Classification with Minimax Distance Measures

Morteza Haghir Chehreghani
Xerox Research Centre Europe - XRCE
6 chemin de Maupertuis
38240 Meylan, France
morteza.chehreghani@xrce.xerox.com

Abstract

Minimax distance measures provide an effective way to capture the unknown underlying patterns and classes of the data in a non-parametric way. We develop a general-purpose framework to employ Minimax distances with any classification method that performs on numerical data. For this purpose, we establish a two-step strategy. First, we compute the pairwise Minimax distances between the objects, using the equivalence of Minimax distances over a graph and over a minimum spanning tree constructed on that. Then, we perform an embedding of the pairwise Minimax distances into a new vector space, such that their squared Euclidean distances in the new space are equal to their Minimax distances in the original space. We also consider the cases where multiple pairwise Minimax matrices are given, instead of a single one. Thereby, we propose an embedding via first summing up the centered matrices and then performing an eigenvalue decomposition. We experimentally validate our framework on different synthetic and real-world datasets.

Introduction

Data is usually described by a set of objects and a corresponding representation. The representation can be for example the vectors in a vector space or the pairwise distances between the objects. In real-world applications, the data is often very complicated and a priori unknown. Thus, the basic representation, e.g., *Euclidean* distance, *Mahalanobis* distance, *cosine* similarity and *Pearson correlation*, might fail to correctly capture the underlying patterns or classes. Thereby, the raw data needs to be processed further in order to obtain a more sophisticated representation. Kernel methods are a common approach to enrich the basic representation of the data and model the underlying patterns (Shawe-Taylor and Cristianini 2004; Hofmann, Schlkopf, and Smola 2006). However, the applicability of this approach is confined by several limitations, such as, i) finding the optimal parameter(s) of a kernel function is often very critical and nontrivial (Nadler and Galun 2007; Luxburg 2007), and ii) as we will see in Section 4.1, kernels assume a global structure which does not distinguish between different type of classes in the data.

A category of distance measures, called *link-based* measure (Fouss et al. 2007; Chebotarev 2011), consider all the *paths* between the objects represented in a graph. The *path-specific* distance between nodes i and j is computed by summing the edge weights on this path (Yen et al. 2008). Their link-based distance is then obtained by summing up the *path-specific* measures of all paths between them. Such a distance measure is known to better capture the arbitrarily shaped patterns compared to the basic representations such as Euclidean or Mahalobis distances. Link-based measures are often obtained by inverting the Laplacian of the distance matrix, in the context of regularized Laplacian kernel and Markov diffusion kernel (Yen et al. 2008; Fouss et al. 2012). However, computing all-pairs link-based distances requires $\mathcal{O}(N^3)$ runtime, where N is the number of objects; thus it is not applicable to large datasets.

A rather similar distance measure, called *Minimax* measure, selects the minimum largest gap among all possible paths between the objects. This measure, known also as *Path-based* distance measure (Fischer and Buhmann 2003), has been first investigated on clustering applications. It was later proposed as an axiom for evaluating clustering methods (Zadeh and Ben-David 2009), as well as for K-nearest neighbor search (Kim and Choi 2007; 2013). A straightforward approach to compute all-pairs Minimax distances is to use an adapted variant of the Floyd-Warshall algorithm. The runtime of this algorithm is $\mathcal{O}(N^3)$ (Aho and Hopcroft 1974; Cormen et al. 2001). This distance measure is also integrated into an adapted variant of K-means providing an agglomerative algorithm whose runtime is $\mathcal{O}(N^2|E| + N^3 \log N)$ (Fischer and Buhmann 2003) ($|E|$ indicates the number of edges in the corresponding graph).

In this paper, we consider classification with Minimax distance measures. Minimax distances have been so far applied only to a very limited type of classification, i.e. to K-nearest neighbor search. The method in (Kim and Choi 2007) presents a message passing algorithm with forward and backward steps, similar to the sum-product algorithm (Kschischang, Frey, and Loeliger 2006). The method takes $\mathcal{O}(N)$ time, which is in theory equal to the standard K-nearest neighbor search, but the algorithm needs several visits of the training dataset. Moreover, this method requires computing a minimum spanning tree (MST) in advance which might require $\mathcal{O}(N^2)$ runtime. Later on, a greedy algorithm (Kim and Choi 2013), proposes to compute the Minimax K nearest neighbors by space partitioning and using Fibonacci heaps whose runtime is $\mathcal{O}(\log N + K \log K)$. However, this method is applicable only to Euclidean spaces and assumes the graph is sparse. Very recently, a linear time Minimax K-nearest neighbor search is proposed (Chehreghani 2016) which is applicable to general graphs and distances. This method, in addition to the search, provides an outlier detection mechanism too.

Motivation Minimax distances enable to cope with arbitrarily shaped classes in the data. For example, it has been shown that Minimax K-nearest neighbor classification is effective on non-spherical data, whereas the standard variant, the metric learning approach (Weinberger and Saul 2009), or the shortest path distance (Tenenbaum, de Silva, and Langford 2000) might give poor results, since they ignore the underlying geometry (see for example Figure 1 in (Kim and Choi 2013)). In particular, four properties of Minimax distances are attractive to us: i) They enable to compute the classes in a non-parametric way, i.e. unlike many kernel methods, they do not require fixing any critical parameter in advance. ii) They extract the class-specific structures, i.e. they adapt appropriately whenever the classes differ in shape or type. iii) They take into account the transitive relations: if object a is similar to b, b is similar to c, ..., to z, then the Minimax distance between a and z will be small, although their direct distance might be large. This property is particularly useful when dealing with elongated or arbitrarily shaped classes. iv) Many classification methods perform on a vector representation of the objects. However, such a representation might not be available. We might be given only the pairwise distances which do not necessarily induce a *metric*. Minimax distances satisfy the *metric* conditions and enable to compute an embedding, as we will investigate in this paper. Our goal is to develop a general-purpose framework wherein many different classification algorithms can be applied to Minimax distances, beyond K-nearest neighbor classification.

Contributions To achieve our goal, we follow a two-step strategy: i) We exploit an efficient approach to compute pairwise Minimax distances, using the equivalence of pairwise Minimax distances over a graph and over a minimum spanning tree constructed on the graph. This approach leads to reduces the runtime of computing pairwise Minimax distance to $\mathcal{O}(N^2)$ from $\mathcal{O}(N^3)$. ii) We employ the *ultrametric* property of Minimax distances to perform an embedding into a vector space, such that the pairwise Minimax distances of the objects in the original space are equal to their squared Euclidean distances in the new vector space. Such an embedding allows us to apply any numerical classification algorithm on the resultant Minimax vectors. Then, we consider the cases that the data is represented in a very high-dimensional space, where the interesting Minimax pattern might be hidden in subspaces instead of the original data space. Hence, we propose an efficient method to compute separately Minimax distances for each dimension and then apply an embedding which corresponds to the whole collection of the Minimax matrices at different dimensions. Finally, we experimentally study our framework on different synthetic and real-world datasets and illustrate its effectiveness and superior performance in different settings.

General Minimax Classification

A dataset can be modeled by a graph $\mathcal{G}(\mathbf{O}, \mathbf{D})$, where $\mathbf{O}$ and $\mathbf{D}$ respectively indicate the set of N objects (nodes) and the corresponding edge weights such that $\mathbf{D}_{ij}$ shows the pairwise distance between objects i and j.[1] In general, $\mathbf{D}$ might not yield a *metric*, i.e. the triangle inequality may not hold. In our study, $\mathbf{D}$ needs to satisfy three basic conditions: i) zero self distances, i.e. $\forall i, \mathbf{D}_{ii} = 0$, ii) non-negativity, i.e. $\forall i, j, \mathbf{D}_{ij} \geq 0$, and iii) symmetry, i.e. $\forall i, j, \mathbf{D}_{ij} = \mathbf{D}_{ji}$. We also assume that there are no duplicates, i.e. the pairwise distance between every two distinct objects is positive. For this purpose, we may either remove the duplicate objects or perturb them slightly to make the zero non-diagonal elements of $\mathbf{D}$ positive. The goal is then to classify the objects (the test subset) according to a representation, which can be the base representation (i.e. $\mathbf{D}$) or the Minimax distances.

Formally, the Minimax (MM) distance between objects i and j is defined as

$$\mathbf{D}^{MM}_{i,j} = \min_{r \in \mathcal{R}_{ij}(\mathcal{G})} \big\{ \max_{1 \leq l \leq |r|-1} \mathbf{D}_{r(l)r(l+1)} \big\}, \qquad (1)$$

where $\mathcal{R}_{ij}(\mathcal{G})$ is the set of all paths between i and j. Each path r is identified by a sequence of object indices, i.e. $r(l)$ shows the l^{th} object on the path.

We aim to propose a unified framework for performing arbitrary numerical classification methods with Minimax distances. To cover different classification algorithms, we pursue the following strategy:

1. We compute the pairwise Minimax distances for all pairs of objects i and j in the dataset.
2. We, then, compute an embedding of the objects into a new vector space such that their pairwise (squared Euclidean) distances in this space equal their Minimax distances in the original space.

Notice that vectors are the most basic way for data representation, since they render a bijective mapping between the objects and the measurements. Hence, any classification method which performs on numerical data can benefit from our approach. Such classification methods perform either on the objects in a vector space, e.g. logistic regression, or on a kernel matrix computed from the pairwise relations between the objects. In the later case, the pairwise Minimax distances can be used for this purpose, for example through an exponential transformation, or the kernel can be computed from the final Minimax vectors.

Computing pairwise Minimax distances

Previous works for computing Minimax distances (e.g. applied to clustering) use a variant of Floyd-Warshall algorithm whose runtime is $\mathcal{O}(N^3)$ (Aho and Hopcroft 1974; Cormen et al. 2001), or combine it with K-means in an

[1] For simplicity of explanation, we assume that the graph is full, i.e. the missing edges are filled by a large value. However, our analysis can be easily extended to arbitrary graphs.

agglomerative algorithm whose runtime is $\mathcal{O}(N^2|E| + N^3 \log N)$ (Fischer and Buhmann 2003). To reduce such a computational demand, we follow a more efficient procedure established in two steps: i) build a minimum spanning tree (MST) over the graph, and then, ii) compute the Minimax distances over the MST.

I. Equivalence of Minimax distances over a graph and over a minimum spanning tree on the graph. We exploit the equivalence of Minimax distances over an arbitrary graph and those obtained from a minimum spanning tree on the graph, as expressed in Theorem 1. The equivalence is concluded in a similar way to the *maximum capacity* problem (Hu 1961), where one can show that picking an edge which does not belong to a minimum spanning tree, yields a larger Minimax distance (i.e., a contradiction occurs).

Theorem 1. *Given graph $\mathcal{G}(\mathbf{O}, \mathbf{D})$, for every pair of objects $i, j \in \mathbf{O}$ their Minimax distance over $\mathcal{G}$, i.e. $\mathbf{D}^{MM}_{ij}$, is identical to their Minimax distance over any minimum spanning tree constructed on that.*

Therefore, to compute the Minimax distances $\mathbf{D}^{MM}$, we need to care only about the edges which are present in an MST over the graph. Then, the Minimax distances are written by $\mathbf{D}^{MM}_{ij} = \max_{1 \leq l \leq |r_{ij}|-1} \mathbf{D}_{r_{ij}(l) r_{ij}(l+1)}$, where r_{ij} indicates the (only) path between i and j. This result does not depend on the particular choice of the algorithm used for constructing the minimum spanning tree. The graph that we work on is a full graph, i.e. we compute the pairwise distances between all pairs of objects. For full graphs, the straightforward implementation of the Prim's algorithm (Prim 1957) using an auxiliary vector requires $\mathcal{O}(N^2)$ runtime which is an optimal choice.

II. All-pairs Minimax distances over a minimum spanning tree At the next step, after constructing a minimum spanning tree, we compute the pairwise Minimax distances over that. A naive and straightforward algorithm would perform a Depth First Search (DFS) from each node to compute the Minimax distances by keeping the track of the largest distance between the initial node and each of the traversed nodes. A single run of DFS requires $\mathcal{O}(N)$ runtime and thus the total time will be $\mathcal{O}(N^2)$. However, such a method might lead to visiting some edges multiple times which renders an unnecessary extra computation. For instance, a maximal edge weight might appear in many DFSs such that it is processed several times. Hence, a more elegant approach would first determine and collect all the objects whose pairwise distances are represented by an edge weight and then assigns the weight as their Minimax distances. According to Theorem 1, every edge in the MST represents some Minimax distances. Thus, we first find the edge(s) which represent only few Minimax distances, namely only one pairwise distance, such that the respective objects can be immediately identified. Lemma 2 suggest existence of this kind of edges.

Lemma 2. *In a minimum spanning tree, for the minimal edge weights we have $\mathbf{D}^{MM}_{ij} = \mathbf{D}_{ij}$, where i and j are the two objects that occur exactly at the two sides of the edge.*

Proof. Without loss of generality, we assume that the edge weights are distinct. Let e_{min} denote the edge with minimal weight in the MST. We consider the pair of objects p and q such that at least one of them is not directly connected to e_{min} and show that e_{min} does not represent their Minimax distance. In a MST, there is exactly one path between each pair of objects. Thus, on the path between p and q there is at least another edge whose weight is not smaller than the weight of e_{min}, which hence represents the Minimax distance between p and q, instead of e_{min}. □

Lemma 2 yields a dynamic programming approach to compute the pairwise Minimax distances from a tree. We first sort the edge weights of the MST via for example merge sort or heap sort (Cormen et al. 2001) which in the worst case require $\mathcal{O}(N \log N)$ time. We then consider each object as a separate component. We process the edge weights one by one from the smallest to the largest, and at each step, we set the Minimax distance of the two respective components by this weight. Then, we remove the two components and replace them by a new component constructed from the combination (union) of the two. We repeat these two steps for all edges of the minimum spanning tree. A main advantage of this algorithm is that whenever an edge is processed, all the nodes that it represents their Minimax distances are ready in the components connected to each side of the edge. Thus, we do not need to visit this edge later for another pair of objects.

Embedding of pairwise Minimax distances

In the next step, given the matrix of pairwise Minimax distances $\mathbf{D}^{MM}$, we obtain an embedding of the objects into a vector space such that their pairwise squared Euclidean distances in this new space are equal to their Minimax distances in the original space. For this purpose, in Theorem 3, we investigate a useful property of Minimax distance measures, called the *ultrametric* property (Leclerc 1981), and use this property to prove the existence of such an embedding.

Theorem 3. *Given the pairwise distances $\mathbf{D}$, the matrix of Minimax distances $\mathbf{D}^{MM}$ induces an $\mathcal{L}_2^2$ embedding, i.e. there exist a new vector space for the set of objects $\mathbf{O}$ wherein the pairwise squared Euclidean distances are equal to the pairwise Minimax distances in the original space.*

Proof. First, we investigate that the pairwise Minimax distances $\mathbf{D}^{MM}$ constitute an *ultrametric*. The conditions to be satisfied are:

1. $\forall i, j : \mathbf{D}^{MM}_{ij} = 0$ if and only if $i = j$. We verify each of the conditions separately. i) If $i = j$, then $\mathbf{D}^{MM}_{ij} = 0$: We have $\mathbf{D}^{MM}_{ii} = \mathbf{D}_{ii} = 0$ because the smallest maximal gap between every object and itself is zero. ii) If $\mathbf{D}^{MM}_{ij} = 0$, then $\mathbf{D}_{ij} = 0$ and $i = j$, because we have assumed that all the distinct pairwise distances are positive, i.e. zero base or Minimax pairwise distances can occur only if $i = j$.
2. $\forall i, j : \mathbf{D}^{MM}_{ij} \geq 0$. All the edge weights, i.e. the elements of $\mathbf{D}$, are non-negative. Thus the minimum of them, i.e. $\min(\mathbf{D})$, is also non-negative. Moreover, by definition we have $\mathbf{D}^{MM}_{ij} \geq \min(\mathbf{D})$. Hence, we conclude $\mathbf{D}^{MM}_{ij} \geq \min(\mathbf{D}) \geq 0$.

3. $\forall i, j : \mathbf{D}^{MM}_{ij} = \mathbf{D}^{MM}_{ji}$. By assumption $\mathbf{D}$ is symmetric, therefore, any path from i to j will also be a path from j to i, and vice versa. Thereby, their maximal weights and the minimum among different paths are identical.
4. $\forall i, j, k : \mathbf{D}^{MM}_{ij} \leq \max(\mathbf{D}^{MM}_{ik}, \mathbf{D}^{MM}_{kj})$. We show that otherwise a contradiction occurs. Suppose there is a triplet i, j, k such that $\mathbf{D}^{MM}_{ij} > \max(\mathbf{D}^{MM}_{ik}, \mathbf{D}^{MM}_{kj})$. Then, according to the definition of Minimax distance, the path from i to k and then to j must be used for computing the Minimax distance $\mathbf{D}^{MM}_{ij}$ which leads to $\mathbf{D}^{MM}_{ij} \leq \max(\mathbf{D}^{MM}_{ik}, \mathbf{D}^{MM}_{kj})$, i.e. a contradiction occurs.

On the other hand, it has been shown that every *ultrametric* induces an $\mathcal{L}_2^2$ embedding (Deza and Laurent 1992). Therefore, $\mathbf{D}^{MM}$ represents the pairwise squared Euclidean distances in a hidden vector space. □

Notice that we do not require $\mathbf{D}$ to induce a *metric*, i.e. the triangle inequality is not necessarily fulfilled. After satisfying the feasibility condition, there exist several ways to compute a squared Euclidean embedding. We exploit a method motivated in (Young and Householder 1938) and further analyzed in (Torgerson 1958). This method works based on centering $\mathbf{D}^{MM}$ to compute a Mercer kernel which is positive semidefinite, and then performing an eigenvalue decomposition, as following:

I) Center $\mathbf{D}^{MM}$ by

$$\mathbf{W}^{MM} \leftarrow -\frac{1}{2}\mathbf{A}\mathbf{D}^{MM}\mathbf{A}. \qquad (2)$$

$\mathbf{A}$ is defined as $\mathbf{A} = \mathbf{I}_N - \frac{1}{N}\mathbf{e}_N\mathbf{e}_N^T$, where $\mathbf{e}_N$ is a vector of length N with 1's and $\mathbf{I}_N$ is an identity matrix of size N.

II) Under this transformation, $\mathbf{W}^{MM}$ is positive semidefinite. Thus, we decompose $\mathbf{W}^{MM}$ into its eigenbasis, i.e. $\mathbf{W}^{MM} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$, where $\mathbf{V} = (v_1, ..., v_N)$ contains the eigenvectors v_i and $\mathbf{\Lambda} = \mathtt{diag}(\lambda_1, ..., \lambda_N)$ is a diagonal matrix of eigenvalues $\lambda_1 \geq ... \geq \lambda_d \geq \lambda_{d+1} = 0 = ... = \lambda_N$. Note that the eigenvalues are nonnegative, since $\mathbf{W}^{MM}$ is positive semidefinite.

III) Calculate the $N \times d$ matrix $\mathbf{Y}^{MM}_d = \mathbf{V}_d(\mathbf{\Lambda}_d)^{1/2}$, with $\mathbf{V}_d = (v_1, ..., v_d)$ and $\mathbf{\Lambda}_d = \text{diag}(\lambda_1, ..., \lambda_d)$.

Here, d shows the dimensionality of the Minimax vectors, i.e. the number of Minimax features. The Minimax dimensions are ordered according to the respective eigenvalues and thereby we might choose only the first most representative ones, instead of taking all.

Embedding of Collective Minimax Matrices

We extend our generic framework to the cases that multiple pairwise Minimax matrices are available, instead of a single matrix. Then, the goal would be to find an embedding of the objects into a new space wherein their pairwise squared Euclidean distances are related to the collective Minimax distances over different Minimax matrices. Such a scenario might be interesting in several situations:

1) There might exist different type of relations between the objects, such that each relation renders a separate graph. Then, we compute several pairwise Minimax distances, each for a specific relation.

2) Minimax distances might fail whenever for example few noise objects connect two compact classes. Then, the inter-class Minimax distances become very small, even if the objects from the two classes are connected via only few outliers. To solve this issue, in a similar way to model averaging, one could use the higher order, i.e. the second, third, ... Minimax distances, e.g. the second smallest maximal gap. Then, there will be multiple pairwise Minimax matrices each representing the k^{th} Minimax distance.

3) In many real-world applications, we encounter high-dimensional data. In this setting, the classes might be hidden in some unknown subspace instead of the whole space, such that they are disturbed in the high-dimensional space due to the curse of dimensionality. Minimax distances rely on the existence of well-connected paths, whereas such paths might be very sparse or fluctuated in high dimensions. Thereby, in a similar spirit to ensemble methods, it is natural to seek for connectivity paths and thus for Minimax distances in some subspaces of the original space. However, investigating all the possible subspaces is computationally intractable as the respective cardinality scales exponentially with the number of dimensions. Hence, we propose an approximate approach based on computing Minimax distances for each dimension, which leads to having multiple Minimax matrices.

In all the aforementioned cases, we need to deal with different matrices of Minimax distances computed for the same set of objects. Then, the next step is to compute an embedding that represents the whole collection of pairwise Minimax matrices. For this purpose, after computing the different Minimax distances, we center them via the transformation in Eq. 2 and then sum them up to obtain a single matrix. This matrix is positive semidefinite, thus it corresponds to an $\mathcal{L}_2^2$ embedding. Hence, finally, we apply the previously mentioned embedding to obtain a set of vectors representing the collection of Minimax distances.

Efficient calculation of dimension-specific Minimax distances. In this paper, we particularly study the use of collective Minimax embedding for high-dimensional data (called the *dimension-specific* variant). The dimension-specific variant requires computing pairwise Minimax distances for each dimension, which can be computationally expensive. However, in this setting, the objects stay in an one-dimensional space. A main property of one-dimensional data is that sorting them immediately gives a minimum spanning tree. We, then, compute the pairwise distances for each pair of consecutive objects in the sorted list to obtain the edge weights of the minimum spanning tree. Finally, we compute the pairwise Minimax distances from the minimum spanning tree.

Experiments

We experimentally study the performance of Minimax classification on a variety of synthetic and real-world datasets and illustrate how the use of Minimax distances as an intermediate step improves the results. In each dataset, each object is represented by a vector. We compute the pairwise squared Euclidean distances between the vectors to construct the base distance matrix $\mathbf{D}$. We use Logistic Regression (LogReg) and Support Vector Machines (SVM) as the baseline methods and investigate how performing these methods on the vectors induced from Minimax distances improves the accuracy of prediction. With SVM, we examine three different kernels: i. linear (lin), ii. radial basis function (rbf), and iii. sigmoid (sig), and choose the best result. With Minimax distances, we only use the linear kernel, since we assume that Minimax distances must be able to capture the correct classes, such that they can be then discriminated via a linear separator.

Experiments with synthetic data

We first perform our experiments on two synthetic datasets, called: i) DS1 (Chang and Yeung 2008), and ii) DS2 (Veenman, Reinders, and Backer 2002), which are shown in Figure 1. The goal is to demonstrate the superior ability of Minimax distances to capture the correct class-specific structures, particularly when the classes have different types (DS2), compared to kernel methods. Table 1 shows the accuracy scores for different methods. The standard SVM is performed with three different kernels (lin, rbf and sig), and the best choice which is the rbf kernel is shown. As mentioned, with Minimax distances, we only use the linear kernel. We observe that performing classification on Minimax vectors yields the best results, since it enables the method to better identify the correct classes. The datasets differ in the type and consistency of the classes. DS1 contains very similar classes which are Gaussian. But DS2 consists of classes which differ in shape and type. Therefore, for DS1 we are able to find an optimal kernel (rbf, since the classes are Gaussian) with a *global* form and parametrization, which fits with the data and thus yields very good results. However, in the case of DS2, since classes have different shapes, then a single kernel *is not* able to capture correctly all of them. For this dataset, LogReg and SVM with Minimax vectors perform better, since they enable to adapt to the class-specific structures. Note that in the case of DS1, using Minimax vectors is equally good to using the optimal rbf kernel. Remember that, to Minimax vectors, we apply only SVM with a linear kernel. Figures 1(c) and 1(d) show the accuracy and eigenvalues w.r.t. different dimensionality of Minimax vectors. As mentioned earlier, the dimensions of Minimax vectors are sorted according to the respective eigenvalues, since a larger eigenvalue indicates a higher importance. By choosing only few dimensions, the accuracy attains its maximal value. We will elaborate in more detail on this further on.

Table 1: Accuracy of different methods on synthetic datasets. Minimax measure improves the results when the classes have different shapes and types (e.g. DS2).

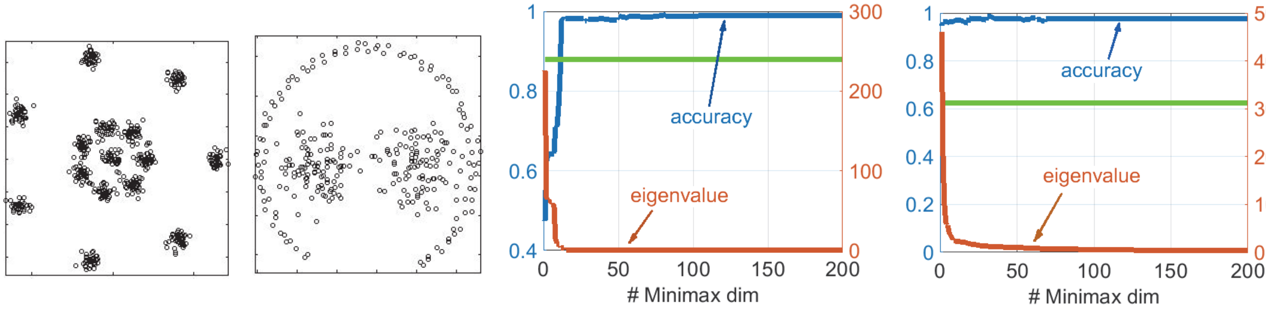
	standard		Minimax	
dataset	SVM-rbf	LogReg	SVM-lin	LogReg
DS1	0.9924	0.8066	0.9917	0.9918
DS2	0.9295	0.6252	0.9950	0.9983

Experiments on real-world data

We perform our real-world experiments on twelve datasets from different domains, selected from the UCI repository (Lichman 2013): (1) *Balance Scale*: contains 625 observations modeling 3 types of psychological experiments. (2) *Banknote Authentication*: includes 1372 images taken from genuine and forged banknote-like specimens (number of classes is 2). (3) *Cloud*: consists of 1024 10-dimensional vectors, each dimension representing a specific parameter. (4) *Contraceptive Method*: contains information of 1473 women, where the three classes are about the pregnancy status. (5) *Glass Identification*: contains 6 types (classes) of glass w.r.t the oxide content. The number of instances is 214. (6) *Haberman Survival*: contains the survival of 306 patients who had surgery for breast cancer. The number of classes is 2. (7) *Hayes Roth*: is about a study on human subjects which contains 160 instances and 3 classes. (8) *Ionosphere*: includes 351 34-dimensional instances collected from radars and organized into 2 classes. (9) *Lung Cancer*: describes 3 types of pathological lung cancer, including 32 instances each with 56 dimensions. (10) *Perfume*: consists of odors of 20 different perfumes (classes), where the data is collected via OMX-GR sensor. There are in total 560 measurements. (11) *Skin Segmentation*: the original dataset contains $245,057$ instances generated using skin textures from face images of different people. However, to make the classification task more difficult, we pick only the first $1,000$ instances of each class (to decrease the number of objects per class). The target variable is skin or non-skin sample, i.e. the number of classes is 2. (12) *User Knowledge*: describes 403 students' knowledge level (4 classes) about the subject of Electrical DC Machines.

Accuracy scores. Table 2 shows the results for different methods applied to the datasets, when 60% of the objects are used for training. We have repeated the random split of the data for 20 times and report the average results. The accuracy scores and the ranking of different methods are very consistent among different splits, such that the standard deviations are low. We observe that often performing the classification methods on Minimax vectors improves the classification accuracy. In only very few cases the standard setup outperforms (slightly). In the rest, either the Minimax vectors or the dimension-specific variant of Minimax vectors yield a better performance. In particular, the Minimax variant is more appropriate for low dimensional data, whereas the dimension-specific Minimax variant outperforms on high-dimensional data.

Model order selection. Choosing the appropriate number of dimensions for Minimax vectors (i.e. their dimensionality) constitutes a model order selection problem. We study in detail how the dimensionality of the Minimax vectors affects the results. Figure 2 shows the accuracy scores for Minimax-LogReg applied to four of the datasets w.r.t. different number of dimensions (the other datasets behave similarly). The dimensions are ordered according to their importance (informativeness). Choosing a very small number of dimensions might be insufficient since we might loose some informative dimensions, which yields underfitting. By increasing the dimensionality, the method extracts more sufficient classes in

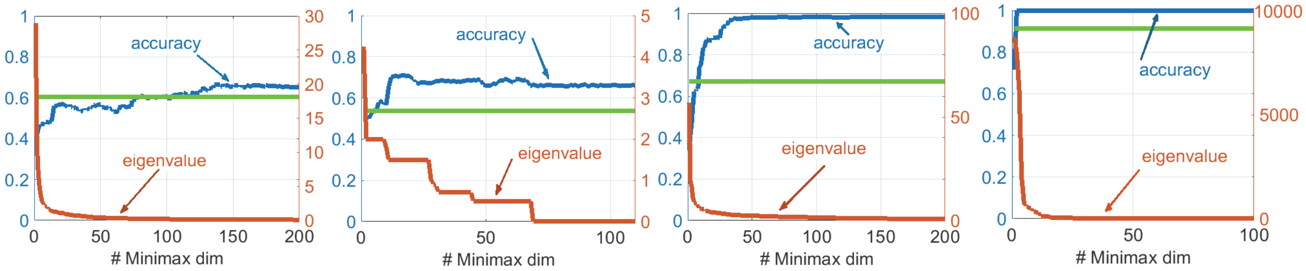


(a) DS1 (b) DS2 (c) results on DS1 (d) results on DS2

Figure 1: Illustration of DS1, DS2 and the accuracy. The accuracy scores (shown for LogReg-MM) are stable w.r.t. the dimensionality of the Minimax vectors (Figures 1(c) and 1(d)). The straight green line shows the accuracy for the base LogReg.

Table 2: Accuracy scores of different methods on real-world datasets, when 60% of the data is used for training. Performing the classification algorithm on Minimax vectors improves the results.

	standard				Minimax		dim.spec. Minimax	
data	SVM-lin	SVM-rbf	SVM-sig	LogReg	SVM-lin	LogReg	SVM-lin	LogReg
Balance Scale	0.8709	0.8974	0.4468	0.8687	0.6187	0.6086	0.9211	0.9739
Banknote Authentication	0.9876	1.0000	0.5577	0.9872	0.9989	1.0000	0.8847	0.9827
Cloud	0.9988	0.5788	0.5349	0.9988	1.0000	1.0000	1.0000	1.0000
Contraceptive Method	0.5190	0.5533	0.4250	0.5107	0.5647	0.5747	0.5392	0.5389
Glass Identification	0.5924	0.6012	0.3371	0.6053	0.5971	0.6671	0.4918	0.6347
Haberman Survival	0.6893	0.7230	0.7344	0.7426	0.7434	0.7377	0.7418	0.7352
Hayes Roth	0.6058	0.7750	0.3596	0.5365	0.7038	0.7115	0.8635	0.8558
Ionosphere	0.8779	0.9300	0.6536	0.8621	0.9457	0.9450	0.8843	0.9336
Lung Cancer	0.6917	0.7500	0.7500	0.6917	0.7750	0.7500	0.8333	0.8333
Perfume	0.7783	0.9318	0.1498	0.6933	0.9865	0.9870	0.9798	0.9830
Skin Segmentation	0.9287	0.9693	0.7635	0.8980	0.9994	0.9994	0.9906	0.9919
User Knowledge	0.7835	0.7233	0.3019	0.8612	0.5893	0.6757	0.6010	0.8592



(a) *Glass Identification* (b) *Hayes Roth* (c) *Perfume* (d) *Skin Segmentation*

Figure 2: Accuracy score of LogReg-MM applied to different datasets when we choose different number of dimensions of Minimax vectors. The straight green lines show the base LogReg result.

the data, thus the accuracy improves. We note that this phase occurs for a very wide range of choices of dimensions. However, by increasing the number of dimensions even more, we might include non-informative or noisy features, where then the accuracy stays constant or decreases slightly, due to overfitting. However, an interesting advantage of this approach is that the overfitting dimensions (if there exists any) have a very small eigenvalue, thus their impact is negligible. This analysis leads to a simple and effective model order selection principle: Fix a very small threshold and pick the dimensions whose respective eigenvalues are larger than the threshold. However, the exact value of the threshold may not play a critical role (or it can be estimated via using an additional validation set).

Conclusion

We developed a framework to apply Minimax distances to any classification algorithm that works on numerical data. Our approach is performed in two steps: First, we compute the pairwise Minimax distances via obtaining a minimum panning tree. Then, we compute an embedding of the objects into a new vector space, such that their pairwise Minimax distance in the original space equals to their squared Euclidean distance in the new space. This embedding provides to apply any numerical classification method or compute a kernel. We also studied the cases where the data is represented in a very high-dimensional space. We first proposed an efficient method to compute the pairwise Minimax distances at each separate dimension. Then, we used an embedding which computes a set of vectors that correspond to the all Minimax matrices at different dimensions. Finally, we investigated our framework on Logistic Regression and SVM and showed that the use of Minimax distances improves the accuracy scores, as well as obviates the need for choosing critical parameters which might be nontrivial.

References

Aho, A. V., and Hopcroft, J. E. 1974. *The Design and Analysis of Computer Algorithms*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1st edition.

Chang, H., and Yeung, D.-Y. 2008. Robust path-based spectral clustering. *Pattern Recogn.* 41(1):191–203.

Chebotarev, P. 2011. A class of graph-geodetic distances generalizing the shortest-path and the resistance distances. *Discrete Appl. Math.* 159(5):295–302.

Chehreghani, M. H. 2016. K-nearest neighbor search and outlier detection via minimax distances. In *Proceedings of the 2016 SIAM International Conference on Data Mining, Miami, Florida, USA, May 5-7, 2016*, 405–413.

Cormen, T. H.; Stein, C.; Rivest, R. L.; and Leiserson, C. E. 2001. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition.

Deza, M., and Laurent, M. 1992. Applications of cut polyhedra. Technical Report BS-R9221, Centrum voor Wiskunde en Informatica (CWI). Amsterdam (NL).

Fischer, B., and Buhmann, J. M. 2003. Path-based clustering for grouping of smooth curves and texture segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* 25(4):513–518.

Fouss, F.; Pirotte, A.; Renders, J.-M.; and Saerens, M. 2007. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Trans. on Knowl. and Data Eng.* 19(3):355–369.

Fouss, F.; Francoisse, K.; Yen, L.; Pirotte, A.; and Saerens, M. 2012. An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification. *Neural Networks* 31:5372.

Hofmann, T.; Schlkopf, B.; and Smola, A. J. 2006. A review of kernel methods in machine learning.

Hu, T. 1961. The maximum capacity route problem. *Operations Research* 9:898–900.

Kim, K.-H., and Choi, S. 2007. Neighbor search with global geometry: a minimax message passing algorithm. In *ICML*, 401–408.

Kim, K.-H., and Choi, S. 2013. Walking on minimax paths for k-nn search. In *AAAI*.

Kschischang, F. R.; Frey, B. J.; and Loeliger, H. A. 2006. Factor graphs and the sum-product algorithm. *IEEE Trans. Inf. Theor.* 47(2):498–519.

Leclerc, B. 1981. Description combinatoire des ultramtriques. *Mathmatiques et Sciences Humaines* 73:5–37.

Lichman, M. 2013. UCI machine learning repository.

Luxburg, U. 2007. A tutorial on spectral clustering. *Statistics and Computing* 17(4):395–416.

Nadler, B., and Galun, M. 2007. Fundamental limitations of spectral clustering. In *in Advanced in Neural Information Processing Systems 19*, 1017–1024.

Prim, R. C. 1957. Shortest connection networks and some generalizations. *The Bell Systems Technical Journal* 36(6):1389–1401.

Shawe-Taylor, J., and Cristianini, N. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Tenenbaum, J. B.; de Silva, V.; and Langford, J. C. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290(5500):2319.

Torgerson, W. 1958. *Theory and methods of scaling*. Wiley.

Veenman, C. J.; Reinders, M.; and Backer, E. 2002. A maximum variance cluster algorithm. *IEEE Trans. Pattern Anal. Mach. Intell* 24:1273–1280.

Weinberger, K. Q., and Saul, L. K. 2009. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.* 10:207–244.

Yen, L.; Saerens, M.; Mantrach, A.; and Shimbo, M. 2008. A family of dissimilarity measures between nodes generalizing both the shortest-path and the commute-time distances. In *KDD*, 785–793.

Young, G., and Householder, A. 1938. Discussion of a set of points in terms of their mutual distances. 3(1):19–22.

Zadeh, R., and Ben-David, S. 2009. A uniqueness theorem for clustering. In *UAI*, 639–646.