재처리를 통한 결정트리의 정확도 개선

이 계 성†

요 약

기계학습은 효율적이고 정확한 재사용을 위해 지식을 재구성한다. 본 논문은 이미 알려진 학습 객체들로부터 지식을 추출하는 '예제에 의한 개념학습 방법'에 관한 연구이다. 대부분 학습 시스템은 처리와 표현에 대한 제약으로 인해 학습 결과를 새로운 객체에 적용할 때 효율성과 정확도가 기대에 못 미치는 경우가 있다. 본 논문에서는 ID3의 바이어스에 대해 조사하고, 다양한 표현 양식을 통해 보다 정확하고 학습적으로 이해하기 쉬운 분류 방법을 제안한다.

Improvement of Accuracy of Decision Tree By Reprocessing

Gyesung Lee†

ABSTRACT

Machine learning organizes knowledge for efficient and accurate reuse. This paper is concerned with methods of concept learning from examples, which glean knowledge from a training set of preclassified 'objects'. Ideally, training facilitates classification of novel, previously unseen objects. However, every learning system relies on processing and representation assumptions that may be detrimental under certain circumstances. We explore the biases of a well-known learning system, ID3, review improvements, and introduce some improvements of our own, each designed to yield accurate and pedagogically sound classification.

키워드 : 기계학습(Machine Learning), 지식표현(Knowledge Representation), ID3

1. Introduction

Learning organizes environmental observations into a knowledge base so as to efficiently and accurately perform some tasks. A widely used form of machine learning is learning from examples. This task assumes that objects are classified by a 'teacher' with respect to a number of object classes. The goal is to define conceptual rules that appropriately delineate the important properties of each class. Concepts allow future observations to be classified efficiently and accurately.

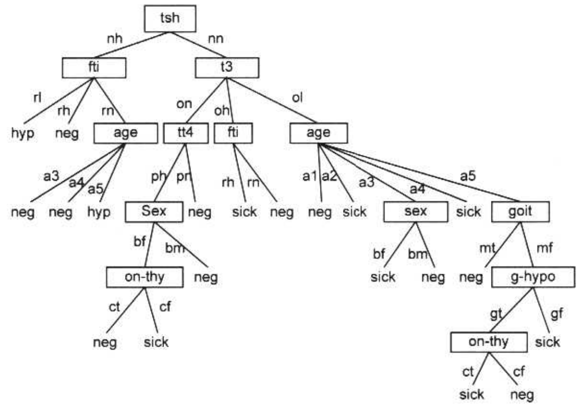
It is assumed that experts can enumerate features that might be relevant and classify examples framed in that featural language, more easily than they can supply rules that may not have been consciously or verbally accessed. Rather, machine learning is better suited to the task of focusing on that subset of the available features that yield a consistent description of expert-supplied examples.

※ 본 논문은 2002학년도 단국대학교 대학 연구비의 지원으로 연구되었음.
† 종신회원 : 단국대학교 컴퓨터과학 교수
논문접수 : 2003년 5월 2일, 심사완료 : 2003년 8월 14일

Despite the promise of machine learning, individual learning systems have built-in procedural and representational biases that may make autonomous knowledge acquisition difficult or impossible ; it is unreasonable to expect that the initial expert-supplied feature language is necessarily well suited to the biases of a given system. This paper investigates the biases of a well-known machine learning system, ID3 [1]. We describe augmentations to the ID3 family of systems. In addition, we describe several alternative approaches that seek to overcome limitations of traditional learning from examples systems. We focus our attention on the degree to which an acquired knowledge base can accurately support classification.

2. ID3

ID3 organizes conceptual knowledge in the form of decision trees. A decision tree recursively divides observations into subsets according to their value along a single attribute. For example, consider the decision tree of (Figure 1). An object is classified via the tree by following a path defined by values matching those of the object. The object is classified by the class given at the leaf of the appropriate path.



(Figure 1) An example decision tree-thyroid diagnosis

The critical issue in decision tree construction is the selection of an attribute whose values will decompose environmental observations at each node. ID3 has traditionally applied an information theoretic measure to the training objects. Under the assumption that the training objects reflect environment wide distributions, the attribute whose values best predict class membership is selected as the root's divisive attribute.

The second critical issue in decision tree construction is when to terminate tree expansion. Early versions of ID3 delved down a path until all objects of the training subset that were classified by the path were members of the same class. However, this method was found to overfit the tree to the peculiarities of individual objects. The presence of noisy (incorrectly described or classified) training objects could detrimentally lead future classification astray. To mitigate these sensitivities, ID3 was altered to cutoff expansion when no attribute divided up the training subset in a statistically significant manner. In particular, a chi-square approximate function is used to determine when optimal rules are likely not to have arisen by chance. Presumably, this strategy screens spurious and irrelevant statistical relationships between attributes and classes during classification and prediction.

ID3 and its extensions yield accurate classifiers [2-4]. Moreover, the decision tree is learned efficiently and is an efficient post-learning classifier. However, there are serious problems with the straightforward approach as we have described. We turn to these problems next and to a variety of solutions that have been proposed.

3. Polythetic versus Monothetic Learning

ID3 is highly efficient, in large part because at each step in decision tree decomposition it only considers the relative merits of single attributes ; for this reason we term ID3 a monothetic learner. The best attribute is selected to decompose the tree at each step. In terms of the search paradigm that dominates AI and machine learning ID3 hill climbs through the space of decision trees. In this hill climbing search it uses very limited look ahead. A problem is that the best attribute considered in isolation may not be the best (or even close to it) when considered in combination with other attributes. The predictive merits of an attribute may only be exposed in the context of other known values.

While these may appear to be extreme examples, logical relationships that can confound ID3's limited lookahead are likely to arise in engineering applications, as is noise. Thus, we are in a quandary as to whether we should keep noise tolerant machinery that may also lead to premature termination of tree expansion. An obvious extension of course is to extend the lookahead from one attribute to many. Because we are basing expansion on many attributes we term this extension a polythetic learner. An obvious strategy would be to extend look ahead by some constant.

As to pruning insignificant subtrees, the polythetic, retrospective-pruning strategy of ID3 takes a step in the direction of consolidating noise tolerance with more extensive look ahead. Unfortunately, the approach only retrospectively prunes the irrelevant attributes at the bottom-most portion of the tree. Irrelevant attributes that were inadvertently selected near the beginning of tree construction (i.e., the most likely time for such selections) will remain. This may adversely affect prediction accuracy, but it also affects the comprehensibility of the tree to outside inspection ; it will appear strange to a knowledgeable user (and will be misleading to a novice user) that irrelevant attributes are being used to guide classification.

4. Decision Tree to Productions and Vice Versa

Problems with retrospective pruning have been extensively investigated in recent researches [5-7]. As before the decision tree is decomposed to uniclass leaves. However, it is then converted to a set of independent production rules : one rule for each path of the tree. Each rule is then independently pruned using a chi-square like measure. They report improved accuracy results using this strategy. In another variation, the tree is converted to rules and then determines whether each attribute value's presence or absence significantly impacts the distribution of objects that match the rule. In this manner, a significance filter is applied to each value of a path, not simply the bottom most. In addition, this strategy carries another advantage ; ID3 divides a tree by the attribute with the most predictive values on average, but particular values may nonetheless be unhelpful. Conversion to production rules allows the predictiveness of each value to be assessed independently of other values.

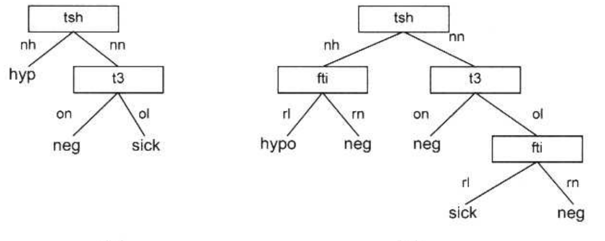
This procedure improves accuracy in noisy domains, without detracting (at least significantly) from accuracy in ideal (non-noisy) domains. However, there are disadvantages to flattening the decision tree in this manner. Most importantly, a production set is not as efficiently examined as a decision tree. The maximum number of decision tree comparisons is bounded by the total number of attributes, while the number of production set comparisons is bounded by the number of attributes times the average number of values per attribute (i.e., the length of each rule). In addition, a decision tree orders the attribute tests, a property that can be useful for guiding queries of an expert and otherwise exploiting commonality between rules. To maintain the accuracy of the pruned production set and the efficiency/pedagogical advantages of a tree structured knowledge base, we propose to return the rule set to a tree format. We describe two strategies for this process.

4.1 Rules to Tree via ID3

The gist of our strategy is to collectively treat the production rules as a training set of partially described 'objects'. However, the conversion from rule set to decision tree is not as straightforward as it might appear. This difficulty stems from the generalized nature of each rule. If an attribute value is not present it does not indicate that the value is missing or unobserved in the object description. Rather, it indicates that the attribute may take on any value (e.g., a 'dont-care' condition). In general, rules or partial rules are not mutually exclusive. For example, the tree of (Figure 3)(a) was created using the standard ID3 algorithm on the pruned rules of (Figure 2). But this tree would not accurately reflect the rules of (Figure 2). An incoming object may satisfy the top right-most test, while satisfying a rule in the left subtree.

Before pruning	
(((tsh nh)(fti ri))	(class hypothyroid))
(((tsh nh)(fti rh))	(class negative))
(((tsh nh)(fti rn)(age a3))	(class negative))
(((tsh nh)(fti rn)(age a4))	(class negative))
(((tsh nh)(fti rn)(age a5))	(class hypothyroid))
(((tsh nn)(t3 on)(tt4 ph)(sex bf)(on-thy ct))	(class negative))
(((tsh nn)(t3 on)(tt4 ph)(sec bf)(on-thy cf))	(class sick-euthyroid))
(((tsh nn)(t3 on)(tt4 ph)(sex bm))	(class negative))
(((tsh nn)(t3 on)(tt4 pn))	(class negative))
(((tsh nn)(t3 oh)(fti rh))	(class sick-euthyroid))
(((tsh nn)(t3 oh)(fti rn))	(class negative))
(((tsh nn)(t3 ol)(age a1))	(class negative))
(((tsh nn)(t3 ol)(age a2))	(class sick-euthyroid))
(((tsh nn)(t3 ol)(age a3)(sex bf))	(class sick-euthyroid))
(((tsh nn)(t3 ol)(age a3)(sex bm))	(class negative))
(((tsh nn)(t3 ol)(age a4))	(class sick-euthyroid))
(((tsh nn)(t3 ol)(age a5)(goitre mt))	(class negative))
(((tsh nn)(t3 ol)(age a5)(goitre mf)(g-hypo gt)(on-thy ct))	(class sick-euthyroid))
(((tsh nn)(t3 ol)(age a5)(goitre mf)(q-hypo gt)(on-thy cf))	(class negative))
(((tsh nn)(t3 ol)(age a5)(goitre mf)(q-hypo gf))	(class sick-euthyroid))
After pruning	
(((tsh nn) (t3 on) (tt4 pn))	(class negative))
(((tsh nh) (fti ri))	(class hypothyroid))
(((tsh nn) (t3 ol))	(class sick-euthyroid))
(((tsh nh))	(class hypothyroid))
(((fti rn))	(class negative))

(Figure 2) Rule sets before and after pruning



(a) (b)

(Figure 3) Tree-structured rule set

The tree building procedure for rules is a simple augmentation of the basic ID3 procedure, called ID3-R. Once the information heuristic selects a best divisive attribute, rules without the attribute are placed in each value subset. This is consistent with the idea that these rules can be satisfied by a test object regardless of its value along the test attribute. In addition, for purposes of determining the best divisive attribute, we consider the 'dont-care' attribute as a value. This occurs of necessity since over a collection of pruned rules, many attributes exhibit only one value. Using the standard evaluation function, a single value attribute would not be regarded as predictive. However, the presence of the value has predictive benefits that are not overlooked if we consider 'dont-care' as a value. 'Dont- care' is not part of the object description language in which test objects are expressed-thus, we do not use 'dont-care' as an arc label, but simply place a copy of each 'dont-care' rule with each value subset to be further distinguished. Thus, these rules are represented at deeper levels of each subtree, and can be recovered regardless of a test object's value at the top node. (Figure 3)(b) shows that the decision tree derived by ID3-R from the pruned rules of (Figure 2).

4.2 Rules to Tree via Conceptual Clustering

An alternative to using ID3-R is the use of conceptual clustering. In particular, we use CLUSTER/2 [8] and REIT [9] to organize the rules into a classification tree. Like ID3, CLUSTER/2 tree-structures a collection of objects in a set of mutually exclusive classes. However, it differs from ID3 in two important respects. First, they do not rely on an a priori classification of objects ; it discovers classes with 'best' conceptual descriptions characterizing objects of the class. Second, it forms a more general classification structure than ID3, since each arc may be labeled by a conjunction of attribute values and not simply a single value. However, like ID3, they are sensitive to missing attributes. In practice many attributes disappear after rule pruning. Thus we need to develop a method to handle missing attributes.

As with ID3-R, we treat a missing attribute as having a special value ('dont-care') for purposes of heuristic evaluation, but do not use these values in the final tree. Unfortunately, at the top levels of the tree, class characterization may be too general to discriminate between siblings because no meaningful attributes remain after removal of 'dont-care's. The tree is refined by removing meaningless nodes of the tree and by adding new links from the parent (root) to the meaningful descendants.

Unlike ID3, we do not place objects with missing 'value' in each subtree during building. Our use of multiple values at each node mitigates our need to do this. Nonetheless, it may be necessary to search over the forest until a concept is found that matches a test object.

REIT was developed based on the category utility of COBWEB. Incremental approach and greedy property of COBWEB is known to be detrimental to stable clustering due to the order dependency of data. It is also known that the COBWEB control structure and evaluation function are oriented toward maximizing predictive accuracy, and therefore the hierarchies it constructs may not reflect the underlying class structure of the data set. The REIT algorithm was developed to improve the drawbacks of COBWEB. The main idea of the REIT algorithm is to select the initial partitions from a classification tree and then introduce a partitive control structure to refine the partition structure. The main procedure of the REIT algorithm is summarized as follows. The procedure is recursively run until the node is composed of same class objects. The detailed description can be found in [9].

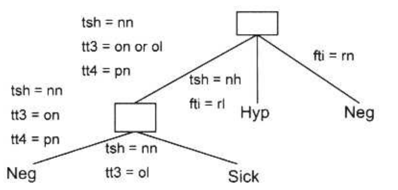
① Construct a COBWEB tree by using category utility.
② Select initial partition from the tree (k clusters)
③ Redistribute data objects over the partition by use of category match [10]
④ Compute new partition score ($\frac{1}{n}\Sigma CU_i$) for n clusters
⑤ If no improvement in partition score, then stop
⑥ Repeat the entire steps.

The hierarchy derived from REIT has node description on the link with multiple set of attribute value pairs. It is also different from CLUSTER/2 in that the occurrences of attribute values on the arc are represented with conditional probabilities. When experimenting with test cases to calculate the accuracy of the outcome, the probabilistic computation is required to determine the final prediction. This type of probabilistic approach for prediction over the test cases makes the algorithm more robust to the noises.

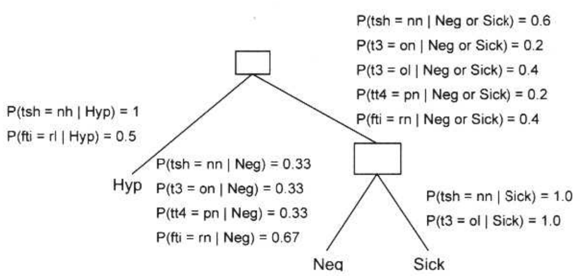
Modifications to the REIT algorithm were made to handle large numbers of systematic missing values. The computation of the category utility function is modified to account for missing attribute values in the data observations by redefining the terms $P(A_i = V_{ij})$ and $P(A_i = V_{ij} \mid C_k)$. $P(A_i = V_{ij})$ is redefined to (the number of objects observed for $(A_i = V_{ij})$)/(the number of object observed for A_i). Furthermore, a salience factor, a post-predictive measure, to represent the probability that a value for an attribute will be observed for a particular class is taken into account to the modified CU for cluster k :

$$P(C_k)(\sum_i P(\mathit{observed}\ A_i \mid C_k)\sum_j P(A_i = V_{ij} \mid C_k)^2 - \sum_i P(\mathit{observed} A_i)\sum_j P(A_i = V_{ij})^2)$$

(Figure 4) and (Figure 5) show the results of class hierarchy for thyroid data and congress data, respectively [11]. Due to the limited space, probabilistic description for REIT structure is removed in (Figure 5). The noticeable difference from the structure of ID3-R is that the arc labels become more complex and are described with different types of description languages. While CLUSTER/2 has arc labels with a set of conjunctive attribute values, REIT takes probabilistic values each illustrating the portion of attribute values in the node. Unlike ID3-like learner, there is no order as to the degree of importance of attributes. Therefore, there must be a mechanism to traverse the tree to find the right class for a new case. Especially, when a test case is fed into the REIT structure, the candidate nodes compete by probabilistic matching with the class descriptions shown in the arcs. The CLUSTER/2 structure is arranged in such a way that order of the search for classifying a new case should be done from most specific node to the general node.

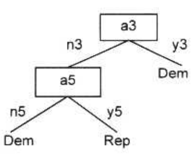


(a) CLUSTER/2 hierarchy

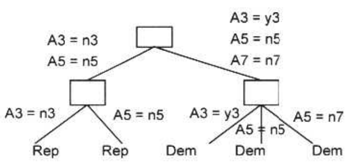


(b) REIT's hierarchy

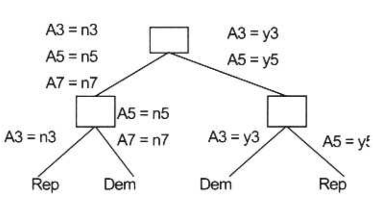
(Figure 4) Thyroid classifications



(a) ID3-R



(b) CLUSTER/2



(c) REIT

(Figure 5) Congress Classifications

The probabilistic description reflects not just the existence of the attribute values but also their strength in the group with respect to the predictability. For example, P(fti = rl | Hyp) = 0.5 means that the half of the node has the value 'rl' for attribute 'fti'. In this regard, the algorithm to search for the final prediction among possible candidates is as follows :

① initialize the sum to o.
② for each child of the node,
  ⓐ if corresponding attribute value matches the one of the test case, add up the probability to sum
  ⓑ otherwise, the probability is subtracted from the sum
③ pick up the best child node with highest sum
④ if the child has no child, return the class
⑤ else set the node to best child node and repeat from 1.

4.3 Accuracy of the algorithms

We have performed experiments for accuracy test. The results were obtained by first submitting a set of training objects to the ID3 system for thyroid and congressional domains [11], which returned a decision tree. Production rules were then generated and pruned. These were submitted to ID3-R, CLUSTER/2, and REIT, which created the corresponding decision and classification trees. The accuracy of the resulting prediction systems was evaluated with sets of unseen test objects (30 data objects for the thyroid domain, and 50 for the congressional domain).

<Table 1> shows prediction accuracy results for the initial ID3 tree, pruned production rules, and three tree-structured rules (ID3-R, CLUSTER/2, REIT).

〈Table 1〉 Accuracy for different algorithms

	ID3	rules	Pruned Rules	ID3-R	CLUSTER/2	REIT
Thyroid	.83	.83	.90	.92	.93	.93
Congress	.90	.90	.96	.94	.96	.96

The results show that the conversion of a decision tree into a set of production rules can be performed without loss of accuracy. But we expect that it is done at a high cost in efficiency due to the nature of rules. The pruning of the production rules produced a dramatic positive effect. Accuracy was improved by 6% (from 90% to 96%) for congress data and 7% (from 83% to 90%) for thyroid data. The conversion of rules to decision trees (ID3-R), and classification trees (CLUSTER/2, REIT) improved the accuracy by 2% for ID3-R and 3% for CLUSTER/2 and REIT for thyroid data.

But the gain in accuracy of REIT and CLUSTER/2 was preserved for congress data. Even a slight amount of gain was lost for ID3-R. Our experiment also shows a slight increase in accuracy of ID3-R for thyroid case, but that may be misleading because the tree created was unable to classify several objects, and only the objects for which a class was given (either correct or incorrect) were counted in the accuracy calculations. If those are counted the accuracy goes down than the pruned rules. We conclude that the clustering algorithms derived from pruned rules outperformed ID3, rules, pruned rules, and ID3-R. We have not fully investigated the efficiency issue. From the efficiency point of view, since a large amount of reduction is done in the rule pruning process the gain of efficiency is relatively small for rebuilt trees.

5. Conclusion

We have described several extensions to the basic ID3 algorithm with an eye towards accuracy and pedagogical advantages. Our preliminary extension to existing techniques is the conversion of a rule set to a tree structure. This conversion created different kinds of tree structures. Single attribute branching of ID3-R would be favored by novices in understanding the classification tree because of its simplicity. If some erroneous attribute appears in the top part of the tree, the accuracy would drop considerably. Ordering of single attributes may not always lead to a strong classifier. As we have seen, the polythetic learner would outperform monothetic learner. The merits of our approaches must await systematic experimentation in other larger domains to see if more significant gain can be realized. We also plan to do research on efficiency issue.

References

[1] Quinlan, J. R., "Induction of Decision Trees," Machine Learning, 1, pp.81-106, 1986.

[2] Utgoff, P., "Decision tree induction based on efficient tree restructuring," Tech. Report 95-18, Dept. of Comp. Sci. Univ. of Mass., Amherst, 1995.

[3] J. R. Quinlan. C4.5 : Programs for Machine Learning. Morgan Kauffmann, Los Altos, CA, 1993.

[4] Jensen, D., and Schmill, M., "Adjusting for multiple comparisions in decision tree pruning," Proc. Of 3rd Int. conf. On KDD, 1997.

[5] Frank, E., "Pruning Decision Trees and Lists," Ph.D. Dissertation, The University of Waikato, 2000.

[6] Elomaa, T. and Kaariainen, M., "An analysis of reduced error pruning," Journal of AI Research 15, 2001.

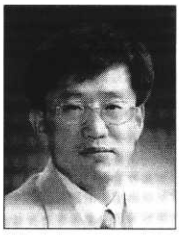
[7] Oates T. and Jensen, D., "The effects of training set size on decision tree complexity," Proc. 14th Inter. Conf. On Machine Learning, 1997.

[8] Michalski, R. S., & Stepp, R. E. "Learning from observation : Conceptual clustering," In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), Machine Learning : An artificial intelligence approach. Los Altos, CA : Morgan Kaufmann, 1983.

[9] Lee, G & Wu, X., "Ordering Data Set for Incremental Clustering," submitted to Int. Conf. On Machine Learning, 2003, Boston, USA.

[10] Fisher, D., "Iterative Optimization and Simplification of Hierarchical Clustering," Journal of AI Research, 4, pp.147-179, 1996.

[11] Blake, C. L. & Merz, C. J., UCI Repository of Machine Learning Databases, http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA : University of California, Department of Information and Computer Science, 1998



이 계 성

e-mail : gslee@dku.edu
1980년 서강대학교 전자공학과(학사)
1982년 한국과학기술원 전자계산학과(석사)
1994년 Vanderbilt University 전자계산학과(공학박사)
1994년~1996년 대구대학교 전산정보학과 전임강사
1996년~현재 단국대학교 컴퓨터과학 전공 부교수
관심분야 : 기계학습, 데이터마이닝, 바이오인포마틱스, 비디오마이닝