Non-myopic attribute estimation in regression

Marko Robnik-Šikonja, Igor Kononenko
University of Ljubljana,
Faculty of Computer and Information Science,
Tržaška 25, 61001 Ljubljana, Slovenia,
tel.: +386-61-1768386, fax: +386-61-1768386,
e-mail: {Marko.Robnik, Igor.Kononenko}@fer.uni-lj.si

Abstract

One of key issues in both discrete and continuous class prediction and in machine learning in general seems to be the problem of estimating the quality of attributes. Heuristic measures mostly assume independence of attributes and therefore cannot be successfully used in domains with strong dependencies between attributes. Relief and its extension ReliefF are statistical methods capable of correctly estimating the quality of attributes in classification problems with strong dependencies between attributes. Following the analysis of ReliefF we have extended it to continuous class problems. Regressional ReliefF (RReliefF) and ReliefF provide a unified view on estimation of quality of attributes. The experiments show that RReliefF successfully estimates the quality of attributes and can be used for non-myopic learning of regression trees.

KEYWORDS: regression, attribute estimation, regression trees

Abstract

One of key issues in both discrete and continuous class prediction and in machine learning in general seems to be the problem of estimating the quality of attributes. Heuristic measures mostly assume independence of attributes and therefore cannot be successfully used in domains with strong dependencies between attributes. Relief and its extension ReliefF are statistical methods capable of correctly estimating the quality of attributes in classification problems with strong dependencies between attributes. Following the analysis of ReliefF we have extended it to continuous class problems. Regressional ReliefF (RReliefF) and ReliefF provide a unified view on estimation of quality of attributes. The experiments show that RReliefF successfully estimates the quality of attributes and can be used for non-myopic learning of regression trees.

1 Introduction

Regression analysis is a technique for modeling relations between independent variables (attributes) and a dependent continuous variable (class). In the context of machine learning it is also referred to as learning of continuous class.

The majority of the current propositional inductive learning systems predict discrete class. They can solve also continuous class problems by discretizing the class in advance. This approach is often inappropriate. Regression learning systems eg. CART (Breiman et al., 1984), Retis (Karalič, 1992), M5 (Quinlan, 1992), predict continuous class directly.

One of key issues in both discrete and continuous class prediction and in machine learning in general seems to be the problem of estimating the quality of attributes. Heuristic measures for estimating attributes' quality mostly assume independence of attributes (eg. information gain (Hunt et al., 1966), gini index (Breiman et al., 1984), distance measure (Mantaras, 1989) and j-measure (Smyth and Goodman, 1990) for discrete class and mean squared error (Breiman et al., 1984) for continuous class) and therefore cannot be successfully used in domains with strong dependencies between attributes.

Relief (Kira and Rendell, 1992a; Kira and Rendell, 1992b) and its extended version ReliefF (Kononenko, 1994) are capable of correctly estimating the quality of attributes in classification problems with strong dependencies between attributes. Similar approaches are contextual merit (Hong, 1994) and geometrical approach (Elomaa and Ukkonen, 1994). Following the analysis of ReliefF we have extended it to continuous class problems.

In the next Section we present and analyze the novel RReliefF (Regressional ReliefF) algorithm. Section 3 describes experiments and results, and last Section gives conclusions and guidelines for further work.

2 RReliefF

2.1 ReliefF for classification

The key idea of the original algorithm Relief is to estimate the quality of attributes according to how well their values distinguish between the instances that are near to each other. For that purpose, given an instance, Relief searches for its two nearest neighbors: one from the same class (called *nearest hit*) and the other from a different class (called *nearest miss*). The original algorithm of Relief (Kira and Rendell, 1992a; Kira and Rendell, 1992b) randomly selects m training instances, where m is the user-defined parameter. The algorithm is given in Figure 1.

```
1. set all weights W[A] := 0.0;
2. for i := 1 to m do begin
3.     randomly select an instance R;
4.     find nearest hit H and nearest miss M;
5.     for A := 1 to #all_attributes do
6.         W[A] := W[A] - diff(A,R,H)/m + diff(A,R,M)/m;
7. end;
```

Figure 1: The basic algorithm of Relief

Function $diff(Attribute, Instance1, Instance2)$ calculates the difference between the values of Attribute for two instances:

- For discrete attribute:

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

- For continuous attribute:

$$diff(A, I_1, I_2) = \frac{|value(A, I_1) - value(A, I_2)|}{max(A) - min(A)} \tag{2}$$

The function $diff$ is used also for calculating the distance between instances to find the nearest neighbors. The total distance is simply the sum of distances over all attributes. It is obvious that Relief's estimate $W[A]$ of quality of attribute A is an approximation of the following difference of probabilities:

$$W[A] = P(\text{different value of A}|\text{nearest instance from a different class})$$
$$-P(\text{different value of A}|\text{nearest instance from the same class}) \tag{3}$$

The complexity of Relief for n training instances and A attributes is $O(m \times n \times A)$. The original Relief can deal with discrete and continuous attributes. However, it can not deal with incomplete data and is limited to two-class problems. Kononenko (1994) has shown, that Relief's estimates are strongly related to impurity functions. Besides, he developed an extension called ReliefF that is, unlike original Relief, able to deal with incomplete, noisy data and with multiclass problems. One difference to original Relief, interesting also for regression, is that instead of one nearest hit and one nearest miss, ReliefF uses k nearest hits and misses and averages their contribution to $W[A]$.

2.2 RReliefF for regression

In regression problems the class is continuous, therefore the (nearest) hits and misses cannot be used. Instead of requiring the exact knowledge of whether two instances belong to the same class or not, we can introduce a kind of probability that two instances are from a different class. This probability can be modeled with the relative distance between the class values of the two instances.

Still, to estimate W[A] in equation 3, the information about the sign of each contributed term is missing. In the following derivation we reformulate equation 3, so that it can be directly evaluated using the probability of two instances belonging to the different class. If we rewrite

$$P_{diffA} = P(\text{different value of A}|\text{nearest instances}) \tag{4}$$

$$P_{diffC} = P(\text{different class}|\text{nearest instances}) \tag{5}$$

and

$$P_{diffC|diffA} = P(\text{different class}|\text{different value of A and nearest instances}) \tag{6}$$

we obtain from (3) using Bayes rule:

$$W[A] = \frac{P_{diffC|diffA}P_{diffA}}{P_{diffC}} - \frac{(1 - P_{diffC|diffA})P_{diffA}}{1 - P_{diffC}} \tag{7}$$

```
set all $N_{dC}$, $N_{dA}[A]$, $N_{dC\&dA}[A]$, $W[A]$ to 0;
for i := 1 to m do begin
    randomly select instance $R_i$;
    select k instances $I_j$ nearest to $R_i$;
    for j := 1 to k do begin
        $N_{dC} := N_{dC} + |class(R_i) - class(I_j)| * f(i,j)$;
        for A := 1 to #all_attributes do begin
            $N_{dA}[A] := N_{dA}[A] + diff(A, R_i, I_j) * f(i,j)$;
            $N_{dC\&dA}[A] := N_{dC\&dA}[A] + |class(R_i) - class(I_j)| *$
                                        $diff(A, R_i, I_j) * f(i,j)$;
        end;
    end;
end;
for A := 1 to #all_attributes do
    $W[A] := N_{dC\&dA}[A]/N_{dC}$ - $(N_{dA}[A] - N_{dC\&dA}[A])/(m - N_{dC})$;
```

Figure 2: Pseudo code of RReliefF (Regressional ReliefF)

Therefore, we can estimate $W[A]$ by approximating terms defined by equations 4, 5 and 6. This can be done by the algorithm on Figure 2.

Term $f(i,j)$ in Figure 2 is used to take into account the distance between the two instances R_i and I_j. In our experiments we used two versions of RReliefF:

RReliefF-k uses constant influence of all k nearest instances I_j for given instance R_i by $f(i,j) = 1/k$.

RReliefF-e exponentially decreases the influence of instance I_j with the distance from given instance R_i:

$$f(i,j) = \frac{f_1(i,j)}{\sum_{l=1}^{k} f_1(i,l)} \quad \text{and} \quad f_1(i,j) = e^{-\left(\frac{rank(R_i,I_j)}{\sigma}\right)^2} \tag{8}$$

where $rank(R_i, I_j)$ is the rank of instance I_j in a sequence of instances ordered by the distance from R_i and σ is the user defined parameter.

Note that the time complexity of RReliefF is the same as that of original Relief, i.e. $O(m \times n \times A)$. The most complex operation within main **for** loop is the selection of **k** nearest instances I_j, which can be done in $O(n \times A)$ steps. $O(A)$ is needed to calculate the distance between R_i and I_j while $O(n)$ is needed to build a heap (from which k nearest instances are extracted in $O(k \log n) < O(n \times A)$ steps).

3 Experiments

We have conducted two different series of experiments to show the advantage of RReliefF. Firstly we have examined the ability of RReliefF to recognize and rank important attributes, and then we have tested it in regression tree building.

We compare the estimates of RReliefF with the mean squared error as the measure of the attribute's quality (Breiman et al., 1984). This measure is standard in regression tree systems. By this criterion the best attribute is the one which minimizes the equation:

$$MSE(A) = p_L s^2(t_L) + p_R s^2(t_R), \tag{9}$$

where t_L and t_R are the subsets of cases that go left and right, respectively, by the split based on A, and p_L and p_R are the proportions of cases that go left and right. $s^2(t)$ is the variance of class values c_i of cases in the subset t :

$$s^2(t) = \frac{1}{N(t)} \sum_{i=1}^{N(t)} (c_i - \overline{c(t)})^2. \tag{10}$$

The minimum of (9) for each attribute is considered its quality estimate and is given in results bellow.

3.1 Estimating the quality of attributes

We have tested RReliefF with several artificial data sets to check its behaviour in different circumstances. We have used the following domains:

MODULO-8-2, MODULO-8-3, MODULO-8-4: each of the three domains is described by 10 attributes, value of each attribute is integer value in the range 0-7. Half of the attributes are treated as discrete and half as continuous; each continuous attribute is exact match of one of the discrete attributes. The value of the class is the sum by modulo 8 of 2, 3 and 4 attributes, respectively. Other attributes are irrelevant (random). These domains are integer generalizations of parity concept (which is the sum by modulo 2) of order 2, 3 and 4. They shall show how well RReliefF recognizes highly dependent attributes and how it ranks discrete and continuous attributes of equal importance.

FRACTION-2, FRACTION-3, FRACTION-4: each domain contains 10 continuous attributes with values from 0 to 1. The value of the class is the fractional part of the sum of 2, 3 and 4 attributes respectively, other attributes are irrelevant (random). These domains are floating point generalizations of parity concept of order 2, 3 and 4, i.e. domains with highly dependent pure continuous attributes.

PARITY-2, PARITY-3, PARITY-4: each domain consists of 10 discrete, Boolean attributes. Domains contain 2, 3 and 4 informative attributes, respectively, the others are irrelevant. The informative attributes define parity concept: if their parity bit is 0, the class value is set to a random number between 0 and 0.5, otherwise the class value is randomly chosen to be between 0.5 and 1. These three concepts present blurred versions of parity concept (of orders 2, 3 and 4).

LINEAR: the domain is described by 10 continuous attributes with values chosen randomly between 0 and 1; the class value is computed by the following linear formula: $C = A_1 - 2A_2 + 3A_3 - 3A_4$. Other attributes are irrelevant. We have included this domain to compare the performance of RReliefF with that of the mean squared error, which is known to successfully recognize linear dependencies.

COSINUS: this domain has 10 continuous attributes with values from 0 to 1; the class value is computed as follows: $C = (-2A_2 + 3A_3)\cos(4\pi A_1)$. Seven attributes are irrelevant. We have included this non-linear dependency to compare the performance of RReliefF and the mean squared error.

For each domain we have generated 1000 examples. In each trial we ran the estimation algorithms on randomly selected 70% of the examples and repeated the process for 30 times. With this we collected enough data to eliminate any probabilistic effect caused by random selection of instances in RReliefF. We were also capable to evaluate the differences between the estimators. In the tables bellow we give averages for 30 trials.

To investigate the behaviour of the estimators with fewer examples, we have repeated the experiments with 500 and 100 examples.

In all experiments the algorithms were run with the same default set of parameters (constant m in RReliefF's main loop = 250, k-nearest for RReliefF-k = 10, k-nearest for RReliefF-e = 200, $\sigma = 30$ (see equation 8)).

In the results bellow (Tables 1 and 2) we have included calculations for the best and the worst estimated important attribute (I_{best} and I_{worst}) and also for the best estimated random attribute (R_{best}). Where appropriate we give evaluation for discrete (D) and continuous (C) attributes. The difference between I_{best} and I_{worst} shows the fluctuations in estimation, and the difference between I_{worst} and R_{best} shows if recognition of important attributes is adequate.

When interpreting the results one should not compare the absolute values of different estimators, but rather compare ranking of attributes. It is also unreliable to compare the absolute values of estimations in different domains. *Note that both RReliefF variants give higher scores to better attributes, while the mean squared error does the opposite.* In all the tables bellow we have also included the indicator of the estimator's success. We declare estimator successful if the estimation of the quality of worst important attribute is better than the estimation of quality of the best random attribute.

Table 1: Estimates for MODULO-8-* domains from 1000 examples.

Estimator	$I_{D,best}$	$I_{D,worst}$	$R_{D,best}$	$I_{C,best}$	$I_{C,worst}$	$R_{C,best}$	?
MODULO-8-2							
RReliefF-e	0.185	0.181	-0.078	0.089	0.088	-0.032	$\surd$
RReliefF-k	0.226	0.217	-0.132	0.103	0.099	-0.048	$\surd$
MSE	5.109	5.119	5.090	5.013	5.031	5.021	$\times$
MODULO-8-3							
RReliefF-e	0.048	0.033	-0.046	0.031	0.025	-0.023	$\surd$
RReliefF-k	0.066	0.052	-0.089	0.036	0.032	-0.033	$\surd$
MSE	5.329	5.357	5.319	5.263	5.319	5.397	$\times$
MODULO-8-4							
RReliefF-e	0.006	-0.006	0.013	0.008	0.004	0.002	$\times$
RReliefF-k	0.005	-0.014	0.016	0.008	0.002	0.005	$\times$
MSE	5.265	5.301	5.271	5.217	5.304	5.303	$\times$

Results for MODULO 8 addition with 1000 examples are given in Table 1. Both RReliefF variants succeeded with the sum of 2 and 3 attributes and failed with the sum of 4 attributes. We can observe that the values of the important attributes for each RReliefF estimator are decreasing with increasing complexity of the problem. With the sum of 4 attributes the difference between important and random attributes diminished, so it seems that 1000 examples is not enough for a problem of such complexity. This claim was confirmed by additional experiments with 8000 examples where RReliefF succeeded. Complexity of the problem grows exponentially: the number of peaks in the instance space for MODULO-m-p domain is m^p. As expected, the mean squared error failed in all MODULO-8 domains.

It is also interesting to note that the important discrete attributes are considered better than their continuous counterparts. We can understand this if we consider the behaviour of $diff$ function (see equations 1 and 2).

Let's take two cases with 2 and 5 being their values of attribute A_i, respectively. If A_i is the discrete attribute, the value of $diff(A_i, 2, 5) = 1$, since the two categorical values are different. If A_i is the continuous attribute, $diff(A_i, 2, 5) = \frac{|2-5|}{7} \approx 0.43$. So, with this form of $diff$ function continuous attributes are underestimated. We can overcome this problem with the ramp function as proposed by (Hong, 1994). It can be used as a generalization of $diff$ function for the continuous attributes:

$$diff(A, I_1, I_2) = \begin{cases} 0 & d \leq t_{equal} \\ 1 & d > t_{different} \\ \frac{d - t_{equal}}{t_{different} - t_{equal}} & t_{equal} < d \leq t_{different} \end{cases} \qquad (11)$$

where $d = |value(A, I_1) - value(A, I_2)|$ present the distance between attribute values of the two instances, and t_{equal} and $t_{different}$ are two user definable threshold values. t_{equal} is the maximum distance between the two attribute values to still consider them equal, and $t_{different}$ is the minimum distance between attribute values to still consider them different. If we set $t_{equal} = 0$ and $t_{different} = \max(A) - \min(A)$ we get equation 2. We have omitted the use of ramp function from this presentation, as it complicates the basic idea.

The experiments with 500 examples show exactly the same success pattern, while with 100 cases, both RReliefFs succeded with MODULO-8-2 and failed on MODULO-8-3 and MODULO-8-4. This is as an indication that more examples are needed to solve the problems of higher complexity.

Table 2 summarizes the experiments in FRACTION and PARITY domains with 1000 examples available. As we can see both versions of RReliefF are successful in all problems. With 500 examples they fail on FRACTION-4 problem, while with 100 examples they fail on orders 3 and 4 of FRACTION and order 4 of PARITY. This supports the claim that we need more examples in order to recognize higher order dependencies.

The mean squared error was not successful on any of FRACTION and PARITY domains.

The results for LINEAR domain with 1000 examples in Table 3 include estimates of quality for all relevant attributes and for the best (as estimated) random attribute. The class is defined as a linear formula: $C = A_1 - 2A_2 + 3A_3 - 3A_4$. We can see that all three estimators are successful. They also correctly order the attributes by importance: A_3 and A_4 are selected as the best, which is compatible with absolute values of their coefficients (3 and -3, respectively). The best random attribute is correctly estimated as worse than important attributes.

The same picture occurs also with 500 and 100 examples. Therefore with linear dependencies the performance of RReliefF is comparable with that of the myopic estimator.

Table 2: Estimates in FRACTION-* and PARITY-* domains with 1000 examples.

Estimator	I_{best}	I_{worst}	R_{best}	?	I_{best}	I_{worst}	R_{best}	?
	FRACTION-2				PARITY-2			
RReliefF-e	0.033	0.032	-0.004	$\surd$	0.158	0.153	-0.001	$\surd$
RReliefF-k	0.034	0.034	-0.004	$\surd$	0.114	0.099	-0.033	$\surd$
MSE	0.084	0.084	0.083	$\times$	0.080	0.081	0.080	$\times$
	FRACTION-3				PARITY-3			
RReliefF-e	0.013	0.010	-0.001	$\surd$	0.106	0.087	-0.012	$\surd$
RReliefF-k	0.014	0.011	-0.002	$\surd$	0.082	0.044	-0.044	$\surd$
MSE	0.088	0.088	0.087	$\times$	0.079	0.079	0.079	$\times$
	FRACTION-4				PARITY-4			
RReliefF-e	0.003	0.002	0.000	$\surd$	0.062	0.048	-0.012	$\surd$
RReliefF-k	0.005	0.003	0.000	$\surd$	0.051	0.024	-0.049	$\surd$
MSE	0.083	0.083	0.082	$\times$	0.083	0.083	0.082	$\times$

Table 3: Estimates in LINEAR domain with 1000 examples.

Estimator	A_1	A_2	A_3	A_4	R_{best}	?
RReliefF-e	0.004	0.021	0.057	0.055	0.000	$\surd$
RReliefF-k	-0.005	0.013	0.038	0.039	-0.011	$\surd$
MSE	1.747	1.646	1.222	1.266	1.819	$\surd$

We continue our analysis with results in Table 4, for COSINUS domain where the class is defined as: $C = (-2A_2 + 3A_3)\cos(4\pi A_1)$. Both versions of RReliefF are still successful in both: separating important from irrelevant attributes and ranking them by importance. Only the latter is true for the mean squared error, which fails by only a small margin.

Results with 500 examples show the same trends, while 100 examples does not seem to be enough for RReliefF to correctly separate the best random attribute from A_2 (A_1 and A_3 still have higher estimate).

3.2 Building regression trees

We have developed a simple system for building binary regression trees using estimates of attributes' quality by either one of the RReliefF variants or by

Table 4: Estimates in COSINUS domain with 1000 examples.

Estimator	A_1	A_2	A_3	R_{best}	?
RReliefF-e	0.023	0.008	0.014	0.003	$\surd$
RReliefF-k	0.022	0.003	0.006	-0.001	$\surd$
MSE	0.682	0.700	0.699	0.698	$\times$

mean squared error (equation 9). If the discrete attribute is selected it is discretized optimally according to the purity of the split (Breiman et al., 1984) and if continuous attribute is chosen, the best split point is found with the same criterion. For simplicity reason our system has no pruning capabilities. As the stopping criterion it uses the minimal number of cases in a leaf (default: 5) or mimimal purity of a leaf (proportion of the root's standard deviation (s) in the leaf, which defaults to 8%). We label a leaf with the mean of the class values of the cases in the leaf. The parameters for the tree building and estimation are set to defaults and are the same in all the experiments.

We have made three series of experiments in each domain: with 1000, 500 and 100 examples. For every domain we used random splits of the data (70% for learning, 30% for testing). We have repeated the process for 30 times and computed the averages. We present results only for 1000 examples in Table 5.

Besides the number of leaves, we give two estimates of the mean squared error of the predictor: the resubstitution estimate R_{train} (error of the training sample) and the test sample estimate R_{test}. If the i^{th} example is written as ordered pair (c_i, x_i), where x_i is vector of attribute values, the error estimate of predictor ϕ on the set t is defined as:

$$R_t(\phi) = \frac{1}{N_t}\sum_{i=1}^{N_t}(c_i - \phi(x_i))^2 \tag{12}$$

where $\phi(x_i)$ is the class value predicted by ϕ. Because of the easier interpretation of the quality of the tree we included the squared root of R_t. In this way the error is expressed in the same units as the predicted value. Since R can be compared only within one domain, we also give the relative mean squared error:

$$RE_t(\phi) = \frac{R_t(\phi)}{R_t(\mu)}, \quad \text{where} \quad R_t(\mu) = \frac{1}{N_t}\sum_{i=1}^{N_t}(c_i - \overline{c_t})^2. \tag{13}$$

Table 5: Results for the regression trees with 1000 examples.

Domain	Estimator	#leaves	$\sqrt{R_{train}}$	RE_{train}	$\sqrt{R_{test}}$	RE_{test}
MODULO-8-2	RReliefF-e	64	0.000	0.000	0.000	0.000
	RReliefF-k	64	0.000	0.000	0.000	0.000
	MSE	187	0.730	0.108	2.354	1.104
MODULO-8-3	RReliefF-e	202	1.055	0.207	2.287	0.967
	RReliefF-k	201	1.032	0.198	2.216	0.908
	MSE	218	1.047	0.204	3.086	1.760
MODULO-8-4	RReliefF-e	193	1.302	0.320	2.953	1.652
	RReliefF-k	203	1.261	0.299	2.985	1.687
	MSE	220	1.068	0.215	3.105	1.824
FRACTION-2	RReliefF-e	194	0.065	0.051	0.227	0.611
	RReliefF-k	201	0.064	0.051	0.236	0.664
	MSE	203	0.078	0.074	0.321	1.251
FRACTION-3	RReliefF-e	235	0.095	0.102	0.343	1.322
	RReliefF-k	235	0.092	0.097	0.346	1.341
	MSE	228	0.095	0.101	0.411	1.891
FRACTION-4	RReliefF-e	241	0.108	0.139	0.389	1.826
	RReliefF-k	240	0.104	0.130	0.389	1.835
	MSE	228	0.090	0.098	0.404	1.960
PARITY-2	RReliefF-e	177	0.115	0.165	0.178	0.390
	RReliefF-k	80	0.132	0.216	0.166	0.337
	MSE	204	0.124	0.192	0.204	0.518
PARITY-3	RReliefF-e	152	0.122	0.187	0.173	0.375
	RReliefF-k	100	0.128	0.206	0.170	0.362
	MSE	207	0.140	0.249	0.233	0.688
PARITY-4	RReliefF-e	162	0.114	0.155	0.165	0.332
	RReliefF-k	141	0.114	0.155	0.165	0.330
	MSE	204	0.158	0.304	0.266	0.866
LINEAR	RReliefF-e	219	0.154	0.013	0.548	0.165
	RReliefF-k	219	0.153	0.013	0.544	0.162
	MSE	202	0.133	0.010	0.504	0.139
COSINUS	RReliefF-e	223	0.160	0.036	0.550	0.427
	RReliefF-k	225	0.146	0.030	0.527	0.393
	MSE	202	0.112	0.018	0.387	0.213

μ is a predictor which always returns the mean value of the class, so sensible predictors have $RE(\phi) < 1$.

In Table 5 we see that on the testing set both versions of RReliefF are much better than MSE on all variants of MODULO-8, FRACTION and PARITY domains.

In LINEAR and COSINUS domains MSE was better. The analysis of the trees showed that RReliefF was choosing exclusively important attributes close to the root, however, near the leaves it was not able to discriminate between important and random attributes. On the other hand MSE was mainly selecting important attributes throughout the tree.

Experiments with 500 examples show approximately the same picture in all domains, while for experiments with 100 examples the differences between the predictors are smaller and almost vanish in domains MODULO-8-4, FRACTION-4 and COSINUS.

4 Conclusions

Our experiments show that RReliefF is capable of discovering strong dependencies between attributes, while in domains without such dependencies it performs the same as the mean squared error. Its use in learning of regression trees seems promising. The RReliefF's estimates as well as the ReliefF's estimates in classification (Kononenko et al., 1996) become unreliable with small number of examples. It seems that in such situation both variants of ReliefF tend to overfit the data. When the number of instances is too small for ReliefF one should switch to estimating the quality of attributes with ordinary impurity measures. Further study shall develop techniques to detect the appropriate point in the tree building to make such a switch.

Analogously to extensions in ReliefF (Kononenko, 1994) we have extended RReliefF with handling of noisy and incomplete data and preliminary results show promising robustness. Both, RReliefF in regression and ReliefF in classification are estimators of equation 3, which gives a unified view on the estimation of quality of attributes for classification and regression.

References

Breiman, L., Friedman, L., Olshen, R., and Stone, C. (1984). *Classification and regression trees*. Wadsworth Inc., Belmont, California.

Elomaa, T. and Ukkonen, E. (1994). A geometric approach to feature selection. In De Raedt, L. and Bergadano, F., editors, *Proceedings of European Conference on Machine Learning*, pages 351–354. Springer Verlag.

Hong, S. J. (1994). Use of contextual information for feature ranking and discretization. Technical Report RC19664, IBM. to appear in IEEE Trans. on Knowledge and Data Engineering.

Hunt, E., Martin, J., and Stone, P. (1966). *Experiments in Induction*. Academic Press, New York.

Karalič, A. (1992). Employing linear regression in regression tree leaves. In Neumann, B., editor, *Proceedings of ECAI'92*, pages 440–441. John Wiley & Sons.

Kira, K. and Rendell, L. A. (1992a). The feature selection problem: traditional methods and new algorithm. In *Proc. AAAI'92*.

Kira, K. and Rendell, L. A. (1992b). A practical approach to feature selection. In D.Sleeman and P.Edwards, editors, *Proc. Intern. Conf. on Machine Learning*, pages 249–256. Morgan Kaufmann.

Kononenko, I. (1994). Estimating attributes: analysis and extensions of Relief. In De Raedt, L. and Bergadano, F., editors, *Proceedings of European Conference on Machine Learning*, pages 171–182. Springer Verlag.

Kononenko, I., Šimec, E., and Robnik Šikonja, M. (1996). Overcoming the myopia of inductive learning algorithms with ReliefF. *Applied Intelligence*. (in press).

Mantaras, R. (1989). ID3 revisited: A distance based criterion for attribute selection. In *Proceedings of Int. Symp. Methodologies for Intelligent Systems*, Charlotte, North Carolina, USA.

Quinlan, J. (1992). Learning with continuous classes. In *Proceedings of AI'92*. World Scientific.

Smyth, P. and Goodman, R. (1990). Rule induction using information theory. In Piatetsky-Shapiro, G. and Frawley, W., editors, *Knowledge Discovery in Databases*. MIT Press.