STUDIA UNIV. BABEŞ-BOLYAI, MATHEMATICA, XXXVI, 3, 1991

A BINARY TREE CLASIFIER BASED ON FUZZY SETS

IOANA MARIA BOIER*

Dedicated to Professor P. T. Mocanu on his 60-th aniversary

Received: February 4, 1991
AMS subject classification: 68P99

Rezumat. Arbore binar de clasificare bazat pe mulţimi fuzzy. În această lucrare este descris un algoritm de proiectare şi implementare a unui clasificator binar. Acest algoritm îşi propune îmbunătăţirea algoritmului propus de Fu şi Mui [3]. O mulţime de date de test este utilizată în construcţia clasificatorului. În abordarea acestei probleme, Fu şi Mui folosesc proiecţia datelor în plan şi inspecţia vizuală ca metode de separare a clusterilor. Abordarea de faţă propune o separare automată, bazată pe mulţimi fuzzy.

The design of the binary tree classifier.

A method to design a binary tree classifier has been proposed in [3]. According to Fu and Mui, there are three major tasks to be implemented, to design a binary tree classifier:

a) a tree skeleton or hierarchical ordering of class labels

b) the choice of features at each nonterminal node

c) the decision rule to be used at each nonterminal node.

These tasks involve the specification of the following parameters:

a) the number of descendant nodes at each nonterminal node

b) the number of features used at each nonterminal node

c) an appropriate decision rule to be considered at each nonterminal node.

Since any conventional single stage classification scheme can be represented by a binary tree classifier which has exactly two immediat descendant nodes for each nonterminal node [3], we

* University of Cluj-Napoca, Faculty of Mathematics, 3400 Cluj-Napoca, Romania

consider the number of descendant nodes at each nonterminal node to be two. The next parameter to be specified is the maximum number of features used at each nonterminal node. This number depends on the specific classification problem and it is a constant for the problem. Let us denote it by K. To determine K, the number of all features, the size of test sample and the average number of samples per class are to be considered. The decision rule chosen at each nonterminal node is:

if $d(X,L^1) \leq d(X,L^2)$ then X is classified into calss A_1 (1)

otherwise X is classified into class A_2,

where X is the feature vector of the unknown sample to be classified, L^i is the prototype of the class A_i (i=1,2) [1] and d is a norm induced by distance in $\mathbb{R}^p$:

$d(x,y)=\|x-y\|$.

The next steps we have to perform are to design the tree skeleton or hierarchical ordering of class labels and to establish the actual features used at each nonterminal node. The fundamental problem which appears when the tree skeleton is built is the separation of the two groups of classes in each nonterminal node and the choice of features which are effective in separating these groups of classes. But, generally, the choise of the most effective features depends on the classes to be separated and the separation of the classes depends on what features are used. A method to break this deadlock is proposed in what follows. Using General Fuzzy Isodata algorithm [1] a fuzzy class is divided into two groups. Then, a method similar to the one presented by Fu and Mui [3] is used to chose the features which are "most effective" in separating the two groups of classes.

Let us assume that the predetermined number of classes is n and that the classes are labeled 1, 2,...,n. We also assume that the dimension of the features space is p. Suppose we have reached with the construction of the tree skeleton to a nonterminal node. Let C be the fuzzy set describing the membership degrees of class label i to this node, for all i from 1 to n. For example, at the beginning, when the nonterminal node is the root, the membership degrees are $C(i)=1.0$ for all i from 1 to n. Further, using the General Fuzzy Isodata algorithm, a fuzzy partition [1] $P = \{A_1, A_2\}$ of C is detected. According to the definition of a fuzzy partition, we have:

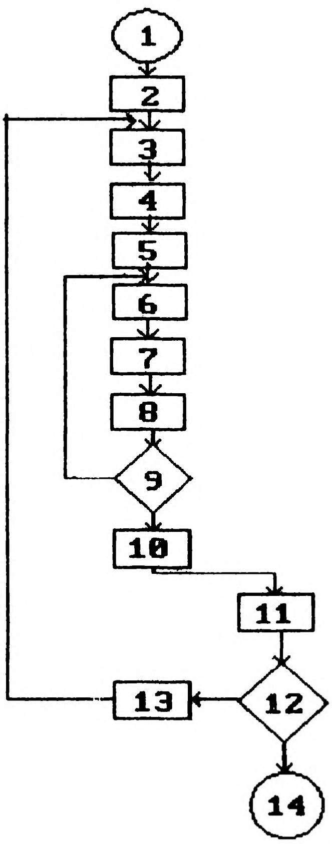
$$C(i) = A_1(i) + A_2(i) \;,\; i=\overline{1,n}$$

For the classification accuracy, the following correction rule is used:

$$\text{if } A_j(i) < 0.1 \text{ then } A_{2-j+1}(i)=C(i) \text{ and } A_j(i)=0.0 \;,\; i=\overline{1,n},\; j=1,2$$

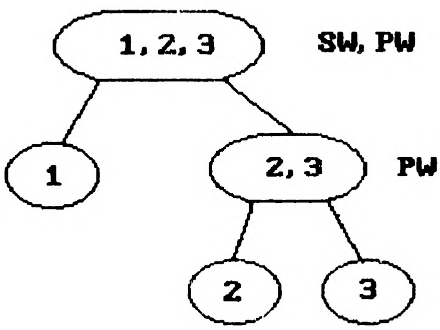
In determining the partition P, we use n feature vectors, representing the mean values of the features for each of the n classes. However, it is possible that not all the p features are needed to split the class C into A_1 and A_2. Using the set of test samples, we shall find the "best" up to K features in separating the two groups of classes. First, the best single feature is selected an this feature is used to perform classification based on the decision rule [1]. The result of the classification is computed and represents the number of test samples well classified. The "best 2" up to the "best K" feature subsets are obtained. The feature subset which give the best classification result of the K "best" feature subsets is chosen as the feature subset for the node considered. When an unknown sample to be classified reaches this node and we use the decision rule to go further, only those features from the feature vector of the unknown sample which correspond to the feature subset associated with the current node will be considered in order to compute the distances to the prototypes of the descendants.

The flowchart which describes the binary tree classifier design process is given below:



1. Start
2. Find the mean values of the features for each of the n classes
3. Obtain separable clusters using General Fuzzy Isodata algorithm
4. If needed, use the correction rule
5. $l = 1$
6. Find the "best l" features
7. Perform classification using these l features
8. $l = l + 1$
9. Is $l > K$?
10. Find the best classification result from the result corresponding to each of the K "best" feature subsets
11. Use the best result obtained to build up the decision tree
12. No new nonterminal node?
13. Get a new nonterminal node
14. Stop

Results. The method described above has been used to design a binary tree classifier for the classification of 147 samples of Iris spread over 3 classes [2]: Iris Setosa, Iris Virginica and Iris Versicolor. There are 4 characteristics taken into consideration: petal width (*PW*), petal length (*PL*), sepal width (*SW*), sepal length (*SL*). Considering for each of the 3 classes the mean values of the 4 characteristics listed below and the set of test samples as consisting of the first 20 samples from each class listed in Anex *A*, the following tree classifier is obtained (K=2):



Although all the lass labels (1,2,3) appear in each node, only those which have the membership degree to the node i (i=1,...,5) greater than have been represented for the node i in the figure above. Beside each nonterminal node is the set of features used.

	Setosa	Versicolor	Virginica
PW	0.2	1.4	2.5
PL	1.4	4.7	6.0
SW	3.5	3.2	3.3
SL	5.1	7.0	6.3

mean values of the 4 characteristics

The classification results are as follows:

	samples no.	well classified	percent
Setosa	49	49	100%
Virginica	49	26	51.02%
Versicolor	49	49	100%
Total	147	127	85.03%

R E F E R E N C E S

1. D.Dumitrescu, *Teoria clasificării*, Universitatea Cluj-Napoca, Facultatea de Matematica, 1991.
2. R.A.Fischer, *The Use of Multiple Measurements in taxonomic problems*, Ann. Eugenics, 7, 179-188, 1936.
3. K.S.Fu and J.K.Mui, *Automated Classification of Nucleated Blood Cells Using a Binary Tree Classifier*, IEEE Transactions on Pattern Analysis and Machine Intelligence", vol. pami-2, no.5, September 1980.

Anexa A

Iris Setosa				Iris Versicolor				Iris Virginica			
SL	SW	PL	PW	SL	SW	PL	PW	SL	SW	PL	PW
5.1	3.5	1.4	0.2	7.0	3.2	4.7	1.4	6.3	3.3	6.0	2.5
4.9	3.0	1.4	0.2	6.4	3.2	4.5	1.5	5.8	2.7	5.1	1.9
4.7	3.2	1.3	0.2	6.9	3.1	4.9	1.5	7.1	3.0	5.9	2.1
4.6	3.1	1.5	0.2	5.5	2.3	4.0	1.3	6.3	2.9	5.6	1.8
5.0	3.6	1.4	0.2	6.5	2.8	4.6	1.5	6.5	3.0	5.8	2.2
5.4	3.9	1.7	0.4	5.7	2.8	4.5	1.3	7.6	3.0	6.6	2.1
4.6	3.4	1.4	0.3	6.3	3.3	4.7	1.6	4.9	2.5	4.5	1.7
5.0	3.4	1.5	0.2	4.9	2.4	3.3	1.0	7.3	2.9	6.3	1.8
4.4	2.9	1.4	0.2	6.6	2.9	4.6	1.3	6.7	2.5	5.8	1.8
4.9	3.1	1.5	0.1	5.2	2.7	3.9	1.4	7.2	3.6	6.1	2.5
5.4	3.7	1.5	0.2	5.0	2.0	3.5	1.0	6.5	3.2	5.1	2.0
4.8	3.4	1.6	0.2	5.9	3.0	4.2	1.5	6.4	2.7	5.3	1.9
4.8	3.0	1.4	0.1	6.0	2.2	4.0	1.0	6.8	3.0	5.5	2.1
4.3	3.0	1.1	0.1	6.1	2.9	4.7	1.4	5.7	2.5	5.0	2.0
5.8	4.0	1.2	0.2	5.6	2.9	3.6	1.3	5.8	2.8	5.1	2.4
5.7	4.4	1.5	0.4	6.7	3.1	4.4	1.4	6.4	3.2	5.3	2.3
5.4	3.9	1.3	0.4	5.6	3.0	4.5	1.5	6.5	3.0	5.5	1.8
5.1	3.5	1.4	0.3	5.8	2.7	4.1	1.0	7.7	3.8	6.7	2.2
5.7	3.8	1.7	0.3	6.2	2.2	4.5	1.5	7.7	2.6	6.9	2.3
5.1	3.8	1.5	0.3	5.6	2.5	3.9	1.1	6.0	2.2	5.0	1.5
5.4	3.4	1.7	0.2	5.9	3.2	4.8	1.8	6.9	3.2	5.7	2.3
5.1	3.7	1.5	0.4	6.1	2.8	4.0	1.3	5.6	2.8	4.9	2.0
4.6	3.6	1.0	0.2	6.3	2.5	4.9	1.5	7.7	2.8	6.7	2.0
5.1	3.3	1.7	0.5	6.1	2.3	4.7	1.2	6.3	2.7	4.9	1.8
4.8	3.4	1.9	0.2	6.4	2.9	4.3	1.3	6.7	3.3	5.7	2.1
5.0	3.0	1.6	0.2	6.6	3.0	4.4	1.4	7.2	3.2	6.0	1.8
5.0	3.4	1.6	0.4	6.8	2.8	4.8	1.4	6.2	2.8	4.8	1.8
5.2	3.5	1.5	0.2	6.7	3.0	5.0	1.7	6.1	3.0	4.9	1.8
5.2	3.4	1.4	0.2	6.0	2.9	4.5	1.5	6.4	2.8	5.6	2.1
4.7	3.2	1.6	0.2	5.7	2.6	3.5	1.0	7.2	3.0	5.8	1.6
4.8	3.1	1.6	0.2	5.5	2.4	3.8	1.1	7.4	2.8	6.1	1.9
5.4	3.4	1.5	0.4	5.5	2.4	3.7	1.0	7.9	3.8	6.4	2.0
5.2	4.1	1.5	0.1	5.8	2.7	3.9	1.2	6.4	2.8	5.6	2.2
5.5	4.2	1.4	0.2	6.0	2.7	5.1	1.6	6.3	2.8	5.1	1.5
4.9	3.1	1.5	0.2	5.4	3.0	4.5	1.5	6.1	2.6	5.6	1.4
5.0	3.2	1.2	0.2	6.0	3.4	4.5	1.6	7.7	3.0	6.1	2.3
5.5	3.5	1.3	0.2	6.7	3.1	4.7	1.5	6.3	3.4	5.6	2.4
4.9	3.6	1.4	0.1	6.3	2.3	4.4	1.3	6.4	3.1	5.5	1.8
4.4	3.0	1.3	0.2	5.6	3.0	4.1	1.3	6.0	3.0	4.8	1.8
5.1	3.4	1.5	0.2	5.5	2.5	4.0	1.3	6.9	3.1	5.4	2.1
5.0	3.5	1.3	0.3	5.5	2.6	4.4	1.2	6.7	3.1	5.6	2.4
4.5	2.3	1.3	0.3	6.1	3.0	4.6	1.4	6.9	3.1	5.1	2.3
4.4	3.2	1.3	0.2	5.8	2.6	4.0	1.2	5.8	2.7	5.1	1.9
5.0	3.5	1.6	0.6	5.0	2.3	3.3	1.0	6.8	3.2	5.9	2.3
5.1	3.8	1.9	0.4	5.6	2.7	4.2	1.3	6.7	3.3	5.7	2.5
4.8	3.0	1.4	0.3	5.7	3.0	4.2	1.2	6.7	3.0	5.2	2.3
5.1	3.8	1.6	0.2	5.7	2.9	4.2	1.3	6.3	2.3	5.0	1.9
4.6	3.2	1.4	0.2	6.2	2.9	4.3	1.3	6.5	3.0	5.2	2.0
5.3	3.7	1.5	0.2	5.1	2.5	3.0	1.1	6.2	3.4	5.4	2.3