Decomposition of Correlation Integral to Local Functions

Jiřina, Marcel
2008

cAk

CENTRUM APLIKOVANÉ KYBERNETIKY

České vysoké učení technické v Praze - fakulta elektrotechnická

Decomposition of Correlation Integral to Local Functions

Technical report

Marcel Jiřina and Marcel Jiřina, jr.

www@c-a-k.cz

2008

Institute of Computer Science
Academy of Sciences of the Czech Republic

Decomposition of Correlation Integral to Local Functions

Marcel Jiřina and Marcel Jiřina, jr.

Technical Report No. V-1025

June 2008

Abstract

We show that the correlation integral can be decomposed into functions each related to a particular point of data space. For these functions, one can use similar polynomial approximations such as the correlation integral. The essential difference is that the value of the exponent, which would correspond to the correlation dimension, differs in accordance to the position of the point in question. Moreover, we show that the multiplicative constant in that polynomial approximation is proportional to the probability density estimation at that point. This finding is used to construct a classifier.

Keywords:

multivariate data, correlation dimension, correlation integral, decomposition, probability density estimation

Decomposition of Correlation Integral to Local Functions

Marcel Jirina[1] and Marcel Jirina, Jr.[2]

[1] Institute of Computer Science AS CR, v.v.i., Pod vodárenskou věži 2, 182 07 Prague 8 – Libeň, Czech Republic, marcel@cs.cas.cz

[2] Faculty of Biomedical Engineering, Czech Technical University in Prague, Nám. Sítná 3105, 272 01, Kladno, Czech Republic, jirina@fbmi.cvut.cz

Contents

1. Introduction

A lot of tasks of data mining have to do with associating objects to a limited number of types or classes. A typical task is whether an e-mail is spam or not. This is a classification into two classes. Many other tasks may be recognized as classification into several classes. Usually, objects to be classified are not used directly, but are described by some number of parameters (or features, variables etc.) There are many approaches to classification, simple ones or very sophisticated ones. In this chapter, an approach closely related to the characterization of fractals by the correlation dimension is introduced.

The correlation dimension [1], [2] as well as other effective dimensions - Hausdorff, box-counting, information dimension [3], [4] - is used to study features of different fractals and data generating processes. For estimation of the value of the correlation dimension in a particular case, linear regression is often used for logarithms of variables [1]. We write it in the form:

$$\ln(s) = \ln(C) + q \ln(r_s), \quad s = 1, 2,.... \tag{1}$$

Here, v is a correlation dimension and C is a multiplicative constant in the relation:

$$s = C r_s^q, \; s = 1, 2, ... \tag{2}$$

Constant C has no particular meaning.

In this Chapter, we show that the correlation integral can be decomposed in functions each related to particular point x of data space. For these functions one can use similar polynomial approximations as given by (2). The value of exponent q, which corresponds to the correlation dimension, differs in accordance to the position of the point x in question. Moreover, we show that the multiplicative constant C in these cases represents the probability density estimation at point x. This finding is used to construct a classifier. Tests with some data sets from the Machine Learning Repository [5] show that this classifier can have a very low classification error.

2. Decomposition of the Correlation Integral

We work in n-dimensional metric space with L_2 (Euclidean) or L_1 (taxicab or Manhattan) metrics.

2.1 Correlation Integral

The correlation integral, in fact, a distribution function of all binate distances in a set of points in a space with a distance was introduced by Grassberger and Procaccia in 1983 [1]. Camastra and Vinciarelli [6] consider the set $\{X_i, i = 1, 2, .. N\}$ of points of the attractor. This set of points may be obtained e.g. from a time series with a fixed time increment. Most pairs (X_i, X_j) with $i \neq j$ are dynamically uncorrelated pairs of essentially random points [1]. However, the points lie on the attractor. Therefore, they will be spatially correlated. This spatial correlation is measured by the correlation integral $C_I(r)$ defined according to:

$$C_I(r) = \lim_{N\to\infty} \frac{1}{N^2} \times \{number of \; pairs(i,j) : \|X_i - X_j\| < r\}.$$

In a more comprehensive form one can write:

$$C_I(r) = \Pr(\|X_i - X_j\| < r) .$$

Grassberger and Procaccia [1] have shown that for a small r the $C_I(r)$ grows like a power $C_I(r) \sim r^{\nu}$ and that the "correlation exponent" ν can be taken as a most useful measure of the local structure of the strange attractor. This measure allows one to distinguish between deterministic chaos and random noise [6]. These authors also mention that the correlation exponent (dimension) ν seems to be more relevant in this respect than the Hausdorff dimension D_h of the attractor. In general, there is $\nu \le \sigma \le D_h$, where σ is the information dimension [4], and it can be found that these inequalities are rather tight in most cases, but not all cases. Given an experimental signal and $\nu < n$ (n is the degree of freedom or the dimensionality or the so-called embedding dimension), then we can conclude that the signal originates from deterministic chaos rather than random noise, since random noise will always result in $C_I(r) \sim r^n$.

The correlation integral can be rewritten in form [6]

$$C_I(r) = \lim_{N\to\infty} \frac{1}{N(N-1)} \sum_{1\le i<j\le N} h(r - \|X_j - X_i\|) ,$$

where $h(.)$ is Heaviside step function. From it

$$\nu = \lim_{r\to\infty} \frac{\ln C_I(r)}{\ln r} .$$

There are methods for estimating the correlation dimension ν, but the problem is that they are either too specialized for one kind of equation or they use some kind of heuristics that usually optimize the size of radius r to get the proper value of the correlation dimension. One of the most cited is Taken's estimator [7], [8], [9].

2.2 Probability distribution mapping function

Two important notions, the probability distribution mapping function and the distribution density mapping function are introduced here. We use these notions for developing a decomposition of the correlation integral and a new classifier. To understand these terms, we give a brief example that demonstrates them.

Let a query point x be placed without loss of generality in the origin. Let us build balls with their centers at point x and with volumes V_i , i =1, 2, ...

The individual balls are in one another, the (i-1)-st inside the i-th are like peels of an onion. Then the mean density of the points in the i-th ball is $\rho_i = m_i/V_i$. The volume of the 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 the i-th ball ρ_i and its radius r_i. Then $\rho_i = f(r_i)$. Using a tight analogy between the density $\rho(z)$ and the 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. This way, a complex picture of the probability distribution of the points in the neighborhood of a query point x is simplified to a function of a scalar variable. We call this function the probability distribution mapping function $D(x, r)$, where x is a query point, and r the distance from it. More exact definitions follow:

Definition 1

The probability distribution mapping function $D(x, r)$ of the neighborhood of the query point x is the 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 a ball with center x and radius r.

Definition 2

The 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 can be seen that for a 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 a fixed x are one-dimensional analogs to the probability distribution function and the probability density function, respectively.

One can write the probability distribution mapping function in the form

$$D(x,r) = \lim_{N\to\infty} \frac{1}{N-1} \sum_{j=1}^{N-1} h(r - r_j), \tag{3}$$

where $h(.)$ is the Heaviside step function. For a finite number of points, we have the empirical probability distribution mapping function

$$D'(x,r) = \frac{1}{N-1} \sum_{j=1}^{N-1} h(r - r_j).$$

2.3 Power approximation of the probability distribution mapping function

Let us introduce a simple polynomial function in the form $D(x, r) = Cr^q$. We shall call it a power approximation of the probability distribution mapping function $D(x, r)$. Exponent q is a distribution-mapping exponent.

Definition 3

The power approximation of the probability distribution mapping function $D(x, r^q)$ is the function r^q such that $\frac{D(x,r^q)}{r^q} \to C$ for $r \to 0+$. The exponent q is a distribution-mapping exponent.

Using this approximation of the probability distribution mapping function $D(x, r)$, we, in fact, linearize this function as a function of the variable $z = r^q$ in the neighborhood of the origin, i.e. in the neighborhood of the query point. The distribution density mapping function $d(x, r)$, as a function of the variable $z = r^q$, is approximately constant in the vicinity of the query point. This constant includes a true distribution of the probability density of the points as well as the influence of boundary effects.

An important fact is that the distribution-mapping exponent reminds us of the correlation dimension by Grassberger and Procaccia [1]. Although, there are three essential differences: First, the distribution-mapping exponent is a local feature of the data set because it depends on a position of the query point, whereas the correlation dimension is a feature of the whole data space. Second, the distribution mapping exponent is related to the data only and not to a fractal or data generating process by which we can have an unlimited number of data points.

Third, the distribution mapping exponent is influenced by boundary effects, which have a larger influence with a larger dimension n and a smaller learning set size [6], [10].

2.4 Decomposition of correlation integral to local functions

We show, in this section, that the correlation integral is the mean of the distribution mapping function and that the correlation dimension can be approximated by the mean of the distribution mapping exponent as shown in the theorem below:

Theorem 1

Let there be a learning set of N points (samples). Let the correlation integral, i.e. the probability distribution of binate distances of the points from the learning set, be $C_I(r)$ and let $D(x_i, r)$ be the distribution mapping function corresponding to point x_i. Then, $C_I(r)$ is a mean value of $D(x_i, r)$:

$$C_I(r) = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} D(x_i, r) \ . \tag{4}$$

Proof

Let $h(x)$ be a Heaviside step function and l_{ik} be the distance of k-th neighbor from point x_i. Then the correlation integral is

$$C_I(r) = \lim_{N\to\infty} \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j=1}^{N-1} h(r - l_{ij})$$

and also

$$C_I(r) = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{N-1} \sum_{j=1}^{N-1} h(r - l_{ij}) \right) \ . \tag{5}$$

Comparing (5) with (3) we get (4) directly.

The correlation dimension ν can be approximated as a mean of the distribution mapping exponents

$$\nu = \frac{1}{N} \sum_{i=1}^{N} q_i \ .$$

2.5 Distribution mapping exponent estimation

Let U be a learning set composed of points (patterns, samples) $x_{cs,}$ where $c = \{0, 1\}$ is the class mark and $s = 1, 2, \ldots, N_c$ is the index of the point within class c. N_c is the number of points in class c and let $N = N_0 + N_1$ be the learning set size.

Let point $x \notin U$ be given and let points x_{cs} of one class be sorted so that index $s = 1$ corresponds to the nearest neighbor, index $s = 2$ to the second nearest neighbor, etc. In Euclidean metrics, $r_s = \|x - x_{cs}\|$ is the distance of the s-th nearest neighbor of class c from point x.

We look for exponent q so, that r_s^q is proportional to index s, i.e. for polynomial approximation

$$s = Cr_s^q \text{ , } s = 1, 2, ..., N_c, \; c = 0 \text{ or } 1, \tag{6}$$

where C is a suitable constant. Using a logarithm we get

$$\ln(s) = \ln(C) + q \ln(r_s) \text{ , } s = 1, 2, ..., N_c \text{ .} \tag{7}$$

On one hand, we exaggerate distances nonlinearly to make small differences in the distance appear much larger for the purposes of density estimation. On the other hand, there is a logarithm of distance in (7), which decreases large influences of small noise perturbations on the final value of q. Note that it is the same problem as in the correlation dimension estimation where equations of the same form as (6) and (7) arise. Grassberger and Procaccia [1] proposed a solution by linear regression. In [2], [9], [11] different modifications and heuristics were later proposed. Many of these approaches and heuristics can be used for distribution mapping exponent estimation, e.g. use a half or a square root of N_c nearest neighbors instead of N_c to eliminate the influence of the limited number of the points of the learning set.

The system of N_c (or $N_c/2$ or $\sqrt{N_c}$ as mentioned above) equation (7) with respect to an unknown q can be solved using standard linear regression for both classes. Thus, for two classes, we get two values of q, q_0 and q_1 and two values of C', C'_0 and C'_1.

At this point we can say that q_c is something like a local effective dimensionality of the data space including the true distribution of the points of each class. At the same time, we get the constant C'_c. The values of q_c and C'_c are related to each particular point x and thus they vary from one point x to another.

2.6 Probability density estimation

Let $n'_c(r)$ be a number of points of class c up to distance r from the query point x. Let q_c be the distribution mapping exponent for the points of class c and let

$$z_c = r^{q_c}. \tag{8}$$

Also, let $n_c(z_c) = n'_c(r) = n'_c(z_c^{1/q_c})$. Then $P_c(z_c) = n_c(z_c)/N$ is a percentage of all points of class c up to distance $r = z_c^{1/q_c}$ from the query point x., i.e. up to a "distance" measured by z_c from point x.

Due to polynomial approximation (6), $n_c(z_c) = C'_c.z_c$. It is a number of points up to distance r, which is related to z_c according to (8). The derivative according to z_c is $dn_c(z_c)/dz_c = C'_c$ and it represents a number of points of class c on a unit[1] of the z_c, i.e., in fact, a density of points with respect to z_c.

By dividing with total number of points N, we get a percentage of points of class c on a unit of z_c. This percentage is equal to $p(c|x, z_c) = C'_c/N$. In the limit case for $r \to 0$ (and z_c as well) there is $p(c|x, 0) = p(c|x) = C'_c/N = C_c$.

Finally, as there are two classes, there must be $p(0|x) = p(1|x) = 1$ and then $C'_0 + C'_1 = N$. This result includes a priori probabilities N_c/N for both classes. When we need to exclude a priori probabilities we use the formula

$$p(c \mid x) = \frac{C'_c / N_c}{C'_0 / N_0 + C'_1 / N_1} \text{ .} \tag{9}$$

[1] We cannot say "unit length" here, as the dimensionality of z_c is $(length)^{q_c}$.

The generalization of the too many classes case is straightforward. For k classes there is

$$p(c \mid x) = \frac{C'_c / N_c}{\sum_{i=1}^{k} C'_i / N_i}, \; c = 1, 2, .. \; k. \tag{10}$$

A more exact development follows:

Definition 4

Let **N** be n-dimensional space with metrics ρ. Let there be a subset $\mathbf{Q} \subseteq \mathbf{N}$ and a number $q \in \boldsymbol{R}+$, $1 \leq q \leq n$ associated with subset **Q**. A q-dimensional ball with center at point $x \in \mathbf{Q}$ and radius r is $B_q = B(q, x, r, \rho) = \{y \in \mathbf{Q}: \rho(x, y) < r\}$. The volume of B_q is $V(q, x, r, \rho) = S(q, \rho).r^q$, where $S(q, \rho)$ is a function independent of r.

Note: The metrics ρ can be omitted when it is clear what metrics we are dealing with.

Lemma 1

Let $B(q, x, R)$ be a q-dimensional ball with center at point $x \in \mathbf{Q}$ and radius R**,** and let $V(q, x, r)$ be its volume. Let points in **Q** in the neighborhood of point x up to distance R be distributed with the constant probability density $p = p_0$. Then, for $r < R$, where r is the distance from point x, the distribution function is given by

$$P(x,r) = \int_{B(q,x,R)} p dr = \int p dV(q,x,r) = p_0 V(q,x,r) \; .$$

The *proof* is obvious.

Conversely, let in **Q** hold $P(x,r) = p_0.V(q,x,r)$, where p_0 is a constant as long as $r < R$. It is obvious that this can be fulfilled even when the distribution density is not a constant. On the other hand, it is probably a rare case. Then we can formulate an assumption.

Assumption 1

If in **Q** holds $P(x,r) = p_0 V(q,x,r)$ then it holds $p(x) = p_0$.

Illustration

A sheet of white paper represents 2 dimensional subspace embedded in 3 dimensional space. Let point x be in the center of the sheet. White points of paper are uniformly distributed over the sheet with some constant (probability) density and a distribution function (frequentistically the number of white points) is proportional to the circular area around point x. Thus, the distribution function grows quadratically with distance r from point x, and only linearly with the size of the circular area. And the size of circular area is nothing other than the volume of the two-dimensional ball embedded in 3 dimensional space.

Theorem 2

Let, in a metric space, each point belongs to one of two classes $c = \{0, 1\}$. Let, for each point x and each class c, a distribution mapping function $D(x, c, r)$ exist where r is the distance from point x. Let Assumption 1 hold and the power approximation of the distribution mapping function be $C_c r^{q_c}$, where q_c is the distribution mapping exponent for point x and class c. Then it holds $p(c \mid x) = C_c S(q) = p_0$.

Proof

Let $z_c = r^{q_c}$ be a new variable. We can rewrite $D(x, c, r)$ as a function of variable z_c in the form $D(x, c, z_c)$. The $D(x, c, z_c)$ is, in fact, a distribution function of the points of class c with respect to variable z_c. When using a power approximation, we can write $D(x, c, z_c) = C_c r^q = C_c z_c$. This distribution function corresponds to uniform distribution in a subspace of dimension q_c. We express r^q with the help of the volume of the ball in q_c-dimensional space with center x and radius r: $D(x, c, z_c) = C_c V(q_c, x, r)/S(q_c) = P(x, r)/S(q_c)$. From Assumption 1, it follows $d(x, c, z_c) = C_c = p(x, r)/S(q_c)$ and then $p(x, r) = C_c S(q_c) = p_0$.

Note: We see that beyond the unit ball volume $S(q_c)$, the proportionality constant C_c governs the probability density in the neighborhood of point x including this point. Also note that due to the ratios in formulas (9) and (10) the volume $S(q_c)$ of the ball in a q_c-dimensional space in the probability estimation is eliminated.

3. Classifier

3.1 Classifier Construction

In this section, we show how to construct a classifier that incorporates the idea above. Using formulas (9) or (10) we have a relatively simple method for estimating the probabilities $p(c|x)$. First, we sort the points of class c according to their distances from the query point x. Then, we solve the linear regression equation

$$q_c \ln(r_s) = \ln(C_c) + \ln(s)\,,\ s = 1, 2, ..., K \tag{11}$$

for the first K points especially with respect to the unknown C_c. Number K may be a half or a square root or so of the total number N_c of the points of class c. This is made for all k classes, $c = 1, 2, .. k$. Finally, we use formula (9) for $k = 2$ or formula (10) for more than two classes. Formulas (9) or (10) give a real number. For two class classification, a discriminant threshold (cut) θ must be chosen, and then if $p(1|x) > \theta$, then x belongs to class 1 or else to class 0. The default value of θ is 0.5.

3.2 Error Analysis

There are two sources of errors. The first one depends on choosing the proper constant K, i.e. the number of nearest points to point x which is also the number of regression equations (11) used for computation of C_c. This is a problem very similar to the problem of the correlation dimension estimation. For correlation dimension estimation, many approaches including a lot of heuristic ones exist, see e.g. [2], [9], [11]; we do not discuss it in detail here.

The other kind of error is an error of estimation by linear regression. The Gauss–Markov theorem [12] states that in a linear model in which the errors have an expectation of zero and are uncorrelated and have equal variance, the best linear unbiased estimators of the coefficients are the least-squares estimators. At the same time, it holds that the regression coefficients, as random variables, have normal distribution [12], [14] each with a mean equal to the true value and with variance given by the well-known formulae [13] [14]. When the data is of the same quality, the variance converges to zero proportionally to $1/K$ for the number of samples K going to infinity.

In our case Gauss-Markov assumptions are well fulfilled, especially the assumption of homoscedasticity, i.e., all errors have the same variance. It is given by fact that each class usually represents a particular "source" of data with a particular statistic. Regression equations are constructed for each class separately here, i.e. all samples should have the same or very similar statistical characteristics including variance.

Variable $\ln(C_c)$ is found by linear regression and has normal distribution. Then variable C_c has lognormal distribution. From it, it follows that if $\mu_{\ln Cc}$ is the mean (also mode and median) of $\ln(C_c)$ and $\sigma^2_{\ln Cc}$ its variance then variable $C_c = \exp(\ln(C_c))$ has the median $M_e = \exp(\mu_{\ln Cc})$. The mean of C_c is $\exp(\ln(C_c)+ \sigma^2_{\ln Cc}/2)$, i.e. it is slightly larger than the median. On the other hand, the mode is slightly smaller as it holds that $M_o = \exp(\ln(C_c)-\sigma^2_{\ln Cc})$. Considering these three measures of position, we use the median for C_c estimation, using formula $C_c = \exp(\ln(C_c))$. $\ln(C_c)$ is found by the linear regression above. For variance of the lognormal distribution, it holds

$$\sigma^2_{Cc} = (\exp(\sigma^2_{\ln Cc}) -1).\exp(2\mu_{\ln Cc} + \sigma^2_{\ln Cc}).$$

From the fact that variance regression coefficients converge to zero proportionally to $1/K$ for the number of samples K going to infinity, the σ^2_{Cc} converges to zero proportionally to $1/K$ as well. Simply, for a small $\sigma^2_{\ln Cc}$ there is $\exp(\sigma^2_{\ln Cc}) \approx 1 + \sigma^2_{\ln Cc}$ and $\exp(2\mu_{\ln Cc} + \sigma^2_{\ln Cc}) = (\exp(\mu_{\ln Cc}))^2(1 + \sigma^2_{\ln Cc}) \approx C^2_c$. Then $\sigma^2_{Cc} \approx \sigma^2_{\ln Cc} C^2_c$ and because $\sigma^2_{\ln Cc} \sim 1/K$ and C^2_c is a constant here then $\sigma^2_{Cc} \sim 1/K$.

We can conclude that variable C_c converges to its true value as fast as the standard linear regression (11) used for estimation of its logarithm $\ln(C_c)$.

Error estimation

When using linear regression for (11), it is easy to state individual residuals ρ_i and thus to know the true sum of the squared residuals $\rho = \sum_{i=1}^{K} \rho_i^2$. The standard deviation on a parameter estimate is $\sigma_j = \sqrt{\frac{\rho}{K-1}\left[(X^t X)^{-1}\right]_{jj}}$, j=1, 2 and the 100(1-α)% confidence interval is $\beta_j \pm t_{\frac{\alpha}{2},K-2}\sigma_j$. Variables ρ and $[(X^tX)^{-1}]_{jj}$ are known during computation of $\ln(C_c)$ and thus one can get the confidence interval for $\ln(C_c)$ which is symmetric. Due to exponential transformation, C_c has an asymmetric confidence interval.

This confidence interval computation can be easily included into the construction of the classifier.

4. Experiments

The method described above has one free parameter to be set up, the number of nearest neighbors used for linear regression. We tested different possibilities, e.g. the square root of the total number of samples N_c of the learning set, one third, and one half of the number of samples of the learning set, and a simplest robust modification of the linear regression. We found that the use of a half of the total number N_c of samples of the learning set often to be quite practical.

Another strategy uses a robust procedure in linear regression. The approach starts with half of the points of the learning set nearest to the query point in the same way as the previous one. In

this step, the largest residuum is found and the corresponding sample of the learning set is excluded. This procedure is repeated until the largest residuum is small enough or ¼ of the total number N_c of the samples of the learning set remain. Then, the result of the linear regression is accepted.

The experiments described below follow the procedures described by Paredes and Vidal [15] as truly thorough tests. The tests consist of three kinds of experiments. The first one is a test with a synthetic data set [15] for which Bayes limit is known and one can estimate how close a particular approach allows one to get close to this limit. The second uses real-life data from the UCI Machine Learning repository [16]. The third consist of a more detailed comparison of the results for three selected data sets from [16].

In the experiments, we compare results obtained by the method described here with the results of some standard methods and up-to date Learning Weighted metrics method by Paredes and Vidal [15]. In each set of the tasks, we give a short description of the problem, the source of data, test procedure, results, and a short discussion.

4.1 Synthetic Data

Synthetic data [15] is two dimensional and consists of three two dimensional normal distributions with identical a-priori probabilities. If μ denotes the vector of the means and $\boldsymbol{C}_m$ is the covariance matrix, there is

Class A: $\mu = (2, 0.5)^t$, $\boldsymbol{C}_m = (1, 0; 0, 1)$ (identity matrix)

Class B: $\mu = (0, 2)^t$, $\boldsymbol{C}_m = (1, 0.5; 0.5, 1)$

Class C: $\mu = (0, -1)^t$, $\boldsymbol{C}_m = (1, -0.5; -0.5, 1)$.

In this experiment, we used a simple strategy of using half of the total number of samples of the learning set nearest to the query point.

Fig. 1 shows the results obtained by different methods for different learning sets sizes from 8 to 256 samples and testing set of 5000 samples all from the same distributions and mutually independent. Each point was obtained by averaging over 100 different runs.

In our method “QCregre”, we used a simple strategy of using half of the total number of samples of the learning set nearest to the query point in this experiment.

The results are shown in Fig. 1. For other methods, i.e. 1-NN method with L2 metrics and variants of the LWM method by Paredes and Vidal [15], the values were estimated from the literature cited.

In Fig 1, it is seen that the use of the class probability estimation with the method presented here in this synthetic experiment outperforms all other methods shown in Fig. 2 and for a large number of samples, it quickly approaches the Bayes limit.

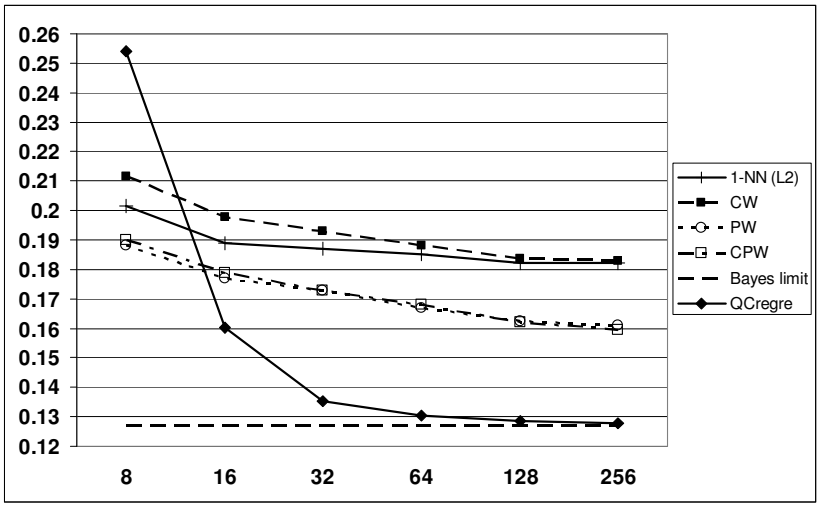

Fig. 1. Comparison of classification errors of the synthetic data for different approaches. In the legend, 1-NN (L2) means the 1-NN method with Euclidean metrics, CW, PW, and CPW are three variants of the method by Paredes and Vidal; points are estimated from the reference [15]. “Bayes” means the Bayes limit. QCregre means the method presented here.

4.2 Data from the Machine Learning Repository

Data sets prepared just for running with a classifier were prepared by Paredes and Vidal and are available on the net [17]. We used all data sets of this corpus. Each task consists of 50 pairs of training and testing sets corresponding to 50-fold cross validation. For DNA data [16], Letter data (Letter recognition [16]), and Satimage (Statlog Landsat Satellite [16]) the single partition into training and testing set according to the specification in [16] was used. We also added the popular Iris data set [16] with ten-fold cross validation.

The results obtained by the QCregre approach presented here, in comparison with data published in [15], are summarized in Table 1. Each row of the table corresponds to one task from [16]. For tasks where the data is not available from [15], only the results for 1-NN method with L2 metrics were amended.

In the QCregre method, we used a rather complex strategy of robust modification of linear regression as described above. The interesting point is the experiment with the simplest strategy of using half of the samples nearest to the query point. For some tasks we obtained very good results. In Table 2, the results are shown together with the results for other methods published in [16] for tasks “Heart”, “Ionosphere”, and “Iris”. Here, we shortly characterize these data sets as follows:

The task “Heart” indicates the absence or presence of heart disease for a patient.

For the 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.

The task “Iris” is to determine whether an iris flower is of class Versicolor or Virginica. The third class, Setoza is deleted, as it is linearly separable from the other two. 100 samples, four parameters and ten-fold cross validation were used, as in [18].

Table 1.
Classification error rates for different datasets and different NN-based approaches by [18] and LWM1. Empty cells denote data not available.

Dataset	L2	CDM	CW	PW	CW	QcregreL2
Australian	34.37	18.19	17.37	16.95	16.83	22.72
balance	25.26	35.15	17.98	13.44	17.6	41.32
cancer	4.75	8.76	3.69	3.32	3.53	4.08
diabetes	32.25	32.47	30.23	27.39	27.33	33.54
DNA	23.4	15	4.72	6.49	4.21	46.63
German	33.85	32.15	27.99	28.32	27.29	42.49
glass	27.23	32.9	28.52	26.28	27.48	49.46
heart	42.18	22.55	22.34	18.94	19.82	22.67
ionosphere	19.03					12.87
iris	6.91					5.00
led17	20.5					21.84
letter	4.35	6.3	3.15	4.6	4.2	44.20
liver	37.7	39.32	40.22	36.22	36.95	43.33
monkey1	2.01					10.91
phoneme	18.01					23.03
Satimage	10.6	14.7	11.7	8.8	9.05	28.95
segmen	11.81					15.87
sonar	31.4					40.01
vehicle	35.52	32.11	29.38	29.31	28.09	45.96
vote	8.79	6.97	6.61	5.51	5.26	8.79
vowel	1.52	1.67	1.36	1.68	1.24	16.81
waveform21	24.1	0	0	0	0	52.97
waveform40	31.66	0	0	0	0	58.12
wine	24.14	2.6	1.44	1.35	1.24	8.68

We do not describe these tasks in detail here as all of them can be found in descriptions of individual tasks of the Repository and also the same approach to testing and evaluation was used. Especially, splitting the data set into two disjoint subsets, the learning set and the testing set and the use of cross validation were the same as in [16] or – for the Iris database as in [18].

We also checked some standard methods for comparison as follows:

- 1-NN – standard nearest neighbor method [19]
- Sqrt-NN – the *k*-NN method with *k* equal to the square root of the number of samples of the learning set [10]
- Bay1 – the naïve Bayes method using ten bins histograms [20]
- LWM1 – the learning weighted metrics by Paredes and Vidal [15].

For *k*-NN, Bayes, LWM and our method the discriminant thresholds θ were tuned accordingly. All procedures are deterministic (even Bayes algorithm) and then no repeated runs were needed.

Table 2.

Classification errors for three different tasks shown for the different methods presented in the Machine Learning Repository. The note [fri] means the results according to the report by Friedman [18]. The results computed by authors are shown in bold.

Heart		**Ionosphere**		**Iris**	
Algorithm	Test	Algorithm	Error	Algorithm	Test
QCregre1	**0.178**	**QCregre1**	**0.02013**	scythe[fri]	0.03
LWM1	0.189	**Bay1**	0.02013	**QCregre1**	**0.04878**
Bayes	0.374	**LWM1**	0.0265	**sqrt-NN**	0.04879
Discrim	0.393	IB3 (Aha & Kibler, IJCAI-1989)	0.033	mach:ln [fri]	0.05
LogDisc	0.396	backprop an average of over	0.04	mach-bth [fri]	0.05
Alloc80	0.407	**sqrt-NN**	0.0537	CART	0.06
QuaDisc	0.422	Ross Quinlan's C4 algorithm	0.06	mach [fri]	0.06
Castle	0.441	nearest neighbor	0.079	mach:ds [fri]	0.06
Cal5	0.444	"non-linear" perceptron	0.08	**1-NN**	0.0609
Cart	0.452	"linear" perceptron	0.093	**LWM1**	0.0686
Cascade	0.467			**Bay1**	0.0854
KNN	0.478			CART	0.11
Smart	0.478			k-NN	0.8
Dipol92	0.507				
Itrule	0.515				
BayTree	0.526				
Default	0.56				
BackProp	0.574				
LVQ	0.6				
IndCart	0.63				
Kohonen	0.693				
Ac2	0.744				
Cn2	0.767				
Radial	0.781				
C4.5	0.781				
NewId	0.844				

5. Discussion

In the first part of this discussion we show that the approach presented can be useful in some cases. Some notes on the computational complexity and relation of the distribution mapping exponent to the correlation dimension follow:

The main goal of this chapter is to show that the correlation integral can be decomposed into local functions – the probability distribution mapping functions (PDMF). Each PDMF corresponds to a particular point of data space and characterizes the probability distribution in some neighborhood of a given point. In fact, the correlation integral is a distribution function of the binate distances of the data set, and PFMF is a distribution function of the distances of the points of the data set from a particular point, the query point x. We have also shown that – similarly as the correlation integral – the PDMF can be approximated by a polynomial function. This polynomial approximation is governed by two constants, the distribution mapping exponent, which can be considered as the local analog to the correlation dimension, and a multiplicative constant. It is proven here that this multiplicative constant is very closely related to the probability density at the given point. The estimation of this constant is used to construct a classifier.

This classifier is slightly related to the nearest neighbor methods. It uses information about distances of the neighbors of different classes from the query point and neglects information about the direction where the particular neighbor lies.

Nearest neighbor methods do not differentiate individual distances of nearest points. E.g. in the k-NN method the number of points of one and the other class among k nearest neighbors is essential, but not the individual distances of points. The method proposed here takes the individual distances into account even if these distances are a little bit hidden in the regression equations. The method outperforms 1-NN, k-NN as well as LWM (learning weighted metrics) by Paredes and Vidal [15] in many cases and can be found as the best one for some tasks.

By use of the notion of distance, i.e. a simple transformation $E_n \rightarrow E_1$, the problems with the curse of dimensionality are easily eliminated at the loss of information on the true distribution of the points in the neighborhood of the query point. The curse of dimensionality [21], [22] means that the computational complexity grows exponentially with dimensionality n, while the complexity only grows linearly here.

The method has no tuning parameters except for those related to linear regression. There is no true learning phase. In the "learning phase" only the standardization constants are computed and thus this phase is several orders of magnitude faster than the learning phase of the neural networks or other various methods.

In the regression equations there are multiplicative constants C_c. We have shown that these constants are proportional to the probabilities $p(c|x)$ that point x is of class c. Thus, C_c allows one to differentiate between the densities of the classes at point x and the distribution mapping exponent q has no use in this task. One can deduce that neither the correlation dimension nor the distribution mapping exponent govern the probability that point x is of a class c. Their role in the probability density estimation and classification is indirect via polynomial transformation only.

There is an interesting relationship between the correlation dimension and the distribution mapping exponent q_c. The former is a global feature of the fractal or data generating process; the latter is a local feature of the data set and is closely related to the particular query point. On the other hand, if linear regression were used, the computational procedure is almost the same in both cases. Moreover, it can be found that values of the distribution mapping exponent usually lie in a narrow interval <-10, +10> percentage around the mean value.

The question arises what is the relation of the distribution mapping exponent statistics to the overall accuracy of the classification.

Acknowledgements

This work was supported by the Ministry of Education of the Czech Republic under the project Center of Applied Cybernetics No. 1M0567, and No. MSM6840770012 Transdisciplinary Research in the Field of Biomedical Engineering II.

References

[1] P. Grassberger, I. Procaccia.: Measuring the strangeness of strange attractors, Physica, Vol. 9D, pp. 189-208, (1983)

[2] A. R. Osborne, A. Provenzale.: Finite correlation dimension for stochastic systems with power-law spectra. Physica D Vol. 35, pp. 357-381, (1989)

[3] N. Lev: Hausdorff dimension. Student Seminar, Tel-Aviv University, 2006, www.math.tau.ac.il/~levnir/files/hausdorff.pdf

[4] E. W. Weisstein: Information Dimension. From MathWorld--A Wolfram Web Resource. http://mathworld.wolfram.com/InformationDimension.html, 2007.

[5] C. J. Merz, P. M. Murphy, D. W. Aha. UCI Repository of Machine Learning Databases. Dept. of Information and Computer Science, Univ. of California, Irvine, http://www.ics.uci.edu/~mlearn/MLSummary.html (1997).

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

[7] F. Takens: On the Numerical Determination of the Dimension of the Attractor. In: Dynamical Systems and Bifurcations, in: Lecture Notes in Mathematics, Vol. 1125, Springer, Berlin, p. 99-106 (1985).

[8] F. Camastra: Data dimensionality estimation methods: a survey. Pattern Recognition Vol. 6, pp. 2945-2954 (2003).

[9] A. Guerrero and L. A. Smith. Towards coherent estimation of correlation dimension. Physics letters A, vol. 318, pp. 373-379, (2003).

[10] R. O. Duda, P. E. Hart and D. G. Stork. Pattern classification, Second Edition, John Wiley and Sons, Inc., New York, (2000).

[11] I. Dvorak and J. Klaschka. Modification of the Grassberger-Procaccia algorithm for estimating the correlation exponent of chaotic systems with high embedding dimension. Physics Letters A, Vol. 145, No. 5, pp. 225-231, (1990).

[12] Wikipedia - Gauss–Markov theorem - http://en.wikipedia.org/wiki/Gauss–Markov_theorem.

[13] Wikipedia – Linear regression - http://en.wikipedia.org/wiki/ Linear_regression.

[14] E. E. Leamer. Specification searches. Ad hoc inference with non-experimental data. John Wiley and Sons, New York (1978).

[15] R. Paredes, and E. Vidal: Learning Weighted Metrics to Minimize Nearest-Neighbor Classification Error. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 20, No. 7, pp. 1100-1110 (2006).

[16] C. J. Merz, P. M. Murphy, D. W. Aha: UCI Repository of Machine Learning Databases. Dept. of Information and Computer Science, Univ. of California, Irvine, http://www.ics.uci.edu/~mlearn/MLrepository.html (1997).

[17] R. Paredes: http://www.dsic.upv.es/~rparedes/ research/CPW/index.html

[18] J. H. Friedmann: Flexible Metric Nearest Neighbor Classification. Technical Report, Dept. of Statistics, Stanford University (1994).

[19] T. M. Cover and P. E. Hart. Nearest Neighbor Pattern Classification. IEEE Transactions on Information Theory, vol. IT-13, No. 1, pp. 21-27 (1967).

[20] J. Gama: Iterative Bayes. Theoretical Computer Science Vol 292, pp. 417-430 (2003).

[21] R. E. Bellman. Adaptive Control Processes. Princeton University Press, (1961).

[22] V. Pestov. On the geometry of similarity search: Dimensionality course and concentration of measure. Information Processing Letters, Vol. 73, No. 1-2, pp. 47-51 (2000).