Feature Influence for Evolutionary Learning*

Raúl Giráldez
Computer Science Dept.
University of Seville
Avda. Reina Mercedes s/n
41012 Seville Spain
giraldez@lsi.us.es

Jesús S. Aguilar–Ruiz
Computer Science Dept.
University of Seville
Avda. Reina Mercedes s/n
41012 Seville Spain
aguilar@lsi.us.es

ABSTRACT

This paper presents an approach that deals with the feature selection problem, and includes two main aspects: first, the selection is done during the evolutionary learning process, i.e., it is a *dynamic* approach; and second, the selection is *local*, i.e., the algorithm selects the best features from the best space region to learn at a given time of the exploration process. While the traditional feature selection is based on the attribute relevance, our approach is based on a new concept, called *feature influence*, which is aware of the dynamics and locality of the concept. The feature influence provides a measure of the attribute relevance at a certain instant of the evolutionary learning process, since it depends on each generation. Experimental results have been obtained by comparing an EA–based supervised learning algorithm to its modified version to include the concept approached. The results show an excellent performance, as the new adapted algorithm achieves the same classification results while using less rules, less conditions in rules and much less generations. The experiments include the statistical significance of the improvement over a set of sixteen datasets from the UCI repository.

Categories and Subject Descriptors

I.2.6 [**Learning**]: Concept learning, Knowledge acquisition

General Terms

Algorithms

Keywords

Evolutionary Learning, Feature Selection, Feature Influence

*This research was supported by the Spanish Research Agency CICYT under grant TIN2004–00159

GECCO'05, June 25–29, 2005, Washington, DC, USA.

1. INTRODUCTION

Classification addresses the computational problem of predicting the class label of an unseen example, where the class is a discrete variable of practical importance. In general, given a collection of examples (training dataset), where each example contains a set of features (or attributes) and one of the features is the class, we try to find a knowledge model for the class as a function of the values of the remaining attributes, in such a way that unseen examples will be assigned a class as accurately as possible. The knowledge models are usually represented as decision rules or trees. A decision rule is a "*if* $\mathcal{C}$ *then* $\mathcal{L}$" type, where $\mathcal{C}$ is a conjunction of conditions that establish what values the attributes can take to classify an example with class label $\mathcal{L}$. When a feature a_i is discrete, the rules take the form of "*if* $a_i \in \{v_1, \ldots, v_k\}$ *then* $\mathcal{L}$", where the values $\{v_1, \ldots, v_k\}$ are not necessarily all those that the feature can take. When a feature a_j is continuous, typically the rules take the form of "*if* $a_j \in [l_j, u_j]$ *then* $\mathcal{L}$", where l_j and u_j are two real values belonging to the range of the feature and $l_j < u_j$.

Decision rule discovery is a problem with very large search space, where an exhaustive search is not applicable in practice. Evolutionary Algorithms (henceforth EAs) have been applied extensively to solve this task, providing an excellent performance. Several approaches have been developed to deal with decision rule discovering, among them: GABIL[6], GIL[11], SIA[16], ECL[7], HIDER[1] and GASSIST[2].

In addition to the efficiency and performance of the EA, the success of classification also depends on other factors, such as the quality of the data. In this sense, feature selection is the process of identifying and removing irrelevant and redundant information from the dataset [10]. The suitable attribute selection is beneficial for improving the performance of common learning algorithms from a double perspective: first, the decrease of the number of features leads to a reduction of the search space size, which has a positive influence on the convergence of the EA; second, the identification of relevant attributes allows the EA to obtain more accurate solutions, decreasing the further classification error provided by the EA.

Most of the feature selection algorithms are applied before the learning algorithm as a preprocessing method, and therefore reduce the number of attributes in a *global* way, that is, they take into account the entire space defined by the attributes [13]. There exist a number of feature selection techniques in the literature, based on statistics [12], on neighborhood [14], on information gain [8], or even on EAs [5, 15, 17] that address the problem by using this *static*

strategy. Then, the learning algorithm would only analyze the selected attributes, reducing the computational cost and generally improving the quality of data. A *dynamic* strategy, in which feature relevance is discovered at the same time that only relevant features are considered to build the prediction model, would be preferred. However, an attribute might be relevant in a specific region of the search space and irrelevant in another one. This aspect, the *local relevance* of a feature, can not be considered by feature selection methods that address only once and globally the search space.

This paper presents an approach that deals with the feature selection problem, and includes two main aspects: first, the selection is done during the learning algorithm, i.e., it is a *dynamic* approach; and second, the selection is *local*, i.e., the algorithm selects the best features from the best space region to learn at a given time of the exploration process. While the traditional feature selection is based on the attribute relevance, our approach is based on a new concept, called *feature influence*, which is aware of the dynamics and locality of our approach, as it will be shown in next sections.

The paper is organized as follows. Section 2 presents the motivation of our work. The EA–based learning algorithm we have used to show the quality of our approach is briefly described in Section 3. The concept of *feature influence* is introduced in Section 4. In Section 5 a very simple case of study is illustrated, in which the advantage of the concept can be graphically observed. An application of the approach idea is described in Section 6. The experimental results carried out with sixteen datasets from the UCI repository are discussed in Section 7. Finally, the conclusions are summarized in Section 8.

2. MOTIVATION

Feature selection has been widely researched in the machine learning literature. The use and importance of attribute selection methods is accepted and used by researchers and practitioners. In general, the goal of feature selection methods for classification is to choose attributes that are relevant to an application in order to achieve the maximum performance with the minimum measurement effort. Liu and Motoda [13] give the following definition: *"a relevant feature is a feature that if it is removed, the measure of the remaining features will deteriorate, be the measure accuracy, consistency, information, distance, or dependence"*.

Although there are many other definitions for features to be relevant [4], all of them define the relevance as a static property, since it is independent on the state of the learning process. However, an attribute might be very relevant at a specific moment and irrelevant or even harmful in others. To illustrate this idea, let us use an example: *Pima Indians Diabetes* [3], with eight continuous attributes $\{a_1,\ldots,a_8\}$ and two discrete class labels {0,1}. After a run, an EA–based learning algorithm obtains the set of rules shown in Figure 1.

Each rule has been obtained by a different call to the EA, following a sequential covering strategy. Thus, the rule $R1$ contained the attributes from a_1 to a_6. Hence, a_7 and a_8 are irrelevant features in the first call to the EA. The second rule $R2$ has only one condition for a_6, therefore the remaining attributes, including those used in $R1$, are irrelevant in the second call to the EA. If we removed all the features, except a_6, the performance of the EA might be higher when the search space and the data dimensionality are significantly

R1: if $a_1 \in [\_, 13.5]$ and $a_2 \in [\_, 143.5]$ and
$a_3 \in [\_, 118]$ and $a_4 \in [\_, 58]$ and
$a_5 \in [\_, 397]$ and $a_6 \in [\_, 58.3]$
then class=0

R2: if $a_6 \in [23.3, \_]$ then class=1

R3: if $a_7 \in [\_, 0.67]$ then class=0

R4: class=1

Figure 1: Set of rules obtained by HIDER for Pima database.

reduced. However, this particular selection is valid only for this execution, and not globally.

Commonly, learning algorithms build the knowledge model by using the set of attributes provided by the feature selection method applied previously. On the contrary, if we were able to select only those attributes that are relevant at each specific phase of the learning process, then we would achieve a dynamic feature selection. Thus, our goal is that the attribute selection process adapts to the state of the learning process, not only for each rule, but at each generation of the EA.

In Section 4, we are going to define a new concept, named feature influence, that represents the local and dynamic attribute relevance at a certain time of the rule discovery process. However, before that, it is necessary to give a brief overview of the EA–based learning algorithm we will use along the paper, and whose experimental results will be shown in Section 7.

3. HIDER

In this section, we will briefly give an overview of the EA–based learning algorithm we are going to use in this paper. As we will see in Section 7, two versions of this algorithm will be used: the original algorithm, without changes, and an adapted version to include the *feature influence* across the evolutionary process. The reader can find a detailed description of HIDER in [1, 9].

HIDER is an EA–based supervised learning tool that produces a set of decision rules. Initially, the set of rules is empty and the training dataset is complete. The call to the evolutionary process generates one rule at a time, which is included in the final set of rules and used to remove examples from the training data. If the dataset still contains examples, the evolutionary process is started again with the reduced training data in order to produce the next rule. This loop is repeated until the set of training data is empty. The final set of rules follows the same order that the rules were generated. Thus, when a new example needs to be classified, the set of rules is sequentially evaluated according to the order, so if an example is not classified by a rule, the next one is tried.

At each iteration, the individuals of the population are evaluated according to the following fitness function

$$\phi(x_{ij}) = N - CE(x_{ij}) + G(x_{ij}) + coverage(x_{ij}) \qquad (1)$$

where x_{ij} is the i^{th} individual of the j^{th} generation; N is the number of examples being processed; $CE(x_{ij})$ is the class

error, i.e. the number of examples belonging to the region defined by the rule that do not have the same class; $G(x_{ij})$ is the number of examples correctly classified by the rule; and $coverage(x_{ij})$ gives the proportion of the search space covered by the rule.

After evaluating all the individuals, the next generation is built as follows: the best one is replicated (elitism); a set of individuals are selected and replicated; another set of individuals are recombined and the offspring are included in the next generation; finally, all new individuals, except the best, are mutated according to the mutation probability.

4. ATTRIBUTE INFLUENCE

Let us assume we have a dataset with N examples and M attributes, where a_k denotes the k^{th} attribute, with $1 \leq k \leq M$. Let G be the number of generations that the EA spends to obtain a rule. Let P_j be the genetic population for the j_{th} generation, with $1 \leq j \leq G$.

Definition 1 (fitness function value)
Let $\phi(x_{ij})$ be the fitness function value *of the individual x_{ij}, where x_{ij} refers to the i^{th} individual of the population P_j (j^{th} generation).*

Equation 1 shows the fitness function used by HIDER. This function must be maximized.

Definition 2 (k–partial fitness function value)
Let $\varphi_k(x_{ij})$ be the k-partial fitness function value *of the individual x_{ij}, where the fitness $\phi(\cdot)$ is applied to all the attributes but the k^{th} attribute in the dataset, i.e., the condition for a_k in the the individual (encoded rule) is assumed to be true.*

Definition 3 (average fitness value)
The average fitness value *of the population P_j, denoted as $\overline{\phi}(P_j)$, is defined as follows:*

$$\overline{\phi}(P_j) = \frac{1}{|P_j|} \sum_{i=1}^{|P_j|} \phi(x_{ij}) \tag{2}$$

Definition 4 (average k–partial fitness value)
The average k-partial fitness value *of the population P_j, denoted as $\overline{\varphi_k}(P_j)$, is defined as follows:*

$$\overline{\varphi_k}(P_j) = \frac{1}{|P_j|} \sum_{i=1}^{|P_j|} \varphi_k(x_{ij}) \tag{3}$$

Definition 5 (influence)
The influence of the attribute a_k for a specific individual x_{ij}, denoted by $\mathcal{I}(a_k, x_{ij})$, is defined as the difference between the fitness value and the k–partial fitness value of that individual. Formally, it is given by the following expression:

$$\mathcal{I}(a_k, x_{ij}) = \phi(x_{ij}) - \varphi_k(x_{ij}) \tag{4}$$

Definition 6 (average influence)
The average influence of the attribute a_k on the population P_j, denoted by $\mathcal{I}(a_k, P_j)$, is defined as the difference between the average fitness value and the average k–partial fitness value at the j^{th} generation. Formally, it is given by:

$$\mathcal{I}(a_k, P_j) = \overline{\phi}(P_j) - \overline{\varphi_k}(P_j) \tag{5}$$

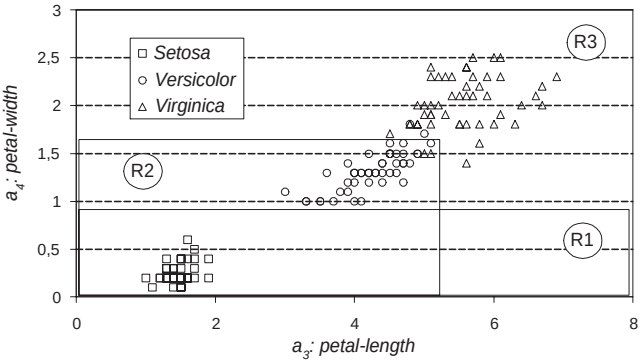

Figure 2: Iris Plants Dataset.

As we can see, there exists a direct relation between the feature influence and both the generation and genetic individuals. The feature influence provides a measure of the attribute relevance at a certain instant of the learning process, since it depends on the generation. Furthermore, this measure is also dependent on the individuals, that are encoded by rules, so it is a relevance value in a specific region of the search space defined by the features. Therefore, we can state that feature influence is a dynamical and local concept, which evolves with the evolutionary learning process.

On the other hand, while the relevance can be only high or low, the influence can be positive, negative or null, independently of the level of such influence. Positive influence ($\mathcal{I}(a_k, P_j) > 0$) means that the fact that the attribute is included in the dataset is beneficial for the accuracy of model, since removing it would imply that the average fitness is lower. On the contrary, negative influence ($\mathcal{I}(a_k, P_j) < 0$) means that the fact that the feature is included in the dataset has negative effect, since removing it would mean that the average fitness is higher. Finally, when the influence is null ($\mathcal{I}(a_k, P_j) = 0$), the atribute is irrelevant, since the fitness value does not vary.

5. A CASE OF STUDY: IRIS DATASET

In this section, we describes a very illustrative example to show graphically the concept of *feature influence*. The Iris dataset[3] is a very well–known dataset, and we are aware of its ease to be analyzed and classified due to the separable distribution of the labels. It is a very simple database with 150 examples, 4 continuous attributes (a_1:*sepal length*, a_2:*sepal width*, a_3:*petal length* and a_4:*petal width*) and 3 classes (*setosa*, *versicolor* and *virginica*). The two first features are irrelevant and the dataset is easily classified by using the features a_3 and a_4. Figure 2 illustrates the distribution of the examples and classes in the space defined by such attributes.

HIDER was run with default parameters values (a population size of 100 individuals and 100 generations), and obtained the set of decision rules shown in Figure 4. These rules are also depicted in Figure 2. Since Iris is a very simple dataset, HIDER found easily the rules without applying feature selection. However, the influence of each attribute was measured during the execution in order to observe its behavior, although these measurements were not used for selecting the relevant attributes. Figure 3 graphically shows the influence of the attributes during the learning process

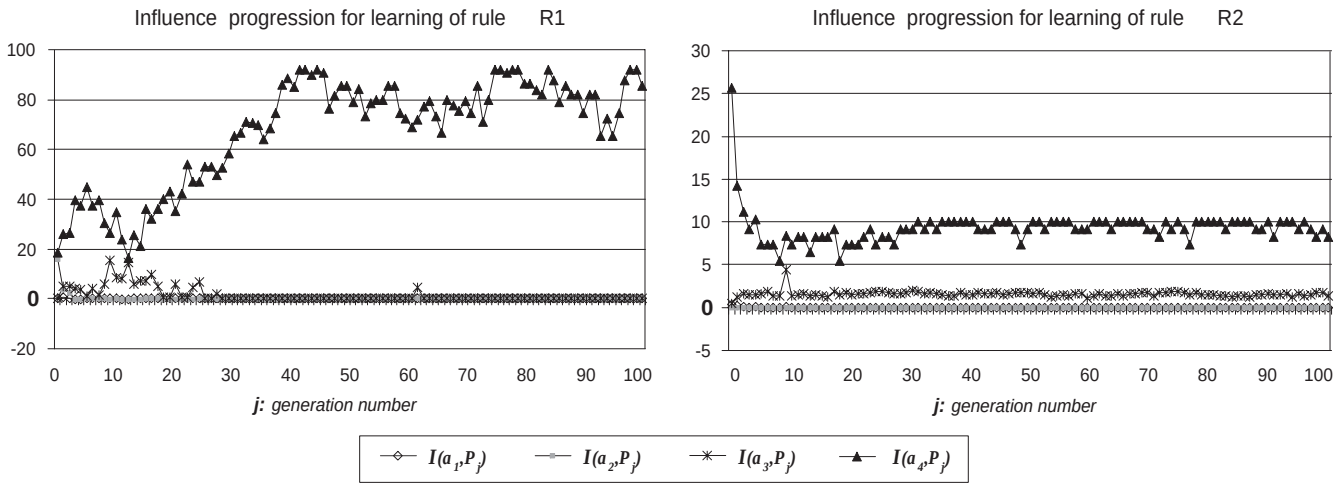

Figure 3: Feature influence for Iris.

R1: if $a_4 \in [\_, 0.8]$ then class=*setosa*

R2: if $a_3 \in [\_, 5.15]$ and
$a_4 \in [\_, 1.65]$
then class=*versicolor*

R3: class=*virginica*

Figure 4: Set of rules obtained by HIDER for Iris database.

of rules $R1$ (on the left) and $R2$ (on the right). The influence progression for the rule $R3$ is not displayed because it is the default rule and it does not contain any condition. As we can notice, the attribute a_4 offers high positive influence ($\mathcal{I}(a_4, P_j) > 0$) during all the evolutionary process for both rules. Therefore, such feature is very important for these rules and it must be considered. The attribute a_3 also shows positive influence ($\mathcal{I}(a_3, P_j) > 0$) for rule $R2$, although it is lower than the influence shown by a_4. Nevertheless, for rule $R1$, a_3 has influence in the first generations, but this influence is null from generation number 29 to the end, so that a_3 could be removed for the rest of generations. This is coherent with the results, because in fact, rule $R1$ does not present any condition for a_3. Finally, a_1 and a_2 show always null influence and they can be removed for all rules, as Figure 4 depicts. It is also worth to note that attribute a_3 starts being less important exactly when the influence of attribute a_4 increases, around generation 15^{th}.

The analysis shows several important points: first, since attributes a_1 and a_2 provides null influence, these attributes should be easily detected by any *global* feature selection technique; second, the influence of attributes a_3 and a_4 is not the same depending on the region of the search space where the EA is exploring at that generation, showing this way its *local* aspect; and third, the influence varies along the time, i.e., it is *dynamic*, and it tends to be stable while the learning process is performing.

This empirical study proves that feature influence can be very useful in EA–based learning approaches similar to HIDER. As an application of the concept we introduce, we will show in Section 6 how to adapt the mutation probability in the EA based on the influence of attributes, and in Section 7, the performance of this idea.

6. APPLICATION: MUTATION

There are several ways to integrate the feature influence in the learning process. In this section we present one of those, where the generalization probability is set according to the influence of attributes.

HIDER encodes each rule (phenotype) as one individual (genotype) by using an original representation, called *natural coding* [9]. This encoding assigns only one gene for each attribute, regardless of its type (continuous or discrete), so that each gene represents only one condition in the rule. With regard to the genetic operators, there are two kinds of mutation: the *normal mutation* causes a simple change in the value of the condition, whereas the *extreme mutation* generalizes the rule by removing completely the condition. In principle, HIDER applies these operators with constant values for the mutation probabilities per gene. In particular, such values are $\frac{1}{m}$ (where m is the number of attributes) and 0.05 for the normal mutation and the extreme mutation, respectively.

Our approach consists in setting dynamically the aforementioned probabilities according to the influence of each attribute. Notice that if we use the value of influence directly, the search could lead to the simplest solutions, but not necessarily the best ones. Thus, we analyze the influence of each feature with regard to the rest of the attributes. If $\mathcal{I}(a_k, x_{ij})$ is the influence of the attribute a_k on a specific individual x_{ij}, then by $\overline{\mathcal{I}}(a_k, x_{ij}) \pm \sigma$ we mean the average of $\mathcal{I}(a_k, x_{ij})$ (with $1 \leq k \leq m$) in the range of the standard deviation. With these values, the mutation operators are applied according to following cases:

- if $\mathcal{I}(a_k, x_{ij}) > \overline{\mathcal{I}}(a_k, x_{ij}) + \sigma$, then the normal mutation is applied with probability 1, so that this mutation favors the exploration for an attribute with high positive influence.
- if $\mathcal{I}(a_k, x_{ij}) < \overline{\mathcal{I}}(a_k, x_{ij}) - \sigma$, then the extreme mutation is applied, in order to delete the condition corre-

Table 1: Parameters of the evolutionary algorithm.

Parameter	HIDER-IF	*Standard*-HIDER
Population Size	100	
Number of Generations	A maximum of 100, depends on learning rate.	
Replication	20%	
Recombination	80%	
Individual Mutation Probability	0.5	
Gene Normal Mutation Probability	self-setting	$\frac{1}{\|attributes\|}$
Gene Extreme Mutation Probability	self-setting	0.05

Table 2: Experimental results.

	HIDER-IF				*Standard*-HIDER			
Dataset	ER $\pm$ σ	NR $\pm$ σ	NC $\pm$ σ	NG $\pm$ σ	ER $\pm$ σ	NR $\pm$ σ	NC $\pm$ σ	NG $\pm$ σ
Breast Cancer	4.1 $\pm$ 1.9	2.0 $\pm$0.0	6.3 $\pm$ 1.0	39.5 $\pm$ 8.0	4.1 $\pm$ 1.9	2.0 $\pm$0.0	5.7 $\pm$ 1.0	42.2 $\pm$ 7.3
Bupa Liver Dis.	34.4 $\pm$10.9	4.1 $\pm$0.7	8.1 $\pm$ 1.5	126.7 $\pm$ 21.6	37.7 $\pm$11.4	4.2 $\pm$0.7	12.8 $\pm$ 3.0	204.1 $\pm$ 48.0
Cleveland	22.7 $\pm$ 6.1	4.0 $\pm$0.6	23.6 $\pm$ 4.3	135.3 $\pm$ 24.1	25.0 $\pm$ 8.2	6.5 $\pm$1.0	42.2 $\pm$ 7.5	347.6 $\pm$ 88.6
German Credit	27.1 $\pm$ 2.2	4.4 $\pm$0.7	39.4 $\pm$ 7.0	150.8 $\pm$ 32.9	28.5 $\pm$ 4.3	9.8 $\pm$1.2	88.6 $\pm$14.6	561.1 $\pm$ 85.0
Glass	33.3 $\pm$ 7.1	10.5 $\pm$1.2	50.8 $\pm$ 7.1	641.5 $\pm$122.7	35.7 $\pm$ 7.5	11.6 $\pm$1.0	80.0 $\pm$ 7.8	874.1 $\pm$ 93.9
Heart Desease	19.6 $\pm$ 9.5	3.7 $\pm$0.8	10.1 $\pm$ 1.8	111.5 $\pm$ 34.2	23.3 $\pm$ 6.0	5.0 $\pm$0.9	20.2 $\pm$ 4.7	273.3 $\pm$ 26.6
Hepatitis	14.0 $\pm$10.5	3.2 $\pm$0.4	16.9 $\pm$ 2.6	92.3 $\pm$ 21.9	15.3 $\pm$10.3	4.3 $\pm$1.0	25.0 $\pm$ 6.6	173.9 $\pm$ 46.4
Horse Colic	19.2 $\pm$ 9.4	4.8 $\pm$0.4	40.0 $\pm$ 4.5	188.3 $\pm$ 30.9	25.8 $\pm$ 8.8	12.5 $\pm$1.6	111.8 $\pm$15.3	747.5 $\pm$114.9
Iris	3.3 $\pm$ 4.5	3.3 $\pm$0.6	3.2 $\pm$ 0.4	72.7 $\pm$ 19.6	4.0 $\pm$ 4.4	3.2 $\pm$0.6	3.3 $\pm$ 0.6	82.6 $\pm$ 13.9
Lenses	25.0 $\pm$25.0	4.5 $\pm$0.8	3.7 $\pm$ 1.1	79.2 $\pm$ 15.3	25.0 $\pm$25.0	4.5 $\pm$0.8	3.7 $\pm$ 1.1	81.2 $\pm$ 15.5
Mushroom	1.2 $\pm$ 0.7	3.1 $\pm$0.3	8.3 $\pm$ 2.8	112.3 $\pm$ 16.2	1.5 $\pm$ 0.5	4.1 $\pm$0.9	17.3 $\pm$ 7.8	220.9 $\pm$ 39.4
Pima Diabetes	26.7 $\pm$ 4.2	3.9 $\pm$0.3	9.8 $\pm$ 1.5	123.4 $\pm$ 16.7	27.0 $\pm$ 3.5	6.0 $\pm$0.9	29.6 $\pm$ 5.1	372.3 $\pm$ 93.2
Vehicle	35.4 $\pm$ 3.7	16.0 $\pm$2.4	204.1 $\pm$28.1	937.9 $\pm$166.0	35.7 $\pm$ 5.4	21.8 $\pm$1.9	325.0 $\pm$33.0	1860.3 $\pm$164.4
Vote	4.4 $\pm$ 2.8	2.1 $\pm$0.3	1.1 $\pm$ 0.3	63.1 $\pm$ 9.2	5.1 $\pm$ 2.5	3.2 $\pm$0.9	4.2 $\pm$ 2.9	103.9 $\pm$ 26.8
Wine	4.7 $\pm$ 5.1	4.0 $\pm$0.0	16.4 $\pm$ 2.2	177.4 $\pm$ 18.5	4.8 $\pm$ 4.4	6.1 $\pm$0.9	56.4 $\pm$10.9	483.2 $\pm$ 73.5
Zoo	6.0 $\pm$ 6.6	7.5 $\pm$0.5	12.2 $\pm$ 4.6	180.5 $\pm$ 20.8	6.0 $\pm$ 6.6	7.9 $\pm$0.8	13.5 $\pm$ 2.7	216.3 $\pm$ 34.0

sponding to an attribute with low positive influence or negative influence.

- if $\overline{\mathcal{I}}(a_k, x_{ij}) - \sigma \leq \mathcal{I}(a_k, x_{ij}) \leq \overline{\mathcal{I}}(a_k, x_{ij}) + \sigma$, then the kind of influence is not clear and normal mutation is applied with a probability proportional to $\mathcal{I}(a_k, x_{ij})$.

In order to show the quality of our approach, we have designed the experiments described in the next section, where we compare the results obtained by the standard HIDER to that obtained by using the dynamic setting of mutation probabilities.

7. EXPERIMENTAL RESULTS

Two different versions of HIDER have been run with a set of datasets from the UCI Repository [3]. The first one is called *Standard*-HIDER and is the version presented in [9]. The second one is the same algorithm but it incorporates the self–setting of mutation probability according to the previous section. We named this version HIDER-IF. Both were run with the parameters given in Table 1.

To measure the performance of each method, a 10–fold cross–validation was achieved with 16 datasets. For each dataset, the algorithms obtained 10 models (set of hierarchical rules), one per fold. The algorithms were run on the same training sets and the knowledge models tested using the same test sets, so the results were comparable. The values that represent the performance are the error rate (ER), the number of rules (NR), the number of conditions (NC) and the number of generations (NG). ER measures the accuracy and is the average number of misclassified examples expressed as a percentage. NR and NC give a measurement of the complexity of the model, being the average number of rules and the average number of conditions per model, respectively. Finally, NG measures the convergence and is the average number of generations that the algorithm iterated for producing a set of rules. Table 2 gives the results produced by each method. The first column shows the datasets used in the experiments; the next four columns give respectively the ER, NR, NC and NG together with the standard deviations (σ) obtained by HIDER-IF. Likewise, the last four columns give the same measurements for *Standard*-HIDER.

As we can see in Table 2, HIDER-IF obtained better results than the standard version for most experiments. The specific improvement produced for each measurement is shown in Table 3. Thus, ϵ_{er} is the improvement for the error rate (ER) and it has been calculated by dividing error rate of HIDER-IF by the corresponding error rate of *Standard*-HIDER. The same operation has been carried out to obtain the other improvement coefficients (ϵ_{nr}, ϵ_{nc} and ϵ_{ng}), but using the corresponding measurements (number of rules, number of conditions and number of generations, respectively). Furthermore, we applied the Student's T–test of difference of means with a critical value $\alpha < 0.05$, in order to determine whether the improvements were statistically significant or not.

Table 3: Improvements.

Dataset	ER			NR			NC			NG		
	ϵ_{er}	*Sig.*	*ST*	ϵ_{nr}	*Sig.*	*ST*	ϵ_{nc}	*Sig.*	*ST*	ϵ_{ng}	*Sig.*	*ST*
Breast Cancer	1	=		1	=		0.9	−		1.07	+	
Bupa Liver Disorder	1.09	+		1.02	+		1.58	+	•	1.61	+	•
Cleveland	1.1	+		1.63	+	•	1.79	+	•	2.57	+	•
German Credit	1.05	+		2.23	+	•	2.25	+	•	3.72	+	•
Glass	1.07	+		1.1	+	•	1.57	+	•	1.36	+	•
Heart Desease	1.19	+		1.35	+	•	2	+	•	2.45	+	•
Hepatitis	1.1	+		1.34	+	•	1.48	+	•	1.88	+	•
Horse Colic	1.35	+		2.6	+	•	2.8	+	•	3.97	+	•
Iris	1.2	+		0.97	−		1.03	+		1.14	+	
Lenses	1	=		1	=		1	=		1.03	+	
Mushroom	1.25	+		1.32	+	•	2.08	+	•	1.97	+	•
Pima Diabetes	1.01	+		1.54	+	•	3.02	+	•	3.02	+	•
Vehicle	1.01	+		1.36	+	•	1.59	+	•	1.98	+	•
Vote	1.16	+		1.52	+	•	3.82	+	•	1.65	+	•
Wine	1.02	+		1.53	+	•	3.44	+	•	2.72	+	•
Zoo	1	=		1.05	+		1.11	+		1.2	+	•
Average	1.1			1.41			1.97			2.08		

In Table 3, there are three columns for each measurement. The first of them gives the improvement coefficient (ϵ_{er}, ϵ_{nr}, ϵ_{nc} or ϵ_{ng}). The next column (*Sig.*) shows the sign of the improvement, the meaning of the symbols being as follows: "−" denotes that HIDER-IF obtained worse result than *Standard*-HIDER; "+" denotes that HIDER-IF obtained better result than the standard version; and "=" denotes that both algorithms obtained the same result. In the last column (*ST*), the results of the statistical test are shown, so that a "•" means that the result is statistically significant. Finally, the last row shows the average results for each improvement coefficient.

As Table 3 shows, the results are clearly favorable for HIDER-IF, specially regarding the complexity and convergence. HIDER-IF reduced the ER for 13 out of 16 datasets, although such improvement is not significant. For the remainder, the ER was the same, resulting a global improvement of 10% on average. The NR was also reduced for 13 datasets, but in this case, 11 out of them were statistically significant, with an average improvement of about 40%. Likewise, HIDER-IF obtained a smaller number of conditions for 14 out of 16 datasets, with 12 positive cases for the statistical test. For the number of conditions, the average improvement was still greater (97%), i.e., HIDER-IF reduces about half the complexity of the models. Although the results show that HIDER-IF has a better performance, we must notice that those numbers were obtained using smaller number of generations for all of datasets. Such reduction was statistically significant in 13 cases. On average, it needed half of the generations used by *Standard*-HIDER.

8. CONCLUSIONS

This paper presents an approach that deals with the feature selection problem, and includes two main aspects: first, the selection is done during the learning process, i.e., it is a *dynamic* approach; and second, the selection is *local*, i.e., the algorithm selects the best features from the best space region to learn at a given time of the exploration process. While the traditional feature selection is based on the attribute relevance, the method presented here is based on a new concept, called *feature influence*, which is aware of the dynamics and locality of the approach. The concept is applied to adapt the mutation probability, and presents an excellent performance as it is shown in the experimental results.

As future work, we will use the approached concept to remove dynamically attributes during the learning process. In this way, we will obtain similar results by using less computational resources, specially when the datasets have thousands of attributes, as it is usual in the bioinformatics field.

9. REFERENCES

[1] J. S. Aguilar–Ruiz, J. C. Riquelme and M. Toro. Evolutionary Learning of Hierarchical Decision Rules. *IEEE Transactions on Systems, Man and Cybernetics – Part B*, **33**(2)(2003), 324–331.

[2] J. Bacardit and J. M. Garrell. Evolving multiple discretizations with adaptive intervals for a Pittsburgh Rule-Based Learning Classifier System. *Genetic and Evolutionary Computation Conference - GECCO 2003*. Lecture Notes in Computer Science 2724, pp. 1818–1831, Springer-Verlag, 2003.

[3] C. L. Blake and C. J. Merz. UCI Repository of machine learning databases [http://www.ics.uci.edu/ mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science, 1998.

[4] A. L. Blum and P. Langley. Selection of Relevant Features and Examples in Machine Learning. Artificial Intelligence on Relevance, pp. 245–271, 1997.

[5] E. Cantú-Paz. Feature Subset Selection, Class Separability, and Genetic Algorithms *Genetic and Evolutionary Computation Conference - GECCO 2004* pp. 959 - 970, Seattle, WA, USA, 2004.

[6] K. A. DeJong, W. M. Spears and D. F. Gordon. Using genetic algorithms for concept learning. *Machine Learning*, 1(13):161–188, 1993.

[7] F. Divina and E. Marchiori, Evolutionary Concept Learning, *Genetic and Evolutionary Computation Conference - GECCO 2002* , pp. 343–350, W.B. Langdon et al. eds, Morgan Kaufmann, NY, USA, 2002.

[8] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. *Proceedings of the International Conference on Information and Knowledge Management*,pp. 148–155, 1998.

[9] R. Giráldez, J. S. Aguilar-Ruiz and J. C. Riquelme. Natural Coding: A More Efficient Representation for Evolutionary Learning. *Genetic and Evolutionary Computation Conference - GECCO 2003*, pp. 279–290. Chicago, USA, 2003.

[10] M. A. Hall, G.Holmes Benchmarking Attribute Selection Techniques for Discrete Class Data Mining. *IEEE Transactions on Knowledge and Data Engineering*, 15(3):1437–1447, 2003.

[11] C. Z. Janikow. A knowledge-intensive genetic algorithm for supervised learning. *Machine Learning*, 1(13):169–228, 1993.

[12] H. Liu and R. Setiono. Chi2: Feature Selection and Discretization of Numeric Attributes *Proceedings of the 7th International Conference on Tools with Artificial Intelligence*, Washington D.C., pp.388-391, 1995.

[13] H. Liu and H. Motoda *Feature Selection for Knowledge Discovery and Data Mining.* Kluwer Academic, 1998.

[14] M.Robnik-Sikonja Theoretical and Empirical Analysis of ReliefF and RReliefF. *Machine Learning Journal*, 53:23-69, 2003.

[15] H. Vafaie and K.A. De Jong. Robust feature selection algorithms. *Proceedings of the International Conference on Tools with Artificial Intelligence*, pp 356-364, IEEE Computer Society Press, 1993.

[16] G. Venturini, SIA: a supervised inductive algorithm with genetic search for learning attributes based concepts, in: *European Conference on Machine Learning*, pp. 281–296, Pavel Brazdil ed, LNCS 667, Springer, Vienna, Austria, 1993.

[17] J. Yang and V. Honavar. Feature Subset Selection Using A Genetic Algorithm. *Genetic Programming 1997: Proceedings of the Second Annual Conference*, Morgan Kaufmann, 1997.