SLAC-PUB-1573 (Rev.)
CS-75-487
April 1975
Revised January 1976

A RECURSIVE PARTITIONING DECISION RULE FOR NONPARAMETRIC CLASSIFICATION *

Jerome H. Friedman
Stanford Linear Accelerator Center
Stanford, California 94305

ABSTRACT

A new criterion for driving a recursive partitioning decision rule for nonparametric classification is presented. The criterion is both conceptually and computationally simple, and can be shown to have strong statistical merit. The resulting decision rule is asymptotically Bayes risk efficient. The notion of adaptively generated features is introduced and methods are presented for dealing with missing features in both training and test vectors.

(Submitted to IEEE Transactions on Computers)

*This work supported by U.S. ERDA under contract AT(043)515

Introduction

In many classification problems, the underlying class conditional probability densities are either partially or completely unknown. Consequently, the classification logic must be designed from information measured from representative samples drawn from each class. The nonparametric classification problem may be stated in the following manner. A random p-dimensional vector of observed features, $\vec{X}$, is thought to belong to one of M populations, $\pi_1, \pi_2 \cdots \pi_M$, characterized by density distributions that are unspecified. On the basis of these features, a decision is made as to which distribution function characterizes $\vec{X}$, using a training set of vectors drawn from each of the populations, $\pi_1, \pi_2 \cdots \pi_M$.

The nonparametric decision rules that have received the most attention are the k-nearest neighbor decision rules first introduced by Fix and Hodges [1,2]. The training samples from the M populations are combined into a single population with each vector tagged as to the class from which it originated. The k closest training vectors to $\vec{X}$ (with respect to a specified distance function and metric) are located, and $\vec{X}$ is assigned to the class with the largest representation in this set. These authors investigated the rule for $k \to \infty$ and showed that the procedure is asymptotically Bayes risk efficient, if k is chosen to be a function of the training sample size, N, such that $\lim_{N \to \infty} k(N) = \infty$, while $\lim_{N \to \infty}[k(N)/N] = 0$.

The rule for fixed k has been investigated by Cover and Hart [3]. They show that for the extreme case of k=1 (nearest neighbor decision rule), the asymptotic probability of misclassification is bounded from above by $R^*[2 - MR^*/(M-1)]$ where R^* is the Bayes probability of misclassification.

Despite their desirable statistical properties and intuitive appeal, the k-nearest neighbor decision rules have not found widespread application to classification problems. This is due, mainly, to their computational complexity. Although considerable progress has been made recently in this regard [4], finding the nearest neighbors to a point in p-dimensional space is relatively expensive computationally. Techniques for using the full training sample to extract a subset of points with relatively high discrimination information have been proposed [5,6]. The k nearest neighbor rule is then applied to this reduced subset.

Another problem with the decision rules discussed above (as well as almost all others) is that they lack an invariance that is intrinsic to the classification problem, namely invariance under all strictly monotone transformations of the feature axes. The maximal invariants are the coordinate-wise ordered population levels of the training sets. Unfortunately, the performance of these decision rules can depend greatly on the choice of a particular transformation. Feature subset selection and choice of metric are examples of trying to find good linear transformations. The optimum transformation, however, may not be linear. For example, a metric that is good in one region of the feature space may not be good in another. A feature subset that contains a great deal of discriminating information in some regions of the space may contain little or none in other regions. Discovering the best nonlinear transformation of the feature axes for a particular decision rule and training data sample is a difficult problem and no general solutions have yet been proposed.

An alternate approach is to design the decision rule so that it contains the desired invariance properties. Anderson [7] presents decision rules based on statistically equivalent blocks or distribution free tolerance regions. These rules partition the multivariate feature space on the basis of a set of prespecified functions. Although these rules possess the desired invariance and can be shown to be asymptotically Bayes risk efficient, they may be no more useful than random assignment for moderate sample sizes.

Henrichon and Fu [8] and Meisel and Michalopoulos [9] present heuristic strategies for feature space partitioning based directly on the class identities of the training samples. These strategies recursively partition the marginal distributions of training sample subsets. These subsets are obtained from previous such partitionings. At each stage, the number of partitions, their location, and the particular feature used for the partitioning is decided, using a heuristic measure of the misclassification rate based on the training sample identities. These decision rules maintain the desired invariance to all monotone transformations of the features. Although asymptotic results are not available concerning their Bayes risk efficiency, empirical evidence and common sense indicate that they can perform well with moderate training sample sizes. In addition, Meisel and Michalopoulos observe that these partitionings can be represented by binary decision trees. They apply a dynamic programming technique for finding the decision tree that tends to minimize the average number of comparisons required to arrive at a decision, given a particular partitioning of the feature space.

This note proposes a different criterion for driving the recursive feature space partitioning algorithms of Henrichon and Fu, and Meisel and Michalopoulos. This criterion is especially simple and is motivated directly from considerations of Bayes risk efficiency. In fact, the decision rule that results can be shown to be asymptotically Bayes risk efficient with no assumptions concerning the underlying class probability densities [10]. Computationally, the procedure is quite fast both in the training and classification stages. Methods for using vectors with missing coordinates in both training and classification are presented.

Recursive Partitioning

Consider first the simplest case of only two classes (M=2). The decision rule for the multiclass problem will be seen below to be a natural extension of the two-class rule. Let $f_1(\vec{x})$ and $f_2(\vec{x})$ represent the (unknown) probability density functions of the two classes and $F_1(\vec{x})$ and $F_2(\vec{x})$ their corresponding cumulative distributions. Assume that the losses for misclassification are ℓ_1 and ℓ_2, respectively, and π_1 and π_2 are the corresponding prior probabilities. We make the restriction $\ell_1\pi_1 = \ell_2\pi_2$. Extensions to the general case are straightforward [10].

Suppose for the moment $F_1(x)$ and $F_2(x)$ are known univariate distributions. Stoller [11] shows that if one were to cut the real line at a point, assigning the left region to one class and the right to the other, the point x* that minimizes the Bayes risk of misclassification is the point that maximizes the quantity

$$D(x) = |F_1(x) - F_2(x)| \quad . \tag{1}$$

that is

$$D(x^*) = \max_x D(x) \quad . \tag{2}$$

The quantity $D(x^*)$ is the well-known Kolmogorov-Smirnov distance between the two distributions. In many situations, a single cut would not provide adequate discrimination; for example, if $f_1(x)$ and/or $f_2(x)$ were multimodel. In this case, the Stoller procedure could be extended by reapplying it to each of the two subintervals defined by the first partitioning, resulting in four cuts. This Stoller partitioning can be recursively applied to each interval defined by the previous partitioning, unless the interval meets a terminal criterion (depending on $F_1(x)$ and $F_2(x)$ in the interval) at which point the interval is not divided further. A terminal interval is called a <u>class one cell</u> if $F_1(x) > F_2(x)$; otherwise, it is called a <u>class two cell</u>.

The Kolmogorov-Smirnoff distance is a well-known measure of the separability of two distribution functions. A natural extension of Stoller partitioning to the multivariate case would be to cut on that feature for which the Kolmogorov-Smirnov distance between the two marginal class distributions is greatest. As with the univariate case, one could apply the partitioning recursively to each subpopulation until it meets a terminal criterion, at which time it is assigned to one of the two classes.

In nonparametric applications, the marginal cumulative distributions $F_1(x)$ and $F_2(x)$ are not known. However, they are easily estimated from the empirical cumulative distributions $\hat{F}_1(x)$ and $\hat{F}_2(x)$ by

$$\hat{F}_i(x) = \begin{cases} 0 & x < x_1^{(i)} \\ k/n & x_k^{(i)} \le x < x_{k+1}^{(i)} \\ 1 & x_n^{(i)} \le x \end{cases} \tag{3}$$

where $x_k^{(i)}$ is the kth point of the ith class with the points ordered in ascending values of x. Here n is the cardinality of the subsample under consideration.

A nonparametric recursive partitioning algorithm for two-class discrimination can proceed as follows. If the subsample meets the terminal criterion, it is assigned to one of the two classes. Otherwise, the Kolmogorov-Smirnov distance between the empirical marginal distributions of the two classes,

$$D(x_j^*) = \max_{x_j} |\hat{F}_1(x_j) - \hat{F}_2(x_j)| \quad , \tag{4}$$

is evaluated for each feature, j, in turn and the one for which $D(x_j^*)$ is largest is chosen as the one to be cut. That is,

$$D(x_{j*}^*) = \max_j D(x_j^*) \quad . \tag{5}$$

The location of the cut is taken to be x_{j*}^*.

Since the partitioning procedure deals only with marginal distributions, there is nothing that restricts it to the p-original features. Based on his knowledge of the problem, the researcher can manufacture new or transgenerated [8] features that are general functions of the original features. At each stage in the partitioning, the feature for which $D(x_j^*)$ is largest will be chosen. This maximization can be performed over all features, original and manufactured. The algorithm chooses the one that yields the best marginal discrimination at each stage of the partitioning. Features containing little or no discriminating information are simply ignored so that there is no loss in adding any number of extra transgenerated features. However, there is a great deal to be gained if one or several of these transgenerated features yield good discrimination for some of the partitioned subsamples.

It is not necessary that these additional features be manufactured in advance of the partitioning. They can be constructed as the partitioning progresses, and made dependent upon the particular subsample to which they are applied. For example, one might add the feature set

$$y_i = \vec{x} \cdot \vec{w}_i \tag{6}$$

where the $\vec{w}_i$ are the eigenvectors associated with the largest several eigenvalues of the matrix BC^{-1}. Here B is the between class scatter matrix and C is the within class scatter matrix for the particular subsample under consideration. Thus, the manufactured feature set can itself adapt to different subsamples and different regions of the feature space. In several applications, we have found it useful to add the single adaptive feature

$$y = x \cdot \hat{f} \tag{7a}$$

where

$$\hat{f} = [V_1 + V_2]^{-1} \cdot (\vec{m}_1 - \vec{m}_2) \tag{7b}$$

is the direction associated with the Fisher linear discriminant. Here $\vec{m}_i$ and V_i $(i=1,2)$ are the subsample mean and covariance matrices of the two classes. Although these generated features may be motivated by parametric considerations, they will be incorporated into the decision rule only if they are found to be useful on the basis of the nonparametric Kolmogorov-Smirnov criterion.

It should be noted that the addition of adaptively generated features can cause the resulting decision rule to no longer be invariant under all strictly monotone transformations of the original features. For those suggested above, however, the rule is invariant to linear transformations.

Terminal Criteria

It remains to specify the criterion that stops the partitioning of a subsample establishing a terminal cell. The partitioning should clearly terminate if the subsample contains training vectors only from a single class, since further partitioning cannot change any class assignments. One possibility is to make this the sole criterion for termination. This results in all of the training vectors themselves being correctly classified by the decision rule. However, this criterion is best only if it is known in advance that there is no overlap in the feature space between the underlying class probability densities. That is,

$$\int f_1(\vec{x})\, f_2(\vec{x})\, d\vec{x} = 0\ . \qquad (8)$$

When the probability densities do overlap, the optimal Bayes decision rule does not correctly classify all of the training vectors. Since the purpose of the nonparametric procedure is to use the training sample to estimate as closely as possible the Bayes decision boundary, requiring it to correctly classify all of the training vectors would degrade its performance in overlap situations.

The class assignment of terminal cells is made on the basis of the estimated density ratio f_1/f_2 within the cell. The cardinality of the subsample within each cell should be large enough to provide a reasonable estimate of this density ratio. Thus, the partitioning of a cell should terminate whenever it cannot be further partitioned in a way that insures at least k subsamples remaining in each of the two daughter cells. Here k is a preset absolute minimum subsample size for all terminal cells. Also, the maximum for the Kolmogorov-Smirnov distance (eqn 1) should be sought in the restricted range $x_{k+1} < x \leq x_{n-k}$ so that no cell can be created with less than k subsamples.

The minimum cell sample size, k, is a parameter of the algorithm. The best choice for its value is problem dependent. It should increase with increasing total sample size N, more slowly than N. Gordon and Olshen [10] prove that the recursive partitioning procedure described in this paper is asymptotically Bayes risk efficient provided

$$\lim_{N \to \infty} \frac{k(N)}{N} = 0 \qquad \text{and} \qquad \lim_{N \to \infty} \frac{k(N)}{\sqrt{N}} = \infty\ . \qquad (9)$$

A method is described below for estimating the best value of k for a particular problem from the training sample itself.

Implementation

Meisel and Michalopoulous [9] note that any partitioning of a coordinate space can be represented by a binary tree. They develop dynamic programming techniques for constructing the particular tree that tends to minimize the average number of comparisons required to arrive at a terminal cell. Their techniques can be applied to the partitioning that results from the algorithm described here.

Because the partitioning in this algorithm is binary at each stage, it is possible to directly build a representative binary tree as the partitioning progresses. A subsample at any stage in the partitioning is represented by a node of the tree. The root of the tree represents the entire training sample. The two sons of each nonterminal node represent the two subsamples defined by its partitioning. The terminal nodes of the tree represent the terminal cells. Each nonterminal node must store the feature number and split point used in its partitioning, as well as pointers to its two sons. If the feature used for splitting was adaptively generated from the subsample itself, then the parameters for generating the feature must be stored. Each terminal node stores the number of training vectors from each class contained in its corresponding terminal cell.

Classification Rule

The rule for classifying a test vector is simply to assign it to the class that the partitioning algorithm has assigned to the cell in which it lies. With the binary tree representation of the partitioning, this can be accomplished easily and quickly. Starting at the root, the test vector is directed down the tree until it arrives at a terminal node. If the terminal node represents a unique class, the test vector is assigned to that class. If the node represents mixed classes, then the test vector is assigned to the class with majority representation. While descending the tree, the decision to go left or right at each nonterminal node is made as follows:

$$\begin{array}{lll} \text{if} \quad x_{j*} \leq x^*_{j*} & & \text{go to left son} \\ \quad\text{else} & & \text{go to right son.} \end{array}$$

Here j^* is the partitioned feature (original, transgenerated, or adaptive) and x^*_{j*} the corresponding split point stored at the node.

Multiclass Discrimination

A straightforward extension of this technique to multiclass problems is to treat an M-class problem as a series of two-class problems. For each two-class problem, a recursive partitioning is performed to separate one of the class populations, i, from all of the others. In each terminal cell of each tree, the number of training vectors, C_i, of the particular class to be separated, and the number, O_i, corresponding to the other classes are stored. A test vector to be classified is directed down all M decision trees to M corresponding terminal cells. The test vector is assigned to the class j for which $C_j - O_j$ is maximum over these M cells.

Although it might appear that this procedure increases the complexity of the decision rule by a factor of M, this is not the case. For each of the M decision trees, the object is to separate a single class, i, from all of the others. Partitioning will occur only near the decision boundaries of class i. Training vectors from other classes not near the boundary will quickly be assigned to large cells containing no class i vectors during the very early stages of partitioning, and thus are removed from consideration in the later stages. Only those non-class i vectors near the class i boundary participate significantly in the partitioning of the feature space for each class i decision tree.

Missing Features

This decision rule can easily accommodate missing features in the test vectors as well as the training vectors. Missing features in a vector to be classified cause problems only if a feature that is not present is used for partitioning at a node on the path from the root to its terminal node. If this does not happen, then the vector will arrive at a unique cell in each decision tree and be classified in the usual way. If a node is encountered in which the discriminating coordinate is missing, then a decision as to which branch to take cannot be made, and the point is directed down both branches. This causes the test vector to ultimately appear in several terminal cells in each tree. The number of cells in which it will appear in each tree is one more than the number of ambiguous nodes it encounters. The vector is assigned to the class with the largest representation in the union of these cells.

Training vectors with missing features are handled similarly. Those vectors with their jth coordinate missing, simply do not participate in evaluation of the Kolmogorov-Smirnov distance for that coordinate. If the jth coordinate turns out to be the one chosen for partitioning, then those vectors missing that coordinate are included in both descendent subsamples.

Transgenerated or adaptive features are functions of the original measured features. One or several missing original features can cause many transgenerated or adaptive features to be uncalculable. If a great many partitioned features turn out to be of this manufactured type, simply taking both branches at each one encountered may discard too much discriminating information. An alternative at each such node would be to substitute for the missing original feature, a nominal value (for example, the mean) taken from the training sample <u>represented by that node</u>. Restricting the subsample to only that represented by the particular node in question, allows any dependencies that may exist in the training data between the measured features, to be used to advantage in estimating a nominal value for the missing feature.

Limitations and Extensions

There is an intrinsic limitation to the recursive partitioning described above (as well as those of References 8 and 9). This limitation is a direct consequence of the fact that information from marginal distributions only is used to drive the partitioning. Although they are unlikely to be encountered in practice, there are special situations in which this limitation can adversely affect the performance of the decision rule.

The general partitioning problem at a particular node in the decision tree can be described as follows: Given (A) the particular set of partitions that lead to the node (i.e., those defined on the path to it from the root) and (B) all possible subsequent partitionings in the subtree below it, choose the best feature and location for the cut at that particular node. The recursive partitioning algorithm described above uses the information only from part A. That is, it makes the best possible cut at each node (given the cuts leading to the subsample represented by the node) assuming that its two sons will be terminal. The procedure does not "look ahead" to all possible sequences of cuts choosing the first of the best sequence [12]. Thus, the resulting feature space partitioning is clearly suboptimal in a statistical sense.

A complete look ahead is not computationally feasible, even for very small sample sizes. However, a restricted L-level look ahead might be feasible for small to moderate training sample sizes. In this mode, each feature is provisionally cut as if it were the one with the maximum Kolmogorov-Smirnov distance. Each set of daughter subsamples are also each provisionally cut along all of the features in the same manner, and so on. The provisional partitioning is continued for L-levels or until nodes become terminal. All of the resulting partitioning sequences are evaluated and the best one is identified. (For p features, p^{L+1} is an upper limit on the number of such sequences). The original cut that leads to the best sequence is the one chosen. This L-level look ahead is restricted in that it looks for the best sequence of cutting features, but does not optimize with respect to cut locations. Each provisional cut is made at that point which maximizes the Kolmogorov-Smirnov distance. Computational considerations usually restrict L to be a very small number. Also, except in unusual situations, very little decrease in expected error rate is obtained by increasing L.

Computational Considerations

Computationally, the partitioning procedure described in the previous sections is quite fast, both in the training and classification stages. The computational requirements depend upon the minimum cell subsample size k and look-ahead level L employed, as well as the separability of the class populations. When the underlying class probabilities overlap very little and the decision boundary between them is relatively simple, the algorithm can quickly construct large cells containing training vectors from a single class. This considerably reduces the number of nodes in the decision tree.

A worst case occurs when there is no difference between the class probability densities. In this case, the partitioning algorithm constructs a random binary tree. Although no discrimination is possible in this situation, we can use it to estimate an upper bound on the average computation. It is well known that the average computation required to build a random binary tree is proportioned to W(N)log n, while the average search requires computation proportional to log n. Here n is the number of nodes in the tree and W(N) is the computation associated with each level in the tree. The number of nodes in the tree is N/k. Because sorting is required for each marginal distribution, the computation required at each level to select the partitions is approximately

$$W(N) \sim p^{L+1} N \log N,$$

so that the total average computation to build the tree is

$$C(p,N,L,k) \sim p^{L+1}(N \log N)\log(\tfrac{N}{k}). \qquad (10)$$

The average computation to descend the tree for classification of a test vector is simply proportional to log(N/k).(1) These calculations are quite crude and represent the average computation only for a worst case, namely, maximal overlap of the underlying probability densities. They do, however, give an indication of how the computational requirements depend upon the various parameters of the problem.(2)

Discussion

The principal difference between the recursive partitioning algorithm described here and earlier ones [8] [9] is the use of the Kolmogorov-Smirnov criterion for selecting both the feature and location for the cut at each stage in the partitioning. This criterion is both conceptually and computationally simple, and can be shown to have strong statistical merit [11]. The resulting decision rule can be shown to be asymptotically Bayes risk efficient [10]. The notion of adaptively generated features is introduced and methods are described for dealing with missing features in both training and test vectors.

In an operational sense, this method differs in that at each stage of the partitioning only a single cut is performed. At first thought, one might be motivated to make several cuts on a single marginal distribution, especially if there were several sequences of runs of a single class. The strategy of the recursive partitioning presented here is to make the single cut that yields the best marginal class separation. (Note that from its definition (eqns 1-3), the Kolmogorov-Smirnov criterion can cause a split only at boundaries between two runs and never within a run.) After the split, all of the marginal distributions of the two daughter subsamples are examined and each subsample is cut on its respective best feature. It may be that the same feature that was used to cut the parent subsample is also the best for cutting both daughters. If so, the Kolmogorov-Smirnov criterion will select it. However, making several cuts on a single marginal distribution presupposes that this is the case without examination of the other marginals after each cut.

An important by-product of purely binary partitioning is that a binary decision tree representing the partitioning can be easily constructed as the partitioning progresses.

There are two parameters associated with this algorithm, the minimum terminal cell sample size, k, and the look ahead level L. Computational considerations usually restrict L to a very small value (for example zero). As discussed above, very little is gained by increasing the level of look ahead, except in special situations. One strategy would be to invoke a look ahead only at those nodes in the decision tree where none of the single marginal distributions provide adequate increase in discrimination. For these cases, it could be that the best pair (or perhaps triple) might provide a substantial increase over that of the best single feature.

The optimum value for k is problem dependent. Experience has indicated that the performance of the decision rule is not particularly sensitive to its value over reasonable ranges. Because of the small computational requirements of this procedure in both training and classification, it is feasible to use the "leave-m-out" technique to estimate the best value of k directly from the training sample. A small number $m < N$ of vectors are deleted from the training set and the remainder are used to design the decision tree. The left out vectors are then classified by the resulting decision rule and the number of errors are recorded. This procedure is repeated $\lceil N/m \rceil$ times, each with a different set of deleted vectors. The error rate averaged over all of these trials is an estimate of the error rate for the decision rule. The value of k can be adjusted to minimize this estimated error rate.

Simulation Experiments

Applications of this decision rule are illustrated on two simulated problems. Simulated data are used so that the performance can be judged in light of the known separability of the underlying class probability densities and the complexity of the decision boundaries. The first example is a two-class problem that is constructed so that all linear classifiers   have no discriminating ability. The second example is a seven-class problem. In both experiments, the decision rule was implemented with no transgenerated features and only one adaptive feature, namely, the Fisher linear discriminant direction (Eqn 7). No look ahead was employed (L=0) and the minimum cell sample size was arbitrarily chosen to be ten (k=10) on the basis of no optimization. The results reported for each simulated experiment were obtained by generating twenty independent sets of training samples and a corresponding twenty sets of 1000 test vectors. For each of these twenty trials, the training sample was used to classify the 1000 test vectors. The statistics listed in Tables 1 - 2 were obtained by averaging the results over these twenty trials. The statistical uncertainties were obtained by dividing the standard deviation about the mean by the square root of twenty.

A. Two-Class Spherical Discrimination

In this problem, the probability density function of one class population completely surrounds that of the other. The first four features of the first population are distributed uniformly within a four-dimensional spherical slab centered at the origin with inner radius 3.5 and outer radius 4.0. The last six features are distributed as a spherical normal distribution located at the origin with unit covariance matrix. All ten features of the second population are normally distributed with unit covariance matrix and zero location. Thus, the two distributions differ only in the first four features and the last six contain no discriminating information. In order to make the problem more realistic, the spherical symmetry was removed by scaling each feature by its feature number. That is, the first coordinate is scaled by unity, the second by two, and so on, the last being scaled by ten. The asymptotic Bayes error rate for this experiment is 0.64%. A training sample size of 500 was used for each class.

Table 1 shows the results of applying the recursive partitioning decision rule to this problem and compares it to nearest neighbor discrimination in terms of average error rate, decision time and memory requirement.(3)

B. Multiclass Simplex Problem

This example consists of seven populations, each normally distributed in six dimensions with unit covariance matrix. Each distribution is located at a different vertex of a six-dimensional regular simplex and separated by a distance of four. A training sample size of 500 was used for each class. The asymptotic Bayes error rate for this example is 9.6%. The results are shown in Table 2.

Although these examples were constructed to be difficult, the recursive partitioning decision rule is seen to have comparable error rate to nearest neighbor discrimination, while requiring substantially less computational resources.

ACKNOWLEDGMENT

Many of the ideas presented here were inspired in discussions with Richard A. Olshen and Charles T. Zahn, Jr.

TABLE 1

Two-Class Spherical Discrimination Problem
Asymptotic Bayes Error Rate = 0.64%
500 Training Vectors/Class

	Recursive Partitioning	Nearest Neighbor
Error Rate (%)	16.9±0.5	35.2±0.3
Average Decision Time (ms)	0.099	12.5
Memory (bytes)	1061	40000

TABLE 2

Seven-Class Simplex Discrimination Problem
Asymptotic Bayes Error Rate = 9.6%
500 Training Vectors/Class

	Recursive Partitioning	Nearest Neighbor
Error Rate (%)	13.3±0.3	15.9±0.3
Average Decision Time (ms)	0.59	7.66
Memory (bytes)	9016	28000

FOOTNOTES

(1) If, during the descent, a substantial number of nodes are encountered that cut on adaptive features, the computation is increased by the time required to compute the features at each of these nodes.

(2) As a point of reference, the computation required to perform the recursive partitioning for the example in Table 1 (p=10, N=1000, k=10, and L=0) was 3.2 CPU seconds. See Footnote 3 below for computational details.

(3) All simulation experiments were performed on an IBM 370/168 computer with programs coded in FORTRAN IV and compiled with the IBM FORTRAN H (extended) compiler at optimization level two. Computational performance is stated in terms of average CPU milliseconds required to classify unknown test vectors. Memory requirements are reported in bytes (8-bits) under the assumption that integer pointers can be stored as halfword quantities (2 bytes), while real variables are stored in full words (4 bytes). The fast near neighbor algorithm of [4] was employed for all nearest neighbor calculations.

REFERENCES

[1] E. Fix and J.L. Hodges, Jr., "Discriminatory analysis, nonparametric classifications," USAF Sch. Aviat. Med., Rep. 4, Feb. 1951.

[2] _____, "Discriminatory analysis, small sample performance," USAF Sch. Aviat. Med., Rep. 11, Aug. 1952.

[3] T.M. Cover and P.E. Hart, "Nearest neighbor pattern classification," IEEE Trans. Inform. Theory, Vol. IT-13, pp 21-27, Jan. 1967.

[4] J.H. Friedman, J.L. Bentley, and R.A. Finkel, "An algorithm for finding best matches in logarithmic time," Stanford Linear Accelerator Center Rep. SLAC-PUB-1549, Feb. 1975. (To be published in ACM Trans. Math. Software).

[5] P.E. Hart, "The condensed nearest neighbor rule," IEEE Trans. Inform. Theory (Corresp.), pp 515-516, May 1968.

[6] C.L. Chang, "Finding Prototypes for Nearest Neighbor Classifiers," IEEE Trans. Comput., Vol. C-23, pp 1179-1184, Nov. 1974.

[7] T.W. Anderson, "Some nonparametric multivariate procedures based on statistically equivalent blocks," in Multivariate Analysis, P.R. Krishnaiah, Ed., Academic Press, New York, pp 5-27, 1966.

[8] E.G. Henrichon, Jr. and K.S. Fu, "A nonparametric partitioning procedure for pattern classification," IEEE Trans. Comput., Vol. C-18, pp 614-624, July 1969.

[9] W.S. Meisel and D.A. Michalopoulos, "A partitioning algorithm with application in pattern classification and the optimization of decision trees," IEEE Trans. Comput., Vol. C-22, pp 93-103, Jan. 1973.

[10] L. Gordon and R.A. Olshen, "Asymptotically Efficient, Computationally Feasible Solutions to the Classification Problem," University of Calif., San Diego preprint, (Submitted to the Annals of Statistics) Nov. 1970.

[11] D.C Stoller, "Univariate Two-population Distribution-free Discrimination," J. Amer. Statist. Assoc. 49, pp 770-775, 1954.

[12] J.A. Songuist, E.L. Baker, and J.N. Morgan, "Searching for Structure," Survey Research Center, University of Michigan, Ann Arbor, Michigan, 1973.