Journal of Classification 32:305-326 (2015)
DOI: 10.1007/s00357-015-9174-2

Classification Using the Zipfian Kernel

Marcel Jiřina

Institute of Computer Science AS CR, Czech Republic

Marcel Jiřina, Jr.

Czech Technical University in Prague, Czech Republic

Abstract: We propose to use the Zipfian distribution as a kernel for the design of a nonparametric classifier in contrast to the Gaussian distribution used in most kernel methods. We show that the Zipfian distribution takes into account multifractal nature of data and gives a true picture of scaling properties inherent in data. We also show that this new look at data structure can lead to a simple classifier that can, for some tasks, outperform more complex systems.

Keywords: Kernel machine; Zipfian kernel; Multivariate data; Correlation dimension; Harmonic series; Classification.

1. Introduction

Proper selection of kernel and window width is essential for a good probability density estimation and for classification in classification tasks (Schölkopf and Smola 2002; Scott 1992). This problem is solved solely from the statistical point of view. However, the role of spatial correlations and the effective data dimensionality are not considered in kernel methods.

This work was supported by Technology Agency CR under project of series ALFA No. TA01010490 and by the Czech Technical University in Prague, Faculty of Information Technology, RVO: 68407700. We also thank the Institute of Computer Science of the Czech Academy of Sciences for its support in submitting application of patent for the classifier described.

Authors' Addresses: M. Jiřina, Institute of Computer Science AS CR, v.v.i., Pod vodárenskou věži 2, 182 07 Prague 8 – Libeň, Czech Republic, email: marcel@cs.cas.cz; M. Jiřina, Jr., Faculty of Information Technology, Czech Technical University in Prague, Thákurova 9, 160 00 Prague 6, Czech Republic, e-mail: marcel.jirina@fit.cvut.cz.

Published online: 12 April 2015

Here we show that the Zipfian distribution (Maslov 2005; Zipf 1968) can take into account the fractal nature of data, and can be a suitable alternative to the standard kernel functions. We prove here that the kernel method with the Zipfian kernel gives an unbiased approximation of the class probability of the given point. For proof we use here or necessarily redefine some notions from the multifractals theory. Singularity exponents, as well as scaling exponents are widely used in multifractal chaotic series analysis and can be related to data that do not form a series. It was shown already by Mandelbrot (1982) that any data might possess fractal or multifractal properties. We use these exponents for proof of our method of classification. As there is no time scale, even no ordering of samples, one cannot use such a tool as wavelet functions.

Furthermore, this classification method is truly nonparametric as there is no need to set up the window width common in most kernel methods.

Our results demonstrate that the kernel method can be related to the fractal nature of data and to the harmonic series (Maslov 2005; Schmuland 2003) via the Zipfian distribution.

This work can be a starting point for more detailed description of local behavior of multivariate and not exactly self-similar fractal data, and for the development of new approaches to data analysis including classification problems.

2. Preliminaries

2.1 Data and the Learning Set

Let the learning set U of total N samples be given. Each sample $x_t=\{x_{t1}, x_{t2},\dots x_{tn}\}$; $t = 1, 2, \dots N$, $x_{tk} \in R$; $k = 1, 2, \dots, n$ corresponds to a point in n-dimensional metric space M_n, where n is the sample space dimension. For each $x_t \in U$ a class function T: $R^n \rightarrow \{1, 2, \dots C\}$: $T(x_t) = c$ is introduced. With the class function the learning set U is decomposed into disjoint classes $U_c = \{x_t \in U \mid T(x_t) = c\}$; $c \in \{1, 2, \dots, C\}$, $\bigcup_{c=1}^{C} U_c$, $U_c \cap U_d = \emptyset$; $c, d \in \{1, 2, \dots, C\}$; $c \neq d.$ Cardinality of set U_c let be N_c ; $\sum_{c=1}^{C} N_c = N$.

As we need to express which sample is closer or further from some given point x, we can rank points of the learning set according to distance r_i of point x_i from point x. Therefore, let points of U be indexed (ranked) so that for any two points $x_i, x_j \in U$ there is $i < j$ if $r_i < r_j$; $i, j = 1, 2, \dots N$, and class $U_c = \{x_i \in U \mid T(x_i) = c\}$. Of course, the ranking depends on point x and eventually metrics of M_n. We use Euclidean (L_2) and absolute (Manhattan, L_1) metrics here. In the following indexing by i means ranking in the way just introduced.

2.2 Zipfian Kernel

The Zipfian distribution (Zipf's law) (Maslov 2005; Zipf 1968) predicts that out of a population of N elements, the frequency of elements of rank i, $f(i;s,N)$, is given by probability mass function

$$f(i;s,N) = \frac{1/i^s}{\sum_{t=1}^{N} 1/t^s}, \tag{1}$$

where N is the number of elements, i is their rank, s is the value of the exponent characterizing the distribution. The law may also be written as:

$$f(i;s,N) = \frac{1}{i^s H_{N,s}},$$

where $H_{N,s}$ is the Nth generalized harmonic number.

The simplest case of Zipf's law is a "$1/f$ function" arising when s = 1. Given a set of Zipfian distributed frequencies of occurrence of some objects, sorted from the most common to the least common, the second most common frequency will occur ½ as often as the first. The third most common frequency will occur 1/3 as often as the first, and so on. Over fairly wide ranges, and to a fairly good approximation, many natural phenomena obey Zipf's law. Note that in the case of a "$1/f$ function", i.e. $s = 1$, N must be finite and its denominator is equal to H_N, the so-called harmonic number, i.e. the sum of truncated harmonic series (Schmuland 2003); otherwise the denominator is a sum of harmonic series, which is divergent. However, if exponent s exceeds 1, $s > 1$, then the series is convergent,

$$\zeta(s) = \sum_{t=1}^{\infty} \frac{1}{t^s} < \infty \ ,$$

where ζ is Riemann's zeta function.

2.3 Kernel Methods

Origin of kernel methods can be seen in the oldest and most widely used density estimator, the histogram and in the k-th nearest neighbor (k-NN) method.

For the histogram, given an origin x_0 and a bin width h, we define the bins of the histogram to be the intervals $[x_0 + mh, x_0 + (m + 1)h]$ for positive and negative integers m. The intervals have been chosen closed on the left and open on the right for definiteness. The histogram density estimate (naive estimator) is then defined by

$$\hat{f}(x) = \frac{1}{Nh}\left(No.\ of\ X_i\ in\ same\ bin\ as\ x\right).$$

Note that, to construct the histogram, we have to choose both an origin and a bin width; it is the choice of bin width, which, primarily, controls the amount of smoothing inherent in the procedure (Silverman 1992). By introducing the weight function

$$w(x) = \begin{cases} 1/2 & if\ |x| < 1 \\ 0 & otherwise \end{cases} .$$

The naive estimator can be rewritten in form

$$\hat{f}(x) = \frac{1}{N}\sum_{i=1}^{n} \frac{1}{h} w\left(\frac{x - x_i}{h}\right) .$$

This naive estimator can be generalized by replacing the weight function *w* by a kernel function *K* that satisfies integral condition

$$\int_{-\infty}^{\infty} K(x)dx = 1 .$$

Usually, *K* will be a symmetric probability density function, the normal density, for instance, or the weight function *w* used in the definition of the naive estimator. By analogy with the naive estimator, the *kernel estimator* with kernel *K* is defined by

$$\hat{f}(x) = \frac{1}{Nh}\sum_{i=1}^{N} K\left(\frac{x - x_i}{h}\right) ,$$

where *h* is the window width, also called the smoothing parameter or bandwidth.

Important are conditions when the estimator is nonparametric, i.e. is consistent in large class of density functions. In that case, the kernel should be Mercer's kernel (Herbrich 2002) which implies that the kernel should be positive semidefinite, which also includes the integral condition above. Not all kernels need to be Mercer's kernels. For example, the neural network with hyperbolic tangent transfer function is a kernel machine but a *tanh* kernel is never positive definite (Schölkopf and Smola 2002).

It turns out that many algorithms may be viewed as generalized kernel estimators; see Scott (1992, Theorem 6.6), which says that any estimator that is continuous and Gateaux generalized differentiable can be written in the form of a mean of kernel functions. In kernel density estimators the kernel function is centered at the data sample location and defines an influence region around it. Usually, the influence region has its maximum at the data sample location and decreases in value with respect to the distance from that location. A scale parameter, also called bandwidth or window width, controls the kernel function smoothing over the surrounding space. Most studies choose the Gaussian function as the kernel due to its properties of approximation and for having derivatives of all orders over the entire space. The performance of kernel estimators crucially depends on the value of the kernel's bandwidth (Bors and Nasios 2013).

The nearest neighbor class of estimators represents an attempt to adapt the amount of smoothing to the "local" density of data (Silverman 1992). The degree of smoothing is controlled by an integer *k*, chosen to be considerably smaller than the learning set size; typically $k < N/2$. The *k*-th nearest neighbor density estimate is then defined by

$$f(x) = \frac{k}{NV_k(x)} ,$$

where $V_k(x)$ is volume of ball centered at *x* and containing *k* points nearest to *x*. While the naive estimator is based on the number of observations falling in a box of fixed width centered at the point of interest, the nearest neighbor estimate is inversely proportional to the size of the ball needed to contain a given number of observations.

It is possible to generalize the nearest neighbor estimate to provide an estimate related to the kernel estimate. Let $K(x)$ be a kernel function integrating to one. Then the generalized *k*-th nearest neighbor estimate is defined by

$$\hat{f}(x) = \frac{1}{NV_k(x)} \sum_{i=1}^{N} K\left(\frac{x - X_i}{d_k(x)} \right)$$

It can be seen at once that this is precisely the kernel estimate evaluated at *x* with window width $V_k(x)$. Thus, the overall amount of smoothing is governed by the choice of the integer *k* but the window width used at any particular point depends on the density of observations near that point. The ordinary *k*-th nearest neighbor estimate is the special case when *K* is the uniform kernel.

Here we show that a suitable alternative to the standard kernel functions can be the Zipfian distribution (Maslov 2005; Zipf 1968) with probability mass function $\frac{1/i}{H_N}$, where *i* is the order number of a near neighbor, and H_N is the *N*-th harmonic number. We show that this distribution can take into account a fractal nature of data. The use of the Zipfian kernel is essential as no other kind of kernel can express it better.

The method proceeds from observation as follows. If any data are of a fractal or multifractal nature then distances between points are governed by the scaling law. It means that distances between pairs of points and especially distances between fixed point *x* and its neighbors follow this behavior. Thus if the *i*-th neighbor lies at distance r_i from point *i* then a scaling exponent q exists such that $i = Kr_i^q$ has the form of linear dependence (due to proper selection of q) as was shown by Grassberger and Procaccia (1983). Now, if we suppose that the influence of the i-th neighbor of some class on the probability that point x is of the same class is inversely proportional to r_i^q i.e. to its index i, we can easily estimate the probability that point x is of that class as mentioned above. The influence function ($1/i$) can be seen as a kernel function centered into the i-th neighbor and the approach as kernel method. It also follows from this that kernel considered is the Zipfian kernel and that kernel of another form cannot combine partial "influences" of individual neighbors in such a simple way.

Considering the two-class classification problem under the assumption of equal priors, the k-NN classifier in its simplest form states $f(x)= j/k$, where j is the number of points of class c among all k points nearest to the point x.

In our method we center the Zipfian kernel on each point x_i of learning data set of size N. At point x the kernel gives value (probability density, in fact probability mass as Zipfian distribution is discrete) $1/i$; x_i is i-th nearest neighbor of x. Then, the sum of all these reciprocals gives the sum of harmonic series up to the N-th element, i.e. the harmonic number H_N. Let the sum of these reciprocals only for points of given class c be S_c. We show here that then the probability that point x is of class c is given by ratio S_c/H_N.

From the point of view of the kernel approach, it is easily seen that the above approach could be considered as kernel method but the Zipfian kernel does not fulfill kernel conditions discussed already. Note that Zipfian distribution has a heavy "hyperbolic" tail such that density mass function diminishes with $1/t$ for large (integer) t (the random variable) while heavy tail Cauchy distribution and Student's t probability density functions diminish polynomially as $1/t^2$ or faster, and the density of Laplace's double exponential distribution diminishes exponentially.

3. The Method

3.1 Intuitive Basis

Here we explain the method of probability estimation proposed on the following illustrative example. Let us consider partial influences of individual points to the probability that point x is of class c; we consider two classes only here. Both classes have the same cardinality. Each point of class c in the neighborhood of point x adds a little to the probability that point x is of class c, where $c = \{1, 2\}$ is the class mark. This influence is the larger the closer the point considered is to point x and vice versa. This observation is based on the finding of Cover and Hart (1967) that the first nearest neighbor has the largest influence on proper estimation to what class point x belongs. Suppose that the influence on the probability that point x is of class c of the nearest neighbor of class c is 1, the influence of the second nearest neighbor is $^1/_2$, the influence of the third nearest neighbor is $^1/_3$ etc. Just these values are related to Zipfian distribution.

From the kernel methods point of view, we center the Zipfian kernel to each point x_i of data set. At point x the kernel gives probability mass proportional to $1/i$ (we use exponent $s = 1$); x_i is i-th nearest neighbor of x. Summing up these values for points x_i of class 1 gives number S_1, for points of class 2 number S_2. The estimate of the probability that point x is of class 1 is

$$\hat{p}(c = 1 \mid x) = \frac{S_1(x)}{S_1(x) + S_2(x)} .$$

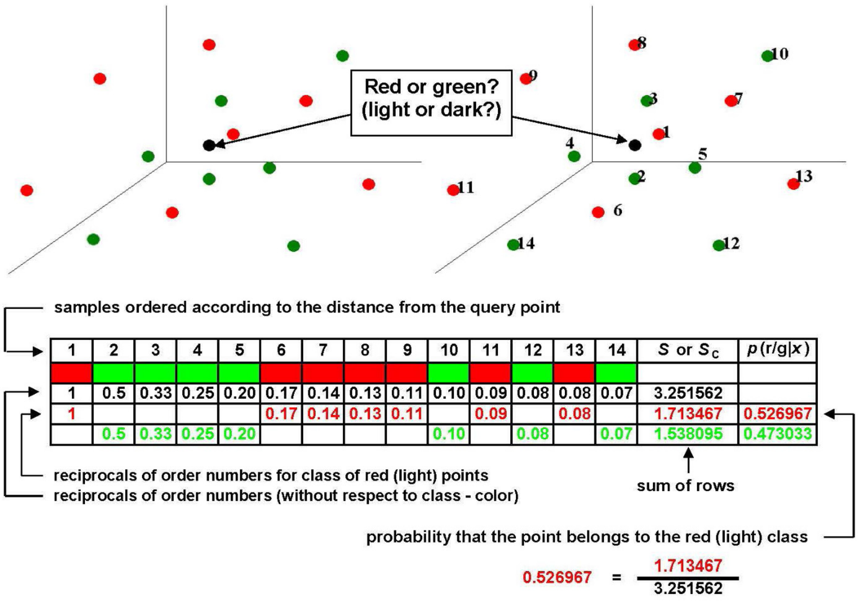
The classification procedure is depicted in Figure 1. The problem is: What is color of given point x depicted in black at the left upper part of picture? First we rank points of the learning set according to their distances from point x as shown at the right upper part of picture. There are 14 points here, 7 red, 7 green as shown in the upper lines in the table below pictures. Reciprocals of rank numbers are in the third line. In the fourth and fifth line there are reciprocals of ranks of points x_i from sets $U_{c=red}$ and $U_{c=green}$. In the rightmost two columns of table are corresponding sums and estimated probabilities that point x is red (0.526967) or green (0.473033). Setting threshold $\theta = 0.5$ we can state that point x is red.

From another point of view, let $\mathbf{Pr}\left(T(x) = c \mid T(x_i) = c\right)$ be the probability that the given point x is of class c if neighbor point number i is of the same class as point x. Note that points of the learning set U are indexed so that for any two points $x_i, x_j \in U$ there is $i < j$ if $r_i < r_j$; $i, j = 1, 2, \dots N$. In the following K is a constant that is used to normalize the probability that point x belongs to a class to 1:

For the first (nearest) point $i = 1$ $\qquad \mathbf{Pr}\left(T(x) = c \mid T(x_1) = c\right) = K\frac{1}{1}$,

for the second point $i = 2$ $\qquad \mathbf{Pr}\left(T(x) = c \mid T(x_2) = c\right) = K\frac{1}{2}$,

and so on, generally for point No. i $\qquad \mathbf{Pr}\left(T(x) = c \mid T(x_i) = c\right) = K\frac{1}{i}$.


Red or green?
(light or dark?)
1
2
3
4
5
6
7
8
9
10
11
12
13
14
samples ordered according to the distance from the query point
reciprocals of order numbers for class of red (light) points
reciprocals of order numbers (without respect to class - color)
sum of rows
probability that the point belongs to the red (light) class
0.526967 = 1.713467 / 3.251562

1	2	3	4	5	6	7	8	9	10	11	12	13	14	S or S_c	p(r/g|x)
🟥	🟩	🟩	🟩	🟩	🟥	🟥	🟥	🟥	🟩	🟥	🟩	🟥	🟩		
1	0.5	0.33	0.25	0.20	0.17	0.14	0.13	0.11	0.10	0.09	0.08	0.08	0.07	3.251562	
1					0.17	0.14	0.13	0.11		0.09		0.08		1.713467	0.526967
	0.5	0.33	0.25	0.20					0.10		0.08		0.07	1.538095	0.473033

Figure 1. Illustration of classification procedure for the simplest case of two classes and of the same number of samples of both classes.

Individual points are independent and then we can sum up these probabilities. Thus, we add the partial influences of k individual points together by summing up

$$\hat{p}(c \mid x) = \sum_{x_i \in U_c} \mathbf{Pr}\left(T(x) = c \mid T(x_i) = c\right) = K \sum_{x_i \in U_c} 1/i\,. \tag{2}$$

Note that the sum goes over indexes i for which the corresponding samples of the learning set are of class c, $c = 1, 2, \dots C$, where C is the number of classes.

Let

$$S_c = \sum_{x_i \in U_c} 1/i\,.$$

Then there is

$$\sum_{c=1}^{C} S_c = H_N\,,$$

where H_N is the N-th harmonic number. The estimation of the probability that the given point x belongs to class c is

$$\hat{p}(x \mid c) = \frac{S_c}{H_N}.$$

This approach is based on the hypotheses that the weight of a neighbor is proportional just to the reciprocal of its order number as well as to its distance from the given point, see Theorem 1.

It can be seen that $K\frac{1}{i}$ is also the value (at point x) of the kernel function with its center at point x_i and having the form of a Zipfian probability mass function (1) for $s = 1$. Summing up these values over all centers that belong to class U_c gives (2).

3.2 The Classification Procedure

The probability estimation above can be used for classification. Let the samples of the learning set (i.e. all samples irrespective of the class) be sorted according to their distances from the given point x. Let indexes be assigned to these points so that 1 is assigned to the nearest neighbor, 2 to the second nearest neighbor of the given point x etc.

Let us compute sums $S_c(x) = \frac{1}{N_c} \sum_{x_i \in U_1} 1/i$, i.e. the sums of reciprocals of the indexes of samples from each class c separately; N_c is the number of samples of class c (cardinality of U_c) and $N_1 = N_2 = ... = N_C$.

The estimate of the probability that point x belongs to class c is

$$\hat{p}(c \mid x) = \frac{S_c}{\sum_{k=1}^{C} S_k}. \tag{3}$$

In the end, the formula above is nothing else than Bayes formula.

Usually we say that point x is of class k if $\hat{p}(k \mid x)$ is the largest of all $\hat{p}(c \mid x)$.

In the case of two classes, when some discriminant threshold θ is chosen then if $p(c = 1 \mid x) \geq \theta$ point x is of class 1, else it is of class 2. This is the same procedure as in other classification approaches where the output is estimation of probability (naïve Bayes) or any real valued variable (neural networks). The value of the threshold can be optimized with respect to the loss function.

For classification into more than two classes we use this formula for all classes and we assign to the given point x a class c for which $p(c \mid x)$ is the largest.

Formally we can rewrite (3) into more comprehensive form. For the two class problem with different number of samples of one and the other class formula we have

$$\hat{p}(c \mid x) = \frac{\frac{1}{N_c}\sum_{x_i \in U_c} 1/i}{\frac{1}{N_1}\sum_{x_i \in U_1} 1/i + \frac{1}{N_2}\sum_{x_i \in U_2} 1/i} .$$

It is seen here the introduction of the relative representation of different numbers of samples of one and the other class, i.e. introducing a priori probabilities.

For C classes there is

$$\hat{p}(c \mid x) = \frac{\frac{1}{N_c}\sum_{x_i \in U_c} 1/i}{\sum_{k=1}^{C}\frac{1}{N_k}\sum_{x_i \in U_k} 1/i} .$$

4. Approximation of The Probability of The Class

4.1 Zipfian Kernel Approach

Let indexes i be assigned to points (samples) of the learning set without respect to a class so that $i = 1$ is assigned to the nearest neighbor of point x, $i = 2$ to the second nearest neighbor etc. We have a finite learning set of size N samples and N_c samples of individuals from class c.

Using Zipfian probability mass function (1) as a kernel function, we have the kernel function in the form $K(\|x-x_i\|/h)=1/(i^s H_{N,s})$. At the same time, product Nh, i.e. the number of samples times the smoothing factor has no significance here and we set $Nh = 1$. Then we get an approximation of the probability that the given point x belongs to class c in the form

$$\hat{p}(c \mid x) = \frac{1}{H_{N,s}}\sum_{i \in U_c}\frac{1}{i^s}, \tag{4}$$

where the sum goes over indexes i for which the corresponding samples of the learning set are of class c. Summing up approximations of probability densities at point x for all classes, we get apparently $\frac{1}{H_{N,s}}\sum_{i=1}^{N}\frac{1}{i^s}$ that is equal to 1 and thus fulfills the assumption that the given point belongs to some class.

Two classes only and the same number of samples of both classes are assumed without loss of generality in the theorem and the proof is as follows.

Theorem 1. *Let the task of classification into two classes be given and let the size of the learning set be N and let both classes have the same number of samples, i.e. there is the same a priori probability. Let i be the index (rank) of the i-th nearest neighbor x_i of point x (without considering the neighbor's class) and r_i be its distance from the point x. Then*

$$\lim_{N\to\infty} \frac{\sum_{x_i \in U_c} 1/i}{H_N} = p(c \mid x) \ . \tag{5}$$

where p(c|x) is the probability that point x belongs to class c.

In the following proof we use some notions known from Jiřina and Jiřina, Jr. (2008; 2009; 2013), briefly summarized as follows.

4.1.1 Mapping the Distribution

Let us have an example of a ball in an *n*-dimensional space containing uniformly distributed points over its volume. Let us divide the ball on concentric "peels" of the same volume. Using the formula $r_i = \sqrt[n]{V_i / S(n)}$, which is, in fact, the inverted formula for volume V_i of *n*-dimensional ball of radius r_i, we obtain a quite interesting succession of radii corresponding to the individual volumes - peels. The symbol *S*(*n*) denotes the volume of a ball with unit radius in E_n; note $S(3) = 4/3\pi$. A mapping between the mean density ρ_i in an *i*-th peel and its radius r_i is $\rho_i = p(r_i)$; $p(r_i)$ is the mean probability density in the *i*-th ball peel with radius r_i. The probability distribution of points in the neighborhood of a given point *x* is thus simplified to a function $p(r_i)$ of a scalar variable r_i. We call this function a probability distribution mapping function *D*(*x*, *r*) and its partial differentiation with respect to *r* the distribution density mapping function *d*(*x*, *r*). Functions *D*(*x*, *r*) and *d*(*x*, *r*) for *x* fixed are, in fact, the probability distribution function and the probability density function of variable *r*, i.e. of distances of all points from the given point *x*. More exact definitions follow (Jiřina and Jiřina, Jr. 2008).

Definition 1. Probability distribution mapping function *D*(*x*, *r*) of the given point *x* is function $D(x,r) = \int_{B(x,r)} p(z)dz$, where *r* is distance from the given point and *B*(*x*, *r*) is ball with center *x* and radius *r*.

Definition 2. Distribution density mapping function $d(x, r)$ of the given 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 given point x and radius r.

Note. When it is necessary to differentiate the class of a point at distance r from point x, we write $D(x, r, c)$ or $d(x, r, c)$.

4.1.2 Correlation Dimension

It is seen that for fixed x the function $D(x, r)$, $r > 0$ is monotonously non-decreasing 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 function, respectively. In fact, $D(x, r)$ is the distribution function of distances of points from the given point x and $d(x, r)$ is corresponding probability density function. So we can write $p(c|x, r) = d(x, r, c)$. Moreover, $D(x, r)$ reminds the correlation integral (Grassberger and Procaccia 1983). The correlation integral

$$C_I(r) = \lim_{N\to\infty} \frac{1}{N^2} \sum_{i,j=1}^{N} h(r - |x_{.i} - x_{.j}|),$$

where $x_{.i}$ and $x_{.j}$ are points of the learning set without respect to class and $h(.)$ is a Heaviside's step function, can be written in the form (Camastra 2003; Camastra and Vinciarelli 2001).

$$C_I(r) = \lim_{N\to\infty} \frac{1}{N(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} h(r - |x_{.i} - x_{.j}|).$$

It can be seen that correlation integral is a distribution function of distances between pairs of data points given. The probability distribution mapping function is a distribution function of distances from one fixed point. In the case of finite number of points N, there are $N(N - 1)/2$ pairs of points and then distances between them, and from them one can construct empirical correlation integral. Similarly, for each point there are $N - 1$ distances and from these $N - 1$ distances one can construct empirical probability distribution mapping function. There are exactly N such functions and mean of these functions gives empirical correlation integral. This is valid also in the limit for the number N of points going to infinity.

On the other hand there are essential differences. The probability distribution mapping function is a local feature dependent on the position of point x. The empirical distribution mapping function also includes boundary effects (Arya, Mount, and Narayan 1996) of the true data set. The correlation integral is a feature of the fractal or data generating process and should not depend on the position of the particular point considered or on the size of the data set at hand.

In a log-log graph of the correlation integral, i.e. the graph of the dependence of C_I on r, the slope gives the correlation dimension ν. In the log-log graph of the probability distribution mapping function $D(x, r)$ the curve is also close to a monotonously and nearly linearly growing function. The slope (derivative) is given by a constant parameter. Let us denote this parameter q and call it the distribution mapping exponent. This parameter is rather close but generally different from ν.

The linear part of the log-log graph means

$$\mathbf{log}\, C_I(r) = a + \mathbf{log}\, \nu ,$$

where a is a constant, and then $C_I(r) = ar^{\nu}$. Thus, $C_I(r)$ grows linearly with variable $z = r^{\nu}$.

Similarly the probability distribution mapping function grows linearly with r^q at least in the neighborhood of point x. Its derivative, the distribution density mapping function, is constant there.

4.1.3 Proof of Theorem 1

There are c spatial distributions $p_c(x)$ of probability that any point x (on the support considered) is of class c. Then for each point x and C classes there is $\sum_{c=1}^{C} p_c(x) = 1$. For each given point x one can state the probability distribution mapping function $D(x, r_i, c)$. We approximate this function so that it holds (K is a constant)

$$D(x, r_i^q, c) = Kr_i^q$$

in the neighborhood of point x. Using differentiation, according to variable $z = r_i^q$, we get $d(x, r_i^q, c) = K$. It means that by the use of $z = r_i^q$, the space is mapped ("distorted") so that the distribution density mapping function is constant in the neighborhood of point x for a particular distribution. Let us consider sum $\sum_{i=1}^{N} d(x, r_i^q, c) / r_i^q$. For this sum we have

$$\lim_{N \to \infty} \sum_{i=1}^{N} d(x, r_i^q, c) / r_i^q = p(c \mid x) \lim_{N \to \infty} \sum_{i=1}^{N} 1 / r_i^q \qquad (6)$$

because $d(x, r_i^q, c) = d(x, z, c) = p(c \mid x)$ for all i (the uniform distribution has a constant density).

By the use of $z_i = r_i^q$, the space is nonlinearly rescaled so that the distribution density mapping function $d(x, z_i, c)$ is constant in the neighborhood of point x. Then r_i^q is proportional to i, $r_i^q = k_1 i$; k_1 is a constant. Exponent q need not be a constant but can be a function $q = q(x, c)$; we write it for point x_i in form $q = q(i, c)$. Let $r_i^{q(i,c)} = k_1 i$ for all i of class c. (From the last formula one could derive the $q(i, c)$, but we need not do it.) We rewrite the equation (6) in the form

$$\lim_{N\to\infty} \sum_{i=1}^{N} d(x, r_i^{q(i,c)}, c) / r_i^{q(i,c)} = p(c \mid x) \lim_{N\to\infty} \sum_{i=1}^{N} 1 / r_i^{q(i,c)}$$

and then in the form

$$\lim_{N\to\infty} \sum_{i=1}^{N} d(x, r_i^{q}, c) / i = p(c \mid x) \lim_{N\to\infty} \sum_{i=1}^{N} 1 / i = p(c \mid x) \lim_{N\to\infty} H_N .$$

Given the learning set, we have the space around point x "sampled" by individual points of the learning set. Let $p_c(r_i)$ be an a-posteriori probability that point i in distance r_i from the given point x is of the class c. Then $p_c(r_i)$ is equal to 1 if point i is of class c and $p_c(r_i)$ is equal to zero, if the point is of the other class. Then the particular realization of $p(c \mid x) H_N$ is sum $\sum_{x_i \in U_c} 1/i$. Using this sum, we can write

$$p(c \mid x) \lim_{N\to\infty} H_N = \lim_{N\to\infty} \sum_{x_i \in U_c} 1/i .$$

Dividing this equation by the limit of sum on the left hand side, we get

$$\frac{\lim_{N\to\infty} \sum_{x_i \in U_c} 1/i}{\lim_{N\to\infty} H_N} = p(c|x)$$

and due to the same limit transition in the numerator and in the denominator we can rewrite it in the form (5).

4.2 Bayes Risk

We have shown that estimate (4) converges to true probability $p(c|x)$. Considering two-class classification with simple loss matrix $L(1, 1) = L(2, 2) = 0$, $L(1, 2) = L(2,1) = 1$ there is conditional Bayes risk of estimating a class of point x

$$R(x) = R(c \mid x) + R(not\ c \mid x) = 2(1 - p(c \mid x)) .$$

It is apparent that its estimate $\hat{R}(x) = 2(1 - \hat{p}(c \mid x))$ converges to $R(x)$ as $\hat{p}(c \mid x)$ converges to $p(c \mid x)$. The $\hat{R}(x)$ can be computed easily having classification error that is equal to $1 - \hat{p}(c \mid x)$ and can be found for example in Table 1, see Section 5.

4.3 Computational Complexity

For total N samples and single given point x the procedure consists of three steps:

- Computation of distances; the computational complexity for one distance is proportional to dimensionality n, of all N distances nN.
- Sorting distances is proportional to $N\log N$.
- Summing up of reciprocals of indexes is proportional to N.

Then the total complexity is $anN + bN\log N + dN = N(an + b\log N + d\ )$, where a, b, d are implementation dependent constants. For larger learning data sets the complexity is governed by sorting. It is also seen that the computational complexity directly depends on the learning set size N and in small extend on dimensionality n.

5. Experiments

5.1 Implementation Details

5.1.1 System and Language Environment

The algorithm was implemented in c++ as a function to separate it from necessary reading and writing data and setting up control parameters. It runs under Windows as well as under Linux environment; the compilers used were Microsoft Visual C++ and c++ of GCC 4.8 Release Series by GNU Project - Free Software Foundation (FSF), respectively.

5.1.2 Input Parameters

To run, the program needs names of arrays for learning and testing data, problem dimensionality n and length (the number of samples) of these arrays. There is also name for array with output values, i.e. estimated class probabilities for each sample of testing data.

5.1.3 Control

There is a single control parameter - user can change metrics L_p by setting up p – usually $p = 1$ or $p = 2$ but any positive real value can be used. Of course, L_p for p less than one is not a metrics. Using large p (say $p = 10$) one gets close to $L_{\inf}$ metrics; the ranking of neighbors is then given by the largest coordinate with very rare exceptions. Distance r_i should be computed using the formula

$$r_i = \sqrt[p]{\sum_{j=1}^{n} \left| x_{ij} - x_j \right|^p} ,$$

but we do not use the p-th root; in sorting values of r_i^p suffice for comparisons. We use standard library quicksort function.

5.1.4 Normalization

For all the data sets, each of all the input features is normalized to values with zero mean and unit empirical variance.

5.2 Tasks from UCI Machine Learning Repository – Comprehensive Tests

Data sets ready for a run with a classifier are available on the net (Lucas and Algoval 2008). For small data sets in this corpus each task consists of 50 pairs of training and testing sets corresponding to 50-fold cross validation. For large data sets, i.e. DNA data (Paredes 2008), Letter data (Letter recognition), and Satimage (Statlog Landsat Satellite) the single partition into training and testing sets according to specification in Bache and Lichman (2013) was used. We also added the popular Iris data set with ten-fold cross validation.

In Table 1 the results obtained by different methods are summarized. The methods are as follows:

L2	The nearest neighbor method by Paredes and Vidal (2006)
1-NN L2	The nearest neighbor method computed by authors

sqrt-NN L2	The *k*-NN method with *k* equal to square root of the number of samples of the learning set computed by authors
Bayes 10	The Bayes naive method with ten bins histograms, computed by authors
CDM	The learning weighted metrics method with class dependent Mahalanobis by Paredes and Vidal (2006)
CW	The learning weighted metrics method with class dependent weighting by Paredes and Vidal (2006)
PW	The learning weighted metrics method with prototype dependent weighting by Parades and Vidal (2006)
CPW	The learning weighted metrics method with class and prototype - dependent weighting by Parades and Vidal (2006)
IINC L1	The method presented here with Manhattan L_1 metrics
IINC L2	The method presented here with Euclidean L_2 metrics

In Table 1 in each row the best result is denoted by bold numerals. Furthermore, in the last column, the values for IINC better with L_2 metrics than with L_1 metrics are shown in italics. There are five such cases out of a total of 24.

5.3 Standalone Real-life Comprehensive Classification Task

This data set was available for tests described in Hakl, Jirina, and Richter-Was (2005) as one of many simulation studies for data processing relating ATLAS experiment at CERN, Geneva, Switzerland. For the description of the particle physics problem we cite Hakl, Jirina, and Richter-Was (2005) verbatim in Table 2.

The data set consists of 7 dimensional vectors of real numbers and class mark, which differentiates between signal samples (events) and background samples. The data set is split into learning and testing set, each of 3279 samples.

In Figure 2 well-known ROC curves are shown for different separation/classification tools including the “cut” method popular in physics studies.

The result obtained with “cuts” method is depicted by the black diamond.

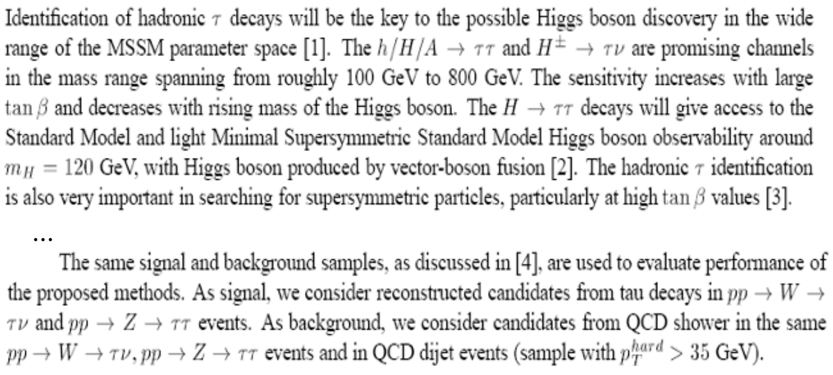
The result obtained by GMDH-MIA algorithm is depicted by the lower bold black line

The results obtained by STATISTICA Neural Networks are depicted by two sets of red, magenta, orange and yellow lines. Each set corres-

Table 1. Classification error rates for different datasets and different approaches. Empty cells denote not available data. For legend see 5.2.

Dataset	L2	1-NN L2	sqrt-NN L2	Bayes 10	SVM	CDM	CW	PW	CPW	**IINC L1**	**IINC L2**
Australian	34.37	20.73	15.50	13.88	35.99	18.19	17.37	16.95	16.83	**13.31**	14.75
Balance	25.26	23.61	32.06	15.17	45.48	35.15	17.98	**13.44**	17.6	32.58	*30.80*
Cancer	4.75	5.07	3.25	**2.68**	16.34	8.76	3.69	3.32	3.53	3.28	3.48
Diabetes	32.25	29.48	26.46	**24.19**	29.64	32.47	30.23	27.39	27.33	26.21	*25.52*
DNA	23.4	25.72	34.06	6.66		15	4.72	6.49	**4.21**	27.82	31.03
German	33.85	32.76	30.90	**24.97**	27.25	32.15	27.99	28.32	27.29	30.91	31.13
Glass	**27.23**	32.72	42.10	47.37		32.9	28.52	26.28	27.48	33.01	35.18
Heart	42.18	25.11	16.89	17.44	38.89	22.55	22.34	18.94	19.82	17.96	***17.93***
Ionosphere	19.03	14.05	14.70	**9.26**						10.82	14.81
Iris	6.91	5.91	7.91	9.82	6.55					7.91	***4.91***
Led17	20.5	11.50	**0.12**	0.00						0.46	***0.45***
Letter	4.35	4.80	18.70	28.98	40.53	6.3	3.15	4.6	**4.2**	4.85	4.98
Liver	37.7	39.59	41.48	39.42	37.68	39.32	40.22	**36.22**	36.95	38.29	39.13
Monkey1	2.01	2.01	9.27	28.01	23.54					4.79	4.79
Phoneme	18.01	11.83	20.71	21.47	21.71					17.55	18.06
Satimage	10.6	10.65	15.20	19.15	44.85	14.7	11.7	**8.8**	9.05	11.00	11.55
Segmen	11.81	3.81	11.41	9.85						**4.12**	5.05
Sonar	31.4	18.37	32.51	31.46						**19.89**	22.85
Vehicle	35.52	30.51	31.51	38.40		32.11	29.38	**29.31**	28.09	29.40	*29.34*
Vote	8.79	8.74	9.60	9.70		6.97	6.61	5.51	**5.26**	8.52	8.89
Vowel	1.52	**1.19**	46.68	26.64		1.67	1.36	1.68	1.24	2.73	2.74
Waveform 21	24.1	23.73	**14.71**	19.26						16.15	16.38
Waveform 40	31.66	28.22	**16.24**	20.31						17.59	18.08
Wine	24.14	5.42	6.15	4.50		2.6	1.44	1.35	**1.24**	4.24	5.66

Table 2. Problem formulation from the point of view of physics.



Identification of hadronic τ decays will be the key to the possible Higgs boson discovery in the wide range of the MSSM parameter space [1]. The $h/H/A \rightarrow \tau\tau$ and $H^{\pm} \rightarrow \tau\nu$ are promising channels in the mass range spanning from roughly 100 GeV to 800 GeV. The sensitivity increases with large $\tan\beta$ and decreases with rising mass of the Higgs boson. The $H \rightarrow \tau\tau$ decays will give access to the Standard Model and light Minimal Supersymmetric Standard Model Higgs boson observability around $m_H = 120$ GeV, with Higgs boson produced by vector-boson fusion [2]. The hadronic τ identification is also very important in searching for supersymmetric particles, particularly at high $\tan\beta$ values [3].

…

The same signal and background samples, as discussed in [4], are used to evaluate performance of the proposed methods. As signal, we consider reconstructed candidates from tau decays in $pp \rightarrow W \rightarrow \tau\nu$ and $pp \rightarrow Z \rightarrow \tau\tau$ events. As background, we consider candidates from QCD shower in the same $pp \rightarrow W \rightarrow \tau\nu, pp \rightarrow Z \rightarrow \tau\tau$ events and in QCD dijet events (sample with $p_T^{hard} > 35$ GeV).

Note that references relate to Hakl, Jirina, and Richter-Was (2005).

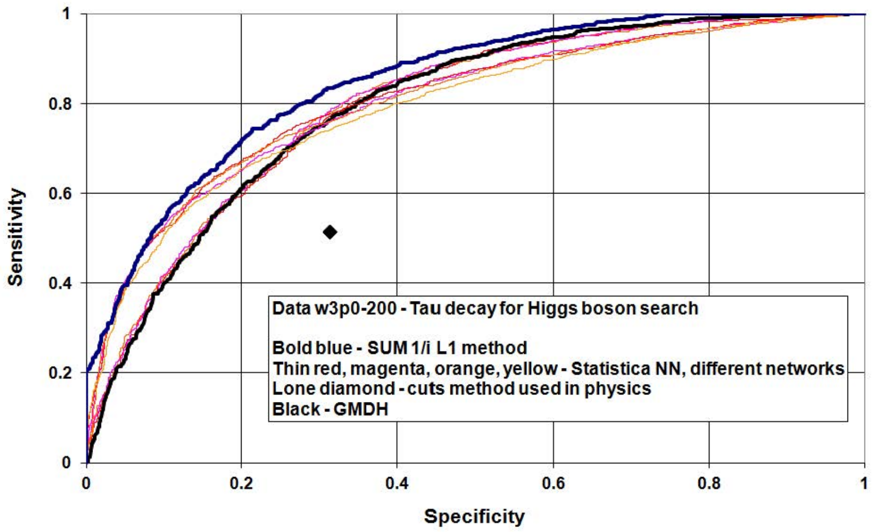


Figure 2. ROC curves for different separation/classification tools including the "cut" method.

ponds to four best results out of ten networks generated. The set going more to the left at level 0.4 or 0.6 of sensitivity (signal acceptance) corresponds to its being set as a classifier; the other set (closer to the black line of GMDH-MIA) corresponds to its being set as an approximator.

The upper bold blue line was obtained by the IINC method described in this paper with L_1 metrics.

6. Discussion

We have proved that probability density approximation can be based on the Zipfian kernel. In the proof we have shown a close relation of the Zipfian distribution (and of the selected harmonic series as well) to the local fractal nature of data. It is especially the use of $1/i$ that has a close connection to the scaling exponent, eventually to the correlation integral, and thus to the dynamics of the processes that generate data we wish to separate. We have shown, in fact, that the influence to the probability that point x is of class c is $1/i$ if the i-th nearest neighbor is of class c. We sum up these influences so that the sum goes over the indexes i for which the corresponding samples of the learning set are of class c. In the case of two classes we get two numbers S_1 and S_2 which together give the sum of N first terms of harmonic series $H_N = 1 + ½ + ⅓+ ¼ + \ldots + 1/N$. ($N$ is the size of the learning set.)

An interesting finding is that the method proposed here and the proof of the theorem uses the notion of distance but no explicit metrics need to be specified.

The method designed has no parameters to be tuned. There is also no problem with the convergence and the curse of dimensionality. The computational complexity grows at most linearly with the dimensionality and quadratically or less with the learning set size depending on the sorting algorithm used.

7. Conclusions

The main merit of the new method presented here is a new view on data space. This view is based on a strange geometry with polynomially expanded distances in dependence on the local dimensionality of data denoted as the distribution mapping exponent. This leads to the use of reciprocals of the neighbor indexes and finally to the probability density estimation. The reciprocals of the neighbor indexes can be understood as "weights" of the learning set samples. It means, in fact, that the probability that the i-th neighbor and the given point are of the same class is given by the Zipfian distribution. In this context, the Zipfian distribution gets a much broader role than its use in linguistics and psychology.

The other question is how the method presented here can be further improved. We suspect e.g. that data of one and the other class can be similarly distributed in the space even if data have different intrinsic dimensionality. Data often lie in clusters which is a fact not tackled here. For given points outside these clusters or on boundaries of clusters the sum of reciprocals of the neighbor indexes of the opposite class may prevail, thus causing misclassification. This is a theme for further research in this field.

References

ARYA, S., MOUNT, D.M., NARAYAN, O. (1996), "Accounting for Boundary Effects in Nearest Neighbor Searching", *Discrete and Computational Geometry, 16,* 155–176.

BACHE, K., and LICHMAN, M. (2013), "UCI Machine Learning Repository", University of California, School of Information and Computer Science, Irvine CA, http://archive.ics.uci.edu/ml

BORS, A.G., and NASIOS, N. (2013), "Kernel Bandwidth Estimation for Nonparametric Data Segmentation", go to: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.159.7219.

CAMASTRA, F. (2003), "Data Dimensionality Estimation Methods: A Survey", *Pattern Recognition*, *36*, 2945–2954.

CAMASTRA, P., and VINCIARELLI, A. (2001), "Intrinstic Dimension Estimation of Data: An Approach Based on Grassberger-Procaccia's Algorithm", *Neural Processing Letters, 14(1),* 27–34.

COVER, T.M., and HART, P.E. (1967), "Nearest Neighbor Pattern Classification", *IEEE Transactions in Information Theory, IT-13(1),* 23–27.

GRASSBERGER, P., and PROCACCIA, I. (1983), "Measuring the Strangeness of Strange Attractors", *Physica*, *9D*, 189–208.

HAKL, F., JIŘINA, M., and RICHTER-WAS, E. (2005), "Hadronic Tau's Identification Using Artificial Neural Network", ATLAS Physics Communication, ATL-COM-PHYS-2005-044, CERN, Geneve, http://www.marceljirina.cz/files/hadronic-tau-s-identification-using-artificial-neural-network.pdf

HERBRICH, R. (2002), *Learning Kernel Classifiers. Theory and Algorithms*, Cambridge MA; London UK: The MIT Press.

HERBRICH, R. (2001), *Learning Kernel Classifiers. Theory and Algorithms*, Cambridge MA: The MIT Press.

JIŘINA, M., and JIŘINA, JR., M. (2008), "Apparatus for Assessing a Control Value", patent pending under number PV 2008-245; Z 7576, submitted on 22 April 2008 to the Industrial Property Office, Prague, Czech Republic.

JIŘINA, M., and JIŘINA JR., M. (2008), "Correlation Integral Decomposition for Classification. in Artificial Neural Networks", *Lecture Notes in Computer Science Vol. 5164*, Berlin: Springer, pp 62–71.

JIŘINA, M., and JIŘINA JR., M. (2009), "Classification by the Use of Decomposition of Correlation Integral" in Studies in Computational Intelligence Vol 205, *Foundations of Computational Intelligence Vol 5*, Berlin: Springer, pp. 39–55

JIŘINA, M., and JIŘINA JR., M. (2013), "Utilization of Singularity Exponent in Nearest Neighbor Based Classifier", *Journal of Classification, 30(1)*, 3–29.

LUCAS, S.M. (2008), "Algoval: Algorithm Evaluation over the Web", http://algoval.essex.ac.uk.

MANDELBROT, B.B. (1982), *The Fractal Theory of Nature*, New York: W. H. Freeman and Co.

MASLOV, V.P. (2005), "On a General Theorem of Set Theory Leading to the Gibbs, Bose-Einstein, and Pareto Distributions as Well as to the Zipf-Mandelbrot Law for the Stock Market", *Mathematical Notes, 78(6),* 807–813.

PAREDES, R. (2008), "CPW: Class and Prototype Weights Learning", http://www.dsic.upv.es/~rparedes/research/CPW/index.html.

PAREDES, R., and VIDAL, E. (2006), "Learning Weighted Metrics to Minimize Nearest Neighbor Classification Error*", IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(7)*, 1100–1110.

SCHMULAND, B (2003), "Random Harmonic Series", *American Mathematical Monthly 110*, 407–416.

SCHÖLKOPF, B., and SMOLA, A.J. (2002*), Learning with Kernels. Support Vector Machines, Regularization, Optimization, and Beyond*, Cambridge MA; London UK: The MIT Press.

SCOTT, D.W. (1992), *Multivariate Density Estimation. Theory, Practice, and Visualization,* New York: John Wiley and Sons.

SILVERMAN, B.W. (1992), *Density Estimation for Statistics and Data Analysis*, CRC Monographs on Statistics and Applied Probability, London: Chapman and Hall.

ZIPF, G.K. (1968), *The Psycho-Biology of Language. An Introduction to Dynamic Philology,* Cambridge: The MIT Press.

ZUO, W., WANG, K., ZHANG, H., and ZHANG, D. (2007), "Kernel Difference-Weighted *k*-Nearest Neighbors Classification", *Lecture Notes in Computer Science Vol. 4682*, Berlin; Heidelberg: Springer-Verlag, pp. 861–870.