Pattern Recognition Letters 52 (2015) 48–52

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Maximum distance minimization for feature weighting ☆

Jens Hocke*, Thomas Martinetz

Institute for Neuro- and Bioinformatics, University of Lübeck, Ratzeburger Allee 160, 23538 Lübeck, Germany

ARTICLE INFO

Article history:
Received 17 April 2014
Available online 16 October 2014

Keywords:
Feature selection
Feature weighting
Metric learning
k-Nearest-Neighbor
Relief
Large Margin Nearest Neighbor Classification

ABSTRACT

We present a new feature weighting method to improve *k*-Nearest-Neighbor (*k*-NN) classification. The proposed method minimizes the largest distance between equally labeled data tuples, while retaining a minimum distance between data tuples of different classes, with the goal to group equally labeled data together. It can be implemented as a simple linear program, and in contrast to other feature weighting methods, it does not depend on the initial scaling of the data dimensions. Two versions, a hard and a soft one, are evaluated on real-world datasets from the UCI repository. In particular the soft version compares very well with competing methods. Furthermore, an evaluation is done on challenging gene expression data sets, where the method shows its ability to automatically reduce the dimensionality of the data.

1. Introduction

The *k*-Nearest-Neighbor (*k*-NN) algorithm is a popular non-linear classifier [1]. The main advantage of this approach is, that it is simple and the results are easy to interpret. A major disadvantage is, however, that the classification results largely depend on the scaling of the input data and the measured features. The scaling may be very arbitrary, and the most commonly used distance measure, the Euclidean distance, may not be a good choice. Therefore, to assure good classification performance, we need to adjust the scaling and distance measure to fit the data.

By scaling the features through appropriate weighting, the classification performance can be improved significantly. This applies not only to the *k*-NN but also to many other distance based classifiers. The quality of the scaling can be measured by the *k*-NN error rate. An optimal rescaling should minimize the classification error E of the *k*-NN algorithm. The goal is to find a weight vector $\vec{w} \in \Re^D, w_\mu \geq 0, \mu = 1, \ldots, D$ that helps the classifier to minimize E. This problem is referred to as the *feature weighting* problem, which is similar to the problem of relevance learning in the context of LVQ classifiers [2]. It is, however, not as closely tied to one classifier as LVQ.

For the Euclidean distance, the weighted distance between two data points $\vec{x}, \vec{x}' \in \Re^D$ is given by

$$d(\vec{x}, \vec{x}') = ||\vec{x} - \vec{x}'||_{\vec{w}} = \sqrt{\sum_{\mu=1}^{D} w_\mu (x_\mu - x'_\mu)^2}, \quad (1)$$

also called the weighted Euclidean distance. This distance measure takes the dimension relevance for classification into account. For irrelevant dimensions, the weights may become zero. This leads to a dimensionality reduction of the data set, which is desirable, because it increases the noise robustness and the generalization performance.

Several methods are available for feature weighting. The most simple approach is to rescale every dimension through normalization of the data distribution variance along every dimension. However, this does not take class label information into account. The Relief algorithm by Kira and Rendell [3] aims to account for this problem. It takes the distance of every data point to its nearest neighbors of the same and a different class. The ratio between these two distances is optimized iteratively such that data from the same class are grouped together. Simba [4] extends this concept. This is done by an iterative update of the nearest neighbors based on the current weight vector for the distance measure. Originally, both methods were developed to select the most important dimensions for classification by learning a weight vector. Other methods for feature weighting are the I-Relief [5] and the loss-margin based algorithm (LMBA). The I-Relief is another extension of Relief that uses the current weight vector when selecting the nearest neighbors. LMBA is derived from the Large Margin Nearest Neighbor Classification described in the next paragraph. For every data point, a circular area is spanned by a *k*-Neighborhood of equally labeled data points. By adapting the weight vector, LMBA tries to clear this area plus some margin from differently labeled data.

Instead of the weighted Euclidean distance, one can also optimize the Mahalanobis distance

$$d(\vec{x}, \vec{x}') = ||\vec{x} - \vec{x}'||_W = \sqrt{(\vec{x} - \vec{x}')^T W (\vec{x} - \vec{x}')} \quad (2)$$

to improve the *k*-NN classification. Here, an entire positive semidefinite matrix W is optimized. By doing so, we are solving a *metric*

☆ This paper has been recommended for acceptance by Jie Zou.

* Corresponding author. Tel.: +49 451 500 5509.
E-mail address: hocke@inb.uni-luebeck.de (J. Hocke).

http://dx.doi.org/10.1016/j.patrec.2014.10.003
0167-8655/

learning problem, by which features can be decorrelated as a major benefit over feature weighting. In case we restrict W to a diagonal matrix, we again obtain the feature weighting problem. Thus feature weighting is a subclass of metric learning. Often, one wants to stick within this subclass. This is to avoid a mixing of all dimensions, which keeps and allows an interpretation of individual dimensions. Large Margin Nearest Neighbor Classification (LMNN) [6,7] is a well known metric learning method, which is closely linked to the concept of the k-NN classifier. Like LMBA, LMNN tries to free an area spanned by k equally labeled data points plus some margin from differently labeled data points. However, instead of the weighting vector a complete metric is adapted.

Learning a linear transformation of the data space and using the Euclidean distance in the transformed space is equivalent to using the Mahalanobis distance in the original space:

$$||\vec{x}-\vec{x}'||_W = \sqrt{(\vec{x}-\vec{x}')^T W(\vec{x}-\vec{x}')} \tag{3}$$

$$= \sqrt{(\vec{x}-\vec{x}')^T L^T L(\vec{x}-\vec{x}')} \tag{4}$$

$$= \sqrt{(L\vec{x}-L\vec{x}')^T(L\vec{x}-L\vec{x}')} = ||L\vec{x}-L\vec{x}'||_2, \tag{5}$$

with the transformation matrix L and a metric $W = L^T L$. To reduce the dimensionality, Principal Component Analysis (PCA) or Independent Component Analysis (ICA) [8] can find such a linear transformation, independent of the labeling. While PCA decorrelates the dimensions, ICA minimizes their statistical dependence. However, since the transformations are found without label information, the transformed space may not be optimal for classification. This is addressed by Linear Discriminant Analysis (LDA) [9]. It optimizes the ratio between scatter of equally labeled data points (intraclass) and differently labeled data points (interclass) in the transformed space. In the original formulation it was limited to a two class setting, but it has been extended to handle multiple classes [10]. These linear transformations share the mixing of dimensions with the metric learning approaches, making it harder to interpret the results in the transformed space.

It is important to note that often not only the weights are updated, but also prototypes which are used to improve the performance of k-NN on large datasets. For example, this can be done by a fuzzy-artificial immune system [11]. However, reduction of the training dataset is out of the scope of this work.

Here, we focus on feature weighting, which allows a better interpretation of the results, but without the decorrelation ability. In the following, we give a detailed description of our linear programming approach for feature weighting [12] and extend it by allowing soft constraints. Besides evaluating the methods on UCI data, we also demonstrate the dimension reduction capabilities on gene expression data in Section 3.

2. Methods

2.1. Maximum distance minimization

A good weighting vector $\vec{w}$ minimizes the classification error E of the k-NN algorithm. The idea of our approach is that the k-NN classification performance should improve, if equally labeled data points are close together and differently labeled data points are far apart. We want to achieve this by minimizing the maximum distance between all pairs of data points of the same class. To avoid the trivial solution $\vec{w} = \vec{0}$, a constraint on the minimum distance for data points of different classes is imposed. Due to these main ideas, which are illustrated in Fig. 1, we call our method Maximum Distance Minimization (MDM). Note, it is possible to do the opposite and impose a maximum intraclass distance while maximizing the minimum interclass distance (Minimum Distance Maximization). Both approaches are mathematically equivalent.

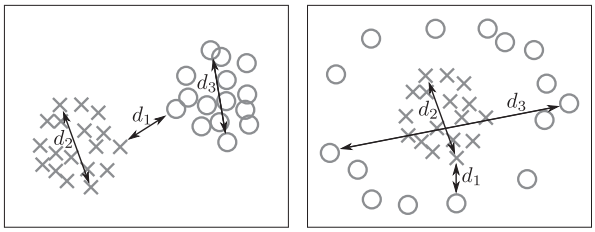

Fig. 1. Part (a) shows two different settings. d_1 denotes the shortest interclass distance. This distance is fixed to one by Eq. (6). The largest intraclass distance for class one (crosses) is d_2 and for class two (circles) d_3. The larger distance of the two (d_2) determines r in Eq. (7) and is minimized.

Given data points $\vec{x}_i \in \Re^D$ with class labels y_i, $i = 1, \ldots, N$, we formally solve the following constrained optimization problem:

$$||\vec{x}_i - \vec{x}_j||^2_{\vec{w}} \geq 1 \qquad \forall i,j : y_i \neq y_j \tag{6}$$

$$||\vec{x}_i - \vec{x}_j||^2_{\vec{w}} \leq r \qquad \forall i,j : y_i = y_j \tag{7}$$

$$\min_{\vec{w}} r \qquad w_\mu \geq 0 \; \forall \mu. \tag{8}$$

The above problem can be formulated as a linear program

$$\min_{\vec{v}} \vec{f}^T\vec{v} \text{ s.t. } A\vec{v} \leq \vec{b}, \;\; \vec{v} \geq \vec{0} \tag{9}$$

with $\vec{v}, \vec{f} \in \Re^{D+1}$, $b \in \Re^{N^2}$, and $A \in \Re^{N^2 \times D+1}$. The vector $\vec{v} = (\vec{w}, r)$ is optimized and with $\vec{f} = (\vec{0}, 1)$, only the r plays a role for the minimization. The constraints are imposed by

$$A_{I(i,j)} = \begin{cases} (-(\vec{x}_i - \vec{x}_j) \circ (\vec{x}_i - \vec{x}_j), 0) & \forall i,j : y_i \neq y_j \\ ((\vec{x}_i - \vec{x}_j) \circ (\vec{x}_i - \vec{x}_j), -1) & \forall i,j : y_i = y_j \end{cases} \tag{10}$$

and

$$b_{I(i,j)} = \begin{cases} -1 & \forall i,j : y_i \neq y_j \\ 0 & \forall i,j : y_i = y_j \end{cases}. \tag{11}$$

Here we use $I(i,j) = (i-1)N + j$ to index the row of A and the element of b. The Hadamard product $A \circ B = (a_{ij} \cdot b_{ij})$ is an element wise multiplication.

In this formulation, the number of constraints grows quadratically with the number of data points. The dimension of the vector $\vec{v}$ to be optimized is $D+1$. Note, that the above constraints can always be fulfilled and, therefore, our optimization problem is always solvable despite having hard constraints.

2.2. Soft maximum distance minimization

Maximum Distance Minimization as introduced above uses hard constraints. This punishes only the most distant pairs and may make MDM sensible to outliers and noisy data. The softness is achieved by introducing slack variables ξ_i for every data point x_i. A punishment for slack variables is added in the optimization criterion weighted by C. This soft approach allows some intraclass distances to be larger than r and hence to reduce the influence of outliers and noise. This is illustrated in Fig. 2. The new optimization problem becomes

$$||\vec{x}_i - \vec{x}_j||^2_{\vec{w}} \geq 1 \qquad \forall i,j : y_i \neq y_j \tag{12}$$

$$||\vec{x}_i - \vec{x}_j||^2_{\vec{w}} \leq r + \xi_i \qquad \forall i,j : y_i = y_j \tag{13}$$

$$\min_{\vec{w}} r + C\sum_i \xi_i \qquad w_\mu \geq 0 \; \forall \mu, \xi_i \geq 0 \; \forall i. \tag{14}$$

The linear program, introduced above, can easily be adopted to the soft version. $\vec{f}$ is extended to $\vec{f} = (\vec{0}, 1, \vec{C})$ and $\vec{v} = (\vec{w}, r, \vec{\xi})$ such that $\vec{f}, \vec{v} \in \Re^{D+1+N}$. The rows of A are changed as follows:

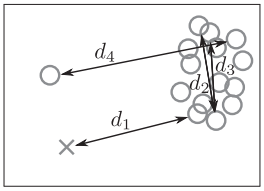

Fig. 2. Illustration of the softness effect. d_1 is the largest interclass distance. There is no change in the influence of d_1 compared to standard MDM. However, there is a change for the intraclass distances d_2 to d_4. In a hard setting r will equal d_4 and, therefore, the outlier will have a major effect on the final weight vector w. The smaller C, the more intraclass tuples are allowed to have a distance larger than r. All these tuples will influence w simultaneously and in that way lower the influence of the outlier. Also the noise effect will be lower, if the final w depends on several similar tuples like d_2 and d_3.

$$A_{I(i,j)} = \begin{cases} (-(\vec{x}_i - \vec{x}_j) \circ (\vec{x}_i - \vec{x}_j), 0, \vec{0}) & \forall i,j : y_i \neq y_j \\ ((\vec{x}_i - \vec{x}_j) \circ (\vec{x}_i - \vec{x}_j), -1, 0, (\delta_{I(i,j)n})_{n=1}^{N}) & \forall i,j : y_i = y_j \end{cases}, \tag{15}$$

with δ being the Kronecker delta. Drawbacks of this soft version are that the dimension of the vector $\vec{v}$ now grows linearly with the number of data points and MDM is not parameter free anymore. C has to be chosen appropriately.

2.3. The main difference

Relief and LMNN use neighbors and neighborhoods for their optimization. This is very intuitive if one wants to improve on the k-NN classification performance, since k-NN also uses neighbors. However, the neighbors have to be selected prior to the optimization process and depend on the initial scaling of the data. One obtains good solutions only if the initial scaling defines neighborhoods which already correspond well to the neighborhoods defined by the optimal scaling. An iterative approach may be used to adopt the neighbors, as it is done in case of Simba. This, however, has the disadvantage that the optimization problem becomes non-convex and, therefore, may lead to even worse results.

We here avoided this problem. Instead of performing a local neighborhood dependent optimization, we opted for a global optimization. Besides being independent of the initial scaling, MDM allows us to formulate a very simple optimization problem. This problem can be solved by linear programming for which many fast solvers are available. An example is the MOSEK-solver,[1] which we used in our experiments.

Our global approach takes only the most distant pairs of equally labeled data points into account. This may not be adequate for every distribution of data points. For some data sets, it may be more beneficial to use local distances like Relief and LMNN. We could also adopt MDM to use such local distances between neighboring data points. This would simply require to predefine which data tuples are neighbors and restrict the optimization problem to these tuples. But this would be at the expense of being dependent on the initial scaling, since the predefinition of adjacent tuples depends on it. Interestingly, as we will see in our experiments, in most cases the fully global approach is superior.

3. Experiments and comparisons

To test MDM, we used datasets from the UCI repository [13] and gene expression datasets available from the Broad Institute website.[2] Both are described in Table 1.

Table 1
Description of the datasets. The top datasets are from the UCI repository, and the bottom ones are gene expression data.

Name	Samples	Dimensions	Classes
Iris	150	4	3
Wine	178	13	3
Breast cancer	683	10	2
Pima diabetes	768	8	2
Parkinsons	195	22	2
Seeds	210	7	3
Breast cancer	98	1213	3
DLBCL	180	661	3
Leukemia	248	985	6
Lung cancer	197	1000	4

We compared MDM with results obtained with the standard Euclidean distance as well as obtained with the feature weighting algorithms Relief and Simba.[3] As a reference and out of competition we also show the results obtained with LMNN as a complete metric learning method. For the soft MDM, the softness parameter C was selected for each split of the data individually. From the set $\{2^{-x} | x \in \{0, \ldots, 10\}\}$, the best C was chosen by 4-fold cross-validation on the training data. For Relief and Simba we need to set the number of training epochs. We here used one epoch, as this is default in the implementation that we used. Longer training sometimes deteriorated the results. Due to the non-convex optimization, five random starting points were chosen for Simba for every training. For LMNN its parameter μ was chosen to be 0.5. The authors described this to be a good choice [7]. To evaluate the classification performance, we split the data into five almost equally large parts and used four of these parts for training and one for testing. The partitioning was used five times for training and testing, with each part being left out once. This was done for 10 different splits of the data, so that 50 different test and training sets were obtained. After the weighting was learned on the training set, k-NN with $k = 3$ was used to obtain the error rates on the independent test set.

First, we compared the classification performance on the UCI datasets. Table 2 shows the results on the raw data. MDM is clearly superior. The error rates are improved significantly compared to standard k-NN based on the original scaling ("Euclidean"). Only for the iris data, the original scaling is a good choice. Relief and Simba sometimes even worsen the classification performance compared to the original scaling.

In Table 3 we see the results after a prior rescaling such that the data distribution is normalized to zero mean and variance one along each dimension. With prior rescaling, Relief and Simba become competitive due to a different initial selection of the neighbors. Obviously, Relief and Simba seem to be very dependent on a good initial scaling. It seems that the initial neighbors more or less remain neighbors during the optimization procedure. But then, obviously, already the initial scaling is a good choice and achieves good results. Even though the other methods in general improve their results on the preprocessed data, MDM remains very competitive. Interestingly, not for all datasets the preprocessing by normalization yields improved results. Especially for the iris data the initial scaling seems to be a better choice. This does however demonstrate that it is not always clear whether a prior rescaling and which rescaling is beneficial. The main advantage of MDM accounts for this problem, since it is independent of such a prior rescaling. Another interesting result is that although LMNN is much more flexible and complex, it does not perform better, at least on these data sets.

[1] http://www.mosek.com/.

[2] http://www.broadinstitute.org/cgi-bin/cancer/datasets.cgi (full dataset names: Breast-A, DLBCL-B, St. Jude Leukemia, Lung Cancer).

[3] We used a implementation by A. Navot and R. Gilad-Bachrach, which is available at http://www.cs.huji.ac.il/labs/learning/code/feature_selection.bak/.

Table 2
Results on UCI datasets. The top entry is the average test error in percent followed by the STD in parentheses. Below the error rates, the average rank, again followed by the STD, is given. In case of the feature weighting methods, the rank is equal to the number of non-zero weights. The best results obtained with feature weighting are indicated by bold face.

	Euclidean	MDM	MDM soft	Relief	Simba	LMNN
Iris	**3.87 (3.32)**	4.33 (3.10)	4.00 (2.94)	4.00 (2.86)	6.27 (3.91)	4.00 (2.86)
	4.00 (0.00)	4.00 (0.00)	4.00 (0.00)	4.00 (0.00)	3.96 (0.20)	4.00 (0.00)
Wine	30.28 (7.25)	2.64 (2.81)	**2.13 (2.56)**	32.80 (6.65)	32.52 (7.19)	5.57 (3.82)
	13.00 (0.00)	12.10 (0.65)	12.08 (0.67)	13.00 (0.00)	13.00 (0.00)	12.12 (0.63)
Breast cancer	39.20 (4.35)	3.50 (1.43)	**2.86 (1.27)**	39.36 (4.30)	39.36 (4.30)	4.06 (1.34)
	10.00 (0.00)	9.94 (0.31)	9.70 (0.51)	10.00 (0.00)	9.64 (0.85)	8.02 (0.25)
Pima diabetes	29.99 (3.45)	27.34 (2.52)	**26.43 (2.90)**	29.41 (3.18)	29.92 (3.53)	28.42 (3.15)
	8.00 (0.00)	7.90 (0.30)	7.76 (0.43)	8.00 (0.00)	7.42 (0.70)	8.00 (0.00)
Parkinsons	14.72 (4.96)	10.31 (4.85)	**7.59 (4.21)**	15.49 (4.86)	15.64 (4.75)	13.49 (4.91)
	22.00 (0.00)	21.20 (0.88)	20.44 (0.73)	22.00 (0.00)	22.00 (0.00)	21.82 (0.39)
Seeds	11.90 (3.63)	7.52 (3.74)	**7.00 (2.98)**	11.71 (3.56)	11.86 (3.65)	4.86 (2.84)
	7.00 (0.00)	7.00 (0.00)	7.00 (0.00)	7.00 (0.00)	7.00 (0.00)	7.00 (0.00)

Table 3
Results on the UCI datasets after prior rescaling. The dimensions were normalized such that the data distributions have a mean equal zero and a variance equal one. The notation and structure of this table is the same as in Table 2.

	Euclidean	MDM	MDM soft	Relief	Simba	LMNN
Iris	5.40 (3.92)	4.33 (3.10)	**4.00 (2.94)**	4.87 (3.10)	4.73 (2.94)	4.47 (3.27)
	4.00 (0.00)	4.00 (0.00)	4.00 (0.00)	4.00 (0.00)	4.00 (0.00)	4.00 (0.00)
Wine	3.88 (2.84)	2.64 (2.81)	**2.13 (2.56)**	3.37 (3.21)	3.43 (3.03)	2.42 (2.12)
	13.00 (0.00)	12.70 (0.51)	12.96 (0.20)	13.00 (0.00)	12.54 (0.91)	13.00 (0.00)
Breast cancer	3.60 (1.43)	3.38 (1.48)	**2.75 (1.36)**	3.35 (1.38)	4.04 (1.42)	3.38 (1.50)
	10.00 (0.00)	10.00 (0.00)	9.88 (0.33)	10.00 (0.00)	9.76 (0.62)	9.48 (0.79)
Pima diabetes	26.73 (2.64)	27.34 (2.52)	**26.53 (2.89)**	26.90 (3.54)	27.20 (3.45)	26.46 (2.67)
	8.00 (0.00)	8.00 (0.00)	8.00 (0.00)	8.00 (0.00)	5.66 (0.85)	8.00 (0.00)
Parkinsons	9.13 (3.85)	10.31 (4.85)	9.59 (5.10)	**5.69 (3.25)**	7.13 (3.92)	5.74 (2.91)
	22.00 (0.00)	21.50 (1.11)	21.16 (1.89)	22.00 (0.00)	21.92 (0.27)	21.96 (0.20)
Seeds	8.05 (3.00)	7.52 (3.74)	**7.00 (2.98)**	10.24 (3.51)	9.57 (3.77)	6.67 (3.50)
	7.00 (0.00)	7.00 (0.00)	7.00 (0.00)	7.00 (0.00)	6.98 (0.14)	7.00 (0.00)

Table 4
Results on gene expression data (after prior rescaling). The notation and structure of this table is the same as in Table 2.

	Euclidean	MDM	Relief	Simba	LMNN
Breast cancer	**8.07 (6.13)**	11.42 (7.25)	13.16 (7.89)	14.47 (7.43)	9.78 (7.13)
	1213.00 (0.00)	364.76 (62.65)	1213.00 (0.00)	1213.00 (0.00)	1137.42 (2.97)
DLBCL	13.11 (5.24)	14.67 (5.33)	**12.00 (5.62)**	13.28 (6.55)	15.44 (4.32)
	661.00 (0.00)	293.86 (34.13)	661.00 (0.00)	661.00 (0.00)	559.56 (1.97)
Leukemia	2.21 (2.27)	**1.74 (1.96)**	2.18 (2.45)	4.12 (2.87)	0.69 (1.33)
	985.00 (0.00)	473.24 (55.28)	985.00 (0.00)	984.96 (0.20)	821.48 (7.53)
Lung cancer	**4.37 (2.77)**	5.49 (3.18)	4.88 (2.78)	8.29 (4.12)	4.78 (2.66)
	1000.00 (0.00)	536.62 (78.55)	1000.00 (0.00)	999.78 (0.46)	870.86 (1.87)

In Table 4 we see the results on the gene expression data. They were obtained with the same prior rescaling as used for the UCI data in Table 3. The gene expression data are a lot more challenging because the number of data dimensions compared to the number of data points is very large, as shown in Table 1. This curse of dimensionality is very challenging for feature weighting and metric learning methods. All methods perform on a similar level as standard Euclidean distance, taking the large standard deviation into account. However, we see a nice feature of our MDM method: MDM remarkably reduces the dimensionality of the data without the use of any parameters. The dimensionality reduction is directly induced by the formulation of the optimization problem. Methods like Relief and Simba, which were specifically designed for this task, need a threshold to be set either by some heuristic or by hand. The dimensionality can be reduced even further if the soft MDM is used, but this comes at the expense of the softness parameter which needs to be set.

4. Conclusion

We presented a simple feature weighting method to handle the problem of arbitrarily scaled data dimensions. The weightings our method identifies yield very competitive k-NN classification results on standard benchmark problems. Since our optimization problem does not directly optimize the error rate of the locally operating k-NN, we expect that also other distance based classification methods would benefit from this approach. Interestingly, the error rates we achieve with MDM are comparable to those of the much more complex metric learning method LMNN for most of the UCI data sets in

our experiments. This shows that the additional flexibility of metric learning is not always needed. Hence, it is better for some applications to use feature weighting and keep the interpretability.

What distinguishes our method from commonly used methods is that it is independent of the initial scaling of the data dimensions. By this it eliminates the need for preprocessing. To achieve this independence, we compare all tuples during optimization and thereby avoid a predefinition of which data points are adjacent. MDM performed well on most of the benchmark data with this global approach. Of course, there might be data sets where local structures should be taken into account. The Parkinsons data of the UCI data sets seems to be an example. There, in our comparison (Table 3) the methods based on local distances perform better.

In addition, we demonstrated the capability of our method to reduce the dimensionality of a given classification problem on high dimensional data. Contrary to other methods, parameters as for example a threshold are not needed to cut off dimensions with small weights. This makes MDM a straightforward method to use for dimensionality reduction.

References

[1] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Trans. Inf. Theory 131 (1967) 21–27.
[2] B. Hammer, T. Villmann, Generalized relevance learning vector quantization, Neural Netw. 158 (2002) 1059–1068.
[3] K. Kira, L.A. Rendell, A practical approach to feature selection, in: Proceedings of the Ninth International Workshop on Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1992, pp. 249–256.
[4] R. Gilad-Bachrach, A. Navot, N. Tishby, Margin based feature selection – theory and algorithms, in: Proceedings of the Twenty-first International Conference on Machine Learning, ACM, New York, NY, USA, 2004, pp. 43–50.
[5] Y. Sun, J. Li, Iterative relief for feature weighting, in: Proceedings of the 23rd International Conference on Machine Learning, ACM, New York, NY, USA, 2006, pp. 913–920.
[6] K.Q. Weinberger, J. Blitzer, L.K. Saul, Distance metric learning for large margin nearest neighbor classification, in: Y. Weiss, B. Schölkopf, J. Platt (Eds.), Advances in Neural Information Processing Systems, 18, MIT Press, Cambridge, MA, 2006, pp. 1473–1480.
[7] K.Q. Weinberger, L.K. Saul, Distance metric learning for large margin nearest neighbor classification, J. Mach. Learn. Res. 10 (2009) 207–244.
[8] P. Comon, Independent component analysis, a new concept?, Signal Process. 363 (1994) 287–314.
[9] R.A. Fisher, The use of multiple measurements in taxonomic problems, Ann. Eugen. 7 (1936) 179–188.
[10] C.R. Rao, The utilization of multiple measurements in problems of biological classification, J. R. Stat. Soc. Series B Methodol. 102 (1948) 159–203.
[11] S. Şahan, K. Polat, H. Kodaz, S. Güneş, A new hybrid method based on fuzzy-artificial immune system and k-nn algorithm for breast cancer diagnosis, Comput. Biol. Med. 373 (2007) 415–423.
[12] J. Hocke, T. Martinetz, Feature weighting by maximum distance minimization, in: ICANN 2013, Springer, 2013, pp. 420–425.
[13] A. Frank, A. Asuncion, UCI machine learning repository, 2010, URL: http://archive.ics.uci.edu/ml.