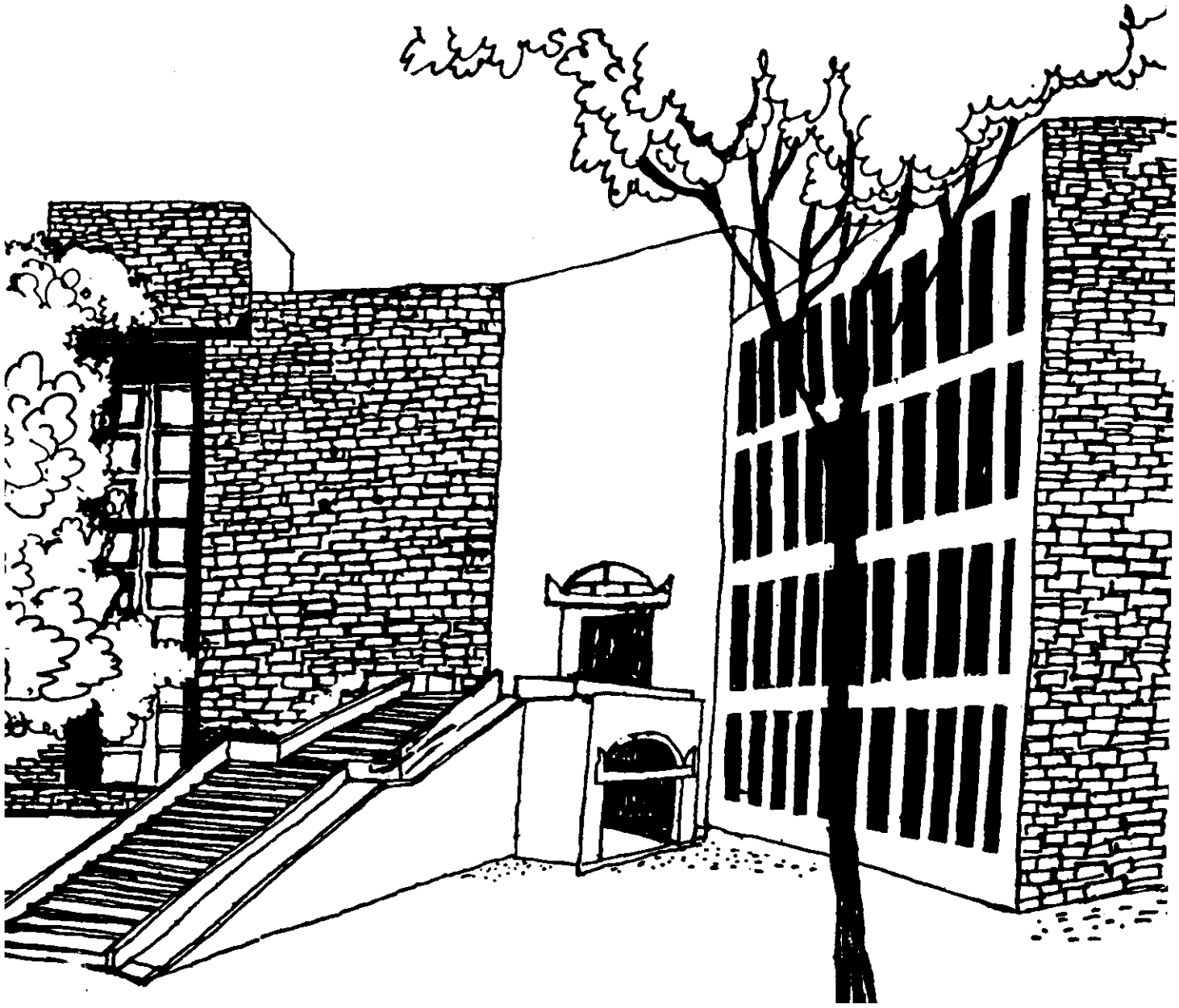
विद्याविनियोगाद्विकासः

IIM

AHMEDABAD

Working Paper



LEARNING CLASS DESCRIPTION FROM EXAMPLES USING A REFERENCE CLASS

By

S. Yegneshwar

&

S. Arunkumar

W P No. 1014
March 1992

INDIAN INSTITUTE OF MANAGEMENT
AHMEDABAD-380 015
INDIA

Learning Class Description from Examples Using a Reference Class

S.Yegneshwar* S.Arunkumar†

Abstract

An important problem of artificial intelligence is learning class description from pre-classified examples. The emphasis of some of the important learning systems such as ID3, INDUCE and CART is to discriminate each class from every other class. In many practical cases such descriptions are very inappropriate. In this paper, we describe a learning system that uses a reference description to learn each class description. The use of the reference description ensures learning of a class description that describes the class in addition to discriminating it from all other classes. Moreover, the description of each class is such that characterising attributes are specified before discriminating attributes. This is a major advantage over an earlier learning system called KAHLE. The reference and class descriptions learnt are shown to converge in the stochastic sense. The class description thus generated is simplified by dropping attributes which do not add to the description in any way. The importance of an attribute for a class is determined from this description. This is used in inference of a test example with missing attribute values. An inference process using the importance of attributes and based on category validity is used to classify test examples. The problem of characterisation of a democrat and a republican based on the machine learning database maintained at the University of California, Irvine is handled well by the proposed system. The results demonstrate that a better description is not at the expense of classification accuracy.

Keywords : Machine Learning, Induction, Knowledge Acquisition, Symbolic Learning from Examples, Case Based Learning.

*Computer and information Systems Group, Indian Institute of Management, Vastrapur, Ahmedabad 380 015, India. Tel: 0272-407241. email : yegnesh@iimahd.ernet.in

†Dept. of Computer Science and Engineering, Indian Institute of Technology, Bombay 400 076, India. Tel : 022-5782545. email : sak@dhruv.ernet.in

1 Introduction

A knowledge based system is a computer system which uses the knowledge of an expert stored in a suitable form to solve problems requiring significant expertise. The need for such systems is felt because experts are few in number and their knowledge is not available to many. The chief bottleneck in the development of knowledge based systems is the acquisition of knowledge from the experts. One of the popular methods of knowledge acquisition is from preclassified examples. The examples are past cases where the attributes (either symptoms or tests) and the corresponding values are recorded. All such examples with the right classification are used for knowledge acquisition. In this paradigm of knowledge acquisition, the description of each class (which is the required knowledge) is learnt by finding patterns in the given examples [Bund 85]. The pioneering domain independent learning systems are ID3 [Quin 79] and INDUCE [Mich 80]. Another well tested system is CART [Brie 84, Craw 89].

ID3 and CART generate a decision tree which helps classify a test example. It is clear that the emphasis of these systems is to generate a description which separates each class from every other class. Most of the extensions to ID3 [Nort 89, Mant 91] deal with alternative ways of selecting attributes at a node so that the size of the decision tree could be reduced. INDUCE finds the most general description of a class that is consistent with examples of all the other classes, i.e., a description generated for a class should not cover an example of any other class. The emphasis of finding a description which covers all the examples of a given class and examples of no other class is the same as finding a discriminating description.

The most important attribute in these learning systems are those which are most discriminating. Attributes whose values are common to examples of the classes would be considered irrelevant because these would not help in discrimination among a given set of classes. However, in practice there could be test examples which may not belong to any of the classes whose description is learnt. In such cases, a test example may wrongly get classified. For instance, given a set of examples for the class stool and the class chair (specified by the three attributes : backrest, noofpersons and function), ID3 would come up with a decision tree equivalent to : if backrest = present then class = chair and if backrest = absent then class = stool. In this case, the attributes : function and noofpersons are redundant for both the classes. This is not intuitively appealing since it is these two attributes alongwith the attribute backrest that actually describe the two given classes. Suppose now a test example with the following

(attribute,value) pairs is given : ((backrest, absent), (noofpersons, three), (function, sitting)). This example would be classified as a stool by the decision tree generated, though this is actually a bench and not a stool. If the attribute noofpersons had been used in some way while generating the description of the classes stool and chair from the learning examples, the test example would not have been wrongly classified.

An initial solution to such problems was a learning system proposed in [Arun 90]. In this system, the description of each class is learnt independent of the other classes. The strategy was to look for the most representative attribute at each stage. In the case of the stool and chair classes described above, this system could come up with the following descriptions when presented with the respective examples. An object with backrest = absent and noofpersons = one and function = sitting is a stool. Similarly, an object with backrest = present and noofpersons = one and function = sitting is a chair. The drawback of this system is that it may not (as in the present case) specify the attributes common to the group first. It may first specify attributes which are discriminating and then attributes which characterise the class. This is not intuitively appealing. An extension of this learning system which first learns the description of a reference class [Yeg 90] (where the reference class is the union of the learning examples of all the classes) before learning the descriptions of the elementary classes (i.e., stool and chair in the present context) is the subject of this paper.

The learning of reference class description is such that an attribute with a value common to most of the examples is considered more important than an attribute with diverse values among the examples. The former kind of attributes help characterise this group of classes and the latter kind of attributes help discriminate each class from the other classes in this group. In the stool and chair case, the attributes with a common value for most of the examples are: function (= sitting) and noofpersons (= one). The attribute which helps discriminate examples of one class from the others is backrest. Elementary class descriptions (i.e., that of stool and chair) is learnt using the reference class. The description of the class chair that would be generated by this learning system would be : An object with function = sitting and noofpersons = one and backrest = present. Similarly, the description of a stool would be : An object with function = sitting and noofpersons = one and backrest = absent. When the test example of a bench (specified earlier) is given, the system would not classify it into either of the classes because the the value of the attribute noofpersons is three and not one.

In the proposed learning system redundancy of an attribute is defined for each class. It is hence possible for an attribute to be relevant for a class and redundant for another class. For instance, given a set of examples (equal proportion of which falls into one of the cases listed below) for the classes stool and chair : ((function, sitting), (noofpersons, one), (backrest, present), (handrest, present) chair), ((function, sitting), (noofpersons, one), (backrest, present), (handrest, absent) chair), ((function, sitting), (noofpersons, one), (backrest, absent), (handrest, absent) stool). The attribute handrest is redundant for the class chair, because given that function = sitting and noofpersons = one and backrest = present, handrest could take any of the domain value, viz., present or absent. In other words, it does not matter what value handrest assumes and so it is redundant. For the class stool, the attribute handrest is not redundant because all stools must necessarily not have a handrest. It is clear from this rather simple case that an attribute can be redundant for a class and relevant for another class.

In this paper, we restrict to examples specified by binary valued attributes. In section 2, an algorithm to learn the description of the reference class is described. Section 3 deals with learning of the description of an elementary class using its examples and the reference class description. In this section, a definition of redundant attributes and a methodology to determine the importance of an attribute is described. Section 4 describes an inference process which uses the descriptions generated. In section 5, the results of applying the learning system to a commonly used database is discussed. Section 6 concludes the work with some directions for future research.

2 Learning Reference Class Description

The learning of the reference class description is enabled by the *maximum representation criterion* defined as one which maximises the number of examples (of the specified class) taking a particular value for an attribute (corresponding to the specified class). We define the Generation Tree (GT) below as the tree representation of a class each of whose nodes represents the attribute with respect to which the branching is done and whose directed arcs from root to leaves carry the attribute value along with the proportionate number of examples taking this value. The node attributes are selected using the *maximum representation criterion* at each

node. The structure of the reference tree is a binary tree. Note that in this tree, relatively more important attributes are selected higher up in the tree.

The concept of Generation Tree was proposed by S.Arunkumar and first reported in [Doct 85] In this proposal, a methodology to generate elementary class description using the *maximum representation criterion* is described for binary valued attributes. However, this proposal did not consider patterns when it did exist in certain cases. A modified version of this methodology (initially used for learning elementary class description and first reported in [Arun 90]) for generating the reference class description is described below.

2.1 Algorithm for Construction of GT

The GT is built using the criterion of maximal representation at each node.

1. Consider the given set of examples at the root node.

2. Select the most representative attribute for the examples at the current node. This is done as follows:

   (a) If both the binary values occur for the given attribute among the set of examples at the current node, then find the cardinality of both the sets, where all the examples in the first set have one value (say 0) for this attribute and all the examples in the other set have the complementary value for this attribute. Else, find the cardinality of the given set.

   (b) Find the maximum of the cardinality of the resulting sets (or set).

   Repeat steps (a) and (b) for all the attributes and select that attribute as most representative which has maximum cardinality as determined from step (b). In case of ties, the criterion is: select that attribute which comes earliest in the attribute sequence.

3. For both the values, branch off from the current node. At the left child node, consider all those examples having value 0 for further branching and at the right child consider all those examples having value 1 for further branching.

4. *Terminating Condtion :* If all the attributes have been selected along all the paths of the tree or if the frequency along all the leaves is less than a threshold called expansion threshold, then stop. Else, go to step 2 and proceed in a depth first fashion.

Note In this context, the *true description* of the class is the binary stochastic description based on the *maximum representation criterion.*

Theorem 1 *The GT for binary valued attributes generated by the above algorithm converges to the true description in the distribution sense.*

Proof *By Glivenko-Cantelli Theorem [DeGr 87],* $\mid F_n(x) - F(x) \mid \longrightarrow 0$, *with probability 1 uniformly in* x, *where* F_n *is the empirical distribution and* F *is the true distribution. This implies that for a given* $\epsilon > 0, \exists n(\epsilon)$ *such that* $|F_n(x) - F(x)| < \epsilon$, *with probability 1,* $\forall n > n(\epsilon)$. *If we assume that the joint frequencies are such that in the true description, there are no ties in the selection of attributes at all the nodes of the GT, we can choose* ϵ *such that for all* $n > n(\epsilon)$, *chosen suitably, the structure and the attribute at each node of the tree remains the same. Also, when this is true, the joint frequencies at each node stabilise.* □

Example 1 *Given the following two sets of examples belonging to two different classes* C_1 *and* C_2

$$S_1 = \{(1,0,1)\ \ (1,0,1)\ \ (1,1,0)\ \ (1,0,0)\ \ (1,0,0)\}$$
$$S_2 = \{(1,1,1)\ \ (1,1,1)\ \ (1,1,0)\ \ (1,1,0)\ \ (1,0,1)\}$$

At the root node attribute a_1 *is selected since* $\mid a_1 = 1 \mid = 10$. *The most representative attribute is hence* a_1. *At the next node attributes* a_2 *and* a_3 *are in contention. They are both equally representative since* $\mid a_2 = 1 \mid = 5$ *and* $\mid a_3 = 1 \mid = 5$. *Attribute* a_2 *is selected since in the case of ties the first attribute is chosen as per the algorithm. Proceeding thus, we get the GT shown in Figure 1.*

Note The GT corresponding to the reference class will hereafter be referred to as reference tree.

3 Learning Elementary Class Description

The description of an elementary class is learnt from its examples and the reference class description. This description represented as a binary tree has the same attribute sequence as the reference tree. This representation enables easy determination of redundancy and importance of attributes with respect to the given class.

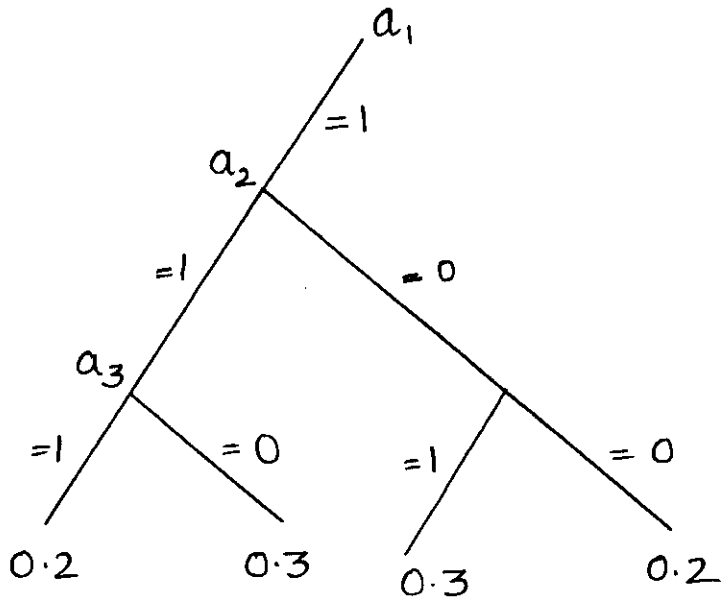


Figure 1: The GT of the Reference Class

3.1 Algorithm for Constructing GT of Elementary Class

1. Set the pointer to the root of the reference tree. The GT of current class C_t has a corresponding node with the same attribute at its root.

2. Consider all the examples of C_t at this node.

3. Label the current node of the tree corresponding to the class C_t with attribute a_r (note the initialisation corresponding to root in step 1).

4. Find the frequency of those examples in C_t at the current node having attribute value 1 and those having value 0. These frequencies are stored along the corresponding arcs to the child nodes. The attribute labels at the child nodes correspond to the label of the corresponding child nodes of the reference tree.

5. Repeat steps 3 and 4 corresponding to the examples of C_t which satisfy the attribute restrictions obtained in the path traversal, until the GT is fully constructed in consonance with the reference tree.

Example 2 *Consider the example sets of Example 1. The GT of the reference class for these example sets is as given in Figure 1. The GTs corresponding to the two classes are built using the attribute sequence in this reference tree as explained below : at the root node the attribute*

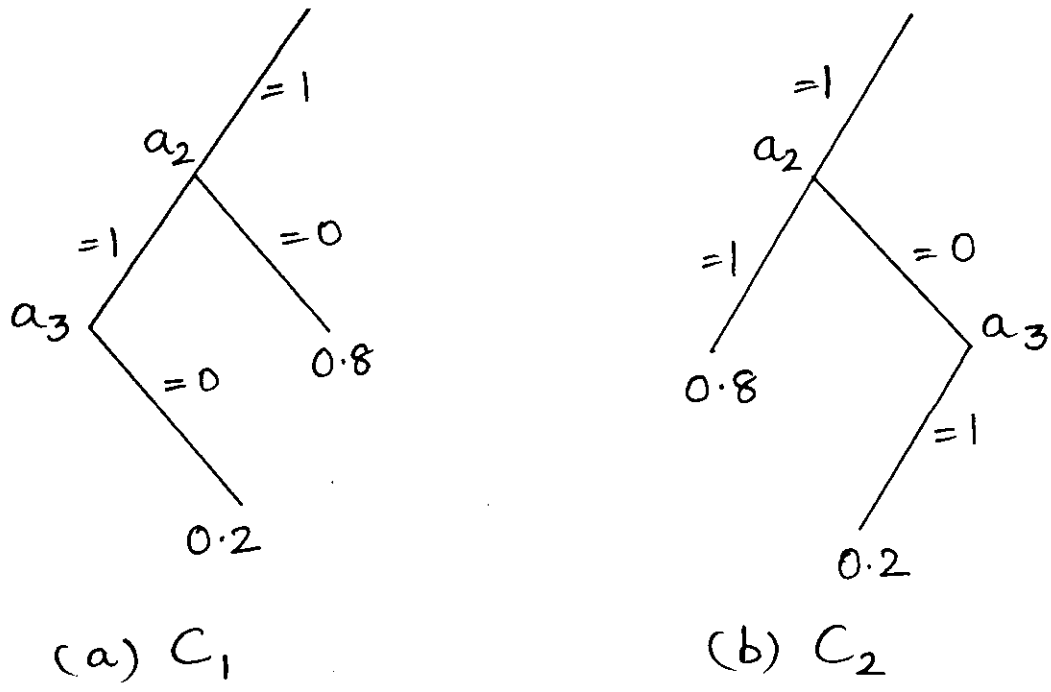


Figure 2: GT for class C_1 and class C_2

a_1 is selected and the frequency of examples assuming value 1 is evaluated. It is equal to 5. At the next node attribute a_2 is selected. Proceeding on the same lines we get the GT shown in Figure 2a for class C_1. Similarly, the GT of class C_2 is constructed. This GT is as described in Figure 2b.

By an argument along the lines of Theorem 1, we get

Theorem 2 *The GT for the elementary class converges to the true description of the class with respect to the reference tree.*

3.2 Redundancy of an Attribute at a Node of a GT

Determining whether or not an attribute is redundant and the elimination of redundancy leads to simplification of class description. This is achieved relatively easily in our framework. While the GT is constructed top-down, redundancy is determined bottom-up eliminating nodes at which the weights are approximately equal. The reason that the complete GT has to be constructed prior to the elimination of redundancy is because of the existence of dependencies, one of which is illustrated in example 3.

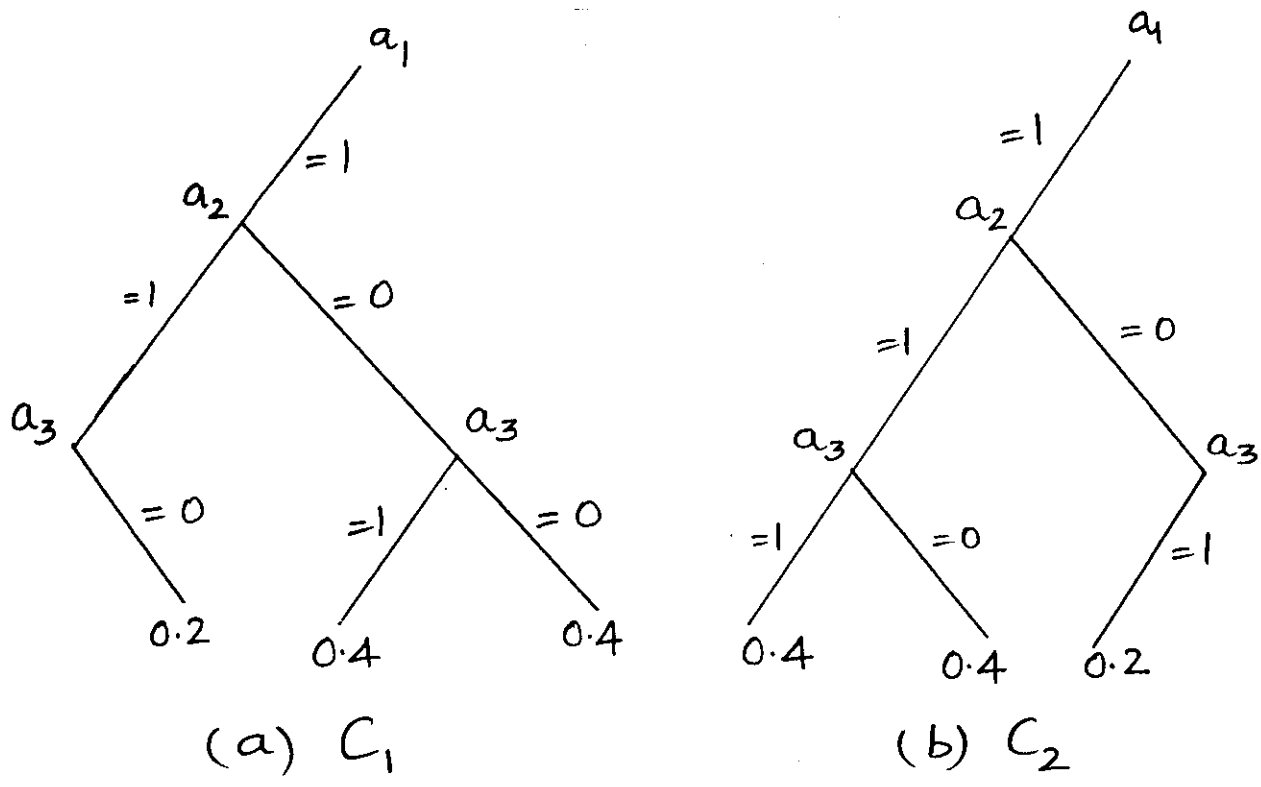


Figure 3: GTs of classes C_1 and C_2 after redundant nodes are removed

Definition 1 *An attribute is said to be redundant at a node if either both its descendants are leaf nodes or the attributes at all its descendant nodes are redundant and the frequency of its left and right arcs are equal.*

Definition 2 *An attribute is redundant for a class if it is redundant at all the nodes where it is chosen, i.e., it does not occur at any node of the GT after the redundancy check is applied.*

Example 3 *Consider the GT of class C_1 in Figure 2. Attribute a_3 at the bottom right hand side node is redundant at that node. This is because the frequencies at the left and right arcs at that node are equal. It can be seen from the GT corresponding to class C_2 in the adjoining figure that attribute a_3 is redundant at the bottom left side node for this class. The resultant GTs of both the classes are as shown in Figure 3.*

Note The equality is in the ideal case. In practice, however, if the percentage difference in frequency is less than a specified threshold, then that node can be deleted.

Definition 3 *A class is said to be a null class, if each element of the cartesian product of the domain of attributes occurs with equal frequency. In the GT framework, such a class should be represented by a single node.*

Note The definition of redundancy helps reduce the GT in such a case to a single node.

Note The redundancy of an attribute for a decision tree is based on the discriminative power, whereas in our system an attribute is redundant because it does not add to the description of the class.

3.3 Importance of Attributes

The importance of an attribute is the relevance of the attribute for a particular class. It is denoted as $I(a_r \mid C_x)$, read as importance of an attribute a_r given class C_x. This is relevant for inference of a new example to be classified at a later instance.

Since the importance of an attribute is judged by the representativeness of the attribute in the class, we consider the importance of an attribute given the class as directly proportional to the frequency at the node at which it occurs and inversely proportional to the distance of the node from the root.

$$I(a_r \mid C_x) \; \alpha \; f_r \tag{1}$$

$$I(a_r \mid C_x) \; \alpha \; 1/d_r \tag{2}$$

where

f_r is the frequency at the node where a_r occurs (this frequency is equal to the sum of the arc frequencies at this node) and d_r is the distance of the node from the root.

As a first approximation we have chosen the following form for $I(a_r \mid C_x)$ which satisfies eqns. (1) & (2).

$$I(a_r \mid C_x) = \sum_{p=1}^{t_r} A * (n - d_p) * f_p \tag{3}$$

and

$$\sum_{r=1}^{n} I(a_r \mid C_x) = 1$$

$$i.e., \quad \sum_{r=1}^{n} \sum_{p=1}^{t_r} A * (n - d_p) * f_p = 1 \tag{4}$$

where

n is the distance from leaf to the root node (i.e. equal to the number of attributes considered)

A is the normalising factor, and the summation is over all the nodes where a_r occurs. The normalising factor in this case would be equal to $\frac{2}{(n*(n+1))}$.

Example 4 *Consider the GTs in Figure 3. The importance of the three attributes a_1, a_2 and a_3 are evaluated as shown below :*

$$I(a_1 \mid C_1) = \frac{2}{(3*(3+1))} * \{(3-0)*1\} = 0.5$$

$$I(a_2 \mid C_1) = \frac{2}{(3*(3+1))} * \{(3-1)*0.2 + (3-1)*0.8\} = 0.33$$

$$I(a_3 \mid C_1) = \frac{2}{(3*(3+1))} * \{(3-2)*0.2\} = 0.033$$

$$I(a_1 \mid C_2) = \frac{2}{(3*(3+1))} * \{(3-0)*1\} = 0.5$$

$$I(a_2 \mid C_2) = \frac{2}{(3*(3+1))} * \{(3-1)*0.8 + (3-1)*0.2\} = 0.33$$

$$I(a_3 \mid C_2) = \frac{2}{(3*(3+1))} * \{(3-2)*0.2\} = 0.033$$

4 The Inference Process

The inference process finds the best match between the test example and the GTs of each class. This is done as follows:

1. Evaluate $f(e \mid C_i)$, the validity of class C_i using the GT corresponding to this class as shown below:

$$\begin{aligned} f(e \mid C_i) &= \{f(e_{p1} \mid C_i) * \sum_{l=1}^{n_1} I(e_l) + f(e_{p2} \mid C_i) * \sum_{l=1}^{n_2} I(e_l) + \\ &\quad f(e_{p3} \mid C_i) * \sum_{l=1}^{n_3} I(e_l) + ... + f(e_{pk} \mid C_i) * \sum_{l=1}^{n_1} I(e_l)\} \\ &= \sum_{j=1}^{k} (f(e_{pj} \mid C_i) * \sum_{l=1}^{n_j} I(e_l) \end{aligned}$$

where

$f(e \mid C_i)$ is the validity of class C_i given example e, pj is the jth path which the example

matches either completely or incompletely (incompletely if the example has missing values), e_{pj} is the set of (attribute, value) pairs along path pj, $f(e_{pj} \mid C_i)$ the validity of class C_i given e_{pj} is the leaf frequency along path pj, nj is the number of attributes along path pj for which attribute value is available in example e.

Note The validity defined above is based on the category validity of Medin , et.al.[Medn 87 which is defined as the probablity that an entity (or example) has some feature given that it belongs to a category (or class). We have used an extension of this which takes into account the absence of certain attributes. This has been done by weighting the category validity by the summation of the importance of only those attributes that are present.

2. Select that C_i for which $f(e \mid C_i)$ is the maximum.

Note It is quite straight forward to establish that the value of any $f(e \mid C_i)$ lies between 0 and 1.

Example 5 *Suppose we have two classes for which the GTs are as given in figure 3. Suppose the test example is* $S = \{(a_1 = 1)(a_2 = ?)(a_3 = 1)\}$ *(where ? implies that the corresponding attribute value is missing) then, we have,*

$$f(S \mid C_1) = 0.8 * (0.5) = 0.4$$

$$f(S \mid C_2) = 0.8 * (0.5) + 0.2 * (0.5 + 0.033) = 0.5066$$

Therefore, S is classified into class C_2. Similarly, it can be shown that the test example $S = \{(a_1 = 1)(a_2 = ?)(a_3 = 0)\}$ is classified into class C_1.

5 Congressional voting pattern problem

The proposed system has been implemented and tested on an application reported in the literature. A comparison of the proposed system is made with other learning systems in respect of classification accuracy (defined as the ratio of rightly classified to the total). Various runs of the proposed system are compared with respect to the classification accuracy and the number of sub-descriptions. The inputs to the proposed system are the learning examples, expansion

Class	freq. of learning examples	number of sub-des	freq. of rightly classd.	freq. of not classd.	freq. of wrongly classd.
1	108	20	53	1	6
2	124	25	131	3	9

Table 1: Frequency details of the experiment

threshold, sample size of each class, attribute names, and the test examples. For each class, the frequency of rightly classified, frequency of not classified, frequency of wrongly classified and the number of sub-descriptions are output. The results of the application is described in the following paragraphs.

The database is the 1984 Congressional voting pattern records consisting of two classes, viz. Democrats and Republicans. There are sixteen binary valued attributes for which each member has to vote. The total number of examples is 435 of which 232 has all the attribute values and 203 has at least one missing attribute value. The first set was used for learning and the second set for testing. The value of expansion threshold is 0.05. The results of the study are as described in Table 1.

The classification accuracy is equal to 92.5%. This compares favourably with [Schl 87] (where the accuracy reported is between 90% to 95%) and is marginally lower than that reported in [Arun 90] (where the accuracy reported is 94.5%).

Since the importance of attributes a_2 and a_{10} were zero for both the classes and the importance of attributes a_1, a_{11} and a_{13} were very low, they were dropped and the system was tested with the same expansion threshold. The number of sub-descriptions for both the classes after deletion of attributes is lower than for the above experiment and much lower than the sample sizes of the two classes. The number of terms per sub-description reduced by 2. Since most of the examples are distinct, the number of initial sub-descriptions is almost equal to the number of examples. The frequency details of the experiment given in Table 2 is exactly the same as that given in Table 1.

Class	freq. of learning examples	number of sub-des	freq. of rightly classd.	freq. of not classd.	freq. of wrongly classd.
1	108	15	53	1	6
2	124	21	131	3	9

Table 2: Frequency details after attributes a_1, a_2, a_{10}, a_{11}, & a_{13} are deleted

Class	freq. of learning examples	number of sub-des	freq. of rightly classd.	freq. of not classd.	freq. of wrongly classd.
1	108	14	55	1	4
2	124	20	130	3	10

Table 3: Frequency details after attributes a_1, a_2, a_{10}, a_{11}, a_{13}, & a_{15} are deleted

In addition to the already listed attributes if attribute a_{15} is also dropped (since its im portance is also relatively low), then the number of sub-descriptions reduces further and th classification accuracy increases marginally to 93%. This substantiates the fact that havin more than the optimal number of attributes tends to degrade the performance at times. Th details of this run is given in Table 3.

6 Conclusion

A domain independent model that learns elementary class descriptions using a reference clas for binary valued attributes was described in this paper. The need for a reference clas was emphasised for a case involving call for promotion. An important feature of this learnin system is that both the reference class description and the elementary class descriptions learn are shown to converge in the stochastic sense. The explicit definition of redundancy an importance of an attribute for a class is an added feature. The inference process defined t take care of missing attribute values is very useful in a practical environment. The empirica

study demonstrates that the proposed system's performance is good. In addition, the system learns a compact description without sacrificing performance.

The results of the chosen application is encouraging, but some more empirical testing is necessary to establish its generality. The proposed system is restricted to binary valued attributes in this paper. An extension for continuous and nominal valued attributes has been proposed [Yeg 90] and is the subject of the next paper. The approach adopted is the use of binary tree structure for both the reference and elementary class trees. Attribute values at a node is split into two ranges based on the maximum distance between the two clusters in this case. The rest of the analysis is the same as that of the binary valued case. The proposed learning system in this paper is non-incremental. A natural extension is development of an incremental version.

References

[Arun 90] Arunkumar S., and Yegneshwar S., "Knowledge Acquisition from Examples using Maximal Representation Learning", in the *7th International Machine Learning Conference*, Texas, USA, 1990.

[Brie 84] Brieman L., Friedman J.H., Olshen R.A., and Stone C.J., "Classification and Regression Trees", Belmont, Wadsworth, 1984.

[Bund 85] Bundy A., Silver B., and Plummer D., "An Analytical Comparison of Some Rule Learning Programs", *Artificial Intelligence*, vol.27, 1985, pp.137-181.

[Craw 89] Crawford S.L., "Extensions to the CART algorithm", *Intl. Jnl. of Man-Machine Studies*, vol.31, 1989, pp.197-217.

[DeGr 87] DeGroot M.H., "Probability and Statistics", Addison-Wesley Publishing Co. 1987

[Doct 85] Doctor M., "Knowledge Acquisition for Expert Systems", *BTech. Dissertation* Dept. of Computer Science and Engineering, Indian Institute of Technology, Bombay, India, 1985.

[Mant 91] Mantaras De Lopez R., "A Distance Based Attribute Selection Measure for Decision Tree Induction", *Machine Learning*, vol.6, 1991, pp.81-92.

[Medn 87] Medin D.L., Wattenmaker D.W., and Michalski R.S., "Constraints and Prefer ences in Inductive Learning : An Experimental Study of Human and Machin Performances", *Cognitive Science*, vol.11, 1987, pp.299-339.

[Mich 80] Michalski R.S. and Chilauski R.L.,"Knowledge Acquisition by Encoding Exper Rules vs. Induction from Examples : A case study involving Soybean Pathology" *Intl. Jnl. of Man-Machine Studies*, vol.12, 1980, pp.63-87.

[Nort 89] Norton S.W., "Generating Better Decision Trees", in Proc. of the Intl. Joint Conf on Artificial Intelligence, Morgan Kaufmann Publishers, 1989, pp.800-805.

[Quin,79] Quinlan J.R., "Discovering Rules by Induction from Large Collections of Exam ples", in *Expert Systems in the Micro-electronic age*, ed.Donald Michie, Edinburgh University Press, 1979, pp.168-201.

[Yeg 90] Yegneshwar S., "A Hierarchical Approach to Knowledge Acquisition from Exam ples", *Doctoral dissertation*, Dept. of Computer Science and Engineering, India Institute of Technology, Bombay, India, 1990.