Discovering Data Structures using Meta-learning, Visualization and Constructive Neural Networks

Tomasz Maszczyk, Marek Grochowski, Włodzisław Duch

In: Advances in Machine Learning II, Springer Studies in Computational Intelligence,
Vol. 262, pp. 467-484.
Eds. Koronacki J, Ras Z.W, Wierzchon S.T, Kacprzyk J.

Abstract Visualization methods are used to discover simplest data transformations implemented by constructive neural networks, revealing hidden data structures. In this way meta-learning, based on search for simplest models in the space of all data transformations, is facilitated.

Key words: Meta-learning, constructive neural networks, projection pursuit, visualization

1 Introduction

Ryszard Michalski has always been interested in discovering comprehensible structures in the data, and has developed his own multistrategy learning approach [10]. In this spirit we shall look at the problem of finding the best set of transformations to solve classification and regression problems, searching for such transformations that will reveal the inherent structure in the data. Instead of trying to solve all problems with the same universal tool this approach leads to a meta-learning scheme [8], building the final model from components that are heterogeneous [7].

Each data model depends on some specific assumptions about the data distribution in the input space, and is successfully applicable only to some types of problems. For example SVM and many other statistical learning methods [21] rely on the assumption of uniform resolution, local similarity between data samples, and may completely fail in case of high-dimensional functions that are not sufficiently smooth [2]. In such case accurate solution may require an extremely large number of training samples that will be used as reference vectors, leading to high cost of computations and creating complex models.

Department of Informatics, Nicolaus Copernicus University, Toruń, Poland,
e-mail: tmaszczyk@is.umk.pl; grochu@is.umk.pl; Google: W Duch

The type of the solution offered by given data models may not be appropriate for the particular data. Each data model defines a hypotheses space, that is a set of functions that this model may easily learn (the bias of model). Although many basis set expansions (such as the multilayer perceptron neural networks, MLPs) are universal approximators, in the sense that given sufficient number of functions they may approximate arbitrary data distributions, models created in this way are not necessarily optimal from the complexity point of view. Linear discrimination models are obviously not suitable for spherical distributions of data, requiring $O(N^2)$ parameters to approximately cover each spherical distribution in N dimensions, where an expansion in radial functions requires only $O(N)$ parameters. On the other hand many spherical functions are needed to approximate a hyperplane. Some problems, such as the multidimensional parity problem, cannot be easily approximated by neither by radial nor by hyperplane functions [6]. An optimal solution may only be found if a model based on suitable transformations is defined.

In general each supervised learning machine may be represented by an operator $\mathscr{T}$ that transforms a given vector $\mathbf{X}$ into some output vector $\mathbf{Y}$

$$\mathscr{T}\mathbf{X} = \mathbf{Y}$$

The goal is to find an operator $\mathscr{T}$ that not only gives correct answers on the training data but also provides a model that has low Kolmogorov complexity, facilitating easy interpretation. Operator $\mathscr{T}$ may in general be created as a sequence of transformations (for simplicity recurrent processes are not considered here):

$$\mathscr{T} = \mathscr{T}_1\mathscr{T}_2\ldots\mathscr{T}_k$$

For example, initial transformations may define data preprocessing, based on linear scaling (standardization or normalization), projection on a low-dimensional space defined by principal components or other criteria, or a simple feature selection. The final transformation should provide desired output; for classification tasks this output Y should provide discrete values of class labels. This representation covers also combinations of many learning algorithms (classifier committees, boosting). Each layer of the MLP network can be seen as a single mapping $\mathscr{T}_i$, although taken separately these transformations do not have a well-defined goal. Data preparation may have a crucial influence on algorithms applicable for further training. For example, some computationally expensive algorithms require dimensionality reduction to deal with large datasets. Therefore model selection cannot focus only on searching for suitable classifier or regression tool, selection of supportive transformations is also very important.

With no *a priori* knowledge about a given problem finding of optimal sequence of transformations is a great challenge. Many meta-learning techniques have recently been developed to deal with the problem of model selection [22, 3]. Most of them search for optimal model characterizing a given problem by some meta-features (e.g. statistical properties, landmarking, model-based characterization), and by referring to some meta-knowledge gained earlier. For example, one can use the classifier that gave the best result on a similar dataset in the StatLog Project [18].

However, choosing good meta-features is not a trivial issue as most of these features do not characterize the complexity of data distributions. In addition the space of possible solutions generated by this approach is bounded to already known types of algorithms. General meta-learning algorithm should browse through all interesting models, searching for the best composition of transformations. Thus the meta-learning problem may be seen as a search in the space of all possible models, for example all transformations in the similarity-based approach [8]. General search in model space requires a very sophisticated intelligent system and search strategies. Some ideas for building such systems and for controlling the process of composing transformations based on observation of machine complexity have recently been proposed in [13, 14].

Proper transformation performed at the beginning of learning does not only lead to simplification of further learning but also can provide useful informations guiding researcher through the set of possible models. Based on structures emerging in low-dimensional visualizations of a given problem an experienced researcher is able to construct the best learning strategies to learn the data. In this paper such approach to meta-learning has been tested. In the next section a few dimensionality reduction algorithms suitable for visualization are presented and in the third section applied to the real and artificial data. Upon visual inspection it becomes quite clear which type of transformation should be applied to the data to create it simplest model.

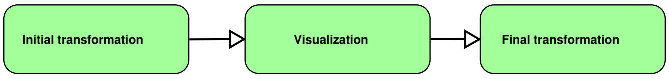

Fig. 1 After initial transformation and visualization the final transformation is selected.

2 Visualization algorithms

Visualization methods are discussed in details in many books, for example [20, 23]. Below a short description of three popular visualization methods is given: principal component analysis (PCA), Fisher discriminant analysis (FDA), and multidimensional scaling (MDS), followed by description of our two new approaches based on the SVM [17] and Quality of Projected Clusters (QPC) algorithms [16]. Only MDS is a non-linear method, and only PCA and MDS are unsupervised.

Principal Component Analysis (PCA) is a linear projection method that finds orthogonal combinations of input features $\mathbf{X} = \{x_1, x_2, ..., x_N\}$, with each new direction accounting for largest remaining variation in the data. This method is unsupervised, therefore PCA transformation may be done on all available data. Principal components $\mathbf{P}_i$ that result from diagonalization of data covariance matrix guarantee minimal loss of information when position of points are recreated from their low-dimensional projections. Taking 1, 2 or 3 principal components and projecting the data into the space defined by these components $y_{ij} = \mathbf{P}_i \cdot \mathbf{X}_j$ provides for each input vector its representative $\left(y_{1j}, y_{2j}, ...y_{kj}\right)$ in the target space.

Fisher Discriminant Analysis (FDA) is a supervised method that uses information about the classes to find projections that separate data from different classes. This popular algorithm maximizes the ratio of between-class to within-class scatter, seeking a direction $\mathbf{W}$ such that

$$\max_{\mathbf{W}} J_{\mathbf{W}} = \frac{\mathbf{W}^T \mathbf{S}_B \mathbf{W}}{\mathbf{W}^T \mathbf{S}_I \mathbf{W}} \tag{1}$$

where the scatter matrices $\mathbf{S}_B$ and $\mathbf{S}_I$ are defined by

$$\mathbf{S}_B = \sum_{i=1}^{C} \frac{n_i}{n} (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T; \qquad S_I = \sum_{i=1}^{C} \frac{n_i}{n} \hat{\Sigma}_i \tag{2}$$

where $\mathbf{m}_i$ and $\hat{\Sigma}_i$ are the means and covariance matrices for each class and $\mathbf{m}$ is the total mean vector [23]. FDA is frequently used for classification and projecting data on a line. For visualization generating the second FDA vector in a two-class problem is not so trivial. This is due to the fact that the rank of the $\mathbf{S}_B$ matrix for the C-class problems is $C-1$. Cheng *et al.* [4] proposed several solutions to this problem:

- stabilize the $\mathbf{S}_I$ matrix by adding a small perturbation matrix;
- use pseudoinverse, replacing S_I^{-1} by the pseudoinverse matrix $S_I^{\dagger}$;
- use rank decomposition method.

In our implementation pseudoinverse matrix has been used to generate higher FDA directions.

Multidimensional scaling (MDS) is a non-linear technique used for proximity visualization [5]. The main idea is to decrease dimensionality of data while preserving original distances between data points as defined in the high-dimensional space. MDS methods need only similarities between objects, so explicit vector representation of objects is not necessary. In metric scaling quantitative evaluation of similarity based on numerical distance measures (Euclidean, cosine, or any other measure) is used, while for non-metric scaling qualitative information about the pairwise similarities is sufficient. MDS methods also differ by their cost functions, optimization algorithms, the number of similarity matrices used, and the use of feature weighting. There are many measures of topographical distortions due to the reduction of dimensionality, most of them variants of the stress function:

$$S_T(\mathbf{d}) = \sum_{i>j}^{n} (D_{ij} - d_{ij})^2 \tag{3}$$

where d_{ij} are distances (dissimilarities) in the target (low-dimensional) space, and D_{ij} are distances in the input space, pre-processed or calculated directly using some metric functions. These measures are minimized over positions of all target points, with large distances dominating in the $S_T(\mathbf{d})$. The sum runs over all pairs of vectors and thus contributes $O(n^2)$ terms. In the k-dimensional target space there are kn parameters for minimization. For visualization purposes the dimension of the target space is $k = 1-3$. The number of vectors n may be quite large, making the approximation to the minimization process necessary [19]. MDS cost functions are not easy to minimize, with multiple local minima representing different mappings. Initial configuration is either selected randomly or based on projection of data to the space spanned by principal components. Orientation of axes in the MDS mapping is arbitrary, and the values of coordinates do not have any simple interpretation, as only relative distances are important. However, if the data has clear clusters in the input space MDS may show it.

Linear SVM algorithm searches for a hyperplane that provides a large margin of classification, using regularization term and quadratic programming. Non-linear versions are based on a kernel trick [21] that implicitly maps data vectors to a high-dimensional feature space where the best separating hyperplane (the maximum margin hyperplane) is constructed. Linear discriminant function is defined by:

$$g_{\mathbf{W}}(\mathbf{X}) = \mathbf{W}^T \cdot \mathbf{X} + w_0 \tag{4}$$

The best discriminating hyperplane should maximize the distance between decision hyperplane defined by $g_{\mathbf{W}}(\mathbf{X}) = 0$ and the vectors that are nearest to it, $\max_{\mathbf{W}} D(\mathbf{W}, \mathbf{X}^{(i)})$. The largest classification margin is obtained from minimization of the norm $||\mathbf{W}||^2$ with constraints:

$$Y^{(i)} g_{\mathbf{W}}(\mathbf{X}^{(i)}) \geq 1 \tag{5}$$

for all training vectors $\mathbf{X}^{(i)}$ that belong to class $Y^{(i)}$. Vector $\mathbf{W}$, orthogonal to the discriminant hyperplane, defines direction on which data vectors are projected, and thus may be used for one-dimensional projections. The same may be done using non-linear SVM based on kernel discriminant:

$$g_{\mathbf{W}}(\mathbf{X}) = \sum_{i=1}^{N_{sv}} \alpha_i K(\mathbf{X}^{(i)}, \mathbf{X}) + w_0 \tag{6}$$

where the summation is over support vectors $\mathbf{X}^{(i)}$ that are selected from the training set. The $x = g_{\mathbf{W}}(\mathbf{X})$ values for different classes may be smoothed and displayed as a histogram, estimating either the $p(x|C)$ class-conditionals or posterior probabilities $p(C|x) = p(x|C)p(C)/p(x)$.

SVM visualization in more than one dimension requires generation of additional discriminating directions. The first projection should give $g_{\mathbf{W}_1}(\mathbf{X}) < 0$ for vectors from the first class, and > 0 for the second class. This is obviously possible only for linearly separable data. If this is not the case, a subset $\mathscr{D}(\mathbf{W}_1)$ of all vectors that have projections in the $[a(\mathbf{W}_1), b(\mathbf{W}_1)]$ interval containing the zero point is selected. This interval should include all vectors for which $p(x|C_i)$ class-conditionals overlap. The second best direction may be obtained by repeating SVM calculations in the space orthogonalized to the already obtained $\mathbf{W}_1$ direction, using only the subset of $\mathscr{D}(\mathbf{W}_1)$ vectors, as the remaining vectors are already well separated in the first dimension. SVM training in its final phase is using anyway mainly support vectors that belong to this subset. However, vectors in the $[a(\mathbf{W}_1), b(\mathbf{W}_1)]$ interval should not include outliers that are far from the decision border, and therefore should generate a significantly different direction. This process may be repeated to obtain more dimensions. Each additional dimension should help to decrease errors, and the optimal dimensionality is obtained when new dimensions stop decreasing the number of errors in crossvalidation tests.

In the case of non-linear kernel, $g_{\mathbf{W}}(\mathbf{X})$ provides the first direction, while the second direction may be generated in several ways. The simplest approach is to repeat training on $\mathscr{D}(\mathbf{W})$ subset of vectors that are close to the hyperplane in the extended space using some other kernel, for example a linear kernel.

The Quality of Projected Clusters (QPC) is a supervised projection pursuit method which search for most interesting and informative linear projections by maximalization of following index [16]:

$$QPC(\mathbf{w}) = \sum_{\mathbf{x}} \left(A^{+} \sum_{\mathbf{x}_k \in \mathscr{C}_{\mathbf{x}}} G\left(\mathbf{w}^T(\mathbf{x}-\mathbf{x}_k)\right) - A^{-} \sum_{\mathbf{x}_k \notin \mathscr{C}_{\mathbf{x}}} G\left(\mathbf{w}^T(\mathbf{x}-\mathbf{x}_k)\right) \right) \tag{7}$$

where $G(x)$ is a function with localized support and maximum in $x = 0$ (e.g. a Gaussian function), and $\mathscr{C}_{\mathbf{x}}$ denote the set of all vectors that have the same label as $\mathbf{x}$. This index achieves maximum value for projections on the direction $\mathbf{w}$ that group vectors belonging to the same class into a compact and well separated clusters. It does not enforce linear separability and is suitable for multi-modal data. Parameters A^{+}, A^{-} control influence of each term in Eq. (7) and for large value of A^{-} strong separation between classes is enforced, while increasing of A^{+} impacts mostly compactness and purity of clusters. The shape and width of the $G(x)$ function used in E.q. (7) influent on learning convergence, if $G(X)$ is continuous then gradient-based methods may be used to maximize QPC index.

Data visualizations presented in next section was obtained by maximization of QPC with an inverse quartic function

$$G(x) = \frac{1}{1+(bx)^4} \tag{8}$$

but all bell-shaped functions (e.g. Gaussian or bicentral function) that achieve maximum value for $x = 0$ and vanish for $x \to \pm\infty$ are suitable here. The QPC index provides a leave-one-out estimator that measures quality of clusters projected on $\mathbf{w}$ direction, thus direct calculation of the QPC index (7), as in the case of nearest neighbor methods, requires $O(n^2)$ operations. The greatest advantage of using this index is that it is able to discover non-local structures and multimodal class distributions (e.g. k-separable datasets with $k > 2$ [6]). The QPC may be used also (in the same way as the SVM approach described above) as a base for creation of feature ranking and feature selection methods where the strength of coefficients w_i in the final projection indicates significance of i-th feature. For noisy and non-informative variables values of associated weights should decrease to zero during QPC optimization.

Not only global, but also local extrema of the QPC index are of interest, as they may also provide useful insight into the structure of data and may be used in a committee-based approach that combines different views on the data. For complex problems usually more than one projection is required therefore one can search for the next linear projection either in the orthogonalized space, or using additional penalty term:

$$QPC(\mathbf{w};\mathbf{w}_1) = QPC(\mathbf{w}) - \lambda f(\mathbf{w},\mathbf{w}_1)\ . \tag{9}$$

This term should provide punishment for solutions similar to those found earlier, thus the value of $f(\mathbf{w},\mathbf{w}_1)$ should become large for direction $\mathbf{w}$ close to the previous direction $\mathbf{w}_1$, e.g. some power of the scalar product between these directions may be used $f(\mathbf{w},\mathbf{w}_1) = (\mathbf{w}_1{}^T \cdot \mathbf{w})^2$. Repeating this procedure leads to a creation of sequence of unique interesting projections [16].

3 Illustrative examples

Visualization methods described above will be used to determine what kind of transformations should be used to discover structures hidden in the multidimensional data. The usefulness of this meta-learning approach has been tested on a several datasets, one artificial binary dataset, and four real datasets downloaded from the UCI Machine Learning Repository [1], and an microarray gene expression data from [11]. A summary of these datasets is presented in Tab. 1; their short description follows:

1. **Parity_8:** 8-bit parity dataset (8 binary features and 256 vectors).
2. **Heart** disease dataset consists of 270 samples, each described by 13 attributes, 150 cases labeled as the "absence", and 120 to the "presence of heart disease".
3. **Wisconsin** breast cancer data [24] contains 699 samples collected by doing biopsies on patients. Among them, 458 biopsies have been labeled as "benign", and 241 as "malignant". Feature 6 has 16 missing values, removing corresponding vectors leaves 683 examples.
4. **Leukemia:** microarray gene expressions for two types of leukemia (ALL and AML), with a total of 47 ALL and 25 AML samples measured with 7129 probes [11]. Visualization and evaluations of this data is based here on pre-selected 100 best features, done by simple feature ranking using FDA index.
5. **Monks_1:** dataset contains 124 cases, where 62 samples belong to the first class, and the remaining 62 to the second. Each sample is described by 6 attributes. Logical function has been used to created class labels.

Title	#Features	#Samples	#Samples per class		Source
Parity_8	8	256	128 C_0	128 C_1	artificial
Heart	13	270	150 "absence"	120 "presence"	[1]
Wisconsin	10	683	444 "benign"	239 "malignant"	[24]
Leukemia	100	72	47 "ALL"	25 "AML"	[11]
Monks_1	6	124	62 C_0	62 C_1	[1]

Table 1 Summary of used datasets

For each dataset one and two-dimensional mappings have been created using MDS, PCA, FDA, SVM and QPC algorithms (Figs. 2-6). Then, on the basis of these visualizations, an optimal classificator that should fit to the particular data distribution has been chosen from the following list:

1. Naive Bayesian Classifier (NBC)
2. k-Nearest Neighbours (kNN)
3. Separability Split Value Tree (SSV) [12]
4. Support Vector Machines with Linear Kernel (SVML)
5. Support Vector Machines with Gaussian Kernel (SVMG)

Some methods before classification require standardization of data. Analyzing visualizations we have tried to choose the best classifier for each dataset. To check if this choice has been optimal comparison with classification accuracies for each dataset using all classifiers listed above has been done, in the original as well as reduced one and two-dimensional space. 10-fold crossvalidation tests results are collected in Tables 2-6, with accuracies and standard deviations given for each dataset. For the kNN classifier the number of nearest neighbours has been automatically selected from the $1-10$ range using crossvalidation estimation. Also the SVM parameters C and σ have been chosen in an automatic way using crossvalidation estimations. All calculations have been performed using the Ghostminer package developed in our group [9].

In all cases visualization helps to estimate reliability of predictions for individual cases, showing how far they are from the decision border where strong overlaps occurs. This is very important in many applications, and may be useful also in cases of complex decision borders, when simple rule-based systems will not be able to provide good approximation.

High-dimensional parity problem is very difficult for most classification methods. Many papers have been published on special neural models for parity functions, and the reason is quite obvious, as Fig. 2 illustrates: linear separation cannot be easily achieved because this is a k-separable problem that should be separated into $n+1$ intervals for n bits [15, 6]. MDS is completely lost and does not show any interesting structure, as all vectors from one class have their nearest neighbors from the opposite class. PCA and SVM find a very useful projection direction $[1,1..1]$, but the second direction does not help at all. FDA shows significant overlaps for projection on the first direction.

Only the QPC index finds both directions that are quite useful. Points that are in small clusters projected on the first direction belong to a quite large cluster projected on the second direction. This is a very interesting example showing that visualization may help to solve a difficult problem in a perfect way even when almost all classifiers fail. Looking at these pictures one can notice that because data are not linearly separable, probably the best classifier to solve this problem should be:

- any decision tree, after transformation to one or two-dimensions by PCA, SVM or QPC;
- NBC, in one or two-dimensions, combining directions for the most robust solution, provided that it will use density estimation based on a sum of Gaussian functions or similar localized kernels;

- kNN on the 1D data reduced by PCA, SVM or QPC, with k=1, although it will make a small error for the two extreme points.

This choice agrees with the results from tables 2-6, where the highest accuracy (99.61 ± 1.21) is obtained by the SSV classifier on the 2D data transformed by SVM or QPC method, but NBC and kNN results are not worse from the statistical point of view (are within standard deviation). kNN results on original data with k$\leq$ 10 are always wrong, as 8 closest neighbors are from the opposite class. After dimensionality reduction k=1 is sufficient. It is also interesting to note the complexity of other models: SVM takes all 256 vectors as support vectors, achieving results around the base rate of 50%, with the exception of SVM with Gaussian kernel that does quite well on the one and two-dimensional data reduced by SVM and QPC projections. SSV creates moderately complex tree with 7 leaves and a total of 13 nodes.

Visualization in Fig. 2 also suggest that using 2D QPC projected data kNN rule may be easily modified: instead of a fixed number of neighbors for vector $\mathbf{X}$, take its projections y_1, y_2 on the two dimensions, and count the number of neighbors $k_i(\varepsilon_i)$ in the largest interval $y_i \pm \varepsilon_i$ around y_i that contain vectors from a single class only, summing results from both dimensions $k_1(\varepsilon_1) + k_2(\varepsilon_2)$. This is an interesting new version of the kNN method, but it will not be explored here further.

Cleveland Heart data Fig. 3 is rather typical for biomedical data. The information contained in the test data is not really sufficient to make a perfect diagnosis. Almost all projections show comparable separation of a significant portion of the data, although looking at probability distributions in one dimensions SVM and FDA seem to have a bit of an advantage. In such case strong regularization is advised to improve generalization. For kNN this means rather large number of neighbors (in most cases 10, the maximum allowed here, was optimal), for decision tree strong pruning (SSV after FDA has only a root and two leaves), while for SVM rather large value of C parameter and (for Gaussian kernels) dispersions, with significant number of support vectors.

The best recommendation for this dataset is to apply the simplest classifier – SSV or linear SVM on FDA projected data. Comparing this recommendation with calculations presented in tables 2-6 confirms that this is the best choice.

Wisconsin breast cancer dataset is similar to Cleveland Heart data, but it shows much stronger separation (Fig. 4) of the cases that belong to the two classes for all types of visualization. It is quite likely that this data contains several outliers. All methods give comparable results, although reduction of dimensionality to two dimensions helps quite a bit to reduce the complexity of the data models, except for the SVM that achieves essentially the same accuracy requiring similar number of support vectors for the original and the reduced data.

Again, the simplest classifier is quite sufficient here, SSV on FDA or QPC projections with a single threshold (a tree with just two leaves), or more complex (about 50 support vectors) SVM model with linear kernel on 2D data reduced by linear projection. One should not expect that more information can be extracted from this data.

Leukemia shows remarkable separation using both one and two-dimensional QPC, SVM and FDA projections (Fig. 5), providing more interesting solution than

MDS or PCA methods. Choosing one of the three linear transformations (for example the QPC), and projecting original data to the one-dimensional space, SSV decision tree, kNN, NBC and SVM classifiers, give 100% accuracy in the 10CV tests (Table 5). All these models are very simple, with k=1, or trees with 3 nodes, and 2 support vectors for linear SVM. Results on the whole data are much worse than on the projected features.

In this case dimensionality reduction is the most important factor, combining the activity of many genes into a single profile. As the projection coefficients are linear the importance of each gene in this profile may be easily evaluated.

	# Features	Parity_8	Heart	Wisconsin	Leukemia	Monks1
PCA	1	99.21±1.65	80.74±6.24	97.36±2.27	98.57±4.51	56.98±14.12
PCA	2	99.23±1.62	78.88±10.91	96.18±2.95	98.57±4.51	54.67±13.93
MDS	1	38.35±7.00	75.55±6.80	96.63±1.95	92.85±7.52	67.94±11.24
MDS	2	30.49±13.79	80.74±9.36	95.16±1.70	98.57±4.51	63.52±16.02
FDA	1	75.84±10.63	85.18±9.07	97.07±0.97	100±0.00	72.05±12.03
FDA	2	74.56±10.69	84.07±8.01	95.46±1.89	100±0.00	64.48±17.54
SVM	1	99.23±1.62	85.92±6.93	95.90±1.64	100±0.00	70.38±10.73
SVM	2	99.21±1.65	83.70±5.57	97.21±1.89	100±0.00	71.79±8.78
QPC	1	99.20±2.52	81.48±6.53	96.33±3.12	100±0.00	72.56±9.70
QPC	2	98.41±2.04	84.44±7.96	97.21±2.44	100±0.00	100±0
	ALL	23.38±6.74	72.22±4.70	95.46±2.77	78.28±13.55	69.35±16.54

Table 2 NBC 10-fold crossvalidation accuracy for datasets with reduced features.

	# Features	Parity_8	Heart	Wisconsin	Leukemia	Monks1
PCA	1	99.20±1.68 (1)	75.92±9.44 (10)	96.92±1.61 (7)	98.57±4.51 (2)	53.97±15.61 (8)
PCA	2	99.21±1.65 (1)	80.74±8.51 (9)	96.34±2.69 (7)	98.57±4.51 (3)	61.28±17.07 (9)
MDS	1	43.73±7.44 (4)	72.96±7.62 (8)	95.60±1.84 (7)	91.78±7.10 (4)	69.48±10.83 (8)
MDS	2	48.46±7.77 (1)	80.37±8.19 (6)	96.48±2.60 (3)	97.32±5.66 (8)	67.75±16.51 (9)
FDA	1	76.60±7.37 (10)	84.81±5.64 (8)	97.35±1.93 (5)	100±0.00 (1)	69.35±8.72 (7)
FDA	2	99.23±1.62 (1)	82.96±6.34 (10)	96.77±1.51 (9)	100±0.00 (1)	69.29±13.70 (9)
SVM	1	99.61±1.21 (1)	82.59±7.81 (9)	97.22±1.98 (9)	100±0.00 (1)	70.12±8.55 (9)
SVM	2	99.61±1.21 (1)	82.96±7.44 (10)	97.36±3.51 (10)	100±0.00 (1)	69.29±10.93 (9)
QPC	1	99.21±1.65 (1)	81.85±8.80 (10)	97.22±1.74 (7)	100±0.00 (1)	81.34±12.49 (3)
QPC	2	98.44±2.70 (1)	85.55±4.76 (10)	96.62±1.84 (7)	100±0.00 (1)	100±0 (1)
	ALL	1.16±1.88 (10)	79.62±11.61 (9)	96.34±2.52 (7)	98.57±4.51 (2)	71.15±12.68(10)

Table 3 kNN 10-fold crossvalidation accuracy for datasets with reduced features (optimal k value in parenthesis).

The last dataset used in this section is Monks_1. This is a very interesting example how visualization can help to choose which classificator should be used. Almost all visualization methods (MDS, PCA, FDA and SVM, Fig. 6) for one and two-dimensional projections do not show interesting structure. However, the 2D scatterplot of the QPC projection shows quite clear structure that can easily be separated

	# Features	Parity_8	Heart	Wisconsin	Leukemia	Monks1
PCA	1	99.21±1.65 (13/7)	79.25±10.64 (3/2)	97.07±1.68 (3/2)	95.71±6.90 (7/4)	57.94±11.00 (3/2)
PCA	2	99.23±1.62 (13/7)	79.62±7.03 (15/8)	97.36±1.92 (3/2)	95.81±5.16 (7/4)	61.34±11.82 (11/6)
MDS	1	47.66±4.69 (1/1)	77.40±6.16 (3/2)	97.07±1.83 (3/2)	91.78±14.87 (3/2)	68.58±10.44 (3/2)
MDS	2	49.20±1.03 (1/1)	81.11±4.76 (3/2)	96.19±2.51 (9/5)	95.71±6.90 (7/4)	66.98±12.21 (35/18)
FDA	1	73.83±6.97 (17/9)	84.07±6.77 (3/2)	96.92±2.34 (3/2)	100±0.00 (3/2)	67.82±9.10 (3/2)
FDA	2	96.87±3.54 (35/18)	83.70±6.34 (3/2)	96.93±1.86 (11/6)	100±0.00 (3/2)	68.65±14.74 (3/2)
SVM	1	99.23±1.62 (13/7)	83.33±7.25 (3/2)	97.22±1.99 (3/2)	100±0.00 (3/2)	70.32±16.06 (3/2)
SVM	2	99.61±1.21 (13/7)	84.81±6.63 (3/2)	97.22±1.73 (3/2)	100±0.00 (3/2)	69.35±9.80 (3/2)
QPC	1	99.20±2.52 (13/7)	82.22±5.46 (9/5)	96.91±2.01 (3/2)	100±0.00 (3/2)	82.43±12.22 (47/24)
QPC	2	99.61±1.21 (13/7)	83.33±7.25 (13/7)	96.33±2.32 (3/2)	100±0.00 (3/2)	98.46±3.24 (7/4)
	ALL	49.2±1.03 (1/1)	81.48±4.61 (7/4)	95.60±3.30 (7/4)	90.00±9.64 (5/3)	83.26±14.13 (35/18)

Table 4 SSV 10-fold crossvalidation accuracy for datasets with reduced features (total number of nodes/leaves in parenthesis).

	# Features	Parity_8	Heart	Wisconsin	Leukemia	Monks1
PCA	1	39.15±13.47 (256)	81.11±8.08 (118)	96.78±2.46 (52)	98.57±4.51 (4)	63.71±10.68 (98)
PCA	2	43.36±7.02 (256)	82.96±7.02 (113)	96.92±2.33 (53)	97.14±6.02 (4)	63.71±10.05 (95)
MDS	1	42.98±5.84 (256)	77.03±7.15 (170)	95.60±2.59 (54)	91.78±9.78 (28)	69.61±11.77 (88)
MDS	2	43.83±8.72 (256)	82.96±6.09 (112)	96.92±2.43 (52)	97.32±5.66 (5)	64.74±16.52 (103)
FDA	1	45.73±6.83 (256)	85.18±4.61 (92)	97.21±1.88 (52)	100±0.00 (2)	69.93±11.32 (80)
FDA	2	44.16±5.67 (256)	84.81±5.36 (92)	96.77±2.65 (51)	100±0.00 (3)	69.23±10.57 (80)
SVM	1	54.61±6.36 (256)	85.55±5.36 (92)	97.22±1.26 (46)	100±0.00 (2)	71.98±13.14 (78)
SVM	2	50.29±9.28 (256)	85.55±7.69 (92)	96.92±2.88 (47)	100±0.00 (5)	72.75±10.80 (80)
QPC	1	41.46±9.57 (256)	82.59±8.73 (118)	96.34±2.78 (62)	100±0.00 (2)	67.50±13.54 (82)
QPC	2	43.01±8.21 (256)	85.92±5.46 (103)	96.62±1.40 (54)	100±0.00 (2)	66.92±16.68 (83)
	ALL	31.61±8.31 (256)	84.44±5.17 (99)	96.63±2.68 (50)	98.57±4.51 (16)	65.38±10.75 (83)

Table 5 SVML 10-fold crossvalidation accuracy for datasets with reduced features (number of support vectors in parenthesis).

	# Features	Parity_8	Heart	Wisconsin	Leukemia	Monks1
PCA	1	99.20±1.68 (256)	80.00±9.43 (128)	97.36±2.15 (76)	98.57±4.51 (20)	58.84±12.08 (102)
PCA	2	98.83±1.88 (256)	80.00±9.99 (125)	97.22±2.22 (79)	97.14±6.02 (22)	67.17±17.05 (99)
MDS	1	44.10±8.50 (256)	73.70±8.27 (171)	95.74±2.45 (86)	91.78±7.10 (36)	64.67±10.88 (92)
MDS	2	43.04±8.91 (256)	82.59±7.20 (121)	96.63±2.58 (78)	98.75±3.95 (27)	62.17±15.47 (104)
FDA	1	77.76±7.89 (256)	85.18±4.61 (106)	97.65±1.86 (70)	100±0.00 (12)	72.37±9.29 (85)
FDA	2	98.84±1.85 (256)	84.81±6.16 (110)	97.07±2.06 (74)	100±0.00 (15)	70.96±10.63 (85)
SVM	1	99.61±1.21 (9)	85.18±4.61 (107)	96.93±1.73 (69)	100±0.00 (14)	72.82±10.20 (77)
SVM	2	99.61±1.21 (43)	84.07±7.20 (131)	96.92±3.28 (86)	100±0.00 (21)	68.65±13.99 (93)
QPC	1	99.21±1.65 (256)	82.59±10.33 (130)	97.07±1.82 (84)	100±0.00 (10)	67.43±17.05 (84)
QPC	2	98.44±2.70 (24)	85.18±4.93 (132)	96.33±1.87 (107)	100±0.00 (12)	99.16±2.63 (45)
	ALL	16.80±22.76 (256)	82.22±5.17 (162)	96.63±2.59 (93)	98.57±4.51 (72)	78.20±8.65 (87)

Table 6 SVMG 10-fold crossvalidation accuracy for datasets with reduced features (number of support vectors in parenthesis).

using decision tree as classifier (SSV) on the reduced data. Moreover, a very simple tree with 7 nodes and 4 leaves is created. Comparing this with the results in Table 4 one can confirm that this is really the best possible classification method for this data, giving in most crossvalidations 100% accuracy. Moreover, analysis of the QPC projection coefficients helps to convert the solution obtained to logical rules in the original space.

4 Conclusions

The holy grail of computational intelligence is to create methods that will automatically discover the best models for a given data. There is no hope that a single method will be always the best and therefore multistrategy approaches [10] should be developed. Most known machine learning methods may be presented as sequences of transformations. Searching in the space of all possible transformations may be done in an automatic way if a restricted framework for building models is provided, such as the similarity based framework [8]. However, in a general case of arbitrary transformations such search may be quite difficult. Linear separability is the most common, but not the best goal of learning. Initial transformation may show nonlinear structures in the data that – if noticed – may be easy to handle with specific transformations.

Visualization may help to notice what type of algorithm is the most promising. Several linear and nonlinear visualization methods presented here proved to be useful in dimensionality reduction, evaluation of the reliability of classification for individual cases, but also discovering whether simple linear classifier, nearest neighbor approach, radial basis function expansion, naive Bayes or a decision tree will provide simplest analysis. In particular the QPC index recently introduced [16] proved to be quite helpful, showing structures in the Monk _1 problem that other methods were not able to reveal. Studying visualization and transformation-based systems should bring us a bit closer to systems that use meta-learning to create automatically the best data models.

References

1. Asuncion, A., Newman, D.: UCI repository of machine learning databases (2007). URL http://www.ics.uci.edu/∼mlearn/MLRepository.html
2. Bengio, Y., Delalleau, O., Roux, N.L.: The curse of dimensionality for local kernel machines. Tech. Rep. Technical Report 1258, Dṕartement d'informatique et recherche opérationnelle, Université de Montréal (2005)
3. Brazdil, P., Giraud-Carrier, C., Soares, C., Vilalta, R.: Metalearning: Applications to Data Mining. Cognitive Technologies. Springer (2009). URL http://www.liaad.up.pt/pub/2009/BGSV09
4. Cheng, Y.Q., Zhuang, Y.M., Yang, J.Y.: Optimal Fisher discriminant analysis using the rank decomposition. Pattern Recognition **25**(1), 101–111 (1992)

5. Cox, T., Cox, M.: Multidimensional Scaling, 2nd Ed. Chapman and Hall (2001)
6. Duch, W.: *k*-separability. Lecture Notes in Computer Science **4131**, 188–197 (2006)
7. Duch, W., Grąbczewski, K.: Heterogeneous adaptive systems. In: IEEE World Congress on Computational Intelligence, pp. 524–529. IEEE Press, Honolulu (2002)
8. Duch, W., Grudziński, K.: Meta-learning via search combined with parameter optimization. In: L. Rutkowski, J. Kacprzyk (eds.) Advances in Soft Computing, pp. 13–22. Physica Verlag, Springer, New York (2002)
9. Duch, W., Jankowski, N., Grąbczewski, K., Naud, A., Adamczak, R.: Ghostminer data mining software. Tech. rep. (2000-2005). Http://www.fqspl.com.pl/ghostminer/
10. (ed), R.M.: Multistrategy Learning. Kluwer Academic Publishers (1993)
11. Golub, T.: Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science **286**, 531–537 (1999)
12. Grąbczewski, K., Duch, W.: The separability of split value criterion. In: Proceedings of the 5th Conf. on Neural Networks and Soft Computing, pp. 201–208. Polish Neural Network Society, Zakopane, Poland (2000)
13. Grabczewski, K., Jankowski, N.: Versatile and efficient meta-learning architecture: Knowledge representation and management in computational intelligence. In: CIDM, pp. 51–58. IEEE (2007)
14. Grabczewski, K., Jankowski, N.: Meta-learning with machine generators and complexity controlled exploration. In: L. Rutkowski, R. Tadeusiewicz, L.A. Zadeh, J.M. Zurada (eds.) ICAISC, *Lecture Notes in Computer Science*, vol. 5097, pp. 545–555. Springer (2008)
15. Grochowski, M., Duch, W.: Learning highly non-separable Boolean functions using Constructive Feedforward Neural Network. Lecture Notes in Computer Science **4668**, 180–189 (2007)
16. Grochowski, M., Duch, W.: Projection Pursuit Constructive Neural Networks Based on Quality of Projected Clusters. Lecture Notes in Computer Science **5164**, 754–762 (2008)
17. Maszczyk, T., Duch, W.: Support vector machines for visualization and dimensionality reduction. Lecture Notes in Computer Science **5163**, 346–356 (2008)
18. Michie, D., Spiegelhalter, D.J., Taylor, C.C.: Machine learning, neural and statistical classification. Elis Horwood, London (1994)
19. Naud, A.: An Accurate MDS-Based Algorithm for the Visualization of Large Multidimensional Datasets. Lecture Notes in Computer Science **4029**, 643–652 (2006)
20. Pękalska, E., Duin, R.: The dissimilarity representation for pattern recognition: foundations and applications. World Scientific (2005)
21. Schölkopf, B., Smola, A.: Learning with Kernels. Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge, MA (2001)
22. Vilalta, R., Giraud-Carrier, C.G., Brazdil, P., Soares, C.: Using meta-learning to support data mining. IJCSA **1**(1), 31–45 (2004). URL http://www.tmrfindia.org/ijcsa/V1I13.pdf
23. Webb, A.: Statistical Pattern Recognition. J. Wiley & Sons (2002)
24. Wolberg, W.H., Mangasarian, O.: Multisurface method of pattern separation for medical diagnosis applied to breast cytology. In: Proceedings of the National Academy of Sciences, vol. 87, pp. 9193–9196. U.S.A. (1990)

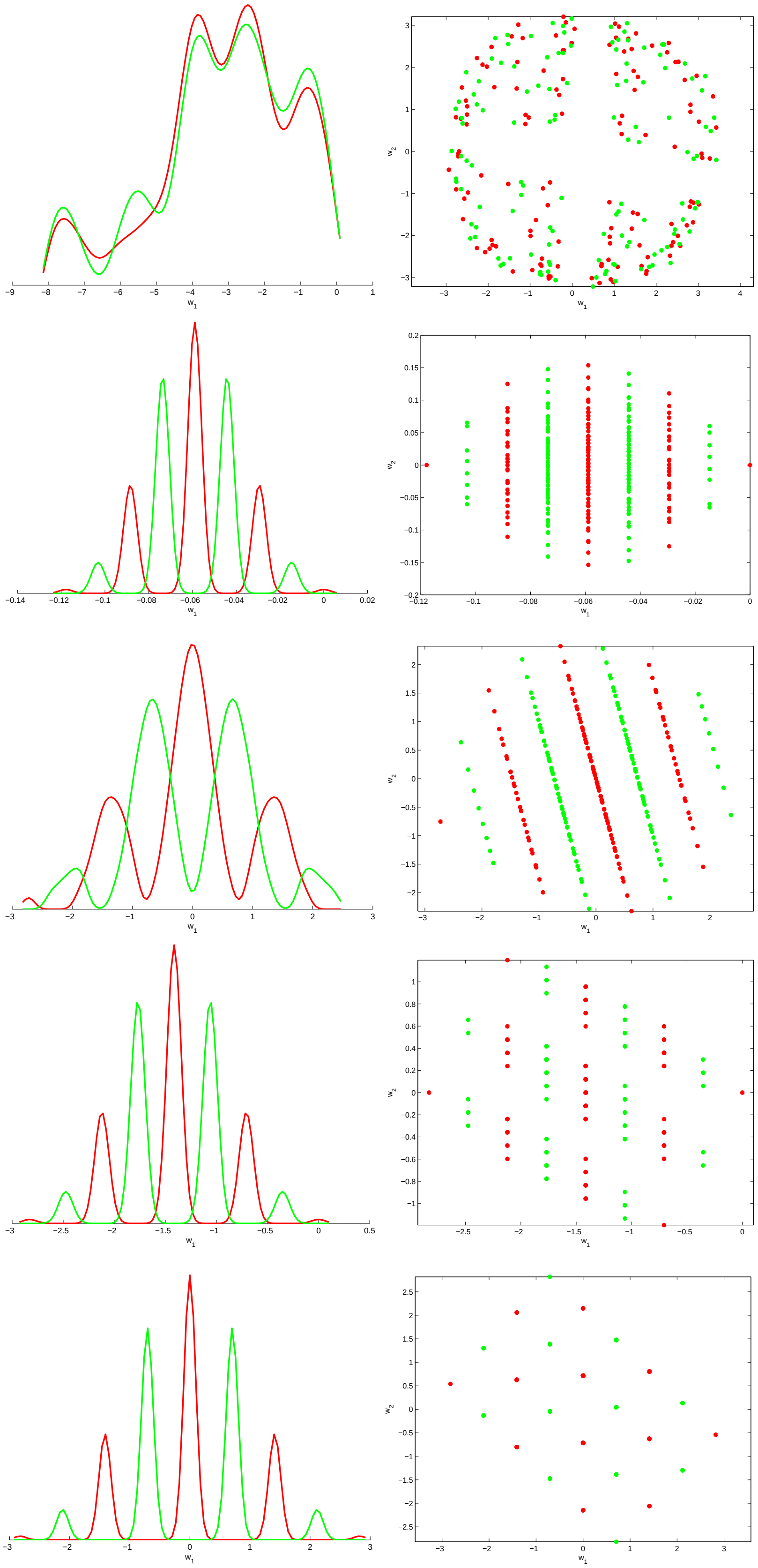

Fig. 2 8-bit parity dataset, from top to bottom: MDS, PCA, FDA, SVM and QPC.

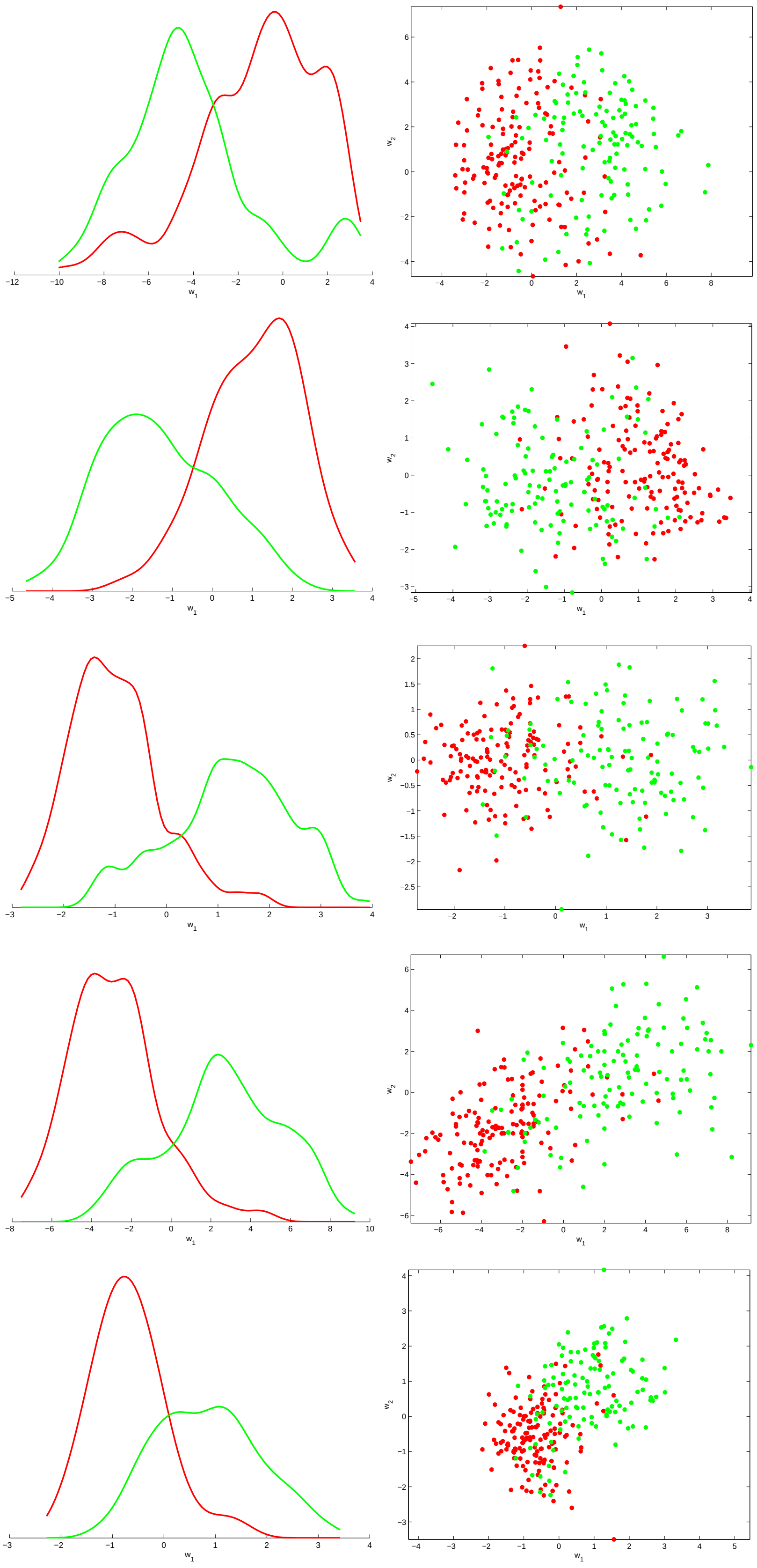

Fig. 3 Heart data set, from top to bottom: MDS, PCA, FDA, SVM and QPC.

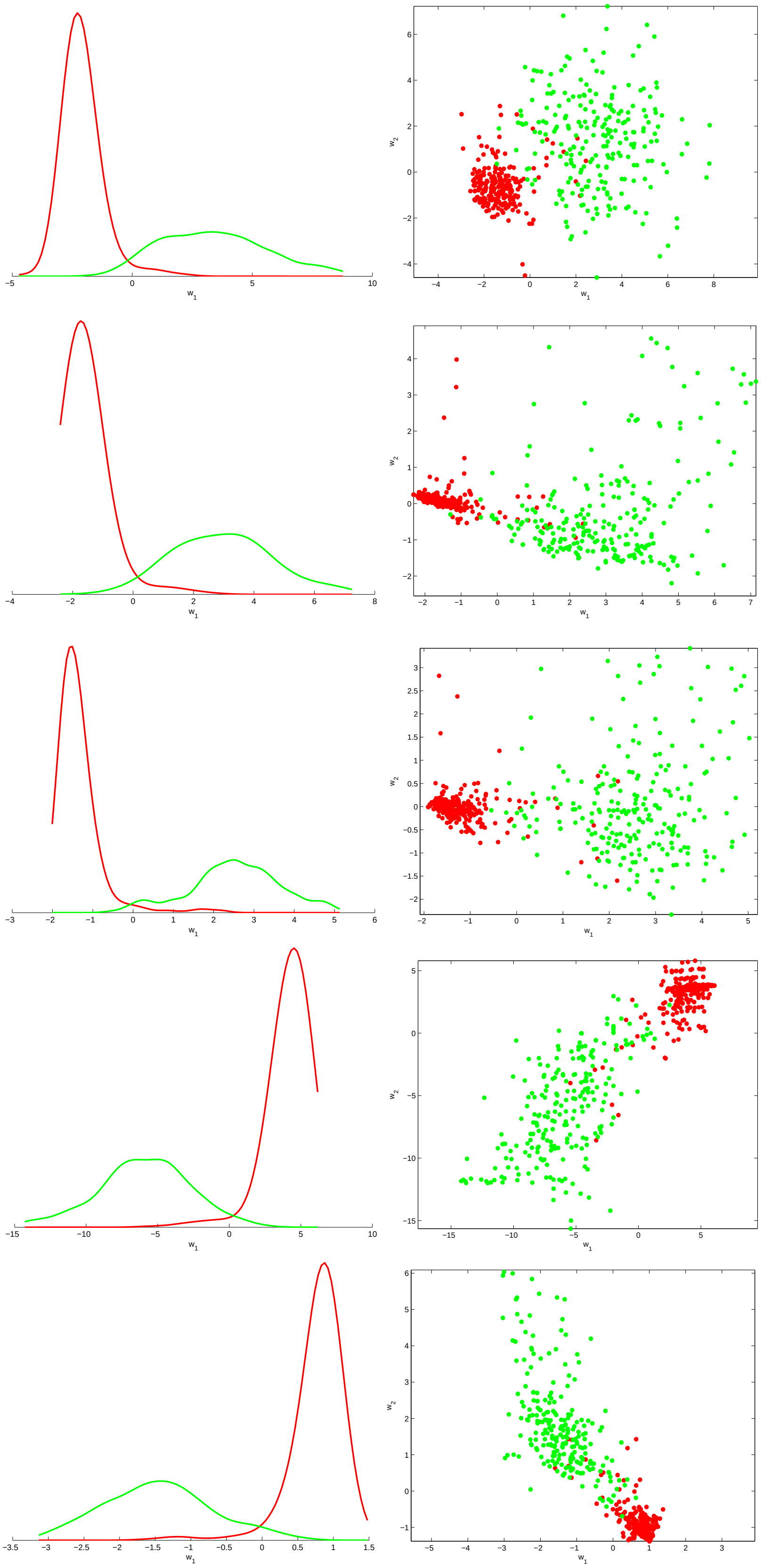

Fig. 4 Wisconsin data set, from top to bottom: MDS, PCA, FDA, SVM and QPC.

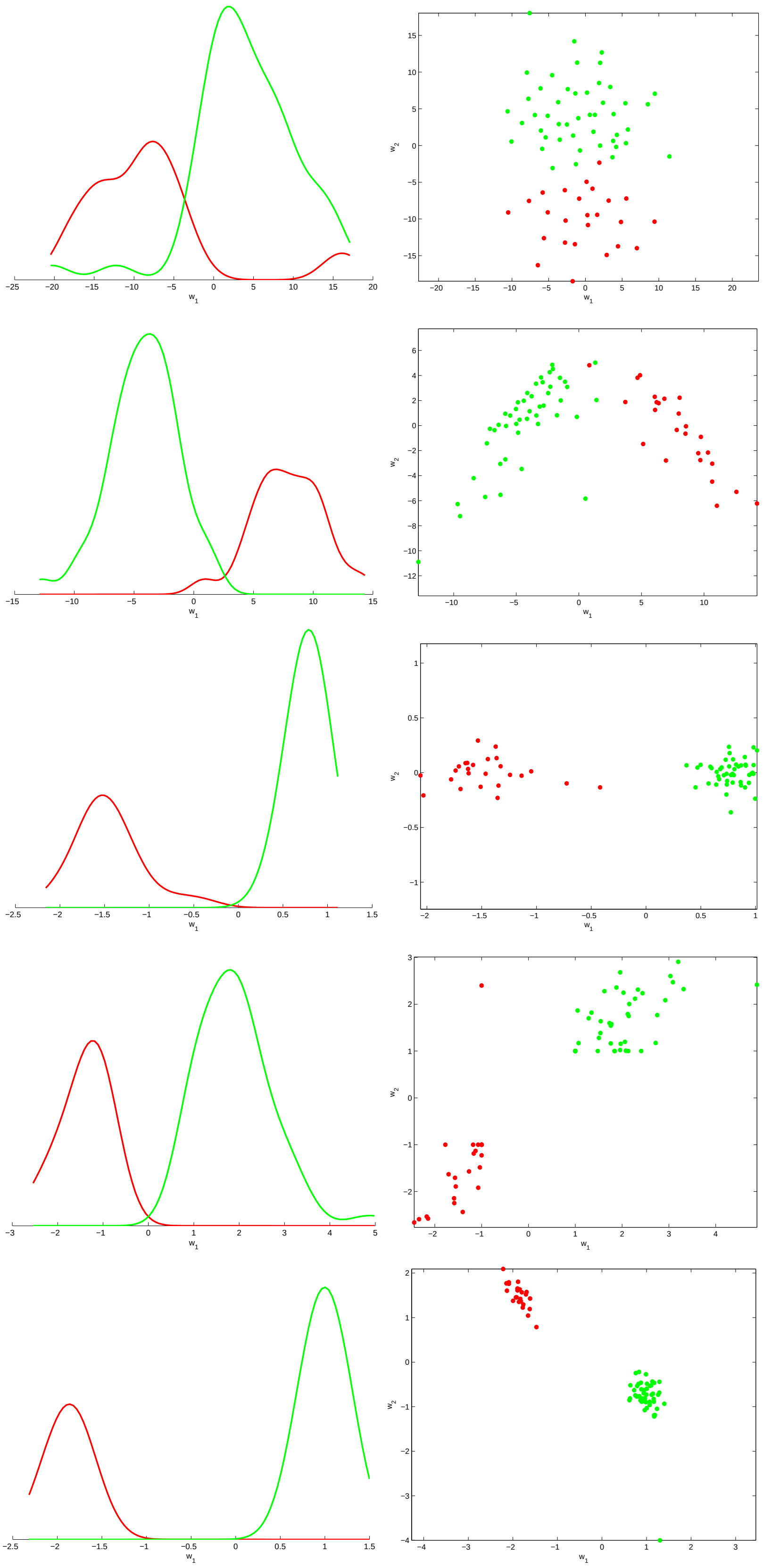

Fig. 5 Leukemia data set, from top to bottom: MDS, PCA, FDA, SVM and QPC.

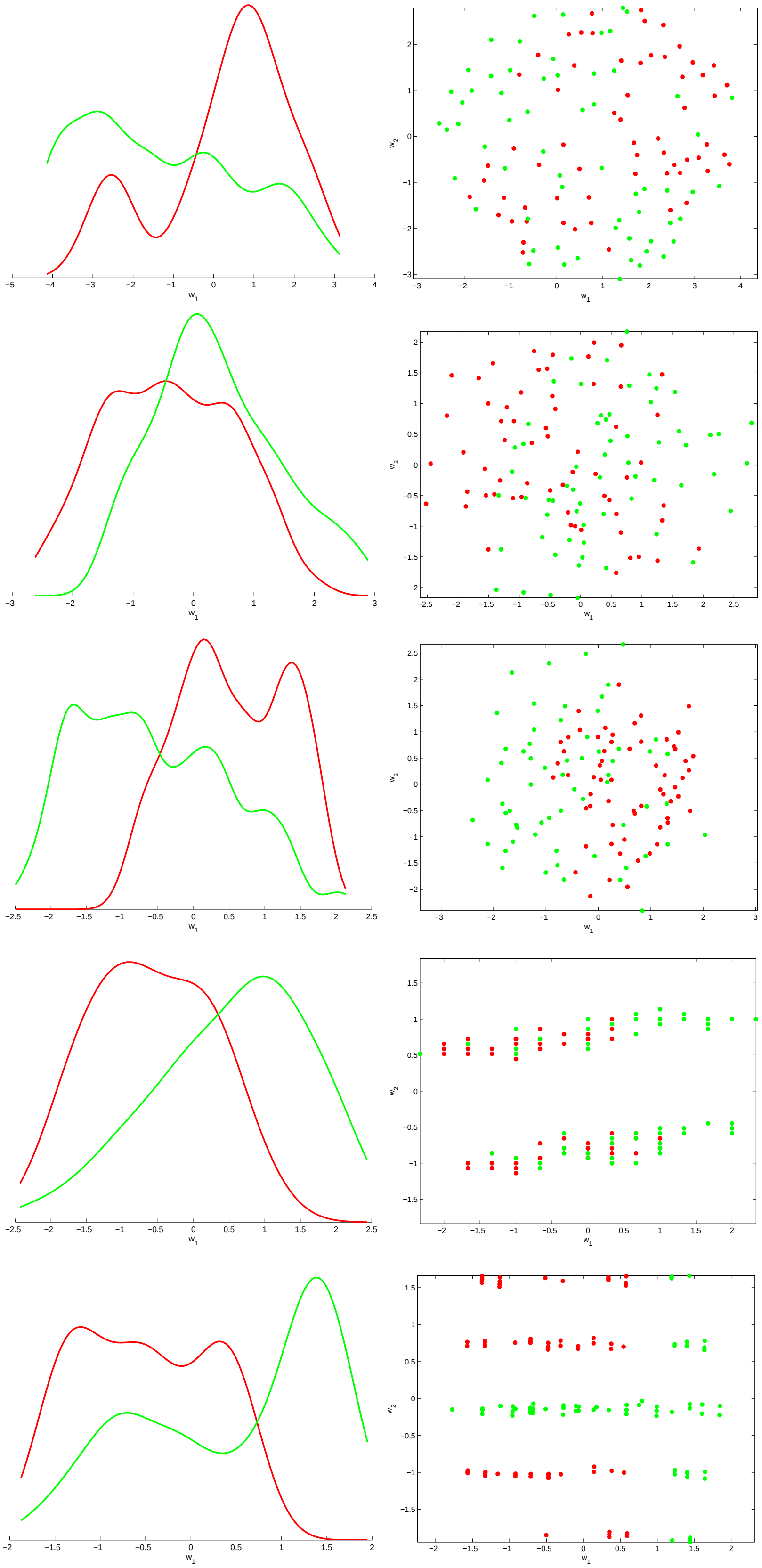

Fig. 6 Monks_1 data set, from top to bottom: MDS, PCA, FDA, SVM and QPC.