Improved Decision Tree Algorithm: ID3$^+$

Min Xu[1,2], Jian-Li Wang[1], and Tao Chen[1]

[1] Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences,
Changchun 130033, China
[2] Graduate School of the Chinese Academy of Sciences, Beijing 100039, China
xumin200602@hotmail.com

Abstract. This paper proposed an improved decision tree algorithm, ID3$^+$. Through the performance of autonomous backtracking, information gain reduction and surrogate value, our method overcomes some ID3's disadvantages, such as preference bias and the inability to deal with unknown attribute values. The experimental results show that our method can competitively and efficiently solve the two problems. The first problems often leads to inferior decision trees, while the second limits ID3's applicability in real-world domains. And the method could be a good start for a more robust decision tree learning system.

1 Introduction

As one of the most popular concept learning methods, decision tree, with its many advantages over alternatives such as neural networks[9], Bayese learning, and nearest neighbors, is one of the most popular machine learning methods. Up to now, ID3[2] is one of the best known decision tree algorithms. The basic idea of ID3 family of algorithms is to infer decision trees by growing them from the root downward, greedily selecting the next best attribute for each new decision branch added to the tree. ID3 searches a complete hypothesis space and is capable of representing any discrete-valued function defined over discrete-valued instances. Effective and expressive as it is on many learning tasks, ID3 in its basic form still has some serve limits.

One shortcoming of ID3 is its inability to handle noisy data, which will lead to overfitting. Solutions of this problem include validation set pruning and introduction of some fuzziness. In fact, previous research reported good performance after these noise-tolerant techniques were employed with ID3. Assuming that the training data set is noise free and adequate, does ID3 always generate a correct decision tree? The answer is no. Preference bias of ID3 can generate inferior decision trees. The most widely used attribute evaluation method is *entropy-based information gain*. From the information theory point of view, it is an attempt to encode a class of randomly drawn member of the training set with smallest number of bits. The problem with these measures is that it is biased toward attributes that have many possible values. These attribute give ID3 an appearance of being good class predictors since they split the training data into perfectly classified partitions. But in many cases they are not. Besides information gain, there are other attribute evaluation

D.-S. Huang, K. Li, and G.W. Irwin (Eds.): ICIC 2006, LNCIS 345, pp. 141 – 149, 2006.

methods. To our knowledge all existent selection criteria proposed so far incur some kind of search bias.

The second shortcoming is that ID3 in the basic form misses opportunities in complex, real-world applications, though it is computationally efficient. One of ID3's non-incremental learning assumptions is that all attribute values are present in both the training examples and test instances. However, the situations that the values of some attributes are missing from training examples often take place. This could happen, for examples, when some sensors fail in the data collection process. Simply ignoring the examples with unknown values lead to bad decisions. ID3 is not supposed to be responsible for incorrect prediction for a test instance with unknown attribute values. But it should be capable of making a reasonable guess according to the non-missing feature elements of this instance, as most medical doctors would do toward a patient in similar circumstances.

The remainder of this paper is organized as follows. Section 2 surveys related work. Section 3 describes our proposed solutions to the above problems. Section 4 gives the results of our simple experiments. Section 5 summarizes our conclusions and discusses future work.

2 Related Work

A large variety of extensions to the basic ID3 algorithm has been developed by different researchers. These include methods for post-pruning trees, handling real-valued attributes, accommodating training examples with missing attribute values, incrementally refining decision trees as new training examples become available, using attribute selection measures other than information gain, and considering costs associated with instance attributes. Several ID3-based systems have been built [3-7].

To the best of our knowledge, there is no previous work that attacked the preference bias problem of ID3 through general backtracking. Dealing with missing attribute values is usually addressed by *incremental learning* systems. These systems try to avoid rebuilding an entirely new decision tree when a new training example arrives. Instead, they change only 'faulty' parts of the learned knowledge base to accommodate the new observations.

Schlimmer's STGGER [6] system represents concepts as a probabilistic summary of important concept subcomponents. Motivated by the fact that real-world applications require resistance to noise, STGGER does not insist on perfect consistency between the decision tree and the training environment, nor does it make abrupt repairs following each misclassification. Experiments demonstrate that probabilistic representations and a conservative revision strategy enables STGGER to deal effectively with noise.

A system that appears quite different than STGGER at a cursory level, but that draws important principles from it, is Schlimmer and Fisher's ID4[5]. At the core of incremental ability of ID4 is the observation that information gain evaluation needs not be computed directly from the set of training examples, but a probabilistic summary of the observations is sufficient. The general findings of Schlimmer and Fisher are that ID4 converges on decision trees equivalent in quality to ID3's, while the cost of updating a decision tree in ID4 is often lower than ID3 when the tree is large.

Utgoff proposed several useful tree revision operators in ID5R[7] for efficient tree reorganization. The two most important ones are tree transposition and cutpoint slewing. When a different test attribute should replace the current one at a decision node, each non-leaf subtree is first recursively revised so that the new test occurs at the root of each subtree. Then the tree is transposed at the decision node. Cutpoint slewing is accomplished through checking the instances everywhere below the current node. Each instance that is in the wrong subtree is backed out to the current node by removing its information from the test along the way.

The most important goal for an incremental method is that its incremental cost be less than the cost of starting from scratch upon the arrival of a new observation. The disadvantage of this approach is that localized changes may result in a decision tree of lower quality. Thus, incremental learning methods trade quality against cost.

3 Proposed Solutions: ID3$^+$

To overcome the two problems discussed in Section 1, we first introduce autonomous backtracking to ID3 to reduce preference bias, then augment ID3 so that it can deal with unknown attribute values in induction and can make reasonable guesses for test instances with unknown attribute values. We named the improved decision tree learning system ID3$^+$.

3.1 Autonomous Backtracking

In section 1, we claim that ID3 may generate an incorrect decision tree even if the training set is self-consistent and adequate. Here are some definitions used in this paper.

Definition 1. A training set is *adequate* if there exists a decision tree such that every leaf of the tree is supported by at least one example in the training set.

Definition 2. A training set is self-consistent if there are no conflicting examples in it.

Intuitively, adequacy means that the training examples provide enough information for humans to be able to understand the instances in a verifiable way. If a leaf in a decision tree is not supported by any example, then it is anything but a sound reasoning because it is not variable by the training set. This kind of tree is not desirable. Rather, we would like each leaf of our decision tree to be reconfirmed by some of the training examples. If the training data were not collected well, people are unable to find a classification without making some guesses for some cases. This in fact is a problem with data collection. The learner is not to blame. The problem with ID3, however, is that even with an adequate training set, it is likely to make inferior splits due to its favor for shallowness over correctness of the tree.

```
*ID3 algorithm in [8]:
ID3(examples, attributes){
If all examples in one category,
    Then return a leaf node with this category as label
If attribute = $\Phi$
=>  Then return a leaf node with the most common category as label
A←the "best" decision attribute of attributes
For each value $V_i$ of A
    Let $examples_i$ be the subset of examples with value $V_i$ for A
    If $examples_i = \Phi$
=>  Then create a leaf node with the most common category as label
  Else
    Call ID3($examples_i$, attributes-A)
 }
```

When we examine the ID3 algorithm, we can find that every step is perfect except when it runs out of attributes or runs out of examples. ID3 uses voting to select some common category as label, as is marked with arrows in the pseudo-code. Since nodes far away from the root typically have a small number of examples, choosing the most common category from under this node does not make probabilistic sense. On the other hand, Choosing the most common category from the entire training set provides little information about a given node. These classification paths are more likely to be wrong than that terminate at "all examples in one category".

There are several possible reasons for an empty attribute set or training data set. One is inadequate attributes or inadequate examples, or both. This is essentially a problem with data collection. It cannot really be solved with any machine learning algorithm without asking for more features or more examples. Another possible reason, however, is bad earlier split due to search bias of ID3. With information gain as the attribute evaluation method, a short and wide tree is preferred over a deep and narrow tree, even if the former might be an incorrect one.

We do backtracking upon either realizing that there are no more attributes to divide the impure example set, or that there is no remaining example that takes this value of the attribute. These are actually the moments we realize that we must have made some incorrect splits earlier, assuming that the training set is adequate and self-consistent. The basic idea is that we don't guess if we can make induction without guess. We backtrack from the dead end and try another split, so we get the augmented algorithm as fellows

```
*ID3$^+$ algorithm:
ID3(examples, attributes)
{
If all examples in one category,
    Then return a leaf node with this category as label
If attribute = $\Phi$ or examples= $\Phi$
Then return NIL
loop:
A←the next "best" decision attribute of attributes
```

```
    For each value $V_i$ of A
        Let $examples_i$ be the subset of examples with value $V_i$ for A
        If ($ID3^+$($examples_i$ , attributes-A)=NIL)
              Then {Nullify subtree(A =$V_i$ )
            goto loop
                  }
  }
```

Note that this autonomous backtracking does not conflict with validation set backtracking. On the contrary, they complement each other. Validation set pruning is controlled backtracking since we need to force n elaborately selected validation set on the (potentially) overfitting decision tree. Usually *the validation set* simply consists of some of the training examples that could have been used in top-down induction. Validation set backtracking helps filter out noise in the training data and avoid coincidental regularities, while autonomous backtracking serves to reduce preference bias. There is no another set of pruning data involved in $ID3^+$ still respects the principles of ID3 so the backtracking is called *autonomous*. To avoid danger of infinite loops with the introduction of backtracking, the attributes need to be tried in a particular order at each node. By associating each node with a list of attributes ordered according their information gains, we can avoid infinite loops as well as repetitive computations of information gain when it backtracks to this node. By trying the attributes in a descending order with respect to their information gains, $ID3^+$ still respects the principles of ID3, though correctness is now put on a higher priority than making the tree wide and shallow.

It is the cases when multiple correct decision trees exist with some training data. In these cases $ID3^+$ simply chooses the one that best fits the information gain criterion. As long as the training examples are self-consistent and adequate, $ID3^+$ will eventually find a correct decision tree. We do not present a formal proof here for this due to space limitation, but the intuitive explanation is that by backtracking $ID3^+$ tries every possible tree if it has found evidence that the training set is adequate.

3.2 Dealing with Unknown Attribute Values

Unknown attribute values need to be taken care of twice in induction, i.e., attribute selection and individual case classification. Several methods have been explored for both phases. We implemented Quinlan's information gain reduction [1,3] during training and borrowed Breiman's surrogate splits [4] during testing. Originally these were results of research in incremental learning. We do not intend to make $ID3^+$ an incremental learner like ID4[5] or ID5R[7]. Delayed batch-style restructuring to accommodate new observations is acceptable in most situations, and usually generates a decision tree of higher quality.

3.2.1 Information Gain Reduction

When evaluating a test based on attribute A, Quinlan[1] reduces the apparent information gain from test set A by the proportion of cases with unknown values of A. The rationale for this reduction is that, if A has an unknown rate of x%, test set A will yield no information x% of the time. ID3+ includes this method to deal with unknown attribute values in training set.

3.2.2 Surrogate Value

By contrast, Breiman[4] tries to “fill in” the missing values of A before calculating the information gain of A. We borrow this idea of surrogate split and apply it to ID3$^+$ in test phase. ID3$^+$ examines all non-missing feature elements of the test sample and takes the values that are most probable according to other instances in the training set. Thus, the convenient value is surrogate value to the real one.

For example, the value for A_1, which is one of m attributes A_1, A_2..., A_m, is missing from a test instance i. We scan the training set and find that among the training examples whose values for the m-1 other attributes all match those of i, 15 examples have the value v_{11} for attribute A_1, 7 examples have the value v_{12}, 2 examples have the value v_{13}. We say that v_{11} is most likely to be the value for A_1 in test instance i. If none of the training examples matches instance i for all m-1 non-missing feature elements, we lower the likeliness threshold from m-1 to m-2. This search continues until no match vales can be found even when the likeliness threshold is as low as, say, $m/2$. At that time we decide that the class of this test instance is simply not predictable. Choosing an educated guess is already beyond the scope of this paper.

4 Experiment Results

In this section, we will give out some experiment to show the efficient and competitive of our algorithm ID3$^+$ to the ID3. Performance of autonomous backtracking is the accuracy that ID3$^+$ demonstrates on self-consistent training and test data. It is compared against the accuracy of original ID3 from [8]. For information gain reduction and surrogate value, performance is tolerance of ID3$^+$ to missing values.

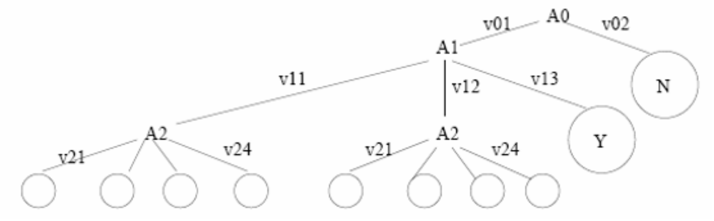
4.1 Autonomous Backtracking

To illustrate why ID3+ can outperform ID3 in terms of accuracy, we have built training set, ID3 trace file and ID3$^+$ trace file, and it is an adequate set of training examples which supports a three-layer tree as in shown in Figure 1(a). All training examples of (A_0=v_{01} &&A_1= v_{13}) are of class Y regardless of their values for attribute A_2.

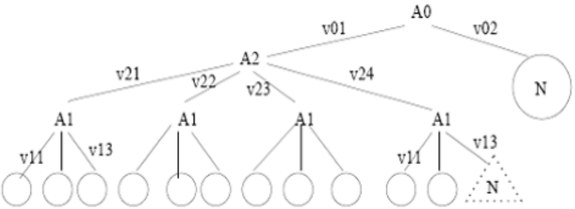
Figure1(b) is the decision tree generated by ID3. After splitting on attribute A_0, ID3 chooses A_2 as the best attribute over A_1 directed by information gain metrics on the branch of (A_0=v_{01}). This is because A_2 has 4 possible values and A_1 only has 3 possible values while both attributes partition the remaining examples equally well. As ID3 goes down (A_0=v_{01} &&A_2= v_{24}), there is not a training example such that (A_1= v_{13}) in this branch, which means ID3 runs out of examples and has to make up a tag for this leaf by guessing. Voting leads to *N* because of the dominating number of N-class examples in branch (A_0=v_{02}). Unfortunately this is a wrong label. In contrast to ID3, ID3$^+$ backtracks and tries partitioning upon attribute A_1 first. It finds that down stream (A_0=v_{01} &&A_1= v_{13}), though A_2 takes three values v_{21}, v_{22}, all remaining examples here are of class Y. So ID3$^+$ comfortably concludes with a tag of Y over this pure set of training examples in Figure 1(a).

How does ID3$^+$ perform compared to ID3 on larger data sets? We have chosen Voting, Students, Intelligent Cards and Weather to carry out the experiment and the numbers of their instances are enough to prove the difference of performance between ID3 and ID3$^+$.

The detailed experimental results are presented in the Table 1. The two rightmost columns are accuracies obtained from applying the programs on test data after training them on training data. Both ID3 and ID3$^+$ attain 100% accuracy on the Voting and Students data sets. The Evaluation of the IC Cards and the weather**,** as to Intelligent Cards, the accuracy of ID3 is 94.1% while ID3$^+$ is 99.5% accuracy and as to Weather, the accuracy of ID3 is 93.3% while ID3$^+$ is 99.5% accuracy. ID3$^+$ classifies all of them exceeding ID3. From this simple example we can see that ID3$^+$ outperforms ID3.



(a) The correct decision tree



(b) The incorrect decision tree

Fig. 1. The Difference of ID3+ and ID3

Table 1. Experiment Results of Autonomous Backtracking

data set	*training instances*	*test instances*	*attributes*	*ID3*	*ID3$^+$*
Voting	400,000	136	16	100%	100%
Students	20,000	320	22	100%	100%
Intelligent Cards	1,860	430	10	94.1%	99.5%
Weather	9,700	200	8	93.3%	99.0%

4.2 Information Gain Reduction and Surrogate Value

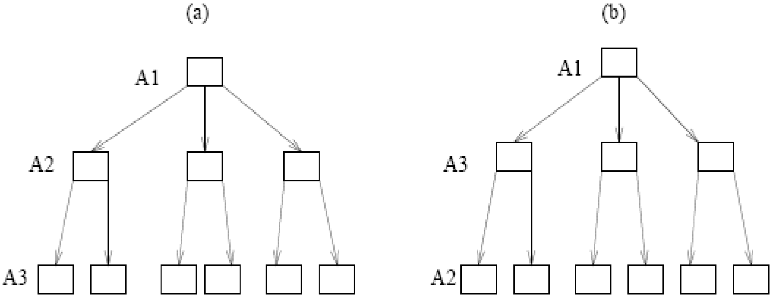
Information gain reduction and surrogate value are both based on probability estimate. We will illustrate how information gain reduction works and obtain performance numbers in the experiment.

Given the 12 training examples in table 2 without instance i$'$, information gains for all the three attributes A_1, A_2 and A_3 are the same, 0.000, since they all evenly split the training objects into two classes. The hierarchical decision tree is A_1—A_2—A_3, from top to bottom (Figure 4(a)). With instance $i^{'}$ added to the training set the information gains for A_1, A_2, and A_3 are now 0.007,0, and 0.004, respectively. The tree will be A_1—A_3—A_2 (Figure 4(b)). Reduced information gain for attribute A_2 moves it one level down the decision tree.

The experiment show that ID3$^+$ can deal with unknown attribute values in the expected way.

Table 2. Results of the 12 Training Examples

attributes instances	A1	A2	A3	
i1	v1	v21	v31	y
i2	v1	v21	v32	n
i3	v1	v22	v31	n
i4	v1	v22	v32	y
i5	v2	v21	v31	y
i6	v2	v21	v32	n
i7	v2	v22	v31	n
i8	v2	v22	v32	y
i9	v3	v21	v31	y
i10	v3	v21	v32	n
i11	v3	v22	v31	n
i12	v3	v22	v32	y
i'	*v1*	*	*v31*	*y*



(a) (b)

Fig. 2. Information Gain Reduction Example

5 Conclusions

In this paper, we have proposed an augmented decision tree, ID3$^+$. Experiment show that it is capable of being expanded to a useful tool, which outperforms ID3. With autonomous backtracking, ID3$^+$ prevents itself from getting stuck in a dead end caused by an earlier inferior split. With information gain reduction and surrogate value, ID3$^+$ can deal with unknown values in training and test set in a reasonable way. ID3$^+$ can efficiently solve the two problems with ID3, preference bias and the inability to deal with unknown attribute values. The first problem often leads to inferior decision trees, while the second limits ID3's applicability in real-world domains.

In future, we will farther study how to make ID3$^+$ noise tolerant and how to adapt ID3 and ID3$^+$ to continuous values for the attribute.

Another interesting future work is tree simplification to improve from subtree replication, in order to improve the space efficiency and understandability of the obtained decision tree.

References

1. Quinlan, J.: Unknown Attribute Values in Induction. International Conference on Machine Learning, (1989)
2. Quinlan, J..: Discovering Rules by Induction From Large Collections of Examples. Expert Systems in the Microelectronic Age, Edinburgh University Press, (1979)
3. Quinlan, J.: Programs for Mchine Learning. Morgan Kaufman, California, (1993)
4. Breiman, L., Nagy, G.: Decision Tree Design Using A Probabilistic Model, IEEE Trans. Information Theory, IT-30 (1984) 93-99
5. Schlimmer, J., Fisher, D.: A Case Study of Incremental Concept Induction. Proc. Fourth National Conference on Artificial Intelligence, (1986) 496-501
6. Schlimmer, J.: Concept Acquisition Through Representational Adjustment. UC Irvine, Dept. of Information and Computer Science, TR, 87-19 (2002)
7. Utgoff, P.: Incremental Induction of Decision Trees. Machine Learning, 4 (1989) 161-186.
8. Mitchell, T.: Machine Learning, McGraw-Hill, Singapore (1997) 52-78
9. Huang, D. S.: Systematic Theory of Neural Networks for Pattern Recognition. Publishing House of Electronic Industry of China, Beijing (1996)