ReliefF for estimation and discretization of attributes in classification, regression, and ILP problems

Igor Kononenko, Marko Robnik-Šikonja, Uroš Pompe
University of Ljubljana, Faculty of computer and information science
Tržaška 25, SI-1001 Ljubljana, Slovenia
e-mail: {igor.kononenko, marko.robnik, uros.pompe}@fri.uni-lj.si

Abstract

Instead of myopic impurity functions, we propose the use of ReliefF for heuristic guidance of inductive learning algorithms. The basic algoritm RELIEF, developed by Kira and Rendell (Kira and Rendell, 1992a;b), is able to efficiently solve classification problems involving highly dependent attributes, such as parity problems. However, it is sensitive to noise and is unable to deal with incomplete data, multiclass, and regression problems (continuous class). We have extended RELIEF in several directions. The extended algorithm ReliefF is able to deal with noisy and incomplete data, can be used for multiclass problems, and its regressional variant RReliefF can deal with regression problems. Another area of application is inductive logic programming (ILP) where, instead of myopic measures, ReliefF can be used to estimate the utility of literals during the theory construction.

1 Introduction

Inductive learning algorithms typically use the greedy search strategy to overcome the combinatorial explosion during the search for good hypotheses.

The heuristic function that estimates the potential successors of the current state in the search space has the major role in the greedy search. Current inductive learning algorithms use variants of impurity functions like information gain (Hunt et al., 1966), gain-ratio (Quinlan, 1986), gini-index (Breiman et al., 1984), distance measure (Mantaras, 1989), and j-measure (Smyth & Goodman, 1990). However, all these measures assume that attributes are independent and therefore, in domains with strong conditional dependencies between attributes, the greedy search has poor chances of revealing a good hypothesis.

Kira and Rendell (1992a;b) developed an algorithm called RELIEF, which was shown to be very efficient in estimating the quality of attributes. For example, in the parity problems of various degrees with a significant number of irrelevant (random) additional attributes RELIEF is able to correctly estimate the relevance of all attributes in a time proportional to the number of attributes and the square of the number of training instances (this can be further reduced by limiting the number of iterations in RELIEF). While the original RELIEF can deal with discrete and continuous attributes, it can not deal with incomplete data and is limited to two-class problems only.

We developed an extension of RELIEF called ReliefF that improves the original algorithm by estimating probabilities more reliably and extends it to handle incomplete and multi-class data sets while the complexity remains the same. ReliefF seems to be a promising heuristic function that may overcome the myopia of current inductive learning algorithms. Kira and Rendell used RELIEF as a preprocessor to eliminate irrelevant attributes from data description before learning. ReliefF is general, efficient, and reliable enough to be used for: the non-myopic discretization of attributes, the estimation of attributes in classification problems, the non-myopic estimation of the utility of literals during the construction of the first order theory by an inductive logic programming (ILP) system, and the regressional variant, RReliefF, can be used for the estimation and discretization of attributes in regression problems.

The paper is organized as follows. In the next section, the original RELIEF is briefly described along with its interpretation and its extended version ReliefF. In Section 3, we present the regressional variant RReliefF. Section 4 discusses the use of ReliefF for discretization of attributes. Section 5 shows how ReliefF can be used within ILP systems. In discussion we review the possible applications of ReliefF.

```
set all weights W[A] := 0.0;
for i := 1 to n do
    begin
      randomly select an instance R;
      find nearest hit H and nearest miss M;
      for A := 1 to #attributes do
        W[A] := W[A] - diff(A,R,H)/n + diff(A,R,M)/n;
    end;
```

Figure 1 The basic algorithm of RELIEF

2 ReliefF

2.1 RELIEF

The key idea of RELIEF is to estimate attributes according to how well their values distinguish among 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 & Rendell, 1992a;b) randomly selects n training instances, where n is the user-defined parameter. The algorithm is given in Figure 1.

Function *diff(Attribute,Instance1,Instance2)* calculates the difference between the values of Attribute for two instances. For discrete attributes the difference is either 1 (the values are different) or 0 (the values are equal), while for continuous attributes the difference is the actual difference normalized to the interval $[0, 1]$. Normalization with n guarantees all weights $W[A]$ to be in the interval $[-1, 1]$.

The weights are estimates of the quality of attributes. The rationale of the formula for updating the weights is that a good attribute should have the same value for instances from the same class (subtracting the difference $diff(A, R, H)$) and should differentiate between instances from different classes (adding the difference $diff(A, R, M)$).

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 differences over all attributes. In fact, original RELIEF uses the squared difference, which for discrete attributes is equivalent to *diff*. In all our experiments, there was no significant difference between results using *diff* or the squared difference. However, we prefer to use non-squared *diff* as it allows the probabilistic interpretation. If N is the number of all training instances then the complexity of the above algorithm is $O(n \times N \times \#attributes)$.

2.2 Interpretation of RELIEF's estimates

The following derivation shows that RELIEF's estimates are strongly related to impurity functions. It is obvious that the RELIEF's estimate $W[A]$ of the attribute A is an approximation of the following difference of probabilities:

$$W[A] = P(\text{different value of A}|\text{nearest instance from a different class}) - P(\text{different value of A}|\text{nearest instance from the same class}) \quad (1)$$

If we eliminate from (1) the requirement that the selected instance is *the nearest*, the formula becomes:

$$W[A] = P(\text{different value of A}|\text{different class}) - P(\text{different value of A}|\text{same class}) \quad (2)$$
$$= P(\text{equal value of A}|\text{same class}) - P(\text{equal value of A}|\text{different class})$$

If we rewrite $P_{eqval} = P(\text{equal value of A})$, $P_{samecl} = P(\text{same class})$, and $P_{samecl|eqval} = P(\text{same class}|\text{equal value of A})$ we obtain using the Bayesian rule:

$$W[A] = \frac{P_{samecl|eqval} P_{eqval}}{P_{samecl}} - \frac{(1 - P_{samecl|eqval}) P_{eqval}}{1 - P_{samecl}}$$

Using equalities

$$P_{samecl} = \sum_C P(C)^2$$

$$P_{samecl|eqval} = \sum_V \left(\frac{P(V)^2}{\sum_V P(V)^2} \times \sum_C P(C|V)^2 \right)$$

we obtain:

$$\begin{aligned} W[A] &= \frac{P_{eqval} \times Gini'(A)}{P_{samecl}(1 - P_{samecl})} \\ &= constant \times \sum_V P(V)^2 \times Gini'(A) \end{aligned} \quad (3)$$

where

$$Gini'(A) = \sum_V \left(\frac{P(V)^2}{\sum_V P(V)^2} \times \sum_C P(C|V)^2 \right) - \sum_C P(C)^2 \qquad (4)$$

is highly correlated with the gini-index gain (Breiman et al., 1984) for classes C and values V of the attribute A. The only difference is that instead of the factor

$$\frac{P(V)^2}{\sum_V P(V)^2} \quad \text{the gini} - \text{index gain uses} \quad \frac{P(V)}{\sum_V P(V)} = P(V)$$

Equation (3) shows strong relation of the RELIEF's weights with the gini index. The probability $\sum_V P(V)^2$ that two instances have the same value of attribute A in eq. (3) is a kind of normalization factor for multi-valued attributes. Impurity functions tend to overestimate multi-valued attributes and various normalization heuristics are needed to avoid this tendency (e.g. gain ratio (Quinlan, 1986) and binarization of attributes (Cestnik et al., 1987)). Equation (3) shows that RELIEF implicitly uses such a normalization. This fact was further investigated and confirmed in (Kononenko, 1995).

The above derivation eliminated the *"nearest instance"* condition from the probabilities. If we put it back we can interpret the RELIEF's estimates as the average over local estimates in smaller parts of the instance space. This enables RELIEF to take into account dependencies between attributes which can be detected in the context of locality. From the global point of view, these dependencies are hidden due to the effect of averaging over all training instances, and exactly this makes impurity functions myopic.

2.3 Extensions of RELIEF

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 only. Equation (1) is of crucial importance for any extension of RELIEF. It turned out that the extensions of RELIEF are not straightforward unless we realized that RELIEF in fact approximates probabilities. The extensions should be designed in such a way that probabilities are reliably approximated. We developed an extension of RELIEF, called ReliefF, that improves the original algorithm by estimating probabilities more reliably and extends

it to deal with incomplete and multi-class data sets. A brief description of the extensions follows.

Reliable probability approximation: The parameter n in the algorithm RELIEF, described in Section 2.1, represents the number of instances for approximating probabilities in eq. (1). The larger n implies more reliable approximation. The obvious choice, adopted in ReliefF for relatively small number of training instances (up to one thousand), is to run the outer loop of RELIEF over all available training instances. Due to efficiency reasons, however, it is often more practical to keep n smaller, such as few hundreds.

The selection of the nearest neighbors is of crucial importance in RELIEF. The purpose is to find the nearest neighbors with respect to important attributes. Redundant and noisy attributes may strongly affect the selection of the nearest neighbors and therefore the estimation of probabilities with noisy data becomes unreliable. To increase the reliability of the probability approximation ReliefF searches for k nearest hits/misses instead of only one near hit/miss and averages the contribution of all k nearest hits/misses. It was shown that this extension significantly improves the reliability of estimates of the attributes' qualities (Kononenko, 1994). To overcome the problem of parameter tuning we proppose the default value $k = 10$ which, empirically, gives satisfactory results.

Incomplete data: In order to deal with incomplete data sets, the function
diff(Attribute,Instance1, Instance2) in ReliefF is extended to missing values of attributes by calculating the probability that two given instances have different values for the given attribute:

- if one instance (e.g. I1) has unknown value:

$$diff(A, I1, I2) = 1 - P(value(A, I2)|class(I1))$$

- if both instances have unknown value:

$$diff(A, I1, I2) = 1 - \sum_{V}^{\#values(A)} (P(V|class(I1)) \times P(V|class(I2)))$$

The conditional probabilities are approximated with relative frequencies from the training set.

Multi-class problems: Kira and Rendell (1992a;b) claim that RELIEF can be used to estimate the attributes' qualities in data sets with more than two classes by splitting the problem into a series of 2-class problems. This solution seems unsatisfactory. To use it in practice, RELIEF should be able to deal with multi class problems without any prior changes in the knowledge representation that could affect the final outcomes.

Instead of finding one near miss M from a different class, ReliefF finds one near miss $M(C)$ for each different class C and averages their contribution for updating the estimate $W[A]$. The average is weighted with the prior probability of each class:

$$W[A] := W[A] - diff(A,R,H)/n + \sum_{C \neq class(R)} \left[\frac{P(C)}{1 - P(class(R))} \times diff(A,R,M(C)) \right] /n$$

The idea is that the algorithm should estimate the ability of attributes to separate each pair of classes regardless of which two classes are closest to each other.

Note that the time complexity of ReliefF is the same as that of RELIEF. The complete algorithm is provided in Figure 2.

3 RReliefF: Regressional ReliefF

The usual impurity measure used in regression is the mean squared error (MSE) (Breiman et al., 1984):

$$MSE = 1/N \sum_{i=1}^{N} (C_i - \overline{C})^2 \tag{5}$$

where

$$\overline{C} = 1/N \sum_{i=1}^{N} C_i$$

is the average class value. There is an interesting relation between the mean squared error and the (prior) gini-index (Breiman et al., 1984). The prior

set all weights W[A] := 0.0;
for i := 1 **to** n **do**
begin
randomly select an instance R;
find k nearest hits H_j
for each class $C \neq class(R)$ **do**
find k nearest misses $M_j(C)$;
for A := 1 **to** #attributes **do**
$W[A] := W[A] - \sum_{j=1}^{k} diff(A, R, H_j)/(n \times k) +$
$\sum_{C \neq class(R)} \left[\frac{P(C)}{1-P(class(R))} \sum_{j=1}^{k} diff(A, R, M_j(C)) \right] /(n \times k)$;
end;

Figure 2 Algorithm ReliefF

gini-index is defined with

$$Gini\_prior = 1 - \sum_{C} P(C)^2 \tag{6}$$

Iif the classification problem with two classes is transformed into regression problem by labeling one class with 0 and the other with 1, then the following equality holds:

$$Gini\_prior = 2 \times MSE \tag{7}$$

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 different classes. This probability can be modeled with the relative distance between the class values of the two instances.

Still, to estimate W[A] in equation (1), the information about the sign of each contributed term is missing. In the following derivation we reformulate equation (1), so that it can be directly evaluated using the probability of two instances belonging to different classes. If we rewrite

$$P_{diffA} = P(\text{different value of A}|\text{nearest instances}) \tag{8}$$

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

Figure 3: Pseudo code of RReliefF (Regressional ReliefF)

$$P_{diffC} = P(\text{different class}|\text{nearest instances}) \tag{9}$$

and

$$P_{diffC|diffA} = P(\text{different class}|\text{different value of A and nearest instances}) \tag{10}$$

we obtain from (1) using Bayes rule:

$$W[A] = \frac{P_{diffC|diffA}P_{diffA}}{P_{diffC}} - \frac{(1 - P_{diffC|diffA})P_{diffA}}{1 - P_{diffC}} \tag{11}$$

Therefore, we can estimate $W[A]$ by approximating terms defined by equations 8, 9 and 10. This can be done by the algorithm on Figure 3.

Note that the time complexity of RReliefF is the same as that of original RELIEF, i.e. $O(n \times N \times \#attributes)$. The most complex operation within the main **for** loop is the selection of **k** nearest instances I_j, which can be done in $O(N \times \#attributes)$ steps. $O(\#attributes)$ 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 $O(k \log N) < O(N \times \#attributes)$ steps).

4 Discretization of attributes

Discretization divides the values of the continuous attribute into a number of intervals. Each interval can then be treated as one value of new discrete attribute. Discretization of attributes can reduce the learning complexity and help to understand the dependencies between the attributes and the target concept. There are several methods that can be used to discretize continuous attributes (Richeldi and Rossotto, 1995).

A usual greedy algorithm for discretization of attributes is as follows (Cestnik, 1989):

BestDiscretization = {}
SetOfBoundaries = {}
repeat
add the split which maximizes the heuristic measure to the SetOfBoundaries
if SetOfBoundaries is best so far **then**
BestDiscretization = SetOfBoundaries
until m times the heuristic is worse than in previous step

At each step the algorithm searches for a boundary which, when added to the current set of boundaries, maximizes the heuristic estimate of the discretized attribute.

Algorithms for discretization use myopic measures, such as a well known measure of dissimilarity (distance) between the attribute and the class (Mantaras, 1989). Such measures typically take into account the conditional probabilities of the attribute values given the class and vice versa, however, ignoring the information of the values of all other attributes. The main advantage of ReliefF is its non-myopic behaviour. ReliefF can predict the quality of strongly dependent attributes. Therefore, using ReliefF in the above algorithm leads to a non-myopic discretization of continuous attributes (Robnik & Kononenko, 1995). The regressional version RReliefF can be used to discretize attributes in regresion problems.

Input::
Literal space LS;
Current training set $T = T^+ \cup T^-$;
T^+,T^-: positive and negative examples respectively
Output:
Weight vector W where $W[L]$ estimates the quality of literal L

---

set all weights W[L] := 0.0;
for i := 1 **to** n **do**
begin
randomly select an instance $R \in T^+$;
find k nearest hits H and k nearest misses M;
for L := 1 **to** #literals **do**
for i := 1 **to** k **do**
$W[L] := W[L] + (Diff_A(L, R, M[i]) - Diff(L, R, H[i]))/(k \times n)$;
end

Figure 4: ReliefF based literal quality assessment

5 ReliefF for inductive logic programming

When dealing with the classification problems current ILP systems (Muggleton, 1992; Lavrač and Džeroski, 1994) often lag behind the state-of-the-art attributional learners. Part of the blame can be ascribed to a much larger hypothesis space which, therefore, can not be so thoroughly explored. ReliefF as described above is suitable for the propositional representation of training examples. A slightly different approach is needed when estimating the quality of literals which are the candidates for augmenting the current clause under construction, when inducing the first order theories with an ILP system.

The main difference stems from the fact that while learning in the propositional language we are only interested in the boundaries between different classes. On the other hand, when learning in the first order language, we are not searching for boundaries but for a theory that explains positive learning instances and does not cover negative ones. A crucial part of ReliefF is the function that measures the difference (distance) between the training instances.

The key idea of using ReliefF within ILP is to estimate literals according to how well they distinguish between the instances that are logically similar. Our algorithm on Figure 4 searches for k nearest hits/misses. The search for the nearest hits and misses is guided by the *total distance* between the two examples ($Diff_T$(*Example1*,*Example2*)). This distance is computed as follows:

$$Diff_T(E_1, E_2) = \frac{1}{|LS|} \sum_{L \in LS} Diff(L, E_1, E_2) \tag{12}$$

It is simply a normalized sum of differences over the literal space LS. It estimates the logical similarity of two instances relative to the background knowledge.

Both the total distance ($Diff_T$) and the estimates W depend on the definition of $Diff$ ($Diff_A$ is an asimetric version of $Diff$). Table 1 shows the definition of $Diff$ and $Diff_A$. The first two columns represent the *coverage*

$L(E_1)$	$L(E_2)$	$Diff(L, E_1, E_2)$	$Diff_A(L, E_1, E_2)$
0	0	0	0
0	1	1	0
1	0	1	1
1	1	0	0

Table 1: Definition of the $Diff$ and $Diff_A$ functions.

of literal L over the examples E_1 and E_2 respectively. The coverage denotes the truth value of some partially build clause Cl' with literal L included when the head of the clause is instantiated with example $E_i, i \in \{1, 2\}$. Note that since E_1 is always from T^+ (Figure 4) the $Diff_A$ function gives the preference to literals covering the positive examples. The details can be found in (Pompe and Kononenko, 1995).

6 Discussion

We have verified the effectivness of ReliefF in various areas: for attribute estimation and discretization in classification and regression problems and

for estimating the utility of literals in ILP. In all areas ReliefF exhibited the advantage over myopic impurity measures in problems with strongly conditionaly dependent attributes, while in domains without such dependencies ReliefF performs the same as ordinary impurity measures. Typical problems that can be effectively solved with ReliefF and cannot be solved with ordinary impurity measures, are the parity classification and relational (ILP) problems and, e.g. the sum-by-modulo regressional problems.

Equation (3) shows an interesting relations between the ReliefF's estimates and impurity measures: the gini-index gain and (indirectly via Equation 7) the mean squared error. Equation (11) can be used for estimating the quality of attributes both in classification and regression problems. Therefore, it provides a unified view on both areas of machine learning.

ReliefF can efficiently estimate continuous and discrete attributes. The implicit normalization in Eq. (3) enables ReliefF to appropriately deal with multivalued attributes (Kononenko, 1995). However, if a machine learning system used Eq. (3) instead of the information gain, it would still be myopic. For example, in parity problems, Eq. (3) would estimate all attributes as equally non-important. Therefore, the reason for the success of ReliefF is in the "nearest hits/misses" heuristic which influences the estimation of probabilities. This heuristic enables ReliefF to detect strong dependencies between the attributes which would be overlooked if the estimates of probabilities would be done on randomly selected instances instead of the nearest ones.

ReliefF is an efficient heuristic estimator of the attribute quality that is able to deal with data sets with dependent and independent attributes. The extensions of the basic RELIEF algorithm enable it to deal with noisy, incomplete, and multi-class data sets. With increasing the number (k) of nearest hits/misses the correlation of the ReliefF's estimates with other impurity functions also increases. When k becomes much greater than the number of instances in the same peak of the instance space, ReliefF becomes myopic as other impurity measures.

The performance of current inductive learning systems can be improved by replacing the existing heuristic functions with ReliefF. Although ReliefF may overcome the myopia, it is not directly useful when the change of the representation is required. In such cases the constructive induction should be applied. A good idea for constructive induction may be to use ReliefF instead or in the combination with the lookahead.

Current ILP systems also use greedy search techniques and the heuristics that guide the search are myopic. Pompe and Kononenko (1995) implemented a version of ReliefF in the FOIL like ILP system called ILP-R and preleminary experiments show advantages of this system over other ILP systems.

References

[1] Breiman L., Friedman J.H., Olshen R.A. & Stone C.J. (1984) *Classification and Regression Trees*, Wadsworth International Group.

[2] Cestnik, B. (1989) Informativity-Based Splitting of Numerical Attributes into Intervals. In Hamza, M.H.(ed):*Expert Systems Theory & Applications, Proc. of the IASTED International Symposium*, Acta Press.

[3] Cestnik, B., Kononenko, I., Bratko, I. (1986) ASSISTANT 86: A Knowledge-Elicitation Tool for Sophisticated Users. In: Ivan Bratko, Nada Lavrač (eds.): *Progress in Machine Learning, Proceedings of EWSL 87.* Wilmslow, Sigma Press, 1987.

[4] Hunt E., Martin J & Stone P. (1966) *Experiments in Induction*, New York, Academic Press.

[5] Kira K. & Rendell L. (1992a) A practical approach to feature selection, *Proc. Intern. Conf. on Machine Learning* (Aberdeen, July 1992) D.Sleeman & P.Edwards (eds.), Morgan Kaufmann, pp.249-256.

[6] Kira K. & Rendell L. (1992b) The feature selection problem: traditional methods and new algorithm. *Proc. AAAI'92*, San Jose, CA, July 1992.

[7] Kononenko I. (1994) Estimating attributes: Analysis and extensions of RELIEF. *Proc. European Conf. on Machine Learning* (Catania, April 1994), L. De Raedt & F.Bergadano (eds.), Springer Verlag.

[8] Kononenko I. (1995) On biases when estimating multivalued attributes, *Proc. Intern. Joint Conf. on Artificial Intelligence, IJCAI-95*, Montreal, Canada, August 1995.

[9] Kononenko, I., Šimec, E. (1995) Induction of decision trees using RELIEFF. In: Della Riccia, G., Kruse, R., Viertl, R., (eds.): *Mathematical and statistical methods in artificial intelligence*, Springer Verlag.

[10] Lavrač N., Džeroski S. (1994) *Inductive logic programming*, Ellis Horwood.

[11] Mantaras R.L. (1989) ID3 Revisited: A distance based criterion for attribute selection, *Proc. Int. Symp. Methodologies for Intelligent Systems*, Charlotte, North Carolina, U.S.A., Oct. 1989.

[12] Muggleton S. (ed.) (1992) *Inductive Logic Programming.* Academic Press.

[13] Pompe U. & Kononenko I. (1995) Linear space induction in first order logic with RELIEFF, In: G.Della Riccia, R.Kruse, R.Viertl (eds.) *Mathematical and statistical methods in artificial intelligence*, Springer Verlag.

[14] Quinlan J.R. (1986) Induction of decision trees, *Machine learning*, Vol. 1.

[15] Richeldi M., Rossotto M.(1995) Class-Driven Statistical Discretization of Continuous Attributes. In Lavrač N., Wrobel S.(eds.) *Machine Learning: ECML-95*, Springer Verlag.

[16] Robnik M., Kononenko I. (1995) Discretization of continuous attributes using ReliefF, *Proc. ERK-95*, Portorož, Sept. 1995. pp.149-152.

[17] Smyth P. & Goodman R.M. (1990) Rule induction using information theory. In. G.Piatetsky-Shapiro & W.Frawley (eds.) *Knowledge Discovery in Databases*, MIT Press.