Discovering Interesting Prediction Rules with a Genetic Algorithm

Edgar Noda [1], Alex A. Freitas [2] and Heitor S. Lopes [3]

[1,3] CEFET-PR
CPGEI.
Av. 7 de Setembro, 3165.
Curitiba - PR.
80230-901. Brazil
edgar@dainf.cefetpr.br,
hslopes@cpgei.cefetpr.br

[2] PUC-PR
PPGIA-CCET.
R. Imaculada Conceicao, 1155
Curitiba – PR.
80.215-901. Brazil
alex@ppgia.pucpr.br
http://www.ppgia.pucpr.br/~alex

Abstract- In essence, the goal of data mining is to discover knowledge which is highly accurate, comprehensible and "interesting" (surprising, novel). Although the literature emphasizes predictive accuracy and comprehensibility, the discovery of interesting knowledge remains a formidable challenge for data mining algorithms. In this paper we present a genetic algorithm designed from the scratch to discover interesting rules. Our GA addresses the dependence modelling task, where different rules can predict different goal attributes. This task can be regarded as a generalization of the classification task, where all rules predict the same goal attribute.

1 Introduction

In essence, the goal of data mining is to discover knowledge which is highly accurate, comprehensible and "interesting" (surprising, novel). Many data mining algorithms are explicitly designed to discover accurate and comprehensible knowledge. Actually, the goal of discovering comprehensible knowledge is significantly facilitated by the use of rule induction algorithms (including decision trees). This kind of algorithm discovers high-level prediction rules, in the form:

IF some conditions on the values of predicting attributes are true
THEN predict a value for some goal attribute .

However, the discovery of truly interesting rules remains a formidable challenge for data mining. Recently, several authors have presented different viewpoints on what makes a discovered rule interesting. For instance, [Freitas, 98], [Suzuki & Kodratoff, 98] discuss some objective measures of rule interestingness (or surprisingness), while [Liu et al. 97] discusses a subjective criterion for evaluating rule interestingness.

This paper proposes a Genetic Algorithm (GA) designed to discover interesting prediction rules. The proposed algorithm combines some characteristics of the GA-Nuggets algorithm [Freitas, 99] with some ideas on how to evaluate rule interestingness in an objective (data-driven, domain-independent) manner [Freitas, 98].

Our GA was designed for dependence modelling, a data mining task which can be seen as a generalization of the classification task. In this latter the aim is to predict the value of a special attribute, called the goal attribute, given the values of other attributes, called predicting attributes. Hence, all rules have the same attribute in their consequent ("*THEN* part"). In dependence modelling, similarly to classification, the aim is to discover rules that predict the value of a goal attribute, given the values of predicting attributes. However, unlike classification, there is more than one goal attribute. Hence, different rules can have different attributes in their consequent.

In our approach for dependence modelling, the user specifies a small set of goal attributes, which (s)he is interested in predicting. Note that this task is different from the discovery of association rules. In the latter the task is complety symmetric with respect to the attributes, i.e. any attribute can occur either in the antecedent or in the consequent of the rule. In contrast, in our dependence modelling task, just a few user selected attributes can occur either in the antecedent or in the consequent of the rule. All the other, non-goal attributes can occur only in the rule antecedent.

Therefore, for a given data set, the dependence modelling task has a search space much larger than the classification task. This is one of the motivations for using a more robust, global-search method like GAs. In contrast, most data mining methods are based on the rule induction paradigm, where the algorithm usually performs a kind of local search (hill climbing).

Another (related) motivation for using GAs is that they tend to cope better with attribute interaction, when compared with most rule induction methods. Indeed, in a GA the fitness function evaluates the individual (in our case, a candidate rule) as a whole, i.e. all the interactions among attributes are taken into account. In contrast, most rule induction methods select one attribute at time and evaluate a partially constructed candidate rule, rather than a full candidate rule. As a result, most rule induction methods tend to be quite sensitive to attribute-interaction problems.

2 The Genetic Algorithm

This section presents our GA designed for dependence modelling. The algorithm is based on GA-Nuggets [Freitas, 99]. The current version of the system handles only categorical attributes. Future enhancements will allow the use of continuous attributes.

2.1 Individual's Encoding

Each individual in this algorithm represents a candidate rule of the form "*if A then C*". The antecedent of this rule can be formed by a conjunction of at most $n - 1$ attributes, where n is the number of attributes being mined. Each condition is of the form $A_i = V_{ij}$, where A_i is the *i-th* attribute and V_{ij} is the *j-th* value of the *i-th* attribute's domain. The consequent consists of a single condition of the form $G_k = V_{kl}$, where G_k is the *k-th* goal attribute and V_{kl} is the *l-th* value of the *k-th* goal attribute's domain. The user specifies a set of potential goal attributes, whose prediction is considered interesting. Of course, when a potential goal attribute occurs in the consequent of the rule, it cannot occur in the antecedent of it. However, if a potential goal attribute does not occur in a rule consequent, it can occur in the antecedent of that rule, as if it was a predicting attribute.

A string of fixed size encodes an individual with n genes representing the values that each attribute can assume in the rule (see Figure 1). The algorithm automatically chooses the best goal attribute to put in the consequent, for a given rule antecedent (see section 2.3).

A_1	A_2	A_3	. . .	. . .	. . .	A_n

Figure 1: Chromosome representation.

If an attribute is not present in the rule antecedent, the value of its gene is "*-1*". This value is a flag to indicate that the attribute does not occur in the rule antecedent. Hence, this encoding effectively represents a variable-length individual (rule).

2.2 Fitness Function

The fitness function consists of two parts. The first one measures the degree of interestingness of the rule, while the second measures its predictive accuracy.

The computation of the degree of interestingness of a rule, in turn, consists of two terms. One of them refers to the antecedent of the rule and the other to the consequent. The degree of interestingness of the rule antecedent is calculated by an information-theoretical measure, which is a normalized version of the measure propose by [Freitas 98]. Initially, as a preprocessing step, the algorithm calculates the information gain of each attribute (*InfoGain*) [Cover & Thomas, 91]. Then, the degree of interestingness of the rule antecedent (*AntInt*) is given by:

$$\text{AntInt} = 1-\left(\frac{\dfrac{\sum_{i=1}^{n} InfoGain(A_i)}{n}}{\log_2(|\,dom(G_k)\,|)}\right) \qquad [1]$$

Where n is the number of attributes in the antecedent and $|Dom(G_k)|$ is the domain cardinality (i.e. the number of possible values) of the goal attribute G_k occurring in the consequent. The log term is included in formula [1] to normalize the value of *AntInt*, so that this measure takes on a value between *0* and *1*. The *InfoGain* is given by:

$$\text{InfoGain}(A_i) = \text{Info}(G_k) - \text{Info}(G_k|A_i) \qquad [2]$$

Where

$$\text{Info}(G_k) = -\sum_{i=1}^{mk}\left(\Pr(V_{kl})\log_2(\Pr(V_{kl}))\right) \qquad [3]$$

$$\text{Info}(G_k|A_i) = \sum_{i=1}^{n_i}\left(\Pr(V_{ij})\left(-\sum_{j=1}^{mk}\Pr(V_{kl}\,|\,V_{ij})\log_2(\Pr(V_{kl}\,|\,V_{ij}))\right)\right) \qquad [4]$$

Where m_k is the number of possible values of the goal attribute G_k , n_i is the number of possible values of the attribute A_i, $Pr(X)$ denotes the probability of X and $Pr(X|Y)$ denotes the conditional probability of X given Y.

The *AntInt* measure can be justified as follows. Attributes with high information gain are good predictors of class, when these attributes are considered individually, i.e. one at a time. However, from a rule interestingness point of view, it is likely that the user already knows what are the best predictors (individual attributes) for its application domain, and rules containing these attributes would tend to have a low degree of interestingness for the user.

On the other hand, the user would tend to be more surprised if (s)he saw a rule containing attributes with low information gain. The user probably considered these attributes as irrelevant, and they are kind of irrelevant for classification when considered individually, one at time. However, attribute interactions can render an individually irrelevant attribute into a relevant one, and this phenomenon is intuitively associated with rule interestingness.

Therefore, all other things (such as the prediction accuracy, coverage and completeness of the rule) being equal, [Freitas 98] argues that rules whose antecedent contain attributes with low information gain are more interesting (surprising) than rules whose antecedent contain attributes with high information gain.

The computation of the consequent's degree of interestingness is based on the following idea [Freitas 99]: the larger the relative frequency (in the training set) of the value being predicted by the consequent, the less interesting it is. In other words, the rarer a value of a goal attribute, the more interesting a rule predicting it is. For instance, a rule predicting a rare disease is much more interesting than a rule predicting a healthy condition, when 99% of the patients are healthy. More precisely, the formula for measuring the degree of interestingness of the rule consequent (*ConsInt*) is:

$$\text{ConsInt} = \left(1 - \Pr\left(G_{kl}\right)\right)^{1/\beta} \qquad [5]$$

where $Pr(G_{kl})$ is the prior probability (relative frequency) of the goal attribute value G_{kl}, and β is a user-specified parameter, empirically set to *2* in our experiments. The exponent $1/\beta$ in the equation [5] can be regarded as a way of reducing the influence of the rule consequent interestingness in the value of the fitness function.

The second part of the fitness function measures the predictive accuracy (*PredAcc*) of the rule, and it is given by:

$$\text{PredAcc} = \frac{|A \,\&\, C| - 1/2}{|A|} \qquad [6]$$

where $|A\&C|$ is the number of examples that satisfy both the rule antecedent and the consequent, and $|A|$ is the number of cases that satisfy only the rule antecedent. The term ½ is subtracted in the numerator of equation [6] to penalize rules covering few training examples – see [Quinlan 87].

Finally, the fitness function is:

$$\text{Fitness} = \frac{w_1 \cdot \dfrac{AntInt + ConsInt}{2} + w_2 . PredAcc}{w1 + w2} \qquad [7]$$

Where W_1 and W_2 are user-defined weights. In our experiment they are set to 1 and 2, repectively. Note that PredAcc is given a greater weight than *AntInt* and *ConsInt*. All the components of equation [7] are normalized.

2.3 Genetic Operators and Rule Consequent Formation

Our GA uses the tournament selection where a pair of individuals is randomly picked and the one with the higher fitness is chosen for reproduction.

The crossover operator follows the idea of uniform crossover [Syswerda 89]. There is a probability for applying crossover to a pair of individuals and another probability for swapping each gene (attribute)'s value in the genome (rule antecedent) of two individuals. After crossover is done, the algorithm analyses if any invalid individual was created. If so, a repair operator is used to produce valid-genotype individuals. The rates used were 0.7 for the crossover operator and 0.5 for attribute value swapping.

The mutation operator randomly transforms the value of an attribute into another value belonging to the domain of that attribute. The mutation rate used was 0.05.

Besides crossover and mutation, there is the insert and remove operators that directly try to control the size of the rules being evolved, so influencing the comprehensibility of the rules. These operators randomly insert or remove, respectively, a condition in the rule antecedent.

The probability of applying each of these operators depends on the number of attributes in the rule antecedent. The insert operator has a null probability of application when the rule antecedent has the maximum number of attributes (as specified by the user). The removal operator works in the opposite way. When the number of attributes in the rule antecedent is minimum (as specified by the user) it has a null probability of application. The insert and remove rates used in our experiment are 0.5 and 0.7 respectively.

The previous genetic operators act in the rule antecedent. Once these operators have been applied and the rule antecedent is formed, the algorithm chooses the best consequent for each rule in such a way that maximizes the fitness of an individual (candidate rule). In other words, the rule consequent is carefully chosen, in a deterministic manner, to produce the best possible rule for a fixed antecedent. In effect, this approach gives the algorithm some knowledge of the data mining task being solved. A similar approach has been used by some GAs designed to perform a classification task – see e.g. [Green & Smith 93].

A kind of elitism is used in our algorithm. In the task of dependence modelling, there are several possible pairs of <goal attribute, goal attribute value> that can occur in the rule consequent. In each generation, the best rule predicting each pair of <goal attribute, goal attribute value> is passed unaltered to the next generation. This corresponds to using an elitism factor of *K*, where *K* is the number of distinct rule consequents occurring in the current population.

3 The Data Sets Used in the Experiments

The data sets used to test the algorithm were obtained from the UCI repository of machine learning databases (*http://www.ics.uci.edu/AI/Machine-Learning.html*). The data sets used are the Zoo and Nursery. They are normally used for evaluating algorithms performing the classification task. In the absence of a specific benchmark data set for the dependence modelling task, these data sets were chosen because they seem to contain more than one potential goal attribute.

The zoo database contains 101 instances and 18 attributes. Each instance corresponds to an animal. In the pre processing phase the attribute containing the name of the animal was removed, since this attribute has no generalization power. The attributes in the zoo data set are all categorical. The attribute names are as follows: hair, feathers, eggs, milk, predator, toothed, domestic, backbone, fins, legs, tail, catsize, airborne, aquatic, breathes, venomous and type. Except type and legs, the attributes are boolean. In our experiments the set of potential goal attributes used was predator, domestic and type. Predator and domestic are boolean attributes, whereas the type attribute can take on seven different values.

The nursery school data set contains 12960 instances and 9 attributes. The attributes are all categorical. The attribute names are as follows: parents, health, form, children, finance, housing, social, has_nurs and recommendation. In our experiments, the attributes used as potential goal attributes were finance, social and recommendation with 2, 3, 5 possible values respectively.

4 Results and Discussion

In a classification task, there are some well-defined measures of predictive accuracy. For instance, a commonly used measure, despite its defects [Hand 97], is the accuracy rate, i.e. the ratio of the number of correctly classified test instances over the total number of test instances.

The target of this work is the dependence modelling task which, as mentioned before, is a generalization of the classification task where different rules can predict different attributes. In both tasks the evaluation of the discovered rules must take into account their predictive accuracy on a separate test set. The difference is as follows. In classification we usually aim at discovering a rule set that can classify any test instance that appears in the future. Hence it makes sense to compute an accuracy rate or related measure over all instances in the test set.

In dependence modelling, in the sense addressed in this paper, we do not aim to classify the whole test set. Rather, the goal is to discover a few interesting rules to be shown to a user. We can think of the discovered rules as the most valuable "knowledge nuggets" extracted from the data. These knowledge nuggets are valuable even if they do not cover the whole test set. In other words, the value of the discovered rules depends on their predictive accuracy on the part of the test set covered by those rules, but not on the test set as a whole. After all, there are several goal attributes, and it is not realistic to expect that the discovered rules can predict the value of all goal attributes for all instances in the test set. In fact, we could mine such a large rule set by running one classification algorithm for each goal attribute, but we would get too many rules, and the task being solved would be simply "multiple classification". In contrast, in the dependence modelling task addressed in this paper we aim at discovering a much smaller set of interesting rules.

Hence, it does not make much sense to evaluate the performance of the discovered rule set as a whole in test set, and the discovered rules are better evaluated on a rule-by-rule basis.

Within this spirits, for each data set (zoo, nursery), we performed two experiments. The first one consisted of using cross-validation with factor 5 to evaluate the quality of the discovered rules. Hence, the data set is divided into 5 mutually exclusive and exhaustive partitions.

The GA is then run 5 times. Each time a different partition is used as the test set and other 4 partitions are merged and used as the training set. The results of the 5 runs are then averaged.

The second experiment consisted of using the full data set (i.e. all the 5 partitions associated with cross validation) to discover the final rules to be reported to the user. Obviously, the predictive power of these final rules is not estimated by their predictive accuracy on the full data set, but rather by the average predictive accuracy on the test set computed in the first experiment (cross-validation procedure).

However, this second experiment (discovering rules from the full data set) is necessary and advisable for two reasons. First, although we can use cross-validation to compute the average of some rule-quality indicators (such predictive accuracy), we cannot use cross-validation to compute "average rules" (note that each of 5 runs associated with cross-validation discovers a potentially different set of rules).

Second, The rules discovered from the full data set have the advantage of being discovered from a somewhat larger data set than the rules discovered during cross-validation.

If the reader is not familiar with above-described usage of cross-validation and subsequent learning from the full data set, (s) he is referred to [Hand 97].

The Results for the Zoo and Nursery data sets are reported and discussed in the next two subsections. For the Zoo data set the population consisted of 100 individuals, and the number of generations was 100. For the nursery data set the population consisted of 50 individuals, and the number of generations was 100.

4.1 Results for the Zoo Data Set

Table 1 shows the average results, over the 5 runs of the cross-validation procedure, for the zoo data set. Each row of this table is associated with the best-discovered rule for a different rule consequent (a pair <goal attribute, goal attribute value>). In the Zoo data set there are 11 distinct rule consequents (since the goal attributes predator, domestic and type can take on 2, 2 and 7 distinct values, respectively). Hence table 1 has 11 rows. (Note that each row is associated with an individual of the last generation, since we used an elitism strategy for preserving the best rule for each rule consequent.)

The first column of Table 1 indicates the pair <goal attribute, goal attribute value> characterizing the rule consequent. The second column shows the average value of the degree of interestingness of the rule antecedent, *AnInt* (equation [1]), for the rules having the consequent specified in the first column. This average was computed over 5 rules, namely the best rule having that consequent in each of the 5 runs of the cross-validation procedure. The third column of table 1 shows the value of the degree of interestingness of the rule consequent, *ConsInt* (equation [5]), for all the rules having the consequent specified in the first column. (Note that the value of *ConsInt* is constant for all the rules with a given consequent.) The fourth and fifth columns of Table 1 show the predictive accuracy on the training and test sets for the rules having the consequent specified in the first column. The sixth column shows the average number of examples covered for the rules having the consequent specified in the first column. The average results reported in the fourth, fifth and sixth columns were computed in a manner similar to the above-described computation of the average for the second column.

The predictive accuracy on the training set was computed as defined in equation [6]. The predictive accuracy on the test was computed by a simplified version of equation [6], namely $|A \ \& \ C| \ / \ |A|$. Note that, in the case of the test set, there is no need to subtract ½ from $|A \ \& \ C|$, as it was done for the training set. That subtraction was used in training set to compensate for the fact that the measure $|A \ \& \ C| \ / \ |A|$ is a too optimistic estimate of the predictive accuracy of a rule on unseen data [Quinlan 87]. Hence, this correction is not necessary in the test set, which contains data unseen during training.

As can be seen in Table 1, overall the discovered rules have a very high value of *AntInt*, i.e. their antecedent are considered (by the measure specified in equation [1]) to be very interesting (surprising).

In addition, Table 1 shows that most of the discovered rules have a good generalization performance on the test set. Five out of the 11 rules have a predictive accuracy of 100% in the test set. Two other rules have a high predictive accuracy in the test set, keeping the same performance level as in the training set. Only four rules have a predictive accuracy in the test set significantly smaller than the one in the training set. Note that the value of $|A|$ is not as small as it might look at first glance, since the test set has only 20 instances.

The final rules discovered from the full data set are show in Table 2. The table shows the best rule for each possible rule consequent. As explained before, the quality of these rules is best estimated by looking up the corresponding rows in Table 1. In any case, for the sake of completeness, we have included in table 2, for each rule, the antecedent's degree of interest, *AntInt*; the consequent's degree of interest, *ConsInt*; the number of examples covered by the rule antecedent, $|A|$; and the number of correctly predicted examples, $|A \ \& \ C|$.

Note that most of the rules are not only interesting and highly accurate, as shown in Table 1, but also comprehensible, at least in the sense of representing high-level knowledge and not being very long.

Table 1: Results of cross-validation for the Zoo data set.

<Goal attribute, Attribute value >	***AntInt***	***ConsInt***	***PredAcc Training***	***PredAcc Test***	***|A| Test***
<predator, false>	0.975044	0.7444062	0.9090476	0.4625	4.0
<predator, true>	0.9485458	0.6670112	0.9427776	0.875	3.0
<Domestic, false>	0.9626236	0.3571276	0.977195	0.9714286	5.0
<Domestic, true>	0.9692068	0.9333076	0.50	0.0	1.33
<type, 1>	0.946871	0.7705938	0.9805696	1.0	6.4
<type, 2>	0.9388816	0.8955094	0.9681758	1.0	3.6
<type, 3>	0.932396	0.9749324	0.8333332	0.0	1.0
<type, 4>	0.9342656	0.9334096	0.950909	1.0	2.2
<type, 5>	0.936234	0.9799894	0.80	1.0	1.0
<type, 6>	0.9238708	0.9595622	0.9050002	1.0	1.25
<type, 7>	0.9530434	0.949211	0.919881	0.91666675	2.25

Table 2: Results of learning from the full zoo data set.

<Goal, Value>	***Discovered Rule***	***AntInt***	***ConsInt***	***|A & C|***	***|A|***
<predator, false>	If (hair=0) and (eggs=1) and (milk=0) and (backbone=1) and (tail=1) and (domestic=1) Then (predator=0)	0.984311	0.744618	4	4
<predator, true>	If(airbone=0) and(aquatic=1) and (backbone=1) and (catsize=1) Then (predator=1)	0.952456	0.667491	11	11
<domestic, false>	If(eggs=1)and(airbone=0) and (predator=1) and (venomous=0) Then (Domestic=0)	0.957968	0.358766	22	22
<domestic, true>	If(eggs=0)and(aquatic=0) and (tail=0) Then (domestic=1)	0.959166	0.933428	1	1
<type, 1>	If(eggs=0)and(venomous=0)and (domestic=0) Then (type=1)	0.949641	0.770752	32	32
<type, 2>	If(feathers=1)and(venomous=0)and (Domestic=0) Then (type=2)	0.955213	0.895533	17	17
<type, 3>	If(eggs=1)and(aquatic=0)and(predator=1)and (toothed=1) and(fins=0) and(domestic=0) and(catsize=0) Then(type=3)	0.936044	0.974933	3	3
<type, 4>	If(aquatic=1)and(breathes=0)and(venomous=0) and(tail=1) Then(type=4)	0.939000	0.933428	12	12
<type, 5>	If(airbone=0)and(aquatic=1)and(toothed=1)and (breathes=1)and(catsize=0) Then(type=5)	0.921090	0.979998	4	4
<type, 6>	If(airbone=1)and(fins=0)and(tail=0) Then(type=6)	0.928637	0.959579	6	6
<type, 7>	If(predator=1)and(breathes=0)and(tail=0)and (Domestic=0) Then(type=7)	0.953098	0.949205	7	7

4.2 Results for the Nursery Data Set

Table 3 shows the average results, over the 5 runs of the cross-validation procedure, for the Nursery data set. The meaning of the columns of table 3 is analogous to the meaning of Table 1's columns, as explained in the previous section. In the nursery data set there are 10 distinct rule consequents (since the goal attribute finance, social and recommendation can take on 2, 3 and 5 values, respectively).

The results reported in Table 3 are even better than the results reported in Table 1. Again, overall the discovered rules have a high value of *AntInt* and have a very good generalization performance on the test set. Seven out of the tem rules have a predictive accuracy of 100% in the test set. The other three rules have a much lower performance in the test set.

The final rules discovered from the full data set are shown in table 4. The table shows the best rule for each possible consequent. The meaning of Table 4's columns is analogous to the meaning of Table 2's columns. Again, we warn the reader that the last 4 columns of Table 4 were included only for the sake of completeness, since the quality of the rules shown in Table 4 is best estimated by the correspond rows in Table 3.

Table 3: Results of cross-validation for the Nursery data set.

<Goal attribute, Attribute value >	***AntInt***	***ConsInt***	***PredAcc Training***	***PredAcc Test***	***|A| Test***
<finance, convenient>	0.9989392	0.7067302	0.9777228	1.0	6.6
<finance, incov>	0.9988632	0.7074786	0.9807822	1.0	7.0
<social, non_prob>	0.9980038	0.8164956	0.5216054	0.285353	16.0
<social, slightly_prob>	0.997638	0.8164952	0.5429434	0.2818183	47.8
<social, problematic>	0.997279	0.816496	0.9875932	1.0	9.0
<recommendation, not_recom>	0.9383624	0.8164954	0.998279	1.0	69.4
<recommendation, recommended>	0.9971632	0.9998826	0.0006376	0.002359	432.0
<recommendation, very_recom>	0.9423296	0.9805278	0.9566708	1.0	3.0
<recommendation, priority>	0.921443	0.7744442	0.9949844	1.0	21.4
<recommendation, spec_prior>	0.9212322	0.8788452	0.9949754	1.0	22.8

Table 4: Results of learning from the full nursery data set.

<Goal, Value>	***Discovered rule***	***AntInt***	***ConsInt***	***|A & C|***	***|A|***
<finance, convenient>	If(has_nurs=very_crit) and(children= more) and(health=recommended) and(class=priority) Then(finance=convenient)	0.998888	0.706733	14	14
<finance, incov>	If(has_nurs=very_crit) and(housing=convenient) and(social=slightly_prob) and (health=recommended) and(class=spec_prior) Then (finance=incov)	0.999111	0.707481	20	20
<social, non_prob>	If(form=complete) and(housing=critical) and(class=very_recom) Then (social=nom_prob)	0.996424	0.816497	10	20
<social, slightly_prob>	If(form=completed) Then (social=slightly_prob)	1.0	0.816497	720	2160
<social, problematic>	If(parents=usual)and (has_nurs=critical) and (health=recommended) and (class=spec_prior) Then (social=problematic)	0.997318	0.816497	63	63
<recommendation, not_recom>	If (parents=usual) and (form=foster) and (housing=less_conv) and (finance=incov)and (health=not_recom) Then (class=not_recom)	0.949673	0.816497	120	120
<recommendation, recommended>	If(children=1) Then (class=recommended)	0.997195	0.999882	2	2160
<recommendation, very_recom>	If(parents=pretentious)and (has_nurs=less_proper) and (housing=convenient) and (finance=convenient) and (social=slightly_prob) and (health=recommended) Then (class=very_recom)	0.943364	0.980528	16	16
<recommendation, priority>	If (parents=pretensious) and (has_nurs=less_proper) and (form=more) and (health=priority) Then(class=priority)	0.934211	0.774445	72	72
<recommendation, spec_prior>	If (parents=usual)and(has_nurs=very_crit) and (form=more) and (finance=incov) and (health=priority) Then (class=spec_prior)	0.934211	0.878845	72	72

5 Conclusion and Future Work

Our genetic algorithm (GA) was designed from the scratch to discover a few interesting rules, which might be called “knowledge nuggets”, rather than to discover a large set of accurate (but not necessarily interesting) rules.

The results reported in this paper are very promising, since the GA did discover rules which were both highly accurate on an unseen test set and interesting.

However, a more extensive empirical evaluation of our GA would be useful, and will be object of future research. We also intend to extend the GA described in this paper to cope with continuous attributes (the current implementation can handle only categorical attributes).

Another direction for further research would be to evaluate the performance of the GA with other rule interestingness measures proposed in the literature.

Bibliography

[Cover & Thomas, 91] Cover, T. M. and Thomas, J. A.. *Elements of Information Theory*. John Wiley &Sons. (1991).

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

[Freitas, 99] Freitas, A. A. *A genetic algorithm for generalized rule induction.* In: R. Roy et al. Advances in Soft Computing - Engineering Design and Manufacturing. 340-353. Springer-Verlag, (1999).

[Greene & Smith 93] Greene, D. P. & Smith, S. F. *Competition-based induction of decision models from examples.* Machine Learning 13, 229-257 (1993).

[Hand 97] Hand, d. *Construction and Assessment of Classification Rules.* John Willey &sons. (1997).

[Liu et al, 97] Liu, B., Hsu, W. & Chen, S. *Using general impressions to analyze discovered classification rules.* Proc. 3rd Int. Conf. on Knowledge Discovery & Data Mining (KDD-97), 31-36. AAAI Press, (1997).

[Quinlan 87] Quinlan, J. R. *Generating production rules from decision trees*. Proc. IJCAI-87, 304-307 (1987).

[Suzuki & Kodratoff, 98] Suzuki, E. & Kodratoff, Y. *Discovery of surprising exception rules based on intensity of implication.* Proc. 2nd European Symp. on Principles of Data Mining and Knowledge Discovery (PKDD-98). Lecture Notes in Artificial Intelligence. 1510, 10-18. (1998).

[Syswerda 89] Syswerda, G. *Uniform Crossover in genetic Algorithms.* Proc. 3th International conference on genetic algorithms (ICGA89). 2 – 9. (1989).