Knowledge-Based Systems 21 (2008) 692–703

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Local distance-based classification

Manuel Laguía [a,*], Juan Luis Castro [b]

[a] Univ. de Cádiz, E.S. Ingeniería de Cádiz, Dept. Lenguajes y Sistemas Informáticos, 11002 Cádiz, Spain
[b] Univ. de Granada, E.T.S. Ingeniería Informática, Dept. C. Comp. e Inteligencia Artificial, 18071 Granada, Spain

ARTICLE INFO

Article history:
Received 27 July 2007
Accepted 24 March 2008
Available online 7 April 2008

Keywords:
Distance measure
k-NN
Classification
Similarity
Machine learning

ABSTRACT

In this paper, we have introduced a new method in which every training point learns what is happening in its neighborhood. So, a hyperplane is learned and associated to each point. With this hyperplane we can define the *bands distance*, a distance measure that bring closer or move away points depending on its classes. We have used this new distance in classification tasks and have performed tests over 68 datasets: 18 well-known UCI-Repository datasets, one private dataset, and 49 ad hoc synthetic datasets. We have used 10-fold cross-validation and, in order to compare the results of the classifiers, we have considered the mean accuracy and have also performed a paired two-tailored *t*-Student's test with a significance level of 95%. The results are encouraging and confirm the good behavior of the new proposed classification method. The bands distance has obtained the best overall results with 1-NN and *k*-NN classifiers when compared with other distances. Finally, we extract conclusions and outline some lines of future work.

1. Introduction

It is usual to define similarity between two objects by means of distances, because we are familiar with them and, from a practical point of view, they are intuitive and easy to define. Thus, popular classification algorithms like 1-NN and *k*-NN [8,5,6] employ distances to search “the most similar cases” to the new one, based on the value of some characteristics, in order to assign a class. Much work has been done about similarity [12,15,16], and about distances and distance based classification methods [13,20]. Some variants of 1-NN methods have been studied in literature, including the IBx series [1,2] to reduce the storage requirements and increase noise tolerance; the nested-generalized exemplar (NGE) theory [17] where hyper-rectangles are used instead of points; the value difference metric (VDM) [18], the modified value difference metric (MVDM) [4], the heterogeneous value difference metric (HVDM) [22], and the simplified value difference metric (SVDM) [7] that statistically derive distances for nominal attributes based on the overall similarity of classification of all instances for each possible value of each feature. Another interesting distance is the local asymmetrically weighted similarity metric (LASM) [14] which defines a local distance that varies along the space and is asymmetric.

This kind of algorithms almost always employ one of the classical distances of the Geometry, mainly the Euclidean distance. It is clear that if we consider only usual distances, then some natural kinds of similarity (and distance) relations cannot be obtained [9], and there are many problems where the examples are grouped according to other patterns. For example, if we know some companies' incomes and expenses and the class is the company's profit, the examples belonging to the same class (and in that sense, similar) are along a line. This is one example of a wide number of problems where the instances are, roughly speaking, grouped into bands, and usual distances fail.

Accordingly, we propose a different approach in this work. First, we suggest understanding distance functions in a wide sense. We need a function that measures similarity or dissimilarity between objects, and provides low values for cases with equal or similar classes. But, in general, we do not need that it satisfies the conditions of a geometrical distance. So we propose flexibility and employ “distance functions” adapted to the specific problem we deal. Second, we propose employing local distance functions and go beyond of the idea of searching the nearest points to the new case *e*. We suggest a training stage in which each point learns a band or hyperplane that passes through itself and better fits the distribution of points in its surrounding area. Then, given a new case *e*, each known point can provide a measure of how distant is *e* from its point of view (according to its band). The usual approach searches the nearest points from *e*, but now each point “says how close or far it sees *e*”.

This approach has several advantages. First, we can see it like a local function of distance: each known point has its own measure of distance that can vary across the space to properly fit the special characteristics of different regions. Second, by hyperplanes we can

[*] Corresponding author. Tel.: +34 956015731; fax: +34 956015101.
E-mail addresses: manuel.laguia@uca.es (M. Laguía), castro@decsai.ugr.es (J.L. Castro).

0950-7051/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.knosys.2008.03.050

locally approximate almost every shape…and we have one hyperplane associated to each point. Third, each point can learn the hyperplane that locally minimizes the distance to points of its class, and even maximizes the distance to points of other classes. Thus the point tends to assign lower distances to points in the direction of the hyperplane (the direction of points of its class) and further distances in the perpendicular direction. With the usual approach we cannot bring closer or move away points depending on its classes because the new case e still has not got a class (we are just trying to assign a class to e!). And finally, we now have much more knowledge than raw points, and could extract information from the hyperplanes that each point has learned.

2. An algorithm to learn bands or hyperplanes

Usual distances are useful in a lot of situations, but sometimes other kind of distances is more appropriated. In a previous work [9], we have presented a distance that groups points according to bands along one hyperplane H in R^n (a line in R^2) $d_{\alpha,\mathrm{wide}}(x,y) : R^n \times R^n \rightarrow R_0^+$ as (Fig. 1):

$$d_{\alpha,\mathrm{wide}}(x,y) = \mathrm{wide}\left|\sum_{i=1}^{n} \cos\alpha_i(x_i - y_i)\right| \tag{1}$$

where $\alpha = (\alpha_1,\ldots,\alpha_n) \in [0, 2\pi]^n$ is the set of angles between the axis and the unitary vector $v = (v_1,\ldots,v_n)$ that is perpendicular to the hyperplane H (Fig. 1). *wide* $\in R^+$ controls the width of the band of points at a given distance (lower values imply approaching points to H), and $|.|$ is the absolute value in R. Moreover, v verifies that $\sum_{i=1}^{n} v_i^2 = 1$, so there are only n degrees of freedom in the parameters. Notice that $(x_i - y_i)$ can be lower, equal or greater than 0. We use R^n for simplicity. In symbolic domains $(x_i - y_i)$ represents the partial distance between x_i and y_i in that domain. This distance is not a mathematical metric, but a mathematical pseudo-metric, because $d(x,y) = 0 \not\Rightarrow x = y \ \forall\, x,y \in D$.

2.1. Band direction

Putting the band width aside, we want that every training point learn what is happening in its neighborhood. If P is the set of known points, we want that every point $p \in P$ learn a band or hyperplane that goes through p and better fits the points rounding it, i.e., the hyperplane H that minimizes the distance $d_{\alpha,wide}$ to the surrounding points. We define the adjustment of a hyperplane $H = v^{\perp}$ as:

$$\mathrm{adjust}_H(p,P) = \sum_{x_i \in P}\left(\sum_{j=1}^{n}(x_{ij} - p_j)v_j\right)^2 \tag{2}$$

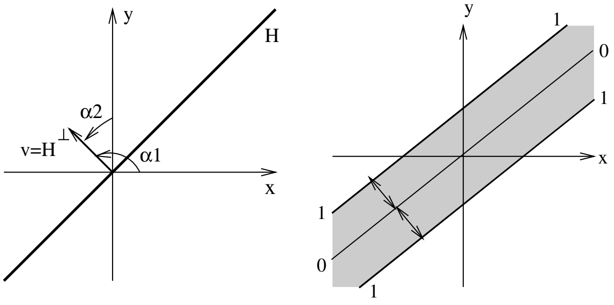


Fig. 1. Definition of a band in R^2 and an example of band along a hyperplane in R^2.

where n is the number of attributes or dimensions, p_j is the jth attribute or coordinate of p, x_{ij} is the jth attribute of x_i, and $v_j = \cos\alpha_j$ is the jth director cosine of H.

Now we can express the hyperplane H that goes through a point and better fits the points around it as the hyperplane with the lowest adjustment value.

We have performed several experiments to observe the behavior of this way of selection of hyperplanes. For instance, we can generate random points of one horizontal band of width 0.2 and one vertical band of width 0.3 (Fig. 2). The points learn reasonably well the direction of the band they are included, but when we merge these two bands, the points have a tendency towards the center of the set of points, instead of learning the direction of the band they are placed.

In this example we clearly show that this method is too global, and the direction that a point selects is heavily affected by distant points. So, we must employ a more local method. We have proposed a variant, of this prior basic method, that takes into account the distance of the points:

$$\mathrm{adjust\_local}_H(p,P) = \sum_{x_i \in E(p,\varepsilon)}\left(\sum_{j=1}^{n}(x_{ij} - p_j)v_j\right)^2 \mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)} \tag{3}$$

where $E(p,\varepsilon) = \{x \in P \mid d(x,p) \leqslant \varepsilon\}$ and $d(x_i,p)$ is the Euclidean distance between x_i and p, i.e., $\sqrt{\sum_{j=1}^{n}(x_{ij} - p_j)^2}$. This variant learns well the single bands, and the bottom-right graph of Fig. 2 shows its behavior when merging two bands.

The new method of Eq. (3) has a parameter ε to permit softly controlling the locality of this method from very local to very global (lower to greater values of ε). With a high value of ε, this method exhibits a behavior similar to the previous one without the kernel function. Decreasing ε, it obtains bands with a more local behavior.

We have here two alternatives to choose the ε value. ε can be a fixed real value, constant for the entire case base (we could directly provide it or it could be computed as the result of some expression that includes the idiosyncrasy of each case base). Or, ε can be a real value different for each point, calculated from the information on

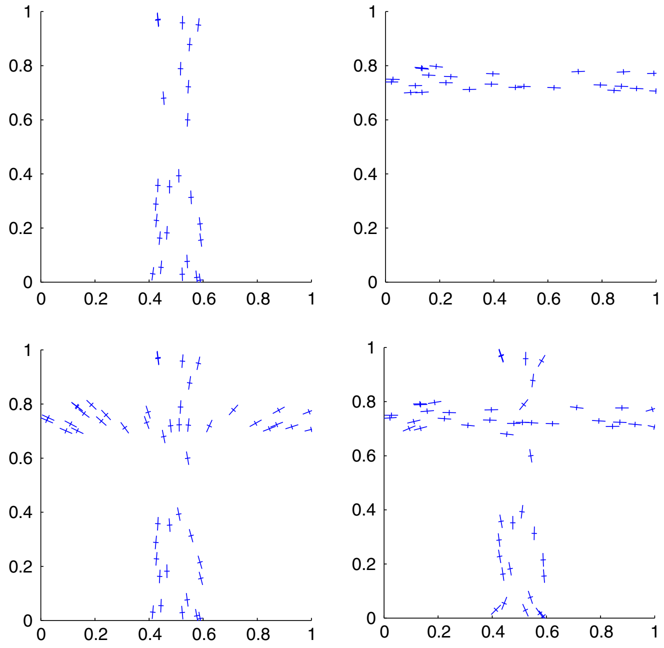


Fig. 2. Bands learned by each point of one vertical and one horizontal band and their union in R^2. For each point we show the direction of its band with the longest segment.

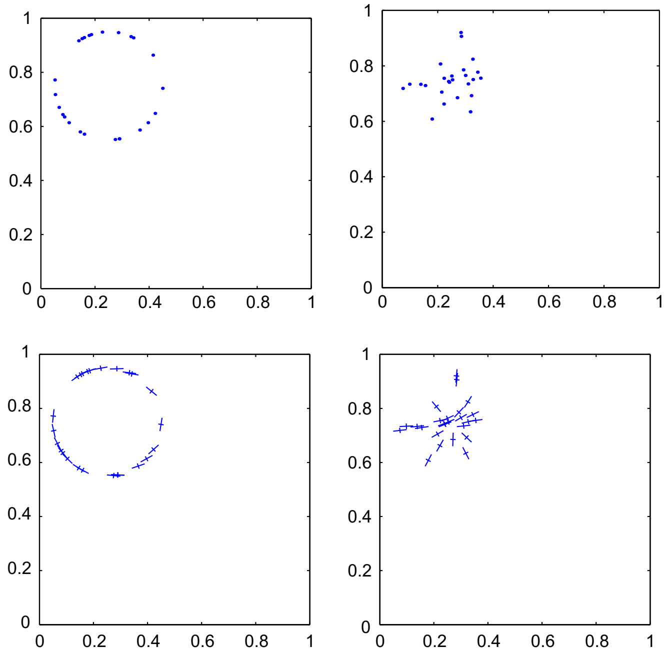
the surroundings of the point, i.e. each point has his own ε that includes the special characteristics of that region. Anyway, the value of ε must be large enough to include the information of the surroundings of the point (ideally the group or cluster where it is placed), but not too large, where it is affected by further points that are outliers or belong to other groups of points or clusters.

We have performed some experiments in order to find a way to obtain a value of ε to each point, but we have find big difficulties and the results were not conclusive. This part of the investigation is directly related to clustering problems and it is outside the main line of our work. So, we have decided to postpone the automatic determination of the value of ε as a line of future work. Here it is feasible to employ techniques yet known in the clustering area in order to determine the size of the cluster where the point is, and choose a value of ε that harvest the information of the points of the cluster but without considering information of others clusters of points.

Figs. 3–5 show the direction of the bands learned by some randomly generated points. Above is showed the position of each random point, and below the direction of the band learned by each point (here, each point is founded in the intersection of two segments of different length, the longest one graphically shows the direction of the band).

In Fig. 3a the points have been randomly chosen along a circumference. They have quite small values of ε, so that bands are quite local. The bottom half of the figure shows that the direction is chosen bearing to the nearest points, and a circumference is "drawn" or locally approximated by means of a set of small segments. If the value ε were larger, the points would tend to point to the center of the circumference. Fig. 3b shows randomly chosen points inside a circle. The cloud of points is quite compact and the bands are pointing to the center of the cloud (with big enough ε values).

Fig. 4 shows three examples with 50 randomly generated points over the $[0,1] \times [0,1]$ square without any restriction. If each point learned the band that better fits its surroundings, it will tend to choose directions pointing to close points. It is trying to find bands of points or figures formed by the points. This fact is very similar to Psychological tests, where a set of abstract pictures is showed to a character that must say what she see in those pictures. The response of two characters may be completely different, because subjectivity is at stake. If we show the clouds of points of the Fig. 4 to a character, and we asked her to try to find shapes or figures with these points, she could answer anything. This is exactly what the bands method is doing. For instance, in the left example of Fig. 4, if we look at the below graph, the bands of the points placed on the upper-left corner, it seems like if there was a circumference or an oval there, and a wide and irregular band from the upper-right corner to the bottom-left corner of the picture. The same happens with the other two examples of Fig. 4, where it could be recognized a crushed X and different clouds of points.



(a) Points of a circumference (b) Points of a circle

Fig. 3. Bands learned by random points in a circumference and a circle. Above it is showed the position of each point, and below the direction of each band (the longest segment).

Another interesting experiment is showed in Fig. 5, where the points are randomly generated inside two squares. In Fig. 5a the two squares are nearly superimposed and the points select their bands like if they belong to only one group of points (and actually there is only one group). In Fig. 5b the squares share a little area, so it seems that the cloud of points is stretched, and it looks like a band of points. The points tend to point according to the direction of the band, and some points head to the center of the cloud. If we move away the clouds of points, they start to behave like two different clouds of points. In Fig. 5c they are completely separated and distant, and the points tend to point to the center of their respective cloud. In this example the points learn that they are in two different clouds, and one cloud has no influence over the other one. If there are outliers they will have no influence over the clouds of points. The outliers could select any direction, but always more influenced by nearest points.

Probably a human observer would also say that there is only one cloud of points in Fig. 5a, there is one stretched cloud in Fig. 5b, and there are two clearly differentiated clouds in Fig. 5c.

2.2. Band direction in multiclass problems

In Eq. (3), the ε values that include other classes points could be penalized, so that, ε will tend to collect information only about the cloud where the point is placed. This could be a good idea if we do not consider different classes or have no information about classes. But in problems with different classes we would prefer choosing a bigger ε value that takes into account neighbors of other classes.

In order to determine the optimal hyperplane we want to choose the direction of nearby points of the same class (bring near same class points) and run away from directions where there are other classes points (move further away, as far as possible, other classes points). To achieve this double objective we propose that the hyperplane H learned by each point minimize the distance $d_{\alpha,\text{wide}}$ of the surrounding points of the same class and maximize the distance $d_{\alpha,\text{wide}}$ of the surrounding points of other classes. This latter objective is equivalent to minimize the distance between the points of other classes and the direction perpendicular to H, i.e. minimize the distance to $v = H^{\perp}$. We define the *multiclass adjustment* of a hyperplane $H = v^{\perp}$ as:

$$\begin{aligned}\text{adjust\_multi}_H(p,P) = {} & \sum_{x_i\in E^{=}(p,\varepsilon)} \left(\sum_{j=1}^{n}(x_{ij}-p_j)v_j\right)^2 \mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)} \\ & + F\sum_{x_i\in E^{\neq}(p,\varepsilon)} \Bigg(\Bigg(\sum_{j=1}^{n}(x_{ij}-p_j)\Bigg)^2 \\ & - \left(\sum_{j=1}^{n}(x_{ij}-p_j)v_j\right)^2\Bigg)\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\Bigg) \qquad (4)\end{aligned}$$

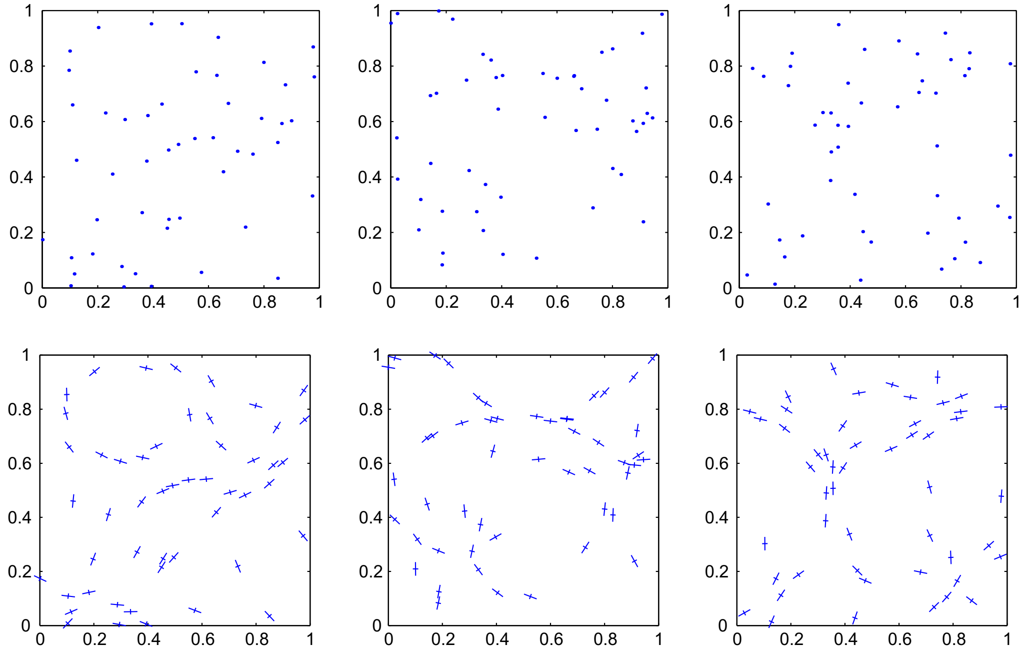
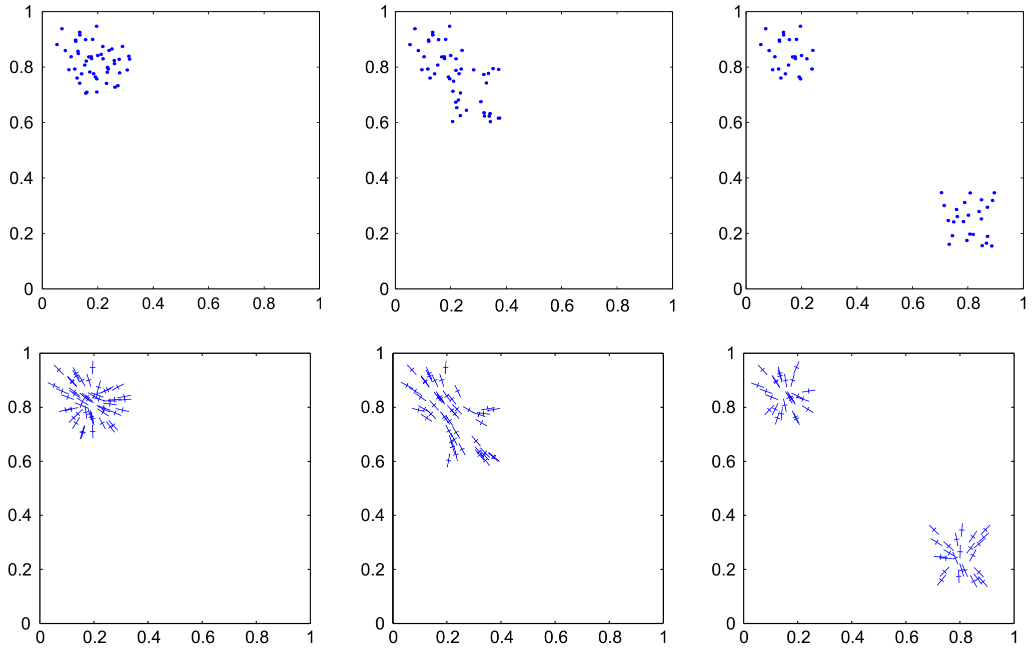


Fig. 4. Bands learned by randomly generated points. Above it is showed the position of each point, and below the direction of each band (the longest segment).



(a) Superimposed squares

(b) squares sharing some area

(c) Distant squares

Fig. 5. Bands that learn random points from two squares. Above it is showed the position of each point, and below the direction of each band (the longest segment).

where $F \in R_0^+$, n is the number of attributes, $E^{=}(p,\varepsilon)$ is the set of points of the same class that p, $E^{\neq}(p,\varepsilon)$ is the set of points of other classes, p_j is the jth attribute of p, x_{ij} is the jth attribute of x_i, and $v_j = \cos\alpha_j$ is the jth director cosine of H. $\sum_{x_i \in E^{=}(p,\varepsilon)}$ only collects information from points of the class of p, and $\sum_{x_i \in E^{\neq}(p,\varepsilon)}$ from other classes.

F is a fixed real value to permit controlling the influence given to points of other classes, i.e., we have one new parameter to control the desired strength or degree in which the band must avoid directions where there are points of other classes. Choosing $F = 0$ is equivalent to ignore other classes points and learn the hyperplane only with the points of the same class. Choosing $F = 1$ attaches the same weight or importance to choose the direction where there are points of the same class and avoid other classes points. Choosing a very high value of F means that we want for the band to run from directions of points of other classes, even though the hyperplane do not point at points of the same class.

Now, again, we can express the hyperplane H that goes through p and better fits the points around p as the hyperplane with the lowest adjustment value adjust_multi_H.

Finding the hyperplane H that minimizes adjust_multi$_H$ (Eq. (4)) is equivalent to find the vector $v = H^{\perp}$ that minimizes adjust_multi$_H$. This problem could be tackled as a constrained minimization problem where it is feasible to employ the Lagrangian-constrained minimization method to obtain a more straightforward solution. In this way we could turn the original problem into solving a system of $n + 1$ equations, easier to solve. Appendix A shows in detail the minimization of adjust_multi$_H$. In some datasets the number of attributes, and accordingly the number of variables, could be high. So, in the experiments we have employed a numerical approximation method to the solution.

2.3. Band width and length

Once we have chosen the ε value and $H = v^{\perp}$ for each point, the problem of the width of the band arises. We do not want infinity bands like the showed in Fig. 1, because in most of the real problems the bands are bounded, and we do not want to assign a distance nearly 0 to a distant point that is placed there by any chance. We propose to employ two real parameters r and R that control, respectively, the width and length of the band. We compute the distance between a point p' and the hyperplane H that goes through the point p as:

$$d_{r,R}(p',H) = \frac{d(p',H)}{r} + \frac{d(p',H^{\perp})}{R} \tag{5}$$

where $d(p',H)$ is the (Euclidean) distance between the point and the hyperplane and $d(p',H^{\perp})$ is the (Euclidean) distance between the point and v, the perpendicular direction to the hyperplane. In this way, the points at a given distance shape a rhombus in R^2, two cones with shared bases in R^3, and two hypercones in R^n (Fig. 6).

Given the hyperplane H by means of the unitary vector $v = (v_1, v_2, \ldots, v_n)$ of its director cosines (equivalent to provide the hyperplane by means of its parametric equations $H \equiv v_1x_1 + v_2x_2+, \cdots, +v_nx_n = 0$), we can express the previous distances from the point $p' = (p'_1, p'_2, \ldots, p'_n)$ to H and $H^{\perp}$ as:

$$\begin{aligned} d(p',H) &= |v_1(p'_1 - p_1) + v_1(p'_2 - p_2)+, \cdots, +v_n(p'_n - p_n)| \\ &= \left|\sum_{i=1}^{n} v_i(p'_i - p_i)\right| \end{aligned} \tag{6}$$

$$\begin{aligned} d(p',H^{\perp}) &= \sqrt{\|p' - p\|^2 - d(p',H)^2} \\ &= \sqrt{\sum_{i=1}^{n} (p'_i - p_i)^2 - \left(\sum_{i=1}^{n} v_i(p'_i - p_i)\right)^2} \end{aligned} \tag{7}$$

where $|.|$ is the absolute value function, and $\|p' - p\|$ is the norm (or module) of the vector from the point p to p'. So it is possible to express Eq. (5) as:

$$\begin{aligned} d_{r,R}(p',H) = &\frac{|\sum_{i=1}^{n} v_i(p'_i - p_i)|}{r} \\ &+ \frac{\sqrt{\sum_{i=1}^{n} (p'_i - p_i)^2 - (\sum_{i=1}^{n} v_i(p'_i - p_i))^2}}{R} \end{aligned} \tag{8}$$

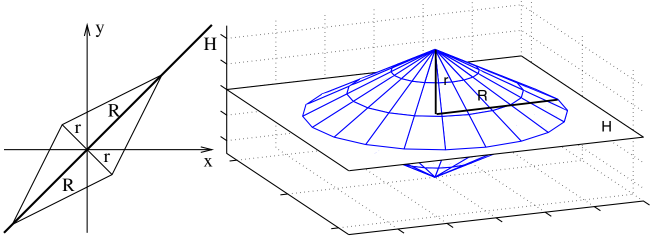


Fig. 6. Width and length of a band or hyperplane in R^2 and R^3.

After studying and testing several alternatives, we have decided choosing the pair of values $\langle r, R\rangle$ for each point according to the algorithm showed in Fig. 7. First, we choose r in a conservative way, without including points of other classes. If there are points of other classes at an (Euclidean) distance lower than ε, then r is chosen equal to the distance from the nearest point of other class to the hyperplane, else it is chosen $r = \frac{\varepsilon}{4}$. After that, we enlarge R, but without including points of other classes. If there are points of other classes inside the infinity band with radius r, then the R value is chosen so that the cone or rhombus goes through just at the middle of the distance to the hyperplane from that point (Fig. 8), else the band will have infinity length.

3. The experiments

We have used a large number of datasets to study the behavior of the different classifiers. All the classifiers are tested with exactly the same examples and using 10-fold cross-validation [19]. We have also performed a paired two-tailored t-Student's test with a significance level of 95%. When we use the term "statistically significant difference", or simply "significant difference", we will refer to the difference is statistically significant according to this t-Student's test.

3.1. Classifiers

To study the importance of the distance function in this kind of methods we consider six different measures: three (normalized) basic distances of the geometry (Euclidean, Manhattan and Chebychev) and their "correlated" variants, where each feature is weighted by its correlation with the class.

We consider some variants of k-NN [20,21] that differ in some characteristics [10,11]:

A (k-NN): basic method, used as reference.
B (ε-ball): impose a distance threshold ε on the set K of nearest neighbors.
C (k–NN heur): select the distance measure, from among a set of previously considered distances, that best performs for 1-NN on the same training set.
D (ε-ball$^{k\text{-NN}}$): impose a distance threshold ε on the set K of nearest neighbors, and whenever K is empty use k-NN.
E (ε-ball heur): impose a distance threshold ε on the set K of nearest neighbors. Select the distance measure, from among a set of previously considered distances, that best performs for 1-NN on the same training set.

```
  Given a set of points $P$, a point $p \in P$, a hyperplane
$H$ that goes through $p$, and a Þxed value $\varepsilon \in \mathrm{R}_0^+$.
      // Choose $r$ in a conservative way, so that it would
      // not include other classes points.
  Choose $q \in P$ such that $d(q,p) = min_{p\prime \in P}(p\prime, p)$ and
  class($p\prime$)$\neq$class($p$)
  if $d(q,p) \leq \varepsilon$
    then $r \leftarrow d(q,H)$.
    else $r \leftarrow \frac{\varepsilon}{4}$
    // $R$ is enlarged is a conservative way, without
    // including other classes points.
  Choose $q\prime \in P$ such that $d(q\prime,H) = min_{p\prime \in P}(p\prime, H)$ and
  class($p\prime$)$\neq$class($p$)
  if $d(q\prime,H) < r$
    then $R \leftarrow \frac{r\ d(q\prime,H)}{r - \frac{d(q\prime,H)}{2}}$
    else $R \leftarrow \infty$
```

Fig. 7. Algorithm to choose the radii r and R of the bands.

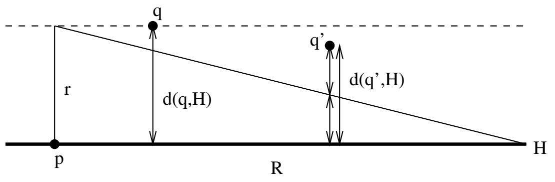


Fig. 8. Example of choice of the radii of a band or hyperplane in R^2.

F (ε-ball$^{k\text{-NN}}$ heur): impose a distance threshold ε on the set K of nearest neighbors, and whenever K is empty, use k-NN. Select the distance measure, from among a set of previously considered distances, that best performs for 1-NN on the same training set.

F1 (ε-ball$^{1\text{-}NN}$ heur): impose a distance threshold ε on the set K of nearest neighbors, and whenever K is empty, use 1-NN. Select the distance measure, from among a set of previously considered distances, that best performs for 1-NN on the same training set.

Bands (k-NN Bands): k-NN with the bands distance presented in the previous section.

It is useful to distinguish the special case F1, as we will discuss later. Classifiers A, B, and D employ the Euclidean distance. To select the distance in the classifiers that use the heuristics, we have also performed a separate 10-CV test with each training set.

The (weighted vote) k-NN, ε-ball, ε-ball$^{k\text{-NN}}$ and bands distance have one or two parameters that control their behavior. To estimate the value of the parameters we have performed a separate 10-CV test with each training set. In k-NN classifiers we have considered the odd values 1, 3, 5,..., 49 for k. In ε-ball and ε-ball$^{k\text{-NN}}$ methods we have considered 30 values in the [0,2] interval for ε, and the odd values 1, 3, 5,...,19 for k.

To use the bands distance we have employed the method of adjustment of hyperplanes to multiclass problems presented in Eq. 4. Here the learning of bands or hyperplanes has two parameters that control its behavior: ε and F. ε controls the locality of the learning method, to allow from a very local to a global learning. F controls the strength or degree to avoid directions where there are points of other classes and allows from ignore these points to avoid them at all costs.

To learn the bands distance we have considered the values 0, 1, 2, 4, 6, 8, 10, 15 for the F parameter, and different real values for the ε parameter, depending on the case base. In nearly all the synthetic datasets we have used ε values 0.1, 0.2, 0.3,...,1; and in most of the UCI-Repository datasets we have used ε values 0.2, 0.4, 0.6,...,2. In short, we have tested a great amount of combinations and employed a lot of computational time to obtain these results.

3.2. Datasets

We have used 68 datasets: 18 well-known UCI-Repository datasets [3], a reduced version of 1000 instances from the Granada handwritten digits[1] (Table 1), and 49 synthetic bases. The bases from the UCI-Repository are frequently used in scientific literature, facilitating comparisons with experimental results obtained by other classifiers introduced in other papers.

The synthetic datasets are useful to study classifiers in a controlled environment. They are constructed *ad hoc* over the $[0,1] \times [0,1]$ square and have 500 instances. We want to study the influence of the distribution of classes, so we consider (Fig. 9):

Bands (5, 10, and 20): point class is assigned according to 5, 10 or 20 horizontal bands.

Gaussian: point class is assigned according to four Gaussian distributions with variance 0.025.

Rings with constant area (3, 6, and 9): the space is divided into 3, 6 or 9 nested rings with equal areas and different radii, one class per ring. The total area of the regions has no influence and we can study the influence of the shape and number of classes.

Rings with constant radius (3, 6, and 9): the space is divided into 3, 6 or 9 nested rings with equal radii and different areas, one class per ring.

Sines (3, 6, and 9): they have two classes, and the decision boundary is a sine curve with 3, 6 or 9 $[0,2\pi]$-intervals fitted in $[0,1] \times [0,1]$.

Squares (2, 4, 6, and 8): the space is divided into a 2×2, 4×4, 6×6, or 8×8 grid. All the variants have four classes with the same amount of space; therefore the total area of the classes has no influence.

Table 1
Databases from the UCI-Repository used in the experiments

Index	Code	Domain	Size	Classes	No. of attributes	
					Numeric	Symbolic
1	IR	Iris plant	150	3	4	0
2	WI	Wine recognition	178	3	13	0
3	PI	PIMA diabetes	768	2	8	0
4	GL	Glass identification	214	6	9	0
5	CL	Cleveland	303	5	5	8
6	GD	Granada digits	1000	10	256	0
7	SN	Sonar	208	2	60	0
8	LD	Liver disorder	345	2	6	0
9	ZO	Zoo	101	7	1	15
10	TT	Tic–tac–toe	958	2	0	9
11	L7	Led 7	5000	10	0	7
12	L24	Led 24	5000	10	0	24
13	W21	Waveform-21	5000	3	21	0
14	W40	Waveform-40	5000	3	40	0
15	F1	Solar flare 1	1066	8	0	10
16	F2	Solar flare 2	1066	6	0	10
17	F3	Solar flare 3	1066	3	0	10
18	SO	Soybean	47	4	35	0
19	LR	Letter recognition	20,000	26	16	0

It seems reasonable that k-NN could have difficulties if the optimal k value is not constant along the whole space, i.e., if in some regions k must be greater than in others. In such conditions it may be feasible for ε-ball and bands methods to exhibit a better behavior because in some regions k-NN will take into account too many or too few points, but ε-balls and bands will consider only the "relevant" points (near enough but not too remote).

We have generated three variants for all the synthetic datasets (except for the Gaussian dataset) to test if k-NN methods are in trouble when the optimal k value varies along the space. Densely populated regions have more points and the optimal k value will be higher. So, in these variants, points are distributed with different probabilities across the space (Fig. 10), and we get a space with different point density (and optimal k value).

In the *uniform* variant the points are uniformly distributed along the space. In the *half* variant points are distributed in two clearly different regions: 30% in the left half of the space ($x < 0.5$) and the remaining 70% in the right half ($x \geqslant 0.5$). In the *progressive* variant the point's acceptance probability is proportional to the addition of its coordinates ($x + y$): the point density progressively increases from the bottom-left corner to the top-right one.

[1] The Granada handwritten digits dataset has 11,000 instances of handwritten digits, 256 numeric attributes (16×16 grid) and 10 classes. This database is private and has been yielded up by IPSA (Investigación y Programas S.A.).

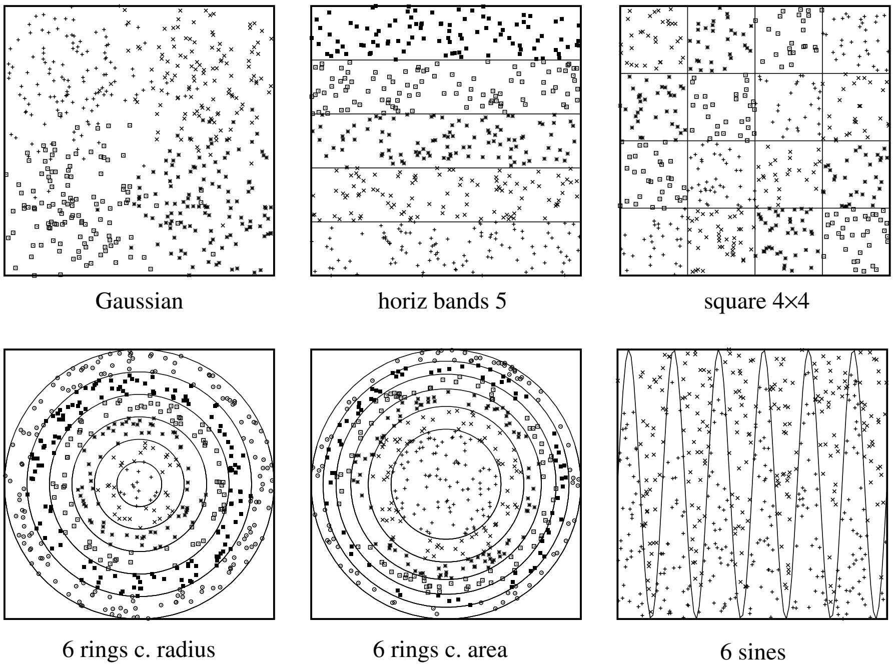


Fig. 9. Some synthetic datasets. Different symbols indicate different classes and lines represent the decision boundaries.

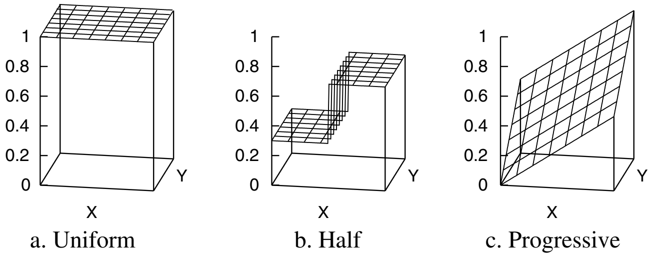


Fig. 10. Points distribution of the variants in the synthetic data sets.

4. The results

We use different ways to compare the behavior of the algorithms to avoid misleading results from a particular method. For each classifier we have calculated the mean accuracy, the improvement of mean accuracy vs. the basic method (Euclidean distance with 1-NN or *k*-NN), the mean accuracy broken down under different datasets, and we have also performed a paired two-tailored *t*-Student's test with a significance level of 95%.

4.1. Results with 1-NN

Table 2 shows the pairwise comparison using the *t*-Student's test with 1-NN using the typical distances of the Geometry and the new distance based on bands. In Tables 3 and 4 the comparison is broken down under UCI-Repository and synthetic datasets. These three tables show that 1-NN with the bands distance significantly improve the results of the other distances very frequently. This fact appears independently of the category taken into consideration: UCI-Repository, Synthetic and "all datasets".

Table 5 shows the mean accuracy of the classifiers broken down in UCI-Repository, Synthetic and "all datasets". The bands distance attains the best results in all the categories. So with no prior information about the dataset, bands distance is a good election.

Table 6 shows the number of datasets where the heuristics choose each distance. In other words, the number of bases in which each distance with the 1-NN classifier attains the best results with the UCI-Repository, Synthetic and all the datasets. Bands distance exhibit the best behavior by far: attains the best results in more than half of the datasets in each category. So, this table again indicates that with no prior information about the dataset is advisable to employ the bands distance.

Table 7[2] shows the detailed results obtained by each one of the 1-NN classifiers with these distances in the UCI-Repository datasets. Observing this table, we could point out the good results of the bands distance, higher than other distances in the Tic–Tac–Toe (+21.92%) and Waveform-21 (+8.10%) datasets, and lower in Granada Digits (−10.80%), Led 24 (−9.16%) and Letter Recognition (−8.20%) datasets. But the bands distance attains an overall improvement in the UCI-Repository datasets (Tables 3 and 5), and attains the best results in 12 of the 19 UCI-Repository bases (Table 6). If there are lots of irrelevant features, the bands distance suffers a higher degradation than the correlated distances (see the results in the Led 7 and Led 24, and Waveform-21 and Waveform-40 datasets), and a very lower degradation than the distances that do not employ correlation. So, if the base case has irrelevant features, is particularly important to employ some previous method to eliminate them, and make subsequently the proper classification.

As regards the synthetic datasets, bands distance attains the best results (Table 5). The outstanding results attained in the Gauss, all the rings (with constant radius and area), and most of the sines datasets are due to the fact that is possible to approximate any figure in R^2 by means of straight lines, and for example the rings could be locally approximated with bands.

It was predictable worse results in the squares datasets, but the difference with the other distances is not too big, and the bands distance even attains the best results in two variants of the squares datasets.

[2] In spite of the fact that the results with the Cleveland dataset could seem abnormally lower, we must take into account that these results are obtained considering five classes, while experiments with this dataset have been concentrated on simply distinguishing presence (values 1–4) from absence (value 0) of heart disease.

Table 2
Pairwise comparison of statistically significant differences between 1-NN classifiers with different distances

	Cheb C	Cheb	Manh C	Manh	Eucl C	Eucl
Bands	36–24–8	29–36–3	38–23–7	32–35–1	33–31–4	28–36–4
Eucl	24–33–11	2–65–1	25–33–10	14–54–0	11–44–10	
Eucl C	22–37–9	9–42–17	26–36–6	18–37–13		
Manh	21–30–17	0–55–13	23–28–17			
Manh C	1–56–11	11–31–26				
Cheb	23–33–12					

Each cell contains, respectively, the number of statistically significant wins, ties and losses between the method in the row and the method in the column. Bands distance attains the best results.

Table 3
Pairwise comparison of statistically significant differences between 1-NN classifiers with different distances in the UCI-Repository datasets

	Cheb C	Cheb	Manh C	Manh	Eucl C	Eucl
Bands	6–11–2	8–9–2	9–9–1	10–8–1	6–10–3	7–10–2
Eucl	2–15–2	1–18–0	4–13–2	9–10–0	1–16–2	
Eucl C	3–14–2	2–16–1	5–14–0	9–10–0		
Manh	1–10–8	0–10–9	2–9–8			
Manh C	0–13–6	2–12–5				
Cheb	2–14–3					

Each cell contains, respectively, the number of statistically significant wins, ties and losses between the method in the row and the method in the column. Bands distance attains the best results.

Table 4
Pairwise comparison of statistically significant differences between 1-NN classifiers with different distances in synthetic datasets

	Cheb C	Cheb	Manh C	Manh	Eucl C	Eucl
Bands	30–13–6	21–27–1	29–14–6	22–27–0	27–21–1	21–26–2
Eucl	22–18–9	1–47–1	21–20–8	5–44–0	13–28–8	
Eucl C	19–23–7	7–26–16	21–22–6	9–27–13		
Manh	20–20–9	0–45–4	21–19–9			
Manh C	1–43–5	9–19–21				
Cheb	21–19–9					

Each cell contains the number of statistically significant wins, ties and losses between the method in the row and the method in the column. Bands distance attains the best results.

Table 5
Results of the classifiers with UCI-Repository, Synthetic and all the datasets

Distance	UCI (%)	Synt. (%)	All (%)
Euclidean	80.18	85.22	83.81
Euclidean with correlation	81.05	84.49	83.53
Manhattan	70.34	84.64	80.65
Manhattan with correlation	77.86	81.93	80.79
Chebychev	80.17	85.25	83.83
Chebychev with correlation	80.93	82.59	82.12
Bands	82.54	88.20	86.62

Bands distance attains very good results.

Table 6
Number of datasets where each distance is chosen by the heuristics (attains the best results) in the UCI-Repository, Synthetic and all the datasets

Distance	UCI	Synt.	All
Euclidean	4	6	10
Euclidean with correlation	4	0	4
Manhattan	0	1	1
Manhattan with correlation	1	6	7
Chebychev	3	9	12
Chebychev with correlation	3	6	9
Bands	12	27	39
Number of datasets	19	49	68

Bands distance attains the best results in more than half of the datasets.

Surprisingly, the bands distance does not obtain the best result in any variant of the bands datasets. Often the best results are obtained by distances that employ the correlation of the attributes with the class, and the worst results are obtained by distances without correlation. In spite of everything, the bands distance obtains results near to the best ones. To explain the results with the bands datasets we must take into account that in practice the bands usually are nearly parallel to X axis, but having a small variation because they are calculated based on a set of specific points in the vicinity. In addition to this fact, sometimes there are more close points of other classes that belong to neighbor bands. Especially we must take into account that correlated distances obtain better results because the correlation detects the irrelevant feature more accurately than the bands. So, an axis parallel band is the most favorable situation to correlated distances. If the bands were not axis parallel, the bands distance would continue identifying the direction of the bands and would obtain similar results, but correlated distances would suffer degradation until their results be similar to the non-correlated distances. In this situation the distance based on bands would attain the best results in all the variants.

4.2. Results with k-NN

Tables 8–10 show the pairwise comparison using the t-Student's test with the k-NN classifiers using the bands distance, and the other methods. These tables show that k-NN method with the bands distance exhibits a clearly superior behavior to A (k-NN), B (ε-ball) and D (ε-ball$^{k\text{-NN}}$), and is only frequently better than methods C (k-NN Heur), E (ε-ball Heur), F (ε-ball$^{k\text{-NN}}$ Heur) and F1 (ε-ball$^{1\text{-NN}}$ Heur).

The high number of datasets where the k-NN method with bands distance is significantly better or worse than C, E and F methods, suggests that datasets where is better to employ one or other kind of methods are different. So it would be possible to improve the behavior of a classifier performing a previous evaluation over the training set to determine the method that attains better results.

Table 11 shows the mean accuracy of all the classifiers broken down in UCI–Repository, Synthetic and "All datasets". k-NN method with bands distance attains the best results in all the categories. Table 12 shows the number of bases each classification method attains the best results with the UCI-Repository, Synthetic and "All datasets". k-NN method with bands distance attains the best results in nearly half of the case bases, both overall and in the UCI-Repository and synthetic sets. So, with no prior information about the dataset, it is advisable to employ the k-NN method with bands distance.

Table 13[2] shows the detailed results obtained in the UCI-Repository datasets by each classification method. We can underline that k-NN with bands distance attains higher results than the other methods in the case bases Tic–Tac–Toe (+14.51%) and Liver Disorder (+6.37%), and lower in Letter Recognition (−8.12%) and Glass (−6.07%). In the UCI-Repository datasets, k-NN with bands distance attains the best mean accuracy and the best results in nine of the 19 datasets (Tables 11 and 12).

The presence of irrelevant attributes affects in different degree the mean accuracy of the k-NN methods with bands distance, depending on the specific case base. So, for example, in the Led 7 and Led 24 datasets the presence of 17 irrelevant features clearly affect the results. But, in the Waveform-21 and Waveform-40 the 19 irrelevant features almost do not affect the results. To sum up we can indicate that if the case base has irrelevant features it is

Table 7
Results of the 1-NN method with the basics measures of distance and the bands distance with the UCI-Repository datasets

	Eucl (%)	Eucl C (%)	Manh (%)	Manh C (%)	Cheb (%)	Cheb C (%)	Bands (%)
IR	95.33	96.00	96.00	94.67	94.00	94.67	96.67
WI	94.94	96.07	94.94	92.70	95.51	96.63	98.31
PI	69.92	70.83	68.36	70.57	69.53	68.36	72.79+
GL	70.09	68.69	66.82	65.89	73.36	72.90	67.29
CL	52.48	54.13	50.83	55.12	52.48	56.44	57.43+
GD	96.70	96.40	64.60−	73.30−	96.30	95.10−	85.90−
SN	87.02	88.94	79.33−	84.62	86.54	87.50	88.46
LD	61.74	60.00	57.97	57.39	61.16	60.00	66.38
ZO	96.04	74.26−	91.09−	96.04	97.03	96.04	
TT	75.57	74.84	56.37−	76.51	75.57%	76.72	98.64+
L7	60.02	60.00	59.88−	60.00	60.02	60.00	59.92
L24	48.98	63.08+	10.20−	62.98+	48.98	63.08+	53.92+
W21	77.06	76.22	76.38	74.78−	76.62	75.86	85.16+
W40	73.24	77.82+	68.04−	75.70+	73.22	76.98+	83.52+
SO	100.00	97.87	53.19−	97.87	100.00	97.87	100.00
F1	73.55	73.64	73.26	73.73	73.64	73.55	74.77+
F2	95.50	95.40	95.40	95.40	95.50	95.31	95.97
F3	99.25	99.34	99.16	99.34	99.25	99.34	99.34
LR	96.02	93.65−	91.55−	77.72−	95.49−	90.32−	87.82−

"+"/"−" represents statistically significant improvement/degradation over the Euclidean distance according to a paired two-tailored t-test at a 95% confidence level.

advisable to employ a previous method to remove them, and prevent their negative effect over the performance of the classifier.

As regards the synthetic datasets, k-NN with the bands distance attains the best overall results (Tables 11 and 12) with outstanding results in the Gauss, all the rings (with constant radius and area), and most of the sines datasets. It was predictable worse results in all the squares datasets, but it obtains the best results in the uniform 2 × 2 squares.

Again, the bands distance does not obtain the best results in any variant of the bands datasets. Methods that employ the heuristics and weight the relevance of each attribute by its correlation with the class are in the first positions: C (k-NN Heur), E (ε-ball Heur) and F (ε-ball$^{k\text{-NN}}$ Heur). In spite of everything, k-NN with the bands distance attains to improve the results of the methods that do not employ correlation: a (k-NN), B (ε-ball) and D (ε-ball$^{k\text{-NN}}$). The reasons of these results with the bands datasets were previously discussed in Section 4.1. We must take into account that an axis parallel band is the most favorable situation to correlated distances.

Table 9
Pairwise comparison of statistically significant differences between classifiers in the UCI-Repository datasets

	F1	F	E	D	C	B	A
Bands	4–12–3	3–13–3	5–11–3	3–13–3	3–14–2	4–12–3	5–12–2
A	0–16–3	0–16–3	0–16–3	0–16–3	1–17–1	1–15–3	
B	0–19–0	0–19–0	0–19–0	0–18–1	3–15–1		
C	1–14–4	1–14–4	1–14–4	1–14–4			
D	0–19–0	0–19–0	0–19–0				
E	0–19–0	0–19–0					
F	0–19–0						

Each cell contains, respectively, the number of statistically significant wins, ties and losses between the method in the row and the method in the column. k-NN with the bands distance is on a statistical significant level with D, F and F1 methods.

5. Conclusions and future work

In this work we have proposed a new distance measure, bands distance, that could be employed in classification problems in conjunction with a classification method. We have made tests with 1-NN and k-NN methods with this new distance. We have studied its utility and have made an exhaustive comparison in 68 datasets employing six basic distances of the Geometry.

The new distance measure has been revealed very useful. When employing the 1-NN method, bands distance attains the highest mean accuracy and the best results in most of the datasets. When employing k-NN method, bands distance attains good results, although they are quite irregular. In some domains the improvement is quite important, even increasing the accuracy in Tic–Tac–Toe almost 15% with k-NN and 22% with 1-NN. In some UCI-Repository domains, she suffers a quite important degradation.

With synthetic datasets, the bands distance attains the best results in all the variants of the rings with constant area and radius, and tends to attain good results with sinus datasets. With squares and horizontal bands the results are average. It was predictable worse results in the squares datasets, but it is especially surprising the results with the horizontal bands. We have analyzed the causes of this behavior, where an axis parallel band is the most favorable situation to the correlated distances. If the dataset has irrelevant features, it seems particularly important to employ some method

Table 8
Pairwise comparison of statistically significant differences between classifiers

	F1	F	E	D	C	B	A
Bands	16–39–13	15–40–13	17–38–13	23–39–6	14–43–11	25–37–6	27–36–5
A	0–53–15	0–53–15	0–53–15	0–63–5	1–53–14	1–62–5	
B	2–55–11	2–55–11	2–55–11	0–67–1	6–48–14		
C	3–58–7	3–58–7	3–58–7	14–47–7			
D	2–55–11	2–55–11	2–55–11				
E	0–68–0	0–68–0					
F	0–68–0						

Each cell contains, respectively, the number of statistically significant wins, ties and losses between the method in the row and the method in the column. k-NN with the bands distance significantly improve the methods C, E, F, and F1 slightly, and very frequently the other methods.

Table 10
Pairwise comparison of statistically significant differences between classifiers in synthetic datasets

	F1	F	E	D	C	B	A
Bands	12–27–10	12–27–10	12–27–10	20–26–3	11–29–9	21–25–3	22–24–3
A	0–37–12	0–37–12	0–37–12	0–47–2	0–36–13	0–47–2	
B	2–36–11	2–36–11	2–36–11	0–49–0	3–33–13		
C	2–44–3	2–44–3	2–44–3	13–33–3			
D	2–36–11	2–36–11	2–36–11				
E	0–49–0	0–49–0					
F	0–49–0						

Each cell contains, respectively, the number of statistically significant wins, ties and losses between the method in the row and the method in the column. *k*-NN with the bands distance significantly improves C, E, F and F1 methods, and very frequently improve the others.

Table 11
Results of the classifiers with UCI–Repository, Synthetic and all the datasets

	Classifier	UCI (%)	Synt. (%)	All (%)
A	*k*-NN	85.03	85.89	85.65
B	ε-ball	84.95	86.06	85.75
C	*k*-NN Heur	85.12	88.14	87.30
D	ε-ball$^{k\text{-}NN}$	85.33	86.09	85.88
E	ε-ball Heur	85.14	88.11	87.29
F	ε-ball$^{k\text{-}NN}$ Heur	85.48	88.13	87.39
Bands	*k*-NN Bands distance	85.70	88.83	87.96

k-NN with the bands distance attains the best results in all the sections.

Table 12
Number of datasets where each classifier attains the best results with the UCI–Repository, Synthetic and all the datasets

Code	Classifier	UCI	Synt.	All
A	*k*-NN	4	4	8
B	ε–ball	8	9	17
C	*k*-NN Heur	6	12	18
D	ε–ball$^{k\text{-NN}}$	10	9	19
E	ε–ball Heur	7	11	18
F	ε–ball$^{k\text{-NN}}$ Heur	9	11	20
Bands	*k*-NN Bands distance	9	24	33
Number of datasets		19	49	68

Table 13
k-NN method results with the UCI-Repository datasets

	A (%)	B (%)	C (%)	D (%)	E (%)	F (%)	Bands
IR	96.00	96.00	96.00	96.00	96.00	96.00	96.00
WI	97.19	97.19	96.63	97.19	95.51	96.63	98.88
PI	75.39	73.31	76.30	73.31	75.52	75.52	76.30
GL	71.50	71.03	72.90	71.50	73.36	75.23	69.16
CL	57.10	58.42	58.09	58.42	57.43	57.43	58.09
GD	96.70	95.70	96.70	96.70	95.70	96.70	93.50
SN	87.02	87.02	88.94	87.98	90.87	90.38	87.50
LD	65.22	63.48	65.22	65.51	63.48	65.51	71.88+
ZO	96.04	96.04	97.03	97.03	96.04	97.03	96.04
TT	84.24	82.57−	78.18−	84.24	80.90	80.90	98.75+
L7	74.48	74.36	74.48	74.36	74.36	74.36	74.48
L24	72.34	73.80+	73.74+	73.80+	73.52+	73.52+	70.14−
W21	85.42	85.04	85.42	85.04	85.04	85.04	86.64+
W40	84.54	84.62	85.40	84.62	84.46	84.46	85.86+
SO	100.00	100.00	100.00	100.00	100.00	100.00	100.00
F1	80.68	82.93+	80.58	82.93+	82.93+	82.93+	81.05+
F2	96.34	96.62	96.25	96.62	96.62	96.62	96.44
F3	99.44	99.53	99.44	99.53	99.53	99.53	99.44
LR	96.02	96.42+	96.02	96.42+	96.42+	96.42+	88.30−

"+"/"−" represents statistically significant improvement/degradation over A (*k*-NN basic method) according to a paired two-tailored *t*-test at a 95% confidence level.

to previously eliminate them, and make subsequently the proper classification.

The value of the ε parameter that controls the locality of the method must be big enough to include the information of the surroundings of the point, but not too large, where it is affected by further points, outliers or points of other groups. We have employed the same ε value with all the datasets, but it is possible to employ different values in different points or areas of the space. This part is closely connected with *clustering*, and it is out of the scope of this work, so we consider it as a line of future work. Here it is possible to employ known *clustering* techniques to determine the size of the cloud where the point is placed, and choose an appropriate ε value.

It is remarkable that now we have much more knowledge than raw points, and we could try to extract information from the hyperplane that each point has learned. This knowledge could affect a subset of points or all the dataset points. When lots of points correspond to a pattern, this behavior could be interpreted in a global way, and, for example, we could propose the projection of these bands to a perpendicular hyperplane to reduce dimensionality. This feature selection could be understood as a coordinate transformation. It is also possible to identify situations where points locally "draw" a circumference or an arc, and then propose a conversion to the polar coordinates selecting as center the curvature center of the figure. In any case, the original problem is transformed into another one equivalent, but easier to solve and frequently easier to understand too.

It is interesting to employ the bands learned by the points to extract information about the case base. We could retain some points together with its bands, and employ this information in classification tasks. We could try different approaches to select the points that must be kept. For example, we could retain the most representative points, or some points of each cloud, or frontier points,. . .In this way, it is possible to reduce the amount of stored information to do classification tasks. A different approach consists in learning big bands, in such a way that it will retain information about every "important" band or cloud of points. We would learn higher level information, and forget specific points. A third approach could be a hierarchical classification: we could learn these metabands or "big bands" and retain information about specific points. When a new point must be classified and is far away from metabands, or some points "see" it at a similar distance, we can employ then specific points to classify it.

Acknowledgement

We acknowledge support for this work from the Dirección General de Investigación of Spain by the Project Grant TIN2006-03122.

Appendix A. adjust_multi$_H$ minimization using the Lagrangian-constrained minimization method

To use the bands distance in classification tasks, we suggested that you should learn the direction of the band of each known point that better fits its vicinity. In Section 2.2 we recommended that you should find the hyperplane H (or the equivalent vector $v = H^{\perp}$) that minimizes adjust_multi$_H$. We have defined the *multi-class adjustment* of a hyperplane $v = H^{\perp}$ as (Eq. 4):

$$\begin{aligned}\text{adjust\_multi}_H(p,P) = &\sum_{x_i\in E^{=}(p,\varepsilon)}\left(\sum_{j=1}^{n}(x_{ij}-p_j)v_j\right)^2 \mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\\ &+F\sum_{x_i\in E^{\neq}(p,\varepsilon)}\left(\left(\sum_{j=1}^{n}(x_{ij}-p_j)^2\right.\right.\\ &\left.\left.-\left(\sum_{j=1}^{n}(x_{ij}-p_j)v_j\right)^2\right)\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\right)\end{aligned}$$

where $F \in R_0^+$, n is the number of attributes, $E^{=}(p,\varepsilon)$ is the set of points of the same class that p, $E^{\neq}(p,\varepsilon)$ is the set of points of other classes, p_j is the jth attribute of p, x_{ij} is the jth attribute of x_i, and $v_j = \cos\alpha_j$ is the jth director cosine of H. $\sum_{x_i\in E^{=}(p,\varepsilon)}$ only collects information from points of the class of p, and $\sum_{x_i\in E^{\neq}(p,\varepsilon)}$ from other classes.

adjust_multi$_H$ minimization could be tackled as a constrained optimization problem using the Lagrange multipliers method to obtain a more straightforward solution. In this way we could turn the original problem into solving a system of n + 1 equations, easier to solve. In some datasets the number of attributes, and accordingly n, could be high. So, we have employed a numerical method of approximation to the solution in the experiments.

In general, given two functions $f(x_1,x_2,\ldots,x_n)$ and $g(x_1,x_2,\ldots,x_n)$ with first partial derivatives, if we want to find the points of the surface given by $g(x_1,x_2,\ldots,x_n)=0$, that minimizes or maximizes the value of the function $f(x_1,x_2,\ldots,x_n)$, we can use the Lagrangian-constrained minimization method. Function $g(x_1,x_2,\ldots,x_n)$ behaves as a restriction, and according to the Lagrange multipliers theorem, the point p, where the maximum or minimum is placed, simultaneously must satisfy the equations

$$\begin{aligned}&g(x_1,x_2,\ldots,x_n)=0\\ &\frac{\partial f(x_1,x_2,\ldots,x_n)}{\partial x_1}=\lambda\frac{\partial g(x_1,x_2,\ldots,x_n)}{\partial x_1}\\ &\frac{\partial f(x_1,x_2,\ldots,x_n)}{\partial x_2}=\lambda\frac{\partial g(x_1,x_2,\ldots,x_n)}{\partial x_2}\\ &\qquad\qquad\vdots\\ &\frac{\partial f(x_1,x_2,\ldots,x_n)}{\partial x_n}=\lambda\frac{\partial g(x_1,x_2,\ldots,x_n)}{\partial x_n}\end{aligned}$$

with some scalar λ. After solving this system of $n+1$ equations and $n+1$ variables ($x_1,x_2,\ldots,x_n$ and λ), the function f could be evaluated in each solution point to check if a maximum or minimum is reached.

We want to find the hyperplane H that minimizes adjust_multi$_H$, and we can face the problem finding the vector $v = H^{\perp}$ that minimizes adjust_multi$_H$. So we must minimize the function adjust_multi$_H$ with the constraint $\sum_{i=1}^{n} v_i^2 = 1$ because $v = H^{\perp}$ must be a unitary vector perpendicular to the hyperplane H, i.e., its components are the director cosines of H. We must find the values $v_1, v_2, \ldots, v_n$ and λ that solve the following set of equations:

$$\begin{aligned}&\sum_{i=1}^{n}v_i^2-1=0\\ &\frac{\partial\text{adjust\_multi}_H(v_1,v_2,\ldots,v_n)}{\partial v_1}=\lambda\frac{\partial\sum_{i=1}^{n}v_i^2-1}{\partial v_1}\\ &\frac{\partial\text{adjust\_multi}_H(v_1,v_2,\ldots,v_n)}{\partial v_2}=\lambda\frac{\partial\sum_{i=1}^{n}v_i^2-1}{\partial v_2}\\ &\qquad\qquad\vdots\\ &\frac{\partial\text{adjust\_multi}_H(v_1,v_2,\ldots,v_n)}{\partial v_n}=\lambda\frac{\partial\sum_{i=1}^{n}v_i^2-1}{\partial v_n}\end{aligned}$$

If we calculate partial derivatives with respect to v_k, we obtain

$$\frac{\partial\sum_{i=1}^{n}v_i^2-1}{\partial v_k}=\frac{\partial\sum_{i=1}^{n}v_i^2}{\partial v_k}=\sum\nolimits_{i=1}^{n}\frac{\partial v_i^2}{\partial v_k}=\frac{\partial v_k^2}{\partial v_k}=2v_k$$

and

$$\begin{aligned}&\frac{\partial\text{adjust\_multi}_H(v_1,v_2,\ldots,v_n)}{\partial v_k}\\ &=\frac{\partial\sum_{x_i\in E^{=}(p,\varepsilon)}\left(\sum_{j=1}^{n}(x_{ij}-p_j)v_j\right)^2\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}}{\partial v_k}\\ &\quad-F\frac{\partial\sum\limits_{x_i\in E^{\neq}(p,\varepsilon)}\left(\sum_{j=1}^{n}(x_{ij}-p_j)v_j\right)^2\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}}{\partial v_k}\\ &=\sum\nolimits_{x_i\in E^{=}(p,\varepsilon)}\frac{\partial\left(\sum_{j=1}^{n}(x_{ij}-p_j)v_j\right)^2}{\partial v_k}\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\\ &\quad-F\sum\nolimits_{x_i\in E^{\neq}(p,\varepsilon)}\frac{\partial\left(\sum_{j=1}^{n}(x_{ij}-p_j)v_j\right)^2}{\partial v_k}\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\\ &=\sum\nolimits_{x_i\in E^{=}(p,\varepsilon)}\left(2\left(\sum\nolimits_{j=1}^{n}(x_{ij}-p_j)v_j\right).\right.\\ &\qquad\left.\times\frac{\partial\left(\sum_{j=1}^{n}(x_{ij}-p_j)v_j\right)}{\partial v_k}\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\right)\\ &\quad-F\sum\nolimits_{x_i\in E^{\neq}(p,\varepsilon)}\left(2\left(\sum\nolimits_{j=1}^{n}(x_{ij}-p_j)v_j\right).\right.\\ &\qquad\left.\times\frac{\partial\left(\sum_{j=1}^{n}(x_{ij}-p_j)v_j\right)}{\partial v_k}\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\right)\\ &=2\sum\nolimits_{x_i\in E^{=}(p,\varepsilon)}\left(\sum_{j=1}^{n}(x_{ij}-p_j)v_j\right)(x_{ik}-p_k)v_k\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\\ &\quad-2F\sum\nolimits_{x_i\in E^{\neq}(p,\varepsilon)}\left(\sum_{j=1}^{n}(x_{ij}-p_j)v_j\right)(x_{ik}-p_k)v_k\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\\ &=2\sum_{j=1}^{n}\left(\sum_{x_i\in E^{=}(p,\varepsilon)}(x_{ij}-p_j)(x_{ik}-p_k)\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\right.\\ &\qquad\left.-F\sum_{x_i\in E^{\neq}(p,\varepsilon)}(x_{ij}-p_j)(x_{ik}-p_k)\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\right)v_j\end{aligned}$$

We can rewrite the previous expression as:

$$\begin{aligned}&\frac{\partial\text{adjust\_multi}_H(v_1,v_2,\ldots,v_n)}{\partial v_k}=\\ &2\sum\nolimits_{j=1}^{n}\left(\sum\nolimits_{x_i\in P}F_{\text{class}}(x_i)(x_{ij}-p_j)(x_{ik}-p_k)\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\right)v_j\end{aligned}$$

where now we do not distinguish points of the class of p from points of other classes, P is the set of points, and the function $F_{\text{class}}(x_i)$ is defined as:

$$F_{\text{class}}(x_i)=\begin{cases}1 & \text{if class}(x_i)=\text{class}(p)\\ -F & \text{otherwise}\end{cases}$$

So, replacing the value of the partial derivatives in the previous system of equations and simplifying we obtain:

$$\begin{aligned}&\sum_{i=1}^{n}v_i^2-1=0\\ &\sum_{j=1}^{n}\left(\sum_{x_i\in P}F_{\text{class}}(x_i)(x_{ij}-p_j)(x_{i1}-p_1)\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\right)v_j=\lambda v_1\end{aligned}$$

$$\sum_{j=1}^{n}\left(\sum_{x_i\in P}F_{\text{class}}(x_i)\big(x_{ij}-p_j\big)(x_{i2}-p_2)\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\right)v_j=\lambda v_2$$
$$\vdots$$
$$\sum_{j=1}^{n}\left(\sum_{x_i\in P}F_{\text{class}}(x_i)\big(x_{ij}-p_j\big)(x_{ik}-p_k)\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\right)v_j=\lambda v_k$$
$$\vdots$$
$$\sum_{j=1}^{n}\left(\sum_{x_i\in P}F_{\text{class}}(x_i)\big(x_{ij}-p_j\big)(x_{in}-p_n)\mathrm{e}^{-\frac{4}{\varepsilon^2}d(x_i,p)}\right)v_j=\lambda v_n$$

So, the original minimization problem of the function adjust_multi$_H$ in n-dimensional space has been converted into solving a system of $n+1$ equations and $n+1$ variables.

References

[1] D.W. Aha, Tolerating noisy irrelevant and novel attributes in instance-based learning algorithms, International Journal of Man–Machine Studies 36 (1992) 267–287.

[2] D.W. Aha, D. Kibler, M.K. Albert, Instance-based learning algorithms, Machine Learning 6 (1991) 37–66.

[3] C.L. Blake, C.J. Merz, UCI repository of machine learning databases, 1998. Available from: <http://www.ics.uci.edu/~mlearn/MLRepository.html>.

[4] S. Cost, S. Salzberg, A weighted nearest algorithm for learning with symbolic features, Machine Learning 10 (1993) 57–78.

[5] T.M. Cover, P.E. Hart, Nearest neighbor pattern classification, Institute of Electrical and Electronics Engineers Transactions on Information Theory 13 (1967) 21–27.

[6] B.V. Dasarathy, Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques, IEEE Computer Society Press, Los Alamitos, CA, 1991.

[7] P. Domingos, Context-sensitive feature selection for lazy learners, Artificial Intelligence Review 11 (1997) 227–253 (Special Issue on "Lazy Learning").

[8] E. Fix, J.L. Hodges, Jr. Discriminatory Analysis, Nonparametric Discrimination, Consistency Properties. Technical report, Randolph Field, TX: United States Air Force, School of Aviation Medicine, 1951, Technical Report 4.

[9] M. Laguía, J.L. Castro. Similarity relations based on distances as fuzzy concepts. In: Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT–2001), September 2001.

[10] M. Laguía, J.L. Castro. A comparison between k-NN and heuristic learning. In: Eleventh World Congress of International Fuzzy Systems Association (IFSA 2005), July 2005.

[11] M. Laguía, J.L. Castro, Algorithms for classification based on k-NN, Mathware & Soft Computing 14 (2007) 5–22.

[12] E. Plaza, R. López, E. Armengol. On the importance of similitude: an entropy-based assessment. In: Third European Workshop on Case-Based Reasoning (EWCBR–96). Springer-Verlag, Berlin, 1996.

[13] F. Ricci, P. Avesani. Learning a local similarity metric for case-based reasoning. In First International Conference on Case-Based Reasoning (ICCBR–95), Sesimbra, Portugal. Springer-Verlag, Berlin, 1995, pp. 301–312.

[14] F. Ricci, P. Avesani, Data compression and local metrics for nearest neighbor classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (4) (1999) 380–384.

[15] M.M. Ritcher, Classification and learning of similarity measures, in: 16. Jahrestagung der Gesellschaft für Klassifikation (GFKL–92), Springer-Verlag, Berlin, 1992.

[16] M.M. Ritcher, On the notion of similarity in case-based reasoning, in: G. delViertl (Ed.), Mathematical and Statistical Methods in Artificial Intelligence, Springer-Verlag, Berlin, 1995, pp. 171–184.

[17] S. Salzberg, A nearest hyperrectangle learning method, Machine Learning 6 (1991) 251–276.

[18] C. Stanfill, D. Waltz, Towards memory-based reasoning, Communications of the ACM 29 (1986) 1213–1228.

[19] S.M. Weiss, C.A. Kulikowski, Computer Systems that Learn, Morgan Kaufmann, Los Altos, CA, 1991.

[20] D. Wettschereck, A Study of Distance-Based Machine Learning Algorithms, Ph.D. Thesis, Oregon State University, 1994.

[21] D. Wettschereck, T.G. Dietterich, An experimental comparison of the nearest-neighbor and nearest-hyperrectangle algorithms, Machine Learning 19 (1995) 5–28.

[22] D.R. Wilson, T.R. Martinez, Improved heterogeneous distance functions, Journal of Artificial Intelligence Reseach (JAIR) 6 (1997) 1–34.