Data mining software using fuzzy inference systems at the World Wide Web

R. C. F. Vidal, N. F. F. Ebecken & A. G. Evsukoff
Department Engineering, Coppe, Federal University of Rio de Janeiro, Brasil

Abstract

In this paper we present a software tool called Fuzzy_Query.Web (FQW), which is a web-based software interface that allows users to extract knowledge from a database using Classification Fuzzy Systems (CFS) and Database Manager System (DBMS). The FQW tool is used as an interpreter for several system structures that use CFS and various techniques of Artificial Intelligence (AI). Indexes for analysis of CFS performance are computed using the FQW tool and solutions for CFS in FQW tool are presented.
Keywords: data mining, database, data model, fuzzy_query.web (FQW), fuzzy query database, fuzzy classification system, fuzzy inference system.

1 Introduction

Microsoft owes a great part of its worldwide commercial success to the friendly user interfaces of its products, which allows easy interaction between the user and machine. The central idea in building systems that allow easy interaction between user and machine is to isolate the user from the internal processes needed to perform some computational task. This is achieved by the development of efficient software interfaces that encapsulate all the internal processes, improving the software usability and making it accessible to the ordinary user. In the present paper we take advantage of the concepts of interaction between computational systems developed in the past years to build a web-based user -friendly software that allows the user to easily access a database and quickly extract useful knowledge from it, which we call Fuzzy_Query.Web (FQW). We used Data Mining (DM) tools together with Artificial Intelligence techniques. This choice is motivated by the fact that the ideas of interaction between distinct computational systems are extensively used in Data Mining techniques. The FQW uses the concept of interaction in order to provide the users with the ability to extract knowledge from a database without any knowledge on data mining techniques and relational data bases (RDB), which makes FQW useable by non-specialists. The FQW uses Fuzzy Classification System (FCS), which is an Artificial Intelligence technique, based on inference fuzzy systems (IFS). Contributions to data mining have been given for many years through the techniques of Fuzzy Logic, Neural Networks, Specialist Systems and genetic algorithms that are part of a new paradigm known as intelligent systems. Those systems have the capability to adapt themselves to the nature of the problem. The novelty that FQW brings is that it can carry out the whole process of the Fuzzy Classification System (FCS), from fuzzification of values, generation of queries, until the decision. Fuzzy queries are generated automatically from the base of rules of the Fuzzy Inference System (FIS).

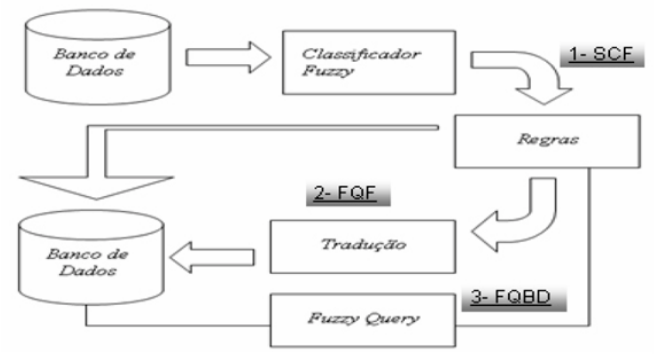


Figure 1: General idea of the process of FQW: flows 1-SCF – fuzzy classification system; 2-FQF – fuzzy query tools; 3-FQBD – fuzzy query the database.

A fuzzy query automatically generates rules on the queries of the Relational Data Base (RDB). Those queries are generated from the Fuzzy Inference System (FIS) and translated in order to be interpreted by the Relational Data Base (RDB), where it uses relational structure instead of a fuzzy structure. The calculations of the Vertical Indexes (VI) and Horizontal Indexes (HI) are given, as well as the process of simplification of queries generated and learned automatically by the fuzzy query. This method is described in [5] and implemented in the FQW tool. The Fuzzy Classification System (FCS) allows the generation of the process of the Fuzzy Classification System (FCS) through one or more databases, getting the base of rules automatically through either the inference fuzzy that gets the result through the classifier or fuzzy query generated by the Relational Data Base (RDB) and interacting with it. The proposed system aims at providing an intelligent system to the scientific community that improves the forms of interaction between users, fuzzy systems and relational databases. The Fuzzy_Query.Web software tool was built to perform the whole process of fuzzy classification system (FCS), such that specialized skills on fuzzy classification systems or artificial intelligence techniques are not required. The result of this project is a fuzzy system-based software in which the users are working at, possibly, distinct environments connected through the World Wide Web.

2 Fuzzy rule-based classifier

Consider the standard classification problem where input variables are presented as a p-dimensional vector x in the input variable domain $X_1 \times X_2 \times \ldots \times X_p = X^p$ and the output variables represented by the set of classes $\mathbf{C} = \{C_1, \ldots, C_m\}$. The classification problem is solved by assigning a class label $C_k \in \mathbf{C}$ to an observation $\mathbf{x}(t) \in X^p$, where t represents a record in the database. Fuzzy classifiers attempt to compute, for each class, a fuzzy membership value $\mu_{C_k}(\mathbf{x}(t))$ that corresponds to the degree of matching of the observation $\mathbf{x}(t)$ to the class C_k. This section describes the weighted fuzzy rule-based classifier approach, which is based on the fuzzy pattern matching approach. Under this approach, the output of the fuzzy classifiers is computed in two steps: For each input, compute partial outputs as the degree of matching of the observed input value to each class, compute the final output by the aggregation of all partial outputs. The following subsections describe the main steps of the fuzzy rule-based classifier approach.

2.1 Fuzzification

In a general application, the input variables may be numeric (discrete or continuous) or nominal. Fuzzy sets allow a unified representation for nominal and numeric variables as fuzzy sets. Fuzzification is thus an important issue in fuzzy query generation since it provides the numeric-to-linguistic interface that allows dealing with numeric values as linguistic terms. Generally, each input variable x_i can be described using ordered linguistic terms in a descriptor set $\mathbf{A}_i = \{A_{i1}, \ldots, A_{in_i}\}$. When the variable is nominal, the descriptor set is the set of possible values for the variable (or a combination of them). When the variable is numeric, the meaning of each term $A_{ij} \in \mathbf{A}_i$ is given by a fuzzy set defined on the variable domain. For a single variable input $x_i(t)$, the fuzzification vector $\mathbf{u}_i(t) = (u_{i1}(t), \ldots, u_{in_i}(t))$ is computed by the fuzzy sets in the fuzzy partition of the input variable domain as:

$$\mathbf{u}_i(t) = \left(\mu_{A_{i1}}(x_i(t)), \ldots, \mu_{A_{in_i}}(x_i(t))\right) \qquad (1)$$

An easy way to parameterize fuzzy sets is to use triangular membership functions that are completely determined by the centers of triangles, which may be considered as prototypes values for the corresponding fuzzy sets (Figure 2).

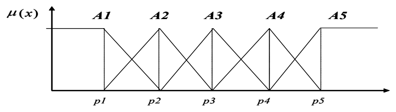


Figure 2: Example of fuzzy partition.

The fuzzification vector generalizes the information contained in the input variable and is computed in the same way if the variable is numeric or nominal. For nominal variables, there is no fuzziness and the fuzzification vector is a binary vector, indicating which nominal value is present on the record.

In a general-purpose fuzzy classifier, multidimensional fuzzification may also be considered and computed by fuzzy clustering methods. Nevertheless, multidimensional fuzzy sets often do not represent linguistic concepts and are not supported by some fuzzy query languages. Multidimensional fuzzy sets are thus out of the scope of this paper.

2.2 Fuzzy rules

A fuzzy rule relates input linguistic terms $A_{ij} \in \mathbf{A}_i$ to the classes $C_k \in \mathbf{C}$ in rules like:

$$if\ x_i(t)\ is\ A_{ij}\ then\ class\ is\ C_k\ with\ cf = \varphi^i_{jk} \tag{2}$$

where $\varphi^i_{jk} \in [0,1]$ is a confidence factor that represents the rule certainty.

The rule (2) describes a flexible constraint on the values of the variable x_i that can be related to the class (or concept) C_k . For instance, if the variable x_i represents the “salary”, then an example of the rule could be: “if the customer’s salary is high then the customer’s class is Gold”. The confidence factor represents how much of this relation is true, for instance 95%. The rule’s certainty factor may be considered as a relative quantifier [4]. The above example could be interpreted as “Most of the high salary customers are Gold class customers”, where “most” is the linguistic quantifier that represent 95% of the records in the database. Linguistic quantifiers can be represented as a fuzzy set defined over the unit interval and have been used to define fuzzy summaries in the language Summary SQL. In this work, the confidence factor φ^i_{jk} represents how much the term $A_{ij} \in \mathbf{A}_i$ is linked to the class $C_k \in \mathbf{C}$ in the model defined by the rule (2). A value $\varphi^i_{jk} > 0$ means that the observation of the term A_{ij} is related with the occurrence of the class C_k in φ^i_{jk} of the records. A set of rules (or a rule base) for each input variable defines a sub-model that is represented by the matrix Φ_i as shown in Table 1. In such matrix, the lines $j = 1 \dots n_i$ are related to the terms in the input variable descriptor set $\mathbf{A}_i$ and the columns $k = 1 \dots m$ are related to classes in the set $\mathbf{C}$, such that $\Phi_i(A_{ij}, C_k) = \varphi^i_{jk}$.

A rule base is defined for each input variable and used to compute partial outputs by fuzzy inference. Fuzzy rules with more than one variable in the antecedent allow representing interactions between input variables but the resulting rule base can grow exponentially for large problems. A trade off between the size and the number of rules in the rule base is a complex optimization problem that is out of the scope of this work.

Table 1: Rule base weights' matrix.

Φ_i	C_1	...	C_m
A_{i1}	$\Phi_i(A_{i1}, C_1)$	...	$\Phi_i(A_{i1}, C_m)$
A_{in_i}	$\Phi_i(A_{in_i}, C_1)$	...	$\Phi_i(A_{in_i}, C_m)$

Fuzzy inference: The fuzzy classifier output is represented by the class membership vector $\mathbf{y}(t) = (\mu_{C_1}(\mathbf{x}(t)), \ldots, \mu_{C_m}(\mathbf{x}(t)))$. Each component $\mu_{C_k}(\mathbf{x}(t))$ is the membership of a given input record $\mathbf{x}(t)$ to the class C_k.

The vector $\mathbf{y}_i(t) = (\mu_{C_1}(x_i(t)), \ldots, \mu_{C_m}(x_i(t)))$ is the partial output membership vector whose components are the classes' membership values considering only the information in the input variable *i*.

The output of each sub-model is computed by composition-projection inference:

$$\mathbf{y}_i(t) = \mathbf{u}_i(t) \circ \Phi_i \tag{3}$$

The composition-projection operation computed by the standard max-min composition operator as:

$$\mu_{C_k}(x_i(t)) = \max_{j=1\ldots n_i}\left(\min\left(\mu_{A_{ij}}(x_i(t)), \Phi_i(A_{ij}, C_k)\right)\right) \tag{4}$$

Equation (4) computes the degree of matching of the fuzzification vector $\mathbf{u}_i(t)$ with the prototype of the class C_k, represented by the corresponding column of the rule base matrix Φ_i.

The final output is computed by the aggregation of all partial outputs by an aggregation operator $\mathbf{H} : [0,1]^p \rightarrow [0,1]$ as:

$$\mu_{C_k}(\mathbf{x}(t)) = \mathbf{H}\left(\mu_{C_k}(x_1(t)), \ldots, \mu_{C_k}(x_p(t))\right) \tag{5}$$

The best aggregation operator must be chosen according to the semantics of the application. A t-norm operator, such as the "minimum", gives good results to express that all partial conclusions must agree. In classification problems, the final decision is computed by a decision rule. The most usual decision rule is the "maximum rule", where the class is chosen as the one with greatest membership value (Figure 3).

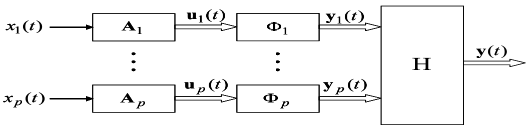


Figure 3: The weighted fuzzy classifier.

The translation of fuzzy rules into fuzzy queries is presented next.

Writing fuzzy queries from weighted fuzzy rules Most of the fuzzy query languages are structured to represent sentences in the form:
SELECT <Atributes> FROM <table> WHERE: <expression>

The attribute list and the table list are usually similar to standard SQL. The expression is a fuzzy logic sentence that is translated from the fuzzy classifier's rules in the proposed method.

Non-weighted fuzzy rules may be directly translated into fuzzy queries. Weighted fuzzy rules may not be directly translated since most of fuzzy query languages do not support weights. The weighted fuzzy rules must be converted into non-weighted fuzzy rules. This conversion is based on the conjunctive interpretation of fuzzy if-then rules, where the implication operator is computed as a t-norm operator. Under this interpretation, a weighted rule may be converted into a non-weighted one by introducing the confidence into the rule premise, such that the rule originally written as (2) is converted to:

$$\textit{if} \quad x_i(t) \quad \textit{is} \quad A_{ij} \quad \textit{and} \quad w_{ijk} \quad \textit{is} \quad \textit{TRUE} \quad \textit{then} \quad \textit{class} \quad \textit{is} \quad C_k \tag{6}$$

where the auxiliary variable w_{ijk} is defined for each rule such that:

$$\mu_{TRUE}(w_{ijk}) = \phi^i_{jk} = \Phi_i(A_{ij}, C_k) \tag{7}$$

The membership value $\mu_{TRUE}(w_{ijk})$ is thus independent of the input variable observation $x_i(t)$ and can be stored in the database.

The AND operator in rule (6) is computed by a t-norm operator and a set of rules in the rule base is aggregated using a t-conorm operator, resulting in the max-min composition operator (4). The translation of a set of fuzzy rules (6) into a fuzzy query sentence must ensure that the result of the query is equivalent to the classifier result. A set of fuzzy rules like (6) are translated into a set of expressions aggregated by disjunctive (OR) operators as: `((x1 is A11 AND w11k is TRUE) OR ...OR (x1 is A1n`$_1$` AND w1n`$_1$`k is TRUE))`

The aggregation between rules is computed by an aggregation operator (5), which is a conjunctive operator (AND) between sentences. The fuzzy query evaluation is thus equivalent to the classifier result. The fuzzy query sentence for the class C_k is: `SELECT <Attributes> FROM <Data Base> WHERE:`

```
((x1 is A11 AND w11k is TRUE) OR ...OR
 (x1 is A1n₁ AND w1n₁k is TRUE)) AND ...
          ... AND ...
((xi is Ai1 AND wi1k is TRUE) OR ... OR
 (xi is Ainᵢ AND winᵢk is TRUE)) AND ...
          ... AND ...
((xp is Ap1 AND wp1k is TRUE) OR ... OR
    (xp is Apnₚ AND wpnₚk is TRUE)).
```

This solution can be easily implemented in most of fuzzy query languages.

The rule base weights are the core of the model described by the fuzzy classifier and their determination from a data set is described in the next section.

2.3 Weights estimation

The rule base weights are computed from a training data set T, where each sample $t = 1..N$ is a pair $(\mathbf{x}(t), \mathbf{v}(t))$, of which $\mathbf{x}(t)$ is the input vector for each record and $\mathbf{v}(t) = (v_1(t), \ldots, v_m(t))$ is the vector containing the correct membership values of $\mathbf{x}(t)$ to each class. In most practical applications, the correct output $\mathbf{v}(t)$ is binary, like the fuzzification of nominal variables.

Fuzzy rule base weights may be computed in many ways. In this work, for each input variable, each component $\Phi_i(A_{ij}, C_k)$ of the rule base Φ_i is computed as
[5]:

$$\Phi_i(A_{ij}, C_k) = \frac{\sum\limits_{t=1..N} u_{ij}(t).v_k(t)}{\sum\limits_{t=1..N} u_{ij}(t)} \tag{8}$$

where $u_{ij}(t) = \mu_{A_{ij}}(x_i(t))$ and $v_k(t)$ is the correct membership of the sample t to the class C_k.

Considering p input variables, each one with n fuzzy sets and m classes, there will be $p.n.m$ rules. Equation (8) must be computed for every rule in the fuzzy rule base, in order the model to be complete.

The final output is an aggregation of all input variables to compute the final class membership. When this aggregation is computed by a t-norm operator (like the minimum or product) the final class membership is a rough estimation of the joint conditional probability of each class, given the observation of all input variables. The expression in (8) has been referred as the $\sum \text{count}$, and has been used to compute the value of relative quantifiers [4]. Linguistic quantifiers can be processed by some fuzzy query languages such as Summary SQL to compute fuzzy summaries. In this work, the fuzzy rules are translated into fuzzy queries in such a way that they can be processed by any fuzzy query language with an equivalent result to the classifier.

The rule base weights are computed for all possible rules. There are generally a large number of possible rules, many of that are useless for the classification, which makes difficult the interpretation of the classifier model and the corresponding query sentence. A pruning procedure must consider the most important rules to generate a compact set of sentences in the fuzzy query.

2.4 Rule base pruning

The pruning of fuzzy rules and simplification is a very active research area. In this work, the pruning of fuzzy rules is necessary to allow a more compact set of query sentences. The pruning procedure is based on the values of two indexes. The Horizontal index (I_H), is given by:

$$I_H(x_i, A_{ij}) = \frac{\sum_{k=1}^{m}\left(\max_{k=1..m}\left(\Phi_i(A_{ij}, C_k)\right) - \Phi_i(A_{ij}, C_k)\right)}{m-1} \quad (9)$$

The Vertical index (I_V), is computed as:

$$I_V(x_i, C_k) = \frac{\sum_{j=1}^{n_i}\left(\max_{j=1..n_i}\left(\Phi_i(A_{ij}, C_k)\right) - \Phi_i(A_{ij}, C_k)\right)}{n_i-1} \quad (10)$$

The horizontal index $I_H \in [0,1]$ measures how much the fuzzy set A_{ij} of the variable x_i is able to discriminate among the classes. The vertical index $I_V \in [0,1]$ measures how much a class C_k is detected according to the possible values of the variable x_i. Fuzzy rules can be pruned by selecting a threshold values for one of these indexes. For general-purpose classifiers, the horizontal index should be used since it allows selecting the fuzzy rules that will result in a better classification performance. For fuzzy queries, however, the queries are executed for a given class independently to retrieve the records that are the best representatives of that class. Thus, the vertical index should be used to select the rules that best discriminates the records of a given class.

2.5 Fuzzy query evaluation

The fuzzy query evaluation is based on the standard metrics for evaluation of information retrieval systems: precision and Recall. The Precision metric is defined as the number of relevant records selected as a fraction of the total number of selected records. The Recall metric is defined as the number of relevant records selected as a fraction of the total number of relevant records.

The Precision and Recall metrics can be computed directly from the confusion matrix, presented in Table 2, which is similar to the one usually used to evaluate classification systems. The rows represent the results of the query system and the columns represent the true information about the selected records.

The values in the Table 2 are the standard ones: *TP* stands for the number of true positive samples, *FP* is the number of false positive samples, *TN* is the number of true negative samples and *FN* is the number of false negative samples.

Table 2: Confusion matrix.

	Relevant	**Not Relevant**
Selected	*TP*	*FN*
Not Selected	*FP*	*TN*

The Precision and Recall metrics are computed as High Precision and Recall rates are desired, but it is generally very difficult to achieve high values of both metrics simultaneously: as the Recall rate increases, the Precision usually decreases and vice-versa.

$$P = \frac{TP}{(TP + FN)} \quad (11)$$

$$R = \frac{TP}{(TP + FP)} \tag{12}$$

Differently from a standard SQL query, a fuzzy query returns all the records in the database, even those associated to a very small membership value. In a practical application, it is necessary to set a membership threshold or the maximum number of returned records. These parameters must be set to run the query. The Precision and Recall metrics, when computed over the training set, are useful to give the user an insight of the membership threshold to be set in the fuzzy query. Moreover the user can adjust the threshold to their own needs, based on the results of the training set, which is an impossible using a standard SQL query.

3 Computational implementation

Great companies of current market use intelligent systems in their technology departments, also known as Intelligent Business Systems (IB).

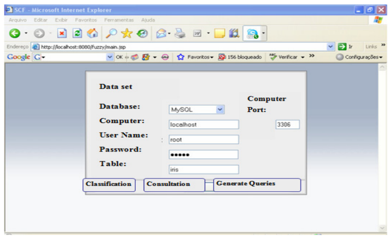
These Intelligent Business Systems have revolutionized the interaction between the user and the machine, providing strategic and intelligent information for various users with specialization on several distinct areas, such as management, financial, law, administrative, human resources etc. The language of these systems is more natural nowadays, that is, it is much closer to men’s written language, and they are used in these systems such that these Intelligent Business Systems (IB) can be used in a simple, interactive, direct, automated and visual way, extracting the necessary information, without any concern about technological factor built-in those systems. The Fuzzy_Query.Web Tool carries out all the processes of a Fuzzy Classification System (FCS), including fuzzification of the numerical values, assembly of the base of rules – that is performed by a learning algorithm and the accomplishment of the fuzzy inference, as well as the aggregation of the partial conclusions. In order to follow the technological trends and the increasing use of world wide web by users located anywhere in the world, the Web-Server environment was chosen for the development of Fuzzy_Query.Web software. The FQW software was developed using Java language. Java became a worldwide standard due to its portability in various existing platforms. Java net resources allow the execution of programs through the internet with restrictions, and they have an extensive library of routines that facilitate the cooperation with TCP/IP protocols, like HTTP and ftp.

3.1 Fuzzy_Query.Web software

Fuzzy_Query.Web software allows flexibility in consultations carried through diverse systems of relational database by means of a graphical interface developed in Java and in its JDBC driver, that uses the portability offered by its technology, accessed through the Web. Fast access and the presence of useful database characteristics, as well as the possibility of customization are some of the strong characteristics of this system. The choice of attributes, class and the fuzzy partitions to be used for the system is available to the user. As a result of these consultations, the SQL Fuzzy is generated and shown to the user automatically. SQL Fuzzy is an adapted language in SQL standard that allows Fuzzy consultations through the Fuzzy_Query.Web software. The user does not need any knowledge on SQL language, because SQL Fuzzy translates this language to the user. Fuzzy query is built and shown automatically through Fuzzy_Query.Web software. All information for the organization of fuzzy query is got by the value of the relevance of the attributes in its partitions, and according to the chosen class, in which they will be associated to the Query standard of the SQL for the accomplishment of the calculations using the operators AND (min) and OR (max), adding the generated rules and allowing the exhibition of all degrees of relevance of the pieces of information being processed.

4 Evaluation of results

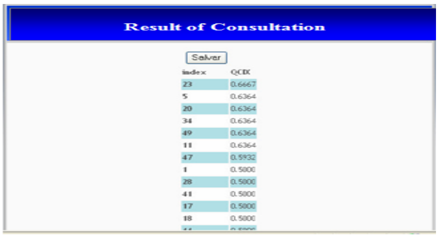
The developed Fuzzy_Query.Web system was tested by the use of distinct databases. These databases are linked to medicine, trade, financial and education fields. These various databases were obtained from the well-known UCI Machine Learning Repository. They were used as benchmarks to show the effectiveness of our software. The choice of the databases followed some criteria in order to have different attributes and classes, that is, they used literal or numerical attributes or classes.


SCF - Microsoft Internet Explorer
Arquivo Editar Exibir Favoritos Ferramentas Ajuda
Endereço http://localhost:8080/fuzzy/main.jsp
Ir
Links
Favoritos
156 bloqueado
Verificar
Configurações
Classification
Consultation
Generate Queries

Data set

Database:	MySQL
Computer:	localhost
User Name:	root
Password:	•••••
Table:	iris
Computer Port:	3306

Figure 4: Main screen of the system: Connect to the database (Iris), and has the following options of classification, consulting, generator consultation fuzzy query.



Result of Consultation

index	QCIX
23	0.6667
5	0.6364
20	0.6364
34	0.6364
49	0.6364
11	0.6364
47	0.5932
1	0.5000
28	0.5000
41	0.5000
17	0.5000
18	0.5000
[illegible]	[illegible]

Figure 5: Result of consultation of fuzzy query tool (base iris-Class 1). The response given by our system to this consultation shows the calculation of training and test sets: the inference and aggregation.

The graph of the Training and Test Errors Iris shown in Figure 6: The small error in the training and test sets shows the good performance of the classifier for the iris base. Analyzing the matrix of confusion, we see that the errors for classes 2 and 3 are small, but the error on Class 1 (error = 0) is even smaller. This agrees with the performance indicators observed in this experiment.

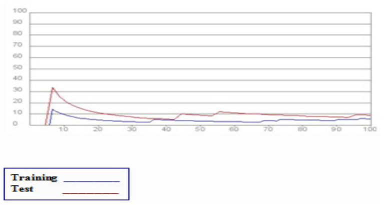
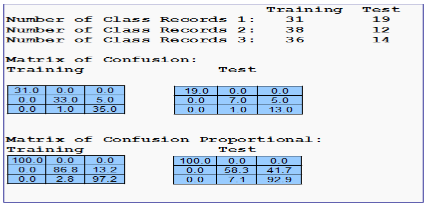


Figure 6: Graph of the training and test errors iris (base iris).



```
                                    Training   Test
Number of Class Records 1:            31        19
Number of Class Records 2:            38        12
Number of Class Records 3:            36        14

Matrix of Confusion:
Training                         Test

| 31.0 | 0.0  | 0.0  |          | 19.0 | 0.0 | 0.0  |
| 0.0  | 33.0 | 5.0  |          | 0.0  | 7.0 | 5.0  |
| 0.0  | 1.0  | 35.0 |          | 0.0  | 1.0 | 13.0 |

Matrix of Confusion Proportional:
Training                         Test
| 100.0 | 0.0  | 0.0  |         | 100.0 | 0.0  | 0.0  |
| 0.0   | 86.8 | 13.2 |         | 0.0   | 58.3 | 41.7 |
| 0.0   | 2.8  | 97.2 |         | 0.0   | 7.1  | 92.9 |
```

Figure 7: Results matrix of confusion: the confusion matrix of iris base generates the table of relevant data and not relevant to the graphic error.

5 Conclusion

In this paper software was proposed and implemented to perform fuzzy queries in order to improve relational databases performance and use. On the other hand, the usability of the system by the users is ensured. Many simulations were performed in order to check Fuzzy_Query.Web system. Through simulations results, its implementation allowed a great connection between fuzzy methodology and relational database. As future work, the implementation of other classifiers in the Fuzzy_Query.Web tool, such as neural net and genetic algorithm. Trapezoids, Gaussian functions, among others also must be implemented in order to compare the results.

References

[1] R. A. Ribeiro and A. M. Moreira. Fuzzy query interface for a business database. *Int. J. Human-Computer Studies* 58 pp. 363–391, 2003.
[2] Y. Takahashi. A fuzzy query language for relational databases. *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 21 No. 6, pp. 1576 - 1579, 1991.
[3] D.-Y. Choi. Enhancing the power of Web search engines by means of fuzzy query. *Decision Support Systems*, 35, pp. 31-44, 2003
[4] L. A. Zadeh. A computational approach to fuzzy quantifiers in natural languages. *Computers and Mathematics*. 9, pp. 149-184, 1983.
[5] A. G. Evsukoff, A. C. S. Branco and S. Gentil. A knowledge acquisition method for fuzzy expert systems in diagnosis problems, *IEEE Fuzzy Systems, FUZZ-IEEE*, Barcelona, July 1997.
[6] M. J. A. Berry, G. Linoff. *Data Mining Techniques: for Marketing, Sales, and Customer Support*, Wiley Computer Publishing, 1997.