | 0.000 | **0.603** | 0.202 | 2 | Hit |
| 2 | 0.000 | **0.635** | 0.125 | 2 | Hit |
| 1 | **0.991** | 0.050 | 0.000 | 1 | Hit |
| 1 | **0.994** | 0.091 | 0.000 | 1 | Hit |
| 1 | **0.987** | 0.070 | 0.000 | 1 | Hit |
| 3 | 0.000 | 0.342 | **0.601** | 3 | Hit |

For implication:

1. $(AND\ "petal\_width"\ "petal\_length")$,
2. $(AND\ sepal\_length\ "petal\_width"\ "petal\_length")$,
3. $(AND\ "sepal\_length"\ "petal\_length")$.

For the correct deductive structure:

1. $(AND\ "petal\_length"\ "sepal\_width")$,
2. $(AND$ sepal_lengthsepal_width $"petal\_length"\ "petal\_length")$

3 $(AND\ "sepal\_length"\ "petal\_width")$

**Table 9.** Classification results for an equivalence operator

| Fold | Setosa | | | | | | Virginica | | | | | | Versicolor | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | P | N | TP | TN | FP | FN | P | N | TP | TN | FP | FN | P | N | TP | TN | FP | FN |
| 1 | 11 | 19 | 11 | 19 | 0 | 0 | 11 | 19 | 11 | 18 | 1 | 0 | 8 | 22 | 7 | 22 | 0 | 1 |
| 2 | 8 | 22 | 8 | 22 | 0 | 0 | 12 | 18 | 12 | 16 | 2 | 0 | 10 | 20 | 8 | 20 | 0 | 2 |
| 3 | 10 | 20 | 10 | 20 | 0 | 0 | 7 | 23 | 7 | 18 | 5 | 0 | 13 | 17 | 8 | 17 | 0 | 5 |
| 4 | 9 | 21 | 9 | 21 | 0 | 0 | 11 | 19 | 11 | 15 | 4 | 0 | 10 | 20 | 6 | 20 | 0 | 4 |
| 5 | 12 | 18 | 12 | 18 | 0 | 0 | 9 | 21 | 8 | 20 | 1 | 1 | 9 | 21 | 8 | 20 | 1 | 1 |

**Table 10.** Classification results for implication operator

| Fold | Setosa | | | | | | Virginica | | | | | | Versicolor | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | P | N | TP | TN | FP | FN | P | N | TP | TN | FP | FN | P | N | TP | TN | FP | FN |
| 1 | 11 | 19 | 11 | 19 | 0 | 0 | 11 | 19 | 11 | 18 | 1 | 0 | 8 | 22 | 7 | 22 | 0 | 1 |
| 2 | 8 | 22 | 8 | 22 | 0 | 0 | 12 | 18 | 12 | 16 | 2 | 0 | 10 | 20 | 8 | 20 | 0 | 2 |
| 3 | 10 | 20 | 10 | 20 | 0 | 0 | 7 | 23 | 7 | 18 | 5 | 0 | 13 | 17 | 8 | 17 | 0 | 5 |
| 4 | 9 | 21 | 9 | 21 | 0 | 0 | 11 | 19 | 11 | 15 | 4 | 0 | 10 | 20 | 6 | 20 | 0 | 4 |
| 5 | 12 | 18 | 12 | 18 | 0 | 0 | 9 | 21 | 8 | 20 | 1 | 1 | 9 | 21 | 8 | 20 | 1 | 1 |

**Table 11.** Classification results for the correct deductive structure

| Fold | Setosa | | | | | | Virginica | | | | | | Versicolor | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | P | N | TP | TN | FP | FN | P | N | TP | TN | FP | FN | P | N | TP | TN | FP | FN |
| 1 | 12 | 18 | 12 | 18 | 0 | 0 | 7 | 23 | 6 | 18 | 5 | 1 | 11 | 19 | 6 | 18 | 1 | 5 |
| 2 | 10 | 20 | 10 | 20 | 0 | 0 | 9 | 21 | 9 | 16 | 5 | 0 | 11 | 19 | 6 | 19 | 0 | 5 |
| 3 | 7 | 23 | 7 | 23 | 0 | 0 | 12 | 18 | 11 | 13 | 5 | 1 | 11 | 19 | 6 | 18 | 1 | 5 |
| 4 | 7 | 23 | 7 | 23 | 0 | 0 | 16 | 14 | 15 | 12 | 2 | 1 | 7 | 23 | 5 | 22 | 1 | 2 |
| 5 | 14 | 16 | 14 | 16 | 0 | 0 | 6 | 24 | 5 | 17 | 7 | 1 | 10 | 20 | 3 | 19 | 1 | 7 |

**Table 12.** Accuracy for equivalence operator

| F | Accuracy | | | |
|---|---|---|---|---|
| | Setosa | Virginica | Versicolor | Av |
| **1** | 100% | 97% | 97% | 98% |
| **2** | 100% | 93% | 93% | 96% |
| **3** | 100% | 83% | 83% | 89% |
| **4** | 100% | 87% | 87% | 91% |
| **5** | 100% | 93% | 93% | 96% |
| | | | **Total** | **94%** |

**Table 13.** Accuracy of classification for implication operator

| F | Accuracy | | | |
|---|---|---|---|---|
| | Setosa | Virginica | Versicolor | Av |
| 1 | 97% | 50% | 53% | 67% |
| 2 | 100% | 80% | 80% | 87% |
| 3 | 100% | 67% | 67% | 78% |
| 4 | 100% | 63% | 63% | 76% |
| 5 | 100% | 63% | 63% | 76% |
| | | | **overall accuracy** | **76%** |

**Table 14.** Accuracy of classification for deductive structure modus ponens

| F | Accuracy | | | |
|---|---|---|---|---|
| | Setosa | Virginica | Versicolor | Av |
| 1 | 100% | 80% | 80% | 87% |
| 2 | 100% | 83% | 83% | 89% |
| 3 | 100% | 80% | 80% | 87% |
| 4 | 100% | 90% | 90% | 93% |
| 5 | 100% | 63% | 63% | 76% |
| | | | **overall accuracy** | **76%** |

**Table 15.** Comparison of accuracy of operators and CDS Modus Ponens

| Proposal | Accuracy |
|---|---|
| Equivalence | **94 %** |
| Implication | 76 % |
| Modus Ponens | 88 % |

Respective truth values are compared to get the highest, which indicates the selected class for each instance (table 8).

RC=Real class, $Se, Vi, Ve$ = truth-value for setosa, virginica, and versicolor, in bold the highest truth-value, OC=Obtained class, CI=Hit or error in classification. For each class $p, n, tp, tn, fp,$ and $fn$ are obtained to get their accuracy, given in tables 9, 10, and 11 for equivalence and implication, operators and the deductive structure modus ponens.

Based on the results of classification with proposed symbolic predicates their accuracy was calculated (Tables 12, 13, and 14):

Comparing the accuracy of operators and deductive structure modus ponens proposed the equivalence obtained the highest (table 15):

For the experiment, 30 runs were executed and Table 16 shows a fragment of the comparisons of these runs between the operators of implication, equivalence and the deductive structure Modus Ponens. To compare the proposals and determine if one is significantly better than the others, the Wilcoxon test was used. For each pair of proposals, the following were compared:

1. the logical operators of implication and equivalence,

**Table 16.** Fragment of the Comparison of accuracy of operators and CDS

| Run | Imp | Eqv | MP |
|---|---|---|---|
| 1 | 76.5 | 93.6 | 87.9 |
| 2 | 77.3 | 90.7 | 83.6 |
| 3 | 78.7 | 90.7 | 87.6 |
| 4 | 84.9 | 90.8 | 83.1 |
| 5 | 77.8 | 91.4 | 85.1 |
| 6 | 77.3 | 94.3 | 90.4 |
| 7 | 75.6 | 95.6 | 81.8 |
| 8 | 76.4 | 87.8 | 82.1 |
| 9 | 76 | 97.8 | 83.1 |
| 10 | 77.8 | 94.8 | 78.7 |

**Table 17.** Bonferroni's test results

| G1 | G2 | stat | PV-Cor | Rej |
|---|---|---|---|---|
| EQV | IMP | 16.01 | 0 | True |
| EQV | MP | 10.21 | 0 | True |
| IMP | MP | -6.85 | 0 | True |

G1=Group1, G2=Group2,PV-Corr=pvalue-Corrected, Rej=rejected

**Table 18.** p-values

| | IMP | EQV |
|---|---|---|
| IMP | - | 1.86E-9 |
| EQV | 1.86E-9 | - |
| MP | 1.41E-6 | 1.86E-9 |

2. the equivalence operator with the modus ponens deductive structure and,
3. the equivalence operator with the deductive structure modus ponens.

The p-value is compared to 0.05, if it is lower, then there is a significant difference between the proposals and otherwise, there is no significant difference. Table 17 shows the p-values between the proposals.

All comparisons of the proposals obtained a p-value less than 0.05, so it is concluded that there is a significant difference between them.

Since there are significant differences between the proposals, the post hoc Bonferroni test is executed, obtaining the results shown in table 18: According to the results of the Bonferroni test, it is concluded that the significant difference is in favor of using the equivalence operator over the implication operator and the modus ponens deductive structure.

## 10 Conclusions

This work proved that a genetic algorithm that uses CFL provides a solid methodology that allows for solving classification problems through fuzzy predicates.

Moreover, the use of the equivalence (EQV) operator and the correct deductive structure modus ponens (MP) as a model compared with the implication operator contributes to a new level of classification.

The combination of a genetic algorithm that uses EQV or MP with classification models based on compensatory fuzzy logic generates predicates that tend to show better accuracy in the classification task.

Results of this comparison indicate that there is a tendency to obtain better accuracy using the equivalence operator with 94% over the implication with 76% and 88% of the predicate using the correct deductive structure.

Results are proof that this model allows to solve classification problems and these results serve in decision-making processes, data analysis, and data diagnosis, among others.

As future work, more experimentation remains to be done using other classification datasets as well as other methods of classification and the implementation of inference tasks.

## Acknowledgments

The authors want to thank Laboratorio Nacional de Tecnologías de Información and the support of (a) Cátedras CONAHCYT Program Number 3058, (b) the TecNM, (c) the support granted through the Scholarship for Postgraduate Studies with CVU 886000.

## References

1. **Cisneros, L., Rivera, G., Florencia, R., Sánchez-Solís, J. P. (2023).** Fuzzy optimisation for business analytics: A bibliometric analysis. Journal of Intelligent & Fuzzy Systems, Vol. 44, No. 2, pp. 2615–2630. DOI: 10.3233/JIFS-221573.
2. **Xexéo, G., Braga, A. (2008).** A tool for fuzzy reasoning and querying. Handbook of Research on Fuzzy Information Processing in Databases IGI Global. pp. 381–406. DOI: 10.4018/978-1-59904-853-6.ch015.
3. **Pérez-Pueyo, R. (2005).** Procesado y optimización de espectros Raman mediante técnicas de lógica difusa: aplicación a la identificación de materiales pictóricos. Universitat Politècnica de Catalunya. DOI: 10. 5821/dissertation-2117-94198.
4. **Padrón-Tristán, J. F., Cruz-Reyes, L., Espín-Andrade, R. A., Llorente-Peralta, C. E. (2021).** A brief review of performance and interpretability in fuzzy inference systems. New Perspectives on Enterprise Decision-Making Applying Artificial Intelligence Techniques, pp. 237–266. DOI: 10.1007/978-3-030-71115-3_11.
5. **Zadeh, L. A. (1965)**. Information and control. Fuzzy sets, Vol. 8, No. 3, pp. 338–353. DOI: 10.1016/S0019-9958(65)90241-X.
6. **Serov, V. V., Sokolov, I. V., Budnik, A. A. (2019)**. Applied calculus of fuzzy predicates for the formalization of knowledge. In IOP Conference Series: Materials Science and Engineering, Vol. 537, No. 4, p. 042043. DOI: 10.1088/1757-899X/537/4/042043.
7. **Espin-Andrade, R. A., Gonzalez, E., Pedrycz, W., Fernandez, E. (2016)**. An interpretable logical theory: the case of compensatory fuzzy logic. International Journal of Computational Intelligence Systems, Vol. 9, No. 4, pp. 612–626. DOI: 10.1080/18756891.2016.1204111.
8. **Espín-Andrade, R. A., Bataller, A. C., Marx-Gomez, J., Racet, A. (2011).** Fuzzy semantic transdisciplinary knowledge discovery approach for business intelligence. Towards a Transdisciplinarian Technology for Business Intelligence: Gathering Knowledge Discovery, Knowledge Management and Decision Making, pp. 13–34.
9. **Galende, M., Sainz, G. I., Fuente, M. J. (2011).** Accuracy-interpretability trade-off for precise fuzzy modeling using simple indices. Application to Industrial plants. IFAC Proceedings, Vol. 44, No. 1, pp. 12656–12661. DOI: 10.3182/20110828-6-IT-1002.02005.
10. **Llorente-Peralta, C. E., Cruz-Reyes, L., Espín-Andrade, R. A. (2021).** Knowledge discovery using an evolutionary algorithm and compensatory fuzzy logic. Studies in Computational Intelligence, pp. 363-383. DOI: 10.1007/978-3-030-68776-2_21.

11. **González-Ramírez, C. M. (2017).** Aproximación al concepto de inferencia desde dos modelos de comprensión: modelo estratégico y modelo de construcción e integración. Literatura y lingüística, No. 35, pp. 295-312. DOI: 10.4067/S0716-5811201700 0100295.
12. **Bunge, M. (2000).** La investigación científica: su estrategia y su filosofía. Siglo XXI.
13. **Mamdani, E. H., Assilian, S. (1975).** An experiment in linguistic synthesis with a fuzzy logic controller. International Journal of Man-Machine Studies, Vol. 7, No. 1, pp. 1–13. DOI: 10.1016/S0020-7373(75)80002-2.
14. **Takagi, T., Sugeno, M. (1985).** Fuzzy identification of systems and its applications to modeling and control. IEEE transactions on systems, man, and cybernetics, No. 1, pp. 116–132. DOI: 10.1109/TSMC.1985.6313399.
15. **Galdi, P., Tagliaferri, R. (2018).** Data mining: accuracy and error measures for classification and prediction. Encyclopedia of bioinformatics and computational biology, Vol. 1, pp. 431–436. DOI: 10.1016/B978-0-12-809633-8-20474-3.
16. **Fernández-Cervantes, V., García, A., Ramos, M. A., Méndez, A. (2015).** Facial geometry identification through fuzzy patterns with RGBD sensor. Computación y Sistemas, Vol. 19, No. 3, pp. 529–546.
17. **Silva, D., Manzo-Martínez, A., Gaxiola, F., Gonzalez-Gurrola, L., Ramírez-Alonso, G. (2022).** Analysis of CNN architectures for human action recognition in video. Computación y Sistemas, Vol. 26, No. 2, pp. 623–641. DOI: 10.13053/cys-26-2-4245.
18. **Pach, F. P., Gyenesei, A., Abonyi, J. (2008).** Compact fuzzy association rule-based classifier. Expert Systems with Applications, Vol. 34, No. 4, pp. 2406–2416. DOI: 10.1016/ j.eswa.2007.04.005.
19. **Binu, R., Isaac, P. (2021).** Multi-attribute neutrosophic decision-making in dosimetric assessment of radiotherapy imaging techniques. In: Smarandache, F., Abdel-Basset, M. (eds) Neutrosophic Operational Research. Springer, Cham. DOI: 10.1007/978-3-030-57197-9_28.
20. **Zadeh, L. A. (1975).** The concept of a linguistic variable and its application to approximate reasoning—I. Information sciences, Vol. 8, No. 3, pp. 199–249. DOI: 10.1016/0020-0255(75) 90036-5.
21. **Gullo, F. (2015).** From patterns in data to knowledge discovery: What data mining can do. Physics Procedia, Vol. 62, pp. 18–22. DOI: 10.3109/0142159X.2014.1001729.
22. **Bigand, A., Colot, O. (2016).** Membership function construction for interval-valued fuzzy sets with application to Gaussian noise reduction. Fuzzy Sets and Systems, Vol. 286, pp. 66–85. DOI: 10.1016/j.fss.2015.07.001.
23. **Comas, D. S., Meschino, G. J., Nowé, A., Ballarin, V. L. (2017).** Discovering knowledge from data clustering using automatically-defined interval type-2 fuzzy predicates. Expert Systems with Applications, Vol. 68, pp. 136–150. DOI: 10.1016/j.eswa.2016.10.018.
24. **Zadeh, L. A. (1989).** Development of Probabilistic and Possebilistic Approaches to Approximate Reasoning and Its Applications. University of California, Computer Science Division, Berkeley. DOI: 10.21236/ADA 221945.

*Article received on 29/02/2024; accepted on 15/05/2024.*
**Corresponding author is Carlos Eric Llorente Peralta.*