ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 169-174

Dimensionality Reduction and Classification Using the Distribution Mapping Exponent

Marcel Jiřina
Institute of Computer Science AS CR
Pod vodárenskou věží 2, 182 07 Praha 8 – Libeň
Czech Republic
marcel@cs.cas.cz

Abstract: Probability distribution mapping function, which maps multivariate data distribution to the function of one variable, is introduced. Distribution-mapping exponent (DME) is something like effective dimensionality of multidimensional space. The method for classification of multivariate data is based on the local estimate of distribution mapping exponent for each point. Distances of all points of a given class of the training set from a given (unknown) point are searched and it is shown that the sum of reciprocals of the DME-th power of these distances can be used as 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

In this paper we deal with distances in multidimensional space and try to simplify a complex picture of probability distribution of points in this space introducing mapping functions of one variable. This variable is the distance from the given point (the query point x [3]) in a multidimensional space. From it it follows that mapping functions are different for different query points and this is the cost we pay for simplification from n variables in n-dimensional space to one variable. We will show that this cost is not very high – at least in the application presented here.

The method proposed is based on the distances of the training set samples x_s, $s = 1, 2, \ldots k$ from point x similarly as in methods based on the nearest neighbors [1][5]. It is shown here that the sum of reciprocals of the q-th power of these distances, where q is a suitable number, it is convergent and can be used as a probability density estimate. It will be seen that the speed of convergence is the better the higher is the dimensionality and the larger q.

The method reminds Parzen window approach [4], [5] but the problem with the direct application of this approach is that the step size does not satisfy a necessary convergence condition.

Because of exponential nature of estimation using q, it is very close to intrinsic dimension of data or correlation dimension [9] and its estimation by the Grassberger-Procaccia's algorithm [10][11]. The essential difference is that q is understood locally, i.e. for each point x separately, and the correlation dimension is the feature of the whole data space. It will be seen, that although q has different objective here, the algorithm is, in fact, a simplified version of Grassberger-Procaccia's algorithm.

Throughout this paper let us assume that we deal with standardized data, i.e. the individual coordinates of the samples of the learning set are standardized to zero mean and unit variance, and the same standardization constants (empirical mean and

ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 169-174

empirical variance) are applied to all other (testing) data. This transformation does not mean any change in the form of the distribution, i.e. uniform distribution remains uniform, 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, ... A complex picture of probability distribution of points in the neighborhood of a query point x can be 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 say that 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 the distance from the query point and $B(x, r)$ is the ball with center x and radius r. 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. For illustration see Fig. 1.

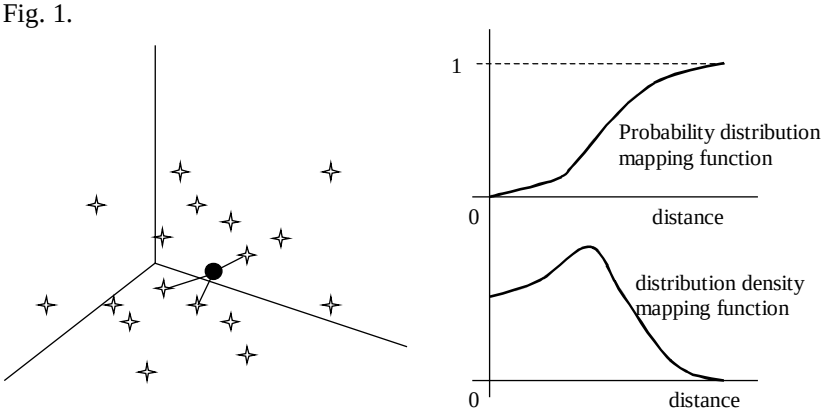

Fig. 1. Data in a multidimensional space and corresponding probability distribution mapping function and distribution density mapping function.

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 go through some characteristic points of the distribution.
Definition. The 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 $\alpha = q/n$ is the distribution mapping ratio.

We often omit a multiplicative constant of the probability distribution mapping function. The distribution-mapping exponent is influenced by true distribution of the points of the learning set in E_n, and by boundary effects, which have the larger the influence, the larger the dimension n and the smaller the learning set size [1], [6].

ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 169-174

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 $U = U_0 \cup U_1$, $U_0 \cap U_1 = \emptyset$, $U_c = \{x_{cs}\}$, $s = 1, 2, \ldots N_c$, $c = \{0, 1\}$. 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 standardized data, i.e. each variable x_{csj} (j fixed, $s = 1, 2, \ldots m_T$, $c = 0$ 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 the definition of the distribution mapping exponent it follows that r_i^q should be proportional to index i, i.e.

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

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

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

The 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 the weighted arithmetic mean, $q = (q_0 N_0 + q_1 N_1)/(N_0 + N_1)$.

At this point we can say that q is something like an effective dimensionality of the data space including true distribution of points of both classes and boundary effect and related to point x. In the next chapter we use it directly instead of dimension.

All learning samples approach

Let us define

$$p_c(x) = \frac{C}{k-1} \sum_{i=2}^{k} 1 / r_i^q \quad , \tag{3}$$

where C is a constant. We show below, that $p_c(x)$ is a probability density estimate. The series $1/r_i^q$ converges with the size of r_i for $q > 1$ and 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. In practical procedure for each query point x we first compute the distribution mapping exponent q using (2) by standard linear regression. Then we simply sum up all components $1/r_i^q$ excluding the nearest point. This is made for both classes simultaneously getting numbers S_0 and S_1 for both classes. Then we can get the Bayes ratio or a probability estimation that the point $x \in E_n$ is of class 1:

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

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.

ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 169-174

Probability density estimation

Assumption 1: Let the points in the Euclidean space E_n be 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

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

Theorem 1

Let Assumption 1 be valid, and let $\overline{\Delta}_i$ be mean of $\Delta_i = r_i^q - r_{i-1}^q$, $\overline{D}_i$ be mean of $D_i = r_i^q$, $\overline{V}_i$ be mean of $V_i = cr_i^n$ where c is a constant. Moreover, let a constant K exists such that $p(\overline{\Delta}_i) = K/\overline{\Delta}_i$. Then for the probability density $p(i) = K'i/\overline{V}_i$ of the points in the neighborhood of point x it holds that $p(\overline{\Delta}_i) = p(\overline{D}_i) = p(i)$, where $p(\overline{D}_i) = \dfrac{iK}{\overline{D}_i}$.

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

Results - testing the classification ability

The classification algorithm was written in c++ as SFSloc7 program and tested using tasks from UCI Machine Learning Repository [7]. In Table 1 the results are shown together with results of other methods as given in [7].

Table 1. (see the next page) Comparison of the classification error of SFSloc7 for different tasks with results for other classifiers as given by [7]. Notes to Table 1:

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

ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 169-174

"German"			**"Heart"**		
Algorithm	Error	Note	Algorithm	Error	Note
SFSloc7	0.520	1; 2	**SFSLoc7**	0.357	3
Discrim	0.535		Bayes	0.374	
LogDisc	0.538		Discrim	0.393	
Castle	0.583		LogDisc	0.396	
Alloc80	0.584		Alloc80	0.407	
Dipol92	0.599		QuaDisc	0.422	
Smart	0.601		Castle	0.441	
Cal	0.603		Cal5	0.444	
Cart	0.613		Cart	0.452	
QuaDisc	0.619		Cascade	0.467	
KNN	0.694		KNN	0.478	
Default	0.700		Smart	0.478	
Bayes	0.703		Dipol92	0.507	
IndCart	0.761		Itrule	0.515	
BackProp	0.772		BayTree	0.526	
BayTree	0.778		Default	0.560	
Cn2	0.856		BackProp	0.574	
Ac2	0.878		LVQ	0.600	
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	

"Adult"			**"Ionosphere"**		
Algorithm	Error	Note	Algorithm	Error	Note
FSS Naive Bayes	0.1405		IB3	0.0330	6; 7
NBTree	0.1410		backprop	0.0400	8
C4.5-auto	0.1446		**SFSloc7**	0.0596	9
IDTM Dec. table	0.1446		Ross Quinlan's C4	0.0600	10
HOODG	0.1482		nearest neighbor	0.0790	
C4.5 rules	0.1494		"non-linear" perceptr.	0.0800	
OC1	0.1504		"linear" perceptron	0.0930	
C4.5	0.1554				
Voted ID3 (0.6)	0.1564				
CN2	0.1600				
Naive-Bayes	0.1612				
Voted ID3 (0.8)	0.1647				
T2	0.1684				
SFSloc7	0.1786	4			
1R	0.1954				
Nearest-neighbor 1	0.2035				
Nearest-neighbor 2	0.2142				
Pebls	Crashed	5			

ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 169-174

Conclusions

In this paper we dealt with simplified representation of probability distribution of points in multidimensional Euclidean space including boundary effects. A new method for classification based on the notion of distribution mapping exponent and its local estimate was developed. It was found that the higher the dimensionality, the better.

The method has no tuning parameters and there is no true learning phase. In the ”learning phase“ only standardization 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], [7], [8].

Acknowledgement

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] 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.

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

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

[6] 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.

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

[8] 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. - Orlando, IIIS 2002, pp. 55-59 (ISBN:980-07-8150-1)

[9] Smith, A. (Ed.): CATS Book of Jargon Normalization. A Dictionary for Interdisciplinary Science. http://www.maths.ox.ac.uk/~hardenbe/defs/defs.html, especially http://www.maths.ox.ac.uk/~hardenbe/defs/node7.html.

[10] Grassberger, P., Procaccia, I.: Measuring the Strangeness of Strange Attractors. Physica Vol. 9D (1983), pp. 189-208.

[11] Camastra, P., Vinciarelli, A.: Intristic Dimension Estimation of Data: An Approach based on Grassberger-Procaccia’s Algorithm. Neural Processing Letters Vol. 14 (2001), No. 1, pp. 27-34.