Discovering Interesting Knowledge from a Science & Technology Database with a Genetic Algorithm

Wesley Romão[1], Alex A. Freitas[2], Itana M. de S. Gimenes[3]

[1,3]Computer Science Dept., Universidade Estadual de Maringá, Brazil
[2]Computing Laboratory, University of Kent, United Kingdom

Abstract

Data mining consists of extracting interesting knowledge from data. This paper addresses the discovery of knowledge in the form of prediction IF-THEN rules, which are a popular form of knowledge representation in data mining. In this context, we propose a Genetic Algorithm (GA) designed specifically to discover interesting fuzzy prediction rules. The GA searches for prediction rules that are interesting in the sense of being new and surprising for the user. This is done adapting a technique little exploited in the literature, which is based on user-defined general impressions (subjective knowledge). More precisely, a prediction rule is considered interesting (or surprising) to the extent that it represents knowledge that not only was previously unknown by the user but also contradicts his original believes. In addition, the use of fuzzy logic helps to improve the comprehensibility of the rules discovered by the GA. This is due to the use of linguistic terms that are natural for the user. A prototype was implemented and applied to a real-world science & technology database, containing data about the scientific production of researchers. The GA implemented in this prototype was evaluated by comparing it with the J4.8 algorithm, a variant of the well-known C4.5 algorithm. Experiments were carried out to evaluate both the predictive accuracy and the degree of interestingness (or surprisingness) of the rules discovered by both algorithms. The predictive accuracy obtained by the proposed GA was similar to the one obtained by J4.8, but the former, in general, discovered rules with fewer conditions. In addition it works with natural linguistic terms, which leads to the discovery of more comprehensible knowledge. The rules discovered by the proposed GA and the best rules discovered by J4.8 were shown to a user (a University Director) in an interview who evaluated the degree of interestingness (surprisingness) of the rules to him. In general the user considered the rules discovered by the GA much more interesting than the rules discovered by J4.8.

Keywords**:** Data Mining, Classification, Genetic Algorithms, Rule Interestingness

1 Introduction

The basic idea of data mining consists of extracting knowledge from data [7], [13]. In this paper we address one general kind of data mining task, which we will refer to as the discovery of prediction rules. By prediction rule we mean an IF-THEN rule of the form:

IF <some_conditions_are_satisfied>
  THEN <predict_the_value_of_some_goal_attribute>.

We aim at discovering rules whose consequent (THEN part) predict the value of some goal attribute for an example (a record of a data set) that satisfies all the conditions in the antecedent (IF part) of the rule. We assume there is a small set of goal attributes whose value is to be predicted. The goal attributes are chosen by the user, according to his/her interest and need.

It can be noted that this task can be regarded as a generalization of the well-known classification task of data mining. In classification there is a single goal attribute to be predicted, whereas we allow more than one goal attribute to be defined by the user. Note that, although there are several goal attributes to be predicted, each rule predicts the value of a single goal attribute in its consequent. However, different rules can predict different values of different goal attributes.

Recent research has shown that evolutionary algorithms can be successfully used for data mining purposes [11], [5], [18], [2]. Based on this, we propose a Genetic Algorithm (GA) designed specifically for discovering interesting fuzzy prediction rules. The main motivation for using a GA in prediction-rule discovery is that GAs, due to their ability to perform

[1] Av. Colombo, 5790, Maringá PR 87020-900 E-mail address: wesley@din.uem.br
http://www.din.uem.br/wesley

[2] Canterbury, CT2 7NF, UK. E-mail address: a.a.freitas@kent.ac.uk
http://www.cs.kent.ac.uk/people/staff/aaf/

[3] Av. Colombo, 5790, Maringá PR 87020-900 E-mail address: itana@din.uem.br
http://www.din.uem.br/~itana

global search, tend to cope better with attribute interaction than most traditional greedy rule induction algorithms [11], [5], [27], [10].

The justification for the "interesting" and "fuzzy" characteristics of the rules discovered by our GA is as follows. In general, fuzzy logic is a flexible way of coping with uncertainties typically found in real-world applications. In particular, in the context of data mining, fuzzy logic seems a natural way of coping with continuous (real-valued) attributes. Using fuzzy linguistic terms, such as *low* or *high*, one can more naturally represent rule conditions involving continuous attributes, by comparison with crisp discretization of those attributes. For instance, the fuzzy condition "*Salary = low*" seems more natural for a user than the crisp condition "Salary < *$14,328.53*".

Although we do use fuzzy logic to improve the comprehensibility of the rules discovered by the GA, the focus of this paper is not on the use of fuzzy logic, but rather on the discovery of "interesting" rules. We emphasize that this is a difficult problem, relatively little explored in the literature. Most algorithms for discovering prediction rules focus on evaluating the predictive accuracy of the discovered rules [14], without trying to discover rules that are truly interesting for the user. A classic example is the decision tree algorithm J4.8 [36], whose results are compared to the proposed GA.

It should be noted that a rule could have a high predictive accuracy but may not be interesting to the user, because it represents some obvious or previously known piece of knowledge. A classic example is the rule: IF <patient is pregnant> THEN <patient is female>.

Hence, this paper proposes a GA that searches for rules that are not only accurate but also interesting, in the sense of being surprising (representing novel knowledge) for the user. As will be seen later, the core of the GA consists of an elaborate fitness function, which takes both these aspects of rule quality into account. The GA evaluates the rules by performing a fuzzy matching between the rules and the records in the data set being mined. It also takes into account the user's general impressions about the application domain, in order to favor the discovery of rules that are more surprising to the user. The basic idea is that, the more the rule contradicts the general impressions of the user, the more surprising it is.

We apply the proposed GA (and the above-mentioned J4.8 algorithm) to the mining of a real-world science & technology data set, containing data about the scientific production of researchers (cientometric data).

The proposed GA was recently proposed by [31]. This paper is an extended version of that previous work. In [31] we compare the results of the GA with the results of a well-known decision tree algorithm, J4.8, with respect to predictive accuracy. In this paper we extend these results, by comparing the degree of interestingness of the rules discovered by the GA with the degree of interestingness of the rules discovered by J4.8. This required the development of a procedure to extract the most interesting rules from J4.8, since J4.8 usually discovers (in the data sets used in our experiments) a very large number of rules, most of which are uninteresting.

The remainder of this paper is organized as follows. Section 2 reviews relevant related work. Section 3 describes in detail the proposed GA for discovering interesting (surprising) fuzzy prediction rules. Section 4 reports the results of computational experiments. Finally, section 5 present conclusions and future work.

2 Related Work

2.1 Evolutionary Algorithms for Discovering Fuzzy Prediction Rules

There has been very extensive research on evolutionary algorithms (EAs) for discovering fuzzy prediction rules. Roughly speaking, the algorithms can be divided into three broad groups [11]:

a) EAs for generating fuzzy rules, with fixed membership functions – In this approach an EA is used to search for good combinations of attribute values that will compose fuzzy rules. However, the membership functions of the attribute values are predefined (either manually or by another algorithm), rather than being evolved by the EA [17], [34].

b) EAs for tuning the membership functions associated with the attributes being fuzzified, with fixed rules – This approach is typically used when crisp rules have already been discovered by another algorithm, and we just want to use an EA to fuzzify the discovered crisp rules [3].

c) EAs for both generating fuzzy rules and tuning membership functions – In this approach an EA is used to optimize both the contents of fuzzy rules (i.e., the combination of attribute values occurring in the rules) and the membership functions of the linguistic values of the attributes being fuzzified. [24], [37], [23].

We follow the first approach (a) with user-defined membership functions, due mainly to the fact that it allows us to incorporate the domain knowledge of the user into the specification of the membership functions. This leads to more comprehensible membership functions for the user. This is important in our data mining application, where a human decision maker directly interpreted the discovered prediction rules. In addition, compared to the third approach (c), the first one has the advantage of reducing the search space. Thus, it is more computationally efficient, since the GA has to search only for combinations of attribute values to be included in the rules.

It should be noted that the above-mentioned projects focus on the discovery of fuzzy rules with high predictive accuracy, without trying to discover surprising rules. Our work differs from these projects in that the proposed GA searches for fuzzy prediction rules that are not only accurate but also surprising for the user, representing knowledge that was previously unknown by the user, as will be seen later.

2.2 EAs for Discovering Interesting Prediction Rules

The discovery of accurate knowledge from data has long been the goal of traditional methods of data analysis, based on statistics and/or machine learning. Arguably, the discovery of knowledge that is not only accurate but also truly interesting and surprising to the user is one of the most important goals in data mining. A focus on this goal is probably the characteristic that most distinguishes data mining from conventional statistics and/or machine learning [11].

There are two broad approaches for discovering interesting rules in data mining: objective and subjective. In general, the objective approach uses a rule-discovery method and a rule-quality measure that are independent of the user and the application domain [15], [25], [26], [8], [22].

By contrast, the subjective approach uses a rule-discovery method and/or a rule-quality measure that take into account the background knowledge of the user about the application domain [33], [19], [20], [21].

Hence, in general the objective approach has more generality and autonomy than the subjective approach, whereas the subjective approach has the important advantage of using the user's background knowledge to guide the search for rules. Therefore, if the application domain is well-defined and an expert user in this application domain is available, it makes sense to use the subjective approach. This is the case of the project reported in this paper, which justifies our choice of the subjective approach.

We now discuss some works that are more related to our research. The first one is the work of [19], [20]. This work also follows the subjective approach. It proposes the use of general impressions to guide the search for interesting rules. General impressions can be thought of as "rules" specified by the user, representing his background knowledge and believes about the application domain. General impressions will be discussed in more detail later. Liu and his colleagues propose the use of general impressions as the basis for a post-processing method to select the most interesting rules, among all discovered rules. That is, first one data mining algorithm is run, discovering a potentially large number of rules. Then the discovered rules are matched against the user-specified general impressions, in order to select the most interesting rules.

Our work also uses the idea of user-specified general impressions to discover interesting rules. However, it differs from the above work in that we use general impressions directly to search rules, rather than as a post-processing method. In other words, instead of first generating a large number of rules and then selecting the most interesting ones, the set of general impressions is directly used by the data mining algorithm to generate only interesting rules. This avoids the unnecessary generation of many rules that will be later discarded due to their lack of interestingness for the user. In addition, we propose a GA for discovering interesting rules, whereas the work of Liu and his colleagues does not use any evolutionary algorithm.

The GA for discovering interesting rules proposed by [25], [26] is another work related to our research. This GA also searches for rules that are both accurate and interesting, according to a certain rule-interestingness measure. However, our work differs from their work in two major points. First our GA discovers fuzzy rules. Second, Noda et al. follow an objective approach for the discovery of interesting rules, whereas our GA follows a subjective approach based on user-specified general impressions, as mentioned above.

Another relevant work involves the use of a GA to evolve a measure of rule interestingness, as proposed by [35]. In this work an interactive GA maintains several independent rule sets (subpopulations of rules). Rule sets are evaluated by different measures of rule interestingness. From time to time, during population evolution, a small subset of rules is chosen from each population and shown to the user, for his/her subjective evaluation – this kind of subjective evaluation characterizes an interactive GA. Apparently, the result of this evaluation is used for evolving rule interestingness measures. However, no details about this process are given in the paper. In addition, in the previously-mentioned reference the author did not present any computational result evaluating the performance of the system. The basic idea of interactive evolution has been more recently used by [28], but this work is concerned with the discovery of association rules, rather than prediction rules. It should be noted that association and classification rules are very different from each other, as explained in [9].

3 A GA for Discovering Interesting Fuzzy Prediction Rules

This section describes a GA that evolves a population of individuals, where each individual represents a prediction rule. More precisely, each individual represents the antecedent (IF part) of a prediction rule. The consequent (THEN part) of the rule is not encoded in the genome. Rather, it is fixed for a given GA run, so that in each run all the individuals represent rules with the same consequent (value predicted for a goal attribute). Therefore, in order to discover rules predicting different goal attribute values, we need to run the GA several times, once for each value of each goal attribute.

Furthermore, the prediction rules represented by the individuals are fuzzy. We stress that only the rule antecedents are fuzzified. Rule consequents are always crisp. Concerning the rule antecedent, only conditions involving continuous (real-valued) attributes are fuzzified. Categorical (nominal) attributes are inherently crisp. For instance, there is no need to fuzzify a rule condition such as "*Sex = female*".

This section is based on the description recently presented in [31]. In any case, the description of the GA is also included here, in order to make this paper self-contained.

3.1 Individual Representation

The genome of an individual represents a conjunction of conditions specifying a rule antecedent. Each condition is represented by a gene, which consists of an attribute-value pair of the form $A_i = V_{ij}$, where A_i is the i-th attribute and V_{ij} is the j-th value belonging to the domain of A_i. In order to simplify the encoding of conditions in the genome, we use a positional encoding, where the i-th condition is encoded in the i-th gene. Therefore, we need to represent only the value V_{ij} of the i-th condition in the genome, since the attribute of the i-th condition is implicitly determined by the position of the gene. In addition, each gene also contains a boolean flag (B_i) that indicates whether or not the i-th condition is present in the rule antecedent. Hence, although all individuals have the same genome length, different individuals represent rules (corresponding to the "phenotype") of different lengths. This flexible representation is, of course, desirable in prediction rules. As one does not know, a priori, how many conditions will be necessary to create a good prediction rule, this number has to be automatically adjusted by the GA based on the data being mined. The structure of the genome of an individual is illustrated in Figure 1, where m is the number of attributes of the data being mined.

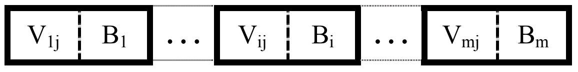

Figure 1: Genome of an individual representing a rule antecedent

We emphasize that the operator "=" is used for both fuzzy conditions and crisp conditions as follows. As usual in data mining and machine learning literature, our GA can cope with two kinds of attributes: continuous (real-valued) and categorical (nominal). Categorical attributes are inherently crisp, so they are associated with crisp conditions such as "*Sex = female*". Continuous attributes are fuzzified, so they are associated with fuzzy conditions such as "*Age = low*", where *low* is a fuzzy linguistic term.

3.2 Fuzzifying Continuous Attributes

Recall that, as discussed earlier, in our GA we use user-defined membership functions. Hence, it evolves the combinations of attribute values considered relevant for predicting a goal attribute, but there is no need to evolve the membership functions.

In our GA the fuzzification of continuous attributes is performed as follows. Each continuous attribute is associated with either two or three linguistic terms (corresponding to the "values" of the fuzzified attribute), namely either {*low*, *high*} or {*low*, *medium*, *high*}. Each of these linguistic terms is defined by a user-specified membership function. The number of linguistic terms was also specified with the help of the user.

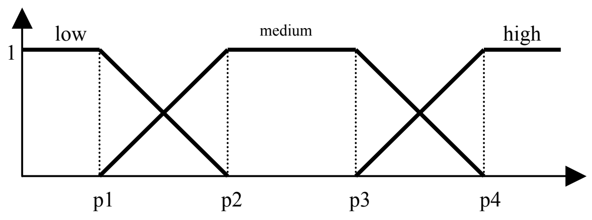

Figure 2: Shape of the membership functions used in the GA

We have used trapezoidal membership functions, as shown in Figure 2. The figure illustrates the case where an attribute was fuzzified into three linguistic values. Note that the membership functions of all these linguistic values are collectively specified by only four parameters, denoted p_1, p_2, p_3 and p_4 in the figure, where $p_1 < p_2 < p_3 < p_4$. Each parameter represents an original (continuous) attribute value that is used to specify the coordinate of two trapezoid vertices belonging to a pair of "adjacent" membership functions. This representation enforces several desirable constraints in the fuzzy sets. First, any element of the universe of discourse (i.e., any original value of the attribute being fuzzified) belongs to at least one of the fuzzy sets. Second, each fuzzy set is unimodal and normal – the normality condition means that the largest membership degree obtained by any element in the set is 1. Third, it contains a small number of linguistic values (just 3). In general these constraints help to ensure that the fuzzy sets are "semantically sound", or have a good "linguistic interpretability" [16], [6], [32].

3.3 Fitness Function

Recall that each individual is associated with a fuzzy prediction rule. In the vast majority of the literature, the main criterion used to evaluate the quality of a fuzzy prediction rule is predictive accuracy [14]. This criterion is also important in our application, but it is not the only one. As discussed in the Introduction, a prediction rule can be accurate but not interesting for the user. This will be the case when the rule represents some relationship in the data that was already known by the user. To avoid this, our fitness function takes into account two criteria:

(a) the predictive accuracy of the rule (Acc);

(b) a measure of the degree of interestingness (or surprisingness) of the rule (Surp).

With respect to the latter criterion, our GA favors the discovery of rules that are explicitly surprising for the user, as will be seen later.

These two criteria are combined into a single formula as follows:

$$\text{Fitness}(i) = \text{Acc}(i) * \text{Surp}(i)$$

The measures of Acc(i) and Surp(i) are described in the next two subsections, respectively, since they are computed by separated elaborate procedures.

3.3.1 Measuring the Predictive Accuracy of a Fuzzy Rule

The first step to measure the predictive accuracy of a fuzzy rule is to compute the degree to which an example belongs to a rule antecedent. Recall that the rule antecedent consists of a conjunction of conditions. We use the standard fuzzy AND operator, where the degree of membership of an example to a rule antecedent is given by:

$$\min_{i=1}^{z}(\mu_i)$$

where μ_i denotes the degree to which the example belongs to the i-th condition of the rule antecedent, z is the number of conditions in the rule antecedent, and min is the minimum operator. The degree to which the example belongs to the i-th condition is directly determined by the value of the corresponding membership function for the example's attribute value associated with that condition. Of course, crisp conditions can have only either 0 or 1 membership degree.

For instance, consider a rule antecedent with the following two rule conditions: (*Age* = *low*) AND (*Sex* = *female*), where the first condition is fuzzy and the second one is crisp. Suppose that a given example has the values *23* and *female* for the attributes *Age* and *Sex*, respectively. Suppose also that the membership function for the *low* linguistic term of *Age* returns the value 0.8 for the value *23*. Then the degree to which this example belongs to this rule antecedent is min(0.8,1.0) = 0.8.

Let A be the antecedent of a given rule. Once the degree to which each example belongs to A has been computed, the predictive accuracy of the i-th individual (fuzzy rule), denoted Acc(i), is computed by the formula:

$$\text{Acc}(i) = (\text{CorrPred} - 1/2) / (\text{TotPred})$$

In this formula, CorrPred (number of correct predictions) is the summation of the degrees of membership in A for all examples that have the value V_{ij} predicted by the rule. TotPred (total number of predictions) is the summation of the degrees of membership in A for all examples. It is essentially a fuzzy version of a crisp measure of predictive accuracy used by some data mining algorithms [29], [25]. The rationale for subtracting 1/2 from CorrPred in the numerator is to penalize rules that are too specific, which are probably overfitted to the data. For instance, suppose CorrPred = 1 and TotPred = 1. Without subtracting 1/2 from CorrPred the modified formula would return a predictive accuracy of 100% for the rule, which intuitively is an over-optimistic estimate of predictive accuracy in this case. However, subtracting 1/2 from CorrPred the above formula returns 50%, which seems a more plausible estimate of predictive accuracy, given that the rule is too specific. Clearly, for large values of CorrPred and TotPred the subtraction of 1/2 will not have a significant influence in the value returned by the formula, so that this subtraction penalizes only rules that are very specific, covering just a few examples.

3.3.2 Measuring the Degree of Surprisingness of a Prediction Rule

We consider a prediction rule interesting to the extent that it is surprising for the user, in the sense of representing knowledge that not only was previously unknown but also contradicts the original believes of the user. Clearly, the problem of discovering surprising rules is a very difficult one, which has been relatively little investigated in the data mining literature. (As mentioned above, the vast majority of the literature focus on the discovery of rules with a high predictive accuracy, without trying to measure how novel or surprising the rule is for the user.)

We follow a subjective approach for discovering surprising rules, based on the use of user-specified general impressions [19], [20]. We emphasize that, in the method proposed by Liu et al., surprising rules are extracted from the entire set of discovered rules, in a post-processing step. That is, first the system discovers a (potentially very large) set of rules, and then a relatively small set of surprising rules is extracted from that entire set of discovered rules. By contrast, our approach is to use a GA to directly search for surprising rules during the evolution of rules, avoiding the need for a post-processing step. That is, the GA directly takes the degree of surprisingness of rules into account, so that it will output only rules that are both accurate and surprising to the user, as will be seen later.

In essence, a general impression specifies some relationship that the user believes to be true in the data being mined. General impressions, like prediction rules, are expressed in the form IF <conditions> THEN <predicted value>. The main difference is that general impressions are manually specified and represent believes of the user about relationships in the data, whereas prediction rules are automatically discovered and represent relationships that seem to hold in the data, according to the criteria used by the data mining algorithm. Therefore, the specification of general impressions assume that the user already has some previous knowledge or hypotheses about relationships that hold in the application domain - in our case, science and technology data.

Let $\{R_1,...,R_i,...R_{|R|}\}$ be the set of rules in the current population of the GA, where |R| denotes the number of rules (individuals); and let $\{GI_1,...,GI_j,...GI_{|GI|}\}$ be the set of general impressions representing the previous knowledge and believes of the user, where |GI| denotes the number of general impressions. Note that the set $\{GI_1,...,GI_j,...GI_{|GI|}\}$ is specified by the user before the GA starts to run, and it is kept fixed throughout the GA run. In order to compute the degree of surprisingness of the rules in current population, each rule is matched against every GI, as shown in Figure 3.

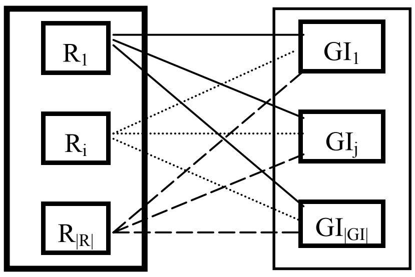

Figure 3: Matching between each rule and every general impression

A rule R_i is considered surprising, in the sense of contradicting a general impression GI_j of the user, to the extent that R_i and GI_j have similar antecedents and contradictory consequents. In other words, the larger the similarity of the antecedents of R_i and GI_j and the larger the degree of contradiction of the consequents of R_i and GI_j, the larger the degree of surprisingness of rule R_i with respect to general impression GI_j.

For each pair of rule R_i and GI_j - where i varies in the range 1,...,|R| and j varies in the range 1,...,|GI| - the GA computes the degree of surprisingness of R_i with respect to GI_j in three steps, as follows.

First step: finding the general impressions whose consequents contradicted the consequent of R_i. We say that the consequent of R_i contradicts the consequent of a general impression GI_j if and only if R_i and GI_j have the same goal attribute but a different goal attribute value in their consequent. For instance, this would be the case if R_i predicts "*production = low*" and GI_j predicts "*production = high*". Note that if R_i and GI_i predict different goal attributes, or if they predict the same value for the same goal attribute, there is no contradiction between them. In this case the degree of surprisingness of R_i with respect to GI_i is considered zero, and the second and third steps, described below, are ignored.

Second step: computing the similarity between the antecedents of R_i and GI_j. For each general impression GI_j found in the previous step (i.e, each general impression GI_j contradicted by R_i), the system computes the similarity between the antecedents of R_i and GI_j. This similarity, denoted $AS_{(i,j)}$, is computed by the formula:

$$AS_{(i,j)} = |A_{(i,j)}| / \max(|R_i|,|GI_j|) ,$$

where $|R_i|$ is the number of conditions (attribute-value pairs) in rule R_i, $|GI_j|$ is the number of conditions in general impression GI_j, max is the maximum operator, and $|A_{(i,j)}|$ is the number of conditions that are exactly the same (i.e., have the same attribute and the same attribute value) in both R_i and GI_j. This formula is a simplified version of the formulas proposed by [19] to measure the similarity between the antecedents of R_i and GI_j. Those authors proposed separate formulas to measure the similarity with respect to attributes and with respect to attribute values. We have chosen to incorporate both aspects of antecedent similarity into a single formula, for the sake of simplicity.

Third step: computing the degree of surprisingness of R_i with respect to GI_j. Let Surp(i,j) denote the degree of surprisingness of R_i with respect to GI_j. Surp(i,j) depends on both $AS_{(i,j)}$, computed in the second step, and on the difference between the rule consequents of R_i and GI_j, computed in the first step, as follows. The goal attribute values in the consequents of R_i and GI_j can be either a value in the set {*low*, *high*} or {*low*, *medium*, *high*}, depending on the goal attribute. (The choice between these two attributes domains is made by the user for each goal attribute, as will be seen later.) If the difference between the consequents of R_i and GI_j is that one of them is *low* and the other is *high*, characterizing the greatest possible difference between those consequents, then Surp(i,j) is assigned the value of $AS_{(i,j)}$, without any modification. If the difference between the consequents of R_i and GI_j is that one of them is *medium* and the other is either *low* or *high*, characterizing a smaller difference between those consequents, then Surp(i,j) is assigned half the value of $AS_{(i,j)}$, i.e. Surp(i,j) = 0.5 x $AS_{(i,j)}$. In the latter case Surp(i,j) is assigned a smaller value than in the former case to reflect the fact that the degree of contradiction is correspondingly smaller.

Finally, once the above three steps have been completed for all general impressions, with respect to a given rule R_i, the system has computed all the degrees of surprisingness of R_i with respect to every general impression GI_j, i.e. all Surp(i,j), j=1,...,|GI|, where |GI| is the number of general impressions. At this point the degree of surprisingness of rule R_i, denoted Surp(i), is simply computed by the formula:

$$Surp(i) = \max_{j=1}^{|GI|}\left[AS_{(i,j)}\right]$$

where max returns the maximum value among its arguments.

3.4 Selection and Genetic Operators

The GA uses tournament selection [1], which essentially works as follows. First, *k* individuals are randomly picked (k = 2 in our experiments), with replacement, from the population. Then the individual with the best fitness value, out of the *k* individuals, is selected as the winner of the tournament. This process is repeated *P* times, where *P* is the population size. Next the *P* selected individuals undergo genetic operators described below.

The GA uses relatively simple crossover and mutation operators. It uses uniform crossover [12]. There is a probability for applying crossover to a pair of individuals and another probability for swapping each corresponding pair of gene (attribute)'s value in the genome of two individuals. The crossover probabilities used were 0.85 for the crossover operator and 0.5 for attribute value swapping. Our choice of uniform crossover was motivated by the fact that this operator has no positional bias. The probability of swapping each pair of attribute values is independent of the position of that attribute value in the genome. This is desirable in our data mining application, where the rule antecedent represented by the genome consists of an unordered set of conditions. The mutation operator randomly transforms the value of an attribute into another (different) value belonging to the domain of that attribute.

In addition to crossover and mutation operators, the GA also uses operators that insert/remove conditions to/from a rule. In essence, the condition-insertion operator switches on the flag of some condition in the genome, rendering it present in the decoded rule antecedent. Conversely, the condition-removal operator switches off the flag of some condition in the genome, which effectively removes that condition from the decoded rule antecedent. The condition-insertion and condition-removal operators perform specialization and generalization operations in the rule, respectively. Hence, they contribute for a broader exploration of the search space, facilitating the exploration of some regions of the search space that might not be so easily accessible to crossover and mutation operators.

3.5 Summary of the Algorithm

In order to summarize the earlier description of the system, the pseudocode of the system is shown, in a very high level of abstraction, in Algorithm 1. Each iteration of the main loop of the algorithm – the FOR EACH loop – discovers the best rule (taking into account accuracy and surprisingness) predicting a given rule consequent – i.e., a given pair <goal attribute, value>. The algorithm returns one rule for each rule consequent to be predicted. As can be seen at the end of the main loop of the algorithm, the rule returned to the user has to satisfy two conditions, namely: (a) Surp > 0; and (b) Acc > max(0.5, RelativeFreq). The first condition simply requires that the degree of surprisingness of the rule be greater than zero, so that the rule is considered at least potentially surprising to the user. The motivation for the second condition can be understood by considering two cases.

Algorithm 1: Pseudocode of the system

```
Specify the membership functions of the attributes being fuzzified, with the help of the user;
Obtain the general impressions (GI's) from the user;
FOR EACH rule consequent (pair <goal attribute, value>):
    Compute the Relative Frequency of this goal attribute value in the training set;
    Generate the initial population at random;
    Call procedure Compute-Fitness;
    FOR g = 1 to Number_of_Generations
        Generate a new population, performing the following operations:
            Selection
            Crossover / Mutation / Condition Insertion / Condition Removal
        Call procedure Compute-Fitness;
    END FOR g
    Return to the user the rule with largest fitness, subject to the conditions:
       (Surp > 0) AND (Acc > max(0.5, RelativeFreq))
END FOR EACH rule consequent

PROCEDURE Compute_Fitness
    Compute the accuracy (Acc) of each rule (individual), by performing
         a fuzzy matching between the rule and the training examples;
    Compute the surprisingness (Surp) of each rule, by matching the rule with the general impressions;
    Compute the Fitness of each rule;
END PROCEDURE Compute-Fitness.
```

First, suppose that the relative frequency (on the training set) of the goal attribute value predicted by this rule is greater than 0.5. It makes sense to require that a rule has a classification accuracy greater than that relative frequency. After all, the latter can be considered a natural baseline accuracy for any rule predicting that goal attribute value, in the sense that a trivial rule with an empty antecedent (i.e., no condition) would have that baseline accuracy. Consider now the case where the relative frequency of the goal attribute value predicted by the rule is smaller than 0.5. It still makes sense to require that Acc > RelativeFreq, but now this condition is relatively easy to be satisfied. For instance, if RelativeFreq = 0.2, and the rule has Acc = 0.3, the condition would be satisfied, but the rule is still a bad rule, having a very low Acc. Hence, we also require that the rule has an Acc > 0.5 to enforce the constraint that a discovered rule have at least a reasonable accuracy. The condition Acc > max(0.5, RelativeFreq) implements the constraints of both cases in a simple, concise formula.

4 Computational Results

In this section we report the results of computational experiments carried out with our proposed GA described in the previous section. The results of our GA and J4.8 [36] are compared. J4.8 is a decision tree algorithm implemented in a public domain tool called Weka:
http://www.cs.waikato.ac.nz/ml/weka/index.html.

The experimental set of general impressions was specified by the Maringá State University Research Director (Brazil) to evaluate the algorithm. The same user also evaluated the interestingness of the rules discovered by our GA and J4.8, as will be seen later. The data set used in our experiments is described in section 4.1.

The rules discovered by the GA and J4.8 were evaluated with respect to two criteria, namely:

(a) Predictive accuracy. As usual in the literature, predictive accuracy was measured in an objective way, by computing the prediction accuracy rate on a test set separate from the training set. The results with respect to predictive accuracy are reported in section 4.2.

(b) *Degree of interestingness (surprisingness)*. This is a measure of how surprising (novel) the rule is for the user, as explained in the previous section. This was measured in a subjective way, by showing the discovered rules to the user and asking him to assess them according to how interesting they were. The results with respect to interestingness are reported in section 4.3.

In the experiments reported in sections 4.2 and 4.3, we used the default parameters of J4.8. To make the comparison with J4.8 as fair as possible, we also used the default parameters of our GA, without any attempt to optimize parameters. The parameters of the GA are as follows:

- Crossover probability = 85%
- Mutation probability = 2%
- Condition-insertion probability = 2%
- Condition-removal probability = 1%
- Tournament size = 2
- Maximum number of conditions in a decoded rule antecedent (i.e., maximum number of active conditions in the genome of an individual) = 5
- Population size = 100 individuals
- Number of generations = 100

4.1 The Data Set

The application domain addressed in this paper involves a science and technology database obtained from CNPq (the Brazilian government's National Council of Scientific and Technological Development). We have mined a subset of the database containing data about the scientific production of researchers in the South region of Brazil. However, it should be noted that the design of our GA is generic enough to allow its use in virtually any other application domain, as long as proper general impressions and membership functions as specified by the user.

The experiments reported in this paper have been performed with 24 attributes. The selection and preparation of these attributes for data mining purposes were time-consuming processes. They took several months, as the original data set was not collected for data mining purposes.

The prepared data set contained 5,690 records (examples). Each record had attributes describing a given researcher and his scientific production in the period from 1997 to 1999. Records that had any attribute with missing value were removed. Out of the 24 attributes, 6 were used as goal attributes to be predicted, and the other 18 attributes were used as predictor attributes. Out of the 18 predictor attributes, 8 were categorical (nationality, continent of origin, sex, state, city, skill in writing English, whether or not she/he was the head of a research group, main research area) and 10 were continuous (educational level, No. of years since last graduation, age, No. of completed technical projects, No. of delivered courses, No. of supervised Ph.D. thesis, No. of supervised M.Sc. dissertations, No. of supervised research essays (at the diploma level), No. of supervised final-year undergraduate projects, No. of supervised undergraduate students with a research scholarship). The 10 continuous attributes were fuzzified for rule-discovery purposes, as previously explained.

For prediction purposes, each goal attribute was discretized into either two values (referring to a low or high scientific production) or three values (referring to a low, medium or high scientific production), as determined by the user.

The 6 goal attributes, denoted $G_1,...,G_6$, have the following meaning and values to be predicted:

G_1 = No. of papers published in national journals - values: low, medium, high;
G_2 = No. of papers published in internat. journals - values: low, medium, high;
G_3 = No. of chapters published in national books - values: low, medium, high;
G_4 = No. of chapters published in international books - values: low, high;
G_5 = No. of national edited/published books - values: low, high;
G_6 = No. of internat. edited/published books - values: low, high.

Therefore, in total there are 15 goal attribute values to be predicted.

4.2 Evaluating the Predictive Accuracy of the Discovered Rules

In order to measure the predictive accuracy of discovered rules, we have performed a well-known 10-fold cross-validation procedure [14], which works as follows. First, the data set is divided into 10 mutually exclusive and exhaustive partitions. Then the data mining algorithm is run 10 times. In the i-th run, i=1…10, the i-th partition is used as the test set, and the remaining 9 partitions are temporarily grouped and used as the training set. In each run the system computes the prediction accuracy rate on the test set, which is the ratio of the number of correct predictions over the total number of predictions. The reported result is the average prediction accuracy rate over the 10 runs.

We have compared the predictive accuracy of the rules discovered by our GA with the accuracy of the rules discovered by J4.8. Note that J4.8 is an algorithm designed for the classification task of data mining, where there is a single goal attribute to be predicted. Similarly, each run of our GA discovers a rule predicting a different goal attribute value. Hence, both J4.8 and our GA have to be run several times in our application, since we are interested in discovering rules predicting several goal attributes. More precisely, J4.8 was run 6 times, whereas our GA was run 15 times, corresponding to the 15 different goal attribute values for all the 6 goal attributes. For both algorithms, each run consisted of the 10 iterations of the 10-fold cross-validation procedure.

Note also that J4.8 and our GA were designed for discovering different kinds of prediction rules. There are two main differences. The first, J4.8 just tries to discover accurate rules. It does not try to discover interesting, surprising rules. In contrast, our GA tries to discover rules that are both accurate and surprising for the user. Second, J4.8 was designed for discovering classification rules covering all examples, which may be called "complete prediction". Given any test example, J4.8 must discover a rule that can be used to predict its class whereas our GA does not try to discover rules covering all examples. It tries to discover only a small set of interesting, surprising rules, the knowledge "nuggets". The discovered rules can collectively cover only a relatively small subset of examples, and yet be considered surprising and high-quality rules. This approach may be called "partial prediction". These two differences make it difficult to compare the two algorithms in a fair way.

In order to make this comparison fairer, we have eliminated the first difference mentioned above. This was achieved by modifying the fitness function of the GA (only in the experiments reported in this section) so that the fitness of an individual (rule) is measured only by its predictive accuracy, ignoring its degree of surprisingness, i.e., the fitness of the i-th individual is given by (see subsection 3.3.1):

$$\text{Fitness}(i) = \text{Acc}(i) = (\text{CorrPred} - 1/2) / (\text{TotPred})$$

Having done that both J4.8 and the GA search only for accurate rules.

The second difference between the two algorithms, as mentioned above, is more difficult to eliminate, and it still remains a difference in our experiments. The only way to eliminate this difference would be either to modify the GA to perform complete prediction or to compare the GA with another rule induction algorithm that performs partial predictions. These possibilities will be considered in future research.

The predictive accuracy obtained by J4.8 and our GA is reported in Table 1.

Table 1: Prediction Accuracy Rate (%) of J4.8 and the GA

Goal attrib.	**Predicted value**	**Freq. (%)**	**J4.8 (%)**	**GA (%)**
G_1	low	46.9	**64.9**	58.8 ±14.0
	medium	50.6	**63.9**	60.4 ±13.3
	high	2.5	**9.1**	00.0 ±0.0 (-)
G_2	low	64.2	76.6	**90.7 ±5.4 (+)**
	medium	29.7	**45.3**	40.0 ±16.3
	high	6.1	**32.2**	25.0 ±13.4
G_3	low	76.9	82.2	**95.2 ±3.0 (+)**
	medium	21.2	45.3	**56.7 ±15.8**
	high	1.9	**27.4**	25.0 ±13.4
G_4	low	93.2	93.4	**98.4 ±0.7 (+)**
	high	6.8	**51.7**	14.3 ±10.4 (-)
G_5	low	83.5	86.0	**89.5 ±9.9**
	high	16.5	54.7	**56.9 ±14.0**
G_6	low	97.9	97.9	**98.9 ±0.5 (+)**
	high	2.1	0.0	00.0 ±0.0

The first column of this table identifies the goal attribute predicted by the rule (see the meaning of G_1...G_6 in the previous section), whereas the second column identifies the value predicted for that goal attribute. The third column identifies the relative frequency (in %) of the corresponding goal attribute value in the training set. The fourth and fifth columns report the prediction accuracy rate (in %) in the test set (10-fold cross-validation) of J4.8 and our GA, respectively. In the last column the results after the "±" symbol denote standard deviations. (Standard deviation values were not available for the results of J4.8.) In each row, we show in bold the larger predictive accuracy rate, out of the accuracy rates obtained by the two algorithms.

As can be seen in the table, the predictive accuracy rate of the GA is larger than the one of J4.8 in seven rows (i.e., seven goal attribute values), whereas the converse is true in other seven rows. The cases where the accuracy of the GA is significantly larger or smaller than the accuracy of J4.8 are indicated by the symbol "(+)" or "(-)", respectively, in the last column. By significantly larger or smaller we mean that the difference between the accuracy of the GA and J4.8 is larger than twice the value of the standard deviation reported in the last column.

With the exception of the goal attribute G_1, the GA outperformed J4.8 in the prediction of goal attribute values with a larger relative frequency in the training set, whereas J4.8 outperformed the GA in goal attribute values with a smaller relative frequency in the training set. This might have been due to the following difference in the search for rules performed by the two algorithms. The GA discovers rules for each class in turn. Hence, both classes with a large frequency and classes with a small frequency are equally considered by the algorithm. Thus, it does not bias the search towards any of these kinds of classes. In contrast, J4.8 discovers rules for all classes (values of a given goal attribute) in each run. As it is easier to discover rules for classes with larger frequency, J4.8 is biased towards discovering rules that predict the most common class(es). That is, it tends to assigns less importance to rare classes.

We now turn to the problem of comparing the degree of interestingness of the rules discovered by the GA and by J4.8, discussed in the next section.

4.3 Evaluating the Interestingness of the Discovered rules

The rules discovered by the GA and J4.8 were also evaluated with respect to their degree of interestingness (surprisingness) for the user. The user subjectively evaluated this degree of interestingness. The comparison of the rules discovered by the two algorithms with respect to interestingness was also challenging, due mainly to the fact that, for the majority of the 6 goal attributes, J4.8 produced a very large decision tree, with literally hundreds of nodes. One exception was the tree for the goal attribute G_6, where the opposite happened: J4.8 produced a degenerated tree, with a single node, which is equivalent to a rule with an empty antecedent (i.e., no rule conditions, so that the rule covers all examples) and a consequent predicting the majority class. Obviously, it was not possible to show all the thousands of rules (considering all goal attributes) to the user. In any case, the vast majority of the rules discovered by J4.8 tend to be uninteresting, since J4.8 tries to maximize predictive accuracy, rather than interestingness, as mentioned in the previous section.

Hence, in order to make a fair comparison of the rules discovered by the GA and J4.8, with respect to their degree of interestingness, we have selected a small subset of the rules discovered by J4.8, and only the rules in that subset were compared with the rules discovered by the GA. The procedure to select the subset of most interesting rules discovered by J4.8 is based on the idea of measuring the interestingness of J4.8's rules in a way which is as similar as possible to the way that interestingness is measured in the fitness function of the GA. This makes the comparison of the results of the two algorithms as fair as possible. This procedure selects exactly one rule, considered the most interesting rule, for each goal attribute – again, this makes the comparison with the GA as fair as possible, since the GA also selects exactly one rule for each goal attribute value. The procedure for selecting the most interesting J4.8 rules is shown, at a high level of abstraction, in Algorithm 2.

As mentioned earlier, the GA found rules for 15 goal attribute values, but J4.8 found rules for 13 goal attribute values only (since it found no rule for goal attribute G_6). For each of those 13 goal attribute values, we selected one rule discovered by J4.8, by running the above procedure. For only 8 of those goal attribute values there was a rule having a degree of interestingness (Surp) greater than 0. In the case of the other 5 goal attribute values, there was no rule with Surp greater than 0, so that the corresponding 5 rules were selected based on their predictive accuracy only, as indicated in the last IF-THEN-ELSE instruction of Algorithm 2. This is by itself a significant evidence that, as expected, J4.8 tends to discover uninteresting rules, since it was not designed to explicitly discover interesting rules.

The user was asked to assess each rule discovered by the GA and by J4.8, according to how interesting/surprising the rule was for him, and then assign each rule to one of the following three degrees of interestingness (surprisingness): low interestingness, medium interestingness or high interestingness. The results of the evaluation performed by the user are reported in Table 2.

The results reported in Table 2 were obtained by using the entire data set (i.e., all the 5,690 examples) as input data for the GA. This procedure is justified because when measuring the degree of interestingness of discovered rules there is no need for dividing the data into training and test sets, since there is no need for measuring predictive accuracy in the test set (which was already measured in the experiments reported in the previous section).

Algorithm 2: Selecting interesting rules from the decision tree produced by J4.8

```
Convert the decision tree into a set of rules in the usual way – i.e., each path from the root node to a leaf node corresponds
    to a rule;

FOR EACH goal attribute value v;
  Initialize a set of rules, called S, with the empty set;
  Select all the rules discovered by J4.8 that are predicting value v and that have at most 5 conditions (attribute- value
  pairs) in their antecedent, and put these rules into S;  /* the limit of 5 conditions is also used in the GA */
  FOR EACH rule in S;
      Compute its predictive accuracy (Acc) by using the same formula used to compute predictive accuracy in the
      fitness function of the GA;   /* see section 3.3.1 */
      IF the rule's Acc satisfies the condition Acc > max(0.5, RelativeFreq)
         THEN keep the rule in S,
         ELSE remove the rule from S;
      Compute the degree of interestingness or surprisingness (Surp) of the rule, by matching it with the user-defined
      general impressions, and by using the same formula used to compute interestingness in the fitness function of the
      GA;   /* see section 3.3.2 */
      Compute the fitness of the rule, using the same formula used to compute the fitness function in the GA, i.e.,
      Fitness = Acc * Surp;      /* see section 3.3 */
  END FOR EACH rule in S
  IF there is at least one rule in S with Surp > 0
     THEN select, out of all rules in S, the rule with the largest fitness, and return this rule to the user,
     ELSE select, out of all rules in S, the rule with the largest Acc, and return this rule to the user;
END FOR EACH goal attribute value v.
```

The experiment reported in this section, involving 15 runs of the GA (one for each goal attribute value being predicted) took about 6 minutes on a Pentium III PC with 866 Mhz. Each run of the GA had a population size of 100 individuals, which evolved during 60 generations.

In Table 2, the rule consequent in the first column consists of an attribute-value pair "G_i= *val*" identifying the goal attribute value predicted by the rule, where G_i denotes the *i*-th attribute, *i*=1...6 (see section 4.1 for the meaning of these goal attributes) and *val* denotes the value predicted for the corresponding goal attribute. The second and third columns of this table show the degree of interestingness assigned to the rules by the user.

We emphasize that the user who evaluated the interestingness of the discovered rules was the same user who specified the general impressions, as mentioned earlier. In addition, when the user was shown a discovered rule, he was also shown his own general impression that was contradicted by that rule.

As can be observed in Table 2, out of the 13 rules extracted from the decision tree built by J4.8, the user considered only 2 rules as having a medium degree of interestingness, and he considered 11 rules as having a low degree of interest. No rule was considered as having a high degree of interestingness.

Table 2: Interestingness of rules discovered by J4.8 and the GA

Rule consequent	**Interestingness for the user**	
	J4.8	**GA**
G_1 = low	low	**high**
G_1 = medium	***medium***	***medium***
G_1 = high	low	***medium***
G_2 = low	low	**high**
G_2 = medium	low	***medium***
G_2 = high	low	low
G_3 = low	low	**high**
G_3 = medium	low	low
G_3 = high	low	low
G_4 = low	***medium***	***medium***
G_4 = high	low	***medium***
G_5 = low	low	**high**
G_5 = high	low	low
G_6 = low	N/A	**high**
G_6 = high	N/A	low

A much better result was obtained by the GA. Out of the 15 rules discovered by the GA, 5 were assigned a high degree of interestingness by the user, 5 were assigned a medium degree of interestingness, and the remaining 5 were assigned a low degree of interestingness. Overall, this seems to be a relatively good result, considering how difficult it is to discover very interesting, surprising rules.

We have observed that there is a relationship between a rule's simplicity (in the sense of having a small number of conditions) and its degree of interestingness for the user. This relationship is due to an interaction between the measure of rule surprisingness used in this work and the kind of general impressions specified by the user, as follows. In our experiments, the user specified mainly short general impressions, having a small number of conditions. As a result, the measure of rule surprisingness favors the discovery of short rules too, since these rules can have a larger degree of similarity between the rule antecedent and the general impression antecedent.

The relationship between a rule's simplicity and its degree of interestingness for the user helps to explain why the rule set discovered by the GA was considerably more interesting than the rule set discovered by J4.8. To understand this point, we have analyzed the difference between the rules discovered by the two algorithms with respect to simplicity. This result of this analysis is shown in Tables 3 and 4, which refer to the rules discovered by J4.8 and the GA, respectively. Each of these two tables is to be interpreted essentially as a 5x3 matrix. It contains five rows, each corresponding to one of the possible number of conditions in a rule antecedent, and three columns, each corresponding to one of the possible degrees of interestingness assigned by the user to a rule. Each cell *i,j* of the matrix contains the number of discovered rules having *i* conditions and having the degree of interestingness *j*. The matrix is extended with totals for each row and each column. The bottom-rightmost cell contains the total number of discovered rules. Note that this total is 13 for J4.8 (in Table 3) and 15 for the GA (in Table 4), since J4.8 did not discover any rule for two goal attribute values, as explained earlier.

Comparing the results in Table 3 with the results in Table 4, it is clear that in general the GA discovered simpler rules. For instance, out of the 13 rules discovered by J4.8, only 2 were very short, having only 1 condition. By contrast, out of the 15 rules discovered by the GA, 6 were very short, having only 1 condition. As mentioned earlier, in this project short rules tended to be more interesting for the user. Indeed, out of the 6 very short rules found by the GA 4 were considered highly interesting by the user. The other highly interesting rule discovered by the GA was also quite short, having only 2 conditions. Out of the 13 rules discovered by J4.8, only 2 had a medium degree of interesting (i.e., no rule was highly interesting), one of them had just 1 condition and the other one had 3 conditions. Both in Table 3 (J4.8) and in Table 4 (GA), all long rules – with 4 or 5 conditions – had a low degree of interestingness to the user.

These results clearly show that the user has a bias favoring simpler rules. Since the GA was designed from scratch to discover interesting rules, it managed to detect this bias implicit in the user's specification of the general impressions and the definition of the measure of surprisingness (subsection 3.3.2). Along many generations, the GA had time and enough adaptive mechanisms to adapt to the bias of the user, and in general it converged to simple rules in most runs. Indeed, only 2 out of the 15 rules discovered by the GA were long rules (with 4 or 5 conditions), as shown in Table 4. By contrast, J4.8 did not have opportunity to adapt to that bias of the user, because when using J4.8 the general impressions are used for selecting the most interesting rules only in a post-processing phase, after J4.8 has built a decision tree. As a result, 9 out of the 13 rules extracted from the decision trees built by J4.8 were long rules, which seems the major reason for the bad results of J4.8 reported in Table 2.

Table 3: Analysis of simplicity for rules discovered by J4.8

Number of conditions	**Interestingness for the user**			**Total**
	low	**medium**	**high**	
1	1	1	0	2
2	0	0	0	0
3	1	1	0	2
4	6	0	0	6
5	3	0	0	3
0	11	2	0	13

Table 4: Analysis of simplicity for rules discovered by the GA

Number of conditions	**Interestingness for the user**			**Total**
	low	**medium**	**high**	
1	0	2	4	6
2	2	1	1	4
3	1	2	0	3
4	1	0	0	1
5	1	0	0	1
Total	5	5	5	15

5 Conclusions and Future Work

We have proposed a GA for discovering interesting fuzzy prediction rules. The proposed GA was evaluated with respect to both the predictive accuracy and the interestingness of the discovered rules. With respect to the former criterion, the performance of the GA was compared with J4.8, a well-known decision-tree-building algorithm. Overall, the proposed GA was found to be competitive with J4.8 with respect to this criterion. In order to evaluate rule interestingness, the rules discovered by the GA and by J4.8 were shown to the user (University Director) in an interview. In general the user considered the rules discovered by the GA much more interesting than the rules discovered by J4.8.

Overall, the GA was able to found several rules that were considered very interesting by the user. For instance, one of the general impressions specified by the user represented his previous knowledge (or belief) that biology researchers of a given region had a high number of international edited/published books. However, the GA was able to find an accurate rule contradicting this general impression. The rule had the same antecedent as the general impression but made the opposite prediction, i.e. it predicted that the researchers in question had a low number of international edited/published books. As another example, the user also believed that researchers in the broad area of Engineering had a high number of national edited/published books, but another rule discovered by the GA revealed that the number in question is low. These two rules were considered very interesting for the user.

There are two directions for future research. First, it would be interesting to develop a multi-objective version of the GA described in this paper. In this case, the GA would search for rules that are non-dominated, in the Pareto sense [4], with respect to both predictive accuracy and rule interestingness. Second, in order to further validate the performance of the GA with respect to rule interestingness, it would be useful to compare the degree of interestingness of the rules discovered by our GA with the degree of interestingness of the rules discovered by another data mining algorithm that was specifically designed for the discovery of interesting (surprising) rules.

References

[1] T. Blickle, Tournament selection, in: T. Back, D. B. Fogel and Z. Michalewicz (Eds.), Evolutionary Computation 1: Basic Algorithms and Operators, Chapter 24 (Institute of Phisics Publishing, 2000).

[2] C. C. Bojarczuk, H.S. Lopes and A.A. Freitas, Genetic programming for knowledge discovery in chest pain diagnosis, IEEE Engineering in Medicine and Biology Magazine, 19(4) – special issue on data mining and knowledge discovery (IEEE Computer Society, Los Alamitos, CA, USA, 2000) 38-44.

[3] K. A. Crockett, Z. Bandar and A. Al-Attar, Soft decision trees: a new approach using non-linear fuzzification, Proc. of the 9th IEEE Int. Conf. on Fuzzy Systems, FUZZ IEEE'2000 (IEEE Computer Society, Los Alamitos, CA, USA, 2000) 209-215.

[4] K. Deb, Multiobjective Optimization Using Evolutionary Algorithms, (John Wiley & Sons, England, 2001).

[5] V. Dhar, D. Chou and F. Provost, Discovering Interesting Patterns for Investment Decision Making with GLOWER – A Genetic Learner Overlaid With Entropy Reduction, Data Mining and Knowledge Discovery 4(4) (Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000) 251-280.

[6] J. Espinosa and J. Vandewalle, Constructing fuzzy models with linguistic integrity from numerical data – AFRELI algorithm, IEEE Trans. on Fuzzy Systems 8(5) (IEEE Computer Society, Los Alamitos, CA, USA, 2000) 591-600.

[7] U. M. Fayyad, G. Piatetsky-Shapiro and P. Smyth, Advances in knowledge discovery & data mining, Chapter 1: From data mining to knowledge discovery: an overview, (AAAI/MIT Press, Menlo Park, CA, USA, 1996).

[8] A. A. Freitas, On objective measures of rule surprisingness, Principles of Data Mining and Knowledge Discovery (Proc. 2nd European Symp., PKDD-98), Lecture Notes in Artificial Intelligence 1510, (Springer-Verlag , Berlin, 1998) 1-9.

[9] A. A. Freitas, Understanding the crucial differences between classification and discovery of association rules – a position paper, ACM SIGKDD Explorations, 2(1), (ACM, New York, USA, 2000) 65-69.

[10] A. A. Freitas, Understanding the Crucial Role of Attribute Interaction in Data Mining, Artificial Intelligence Review 16(3), (Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001) 177-199.

[11] A. A. Freitas, Data Mining and Knowledge Discovery with Evolutionary Algorithms, (Springer-Verlag, Berlin, 2002).

[12] D. E. Goldberg, Genetic algorithms in search, optimization, and machine learning (Addison-Wesley Publishing Company, Inc., New York, 1989).

[13] J. Han and M. Kamber, Data Mining: Concepts and Techniques (Morgan Kaufmann Publishers, San Francisco, USA, 2000).

[14] D. J. Hand, Construction and Assessment of Classification Rules (John Wiley & Sons, England, 1997).

[15] R. J. Hilderman and H.J. Hamilton, Knowledge discovery and measures of interest (Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001).

[16] K. Hirota and W. Pedrycz, Fuzzy computing for data mining, Proceedings of the IEEE, 87(9), (IEEE Computer Society, Los Alamitos, CA, USA, 1999) 1575-1600.

[17] H. Ishibuchi and T. Nakashima, Designing Compact Fuzzy Rule-Based Systems with Default Hierarchies for Linguistic Approximation, CEC-99, (Washington DC, USA, 1999) 2341-2348.

[18] Y. Kim, W.N. Street and F. Menczer, Feature selection in unsupervised learning via evolutionary search, Proc. 6th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD—2000), (ACM Press, USA, 2000) 365-369.

[19] B. Liu and W. Hsu, Post-analysis of learned rules, AAAI-96, (AAAI Press, Portland, Oregon, USA, 1996) 828-834.

[20] B. Liu, W. Hsu and S. Chen, Using general impressions to analyze discovered classification rules, Third Int. Conf. on Knowledge Discovery and Data Mining, KDD-97, (AAAI Press, NewPort Beach, CA, USA, 1997) 31-36.

[21] B. Liu, W. Hsu, S. Chen and Y. Ma, Analyzing the subjective interestingness of association rules, IEEE Intelligent Systems 15(5), (IEEE Computer Society, Los Alamitos, CA, USA, 2000) 47-55.

[22] J. A. Major and J. J. Mangano, Selecting among rules induced from a hurricane database, Knowledge Discovery in Databases Workshop at AAAI-93, (AAAI Press,Washington DC, USA, 1993) 28-44.

[23] R. R. F. Mendes, F. B. Voznika, A. A. Freitas and J. C. Nievola, Discovering fuzzy classification rules with genetic programming and co-evolution, Principles of Data Mining and Knowledge Discovery (Proc. 5th European Conf., PKDD 2001) - Lecture Notes in Artificial Intelligence 2168, (Springer-Verlag, Berlin, 2001) 314-325.

[24] C. Mota, H. Ferreira and A. Rosa, Independent and Simultaneous Evolution of Fuzzy Sleep Classifiers by Genetic Algorithms, GECCO-99, (Morgan Kaufmann Publishers, San Francisco, USA, 1999) 1622-1629.

[25] E. Noda, A. A. Freitas and H. S. Lopes, Discovering interesting prediction rules with a genetic algorithm, Proc. Congress on Evolutionary Computation (CEC-99), (Washington D.C., USA, 1999) 1322-1329.

[26] E. Noda, A. A. Freitas and A. Yamakami, A distributed-population genetic algorithm for discovering interesting prediction rules, 7th Online World Conference on Soft Computing (WSC7), Held on the Internet, 2002.

[27] A. Papagelis and D. Kalles, Breeding decision trees using evolutionary techniques, Proc. 18th Int. Conf. on Machine Learning (ICML-2001), (Morgan Kaufmann Publishers, San Francisco, USA, 2001) 393-400.

[28] J. Poon and S. Prasher, A hybrid approach to support interactive data mining, Proc. of the Genetic and Evolutionary Computation Conf. (GECCO'2001), (Morgan Kaufmann Publishers, San Francisco, USA, 2001) 527-534.

[29] J. R. Quinlan, Generating production rules from decision trees, Proc. IJCAI-87, (Science & Technology Books, Milan, Italy, 1987) 304-307.

[30] J. R. Quinlan, C4.5: Programs for Machine Learning, (Morgan Kaufmann Publishers, San Francisco, USA, 1993).

[31] W. Romão, A Genetic Algorithm for Discovering Interesting Fuzzy Prediction Rules: applications to science and technology data, in: W.B. Langdon and E. Cantu-Paz et al, editors, Proceedings of Genetic and Evolutionary Computation Conference (GECCO-2002), (Morgan Kaufmann Publisher, San Francisco, CA, USA, 2002) 1188-1195.

[32] M. Setnes and H. Roubos, GA-fuzzy modeling and classification: complexity and performance, IEEE Trans. on Fuzzy Systems 8(5), (IEEE Computer Society, Los Alamitos, CA, USA, 2000) 509-522.

[33] A. Silberschatz and A. Tuzhilin, What Makes Patterns Interesting in Knowledge Discovery Systems, IEEE Transactions on Knowledge and data engineering, Vol. 8, No. 6, (IEEE Computer Society, Los Alamitos, CA, USA, 1996) 970-974.

[34] D. Walter and C. K. Mohan, ClaDia: a fuzzy classifier system for disease diagnosis, Proc. Congress on Evolutionary Computation (CEC-2000), (La Jolla, CA, USA, 2000).

[35] G. J. Williams, Evolutionary hot spots data mining: an architecture for exploring for interesting discoveries, Proc. 3rd Asia-Pacific Conference on Knowledge Discovery and Data Mining (PAKDD'99), (Springer-Verlag, Berlin, 1999) 184-193.

[36] I. H. Witten and E. Frank, Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, (Morgan Kaufmann Publishers, San Francisco, USA, 2000).

[37] N. Xiong and L. Litz, Generating Linguistic Fuzzy Rules for Pattern Classification with Genetic Algorithms, Proc. 3rd European Conf. on Principles of Data Mining and Knowledge Discovery (PKDD-99), (Springer-Verlag, Berlin, 1999) 574-579.