Pattern Classification Based on Local Learning

Jing Peng and Bir Bhanu

College of Engineering, University of California, Riverside, CA 92521, USA

Abstract. Local learning methods approximate a target function (*a posteriori* probability) by partitioning the input space into a set of local regions, and modeling a simple input-output relationship in each one. In order for local learning to be effective for pattern classification in high dimensional settings, regions must be chosen judiciously to minimize bias. This paper presents a novel region partitioning criterion that attempts to minimize bias by capturing differential relevance in input variables in an efficient way. The efficacy of the method is validated using a variety of real and simulated data.

1 Introduction

In pattern classification, a feature vector $\mathbf{x} \in \Re^p$ is assumed to be one of J classes $\{C_i\}_1^J$, and the objective is to assign $\mathbf{x}$ to the correct class. For a given cost matrix L_{ik}, where L_{ik} denotes the cost associated with assigning $\mathbf{x}$ to C_i when $\mathbf{x} \in C_k$, the Bayes decision rule assigns $\mathbf{x}$ to C_i such that the expected loss $L_i = \sum_{k=1}^J L_{ik} \Pr(k|\mathbf{x})$ is minimized. If all misclassifications induce an equal cost, then the Bayes rule reduces to: $i^* = \arg\max_{1 \le i \le J} \Pr(i|\mathbf{x})$. To apply the Bayes rule, therefore, $\{\Pr(i|\mathbf{x})\}_1^J$ must be estimated.

One broad class of methods for estimating $\{\Pr(i|\mathbf{x})\}_1^J$, based on supervised learning, is to treat the class label C at query $\mathbf{x}$ as a random variable from the distribution $\{\Pr(i|\mathbf{x})\}_1^J$. Following Friedman's notation [5], C at $\mathbf{x}$ can be characterized by $y_i|_\mathbf{x} = 1$ if $C|_\mathbf{x} = i$ or 0 if $C|_\mathbf{x} \neq i$. This gives rise to

$$f_i(\mathbf{x}) \doteq \Pr(i|\mathbf{x}) = \Pr(y_i = 1|\mathbf{x}) = E(y_i|\mathbf{x}), \tag{1}$$

with $0 \le f_i(\mathbf{x}) \le 1$, and $\sum_{i=1}^J f_i(\mathbf{x}) = 1$. The learning methods can then be applied to estimate each $f_i(\mathbf{x})$ from the corresponding training data $\{x_k, y_i\}_{k=1}^N$, $i = 1, \cdots, J$. This learning paradigm has been a basis for many techniques developed in neural networks, machine learning, and memory-based regression for pattern classification. In what follows, we drop subscript i from $f_i(\mathbf{x})$ in (1) for clarity.

2 Local Learning

Local learning methods [2, 3, 4, 9, 10, 12] attempt to learn the target function $f(\mathbf{x})$ (1) by first partitioning the space $\mathbf{x} \in \Re^p$ of input variables into a set of local regions $\{R_m\}_1^M$, where $R_m \subset \Re^p$ and $\cup_1^M R_m = \Re^p$. These methods then learn a separate approximator $\hat{f}_m(\mathbf{x})$ (usually low order polynomials) individually in each local region. The accuracy of learning can be measured by the error function

$$L(f(\mathbf{x}), \hat{f}(\mathbf{x})) = |f(\mathbf{x}) - \hat{f}(\mathbf{x})|^2. \tag{2}$$

The "local" notion can be characterized by the size of the regions, $size(R_m) = ave_{\mathbf{x},\mathbf{x}' \in R_m} \parallel \mathbf{x} - \mathbf{x}' \parallel$, where ave is an averaging function. For $p = 1$, i.e. $\mathbf{x} \in \Re$, the regions can be described by their locations and sizes. For $p \geq 2$ regions have "shape" as well as location and size. One can define the shape of a region by its size along all directions in the input space, that is, by the function [6]

$$s_m(\alpha) = ave_{\mathbf{x} \in R_m} |\alpha^t(\mathbf{x} - \mathbf{u}_m)| \tag{3}$$

where α is a unit vector in $\Re^p$ and $\mathbf{u}_m$ is the center of R_m.

Local learning methods have shown promise in low dimensional settings [5, 9]. However, these methods tend to be less effective as dimensionality p increases. This is a result of the "curse-of-dimensionality" [1]. Given the fact that the expectation of (2) can be decomposed into

$$E[f(\mathbf{x}) - \hat{f}(\mathbf{x})]^2 = [f(\mathbf{x}) - E\hat{f}(\mathbf{x})]^2 + E[\hat{f}(\mathbf{x}) - E\hat{f}(\mathbf{x})]^2 \tag{4}$$

where the first term is the squared bias and the second term is the variance, it can be seen that in ordor to minimize the error function (2) in expectation one must minimize both squared bias and variance. However, these are competing goals. While minimizing squared bias demands small region size, small regions create large variance. Techniques, such as recursive covering [6] and "soft" partitioning [7], exist that accommodate, to the extent possible, these competing demands. The goal of this paper is to introduce a novel technique that minimizes squared bias by choosing region shape (3) more judiciously. It represents an attempt to mitigate the curse-of-dimensionality.

3 Splitting Criteria

In order to produce a good solution to the classification problem a splitting criterion is required that, when combined with recursive partitioning, selects region shape (3) such that $\hat{f}_m(\mathbf{x})$ approximates the target $f(\mathbf{x})$ (1) well within each region, thereby minimizing the overall error function (2). It is a step where care must be taken for achieving optimal pattern classification.

3.1 Residual Splitting

In [6], it is argued that the optimal shape for a region R_m should be governed by the properties of the target function $f(\mathbf{x})$ within it. Specifically, its size in any direction α (3) should be inversely proportional to the rate of change of the absolute bias in that direction

$$s_m(\alpha) \sim 1/ave_{\mathbf{x} \in R_m}(\alpha^t \frac{\partial}{\partial \mathbf{x}} |f(\mathbf{x}) - \hat{f}_m(\mathbf{x})|). \tag{5}$$

It is clear that the size depends on the target function $f(\mathbf{x})$, the local approximator $\hat{f}_m(\mathbf{x})$ used, and the location $\mathbf{x}$ of the region in the input space. In words, equation (5) states that the input space should be finely partitioned in areas where $f(\mathbf{x})$ is not smooth.

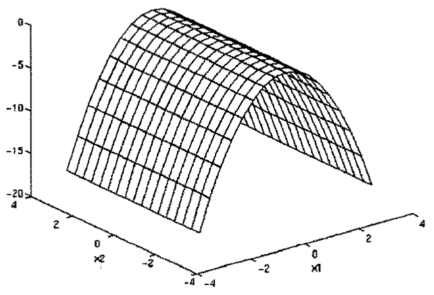

Fig. 1. A quadratic function.

Based on (5) the splitting criterion for recursive partitioning proposed in [6] is the following. Let $\{r_i = y_i - \hat{f}_R(\mathbf{x}_i)|\mathbf{x}_i \in R\}$ be the residuals resulting from the approximator $\hat{f}_R(\mathbf{x})$ fit to the data in R. Then the criterion (called *residual criterion* here) to be maximized for choosing the split direction α is

$$ResidualCri(\alpha) = |ave_{\mathbf{x}_i \in R}\{r_i|\mathbf{x}_i \leq s(\alpha)| + |ave_{\mathbf{x}_i \in R}\{r_i|\mathbf{x}_i > s(\alpha)| \quad (6)$$

where $s(\alpha)$ is the split point (the median of $\{\alpha^t\mathbf{x}_i|\mathbf{x}_i \in R\}$). It is a computationally feasible approximation to (5). Maximizing (6) yields the direction in which the points to be removed deviate most from the local approximation in the region being split. It has been demonstrated that the splitting criterion (6) gives good performance in a variety of target functions.

3.2 Limitation of Residual Splitting

Let us consider the following quadratic target function defined over the two dimensional input space $\mathbf{x} \in [-4, 4]^2$ $f(\mathbf{x}) = -x_1^2$. The target $f(\mathbf{x})$ is clearly asymmetric in the input variables as shown in Fig. 1. That is, for an arbitrary rotation matrix R, $f(\mathbf{x}) \neq f(R\mathbf{x})$. The target $f(\mathbf{x})$ changes rapidly along the x_1 axis while remaining constant along x_2. Suppose one uses a local constant approximator (zero-degree polynomial or mean value) in each region. Then it can be shown that the optimal recursive split should be carried out along x_1 only. Assume we are given the following training data: $\{(-3,-2),-9;(-1,-2),-1;(1,-2),-1;(3,-2),-9;$ $(-3,2),-9;(-1,2),-1;(1,2),-1;(3,2),-9\}$.

Assume further without loss of generality that the directions to be optimized are taken to be the original coordinate axes and that the split points are taken to be mean values along each axis. When (6) is applied to the above data, it fails to yield the optimal direction (x_1 in this case) along which the input space should be split. In fact, $ResidualCri(\mathbf{e}_1) = ResidualCri(\mathbf{e}_2) = 0$, where $\mathbf{e}_i$ is a unit vector along the ith input coordinate. That is, (6) cannot distinguish high differential relevance between the two input variables.

While this target function is an especially simple one, the local constant approximator is quite general. It is meant to illustrate the problems involved; it shows that (6) is quite limited in producing the desired goal in a simple setting and that (6), while computationally simple, is far from approximating (5). Better criteria that are sensitive to input differential relevance must be sought.

3.3 Differential Splitting

The optimal partition for any region should be driven by the properties of the target function $f(\mathbf{x})$ within it, particularly the derivatives (first order), as indicated by (5). In this sense, the splitting criterion (6) represents only a zero-order approximation to (5) in that it finds a direction in which the points to be removed (to create a new partition) minimize the residuals from the local fit to the data within the region being split. However, it is possible that even though the input variables have high differential relevance, the average residual remains the same along the directions under consideration, as evidenced by the above quadratic function example (Section 3.2).

The splitting criterion we propose here represents a computationally feasible approximation to (5) and is a generalization of (6). It attempts to capture input differential relevance by estimating the (first order) derivatives of target functions, at least in some situations.

From the Taylor series expansion, for a sufficiently small δ

$$f(\mathbf{x}+\delta\mathbf{e}_i) \simeq f(\mathbf{x})+\delta\mathbf{e}_i^t g(\mathbf{x})+\frac{1}{2}\delta^2\mathbf{e}_i^t G(\mathbf{x})\mathbf{e}_i = f(\mathbf{x})+\delta[g]_i(\mathbf{x})+\frac{1}{2}\delta^2[G]_{ii}(\mathbf{x}) \quad (7)$$

where $\mathbf{e}_i$ is the unit vector along the ith coordinate direction, $[g]_i$ is the ith element of g and $[G]_{ii}$ is the iith element of G. Thus, $[g]_i$ can be approximated by $[\hat{g}]_i$

$$[\hat{g}]_i(\mathbf{x}) = \frac{f(\mathbf{x}+\delta\mathbf{e}_i)-f(\mathbf{x})}{\delta} - \frac{1}{2}\delta[G]_{ii}(\mathbf{x}) \quad (8)$$

When the second term involving $[G]_{ii}$ is ignored, which is the second-order term in δ in the Taylor series, we have the *forward difference approximation* [11] that is exact for linear functions. The alternative, *central difference approximation* [11], neglects only third-order terms in δ and is, therefore, exact for quadratic functions. However, the increase in accuracy is at the expense of increased computation.

Let

$$\{[\hat{g}]_k = \frac{y_i - y_j}{[\mathbf{x}_i]_k - [\mathbf{x}_j]_k} | \mathbf{x}_i, \mathbf{x}_j \in R, 0 < |[\mathbf{x}_i]_k - [\mathbf{x}_j]_k| \leq \delta, \| \mathbf{x}_i - \mathbf{x}_j \|_\infty \leq \theta\}_1^n \quad (9)$$

be the finite difference approximation to the first derivative of the target function $f(\mathbf{x})$ in the region R being split, where δ and θ are the procedural ("meta") parameters. The conditions, $0 < |[\mathbf{x}_i]_k - [\mathbf{x}_j]_k| \leq \delta$ and $\| \mathbf{x}_i - \mathbf{x}_j \|_\infty \leq \theta$, state that two points $\mathbf{x}_i$ and $\mathbf{x}_j$ must be close for the approximation to be valid. Note that it is possible that such an approximation may not exist. Let

$$\{[dr]_i(\mathbf{x}) = [\hat{g}]_i(\mathbf{x}) - \partial \hat{f}_R/\partial[\mathbf{x}]_i | \mathbf{x} \in R, i = 1, \cdots, n\} \quad (10)$$

where $\hat{f}_R$ is the local approximator in R. If $[\hat{g}]_i(\mathbf{x})$ does not exist, $[dr]_i(\mathbf{x}) = 0$. Also let the derivative criterion

$$DeriCri(\alpha) = \max_{i,\mathbf{x}\in R}\{|[dr]_i(\mathbf{x})||\mathbf{x} \leq s(\alpha)\} + \max_{i,\mathbf{x}\in R}\{|[dr]_i(\mathbf{x})||\mathbf{x} > s(\alpha)\} \quad (11)$$

where $s(\alpha)$ is the split point (the median of $\{\alpha^t\mathbf{x}|\mathbf{x} \in R\}$). Maximizing (11) produces the direction along which the first order derivative of the local approximation deviates most from that of the target function in the region being split. The differential splitting criterion (to be maximized) for selecting the split direction α can then be described as

$$SplitCri(\alpha) = \lambda ResidualCri(\alpha) + (1-\lambda)DeriCri(\alpha) \quad (12)$$

where $\lambda \in [0,1]$. If the input variables are equally relevant, then how well the local approximator fits to the data should determine the split direction. In this case, one prefers $\lambda = 1$. On the other hand, if the input variables are unequal in their differential relevance, then the direction selected should inversely reflect the extent to which such relevance can be captured by the local approximation. This corresponds to $0 \leq \lambda < 1$, and is related to the notion of differential scaling of input variables. At the extreme where $\lambda = 0$, partition is determined solely by (11). In theory, (11) by itself can produce the desired goal. In practice, however, insufficient data might impede the direct application of (11). Our experiments show that (12) with $\lambda \in (0,1)$ achieves the best of both (6) and (11) worlds.

In order to gain an intuitive perspective on (12), we apply it to the example described in Section 3.2 with the following parameter settings: $\lambda = 0.9$, $\delta = 2$, $\theta = 0.1$, and again the split points are taken to be the mean. We then have $SplitCri(\mathbf{e}_1) = 3.6$ and $SplitCri(\mathbf{e}_2) = 0$. Thus x_1 is the (optimal) direction selected for splitting.

4 Empirical Results on Simulated Data

This section presents results of applying the two splitting criteria (6) and (12) to artificial data that simulate a variety of target functions (1). For each target function $f(\mathbf{x})$ and each method k, the mean absolute target error, $e_k = mean|f(\mathbf{x}) - \hat{f}_k(\mathbf{x})|$, was computed over 5000 independently generated test observations. The performance measure used for comparison is

$$r_k = e_k / \min_t e_t. \quad (13)$$

The distribution of r_k values provides relative performance for each method. Thus, this distribution for a method that is best for all target functions should be a point mass at the value 1. In all the experiments reported below, λ (12) was set to 0.9. In addition, the values of δ and θ (9) used to produce the results for each target were selected as the best from among several runs.

4.1 Quadratic Function

The target function to be studied in this section is taken to be the same quadratic function as in Section 3.2. The two input variables are randomly generated from a

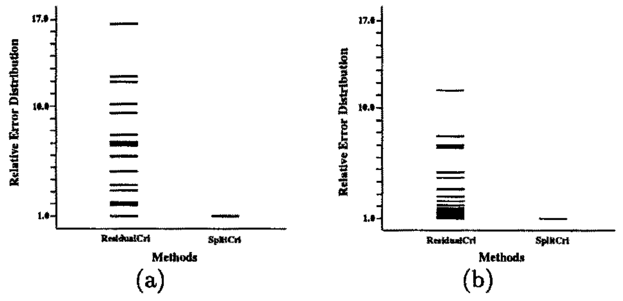

Fig. 2. Distribution of relative errors for the quadratic target function. (a) Local constant approximator. (b) Local linear approximator.

uniform distribution $\mathbf{x} \sim U^2[-4,4]$, and the corresponding output y is generated from $-x_1^2$. 50 sets of training data with $N = 500$ were independently generated. The training observations in each terminal region of the final partition were set to 5 (stopping criterion).

Fig. 2 shows the distributions of relative errors for the two methods on an independent test set of 5000 observations. Fig. 2(a) shows the results using a local constant approximator, while Fig. 2(b) the results using a local linear approximator. It can be seen that high differential relevance is exploited by the new splitting criterion (12).

4.2 Randomly Generated Functions

The quadratic target function was purposefully chosen to show the ability of the new splitting criterion to differentiate input relevance in some cases. To further illustrate the relative merits of the splitting criterion it is applied to a variety of randomly generated target functions [6]. Each one has the form: $f(\mathbf{x}) = \sum_{k=1}^{K} a_k h(\mathbf{x}, \mathbf{z}_k, \mathbf{V}_k)$, where $h(\mathbf{x}, \mathbf{z}, \mathbf{V}) = \frac{1}{|\mathbf{V}|}\exp[-\frac{1}{2}(\mathbf{x}-\mathbf{z})^t\mathbf{V}^{-1}(\mathbf{x}-\mathbf{z})]$, $\{a_k \sim U[-1,1]\}_1^K$, and $\{\mathbf{z}_k \sim U^n[0,1]\}_1^K$. In addition, the eigenvectors of each $\mathbf{V}_k$ are generated uniformly randomly on the unit n-sphere subject to the orthogonality constraint, and the eigenvalues are generated uniformly from $U[0.525\sqrt{n}, 0.025\sqrt{n}]$. This allowed us to generate a wide variety of target functions in terms of the geometric shapes of their contours.

These experiments consist of 100 such randomly generated functions with $p = 10$ input variables. The study of the 100 target functions is divided into two groups of 50 targets each, the first group with $K = 5$, and the second with $K = 10$. The input points for each target were generated randomly from a uniform distribution $\mathbf{x} \sim U^{10}[0,1]$. The training observations in each terminal region of the final partition were set to 5 for local constant fitting, while the training observations were set to 20 for local linear fitting. The quantity for comparison is provided by (13) over 5000 independently generated test observations.

Fig. 3 shows the distributions of relative performance of the two splitting

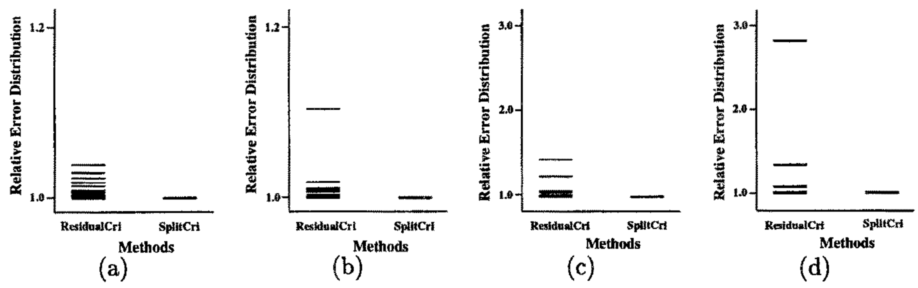

Fig. 3. Distributions of relative errors for the target functions. (a)(c) $K = 5$. (b)(d) $K = 10$. (a)(b) Local constant approximator. (c)(d) Local linear approximator.

criteria for the target functions. In both cases, each target was learned using a training set of $N = 1500$ observations. The results show convincingly that the new criterion (12) out performed (6) on all the target functions. It can also be seen that local constant fitting seems to give better performance than local linear fitting, which is consistent with the results reported in [6]. Finally, an additional experiment, whose details we omit here, was carried out to examine the effect of λ on the performance of (12). The results show that the overall performance of (12) as a function of λ is approximately bell-shaped, indicating neither (6) nor (11) alone achieved good performance over target functions under study, and that good performance can be obtained over a wide range of λ values.

5 Empirical Results on Real Data

In order to further evaluate our local learning method for pattern classification, it is applied to the Letter Image Recognition Data (LIRD) from the UCI repository [8]. LIRD consists of a large number of black-and-white rectangular pixel displays as one of the 26 upper-case letters in the English alphabet. The characters are based on 20 different fonts and each letter is randomly distorted to produce a file of 20,000 unique stimuli that are then converted into 16 primitive numerical features (statistical moments and edge counts). A subset of LIRD is used to produce the results reported here. It has two classes (letters C and G) with total 1509 instances (736 for C and 773 for G). The last 11 of the 16 features (first 5 features, such as box positions and the size of box, are not significant) are used here and scaled to fit into a range of real values from 0 to 1. A local constant approximator is used, and the training observations in each terminal region of the final partition are set to 3.

In the *first* experiment, 1000 instances are randomly selected as training data, and the rest as test data upon which the resulting classification model is evaluated. This is repeated for 7 times, and the median classification accuracy is obtained. This process is done 10 times and the mean value is reported here. The method (6) achieves an average classification accuracy of 93.4%, whereas the new method (12) (with $\lambda = 0.01$, $\delta = 0.12$ and $\theta = 0.62$) has an average accuracy of 94.4%, with standard deviations 0.3 and 0.2, respectively. In the *second* experiment, only 500 instances are used as training data, and the results are 91.6% and 92.4% (with $\lambda = 0.01$, $\delta = 0.1$ and $\theta = 0.6$) with standard deviations 0.1 and 0.2, respectively. One thing to notice from the experiments is that, while significant difference exists in performance between the two methods, the results are close. This might be attributed to the fact that differential relevance along directions under consideration is similar.

6 Conclusions

The local learning method developed in this paper for pattern classification achieves superior performance in a wide variety of target functions and.classification tasks by capturing differential relevance in input variables. Although the local learning method has been developed in the context of recursive partitioning, it can be easily extended to recursive covering [6] and "soft" partitioning [7] as well. New criteria, both definitional and procedural, will be developed in the future that are more efficient computationally and lead to even better performance for pattern classification.

Acknowledgements
This work was supported in part by ONR grant N00014-96-C-0321 and by DARPA/AFOSR grant F49620-95-1-0424 and F49620-97-1-0184.

References

1. R.E. Bellman, *Adaptive Control Processes.* Princeton Univ. Press, 1961.
2. L. Bottou and V. Vapnik, Local learning algorithms. *Neural Computation*, 4(6), 888-900, 1992.
3. L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone, *Classification and Regression Trees.* Wadsworth, 1984.
4. W.S. Cleveland and S.J. Devlin, Locally weighted regression: An approach to regression analysis by local fitting. *J. Amer. Statist. Assoc.* **83**, 596-610, 1988.
5. J.H. Friedman, *Flexible metric nearest neighbor classification.* Tech. Report, Dept. of Statistics, Stanford University, Stanford, CA 94305, 1994.
6. J.H. Friedman, Local Learning Based on Recursive Covering. Tech. Report, Dept. of Statistics, Stanford University, Stanford, CA 94305, 1996.
7. M.I. Jordan and R.A. Jacobs, Hierarchical mixtures of experts and the EM algorithm. *Neural Computation* **6**, 181, 1994.
8. P.M. Murphy and D.W. Aha, UCI repository of machine learning databases. http://www.cs.uci.edu/~mlearn/MLRepository.html, 1995.
9. J. Peng, Efficient Memory-Based Dynamic Programming. *Proceedings of the 12th International Conference on Machine Learning*, 438-446, 1995.
10. J.R. Quinlin, *C4.5: Programs for Machine Learning.* Morgan-Kaufmann Publishers, Inc., 1993.
11. L.E. Scales, *Introduction to Non-Linear Optimization.* New York: Springer-Verlag, 1985.
12. C.J. Stone, Nonparametric regression and its applications (with discussion). *Ann. Statist.* **5**, 595, 1977.