Generating Fuzzy Queries from Weighted Fuzzy Classifier Rules

Antonio C. S. Branco
COPPE/Federal University of Rio de Janeiro, Brazil
saraivabranco@gmail.com

Alexandre G. Evsukoff
COPPE/Federal University of Rio de Janeiro, Brazil
evsukoff@coc.ufrj.br

Nelson F. F. Ebecken
COPPE/Federal University of Rio de Janeiro, Brazil
nelson@ntt.ufrj.br

Abstract

This work presents a methodology to generate automatically a set of fuzzy queries that are translated from a set of fuzzy rules learned from a data set. A rule simplification approach is proposed, which allows a better interpretability of the fuzzy query sentence. The fuzzy query sentences are generated in such a way that they can be processed by a standard fuzzy query system. The methodology is applied to a benchmark data set considering a scenario of target selection in a direct market application. The results obtained by the current approach are compared with the results of a standard a decision tree induction algorithm.

1. Introduction

The integration of Data Mining (DM) tools with Data Base Management Systems (DBMS) is now more than trend, it is a reality. The major DBMS vendors have already integrated DM solutions within their products. On the other hand, the main DM suites have also provided the integration of DM models into DBMS trough modeling languages such as the Predictive Model Markup Language (PMML). It is thus a fact that future research on new DM tools and methods must consider their integration with DBMS.

On the application side, data intensive industries, such as insurance, accounting and telecommunications, among others, need frequently to retrieve their costumers for different marketing relationship actions. Defining a query sentence that captures the main features of a subset of records is a non trivial task. Generally, however, past experiences can be used to guide the new actions, such that queries generation can thus be regarded as a DM task.

Structured Query Languages (SQL) provides a structured description of the retrieved records. A standard SQL sentence representing a concept can be translated from rules generated by an algorithm for decision-tree or rule induction. The translation of machine learning rules into SQL sentences is straightforward: for each class, each rule corresponds to a sentence (in the WHERE clause), and all rules related to the same class are aggregated with a disjunctive (OR) operator. A standard query, however, will return all the records that match the query sentence. However, it is frequently desirable to rank those records, such that the manager is able to define priorities.

Neural networks and Bayesian classifiers are also frequently found in most of DM suites. Such models may be coded into DBMS, via PMML, to retrieve a ranked list of records. Nevertheless, neural networks and Bayesian classifiers models are not linguistically understandable, such that managers cannot directly validate the knowledge extracted by the DM algorithm.

Fuzzy queries have emerged in the last 25 years to deal with the necessity to soften the Boolean logic in relational databases. A fuzzy query system is an interface to human users to get information from database using (quasi) natural language sentences [1][2][3]. Many fuzzy queries implementations have been proposed, resulting in slightly different languages such as SQLf [4], FQUERY [5] and Summary SQL [6][7], among others. Although some variations according to the particularities of different implementations, the answer to a fuzzy query sentence is a generally a list of records, ranked by the degree of matching.

The focus of attention of the fuzzy query research has been on the expressive power of the fuzzy query language, since the query is usually provided by the user. Nevertheless, the definition of a query to retrieve the records according to their features' values is a very difficult task. Not due to the lack of expressive power of the query language, but due to the difficulty on defining the concept itself.

Fuzzy queries sentences are structured definitions of fuzzy concepts. Under this assumption, fuzzy queries can be automatically generated by fuzzy rule based classifiers.

Fuzzy rule based classifiers are a very active research field [8][9][10]. The weighted fuzzy rule based approach allows better classification results since it allows to define more precisely the decision boundary among the classes [11][12]. The translation of weighted fuzzy rules into fuzzy queries is not straightforward since most of fuzzy queries languages do not support weights.

In this work, it is proposed a methodology to translate a set of rules computed by a weighted fuzzy classifier into fuzzy queries, such that they can be coded into most of available fuzzy query languages. The whole procedure is sketched in Figure 1. The left side of the figure shows an

existing fuzzy query system (denoted Fuzzy SQL) used to process the fuzzy queries connected to a DBMS. The proposed methodology is showed in the right side of the figure. It is considered that a labeled database exists from which a training set may be selected to generate a set of fuzzy rules by the fuzzy classifier. The set of rules generated by the fuzzy classifier may be very large, containing many useless rules. The fuzzy rule base is thus pruned and then translated into a set of fuzzy query sentences. The fuzzy query system is used to retrieve an ordered set of records from a new and unlabelled database, corresponding to a given class.

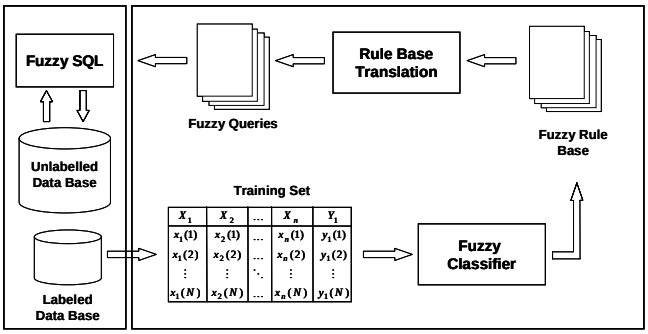

Figure 1. The proposed approach.

The remainder of this paper is organized as follows. Next section presents the weighted fuzzy rule-based classifier and the translation of fuzzy rules into fuzzy queries. Section 3 describes the learning algorithm of fuzzy rules and the pruning procedure. Section 4 presents the results of the application of the proposed methodology for a set of benchmark data sets and a case study using the Credit Card data set. The fuzzy queries results are compared to standard queries generated by a standard decision-tree induction algorithm. Finally some conclusions are drawn.

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 classes' set $\mathbf{C} = \{C_1, \ldots, C_m\}$. The solution to the classification problem is to assign 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 correspond 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 [8]. Under this approach, the output of the fuzzy classifiers is computed in two steps:

1. for each input, compute partial outputs as the degree of matching of the observed input value to each class,
2. 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:

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

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 (see Figure 2).

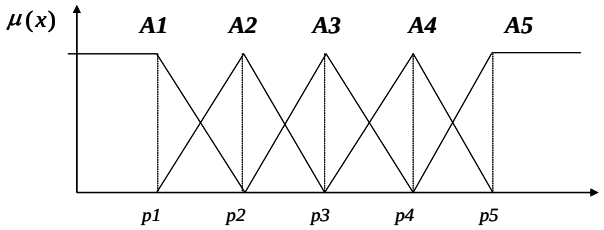

Figure 2: Example of fuzzy partition.

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:

$$\textit{if } x_i(t) \textit{ is } A_{ij} \textit{ then class is } C_k \textit{ with } cf = \varphi_{jk}^i . \quad (2)$$

where $\varphi_{jk}^i \in [0,1]$ 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 [13]. 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 [6].

In this work, the confidence factor φ_{jk}^i 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 $\varphi_{jk}^i > 0$ means that the observation of the term A_{ij} is related with the occurrence of the class C_k in φ_{jk}^i 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 $j = 1...n_i$ are related to the terms in the input variable descriptor set $\mathbf{A}_i$ and the columns $k = 1...m$ are related to classes in the set $\mathbf{C}$, such that $\Phi_i(A_{ij}, C_k) = \varphi_{jk}^i$.

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}	$\Phi_i(A_{in_i}, C_1)$	…	$\Phi_i(A_{in_i}, C_m)$

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 grows 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.

2.3. Fuzzy inference

The fuzzy classifier output is represented by the class membership vector $\mathbf{y}(t) = \left(\mu_{C_1}(\mathbf{x}(t)), \ldots, \mu_{C_m}(\mathbf{x}(t))\right)$. 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 $\mathbf{y}_i(t) = \left(\mu_{C_1}(x_i(t)), \ldots, \mu_{C_m}(x_i(t))\right)$ 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:

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

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

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

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 $\mathbf{H}: [0,1]^p \to [0,1]$ as:

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

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).

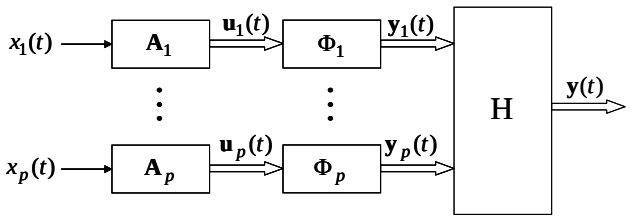

Figure 3. The weighted fuzzy classifier.

The translation of fuzzy rules in to fuzzy queries is presented next.

2.4. Writing fuzzy queries from weighted fuzzy rules

Most of the fuzzy query languages are structured to represent sentences in the form [4][5][6]:

```
SELECT <Atributes> FROM <table>
WHERE:
  <expression>
  <expression>
  …
  <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:

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

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

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

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 to a set of expressions aggregated by disjunctive (OR) operators as:

```
((x1 is A11 AND w11k is TRUE) OR ...OR
   (x1 is A1n₁ AND w1n₁k 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.

3. Rule Base Learning

This section comments the estimation of the rule base weights from a data set. The weights estimation procedure computes weights for all possible rules. Generally a large number of fuzzy rules is generated, which makes difficult the linguistic interpretation of the model. A prune procedure is thus also proposed to reduce the rule base size and, consequently, the size of the corresponding fuzzy query.

3.1. 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 variables 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 of 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 [10]. In this work, for each input variable, each component $\Phi_i(A_{ij}, C_k)$ of the rule base Φ_i is computed as [14]:

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

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.

In a probabilistic framework, equation (8) for a nominal input variable x_i is computed as:

$$\Phi_i(A_{ij}, C_k) = \frac{N_{(A_{ij} \cap C_k)}}{N_{A_{ij}}}. \quad (9)$$

where $N_{(A_{ij} \cap C_k)}$ is the number of samples in the training set that have the value A_{ij} for the variable x_i and are classified as class C_k; and $N_{A_{ij}}$ is the total number of samples that have the value A_{ij} for the variable x_i.

The conditional probability of the class C_k, given the observation of the value A_{ij}, computed from Bayes rule is:

$$P(C_k \mid A_{ij}) = \frac{P(A_{ij} \mid C_k).P(C_k)}{P(A_{ij})} = \frac{\frac{N_{(A_{ij} \cap C_k)}}{N_{C_k}} \frac{N_{C_k}}{N}}{\frac{N_{A_{ij}}}{N}} \quad (10)$$

where N_{C_k} is the total number of samples classified as class C_k and N the total number of samples in the training set.

Comparing equations (8) and (10), it can be seen that the rule base weights are an estimation of the *a posteriori* probability of the class occurrence given by the observation of the value A_{ij}, and:

$$\Phi_i(A_{ij}, C_k) \approx P(C_k \mid A_{ij}). \quad (11)$$

Equality in (11) is achieved for nominal variables. For continuous input variables, the result for rules' output provides the interpolation of the conditional probabilities, weighted by the membership to the corresponding fuzzy set.

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 $\sum$count, and has been used to compute the value of relative quantifiers [13]. Linguistic quantifiers can be processed by some fuzzy query languages such as Summary SQL [7] 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.

3.2. Rule Base Pruning

Fuzzy rules' pruning and simplification is a very active research area [15][16]. In this work, fuzzy rules' pruning 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), computed as:

$$I_H(x_i, A_{ij}) = \frac{\sum_{k=1}^{m} \left(\max_{k=1..m} \left(\Phi_i(A_{ij}, C_k) \right) - \Phi_i(A_{ij}, C_k) \right)}{m-1} \quad (12)$$

- The Vertical index (I_V), computed as:

$$I_V(x_i, C_k) = \frac{\sum_{j=1}^{n_i} \left(\max_{j=1..n_i} \left(\Phi_i(A_{ij}, C_k) \right) - \Phi_i(A_{ij}, C_k) \right)}{n_i - 1} \quad (13)$$

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 best representatives' records of that class. Thus, the vertical index should be used to select the rules that best discriminates the records of a given class.

3.3. 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:

$$P = \frac{TP}{(TP + FN)}. \quad (14)$$

$$R = \frac{TP}{(TP + FP)}. \quad (15)$$

Precision and Recall rates are both 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.

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.

4. Results and Discussion

The evaluation of the current approach is performed under two perspectives. The fuzzy weighted classifier is evaluated from a set of benchmark classification data sets and results are compared with the J4.8 decision tree induction and the Naïve Bayes algorithms, both computed within the Weka suite [17]. The evaluation of the fuzzy query generated by the proposed approach is performed under a target marketing scenario, using the Australian Credit Card data set. The fuzzy query results are compared with the results of the J4.8 decision tree induction algorithm.

4.1. Fuzzy Weighted Classifier Evaluation

A fuzzy weighted classifier, as described above, was generated for a set of benchmarks data sets. For all data sets the fuzzy classifier was generated using $n = 5$ fuzzy membership function of all numerical variables.

Table 3 presents, for each benchmark data, the number of variables, the number of classes and the error rates for the Fuzzy Weighted Classifier (FWC), for the J4.8 algorithm and for the Naïve Bayes (NB) algorithm. The J4.8 and NB algorithms were computed with all default parameters and the results in Table 3 are the average values of the 10-fold cross validation for both algorithms. Bold values indicate the best values for each benchmark.

The FWC will have a better accuracy than the decision tree induction algorithm when numerical variables are predominant and variables are roughly statistically independent [14]. The NB algorithm performs better when the variables are independent and probabilities' densities may be well approximated by the normal probability distribution function. Nevertheless, in general, the classifiers' performance depends mainly on the characteristics of the problem.

Table 3: Classifiers' results

Benchmark	# Vars	# Class	FWC	J4.8	NB
Abalone	8	3	44.00	**39.70**	42.25
Balance Scale	4	3	**36.46**	43.20	36.48
Credit Card	15	2	**13.63**	14.20	23.27
Prima indians	8	2	31.38	25.90	**23.69**
Ionosphere	34	2	**6.27**	8.55	29.91
Breast Cancer	9	2	**5.76**	8.40	5.75
Glass	9	7	47.64	**29.40**	51.87
Wine	13	3	5.62	7.90	**2.80**

The results in Table 3 were obtained without rule base pruning. Although, fuzzy rule base pruning may slightly improve the performance of the FWC [12][16], it may generate "roles" in the model. In this work rule base pruning is used only for the query generation.

4.2. Fuzzy Query Generation

In order to evaluate the fuzzy queries generated by the proposed approach, the Credit Card approval data set was used. This data set contains 690 examples, each with 6 numeric and 9 categorical attributes. For confidentiality reasons, the attributes' descriptions have been omitted. The records are originally classified into two classes: "approved" (C_1) or "rejected" (C_2), respectively with 45.3% and 54.7% of *apriori* probability. The data set was randomly divided into a training data set and a testing data set containing respectively 460 and 230 records with the same class distribution.

As a "proof of concept", the Fuzzy Query™ tool [20] is used to implement the fuzzy queries generated by the method. The software is a Win32-based application designed to run in conjunction with any ODBC-compliant database management systems.

The fuzzy rule based classifier computes a solution with all rules. The number of rules generated by the classifier is very large, which makes difficult the interpretation. The pruning procedure presented above was thus applied to the fuzzy rule base. The original rule base with 75 rules for each class was thus reduced to 10 rules by selecting the rules where $I_V > 0.5$. The decision tree induced by the J4.8 algorithm has 30 rules.

The simplified rule base was translated into a fuzzy query for class C_1. The resulting fuzzy query is very simple and represents the "good customer" model generated by the fuzzy classifier in query sentences as presented in section 2.4. The understanding of the queries generated in this example is not possible since the interpretations of variables are not available for this benchmark.

4.3. Fuzzy Query Evaluation

Direct marketing has become one of potential applications of data mining techniques, due to the great volume of available data from transactional databases [18][19]. The objective of direct marketing is to contact customers to offer products and services according to their specific needs. In a direct marketing campaign, the potential customers must be selected from the entire database in order to maximize the profit and minimize the losses due to an overcharge of marketing material sent to wrong customers. Customers' selection or generally "target" selection is thus an important issue in direct marketing and several data mining techniques may be applied to model customer profile according to the objective of the campaign.

The evaluation of the fuzzy queries generated by the proposed approach is done for a target marketing scenario, using the Credit Card data set. In this data set, the relevant class for the application discussed in this section is the class C_1, which represents credit approvals, such that the fuzzy query designed for the class C_1 should not select a record from the class C_2.

The Precision and the Recall metrics as a function of the membership values of the returned records for the training set are shown in Figure 4. The results show that, as expected, as grater the membership value is, the higher is the Precision of the selected records, but the lower is the Recall. An optimal threshold value can thus be chosen to allow high Precision and Recall rates. In this application, the optimal threshold value must be chosen around 0.25.

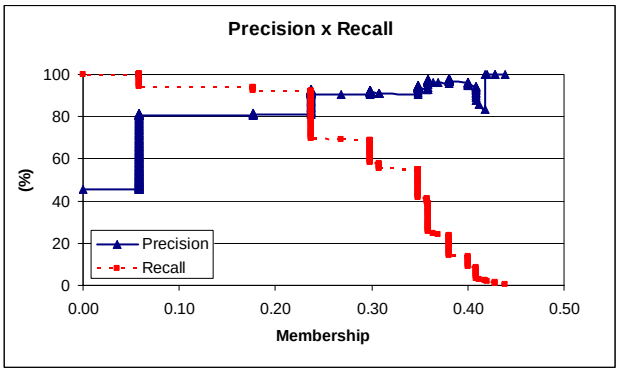

Figure 4: Precision and Recall for the training set.

The Precision and Recall measures computed by the fuzzy weighted query (Fuzzy SQL) with two different thresholds values over the testing set are shown in Table 4. The Precision and Recall metrics computed by the J4.8 algorithm over the same testing set are also shown in Table 4. It can be seen that the fuzzy query results over the testing set correspond roughly to the ones computed from the training set. Moreover the selection of the threshold allows the user to choose between a greater Precision and Recall value, while the J4.8 result is fixed.

The relative importance of Precision and Recall for a certain application depends on the user. Experienced users can work with high recall and low precision, as they are capable to reject irrelevant information. Novice users, on the other hand, need higher precision because they lack the judgment of the experienced users.

Table 4: Fuzzy query evaluation in the testing set

	Precision (%)	Recall (%)
Fuzzy SQL $\delta = 0.25$	92.30	60.60
Fuzzy SQL $\delta = 0.20$	79.27	88.88
J4.8	89.10	75.20

In most of applications in data intensive industries, the number of returned records is very large. Thus it is necessary to select the more suitable record for a given application. As it can be seen from Figure 4, the precision of the records returned by the fuzzy query with grater membership values is very large.

The decision tree induced by the J4.8 algorithm could be used to construct a standard SQL query to retrieve potential customers in the data set. Nevertheless, the query result would not be ordered as the fuzzy query result.

5. Conclusions

This work has shown a methodology that automatically generates fuzzy queries from a training data set. The fuzzy queries are translated from a set of fuzzy weighted classifier rules. A pruning procedure to simplify the fuzzy rule base and the resulting set of fuzzy queries was also proposed.

This methodology has been implemented in a general-purpose fuzzy query tool, which can be connected to any ODBC-compliant database management system.

The fuzzy queries are able to retrieve an ordered list of the records according to the fuzzy rule based model learned by the fuzzy classifier. The Precision and Recall analysis of the selected records of the fuzzy query allow the user to select an optimal threshold for other data.

The proposed methodology was developed focusing direct marketing application, but may be useful in other kind of application such as case based reasoning or information retrieval.

The future directions of this work are to apply this methodology in unstructured data for application to text mining and web mining.

Acknowledgments

The authors are grateful to the Brazilian Research Agencies FINEP, CNPq and CAPES for the financial support for this research.

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] P. Bosc and O. Pivert. SQLf: a relational database language for fuzzy querying. *IEEE Transactions on Fuzzy Systems*, Vol. 3 No. 1 pp. 1-17, 1995.

[5] J. Kacprzyk and S. Zadrozny. SQLf and FQUERY for Access. *Proceedings of the Joint 9th IFSA World Congress and 20th NAFIPS International Conference*, Vol. 4 pp. 2464 - 2469, 2001.

[6] D. Rasmussen and R. R. Yager. Summary SQL - A Fuzzy Tool for Data Mining. *Intelligent Data Analysis*, 1, pp. 49-58, 1997.

[7] D. Rasmussen and R. R. Yager. Finding fuzzy and gradual functional dependencies with Summary SQL. *Fuzzy Sets and Systems*, 106, pp. 131-142, 1999.

[8] D. Dubois, H. Prade, C. Testemale, Weighted fuzzy pattern matching, *Fuzzy Sets and Systems* 28, pp. 313–331, 1988.

[9] K. Nozaki, H. Ishibuchi and H. Tanaka. Adaptive fuzzy rule-based classification systems. *IEEE Transactions on Fuzzy Systems*, Vol. 4, No. 3, pp. 238 – 250, 1996.

[10] A. Devillez. Four fuzzy supervised classification methods for discriminating classes of non-convex shape. *Fuzzy Sets and Systems*, Vol. 141, No. 2, pp. 219-240, 2004.

[11] H. Ishibuchi and T. Nakashima. Effect of rule weights in fuzzy rule-based classification systems. *IEEE Transactions on Fuzzy Systems*, Vol. 9, No. 4, pp. 506 – 515, 2001.

[12] H. Ishibuchi and T. Yamamoto. Rule Weight Specification in Fuzzy Rule-Based Classification Systems, *IEEE Transactions on Fuzzy Systems*, Vol. 13, No. 4, pp. 428 – 435, 2005.

[13] L. A. Zadeh. A computational approach to fuzzy quantifiers in natural languages. *Computers and Mathematics with Application*. 9, pp. 149-184, 1983.

[14] A. G. Evsukoff, A. C. S. Branco and S. Gentil. A knowledge acquisition method for fuzzy expert systems in diagnosis problems, *IEEE International Conference on Fuzzy Systems, FUZZ-IEEE'97*, Barcelona, July 1997.

[15] W. E. Combs, J. E Andrews. Combinatorial explosion eliminated by a fuzzy rule configuration, *IEEE Trans. on Fuzzy Systems*, Vol. 6 No. 1, pp. 1-11, 1998.

[16] R. P. Espíndola, N. F. F. Ebecken. Data classification by a fuzzy genetic system approach, *Third International Conference on Data Mining*, Bologna, 2002.

[17] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[18] M. J. A. Berry, G. Linoff. *Data Mining Techniques: for Marketing, Sales, and Customer Support*, Wiley Computer Publishing, 1997.

[19] U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth and R. Uthurusamy. *Advances in Knowledge Discovery and Data Mining*, AAAI Press/The MIT Press, 1996.

[20] Sonalysts Inc. Fuzzy Query for Windows ((http://fuzzy.sonalysts.com/fuzzyquery.htm), 1998.