ADAPTIVE NEAREST NEIGHBOR CLASSIFIER

ANIL K. GHOSH

Department of Mathematics and Statistics, Indian Institute of Technology Kanpur, Uttar Pradesh 208016, India.
E-mail: anilkghosh@rediffmail.com

In nearest neighbor classification, one normally uses cross-validation type methods to estimate the optimum value of k, and that estimated value is used for classifying all observations. However, in classification problems, in addition to depending on the training sample, a good choice of k depends on the specific observation to be classified. In this article we propose an adaptive nearest neighbor classification technique, where the value of k is selected depending on the distribution of competing classes in the vicinity of the observation to be classified. Results on some simulated and benchmark examples are presented to show the utility of the proposed method.

Keywords: Bayesian strength function; cross-validation; posterior probability; prior distribution; sequential technique

1. Introduction

In nearest neighbor classification,[1,2] one assumes posterior probabilities of different classes to be constant over a small neighborhood around a query point $\mathbf{x}$ and estimates those probabilities using the proportions of different classes in the training sample lying in that neighborhood. As a result, the most frequent class in that neighborhood has the largest estimated posterior $\hat{p}(\cdot \mid \mathbf{x})$, and it turns out to be the winner. Usually, a closed ball of radius $r_k(\mathbf{x})$ is taken as this neighborhood, where $r_k(\mathbf{x})$ is the distance between $\mathbf{x}$ and its k-th nearest data point in the training sample. Clearly, the size of this neighborhood and hence the classification result depend on the parameter k. Existing asymptotic results[2,3] suggest that k should vary with the training sample size n in such a way that $k \to \infty$ and $k/n \to 0$ as $n \to \infty$. However, when the sample size is small or moderately large, there is no theoretical guideline for choosing the value of k. For a given data set, the optimum value of k is usually estimated by minimizing the cross-validation estimate[4] of misclassification probability. Of course one can also follow the idea of likelihood cross-validation technique (LCV),[5,6] which estimates the optimum value of k by maximizing the loglikelihood score $\sum_{t=1}^{n} log\{p_{-t}(c_t \mid \mathbf{x}_t)\}$, where c_t is the class label of the observation $\mathbf{x}_t$, and $p_{-t}(j \mid \mathbf{x}_t)$ is the posterior probability estimate for the j-th class at $\mathbf{x}_t$, when $\mathbf{x}_t$ is not used as a training data point. These cross-validation and LCV methods use the training data to select a single value of k, and then that selected value is used for classifying all observations. However, one should note that in classification problems, in addition to depending on the training sample, a good choice of k depends on the specific observation to be classified. Therefore, instead of using a fixed value of k over the entire measurement space, spatially adaptive choice of k may be more useful for classification. In this article, we propose one such adaptive nearest neighbor classifier, which adopts a sequential approach (to be discussed in Section 3) to select the optimum value of k for classifying an observation.

Throughout this article, we will assume continuity of all population density functions and use the Euclidean distance for classification. However, the description of the adaptive nearest neighbor classifier given in Section 2 will make it quite clear that the use of the proposed classifier is not restricted to any special type of distance function, and one may use Mahalanobis distance[7] or any other flexible and adaptive metric[8,9] as well.

2. Methodology

As we have mentioned before, in usual nearest neighbor classification, posterior probabilities of competing classes are assumed to be constant over a neighborhood around $\mathbf{x}$. Here, we deviate from this usual notion, and instead of considering $p(j \mid \mathbf{x})$'s as fixed and non-random, we assume a prior distribution of $p(1 \mid \mathbf{x}), p(2 \mid \mathbf{x}), \ldots, p(J \mid \mathbf{x})$. Since it is quite evident that the calculations have to be done at each $\mathbf{x}$ separately, for convenience of our notation, we will drop the term $\mathbf{x}$, and denote $p(j \mid \mathbf{x})$ by p_j. Similarly, the vector of conditional probabilities $(p_1, p_2, \ldots, p_J)$ $[\sum_{j=1}^{J} p_j = 1]$ will be denoted by $\mathbf{p}$.

Suppose that for some given k, $\xi_k(\mathbf{p})$ is the prior distribution of $\mathbf{p}$ in the neighborhood (ball of radius $r_k(\mathbf{x})$) around $\mathbf{x}$. If t_{j_k} of these k neighbors come from the j-th class, the conditional distribution of $\mathbf{t}_k = (t_{1_k}, t_{2_k}, \ldots, t_{J_k})$ $[\sum_{j=1}^{J} t_{j_k} = k]$ for given $\mathbf{p}$ is multinomial and its probability mass function can be expressed as

$$\psi_k(\mathbf{t}_k \mid \mathbf{p}) = \frac{k!}{t_{1_k}! t_{2_k}! \ldots t_{J_k}!} \prod_{j=1}^{J} p_j^{t_{j_k}}.$$

Therefore, for some fixed k and $\mathbf{t}_k$, the conditional distribution of $\mathbf{p}$ is given by

$$\zeta_k(\mathbf{p} \mid \mathbf{t}_k) = \xi_k(\mathbf{p})\, \psi_k(\mathbf{t}_k \mid \mathbf{p}) \Big/ \int \xi_k(\mathbf{p})\, \psi_k(\mathbf{t}_k \mid \mathbf{p})\, d\mathbf{p}.$$

In a two-class problem, one will naturally prefer the first class as compared to the second one if $P(p_1 > p_2 \mid \mathbf{t}_k) > P(p_2 > p_1 \mid \mathbf{t}_k)$, and the evidence in favor of the first class will increase with the probability function $P(p_1 > p_2 \mid \mathbf{t}_k)$. Following similar idea, one can use $\zeta_k(\mathbf{p} \mid \mathbf{t}_k)$ to define the Bayesian measure of strength for different populations, where the strength function for the j-th $(j = 1, 2, \ldots, J)$ population is given by

$$S_k(j) \;=\; \int\limits_{p_j = max\ \{p_1, p_2, \ldots, p_J\}} \zeta_k(\mathbf{p} \mid \mathbf{t}_k)\; d\mathbf{p}.$$

It is quite transparent from the definition that $0 \leq S_k(j) \leq 1$ and $\sum_{j=1}^{J} S_k(j) = 1$ if $\xi_k(\mathbf{p})$ is the probability density function of a continuous distribution. Also, these Bayesian strength functions (BSF) preserve the ordering of usual posterior probability estimates (i.e., proportions of different classes in the neighborhood).[5] Therefore, for any fixed value of k, if one goes for classification based on the values of $S_k(j)$, it will lead to the same result that of usual k-nearest neighbor classification. Note that the value of $S_k(j)$ depends on the prior distribution $\xi_k(\mathbf{p})$ as well, and it has to be chosen appropriately. Uniform prior distribution is the most convenient one to handle with. Not only it makes the computation of BSF simpler (we will discuss it in Section 3), it is non-informative and gives no preference to any of the classes. Throughout this article, for all data analytic purpose, we will assume that $\xi_k(\mathbf{p})$ is uniform for all values of k.

In adaptive nearest neighbor classification, instead of using a fixed value of k, we study the results for a sequence of values of k and choose that one which leads to the maximum BSF in favor of the winning population. However, it is not useful to consider all possible values of k since the use of very large values not only increases the computational cost but also fails to represent the local pattern of the measurement space. So, if the training samples from competing classes are not of comparable size, use of large k leads to a decision in favor of one of the larger classes. Therefore, one should set some reasonable upper bound for k. This upper bound is expected to increase with the sample size n. In nearest neighbor classification, for the consistency of posterior probability estimates, one needs to shrink the neighborhood to zero as the sample size increases. So, it is somewhat reasonable to use a function β_n of n as the upper bound, which increases with n but $\beta_n/n \to 0$ as $n \to \infty$. We look at the results for all $k \leq \beta_n$, and finally use that one which leads to the highest strength in favor of the winning population. Unlike usual nearest neighbor classification, here we do not need to go for cross-validation type algorithm to predefine the value of k, and that leads to substantial saving in computational cost, especially when the training sample is large.

3. Data analytic implementation

We adopt a sequential type approach to find adaptive values of k. For classifying an observation $\mathbf{x}$, we start with the data point nearest to $\mathbf{x}$ and then gradually increase the value of k by considering other data points one by one according to their distances from $\mathbf{x}$. At each stage, we compute BSFs for different populations and keep track of the population, which attained the maximum BSF up to that stage. Given the value of β_n and the first k neighbors, one can easily compute the largest possible strength in future in favor of other populations. Clearly, the largest value of this future strength will be attained if the rest of the observations come from the second best class (i.e. the class with the maximum number of representatives among the challengers at that stage). If this largest possible value is smaller than the observed maximum BSF up to that stage, one can stop there and classify the observation to the class that achieved the maximum BSF. For instance, in a two-class problem, if the upper bound β_n is set at 5, and the first two neighbors of $\mathbf{x}$ come from the first population, one can stop there and take the decision in favor of the first population. Note that given the first two

neighbors of $\mathbf{x}$, the second population can at most enjoy the 3-2 majority, but the BSF in favor of the winning class in that case is smaller than that in the 2-0 case. Therefore, in this case, there is no need to go for further calculations.

From our discussion in the previous section, it seems that $\beta_n = C\sqrt{n}$ could be a good choice for the upper bound of k, where C is an appropriate constant. For this choice of β_n, the adaptive nearest neighbor classifier requires only $O(n)$ calculations to classify an observation, whereas it requires $O(n\ log(n))$ computations if we use all possible values of k. Note that this order of computation $O(n)$ is the same as that required by the nearest neighbor classifier with a fixed value of k (follows from computational complexity of order statistics[10]). Our empirical experience suggests that the final result is reasonable and not very sensitive to the choice of C if it lies between 2 and 3. Throughout this article for all data analytic purpose, we use $\beta_n = 2\sqrt{n}$. However, if this value of β_n exceeds $n_0 = min\{n_1, n_2 \ldots, n_J\}$, we take $\beta_n = n_0$. Though this choice of upper bound is somewhat subjective, it led to fairly good results in our experiments, which we will see in Section 4.

To compute BSFs for different populations, one can use the numerical integration method. Given the value of k and the vector $\mathbf{t}_k$, the required number of computations for this approximation is proportional to γ^{J-1}, where γ is the number of grid points chosen on each axis. Clearly, this computational cost grows up rapidly with the number of classes J. Therefore, in the presence of several competing populations, one has to look for an alternative approximation procedure. If $\xi_k(\mathbf{p})$ is uniform, it is easy to see that given the value of k and $\mathbf{t}_k$, $\mathbf{p}$ follows a Dirichlet distribution with parameters $k+1;\ t_{1k}+1, t_{2k}+1, \ldots, t_{Jk}+1\ \ (\sum_{j=1}^{J} t_{jk} = k)$. Therefore, one can easily generate observations from the appropriate Dirichlet distribution to approximate BSFs for different populations. We know that if $X_1, X_2, \ldots, X_J$ are independent and $X_j \sim Gamma(t_{jk}+1)$ for $j = 1, 2, \ldots, J$, $(X_1/S, X_2/S, \ldots, X_J/S)$ jointly follows a Dirichlet distribution with parameters $k+1;\ t_{1k}+1, t_{2k}+1, \ldots, t_{Jk}+1\ \ (\sum_{j=1}^{J} t_{jk} = k)$, where $S = \sum_{j=1}^{J} X_j$. Therefore, $P\{p_j > p_i\ \forall i \neq j \mid \mathbf{t}_k\} = P\{X_j > X_i\ \forall i \neq j\}$, and hence BSF can be approximated by generating observations from gamma distributions. Now, noting the fact that $Gamma(t_{jk}+1)$ variate is a sum of $t_{jk}+1$ iid exponential variables with unit mean, one can keep on generating observations from exponential distributions to approximate BSF for different populations and for different values of k in a sequential way. For our data analytic purpose, we adopted the numerical integration method when $J \leq 3$. For $J \geq 4$, BSFs were approximated using 10,000 sets of observations generated from appropriate gamma distributions.

4. Results

We use some simulated and benchmark data sets to study the performance of the proposed classifier. For the simulated example (wavefrom data[11]), taking almost equal number of observations from competing classes, we generated 1000 different training and test sets of sizes 100 and 1000, respectively. In cases of salmon data[12] and Iris data,[13] we formed these training and test sets by randomly partitioning the data set 1000 times into two equal halves consisting of equal number of observations from the competing classes. For these data sets, average test set misclassification rates of the adaptive nearest neighbor classifier over these 1000 trials are reported in Table 1 along with their corresponding standard errors inside the braces. However, for the vowel recognition data,[14] there are specific training and test sets, and in this case we report the test set error rate only. Error rates are also reported for LCV and usual cross validation techniques. A brief description of the data sets that we use in this section is given below.

Waveform data:[11] This is a simulated data set. For $i = 0, 1, \ldots, 20$, define $h_1(i) = max\{6 - |i - 10|, 0\}, h_2(i) = h_1[(i-4)\ mod\ 21]$ and $h_3(i) = h_1[(i+4)\ mod\ 21]$. Suppose that $U \sim U(0,1)$ and ϵ_i's $(i = 0, 1, \ldots, 20)$ are *i.i.d.* $N(0,1)$. Now, for the three competing classes, measurement variables X_i $(i = 1, 2, \ldots, 21)$ are distributed as

Class-1: $X_i = Uh_1(i-1) + (1-U)h_2(i-1) + \epsilon_{i-1}$.
Class-2: $X_i = Uh_1(i-1) + (1-U)h_3(i-1) + \epsilon_{i-1}$.
Class-3: $X_i = Uh_2(i-1) + (1-U)h_3(i-1) + \epsilon_{i-1}$.

Salmon data:[12] It consists of 100 bivariate observations on growth ring diameter (freshwater and marine water) of salmon fish coming either from Alaskan or from Canadian water.

Iris data:[13] It contains measurements on four different features (sepal length, sepal width, petal length and petal width) on each of 150 observations

coming from three different types of Iris plants : 'Iris Setosa', 'Iris Virginica' and 'Iris Versicolor' (50 from each class).

Vowel recognition data:[14] It was created by a spectrographic analysis of vowels in words formed by an 'h' followed by a vowel and then by a 'd'. There were 67 persons who spoke different words, and two lowest resonant frequencies of a speaker's vocal track were noted for 10 different vowels. This data set was split into a training and a test set of sizes 338 and 333, respectively.

As it has been discussed before, in adaptive nearest neighbor classification, we generate an ensemble of classifiers and then choose that one which leads to the maximum BSF in favor of the winning class. Instead of taking only one value of k, one may also like to consider the results for all possible values of k ($1 \leq k \leq n-1$) and use voting or other aggregation methods to arrive at the final decision. Bagging[15] and boosting[16,17] are the two most popular aggregation methods available in the literature. They use bootstrap[18] or weighted bootstrap technique to generate different subsamples from the training data, and based on those different subsamples, different classifiers are developed. Results of these classification rules are aggregated using some weight function. Depending on the error rate Δ of a classifier, boosting assigns an weight $w = log\{(1-\Delta)/\Delta\}$ to it. Bagging on the other hand uses equal weights for all classifiers. In our method, we do not require any resampling technique for generating the classifiers. Use of different values of k leads to different classification rules, which can be aggregated using some weight function. Here, we adopt the weight functions used by bagging (i.e. uniform weight) and boosting (i.e. weight proportional to log of the odd ratio) for aggregation. This essentially leads to simple voting and weighted voting of all nearest neighbor classifiers. In Table 1, we report the error rates for these two voting methods as well.

Apart from the salmon data, in all other cases our proposed method led to the best error rate among the nearest neighbor classifiers considered here. In most of these cases, its performance was significantly better than its competitors. Weighted voting method had the best error rate on the salmon data, but even in that case the adaptive nearest neighbor classifier did a reasonably good job. In view of the above data analysis, adaptive nearest neighbor classification seems to be better than using a single value of k for classification of all observations.

Table 1. Misclassification rates of different classifiers

	Waveform	Salmon	Iris	Vowel
LCV	20.1 (.06)	9.2 (.10)	3.9 (.06)	25.5
Cross-valid.	20.1 (.07)	10.1 (.11)	4.2 (.06)	21.9
Voting	26.7 (.09)	9.1 (.11)	6.3 (.08)	67.2
Wt. voting	22.9 (.08)	8.9 (.11)	5.6 (.08)	22.8
Adaptive	18.9 (.06)	9.2 (.09)	3.8 (.06)	21.3

References

1. E. Fix and J. L. Hodges, Project 21-49-004, Rep. 4, pp. 261-279. US Air Force School of Aviation Medicine, Randolph Field (1951).
2. T. M. Cover and P. E. Hart, *IEEE Trans. Info. Theory*, **13** 21 (1967).
3. D. O. Loftsgaarden and C. P. Quesenberry, *Ann. Math. Statist.*, **36**, 1049 (1965).
4. P. A. Lachenbruch and M. R. Mickey, *Technometrics*, **10**, 1 (1968).
5. A. K. Ghosh, *Comput. Statist. Data Anal.*, **50**, 3113 (2006)
6. B. W. Silverman, *Density Estimation for Statistics and Data Analysis* (Chapman and Hall, London, 1986).
7. P. C. Mahalanobis, *Proc. Nat. Acad. Sci., India*, **12**, 49 (1936).
8. J. H. Friedman, Technical Report 113, Stanford University Statistics Department (1994).
9. T. Hastie and R. Tibshirani, *IEEE Trans. Pattern Anal. Machine Intell.*, **18**, 607 (1996).
10. A. V. Aho *et. al.*, *Design and Analysis of Computer Algorithms* (Addison-Wesley, London, 1974).
11. L. Breiman *et. al.*, *Classification and Regression Trees* (Chapman and Hall, London, 1984).
12. R. A. Johnson and D. W. Wichern, *Applied Multivariate Statistical Analysis* (Prentice Hall, New Jersey, 1992).
13. R. A. Fisher, *Ann. Eugenics,* **7**, 179 (1936).
14. G. E. Peterson and H. L. Barney, *J. Acoust. Soc. Amer.*, **24**, 175 (1952).
15. L. Breiman, *Machine Learning*, **24**, 123 (1996).
16. R. E. Schapire *et. al.*, *Ann. Statist.*, **26**, 1651 (1998).
17. J. H. Friedman *et. al.*, *Ann. Statist.*, **28**, 337 (2000).
18. B. Efron and R. Tibshirani, *An Introduction to Bootstrap* (Chapman and Hall, New York, 1993).