A NOVEL FEATURE RANKING ALGORITHM

Salim CHIKHI Sadek BENHAMMADA
Department of Computer Science,
Mentouri University, Constantine, Algeria
E-mail: s.chikhi@arn.dz

Abstract

The estimation of the quality of attributes is an important issue in machine learning and data mining. There are several important tasks in the process of machine learning like feature subset selection, constructive induction, and decision tree building, which contain the attribute estimation procedure as their principal component. Relief algorithms are successful attribute estimators. They are able to detect conditional dependencies between attributes and provide a unified view on the attribute estimation. They have commonly been viewed as feature subset selection methods that are applied in pre-processing step before a model is learned. In this paper, we propose a variant of ReliefF algorithm: ReliefMSS. We analyse the ReliefMSS parameters and compare ReliefF and ReliefMSS performances as regards the number of iterations, the number of random attributes, the noise effect, the number of nearest neighbours and the number of examples presented. We find that for the most of these parameters, ReliefMSS is better than ReliefF.

Keywords: *feature selection, Relief algorithms, number of nearest neighbours.*

1- INTRODUCTION

Many factors affect the success of machine learning on a given task. The representation and quality of the example data is first and foremost. Theoretically, having more features should result in more discriminating power. However, practical experience with machine learning algorithms has shown that this is not always the case [5]. Not all the available features are pertinent. It is possible that some correspond to the noise. They can also bring few information, be correlated or even useless to the system for the fulfilment of its task. According to Jain [6], the performance of a classification system depends strongly on relations between the number of used samples, the number of considered features and the system complexity. Indeed, to obtain a powerful system it is necessary to have an enough great number of samples which represent well the different phenomena to model, so as to estimate correctly its parameters. Moreover, an exponential relation binds the number of samples to the number of characteristics used by the system: for the addition of a new characteristic, the number of samples must be increased in an exponential way. These different remarks show clearly that it is necessary to restrict the number of considered features, during a system construction, so as to optimise its performances. By this fact, the attribute selection is an active research domain since many decades. A lot of works and publications deal with these techniques which are applied to many areas [4]. The ReliefF algorithm is one of these techniques; it is a successful attribute estimator. Unlike the majority of the heuristic measures for estimating the quality of the attributes, the Relief algorithms do not make the assumption of conditional independence between attributes. They are able to detect conditional dependencies between attributes and provide a unified view on the attribute estimation. In addition, their quality estimates have a natural interpretation. While they have commonly been viewed as feature subset selection methods that are applied in prepossessing step before a model is learned, they have actually been used successfully in a variety of settings, e.g., to select splits or to guide constructive induction in the building phase of decision tree learning, as the attribute weighting method and also in the inductive logic programming [10]. Fu and Wang have carried out a data dimensionality reduction and a rule extraction techniques based on a novel separability-correlation measure for ranking the importance of attributes [3]. Other researchers have used the ReliefF algorithm to select genes for cancer classification [13], to select feature for image classification [2], to select features from the multi-wavelength data [14], and as attribute estimation measure for building classification trees in the data analysis of a controlled clinical study of the chronic wound healing acceleration as a result of electrical stimulation [11]. Original Relief as developed by Kira and Rendell [7] can deal with discrete and continuous attributes and is limited to only two-class problems. This basic algorithm was

quickly extended by Kononenko [8]. His ReliefF algorithm is more robust than the original because it selects a set of nearby hits and a set of nearby misses for every target sample and averages their distances. Many extensions of ReliefF have been proposed to deal with noisy, incomplete, and multi-class data sets [1] [12]. In this paper, we present the ReliefMSS algorithm, which is a new variant of the ReliefF algorithm. We examine and analyse the parameters, the robustness regarding to the number of examples, the number of iterations, the noise, the number of attributes and the number of the nearest neighbours concerning the proposed approach.

2. RELIEF ALGORITHM

A key idea of the original Relief algorithm [7], given in Figure 1, is to estimate the quality of attributes according to how well their values distinguish between instances that are near to each other. For that purpose, given a randomly selected instance *Ri* (line 3), Relief searches for its two nearest neighbours: one from the same class, called *nearest hit H*, and the other from the different class, called *nearest miss M* (line 4). It updates the quality estimation *W*[*A*] for all attributes *A* depending on their values for *Ri*, *M*, and *H* (lines 5 and 6). If instances *Ri* and *H* have different values of the attribute *A* then the attribute *A* separates two instances with the same class which is not desirable so we decrease the quality estimation *W*[*A*]. On the other hand if instances *Ri* and *M* have different values of the attribute *A* then the attribute *A* separates two instances with different class values which is desirable so we increase the quality estimation *W*[*A*]. The whole process is repeated for *m* times, where *m* is a user-defined parameter.

Input: for each training instance a vector of attribute values and the class value
Output: the vector W of estimations of the qualities of attributes

1. *set all weights W*[*A*] *:= 0:0;*
2. **for** *i := 1* **to** *m* **do** *begin*
3. *randomly select an instance Ri;*
4. *find nearest hit H and nearest miss M;*
5. **for** *A := 1* **to** *a* **do**
6. *W*[*A*]*:=W*[*A*]*-diff(A,Ri,H)/m+diff(A,Ri,M)/m*
7. **end**

Figure 1: Pseudo code of Relief algorithm.

Function diff(A,I1,I2) calculates the difference between the values of the attribute A for two instances I1 and I2.

For nominal attributes it was originally defined as:

$$diff\ (A, I_1, I_2) = \begin{cases} 0 : value\ (A, I_1) = value\ (A, I_2) \\ 1 : otherwise \end{cases} \quad (1)$$

and for numerical attributes as:

$$diff\ (x_i, d, hits_j) = \frac{|value\ (A, I_1) - value\ (A, I2)|}{\max(A) - \min(A)} \quad (2)$$

Max and Min designate respectively the maximal and the minimal value that attribute A can hold over the data set. The function *diff* is used also for calculating the distance between instances to find the nearest neighbours. The original Relief can deal with nominal and numerical attributes. However, it cannot deal with incomplete data and is limited to two-class problems. Its extension, which solves these and other problems, is called ReliefF [8].

3. RELIEF-F ALGORITHM

The ReliefF (Relief-F) algorithm [8] (see Figure 2) is not limited to two class problems, is more robust and can deal with incomplete and noisy data. Similarly to Relief, ReliefF randomly selects an instance *Ri* (line 3), but then searches for *k* of its nearest neighbours from the same class, called nearest hits H_j (line 4), and also *k* nearest neighbours from each of the different classes, called nearest misses $M_j(C)$ (lines 5 and 6). It updates the quality estimation *W*[*A*] for all attributes *A* depending on their values for *Ri*, hits H_j and misses $M_j(C)$ (lines 7, 8 and 9). The update formula is similar to that of Relief (lines 5 and 6 on Figure 1), except that we average the contribution of all the hits and all the misses. The contribution for each class of the misses is weighted with the prior probability of that class *P*(*C*) (estimated from the training set). Since we want the contributions of hits and misses in each step to be in [0,1] and also symmetric (we explain reasons for that below) we have to ensure that misses' probability weights sum to 1. As the class of hits is missing in the sum we have to divide each probability weight with factor 1-$P(class(R_i))$ (which represents the sum of probabilities for the misses' classes). The process is repeated for *m* times. Selection of *k* hits and misses is the basic difference to Relief and ensures greater robustness of the algorithm concerning noise. User-defined parameter *k* controls the locality of the estimates.

Input: for each training instance a vector of attribute values and the class value
Output: the vector W of estimations of the qualities of attributes
1. *set all weights W[A] := 0.0;*
2. **for** *i := 1* **to** m **do begin**
3. *randomly select an instance Ri;*
4. *find k nearest hits Hj;*
5. **for** *each class C ≠classe(Ri)* **do**
6. *from class C find k nearest misses Mj(C);*
7. **for** *A := 1* **to** *a* **do**
8. $$W[A] := W[A] - \sum_{j=1}^{k} \frac{diff(A, R_i, H_j)}{(m.k)} + \sum_{c \neq class(R_i)} \frac{P(c)}{1 - P(class(R_i))} \sum_{j=1}^{k} \frac{diff(A, R_i, M_j(C))}{(m.k)}$$
9. **end**

Figure 2: Pseudo code of ReliefF algorithm.

4. RELIEFMSS: THE PROPOSED VARIANT

In the same way as ReliefF, ReliefMSS selects randomly an instance R_i (Figure 3, line 3), and research the k nearest neighbours of the same class as R_i (H_j) (Figure 3, line 4,), and also the *k* nearest neighbours in each of the other classes, $M_j(C)$ (Figure 3, lines 5 and 6). However, the update of the weight $W[A]$ for any attribute A is not only made by its values for the instances R_i, H_j and $M_j(C)$, but also by the values of the other attributes, as explained in what follows :

– If the difference between the values of the attribute A for the selected instance R_i and a nearest neighbour of the same class H_j : $diff(A, R_i, H_j)$ (equations (1) and (2)) is greater than the values differences average of the other attributes for these two instances $DM(A, R_i, H_j)$ (equation (3)), then the attribute A will undergo a decrease of its quality according to the distance between $diff(A, R_i, H_j)$ and $DM(A, R_i, H_j)$ (figure 3, line 8).

– If the difference between the values of the attribute A for the selected instance R_i and a nearest neighbour of another class $M_j(C)$: $diff(A, R_i, M_j(C))$ (equations (1) and (2)) is greater than the values differences average of the other attributes for these two instances $DM(A,Ri,Mj(C))$, then the attribute A will undergo an increase of its quality according to the distance between $diff(A,R_i,H_j)$ and $DM(A,R_i,H_j)$ (equation (3)) and figure 3, line 9).

– In other cases, we do not make any change.

The function $DM(A_j,I_1,I_2)$ calculates the values differences average of all attributes except attribute A for the two instances I_1 *and* I_2. It is defined by:

$$DM(A_j, I_1, I_2) = \frac{1}{a-1} \sum_{i \neq j}^{i=1,a} diff(A_i, I_1, I_2) \qquad (3)$$

For the legibility of the algorithm we put**:**

- $diff(A,R_i,H_j)=diffH$
- $diff(A,R_i,M_j(C))=diffM$
- $DM(A,R_i,H_j)=DMH$
- $DM(A,R_i,M_j(C))=DMM$

Input: for each training instance a vector of attribute values and the class value
Output: the vector W of estimations of the qualities of Attributes
1. *set all weightsW[A] := 0.0;*
2. **for** *i := 1* **to** *m* **do begin**
3. *randomly select an instance Ri;*
4. *find k nearest hits Hj;*
5. **for** *each class C ≠classe(Ri)* **do**
6. *from class C find k nearest misses Mj(C);*
7. **for** *A := 1* **to** *a* **do**
8. $$W[A] := W[A] - \sum_{diffH > DMH}^{j=1,k} \frac{diffH - DMH}{(m.k)} + \sum_{c \neq class(R_i)} \frac{P(c)}{1 - P(class(R_i))} \sum_{diffM > DMM}^{j=1,k} \frac{diffM - DMM}{(m.k)}$$
9. **end**

Figure 3: Pseudo code of ReliefMSS.

Intuitively, the principle of the proposed variant means what follows.

For the negative updates, the condition $diffH>DMH$ and the value of DMH (figure 3, line 8) allow to diminish the negative update of the attributes which taking a close values for the instances from the same class (important attributes).

For the positive updates, the condition $diffM>DMM$ and the value of DMM (figure 3, line 8) allow to diminish the positive updates of the attributes which taking a close values for the instances from different classes (insignificant attributes).

4.1 PARAMETER ANALYSIS

In this section we consider different ReliefMSS parameters: the number of nearest neighbours to be used, the distance and the number of iterations.

4.1.1 THE NUMBER OF NEAREST NEIGHBOURS

To study the ReliefMSS algorithm behaviour as regards the number of nearest neighbours (nnn), we generated 200 instances of the parity problem: $(A_1 \oplus A_2)$ [11], with 10 random attributes, In a way that each of 2 classes contains exactly 100 instances (so that we can vary the nnn of 1 until 99). We observed the estimations of the attributes according to nnn. Figures 4 and 5 show respectively ReliefF and ReliefMSS estimations according to the considered nnn. We notice that

the estimations of both algorithms of one of the informative attributes become more and more short-sighted with the growth of nnn, however ReliefMSS discerns between the informative attribute and the insignificant attribute even for 99 nearest neighbours, differently from the ReliefF algorithm whose estimation of the informative attribute becomes finally imperceptible of that of the insignificant attribute.

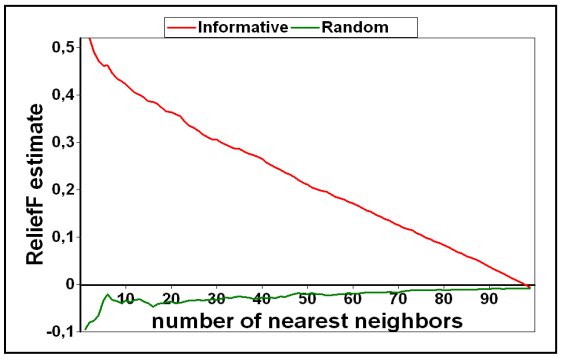

Figure 4: ReliefF estimates according to the number of nearest neighbours on $(A_1 \oplus A_2)$ problem.

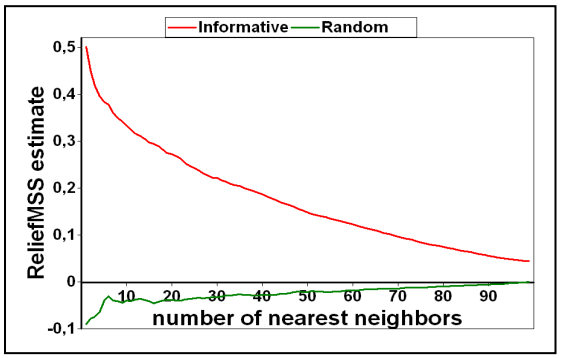

Figure 5: ReliefMSS estimates according to the number of nearest neighbours on $(A_1 \oplus A_2)$ problem.

We also examine the ReliefMSS estimations according to nnn in the Boolean problem defined by:

$$C = (A_1 \oplus A_2) \vee (A_3 \wedge A_4) \qquad (4)$$

We generate iteratively 200 instances in such way as every class contains exactly 100 instances. Figures 6 and 7 show the results obtained for ReliefF and ReliefMSS respectively.

We know that A1 and A2 are more important than A3 and A4 for the determination of the class. ReliefMSS recognizes this even for 99 nearest neighbours (the greatest possible number of nearest neighbours); on the other hand the algorithm ReliefF recognizes this to only 80 nearest neighbours.

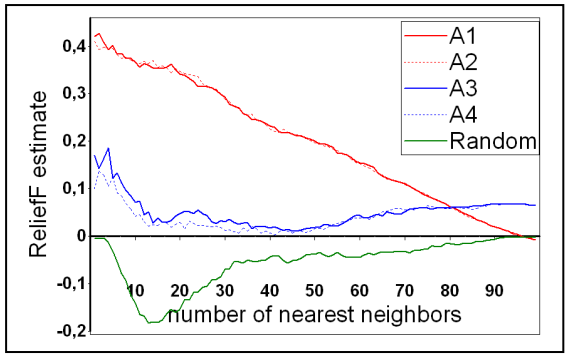

Figure 6: ReliefF estimates according to the number of nearest neighbours on problem defined by equation (4).

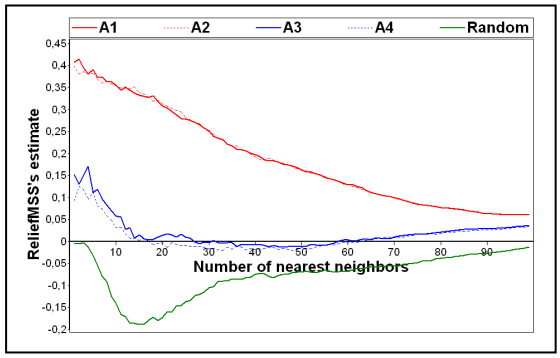

Figure 7: ReliefMSS estimates according to the number of nearest neighbours on problem defined by equation (4).

4.1.2 TAKING THE DISTANCES INTO ACCOUNT

When we take the nearest neighbours we reduce the risk of the following pathological case:
we have a large number of instances and a mix of nominal and numerical attributes where numerical attributes prevail; it is possible that all the nearest neighbours are closer than 1 so that there are no nearest neighbours with differences in values of a certain nominal attribute. If this happens in a large part of the problem space this attribute gets zero weight (or at least small and unreliable one) [10]. However, the use of a large nnn causes a degradation of the weights assigned to the important attributes (figure 5 and 7). By using fewer instances, less iterations, complex problems or noised data, the algorithm becomes less robust regarding to the nnn, and we face the risk of having incompatible estimations with the importance of the attributes.
In the example of figure 7 we selected all the instances (m=n), which makes it possible to obtain relatively stable weights, figure 8 represents the weights of the attributes according to the nnn for the same problem (equation 4), such as the number of iterations is m=30.

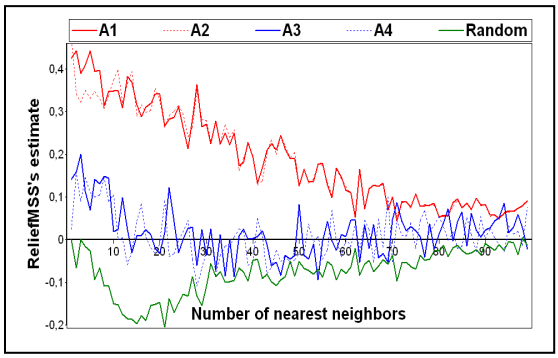

Figure 8: ReliefMSS estimates according to the number of nearest neighbours on problem defined by equation (4) (m=30).

We note that the weights assigned to the attributes become unstable; this increases the risk to have weights incompatible with the importance of the attributes with the growth of nnn.

One of the solutions to this problem is a taking the distances into account [10]. It consists in taking a large nnn by giving more impact to the nearest instances. The estimate of their impact is then inversely proportional to their distances of the point in question. ReliefMSS can be adjusted to taking the distances into account by changing the way of the weights update (figure 3, line 8 and 9). Figure 9 represents the ReliefMSS estimations considering the distance for the same problem (equation 4) with 200 instances, 30 iterations, and the distance factor $\sigma = 5$.

$$W[A] := W[A] - \frac{1}{m} \sum_{diffH>DMH}^{j=1,k} (diffH - DMH).d(R_i, H_j)$$

$$+ \frac{1}{m} \sum_{c \neq class(R_i)} \frac{P(c)}{1 - P(class(R_i))} \sum_{diffM>DMM}^{j=1,k} (diffM - DMM).d(R_i, M_j(C)) \quad (5)$$

The distance factor of two instances $d(I_1,I_2)$ is defined by :

$$d(i,j) = \frac{d_1(i,j)}{\sum_{l=1}^{k} d_1(i,l)} \quad (6)$$

Such as :

$$d_1(i,j) = e^{\frac{-rank(R_i,I_j)}{\sigma}} \quad (7)$$

By comparing figure 8 with figure 9, we notice that the consideration of the distance allowed strengthening the robustness of ReliefMSS with regard to nnn.

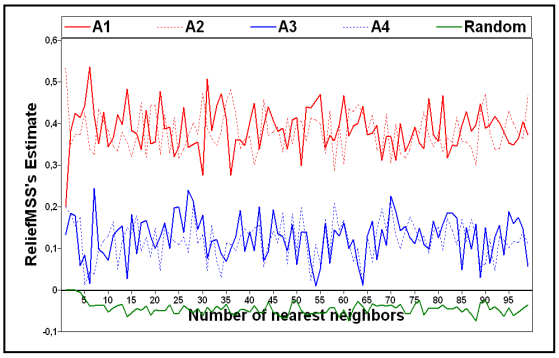

Figure 9: ReliefMSS estimates considering the distances according to the number of nearest neighbours on problem defined by equation (4) (m=30, $\sigma = 5$).

4.1.3 THE NUMBER OF ITERATIONS

Figure 10 shows the estimates of ReliefMSS according to the number of iterations, for the Boolean problem defined by equation 4. We note that with a small number of iterations the weights assigned to the attributes are not stable any more. For example, the quality difference between the attribute A3 and the random attribute is not resolved until around 71 iterations, the quality difference between the attributes A1 and A3 is not resolved until around 84 iterations. With the growth of the number of iterations, the weights assigned to the attributes approach the stable values more and more.

The random selection of *m* instances (figure3, line 2) in a space of the unknown problem can be in favour of the non-representative instances rather than the most representative ones.

If the dataset is reasonably large, the solution consists in selecting all the instances. If we have a very large data set, the selection of all the instances is not possible because of the increasing of calculations complexity. In this case, the solution is the use of the selective sampling approach which attempts to select only representative instances with a high probability to be informative in determining feature relevance [9].

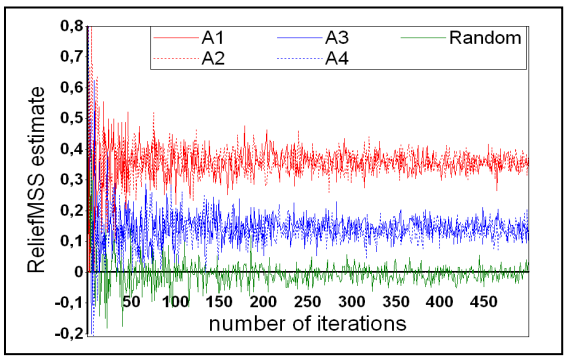

Figure 10: ReliefMSS estimates according to the number of iterations on problem defined by equation (4).

4.2 PERFORMANCE ANALYSIS

In this section we compare ReliefMSS and ReliefF performances as regards the number of examples and iterations which we need for a reliable estimation, the robustness concerning the noise, the number of random attributes and the number of nearest neighbours (nnn). We use artificial datasets because we want to control the environment: in real-world datasets we do not fully understand the problem and the relation of the attributes to the target variable. Therefore we do not know what a correct output of the feature estimation should be and we cannot evaluate the quality estimates of the algorithms. We use the following measures defined in [10]:

Separability *is* the difference between the lowest estimate of the important attributes and the highest estimate of the unimportant attributes.

$$s = W_{Iworst} - W_{Rbest} \qquad (8)$$

We say that a heuristics is successful in separating between the important and unimportant attributes if $s > 0$.

Usability u is the difference between the highest estimates of the important and unimportant attributes.

$$u = W_{Ibest} - W_{Rbest} \qquad (9)$$

4.2.1 NUMBER OF EXAMPLES

To examine the influence of the number of examples available on the estimates of ReliefMSS, we applied the algorithm to the various artificial problems described in [10], the objective being the search for the minimal necessary number of examples so that the separability (usability) is positive. Thus for each problem we varied the number of examples, and for each number of examples we repeated the calculation of the weights of the attributes and their values of S and U 100 times, until the acquisition of the number of examples allowing to have a separability (usability) greater than in 99 experiments at least.
In table 1, the two right-hand side columns represent the numbers of examples necessary for ReliefF, while the two left-hands columns represent the example numbers necessary for ReliefMSS. We note that both algorithms require a small number of examples to separate the important attributes from the insignificant ones (separability) and a smaller number to separate an important attribute from the unimportant ones (usability). The number of necessary examples increases with the growth of the problem difficulty (variation). ReliefMSS requires fewer examples than ReliefF on 4 problems for the separability and 4 problems for the usability and requires more examples on 2 problems for the separability and 2 problems for the usability. However, the difference with the number of necessary examples for the two algorithms remains light for all the problems.

	ReliefMSS		ReliefF	
Problem	*S*	*U*	*S*	*U*
Simple-Bool	170	42	165	42
Modulo-2-2-c	38	31	40	35
Modulo-2-3-c	120	77	141	78
Modulo-2-4-c	305	240	305	240
Modulo-5-2-c	95	77	104	100
Modulo-5-3-c	547	492	545	508
Modulo-10-2-c	245	202	202	198
MONK1	61	49	73	46

Table 1: Results of varying the number of examples.

4.2.2 NOISE PERCENTAGE

To compare the robustness of ReliefMSS and ReliefF concerning the noise, we consider the same previous problems, the objective being to find the maximum noise percentage that the algorithms can support. Thus for each problem we varied the noise percentage by changing some percentage of classes values by random values. For each noise percentage, we repeated the calculation of the attribute weights and their *S* and *U* values 100 times, until obtaining the maximal noise percentage to maintain a positive separability (usability) in 99 experiments at least. The number of instances used for each problem is a number sufficient for the two algorithms. For example, the Modulo-5-3-c problem requires 545 instances for ReliefF and 547 instances for ReliefMSS (columns *S* in Table1) whereas the number of instances used is 550.
In the results shown in table 2, the #Ex column represents the number of instances used. We can note that ReliefMSS and ReliefF show a noise tolerance even with the minimal number of instances necessary to separate the important attributes from the unimportant ones. The ReliefMSS noise tolerance is equivalent to that of ReliefF for the majority of problems. However, it presents more tolerance for the problems Modulo-5-2-C and Modulo-5-3-C.

		ReliefMSS		ReliefF	
Problem	# Ex	*s*	*u*	*s*	*u*
Bool-Simple	170	5	70	5	75
Modulo-2-2-c	45	2	20	2	15
Modulo-2-3-c	145	10	25	10	25
Modulo-2-4-c	305	30	25	30	25
Modulo-5-2-c	105	45	50	15	35
Modulo-5-3-c	550	15	30	10	20
Modulo-10-2-c	245	20	20	20	20
MONK1	75	15	35	15	35

Table 2: Results of varying the noisy prediction value.

Table 3 shows the noise percentages for ReliefMSS and ReliefF, where the number of examples used for each problem is twice the number of examples used in Table 2. We note that the ReliefMSS noise tolerance increases considerably in all the problems.

		ReliefMSS		ReliefF	
Problem	# Ex x 2	*s*	*u*	*s*	*u*
Bool-Simple	340	15	75	15	75
Modulo-2-2-c	90	35	40	35	40
Modulo-2-3-c	290	50	55	50	55
Modulo-2-4-c	610	40	65	40	65
Modulo-5-2-c	210	30	45	30	40
Modulo-5-3-c	1090	50	65	45	65
Modulo-10-2-c	490	45	50	50	55
MONK1	150	40	65	50	65

Table 3: Results of varying the noisy prediction value using twice the number of examples used in table 2.

4.2.3 NUMBER OF ITERATIONS

To examine the influence of the iteration number on the ReliefMSS estimates, we also consider the same preceding problems, the objective being to find a minimal number of iterations so that the separability (usability) remains positive. Thus for each problem we varied the number of iterations, and for each number of iterations we repeated the calculation of the attributes weights and their values of S and U 1000 times, until the acquisition of the minimal number of iterations allowing to have a separability (usability) positive in 999 experiments at least.
table 4 shows the obtained results, we note that ReliefMSS requires less iteration than ReliefF for the majority of the problems. We repeated the experiment by multiplying the examples number used in Table 4 by 2.

		ReliefMSS		ReliefF	
Problem	# Ex	*s*	*u*	*s*	*u*
Bool-Simple	170	147	24	150	25
Modulo-2-2-c	45	15	15	20	15
Modulo-2-3-c	145	56	10	77	10
Modulo-2-4-c	305	173	98	231	105
Modulo-5-2-c	105	58	35	96	68
Modulo-5-3-c	550	223	97	444	206
Modulo-10-2-c	245	56	41	25	47
MONK1	75	45	4	63	4

Table 4: Results of varying the number of iterations.

Table5 shows the obtained results, the necessary number of iterations decreases considerably in all the problems.

		ReliefMSS		ReliefF	
Problem	# Ex x 2	*s*	*u*	*s*	*u*
Bool-Simple	340	7	2	7	2
Modulo-2-2-c	90	10	5	10	5
Modulo-2-3-c	290	13	2	16	4
Modulo-2-4-c	610	79	28	98	29
Modulo-5-2-c	210	16	13	17	16
Modulo-5-3-c	1100	109	23	201	25
Modulo-10-2-c	490	21	20	17	17
MONK1	150	32	3	28	3

Table 5: Results of varying the number of iterations using twice the number of examples used in table 4.

4.2.4. NUMBER OF RANDOM ATTRIBUTES

To examine the influence of the number of random attributes on the estimates of ReliefMSS, we used the same previous problems except Monk1 problem (the number of attributes in its data sets is fixed). The objective is to find the maximal number of random attributes allowing separability (utility) to be preserved as positive. Thus for every problem we varied the number of random attributes, and for every number of random attributes we repeated the calculation of the attribute weights and their *s* and *u* values 100 times, until obtaining the maximal number of random attributes allowing to preserve a separability (utility) positive in at least 99 experiments.
Table 6 shows the obtained results, We notice that we can give only a moderate number of random attributes to separate all the important attributes from the insignificant attributes (separability). This number is bigger to separate an important attribute from the unimportant attributes (usability). ReliefMSS and ReliefF support the same number of random attributes for the majority of the problems.

		ReliefMSS		ReliefF	
Problem	# Ex	*s*	*u*	*s*	*u*
Bool-Simple	170	14	180	14	180
Modulo-2-2-c	45	16	25	16	25
Modulo-2-3-c	145	12	17	13	17
Modulo-2-4-c	305	13	13	13	15
Modulo-5-2-c	105	10	22	22	23
Modulo-5-3-c	550	12	25	12	26
Modulo-10-2-c	245	21	21	14	21

Table 6: Results of varying the number of random attributes.

4.2.5 NUMBER OF THE NEAREST NEIGHBORS (NNN)

To examine the robustness of ReliefMSS to nnn, we also used the same preceding problems. The objective is to find the maximal nnn allowing

keeping a positive separability (usability). Thus for each problem we varied nnn, and for each nnn we repeated the calculation of the attributes weights and their values of S and U 100 times, until the acquisition of the maximal nnn allowing to have a separability (usability) positive in 99 experiments at least. Table 7 shows the obtained results, symbol "-" means that the algorithm resists even with the greatest possible nnn, we notice that ReliefMSS is more robust than ReliefF for all the problems.

		ReliefMSS		ReliefF	
Problem	# Ex	*s*	*u*	*s*	*u*
Bool-Simple	170	29	-	34	-
Modulo-2-2-c	45	16	18	21	-
Modulo-2-3-c	145	13	20	16	22
Modulo-2-4-c	305	13	17	16	19
Modulo-5-2-c	105	11	14	-	-
Modulo-5-3-c	550	11	14	16	21
Modulo-10-2-c	245	14	15	15	18
MONK1	75	10	-	13	-

Table 7: Results of varying the number of nearest neighnours

5. CONCLUSION

To update an attribute weight, ReliefF uses only the differences between the values of this attribute for the selected instance and the nearest neighbours of this instance. The attribute will undergo a reduction of its quality if there is a difference between its values for the selected instance and a nearest neighbour of the same class. It will obtain an increase of its quality if there is a difference between its values for the selected instance and a nearest neighbour of another class. In this paper we propose a ReliefMSS algorithm which is a variant of ReliefF. To update an attribute weights, we introduce the average of the differences of the values of other attributes. the attribute will undergo a reduction of its quality if the difference between its values for the selected instance and a nearest neighbour of the same class is greater than the values differences average of the other attributes for these two instances. It will obtain an increase in its quality if the difference between its values for the selected instance and a nearest neighbour of another class is greater than the values differences average of the other attributes. A comparison of performances was carried out by using artificial data. ReliefMSS presented better performances than ReliefF in particular with regard to the number of nearest neighbours and the number of iterations. The obtained results we encourage to the application of ReliefMSS by using real data

References

[1]Bins J. and Draper B. "Feature selection from huge feature sets", *In Proc. 8th Intern. Conf. on Computer vision*, Vancouver, Canada, pp. 159-165, 2001.

[2]Campedel M. and Moulines E. "Classification and texture feature selection", *Artificial Intelligence journal*, ENST, France.Vol 0/2004 pp 1-25, 2004.

[3]Fu X. and Wang L. "Data dimensionality reduction with application to improving classification performance and explaining concepts of data sets", *Int. J. Business Intelligence and Data Mining*, Vol. 1, No. 1, pp.65–87, 2005.

[4]Grandidier F., "A new feature selection algorithm-Application to automatic reading of cursive handwriting", *PhD Thesis,* High Technology School, Montreal University, January, 2003.

[5]Hall M. A. "Correlation-based Feature Selection for Machine Learning", *PhD Thesis,* New Zeland Univ, April, 1999.

[6]Jain A.K., Duin R.P.W. and Mao J. "Statistical pattern recognition : A review". *IEEE Trans. on Pattern Analysis and Machine Recognition*, pp. 4-37, 2000.

[7]Kira K. and Rendell L. A. "A practical approach to feature selection", *In Proceedings of the Ninth Int. Workshop on Machine Learning,* pp. 249–256, 1992.

[8]Kononenko I. "Estimating attributes: analysis and extensions of Relief", *Machine Learning: ECML-94*. Springer Verlag, pp. 171–182, 1994

[9]Liu H., Motoda H. and Yu L. "A Selective Sampling Approach to Active Feature Selection", *in the proceedings of the 19th International Conference on Machine learning,* Arizona State University, 2002.

[10]Robnik-Sikonja M. and Kononenko I. "Theoretical and empirical analysis of Relief and ReliefF", *Machine Learning*, 53, pp:23–69, 2003.

[11]Robnik-Šikonja M, Cukjati D, and Kononenko I. "Comprehensible evaluation of prognostic factors and prediction of wound healing" *Artificial Intelligence in Medicine* N°29, pp 25-38, 2003.

[12]Sun Y. and Li J. "Iterative RELIEF for feature weighting", *in Proceedings of the 23rd international conference on Machine learning* Pittsburgh, Pennsylvania pp. 913 – 920, 2006

[13]Wang Y., Makedon F. "Application of ReliefF Feature Filtering Algorithm to Selecting Informative Genes for Cancer Classification using Microarray Data", *in Proceedings of the IEEE Computational Systems Bioinformatics Conference.* Stanford, California, pp.477-478, 2004.

[14]Zhang Y., Luo A. and Zhao Y. "An automated classification algorithm for multi-wavelength data", *In Proceedings of the SPIE. National Astronomical Observatories*, Chinese Academy of Sciences, China. Volume 5493, pp. 483-490, June 2004.