Proceedings of EIS2004:
Fourth International ICSC Symposium on
ENGINEERING OF INTELLIGENT SYSTEMS
February 29-March 2, 2004 Island of Madeira, Portugal

1089

LOCAL ESTIMATE OF DISTRIBUTION MAPPING EXPONENT FOR CLASSIFICATION OF MULTIVARIATE DATA

Marcel Jiřina
Institute of Computer Science AS CR

ABSTRACT

Methods for classification of multivariate data which are based on the nearest neighbors approach solve the problem of classification by an estimate of the probability density at point x of the data space by ratio i/V. i is the number points of a given class of the training set in a suitable ball of volume V with center at point x. The method proposed is based on notion of distribution mapping exponent and its local estimate q for each point x. Distances of all points of a given class of the training set from a given (unknown) point x are used for the probability density estimate. It is shown that the sum of reciprocals of q-th power of these distances can be used as the probability density estimate. The classification quality was tested and compared with other methods using multivariate data from UCI Machine Learning Repository. The method has no tuning parameters.

INTRODUCTION

There is a lot of methods of classification based on the nearest neighbors [1]. They estimate the probability density at point x (a query point [4]) of the data space by ratio i/V_i of number i of points of a given class in a suitable ball of volume V_i with center at point x [6]. These methods need to optimize the best size of the neighborhood, i.e. the number i of points in the neighborhood of the point x or size of volume V_i. The probability density in the feature (data) space is given by training data. The optimal neighborhood size depends on the training data set, i.e. on the character of data as well as on the number of samples of a given class in the training set. Often it is recommended to choose neighborhood size equal to the square root of number of samples of the training set [6].

The method proposed is based on distances of the training set samples x_s, $s = 1, 2, \ldots k$ from point x. It is shown that the sum of reciprocals of q-th power of these distances, where q is a suitable number, is convergent and can be used as a probability density estimate. From the fact of high power of distances in multidimensional Euclidean space, fast convergence, i.e. small influence of distant samples, follows. The speed of convergence is the better the higher dimensionality and the larger q.

The method reminds Parzen window approach [5], [6] but the problem with direct application of this approach is that the step size does not satisfy a necessary convergency condition.

Using distances, i.e. a simple transformation from n-dimensional Euclidean space E_n to one-dimensional Euclidean space E_1, and no iterations, the curse of dimensionality is straightforwardly eliminated. The method can be also considered as a variant of the kernel method, based on a probability density estimator but using a much simpler metric.

Throughout this paper let us assume that we deal with normalized data, i.e. the individual coordinates of the samples of the learning set are normalized to zero mean and unit variance and the same normalization constants (empirical mean and empirical variance) are applied to all other (testing and of unknown class) data. This transformation does not mean any change in form of the distribution, i.e. uniform distribution remains uniform, exponential distribution remains exponential (with $\lambda = 1$ and shifted by 1 to the left), etc.

PROBABILITY DISTRIBUTION MAPPING FUNCTION

Let a query point x be placed without loss of generality in the origin. Let us build balls with their

centers in point x and with volumes V_i, i =1, 2, ... Individual balls are one in another, the (i-1)-st inside the i-th like peels of onion.

The mean density of points in i-th ball is $\rho_i = i/V_i$. The radius of this ball is $r_i = const.\sqrt[n]{V_i}$ because the volume of a ball of radius r in n-dimensional space is $V(r) = const.r^n$. Thus we have constructed a mapping between the mean density ρ_i in an i-th ball and its radius r_i. Then $\rho_i = f(r_i)$. Using tight analogy between density $\rho(z)$ and probability density $p(z)$ one can write $p(r_i) = f(r_i)$ and $p(r_i)$ is the mean probability density in the i-th ball with radius r_i here.

This way a complex picture of probability distribution of points in the neighborhood of a query point x is simplified to a function of a scalar variable. We call this function a probability distribution mapping function $D(x, r)$, where x is a query point, and r the distance from it. More exact definitions follow.

Definition
Probability distribution mapping function $D(x, r)$ of the neighborhood of the query point x is function $D(x,r) = \int_{B(x,r)} p(z)dz$, where r is distance from the query point and $B(x, r)$ is ball with center x and radius r.

Definition
Distribution density mapping function $d(x, r)$ of the neighborhood of the query point x is function $d(x,r) = \frac{\partial}{\partial r} D(x,r)$, where $D(x, r)$ is a probability distribution mapping function of the query point x and radius r.

Note. It is seen that for fixed x the function $D(x, r)$, $r > 0$ is monotonically growing from zero to one. Functions $D(x, r)$ and $d(x, r)$ for x fixed are one-dimensional analogs to the probability distribution function and the probability density functions, respectively.

Power approximation of the probability distribution mapping function
Let us approximate the probability distribution mapping function by parabolic function in form $D(x, r^n) = const.(r^n)^\alpha$. This function is tangent to the vertical axis in point (0, 0) and let it is going through some characteristic points of the distribution.

Definition
Power approximation of the probability distribution mapping function $D(x, r^n)$ is function r^q such that $\frac{D(x,r^n)}{r^q} \to const$ for $r \to 0+$. The exponent q is a distribution mapping exponent. The variable $\alpha = q/n$ we call distribution mapping ratio.

Note. We often omit a multiplicative constant of the probability distribution mapping function.

Using approximation of the probability distribution mapping function by $D(x, r^n) = const.(r^n)^\alpha$ the distribution mapping exponent is $q = n\alpha$.

Note that distribution mapping exponent is influenced by two factors

- True distribution of points of the learning set in E_n.
- Boundary effects, which have the larger influence the larger dimension n and the smaller the learning set size [7].

To overcome the problem of estimation of q using real data, this exponent is estimated by linear regression for each query point as shown in the next Chapter.

DISTRIBUTION MAPPING EXPONENT ESTIMATION

Let the learning set U of total m_T samples be given in the form of a matrix X_T with m_T rows and n columns. Each sample corresponds to one row of X_T and, at the same time, corresponds to a point in n-dimensional Euclidean space E_n, where n is the sample space dimension. The learning set consists of points (rows) of two classes $c \in \{0, 1\}$, i.e. each row (point or sample) corresponds to one class. Then, the learning set $U = U_0 \cup U_1$, $U_0 \cap U_1 = \varnothing$, $U_c = \{x_{cs}\}$, $s = 1, 2, \ldots N_c$, $c = \{0, 1\}$. N_c is the number of samples of class c, $N_0 + N_1 = m_T$, and $x_{cs}=\{ x_{cs1}, x_{cs2},\ldots x_{csn}\}$ is the data sample of class c.

We use normalized data, i.e. each variable x_{csj} (j fixed, s = 1, 2, ... m_T, c = 0 or 1 corresponds to the j-th column of matrix X_T) has zero mean and unit variance.

Let point $x \notin U$ be given and let points x_{cs} of one class be sorted so that index i = 1 corresponds to the nearest neighbor, index i = 2 to the second nearest neighbor, etc. In the Euclidean metrics, $r_i = \|x, x_{ci}\|$ is the distance of the i-th nearest neighbor of class c from point x.
From definition of the distribution mapping exponent it follows that r_i^q shoud be proportional to index i, i.e.

$$r_i^q = ki\,, \quad i = 1, 2, \ldots N_c, \quad c = 0 \text{ or } 1,$$

and where k is a suitable constant. Using logarithm we get

$$q\ln(r_i) = k' + \ln(i)\,, \quad i = 1, 2, \ldots N_c\,. \qquad (1)$$

System of these N_c equations with respect to unknown q can be solved using standard linear regression for both

classes. Thus we get two values of q, q_0 and q_1. To get a single value of q we use arithmetic mean, $q = (q_0N_0 + q_1N_1)/(N_0 + N_1)$.
At this point we can say that q is something like effective dimensionality of the data space including true distribution of points of both classes and boundary effect. In the next chapter we use it directly instead of dimension.

ALL LEARNING SAMPLES APPROACH

In standard nearest neighbors approach a ball with center at point x and a radius sufficiently large to contain just i points nearest to the point x is considered. The volume of the ball is V_i = const. r_i^n in E_n. For each ball with index i and having i points inside it, the probability density estimate is given by formula (C is a constant independent of class)

$$p(x,i) = C\frac{i}{V_i} \cdot$$

For probability density estimation in point x a suitable number of points is used. Often is recommended $i = \sqrt{m_T}$ [6]. We take average values of i/V_i for several i´s here.

We have found that it is possible to use the true distance r_i of point number i from point x instead of numerator i in each fraction i/V_i. Thus if C‘ is a constant independent of class, the probability estimate that x belongs to class c is

$$\overline{p}_c(x) = \frac{C'}{k-1}\sum_{i=2}^{k}\frac{r_i}{V_i} = \frac{C'}{k-1}\sum_{i=2}^{k}1/r_i^{n-1} \cdot \qquad (2)$$

It is easy to see that r_i/V_i as function of i or r_i is monotonically decreasing but i/V_i need not be. It is illustrated in Fig. 1.

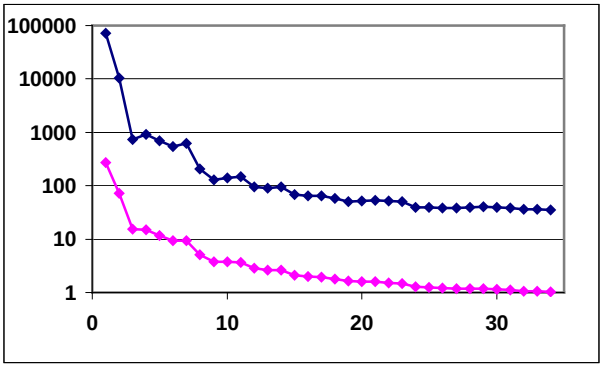

Fig. 1 i/r_i^n (upwards) and $1/r_i^{n-1}$ (below) vs. i.

We have also found that that (2) gives best results for data where $q = n$-1. Based on this experience, let

$$\overline{p}_c(x) = \frac{C'}{k-1}\sum_{i=2}^{k}1/r_i^q \cdot \qquad (3)$$

Note that (3) reminds Parzen window approach [5], [6 Chap. 4.3] with weighting function $K(y) = y^{-q}$ for $y > r_1^q$ and $K(y) = 0$ otherwise, $q \in$ <1, n> and with window width $h = 1$. The problem with direct application of this approach here is that h does not satisfy the necessary condition $\lim_{i\to\infty} h(i) = 0$ [5, eq. (1.8)].
We will prove below that the series $1/r_i^q$ converges with size of r_i for $q > 1$. Thus we have no reason to limit ourselves to the nearest k points and we can use all points in the learning set using $k = N_c$, $c = 0$ or 1. At the same time the ordering of individual components is not essential and we need not sort the samples of X_T with respect to their r_i when using the nearest neighbor approach. (But we need to sort them when estimating distribution mapping exponent q.)

In practical procedure for each query point x we first compute the distribution mapping exponent q using (1) by standard linear regression. After it, we simply sum up all components $1/r_i^q$ and, at the same time, we store the largest component which corresponds to the nearest neighbor of point x which has the smallest r_i^q. In the end we subtract it thus excluding the nearest point. This is made for both classes simultaneously getting numbers S_0 and S_1 for both classes. Their ratio gives a value of the discriminant function, here the Bayes ratio. We can get also probability estimation that the point $x \in E_n$ is of class 1:

$$R(x) = \frac{S_1}{S_0} \quad \text{or} \quad p_1(x) = \frac{S_1}{S_1 + S_0} \cdot$$

Then for a threshold (cut) θ chosen, if $R(x) > \theta$ or $p_1(x) > \theta$ then x belongs to class 1 else to class 0.

The method is very close to the nearest neighbor as well as kernel methods. The procedure described in the text of the above Eq. (2) is nothing else than the nearest neighbor method. Simply an average of several neighborhoods is taken, but the number of points inside the ball is changed to distances. From the point of view of kernel methods, the kernel is or would be $K(x) = \| x - x_i \|^{-q}$ with Euclidean norm $\|.\|$ in E_n. There is no smoothing (bandwidth) parameter. The problem is that this kernel is difficult to consider as a probability distribution function according to the definition of a kernel [1]. Taking $\|x\text{-}x_i\| = r$ we have $K(r) = r^{-q}$ and

integrals $\int_{-\infty}^{\infty} K(r)dr$ or $\int_{0}^{\infty} K(r)dr$ are not convergent; they should be equal to 1 or at least finite.

PROBABILITY DENSITY ESTIMATION

Let us look at the problem what is the relation of part D_i of space E_n which falls on i nearest neighbors of the given point x. We will assume the following:

Assumption 1: Let there be points in the Euclidean space E_n distributed uniformly in the sense that the distribution of each of the n coordinates is uniform. Let i be the order number of the i-th nearest neighbor to the point x. Let r_i be the distance of the i-th nearest neighbor of the given point $x \in E_n$ from point x_i. Let D be a constant, $q \in (1, n)$ be a constant, and $\overline{D}_i$ be the mean value of the variable r_i^q, and let it hold

$$\overline{D}_i = iD \ .$$

Comment: Under Assumption 1 by "the part D_i of the space E_n" we do not mean a volume of a ball with the center in the point x and radius r_i but, in fact (except for a multiplicative constant), a ball of the same center and radius but in the space of dimension given by constant q, i.e. in the E_q. The basis for introducing Assumption 1 is finding as follows. By simulation one can find that the relation $\overline{V}_i = iV$ where V is a constant does not hold but for some $q \le n$ it holds $\overline{D}_i = \overline{r}_i^{\,q} = iD$ where i is the number of the i-th nearest neighbor of point $x \in E_n$ and D is a constant. From it follows that the q-th power grows linearly and it is basis of Assumption 1.

Theorem 1: Let Assumption 1 be valid, and let $\overline{D}_i$ be mean of $D_i = r_i^q$, $\overline{V}_i$ be mean of $V_i = cr_i^n$ where c is a constant. Then for the probability density $p(i)$ of points in the neighborhood of point x it holds $p(i) = p(\overline{D}_i)$.

Proof: The $p(i)$ is probability density and at the same time due to Assumption 1 $1/\overline{D}_i$ is proportional to $p(i)$. Then there is a constant K that $p(\overline{D}_i) = \frac{K}{\overline{D}_i}$ and $p(\overline{D}_i) = p(i)$. □

THE PROOF OF CONVERGENCE

Theorem 1 states that probability density is propotional to $1/r_i^q$ and formula (3) uses the sum of these ratios supposing to get a reasonable number for probability density estimation. So it is supposed that for a number of samples going to infinity, the sum would be convergent.

Theorem 2: Let exist a mapping of probability density of points of class c in E_n, $E_n \to E_1$: $p(x_{ci}) = p(r_{ci}^q)$ so that

$$K/r_{c1}^q = p(x_{c1}), \quad K/(r_{c2}^q - r_{c1}^q) = p(x_{c2}),$$
$$\cdots K/(r_{cNc}^q - r_{c(Nc-1)}^q) = p(x_{cNc}), \tag{4}$$

where K is a fixed constant that has the same value for both classes. Let exist a constant $\varepsilon > 0$ and index $k > 2$ so that for each $i > k$ it holds

$$p(x_{ci}) \le \frac{p(x_{c2})}{(1+(i-k)\varepsilon)^{i-k}}. \tag{5}$$

Then

$$S_c = \sum_{i=2}^{N_c} \frac{1}{r_{ci}^q} = p(x_{c2})K(1+C_c), \tag{6}$$

where K and C_c are finite constants.

Proof: First we arrange (6) in form

$$S_c = \sum_{i=2}^{N_c} \frac{1}{r_{cj}^q} = \frac{1}{r_{c2}^q} + \sum_{i=3}^{N_c} \frac{1}{r_{c2}^q + \Delta_{c3} + \Delta_{c4} + \mathrm{K} + \Delta_{ci}} \ .$$

Then using mapping (4) introduced we get

$$S_c = Kp_{c2} + K\sum_{i=3}^{Nc} \frac{1}{\frac{1}{p_{c2}} + \frac{1}{p_{c3}} + \mathrm{K} + \frac{1}{p_{ci}}} =$$
$$= p_{c2}K(1 + \sum_{i=3}^{Nc} \frac{1}{1 + \frac{p_{c2}}{p_{c3}} + \mathrm{K} + \frac{p_{c2}}{p_{ci}}}) \equiv p_{c2}K(1 + \sum_{i=3}^{Nc} P_i) \tag{7}$$

For individual elements p_{c2} / p_{cj} in denominators of fractions in the sum it holds

$$\frac{p_{c2}}{p_{cj}} = \frac{p_{c2}(1+(i-k)\varepsilon)^{i-k}}{p_{c2}} = (1+(i-k)\varepsilon)^{i-k}.$$

Using condition (5) the summed elements P_k, P_{k+1}, … in (7) since the k-th have form

$$P_k = \frac{1}{C}, \quad P_{k+1} = \frac{1}{C+1+\varepsilon}, \quad P_{k+2} = \frac{1}{C+1+\varepsilon+(1+\varepsilon)^2},$$
$$P_{k+i} = 1/[C + (1+\varepsilon) + (1+2\varepsilon)^2 + \ldots + (1+i\varepsilon)^i] \ .$$

Then according to d'Alembert's criterion

$$\frac{P_{k+i+1}}{P_{k+i}} = \frac{C + (1+\varepsilon) + (1+2\varepsilon)^2 + \ldots + (1+i\varepsilon)^i}{C + (1+\varepsilon) + (1+2\varepsilon)^2 + \ldots + (1+i\varepsilon)^i + (1+(i+1)\varepsilon)^{i+1}} < 1$$

$\forall i > 0$ and $\forall \varepsilon > 0$. Then the series is convergent. □

Notes:

a) In the statement of the theorem the sum need not start just by index $i = 2$. We can start with the nearest neighbor ($i = 1$) or other neighbors ($i > 2$). The value

$i = 2$ is given by a compromise between the error caused by the small value and the large variability of $\Delta_{c1} = r_{c1}$, and the inaccuracy caused by the larger distance from point x for $i > 2$, see Chap. III.

b) The last condition (5) defines the speed of diminishing the tail of the distribution; probably condition that the distribution should have the mean would suffice.

RESULTS - TESTING THE CLASSIFICATION ABILITY

The classification algorithm was written in c++ as SFSloc7 program and tested using tasks from UCI Machine Learning Repository [8]. Tasks of classification into two classes for which data about previous tests are known were selected: "Adult", "German", "Heart", "Ionosphere" , and "Vote".

Table 1 shows results obtained by SFSloc7 in comparison with results found in [8].

The task "Adult" is to determine whether a person makes over 50000 $ a year.

The task "German" is about whether the client is good or bad to lend him money.

The task "Heart" indicates absence or presence of heart disease for patient.

For he task "Ionosphere" the targets were free electrons in the ionosphere. "Good" radar returns are those showing evidence of some type of structure in the ionosphere. "Bad" returns are those that do not; their signals pass through the ionosphere.

We do not describe these tasks in detail here as all can be found in [8]. For each task the same approach to testing and evaluation was used as described in [8]. In Table 1 results are shown together withs results for other methods as given in [8]. For each task methods are sorted according to classification error, the method with the best – lowest error first. It is seen that for some tasks SFSloc7 is good but there are tasks where the method is worse than average – it would be strange to outperform all methods for all tasks. The method is totally parameterless. There is no parameter for tuning to get the best result. The method simply works satisfactorily or not; there is nothing to try more.

CONCLUSIONS

The method described is based on notion of distribution mapping exponent q and its local estimate for each query point x. The theorem on convergence was formulated and proved and convergence estimation was shown. It was found that the higher dimensionality, the better.

The method has no tuning parameters: No neighborhood size, no convergence coefficients etc. need to be set up in advance to assure convergence. There is no true learning phase. In the „learning phase" only normalization constants are computed and thus this phase is several orders of magnitude faster than the learning phase of neural networks or many other methods [2], [3]. In the recall phase for each sample to be classified the learning set is searched twice, once for finding the local value of the distribution mapping exponent q, and second for all samples of the learning set elements of sum (3) are computed. The amount of computation is thus proportional to the learning set size, i.e. the dimensionality times the number of the learning samples.

ACKNOWLEDGMENT

This work was supported by the Ministry of Education of the Czech Republic under project No. LN00B096.

REFERENCES

[1] Silverman, B. W.: Density Estimation for Statistics and data Analysis. Chapman and Hall, London, 1986.

[2] Bock, R. K. et al.: Methods for multidimensional event classification: a case study. To be published as Internal Note in CERN, 2003.

[3] Hakl, F., Hlaváček, M., Kalous, R.: Application of Neural Networks Optimized by Genetic Algorithms to Higgs Bosson Search. In: The 6th World Multi-Conference on Systemics, Cybernetics and Informatics. Proceedings. (Ed.: Callaos, N., Margenstern, M., Sanchez, B.) Vol.: 11. Computer Science II. - ISSS, Orlando 2002, pp. 55-59 (ISBN: 980-07-8150-1) Held: ISAS SCI 2002 /6./, Orlando, US, 02.07.14-02.07.18

[4] Hinnenburg, A., Aggarwal, C.C., Keim, D. A.: What is the nearest neighbor in high dimensional spaces? Proc. of the 26th VLDB Conf., Cairo, Egypt, 2000, pp 506-515.

[5] Parzen, E.: On Estimation of Probability Density Function and Mode. The Annals of Mathematical Statistics, Vol. 33, No. 3 (Sept. 1962), pp. 1065-1076.

[6] Duda, R., Hart, P., Stork, D. G.: Pattern Classification. John Wiley and Sons, 2000.

[7] Arya, S., Mount, D.M., Narayan, O., Accounting for Boundary Effects in Nearest Neighbor Searching. Discrete and Computational Geometry Vol.16 (1996), pp. 155-176.

[8] UCI Machine Learning Repository. http://www.ics.uci.edu/~mlearn/MLSummary.html

"German"			**"Heart"**			**"Adult"**			**"Ionosphere"**		
Algorithm	Error	Note	Algorithm	Error	Note	Algorithm	Error	Note	Algorithm	Error	Note
SFSloc7	0.520	1; 2	**SFSLoc7**	0.357	3	FSS Naive Bayes	0.1405		IB3	0.0330	6; 7
Discrim	0.535		Bayes	0.374		NBTree	0.1410		backprop	0.0400	8
LogDisc	0.538		Discrim	0.393		C4.5-auto	0.1446		**SFSloc7**	0.0596	9
Castle	0.583		LogDisc	0.396		IDTM (Decision table)	0.1446		Ross Quinlan's C4	0.0600	10
Alloc80	0.584		Alloc80	0.407		HOODG	0.1482		nearest neighbor	0.0790	
Dipol92	0.599		QuaDisc	0.422		C4.5 rules	0.1494		"non-linear" perceptron	0.0800	
Smart	0.601		Castle	0.441		OC1	0.1504		"linear" perceptron	0.0930	
Cal	0.603		Cal5	0.444		C4.5	0.1554				
Cart	0.613		Cart	0.452		Voted ID3 (0.6)	0.1564				
QuaDisc	0.619		Cascade	0.467		CN2	0.1600				
KNN	0.694		KNN	0.478		Naive-Bayes	0.1612				
Default	0.700		Smart	0.478		Voted ID3 (0.8)	0.1647				
Bayes	0.703		Dipol92	0.507		T2	0.1684				
IndCart	0.761		Itrule	0.515		**SFSloc7**	0.1786				
BackProp	0.772		BayTree	0.526		1R	0.1954				
BayTree	0.778		Default	0.560		Nearest-neighbor (3)	0.2035	4			
Cn2	0.856		BackProp	0.574		Nearest-neighbor (1)	0.2142				
Ac2	0.878		LVQ	0.600		Pebls	Crashed	5			
Itrule	0.879		IndCart	0.630							
NewId	0.925		Kohonen	0.693							
LVQ	0.963		Ac2	0.744							
Radial	0.971		Cn2	0.767							
C4.5	0.985		Radial	0.781							
Kohonen	1.160		C4.5	0.781							
Cascade	100.0		NewId	0.844							

Notes

1	for threshold 0.413	6	parameter settings: 70% and 80% for acceptance and dropping respectively
2	numeric data	7	(Aha & Kibler, IJCAI-1989)
3	for threshold 0.24	8	an average of over ..
4	for threshold 0.868482	9	for threshold 0.550254
5	Unknown why (bounds WERE increased)	10	no windowing

Table 1. Comparison of classification error of SFSloc7 for different tasks with results for another classifiers as given by [8].