The Development of Holte's 1R Classifier

Craig Nevill-Manning, Geoffrey Holmes and Ian H. Witten
Department of Computer Science,
University of Waikato,
Hamilton, New Zealand.
cgn,geoff,ihw@cs.waikato.ac.nz

Abstract

The 1R procedure for machine learning is a very simple one that proves surprisingly effective on the standard datasets commonly used for evaluation. This paper describes the method and discusses two areas that can be improved: the way that intervals are formed when discretizing continuously-valued attributes, and the way that missing values are treated. Then we show how the algorithm can be extended to avoid a problem endemic to most practical machine learning algorithms—their frequent dismissal of an attribute as irrelevant when in fact it is highly relevant when combined with other attributes.

1 Introduction

In a contentious paper demonstrating the inadequacies of datasets used to benchmark machine learning algorithms, Robert Holte of the University of Ottawa described a very simple learning algorithm, which he called 1R, that competes favourably with state-of-the-art techniques in the field [Holte, 1993]. Holte did not promote the use of 1R as a rival mainstream learning technique; rather, he used it to show that most of the datasets that researchers were using to test their algorithms did not embody very complex rules.

Holte went on to debate the question of whether or not real-world datasets contain complex relationships. Citing some documentary evidence that they do not, he concluded with a salutary appeal to researchers to use a "simplicity first" methodology in machine learning. We have adopted this philosophy in our own work and have been able to demonstrate the efficacy of 1R as a filter to select relevant subsets of attributes prior to learning [Holmes and Nevill-Manning, 1995].

Given Holte's motivation for developing 1R, it is not surprising that some of the details of the algorithm have not been fully explored, namely quantization of continuously-valued attributes and the handling of missing values. Since we now put 1R to regular use and have extended its application to attribute selection, we were keen to tidy up these details.

In this paper we present our improvements to the basic algorithm. We also extend it to find rules from combinations of attributes—this was mentioned in Holte's paper, but not implemented. The sections that follow describe the original implementation, the enhancements we have made to it, our extension to avoid greedy selection, and some preliminary experimental results that show that the changes are indeed beneficial.

2 The 1R Algorithm

Like other empirical learning methods, 1R takes as input a set of examples, each with several attributes and a class. The aim is to infer a rule that predicts the class given the values of the attributes. The 1R algorithm chooses the most informative single attribute and bases the rule on this attribute alone. Full details can be found in Holte's paper, but the basic idea is:

```
For each attribute $a$, form a rule as follows:
  For each value $v$ from the domain of $a$,
    Select the set of instances where $a$ has value $v$.
    Let $c$ be the most frequent class in that set.
    Add the following clause to the rule for $a$:
      if $a$ has value $v$ then the class is $c$
  Calculate the classification accuracy of this rule.
Use the rule with the highest classification accuracy.
```

The algorithm assumes that the attributes are discrete. If not, then they must be discretized, and Holte presents a technique for this (see Section 2.2). Missing values are handled in the algorithm by treating them as a separate value in the enumeration of an attribute (see Section 2.4).

2.1 A Worked Example

Table 1 shows the golf data [Quinlan, 1994], a small illustrative dataset that uses weather information to decide whether or not to play golf. The dataset has two nominal attributes, *outlook*

outlook	*temp.*	*hum.*	*windy*	*class*
sunny	85	85	false	Don't Play
sunny	80	90	true	Don't Play
overcast	83	78	false	Play
rain	70	96	false	Play
rain	68	80	false	Play
rain	65	70	true	Don't Play
overcast	64	65	true	Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
rain	75	80	false	Play
sunny	75	70	true	Play
overcast	72	90	true	Play
overcast	81	75	false	Play
rain	71	80	true	Don't Play

Table 1: The golf dataset

(with values *sunny*, *overcast* and *rain*), and *windy* (with values *true* and *false*), and two continuous-valued ones, *temperature* and *humidity*. In order to demonstrate the basic workings of the algorithm, we consider only the nominal attributes.

The frequencies of each class for each value of the nominal attributes are shown in Table 2. The rules derived from these Tables, and their accuracies, are shown in Table 3. For each attribute and value, the class chosen is the one that occurs most frequently in that combination—for example, when the *outlook* attribute is *sunny*, the class chosen is *Don't Play* because, as Table 2 shows, that occurs three times whereas the *Play* class occurs only twice. Where the highest frequencies are equal, a random choice is made. For example, in the *windy* rule of Table 3, the *if true then Play* choice would be just as acceptable as the *if true then Don't Play* choice that is shown: from these examples it seems that the *windy* attribute being *true* has no significance in deciding whether or not to play golf.

2.2 Quantization

Any method for turning a range of values into disjoint intervals must take care to avoid creating large numbers of rules with many small intervals. This is known as the problem of "overfitting," because such rules are overly specific to the data set and do not generalize well. Holte achieves this by requiring all intervals (except the rightmost) to contain more than a predefined number of examples in the same class. Empirical evidence led him to a value of six for datasets with large numbers of instances and three for smaller datasets (with less than about 50 instances) [Holte *et al*, 1989].

outlook	Play	Don't Play
overcast	4	0
sunny	2	3
rain	3	2

windy	Play	Don't Play
true	3	3
false	6	2

Table 2: Frequencies of values in nominal attributes

outlook	**if** overcast **then**	Play	(4/4)
	else if sunny **then**	Don't Play	(3/5)
	else if rain **then**	Play	(3/5)
	Accuracy = 10/14 (71.4%)		
windy	**if** true **then**	Don't Play	(3/6)
	else if false **then**	Play	(6/8)
	Accuracy = 9/14 (64.3%)		

Table 3: Rules derived from Table 2

As an example, the *temperature* attribute of the golf dataset gives the following value/classification pairs:

```
64 65 68 69 70 | 71 72 72 75 75 80 81 83 | 85
 P  D  P  P  P |  D  P  D  P  P  D  P  P |  D
```

Holte's technique would form an interval of class P stretching from 64 to 71, one of class P from 71 to 83, and another of class D including just 85. The two leftmost intervals would then be merged, as they predict the same class. The accuracy of this quantization is 10/14 (there are four misclassifications in the leftmost interval).

2.3 New Approach

Our algorithm for splitting a continuous range of these pairs into discrete intervals is as follows:

1. Sort the tuples by attribute value.
2. Form intervals by placing a split point between every pair of different values.
3. Repeat
    a. remove split points between intervals that predict the same class,
    b. examine the decrease in accuracy which would result from removing each split point,
    c. remove the least costly split point (in the event of a tie, choose one at random);

    until there are no more split points.
4. Choose the best split point on the accuracy *vs* number of splits curve.

This is how we would proceed on the *temperature* data.

```
| 64 | 65 | 68 | 69 | 70 | 71 | 72 72 | 75 75 | 80 | 81 | 83 | 85 |
|  P |  D |  P |  P |  P |  D |  P  D |  P  P |  D |  P |  P |  D |
```

Removing the split points between intervals that predict the same class gives

```
| 64 | 65 | 68 69 70 | 71 72 72 | 75 75 | 80 | 81 83 | 85 |
|  P |  D |  P  P  P |  D  P  D |  P  P |  D |  P  P |  D |
```

Each of the splits separates intervals of different classes, and removing any of them decreases the overall accuracy of the quantisation. For example, removing the split-point between 65 and 68 creates a new interval from 65 to 70, whose predominant class is *Play*. The *Don't Play* tuple is now misclassified as *Play*, resulting in a reduction in accuracy of one example, or 7%. This turns out to be the least costly split point to remove (although there are others that involve the same cost).

```
| 64 | 65 68 69 70 | 71 72 72 | 75 75 | 80 | 81 83 | 85 |
|  P |  D  P  P  P |  D  P  D |  P  P |  D |  P  P |  D |
```

The split point between 64 and 65 can now be removed without further loss of accuracy.

```
| 64 65 68 69 70 | 71 72 72 | 75 75 | 80 | 81 83 | 85 |
|  P  D  P  P  P |  D  P  D |  P  P |  D |  P  P |  D |
```

The algorithm continues to remove split points and record the resulting accuracy until none are left. The outcome of the exercise can be summarized in a table charting the tradeoff as split points are removed, as shown in Table 4.

Plotting the number of intervals against the resulting accuracy shows the effect of the algorithm as it progresses. We seek the “knee” of this curve—the optimal tradeoff between overfitting the data and obtaining good accuracy. This can be obtained by finding the maximum value of the second derivative of the curve (once the axes have been converted to comparable units). The calculations involved are beyond the scope of this paper. However, we demonstrate the process with an example.

The golf data has insufficient complexity to define an interesting knee point on the tradeoff curve. Consider instead a widely-used machine learning dataset that has nine continuous attributes (the so-called “glass” dataset G2).

It is clear from Figure 1 that attributes *Rl* and *Ca* exhibit points on their tradeoff curves where dramatic changes take place. The curve for attribute *Na* is more problematic. There is no useful maximum for this curve. Our hypothesis is that attributes having this characteristic curve are irrelevant ones—their values are randomly scattered across the real line, and they make no contribution to classification accuracy. If this hypothesis is true, it provides a further piece of information that 1R can use when determining the relevance of attributes.

2.4 Missing Values

Missing values are treated by Holte’s system as a separate value that an attribute may assume. This implies that whether or not an attribute is missing constitutes information that is useful for prediction. In some circumstances this is plausible, but it is a risky assumption across all datasets. When using 1R as a filter, it can be particularly misleading to choose attributes with large numbers of missing values that seem to make highly accurate predictions. Consider, for example, this rule formed from the *protime* attribute by 1R from one of Holte’s datasets (HE):

if protime is missing **then** live (53/67)
else if protime < 36 **then** die (8/12)
else if protime ≥ 36 **then** live (66/76)

There are 155 instances of which approximately 1/3 are missing. This attribute is ranked fourth in accuracy (of 19) by 1R, but with so many missing values it is difficult to conclude that it is really relevant.

splits	*accuracy*
7	13
6	12
5	12
4	11
3	10
2	10
1	9
0	9

Table 4: Tradeoff between splits and accuracy

2.5 New Approach

Our approach to missing values is well demonstrated in the example above. We assume that the fact that an attribute value is missing implies that it contains no information. Accordingly, when data is missing we predict the default class—the most commonly occurring class overall.

In the majority of cases that Holte examined this new approach arrives at the same result as his original method, because for these datasets the class predicted by the missing values is indeed the default class. Although our more conservative approach often arrives at the same result, it is more satisfactory when 1R is used for automatic attribute selection.

2.6 Avoiding Greedy Decisions

Most machine learning schemes operate *greedily*, by considering attributes individually and choosing the best one at each point. This process does not guarantee to find the best decision tree or rule set overall, but is employed because it is computationally feasible.

John *et al.* (1994) have shown that considering attributes individually when building decision trees can result in larger and less accurate trees than if attributes are considered in combination. They give an example where four attributes predict the class perfectly when taken together, but where C4.5 prefers attributes that were generated randomly.

Because 1R is extremely efficient in its evaluation of attributes, it can be used to identify promising attribute combinations. This process, which we call 2R, produces new attributes by concatenating pairs of attributes, then runs 1R on this new dataset. The best 2-rule formed by this process indicates the best pair of attributes in the dataset.

For example, in the golf dataset, one new attribute would result from the concatenation of *outlook* and

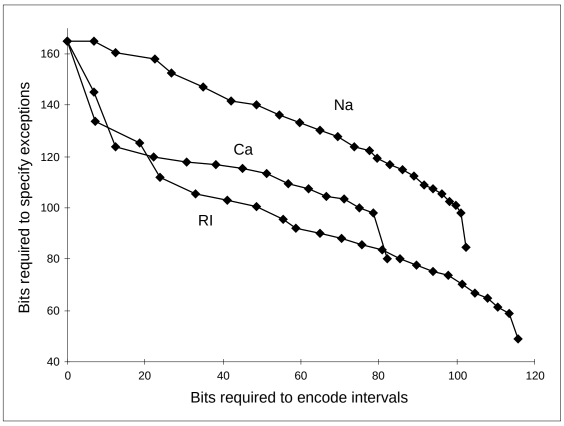

Figure 1: Tradeoff curves for three attributes from G2

windy. The values of this new attribute would be *sunny-false*, *sunny-true*, *overcast-false*, *overcast-true*, *rain-false* and *rain-true*. These new attributes gain accuracy partly as a result of the greater number of unique values that they contain, so comparisons with the individual attributes are meaningless. However comparison with other derived attributes is informative.

The best of the resulting 2-rules does not always include the attribute which produces the best 1-rule. When it does not, the 2-rule contains a useful pair of attributes which would have been ignored by greedy schemes like C4.5.

Some problems may require three or more attributes to be combined before the combination shows its worth, as in John *et al.*'s example. Our concatenation program can be applied several times, to yield 4-rules, 8-rules and so on. Using 4R, we are able to detect the pattern in the example that John *et al.* provide.

3 Conclusion

In an earlier paper [Holmes and Nevill-Manning, 1995], we demonstrated the efficacy of 1R as an attribute subset selection algorithm. However, we were not satisfied with two issues that arose: the quantization of continuous-valued attributes, and the handling of missing values.

In this paper we have addressed these issues by making changes to the original 1R algorithm. We have not proved conclusively that these changes are better, but initial experiments show considerable promise.

The treatment of missing values is something of a philosophical difference in approach. In practice, the effect of the two approaches is quite similar. For the quantization problem it should be possible to show improvements, at least experimentally, over the original. In point of fact, experimental evidence already exists to show that the original method is not ideal [Dougherty, Kohavi and Mahsami, 1995].

Our quantization method relies on finding maxima of the second derivative of the tradeoff curves. It is not clear that this can be reliably determined given that “random” curves are common in most real-world datasets. We will be performing further studies which will aim to try to detect the knee points reliably.

Finally, we presented an extension to 1R which helps to avoid the problem of making greedy decisions early in attribute selection. The complexity of the 1R algorithm is O(n) for 1-rules (n attributes) and O(n^2) for 1-rule pairs. This could prohibit its use as a practical tool. Our current implementation represents something of a brute-force approach, and we intend to spend time in the future developing a more efficient algorithm.

References

Dougherty, J., Kohavi, R., and Sahami, M. (1995). Supervised and Unsupervised Discretization of Continuous Features. To appear in *Machine Learning: Proceedings of the Twelfth International Conference.* Morgan Kaufmann, San Francisco, CA.

Holte, R.C., Acker, L., Porter, B.W. (1989). Concept learning and the problem of small disjuncts. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (pp 813-818). San Mateo, CA: Morgan Kaufmann.

Holte, R.C. (1993). Very Simple Classification Rules Perform Well on Most Commonly Used Datasets. *Machine Learning, 11*, pp 63-90.

Holmes, G., and Nevill-Manning, C.G. (1995). Feature Selection via The Discovery of Simple Classification Rules. To appear in *Proceedings of International Symposium on Intelligent Data Analysis (IDA-95)*, Baden-Baden, Germany.

John, G., Kohavi, R., and Pfleger, K. (1994). Irrelevant Features and the Subset Selection Problem. In *Proceedings of the Eleventh International Machine Learning Conference*, pp 121-129. New Brunswick, NJ: Morgan Kaufmann.