Российская Академия Наук

РОССИЙСКАЯ АКАДЕМИЯ НАУК

ИНСТИТУТ ПРОБЛЕМ БЕЗОПАСНОГО РАЗВИТИЯ АТОМНОЙ ЭНЕРГЕТИКИ

RUSSIAN ACADEMY OF SCIENCES

NUCLEAR SAFETY INSTITUTE

Препринт ИБРАЭ № IBRAE-2002-09

Preprint IBRAE-2002-09

A. Pozdnukhov, V. Timonin, M. Kanevski, E. Savelieva, S. Chernov

CLASSIFICATION OF ENVIRONMENTAL DATA WITH KERNEL BASED ALGORITHMS

Москва
2002

Moscow
2002

УДК 502.3

Позднухов А., Тимонин В., Каневский М., Савельева Е., Чернов С. КЛАССИФИКАЦИЯ ДАННЫХ ПО ОКРУЖАЮЩЕЙ СРЕДЕ С ИСПОЛЬЗОВАНИЕМ «ЯДЕРНЫХ АЛГОРИТМОВ». (На англ. яз.). Препринт № IBRAE-2002-09. Москва: Институт проблем безопасного развития атомной энергетики РАН, 2002. 23 с.

Аннотация

Классификация типов почв является важной проблемой с различных точек зрения. В качестве примера можно рассмотреть вертикальную миграцию радионуклидов в почве. Процесс миграции в почвах зависит от различных характеристик и радионуклидов, и почв. Все свойства почвы сильно взаимосвязаны с типом почвы. Официальные карты типов почв недостаточно хороши, для того чтобы их использовать для решения проблем миграции радионуклидов. Реальный тип почвы является более переменной величиной, чем обычно представлено в официальных картах. Картирование типов почв может быть улучшено при использовании дополнительной информации, полученной во время измерений концентрации радионуклидов. В данной работе проблема классификации почв решается с использованием методов «машинного обучения», таких как вероятностные нейронные сети и машины поддерживающих векторов (Support Vector Machines). Преимущество обоих методов заключается в нелинейном моделировании, которое позволяет избежать непосредственного моделирования пространственной корреляционной структуры. Проведено сравнение этих методов с методом «ближайшего соседа», который является самым простым подходом для решения задач пространственной классификации.

Pozdnukhov A., Timonin V., Kanevski M., Savelieva E., Chernov S. CLASSIFICATION OF ENVIRONMENTAL DATA WITH KERNEL BASED ALGORITHMS. Preprint IBRAE-2002-09. Moscow: Nuclear Safety Institute RAS, 2002. 23 p.

Abstract

Soil type classification is an important problem from different points of view. Vertical migration of radio-nuclides in soils can be mentioned as an example. The process of migration in soils depends on a number of different properties corresponding both to radio-nuclides and soils. All soil properties are strongly connected with a soil type. Official soil type maps are not good enough to be used for migration problems. Real soil type is more variable value, than it is usually presented in official maps. Soil type mapping can be improved by using additional information obtained during radio-nuclide concentration measurement.

In this work the classification problem is solved by machine learning methods such as probabilistic neural networks PNN (supervised learning algorithm) and Support Vector Machines SVM (based on statistical learning theory). Both methods are called “kernel methods” but “kernel” in SVM defines mapping into feature space and it should not be confused with “kernel” in PNN that follows from the non-parametric kernel density estimate. The advantages of both methods are general non-linear modelling that avoids the direct modelling of spatial correlation structure. The methods are compared with the nearest neighbour method, the simplest approach to spatial classification.

Classification of Environmental Data with Kernel Based Algorithms

A. Pozdnukhov, V.Timonin, M. Kanevski, E. Savelieva, S. Chernov

THE INSTITUTE OF NUCLEAR SAFETY
113191, Moscow, B. Tulskaya, 52

tel.: (095) 955-22-31, fax: (095) 958-11-51, E-mail: anp@ibrae.ac.ru; http://www.ibrae.ac.ru/~mkanev

Contents

Contents 3
Introduction 4
1 Basic theory of applied methods 4
1.1. Support Vector Machines 4
1.1.1. Main principles of SVM 4
1.1.2. Maximum Margin Classifier 5
1.1.3. Linearly separable case 6
1.1.4. Soft Margin Classifier 7
1.1.5. SVM non-linear classifier 7
1.1.6. Multi-class classification 9
1.2. Probabilistic Neural Network 10
2 Case study 12
2.1. Data description 12
2.2 Variography 13
2.3. Training of the Models 16
2.3.1. Support Vector Machines 16
2.3.2. Probabilistic Neural Network 17
2.3.3 Method with K Nearest Neighbours 18
2.4. Accuracy Test (Analysis of Training Error) 18
2.5. Validation Data Set Classification 20
2.6. Cegorical Data Mapping 22
Conclusions 23
Acknowledgements 23
References 23

Introduction

Soil type classification is an important problem from different points of view. Vertical migration of radio-nuclides in soils can be mentioned as an example. The process of migration in soils depends on a number of different properties corresponding both to radio-nuclides and soils. All soil properties are strongly connected with a soil type. Official soil type maps are not good enough to be used for migration problems. Real soil type is more variable value, than it is usually presented in official maps. Soil type mapping can be improved by using additional information obtained during radio-nuclide concentration measurement.

Real classification of the soil types problem is a multi class one. It also can be considered as a regionalized categorical variable mapping.

Different classification methods have been developed to solve classification problems. The classic approach is based on discriminate analysis. But such approach deals with linear separable cases only. Some modifications use a non-linear formula for line description. But these modifications appeared to be too complex for practical use, except some special cases when the form of line is directly observed.

Geostatistical indicator approach can be proposed for environmental classification problems also. The problem with all geostatistical methods is in necessity of estimation and modelling of the spatial correlation structure (variogram). The quality of this modelling strongly influences on the final result. The procedure of spatial correlation structure modelling often appears to be significantly complex, because of the low number of class members, their distribution, etc. In such situations the usage of geostatistics is highly complicated.

In recent years the analysis and processing of spatially distributed and time dependent data has become a very important subject due to from one side comprehensive development of environmental and pollution monitoring networks even leading to data mining problems and from another side to much better understanding of data driven machine learning models. Nowadays, several approaches are widely used for spatio-temporal data analysis and modelling: artificial neural networks (MLP multilayer perceptrons, RBFNN radial basis function neural networks, SOMs self-organized maps, GRNN general regression neural networks, etc.).

In this work the classification problem is solved by machine learning methods such as probabilistic neural networks PNN (supervised learning algorithm) and Support Vector Machines SVM (based on statistical learning theory). Both methods are called “kernel methods” but “kernel” in SVM defines mapping into feature space and it should not be confused with “kernel” in PNN that follows from the non-parametric kernel density estimate. The advantages of both methods are general non-linear modelling that avoids the direct modelling of spatial correlation structure. The methods are compared with the nearest neighbour method, the simplest approach to spatial classification.

1 Basic theory of applied methods

1.1. Support Vector Machines

Support Vector Machines (SVM) is an approach based on Statistical Learning Theory or Vapnik-Chervonenkis (VC)-theory [1]. In this work it is used for analysis and modeling of spatially distributed environmental information (categorical data).

SVM is a learning algorithm, which attempts to minimize simultaneously the empirical risk or error (estimation of an error on the training data set) and the structural risk (complexity of the model). By opposition to the traditional methods based on experimental statistical moments of data such as mean and variance, SVM are focusing on the marginal data and not on statistics. SVM promises to give good generalization abilities, i.e. it concentrates on the regular structure of the data.

Originally SVM's were developed for the binary (2 class) classification. Recently different generalization schemes of 2-class classification problem to multi-class classification were developed as well.

1.1.1. Main principles of SVM

Let us first consider classical 2-class classification problem, formulated as follows:

- Given: a set of samples $\{x_1, y_1\} \ldots \{x_L, y_L\}$, where x_i for $i=1 \ldots L$ is a feature vector and $y_i=\{+1, -1\}$ is a class label for x.
- Find: a classifier of a kind $f(x)$ such that $y = f(x)$, where y is the class label for x.

Such classifier is developed from a learning machine with adjustable set of intrinsic parameters λ. To solve the given classification task the machine will tune its parameters to learn the required mapping from the set of given samples (training set). This will result in the fixed value of parameter λ. The next step is to evaluate the performance of this machine. It can be measured by:

$$R(\lambda) = \int E(y, f(x,\lambda))dP(x,y), \quad \text{where } E(y, f) = \begin{cases} 0, \text{ if } y = f \\ 1, \text{ otherwise} \end{cases}. \tag{1}$$

Here $R(\lambda)$ is called an expected risk. The problem is that it is unknown, since $P(x, y)$ is unknown for most real data sets. Hence, we can't compute the expected risk directly, but we can compute the empirical risk:

$$R_{emp}(\lambda) = \frac{1}{L}\sum_{i=1}^{L} E(y, f(x,\lambda)). \tag{2}$$

Formula (2) is just the measure of the mean error over available data set. Most of the traditional algorithms for learning machines minimize the empirical error using Maximum Likelihood estimation for the parameter λ and do not consider the capacity of the machine. This can results in much more capacity than required to the set classification. Small training error doesn't guarantee high generalization ability, i.e. small error on the independent testing data set. The best approach to the problem is to develop a machine which would find a good trade-off between the low empirical error and small capacity of the machine. This approach was formulated in Structural Risk Minimization principle, which is based on the fact that the following estimation holds with possibility of at least $1-\eta$:

$$R(\lambda) \le R_{emp}(\lambda) + \sqrt{\frac{h(\log\frac{2L}{h}+1)-\log(\frac{\eta}{4})}{L}} \tag{3}$$

The parameter h is called the VC (Vapnik-Chervonenkis)-dimension and describes the capacity of the set of functions. As it can be seen from (3), there is an optimum point in a trade-off between R_{emp} and the VC-dimension h for the given machine. The machine, learned according to Structural Risk Minimization principle, gives low training error and has good generalization abilities.

1.1.2. Maximum Margin Classifier

Now let us consider two linearly separable sets in R^N, $(x_1, y_1)\ldots(x_L, y_L)$, $y = \{-1, +1\}$. The decision function of the classifier for such sets is $f(x,\{\mathbf{w}, b\}) = sign(\mathbf{w}\cdot\mathbf{x} + b)$, where $x, w \in R^N$. The pair of parameters $\{\mathbf{w}, b\}$ defines the separating hyper-plane. The Support Vector algorithm is based on a structure of set of these separating hyper-planes. Notice, that the choice of $\{\mathbf{w}, b\}$ is arbitrary, so we can scale $\{\mathbf{w}, b\}$ such that $\min_i |(w \cdot x_i) + b| = 1$. If the function $f(x,\{\mathbf{w}, b\})$ exists (in linearly separable case it is true always), the following condition is held for every point of the data set:

$$y_i(w \cdot x_i + b) \ge 1, \ i = 1,\ldots,L. \tag{4}$$

The equality holds when the point lies the most closely to the hyper-plane. At the same time it means that the distance from the hyper-plane to the closest points of the training set will be $1/\|w\|$. Thus, the margin between the classes will be at least $2/\|w\|$. It is intuitively clear that the classifier with the largest margin is optimal. It can be proved [1], that such hyper-plane satisfies the SRM principle, i.e. keeps both the training error and the VC-dimension small.

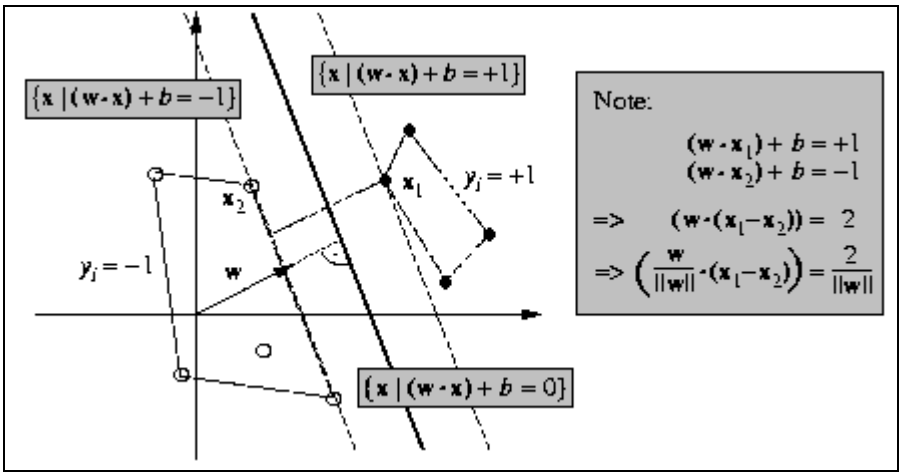


Fig. 1. Separating hyperplane for the 2D problem.

1.1.3. Linearly separable case

The data set *S* is linearly separable if there exist $W \in R^2, b \in R$, such that

$$Y_i(W^T X_i + b) \geq +1, \quad i = 1,...N \tag{5}$$

The pair *(W,b)* defines a hyper-plane by an equation

$$(W^T X + b\ ) = 0\,.$$

Linearly separable problem is stated as follows: for given training samples $\{X_i,Y_i\}$ find the optimum values of the weight vector *W* and bias *b* such that they satisfy constraints

$$Y_i(W^T X_i + b) \geq +1, \quad i = 1,...N \tag{6}$$

and such that the weight vector *W* minimizes the cost function (maximization of the margins)

$$F(W) = W^T W / 2\,. \tag{7}$$

The cost function is a convex function of W and the constraints are linear in W.

This constrained optimization problem can be solved by using Lagrange multipliers. Lagrange function is defined by

$$L(W,b,\alpha) = W^T X / 2 - \sum_{i=1}^{N} \alpha_i \left[Y_i(W^T X_i + b) - 1\right],$$

where Lagrange multipliers $\alpha_i \geq 0$.

The solution of the constrained optimization problem is determined by the saddle point of the Lagrangian function $L(W,b,\alpha)$, which has to be minimized with respect to W and *b* and to be maximized with respect to α.

Because constrained optimization problem deals with a convex cost function, it is possible to construct dual optimization problem. The dual problem has the same optimal value as the primal problem, but with the Lagrange multipliers providing the optimal solution. The dual problem is formulated as follows:

1. Maximize the objective function (8)

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - (1/2) \sum_{i=1}^{N} \alpha_i \alpha_j Y_i Y_j X_i^T X_j \tag{8}$$

2. Subject to the constraints (9, 10)

$$\sum_{i=1}^{N} \alpha_i Y_i = 0, \qquad \alpha_i \geq 0, \mathrm{i} = 1,...\mathrm{N} \tag{9,10}$$

Note that the dual problem is presented only in terms of the training data. Moreover, the objective function $Q(\alpha)$ (8) depends only on the input patterns in the form of a set of dot products $\{X_i^T X_j\}_{i=1,2,...N}$.

After determining optimal Lagrange multipliers α_{i0} the optimum weight vector is defined by (4) and bias is calculated as $b = 1 - W^T X_i^S$, for $Y^{(s)} = +1$

Note that from the Kuhn-Tucker conditions it follows that

$$\alpha_i \left[Y_i (W^T X_i + b) - 1 \right] = 0 \tag{11}$$

Only non-zero α_i in this equation are those for which constraints are satisfied with the equality sign. The corresponding points X_i are called *Support Vectors.* They are the points of the set ***S*** the closest to the optimal separating hyper-plane. In many applications number of support vectors is much less that original data points.

The problem of classifying a new data point X is performed by computing $F(X)$ with the optimal weights W and bias b (12).

$$F(X) = sign(W^T X_i + b). \tag{12}$$

1.1.4 Soft Margin Classifier

All the techniques described before for linearly separable sets can be extended to the non-separable sets by adding a slack variables $\xi_i \geq 0$ to the constraints (6) [Cortes and Vapnic, 1995]:

$$Y_i (W^T X_i + b) \geq +1 - \xi_i, \ \xi_i \geq 0, \forall i \tag{12}$$

Values ξ_i should be non zero as few as possible. So now the task is to minimize the functional:

$$F(W) = W^T W / 2 + C \sum_{i=1}^{N} \xi_i \tag{13}$$

subject to the constraints (13). The first term in (13) corresponds to the minimizing of the VC-dimension, the second one corresponds to minimizing the number of misclassified points of the training set. Positive constant C is weighting the second criterion with respect to the first one. The dual form of this problem is:

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - (1/2) \sum_{i=1}^{N} \alpha_i \alpha_j Y_i Y_j X_i^T X_j \tag{14}$$

$$\sum_{i=1}^{N} \alpha_i Y_i = 0, \quad 0 \leq \alpha_i \leq C, \ \mathrm{i} = 1,...\mathrm{N} \tag{15,16}$$

The parameter C has to be selected by user. This is usually done by one of the two following ways:

1) C is determined experimentally via the standard use of a training and testing data sets, which is a form of re-sampling;
2) It is determined analytically by estimating VC dimension and then by using bounds on the generalization performance of the machine based on a VC dimension [1].

1.1.5. SVM non-linear classifier

In most practical situations the classification problems are non-linear and the hypothesis of linear separation in the input space is too restrictive.

The basic idea of Support Vector Machines can be expressed in two following statements:

1) to map the data into a high dimensional feature space (possibly of infinite dimension) via a non-linear mapping;

2) to construct the optimal hyper-plane for separating features (application of the linear algorithms described above).

The first item is in agreement with Cover's theorem on the separability of patterns. It states that input multidimensional space can be transformed into a new feature space where the patterns are with high probability linearly separable. The separability is provided by:

1. the transformation is non-linear;
2. the dimension of the feature space is high enough [1]-[3].

The idea of the non-linear mapping into high-dimensional space is illustrated in figure 2.

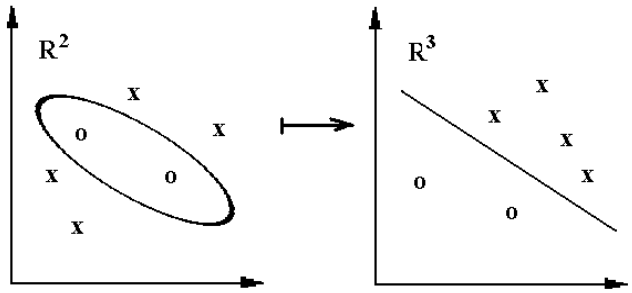


Fig.2. Non-linear mapping sample. $(x, y) \to (x^2, \sqrt{2}xy, y^2)$

Cover's theorem does not discuss the optimality of the separating hyper-plane. By using Vapnik's optimal separating hyper-plane VC dimension is minimized and generalization is achieved. Notice that optimization procedure in the dual form and the decision function calculation requires nothing except the evaluation of dot products.

Let $\{\varphi_j(x)\}_{j=1,2...}$ denote a set of non-linear transformations from the input space to some *m*-dimensional space named feature space. Then the dot product of the transformed vectors can be evaluated as:

$$\sum_{j=1}^{\infty} \varphi_j(X)\varphi_j(Y) = \varphi^T(X)\varphi(Y) = K(X,Y), \tag{17}$$

where the inner-product kernel function $K(X,Y)$ is introduced. It can be used to construct the optimal hyper-plane in the feature space without having to consider the feature space itself in explicit form. The existence of such representation is based on the Mercer theorem, stating that a continuous symmetric function $K(x,y)$ such that

$$\iint_{DxD} K(x,y)g(x)g(y)dxdy \geq 0 \tag{18}$$

for all $g(x) \in L^2(D)$ where D is compact subset of R^N can be expanded in a uniformly convergent series in terms of the eigen-functions of the corresponding integral operator:

$$K(x,z) = \sum_{j=1}^{\infty} \lambda_j \phi_j(x)\phi_j(z) \tag{19}$$

with associated eigenvalues $\lambda_j > 0$. After rescaling we obtain the expression (17).

The following three common types of kernels are widely used in Support Vector Machines:

1. Polynomial kernel – $K(X, X_j) = (X^T X_i + 1)^p$, where power p is specified a priori by the user. Mercer's conditions are always satisfied.
2. Radial basis function (RBF) kernel – $K(X, X_j) = \exp\left\{-\left\|X - X_j\right\|^2 / 2\sigma^2\right\}$. The kernel's bandwidth

σ (sigma value) is usually specified by a user on the base of a priori knowledge. In general, Mahalanobis distance is a sufficient variable. Mercer's conditions are always satisfied.

3. Two-layer perceptron – $K(X, X_j) = \tanh\{\beta_0 X^T X_j + \beta_1\}$. Mercer's conditions are satisfied only for some values of β_0 and β_1.

We can obtain the non-linear classifier by substituting the dot products in (14)-(16) by kernels and the optimization problem for non-linear case in the dual form is the following:

Given the training data maximize the objective function (find the Lagrange multipliers)

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - (1/2) \sum_{i=1}^{N} \alpha_i \alpha_j Y_i Y_j K(X_i, X_j), \tag{20}$$

where $K(X_i,X_j)$ is a kernel function, subject to the constraints (9) and also

$$0 \le \alpha_i \le C, \mathrm{i} = 1,...\mathrm{N} \tag{21}$$

The optimal hyper-plane is now defined as

$$f(X) = \sum_{j=1}^{N} \alpha_j Y_j K(X, X_j) + b \tag{22}$$

Finally, the non-linear decision function is defined by the following relationship analogous to (12):

$$F(X) = sign\left[W^T K(X, X_j) + b\right] \tag{23}$$

1.1.6. Multi-class classification

A multi-class classification problem can be solved by the different reductions of the primary problem to several dichotomies. So, there are several possibilities for the multi-class classification with SVM. The most evident method is *one-to-rest or one-against-all* classification, when M binary classifiers are constructed, one for each class, and the samples of each class are compared with all other samples. Since we train M SVMs - and M sets of optimized weights and biases, the resulting decision function will be:

$$y_i = \arg\max_m \sum_i \alpha_i^{(m)} Y_i K(X, X_i) + b^{(m)} \tag{24}$$

where over all training samples are summarized and *argmax* over all the decision functions is taken.

Another way of reducing the *M*-class classification to a binary one is a *pair-wise* approach, when *M(M-1)* binary classifiers are constucted to compare all pairs of classes. In any case binary decision hypotheses should be anyhow combined. Here it is important to choose the appropriate decision rule for the winner class. One of the possible variants is:

$$y_i = \arg\max_m \frac{2}{K(K-1)} \sum_{i \ne j} f(i \mid i, j), \tag{25}$$

where the pairwise probabilities to be proportional to the decision functions are estimated.

If following the idea of binary classifiers' combination, some other methods generalizing those mentioned above were suggested. For example, such as error correcting output codes algorithm and its generalizations. Another way is direct generalization of the SVM to multi-class problems. The main disadvantage of this method is that the QP-problem size becomes very large.

One-to-rest and pair-wise schemes seem to give good enough results for the geostatistical applications. When constructing the set of binary classifiers different (i.e. class-adaptive) kernel parameters for every classifier are used. It allows to take somehow into account different spatial variability of classes. The main practical problem in this approach is the necessity to tune the parameters of large amount of classifiers.

1.2. Probabilistic Neural Network

Probabilistic Neural Network (PNN) is a supervised neural network widely used in the area of pattern recognition, non-linear mapping, and estimation of the probability of the class membership. Let us consider *K* classes (or generators of random variables) c_i (i=1,2,..,K), each class having a specific probability density function (p.d.f).

Each of these generators produces realizations (samples) $\boldsymbol{x}$ with some *prior* probability $P(c_i)$. Prior probability can be interpreted as the initial (guess) class conditional distribution $p(\boldsymbol{x}|c_i)$ for all $\boldsymbol{x}$. Generally, the prior class distribution is highly dependent on the specific task and should be determined by an additional (physical) knowledge of the problem. When none of such additional information is available, all $P(c_i)$ are assumed to be equal.

The classification problem is to construct a classifier (model) able to decide to which one of *K* classes does belong the unknown sample $\boldsymbol{x}$. PNN uses Bayesian optimal or *maximum a posterior* (MAP) decision rule:

$$C(x) = \{c_1, c_2, ..., c_K\} = \underset{c_i}{argmax}\, P(c_i)\, p(x \mid c_i) \qquad i = 1,2,...,K \tag{18}$$

To realize this approach, the density of each class $p(\boldsymbol{x}|c_i)$ have to be estimated first. There are two basic approaches for density estimation: *parametric* and *non-parametric*. In the first one, the model of distribution (multivariate Gaussian, for example) is assumed and the necessary parameters are estimated from the training set. In the second approach, $p(\boldsymbol{x}|c_i)$ is estimated directly from the training set without applying specific hypotheses on distributions.

The most often used method for non-parametric density estimation was suggested by Parzen [8]. It uses *weight*, or *potential*, or *kernel* function $W(x)$ that must:

- be bounded: $\sup_x \left| W(x) \right| < \infty$
- rapidly go to zero as its argument increases in absolute value: $\int_{-\infty}^{\infty} \left| W(x) \right| dx < \infty \qquad \lim_{x \to \infty} \left| x\, W(x) \right| = 0$
- be properly normalized: $\int_{-\infty}^{\infty} W(x)\, dx = 1$

Different kernels also can be used, for example triangular:

$$W(x) = \begin{cases} 1 - x & x < 1 \\ 0 & else \end{cases}, \tag{19a}$$

or rectangular one:

$$W(x) = \begin{cases} 0.5 & x < 1 \\ 0 & else \end{cases}. \tag{19b}$$

But the Gaussian function is the most often used kernel:

$$W(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\left(\frac{-x^2}{2\sigma^2}\right)} \qquad \sigma > 0, \tag{19}$$

here $\sigma>0$ is the scaling parameter, or the smoothing factor ("bandwidth" of the bell). For multivariate case it can be rewritten as:

$$W(X) = \prod_{i=1}^{p} W_i(x_i) = \frac{1}{(2\pi)^{p/2}\sigma^p} e^{\left(\frac{-\|X\|^2}{2\sigma^2}\right)} \quad X = (x_1, \ldots, x_p) \tag{20}$$

where $\boldsymbol{p}$ is dimension of $\boldsymbol{X}$.

Parzen's p.d.f. estimator is:

$$p.d.f(X) = \frac{1}{N}\sum_{n=1}^{N} W(X - X_n) = \frac{1}{(2\pi)^{p/2}\sigma^p N}\sum_{n=1}^{N} e^{\frac{-\|X-X_n\|^2}{2\sigma^2}} \tag{21}$$

where $\boldsymbol{N}$ is a number of samples in the training set.

The general PNN structure was proposed by Specht in [9] and is a direct implementation of the above p.d.f. estimator and Bayesian decision rule. It consists of three feed-forward layers (Fig. 3):

- *Input layer:* accepts sample vectors $\boldsymbol{X}$ and supplies them to all of the neurons in the next pattern layer;
- *Pattern layer:* consists of $\boldsymbol{K}$ pools of pattern neurons corresponding to $\boldsymbol{K}$ classes. In each pool $\boldsymbol{i}$, there are N_i number of pattern neurons;
- *Summation layer:* consists of $\boldsymbol{K}$ neurons, where $\boldsymbol{i}$-th neuron forms the average sum of all outputs from the $\boldsymbol{i}$-th pool of the previous pattern layer.

So, the output of the summation layer for $\boldsymbol{i}$-th class is:

$$p(x \mid c_i) = \frac{1}{(2\pi)^{p/2} N_i \sigma^p}\sum_{n=1}^{N_i} e^{\frac{-\|x-x_i^{(n)}\|^2}{2\sigma^2}} \qquad \sigma > 0, \tag{22}$$

where N_i is a- number of samples that belong to class c_i (class size), $x_I^{(n)}$ represents the $\boldsymbol{n}$-th sample of class c_i.

Output neuron just compares $\boldsymbol{K}$ outputs from the previous summation layer with weights determined by the prior class distribution $P(c_i)$, and make the decision. Sample $\boldsymbol{x}$ belongs to class with the largest output value from the summation layer. Furthermore, due to assumption that any sample $\boldsymbol{x}$ belongs to one of $\boldsymbol{K}$ classes, the Bayesian confidence (a posterior probability of belonging $\boldsymbol{x}$ to class c_i) can be estimated as follows:

$$P(x \mid c_i) = \frac{p(x \mid c_i)}{\sum_{k=1}^{K} p(x \mid c_k)} \tag{23}$$

The example of a simple PNN with $\boldsymbol{p} = 2$, $\boldsymbol{K} = 2$, $N_1 = N_2 = 2$ is shown in Figure 3.

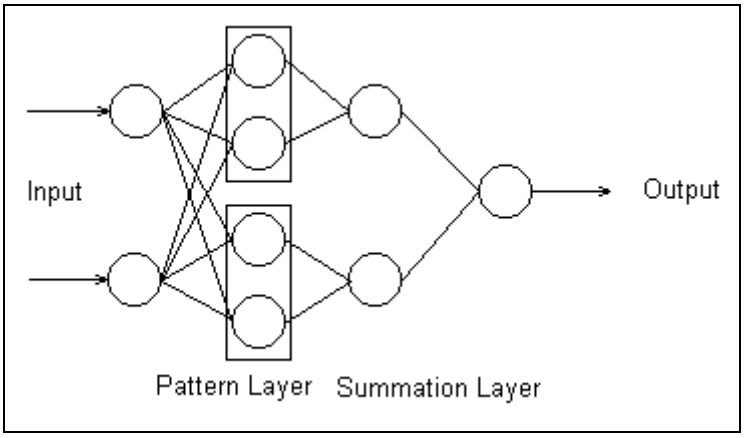


Fig. 3. PNN structure with $\boldsymbol{p} = 2$, $\boldsymbol{K} = 2$, $N_1 = N_2 = 2$

The above discussion shows that the training of a PNN is very simple, because the only parameter σ needs to be optimized. Low and upper boundaries of σ can be easily select, and then 1D-minimization procedure locates one well-defined global minimum. Well-known in statistics k-fold cross-validation method can be used as criterion of the quality of developed PNN.

Using Bayes posterior probability, continuous error function for σ optimization procedure can be estimated as:

$$e(x \mid c_i) = \left[1 - P(x \mid c_i)\right]^2 + \sum_{k \neq i}\left[P(x \mid c_k)\right]^2 \tag{24}$$

Second term of this error function adds a greater penalty if the error concentrates in a single class than if the error is uniformly distributed among all other classes. In other words, a single major threat is more likely to cause misclassification than many small threats.

Obviously, one σ for all directions is not a good choice for all cases. If data have anisotropic structure, better results can be achieved with different values of σ for different orientations (directions – σ_j, $j=1,2,\ldots,p$). Such improvement, of course, complicates the training procedure. In practice, result of one σ *k-fold* cross-validation procedure is used as a starting point for further gradient descent optimization. Also method can be modified by applying class-dependent σ values.

2 Case study

2.1. Data description

The real case study deals with the soil types classification in Briansk region, Russia. This is the most contaminated part of Russia by Chernobyl radionuclides. Actually, prediction mapping of environment contamination includes both physico-chemical modelling of radionuclides' migration in environment and spatial data analysis and modelling [Demyanov et al. 1999]. Migration of radionuclides in soil depends on properties of radionuclides, soil types, precipitation, etc. High variability of environmental parameters and initial fallout at different scales highly complicates the solution of the problem.

Influence of soil types on Sr90 radionuclide vertical migration in soil is presented in Figure 4. Detailed information on radionuclides migration and corresponding models and software tools can be found in [Kanevski et al: IBRAE preprint on soil migration].

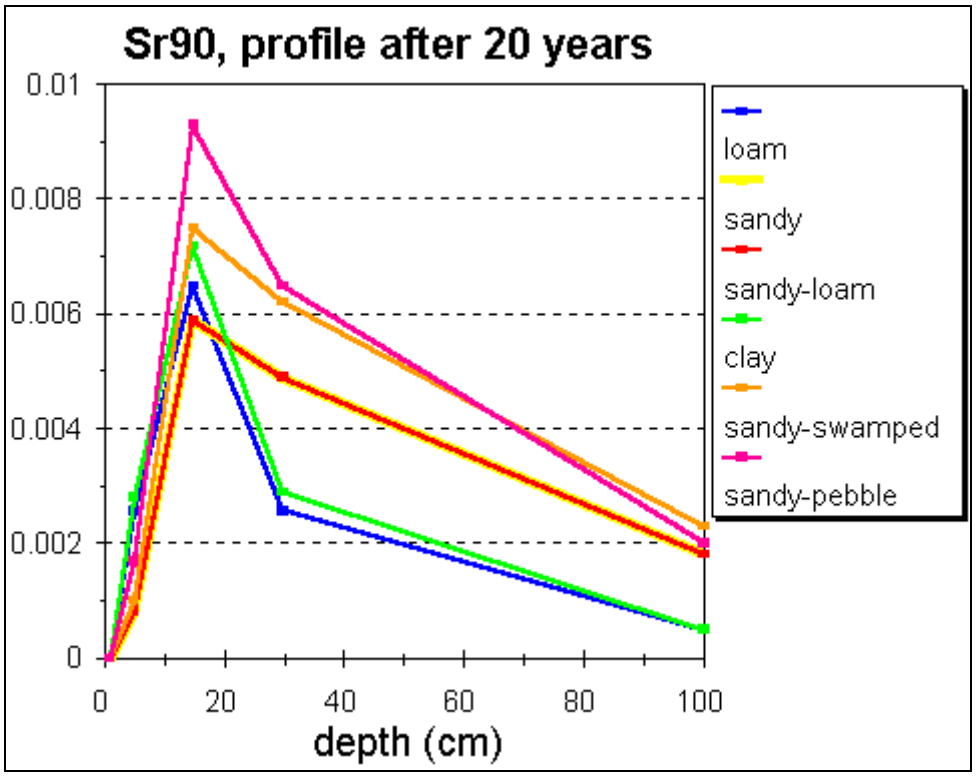


Fig. 4. Vertical distribution of Sr90 radionuclide after 20 years of deposition

Before training all original geographical coordinates were transforned to Lambert map projection and then linearly projected to (-1; 1) segment. All data and results are presented in this coordinate system.

In the region considered there are 5 major soil types. The original data set where soil types have been determined contains 810 sample points. For the comparison of methods it was divided into 2 parts: training and validation data sets. Training set was used for training/developing models. It contains 310 samples distributed rather homogeneously overall the region. Validation set was used for the validation and for the comparison of

obtained results. It contains remaining 500 samples. They also cover all parts of the region under study. All 5 soil types are present in the both data sub-sets. Prediction grid contains 4321 points. Distribution of training, validation and prediction locations are presented in Figure 5.

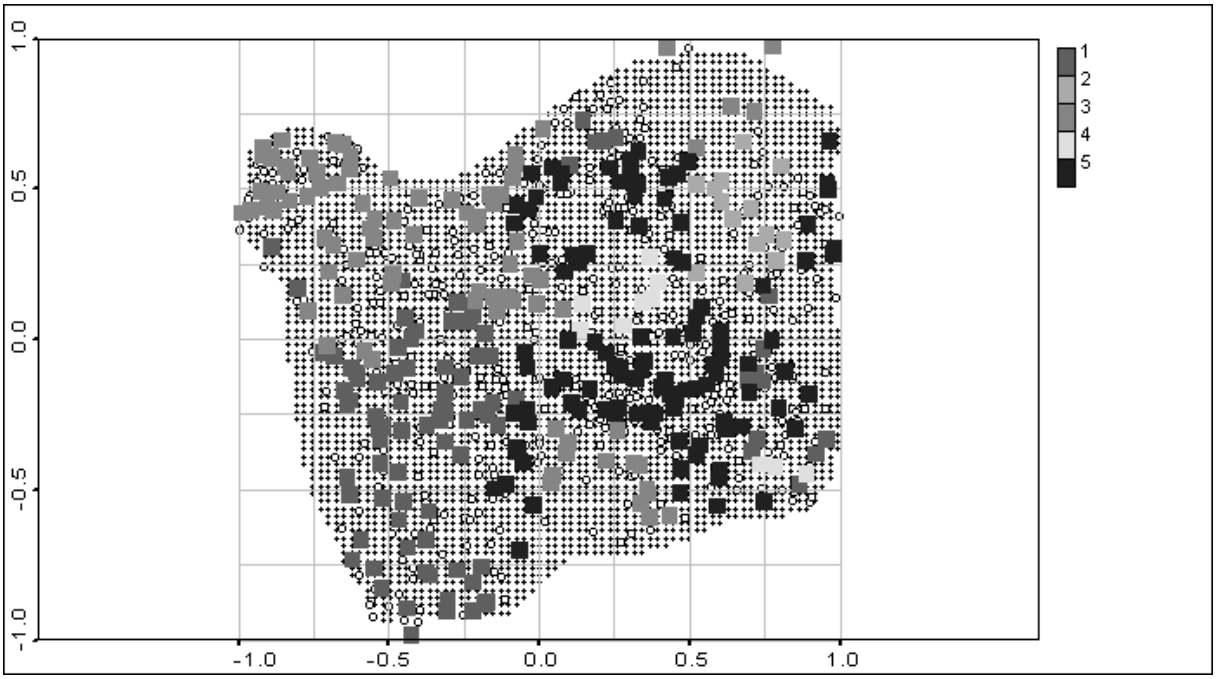


Fig. 5. Raw data on soil types in the Briansk region. Training data set (filled rectangles) and validation data set (circles)

2.2 Variography

The estimation of spatial correlation structure of the data is significant for all kinds of analysis of distributed data. Semivariogram is mostly often used as a measure of spatial correlation:

$$\gamma(\vec{h}) = Var(Z(x+\vec{h}) - Z(x)),$$

where *Var* means the variance for all pairs of samples separated by vector $\vec{h}$, *Z(.)* is a variable under study.
In our case working with categorical data we can't construct the semivariogram of raw data (number of class), as semivariogram can be used for continuous variable (it can be also considered as a measure of continuity). So first indicator transform has to be performed. For each class (soil type) indicator transform is performed using the following formula:

$$I(x,c) = \begin{cases} 1, & C(x) = c \\ 0, & otherwise \end{cases}, \qquad (25)$$

where *c* is a current class and *C(x)* is a class to which belong the position *x*. Results of transformation of training and test data sets are presented in figures 6 - 7.

Indicator can be considered as a continuous variable, it signifies the variability to belong to a class. So semivariograms of indicators constructed for all classes can be estimated. They are estimated using formula (25) where *I(.,c)* is used for *Z(.)*. Raw indicator semivariograms for training and testing data sets are presented in the figures 8 - 9.

For all classes correlation presents the anisotrophic structure. The directions of anisotrophy principal components are various for different classes.

It can be remarked that modelling of indicator variogram for class 4 is complicated. It can be explained by few members of this class. What is also interesting to remark, is that possible variogram models for class 1 constructed on the base of training and testing data sets variograms are different. For all other classes raw variograms for training and testing data sets look rather similar.

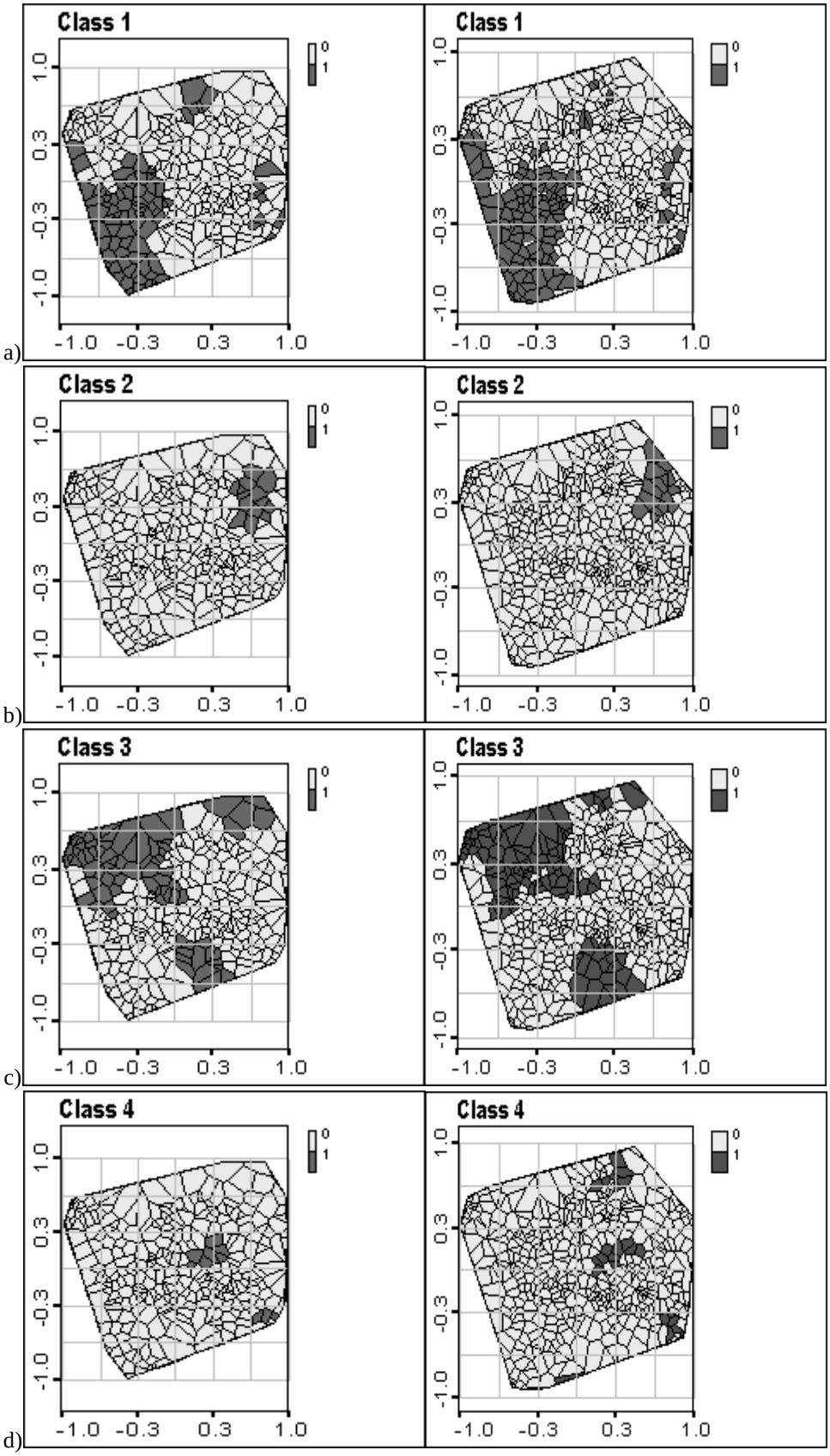


Fig. 6. Indicator transformed training (left) and test (right) data sets (a – class 1, b – class 2, c – class 3, d – class 4)

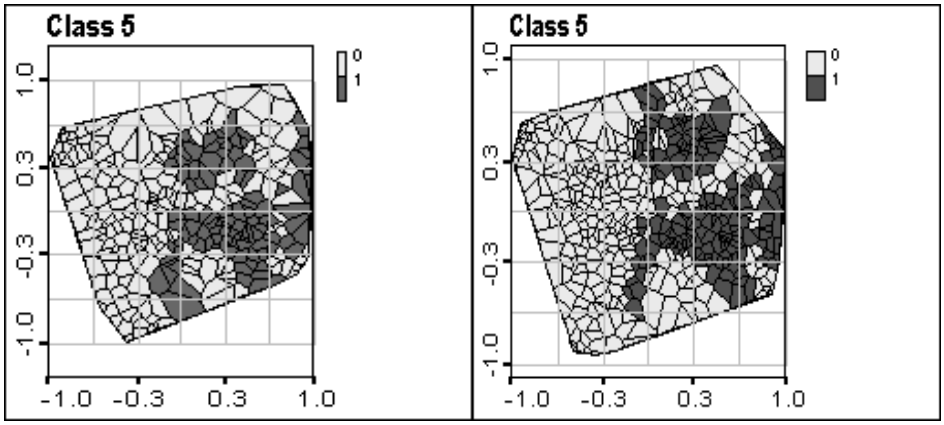
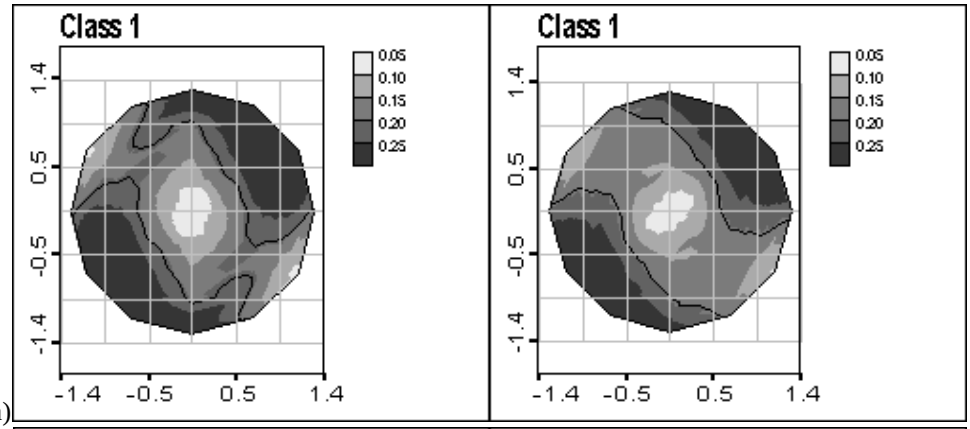
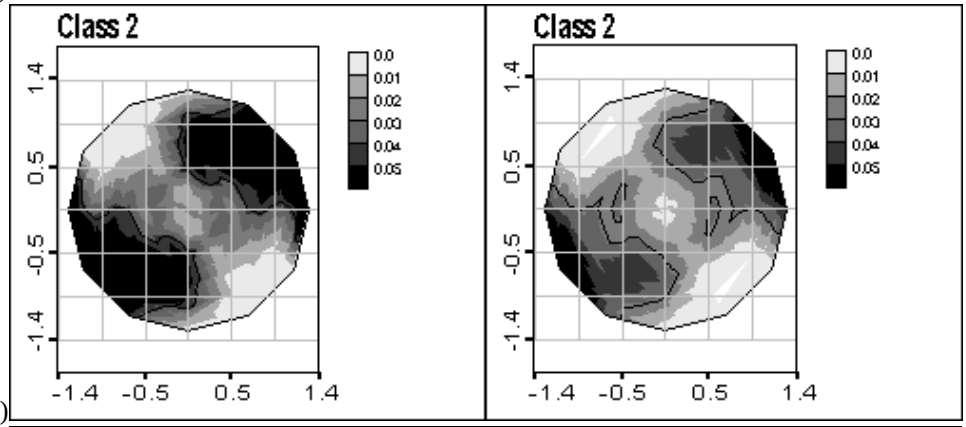


Fig. 7. Indicator transformed training (left) and test (right) data sets for class 5

a)



b)



c)

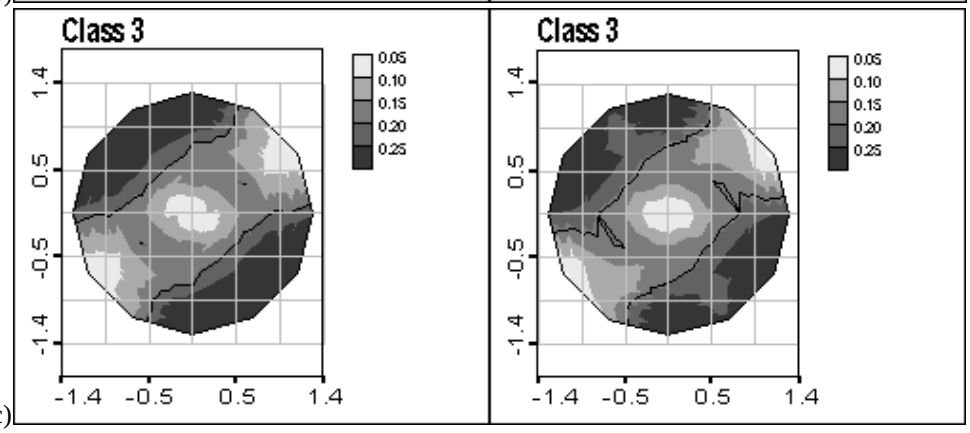


Fig. 8. Indicator variograms for training (left) and test (right) data sets (a – for class 1, b – for class 2, c – for class 3)

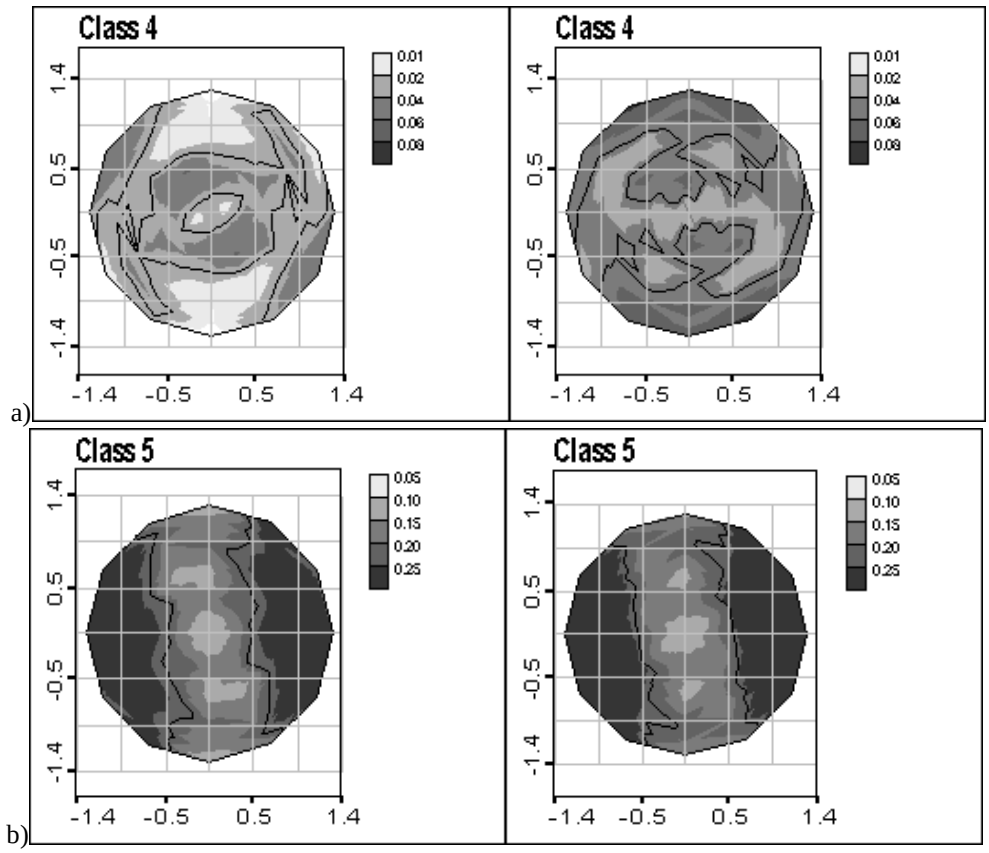


Fig. 9. Indicator variograms for training (left) and test (right) data sets (a – for class 4, b – for class 5)

2.3. Training of the Models

2.3.1. Support Vector Machines

There are several possibilities for the multi-class classification with SVM using binary models: one-to-rest classification, pair-wise classification, direct generalization of the SVM to multi-class problems and others [1],[6],[7]. In this work one-to-rest classification schemes is analysed in detail as it appears that more complicated pair-wise classification did not improve significantly the results

M- models are developing from binary classification by applying the simplest algorithm if using One-to-Rest class-insensitive classification. M-classifiers have the same kernel bandwidths. If classes have different variability at different scales and directions the “optimal kernel bandwidth” characterises some averaged scale of variability. Of course, what is optimal for one class, can be over-fitting or over-smoothing for the others. Class insensitive approach is fast and gives general overview of the problem. In our case it gives satisfactory results.

RBF kernal function was used for constructing the classifier. Kernel approach allows to build very flexible models through the data-dependent adaptation of the hyper-parameters. Figure 10 illustrates different prediction maps, obtained from varying the isotropic RBF kernel bandwidth from small values (0.03) – overfitting to large ones (0.5) – oversmoothing.

There are several strategies to tune the hyper-parameters. One way is to split the whole data set into training and testing subsets and use the testing error curves. Examples of testing error curves are presented in figure 11. Another way is to use the theoretical estimation of the generalization ability. More simplified method, following from theory, is to control the number of the support vectors. In this work method of splitting data into training and testing subsets combined with minimization of the number of support vectors was applied.

In the case of class-adaptive one-to-rest generalization scheme M models (dichotomies) with different optimal kernel bandwidths and C parameter are tuned. Minimal testing error and the minimum number of the support vectors give the same values for the hyper-parameters. The results are shown in Table 1. Such high difference in the values of C can be for example explained by the high difference in the number of class members.

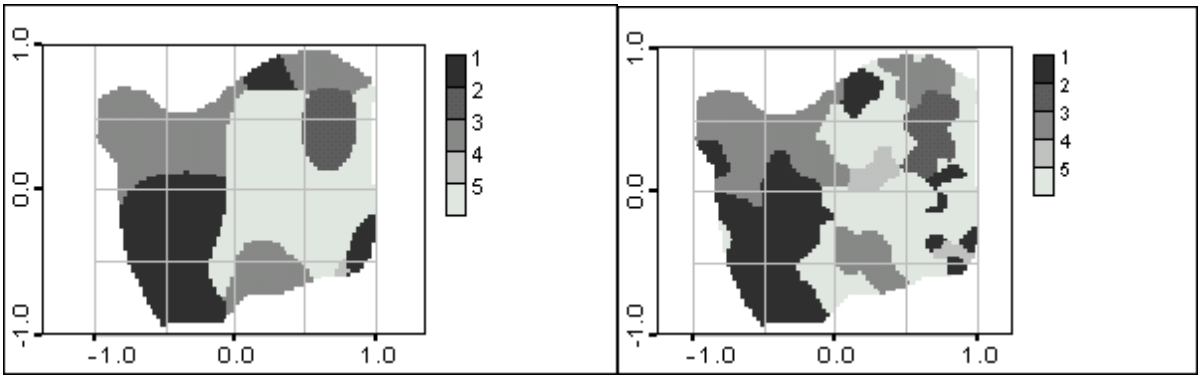


Fig.10. Examples of prediction mapping by SVM with different bandwidth values of RBF kernel. Large bandwidth (0.5) – oversmoothing (left), small bandwidth (0.03) – overfitting (right)

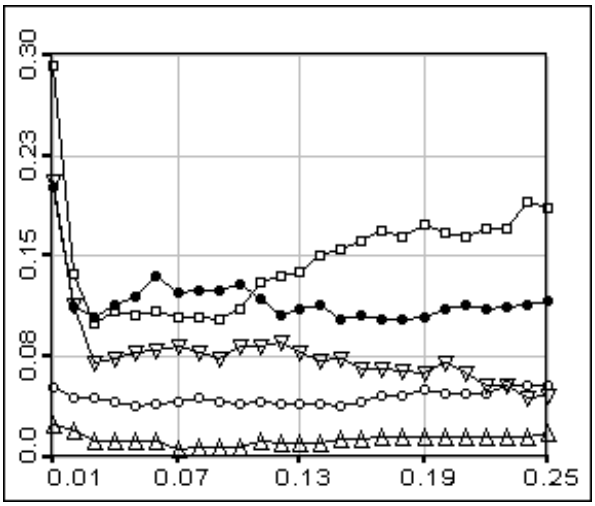


Fig. 11. Testing curves for one-to-rest classifiers. • - class 1, ▽- class 2, Δ - class 3, ○ - class 4, □ - class 5

Table 1. Parametrs of models (dichotomies)

	Sigma	C
Class 1	0.04	100
Class 2	0.07	100
Class 3	0.19	1000
Class 4	0.05	100
Class 5	0.14	1

2.3.2. Probabilistic Neural Network

There was no any additional information concerning probability distribution function of classes, so initial probability was thought to be equal for all classes: $P(c_i)=1/5$ for all $i=1…5$.

Learning procedure to adjust the optimal σ values for each class has been made in three steps:

1. common σ for all classes and all directions was found using leave-one-out cross-validation method (see figure 12).
2. different sigmas were found by the same leave-one-out cross-validation method for X and Y directions (σ_X and σ_Y) – see figure 13. The common σ obtained at the first step was used as a mean of the range where σ_X and σ_Y were searching. Optimal directions X and Y have been found by coordinate system rotating.
3. tuning σ_X and σ_Y for each class. Scaled conjugate gradient algorithm was efficiently used for this purpose. The optimal values of sigmas obtained on training data set is presented in Table 2.

To select a winner between classes the threshold of a probability equal to 0.5 was used.

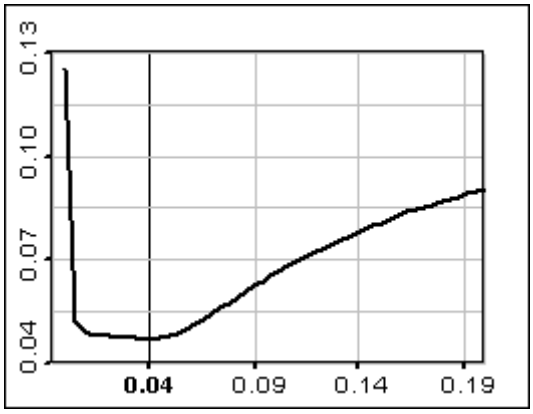


Fig.12. The error curve for adjustment of common σ. The best σ=0.04

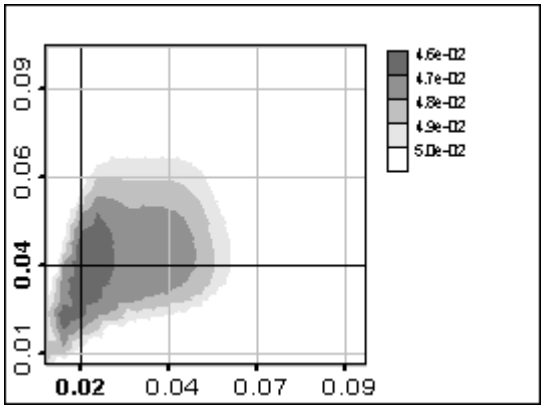


Fig. 13. Cross-validation procedure for adjusting directional σ. The best values are – σ_X= 0.02, σ_Y= 0.04

Table 2. Optimal values of σ for all classes

Direction\Class	1	2	3	4	5
X	0.013	0.018	0.013	0.016	0.015
Y	0.027	0.027	0.026	0.025	0.027

2.3.3 Method with *K* Nearest Neighbours

The simplest version of *K* nearest neighbours method was applied to classify the data and to compare the result with mashine learning algorithms.

This method needs no any training procedure. It is based on the training data set. The procedure of classification is the following:

1. for each point to be classified *K* nearest neighbours from the training data set are found. The nearest is called the point for which the Euclidian distance between it and the point under study is low then for any other from the training data set. If *K>1*, the first found nearest point is excluded from the training data set and the second nearest is looked for. This procedure is repeated *K* times.
2. The competition between classes takes place. The winner is the class that appears more times among the found in the previous step *K* neighbours. If there are several classes winners as a final winner is selected the class whose owner (the sample point) is closer to the point under classification.
3. The point under classification is atributed to the class winner of the previous step.

The method can be modified so to include preferable directions, another types of competitions between several neighbours and so on. In this work the most simple version was used.

2.4. Accuracy Test (Analysis of Training Error)

After all approaches under study have been learnt (found their optimal internal parameters) the accuracy test was performed. All methods were applied to classify the data set they were trained on. All methods perform well on training data. It means that both algorithms learned well data they were learned on. The 1 nearest neighbour and

2 nearest neighbours has zero training errors because of the classification algorithm. Results on misclassification are presented in Table 3. It is seen that using more neighbours in *K* nearest neighbour method doesn't improve the result. So for the following comparison classical nearest neighbour method will be used.

Table 3. Results of accuracy test

Method	Number of misclassified points	Misclassification error (%)
SVM	2	0.65
PNN	4	1.29
3 NNM	20	6.45
4 NNM	14	4.5
5 NNM	23	7.4
6 NNM	19	6.13
7 NNM	29	9.4
8 NNM	22	7.1
9 NNM	34	10.97
10 NNM	32	10.32

SVM and PNN provide suffiecient results on the training data set. As for *K* nearest neighbours it can be seen that in general misclassification error is growing with the number of neighbours used for classification. But 3, 4 and 6 nearest neighbours provide the best results. They can be compared with other methods.

In addition to the classification result PNN provides additional useful information available for each point under classification, such as:

1. Posterior probability to belong to each class.
2. The estimation of the classification error.

Results of such type are presented in figures 14 and 15. They contain posterior probability of the winner class, posterior probability and error of classification for class 1 as an example.

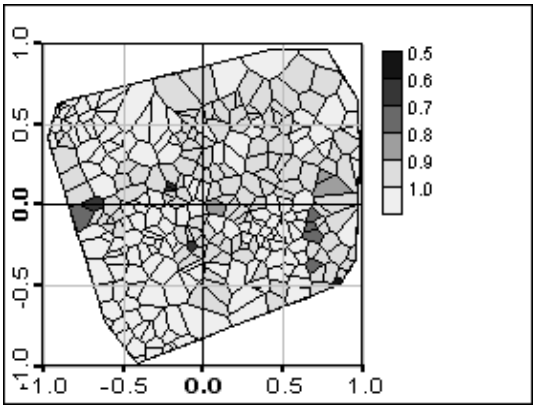


Fig. 14. Result of PNN classification on training data set. Posterior probability of winner class

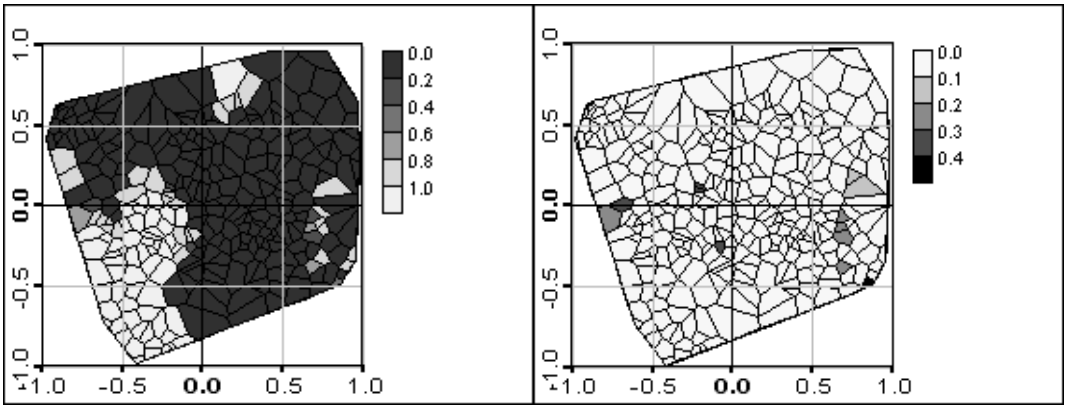


Fig. 15. PNN description of the 1 class. Posterior probabilities (left) and error of prediction (right)

2.5. Validation Data Set Classification

After rather successful passing of the accuracy test the same methods were applied to classify the validation data set. The results on misclassification are collected in Table 4. Tables 5, 6 and 7 present confusion matrix for SVM, PNN and NNM correspondingly. Reject column in PNN confusion matrix is devoted to sample points where posterior probability to belong to each of classes is lower than the threshold value, but according to max probability it was ascribed to a class correctly.

Results presented by all methods on validation data set can be considered as satisfactory. All methods can be used for prediction.

Table 4. Misclassification on the validation data set

Method	Number of misclassified points	Misclassification error (%)
SVM	64	12.8
PNN	91	18.2
NNM	89	17.8
3NNM	93	18.6
4NNM	94	18.8
6NNM	93	18.6

Table 5. Confusion matrix of SVM classification on validation data set

Class		1	2	3	4	5	Error/%
	Sum **500**	141	19	131	12	197	64/12.8%
1	134	***116***	0	**6**	**1**	**11**	18/13.4%
2	15	0	***15***	0	0	0	0/0%
3	131	**9**	0	***117***	0	**5**	14/10.7%
4	25	**7**	**1**	**4**	***10***	**3**	15/60%
5	195	**9**	**3**	**4**	**1**	***178***	17/8.7%

Table 6. Confusion matrix of PNN classification on validation data set

Class		1	2	3	4	5	Error/%
	Sum **500**	143	23	114	19	201	91/18.2
1	134	***111***	0	**6**	**1**	**16**	23/17.16
2	15	0	***13***	**1**	0	**1**	2/13.33
3	131	**17**	0	***103***	**1**	**10**	28/21.37
4	25	**5**	**1**	**1**	***13***	**5**	12/48.00
5	195	**10**	**9**	**3**	**4**	***169***	26/13.33

Table 7. Confusion matrix of NNM classification on validation data set

Class		1	2	3	4	5	Eror/%
	Sum **500**	147	17	117	16	203	89/17.8
1	134	***111***	0	**5**	**1**	**17**	23/17.16%
2	15	0	***13***	**1**	0	**1**	2/13.33%
3	131	**15**	0	***106***	**1**	**9**	25/19.08%
4	25	**9**	0	**1**	***10***	**5**	15/60%
5	195	**12**	**4**	**4**	**4**	***171***	24/12.3%

The results of classification obtained by all three methods are presented in figures 16 and 17.

Results concerning posterior probabilities and classification errors for PNN on validation data set are presented in figures 18 and 19. These figures contain posterior probabilities of the winner class, posterior probability of class 1 and classification error.

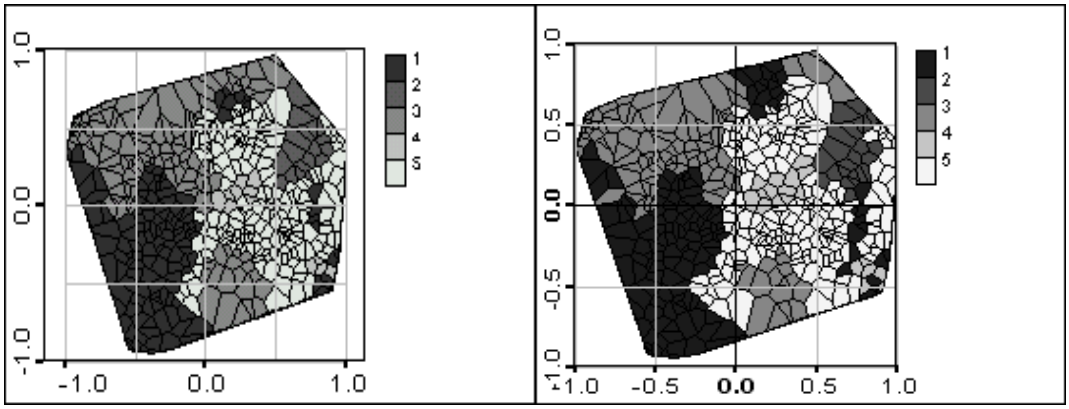


Fig. 16. Results of classification on validation data set. SVM (left), PNN (right)

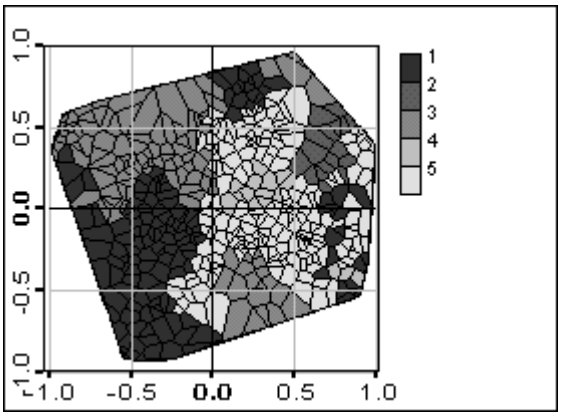


Fig. 17. Results of classification on validation data set by NNM

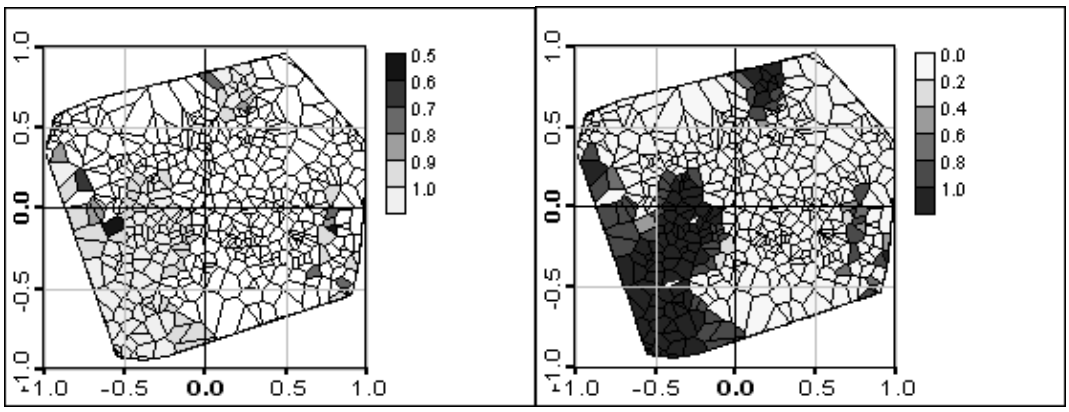


Fig. 18. Posterior probabilities obtained by PNN on validatin data set. The winner class (left), class 1 (right)

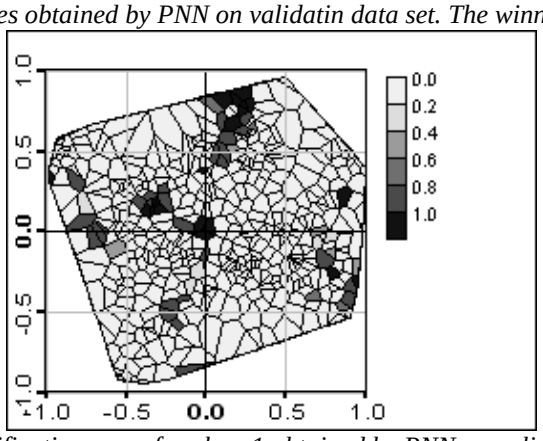


Fig. 19. Classification error for class 1 obtained by PNN on validation data set

2.6. Categorical Data Mapping

At last all methods discussed above were used to predict distribution of soil types on the dense regular grid. For prediction SVM with the generalization of binary model using pair-wise classifications was also used. It was trained using the same method of splitting data into training and testing subsets combined with minimization of the number of support vectors. In this case M(M-1)/2 models were developed. Results obtained by different SVM modifications are presented in figure 20. Results of soil types (classes) prediction by PNN and SVM are presented in figures 21. PNN prediction leaves 12 points as undefined. In these points max probability (the probability of the winner class) is less than the threshold. The worst situation is with the value of max probability equal to 0.4. In figure such points are presented as white points.

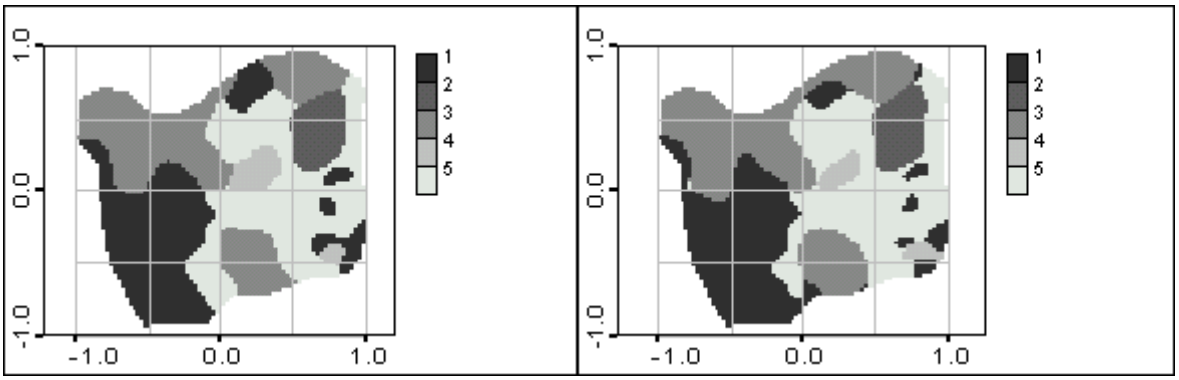


Fig. 20. SVM one-to rest mapping with class-adaptive bandwidths (left) and SVM pair-wise mapping with class adaptive bandwidths (right)

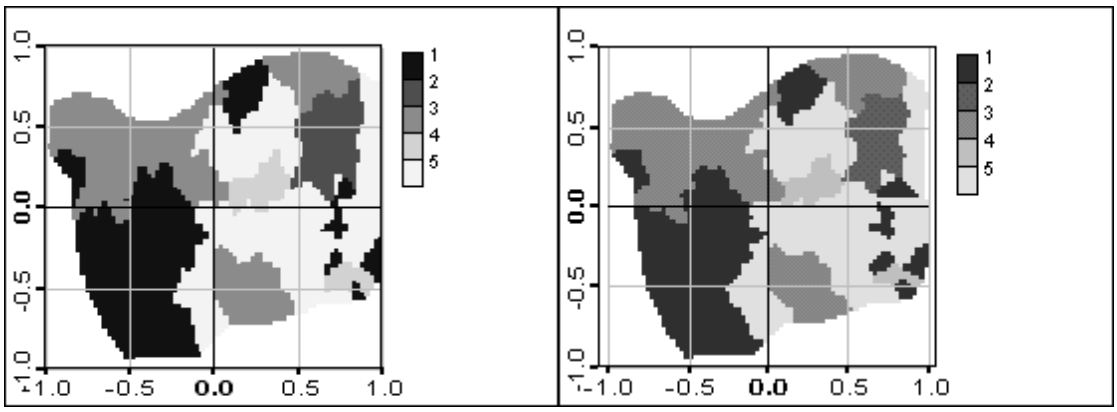


Fig. 21. PNN (left) and NNM (right) prediction on the regular grid

In figure 22 the posterior probabilities obtained by PNN are presented. Posterior probabilities of the winner class can be considered as a good ilustration of decision boundaries (see fig. 22 left). Posterior probabilities can be preseneted for each of 5 classes, class 1 (see fig. 22 right) was selected as an example.

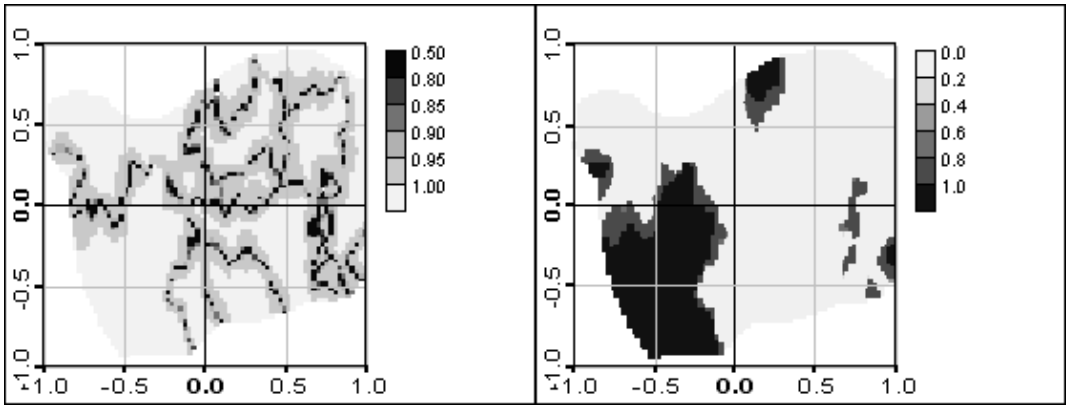


Fig. 22. PNN estimated posterior probabilities of winner class (left) and of class 1 (right)

Conclusions

Multi-class problems was investigated using real data on soil types. The following conclusions can be made on the base of performed analysis:

- Kernel based methods are promising tools for environmental data classification;
- Nearest neighbor is useful for preliminary analysis and fast classification;
- learning based on SVM approach is an efficient algorithm for the spatial data classification. In the current work Several models generalising 2 class SVMs for multiclass task were applied: Class-insensitive, Class adaptive and Pair-wise classification. Some schemes can take into account rather complicated spatial structures of different classes. In the current case study SVM with class adaptive generalization scheme provided the best result on the validation data set.
- PNN provides very important additional information – probabilistic model (class probabilities). It can be used for description of uncertainty of classification. Also PNN allows to take into account a priori information – size of class members, for example.

The further developments will be directed to adapt SVM for probabilistic description. Also some more experiments both with simulated and real data have to be carried out for better understanding of SVMand PNN application to spatial data and their behaviour.

Acknowledgements

The work was partly supported by INTAS grant 99-00099, crdf grant RG2-2236 and ISTC 1224 .

References

[1] Vapnik V. Statistical Learning Theory. John Wiley & Sons, 1998.

[2] Cristianini N. and Shawe-Taylor J. An Introduction to Support Vector Machines and other kernel-based learning methods. Cambridge University Press, 2000 189 pp.

[3] Burgess C. A tutorial on Support Vector Machines for pattern recognition. Data mining and knowledge discovery, 1998.

[4] Kanevski M., N. Gilardi, M. Maignan, E. Mayoraz. Environmental Spatial Data Classification with Support Vector Machines. IDIAP Research Report. IDIAP-RR-99-07, 24 p., 1999a. (www.idiap.ch)

[5] N Gilardi, M Kanevski, E Mayoraz, M Maignan. Spatial Data Classification with Support Vector Machines. Accepted for Geostat 2000 congress. South Africa, April 2000.

[6] Weston J., Watkins C. Multi-class Support Vector Machines. Technical Report CSD-TR-98-04, 9p, 1998.

[7] E. Mayoraz and E. Alpaydin Support Vector Machine for Multiclass Classification, , IDIAP-RR 98-06, 1998 (www.idiap.ch)

[8] Parzen, E. (1962). "*On Estimation of a Probability Density Function and Mode*." Annals of Mathematical Statistics, **33**: 1065-1076.

[9] Specht, Donald (1990). "*Probabilistic Neural Networks.*" Neural Networks, **3**: 109-118.

[10] M. Kanevski, N. Koptelova, V. Demyanov. RamisW - Software for Modelling Migration of Radionuclides in Soil. Institute of Nuclear Safety (IBRAE). Preprint IBRAE 97-16, Moscow, 1997, 21 p.