Available online at www.sciencedirect.com
ScienceDirect
Pattern Recognition 41 (2008) 1961 – 1974
PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/pr

Maximizing the area under the ROC curve by pairwise feature combination

C. Marrocco[a,*], R.P.W. Duin[b], F. Tortorella[a]

[a] *Dipartimento di Automazione, Elettromagnetismo, Ingegneria dell'Informazione e Matematica Industriale, Università degli Studi di Cassino, Via G. di Biasio 43, 03043 Cassino (FR), Italy*

[b] *Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands*

Received 1 June 2007; received in revised form 21 September 2007; accepted 17 November 2007

Abstract

The majority of the available classification systems focus on the minimization of the classification error rate. This is not always a suitable metric specially when dealing with two-class problems with skewed classes and cost distributions. In this case, an effective criterion to measure the quality of a decision rule is the area under the Receiver Operating Characteristic curve (AUC) that is also useful to measure the ranking quality of a classifier as required in many real applications. In this paper we propose a nonparametric linear classifier based on the maximization of AUC. The approach lies on the analysis of the Wilcoxon–Mann–Whitney statistic of each single feature and on an iterative pairwise coupling of the features for the optimization of the ranking of the combined feature. By the pairwise feature evaluation the proposed procedure is essentially different from other classifiers using AUC as a criterion. Experiments performed on synthetic and real data sets and comparisons with previous approaches confirm the effectiveness of the proposed method.

Keywords: Two-class problems; ROC curve; Ranking; AUC

1. Introduction

In many complex classification problems highly discriminant classifiers are needed especially when dealing with dichotomous decisions that required to choose between two possible alternative classes. Applications such as automated cancer diagnosis, currency verification or fraud detection fall in this category. Since in such cases a classification error could frequently have serious consequences, the employed classifiers should ensure a high reliability to avoid erroneous decisions. Anyway, the mainstream classifiers are usually designed to minimize the classification error and this has been shown not to be a reliable metric when the two-class problems have skewed class or cost distributions.

Under these conditions it is better to refer to the *receiver operating characteristic* (*ROC*) curve. ROC curves were originally introduced in signal detection theory and the main reason they became an useful performance evaluation tool is the fact that the produced curve is independent of class distribution and underlying misclassification costs [1]. A ROC curve is the plot of *true positive rate* (*TPR*) versus *false positive rate* (*FPR*) by varying the threshold, which is simply a score produced by a decision function, usually the probability of membership to a class or a distance to a decision surface. Moreover, as well described in many papers [1–4] the geometrical properties of the ROC curve can be profitably used to optimize the performance of a dichotomizer with reference to various metrics and classification requirements.

However, if we are faced with different classifiers, it is often preferable for their comparison to employ a single scalar value such as the *area under the ROC curve* (*AUC*) [5,6]. Larger AUC values indicate on average better classifier performance, even though it is possible that a classifier with high AUC can be outperformed by a lower AUC classifier at some region of the ROC space. Recently, some studies on the relationships between AUC and accuracy have been performed; in Refs. [7,8] it has been established that AUC is a statistically consistent and more discriminating value than accuracy (i.e. of error rate). Moreover, AUC is also a suitable measure to evaluate the classifier's ability to rank instances in two-class classification problems. In particular, it is the probability that a randomly

* Corresponding author. Tel.: +39 0776 2993986; fax: +39 0776 2993987.
E-mail addresses: c.marrocco@unicas.it (C. Marrocco), r.p.w.duin@tudelft.nl (R.P.W. Duin), tortorella@unicas.it (F. Tortorella).

0031-3203/$30.00
doi:10.1016/j.patcog.2007.11.017

chosen positive example has a higher decision function value than a random negative example.

Ranking is a popular topic in the machine learning field and on these bases several learning algorithms have been proposed in the recent literature. In Ref. [9] an algorithm to combine rankings based on the boosting approach has been introduced and in Ref. [10] a variation to this algorithm is proposed. The maximization of AUC has been applied also in decision trees [11] and logistic regression [12]. Recently, rank optimizing support vector machines (SVM) have came into focus. In Ref. [13] rank optimizing kernels have been investigated while Ref. [14] introduces a similar kernel formulation which led to a better ranking performance compared to the previous work. A linear programming approach similar to l_1-norm SVM has been developed in Ref. [15] while in Ref. [16] a similar linear weighting of features has been successfully applied to the detection of the interstitial lung disease. Also combinations of rules have been analyzed to directly maximize AUC of the combiner and a method to evaluate the weight of the linear combination of two classifier has been proposed in Ref. [17].

In this paper we propose a nonparametric linear classifier able to maximize AUC. The procedure is based on an iterative pairwise coupling of the features suitable to create a ranker in feature space (i.e. a classifier after an opportune thresholding). The coupling is performed finding weights of the linear combination of two features for the maximization of AUC evaluated through the Wilcoxon–Mann–Whitney (WMW) statistic. Experiments performed on synthetic and real data sets and a comparison with previous approaches have confirmed the effectiveness of the proposed method.

The rest of the paper is organized as follows: in the next section a short description of the ROC curve and of AUC measure is presented for a linear discriminant function. Section 3 describes the proposed approach for two features while in Section 4 this method is extended to N features with a greedy approach. Then, experiments performed are reported and discussed in Section 5 while in the last section some conclusions and possible future developments are proposed.

2. Linear discriminant functions and the ROC curve

In two-class classification problems, the goal is to build a classifier f that assigns a sample $\mathbf{x}$ (say Q-dimensional) represented in an instance space X to one of two mutually exclusive classes that can be generally called *Positive* (P) and *Negative* (N) class. Let us consider, without loss of generality, that a dichotomizer f provides a real output $f(\mathbf{x})$ for each sample it represents a confidence degree that the sample belongs to one of the two classes, e.g. P. A way to assess the quality of such rule is to evaluate the performance obtained on each class varying a suitable threshold.

Let us focus on linear classifiers. The linear discriminant function can be written as

$$f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0 = \sum_{i=1}^{Q} w_i x_i + w_0,$$

where $\mathbf{w}$ is the weight vector and w_0 the threshold weight. In this way the sample $\mathbf{x}$ is assigned to the class P if $f(\mathbf{x}) > 0$ and to the class N if $f(\mathbf{x}) < 0$, i.e. $\mathbf{x}$ is assigned to P if $\mathbf{w}^{\mathrm{T}}\mathbf{x}$ exceeds the threshold $-w_0$ and to N otherwise. The equation $f(\mathbf{x}) = 0$ defines the decision boundary ϕ that separates the two decisions regions. In our case the decision boundary is a hyperplane. For a given threshold $w_0 = -t$ it is possible to define a *TPR* and a *FPR* as

$$TPR(t) = \int_t^{+\infty} f_p(x)\,\mathrm{d}x,$$

$$FPR(t) = \int_t^{+\infty} f_n(x)\,\mathrm{d}x,$$

where $f_p(x)$ and $f_n(x)$ are the density functions of the classifier output for the positive and negative class, respectively. Taking into account samples with score less than the threshold it is possible to define a *true negative rate* (*TNR*) and a *false negative rate* (*FNR*) as

$$TNR(t) = \int_{-\infty}^{t} f_n(x)\,\mathrm{d}x = 1 - TPR, \tag{1a}$$

$$FNR(t) = \int_{-\infty}^{t} f_p(x)\,\mathrm{d}x = 1 - FPR. \tag{1b}$$

Eq. (1) demonstrates that the pair ($FPR(t)$, $TPR(t)$) is sufficient to completely characterize the performance of a decision rule for a given threshold. Most importantly such indices are independent of the a priori probability of the classes because they are separately evaluated on the different classes.

The ROC curve details the *TPR* versus the *FPR* over the range of all the possible threshold values, thus providing a description of the performance of the classifier at different operating points even when the prior distributions of the classes or the cost distributions are not known [1,2]. The shape of the optimal ROC curve depends on how the classes are distributed: qualitatively, the closer the curve to the upper left corner, the more discriminable the two classes.

To demonstrate the relation between the discriminant linear function and the ROC curve let us focus on a two-dimensional problem. In this case the decision function simplifies to:

$$f(\mathbf{x}) = w_1 x_1 + w_2 x_2 + w_0 \tag{2}$$

and the sample $\mathbf{x}$ will be assigned to P if $w_1x_1 + w_2x_2 + w_0 > 0$ and to N otherwise. In this case the hyperplane collapses in a straight line with slope equal to $-w_1/w_2$. When we fix the two weights w_1 and w_2 we are defining the slope of the decision boundary while a change of the value $-w_0$ corresponds to a translation of the decision boundary in feature space. Once a particular value for the slope has been chosen, varying the value w_0 produces a family of lines (decision boundaries) with the same slope. Each of them defines a particular classifier which produces a certain pair (*TPR*, *FPR*) corresponding to a particular point on the ROC curve. When the value of w_0 is varied (and thus the decision boundary is translated) the whole ROC curve is drawn. In summary, each value for the slope produces a particular ROC curve where each point is associated

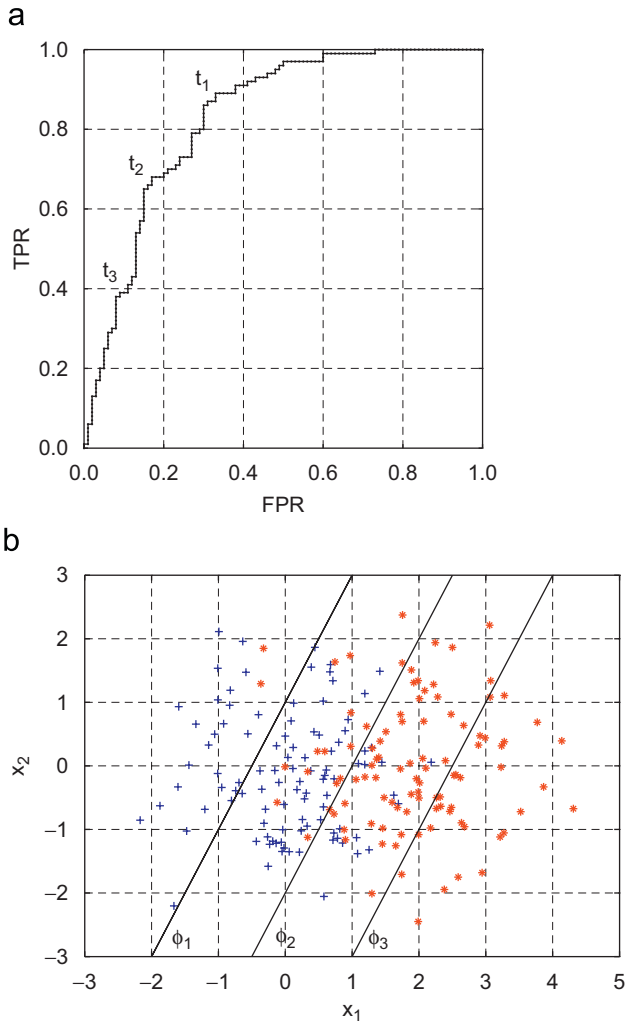

Fig. 1. Example of a two-dimensional problem with two overlapping classes. Fig. (a) shows the ROC curve for a linear classifier with three different operating points corresponding to the three straight lines shown in fig. (b).

to one of the parallel lines with that slope in feature space. Fig. 1 shows an example of a two-dimensional problem with two overlapping classes. Fig. 1(a) shows the ROC curve for a linear classifier with three different operating points corresponding to three different straight lines in the feature space represented in Fig. 1(b). Each point (*TPR*, *FPR*) on the curve corresponds to a particular (operating point for the) classifier, i.e. to a particular value for the threshold $-w_0$ and, consequently, to one of the parallel lines defined by the chosen slope.

Although the ROC curve provides a comprehensive evaluation of the performance of the decision rule, a scalar measure, the AUC, can be used to give an estimate of the quality of the dichotomizer (AUC=0.5 for a nondiscriminating rule, AUC=1 for a perfectly discriminating rule). AUC has been recently proposed in many papers as an alternative measure to the accuracy (i.e. to the classification error) due to its independence of the decision threshold and the prior class and cost distributions [5,8]. Moreover AUC has an important statistical property, i.e. it is an estimator of the probability of correct pairwise ranking [18,6] that is the probability that a sample belonging to the positive class P has a confidence degree higher than a sample belonging to the class N. Such probability can also be estimated by means of the WMW statistic [19]; let us consider l_N samples belonging to the class N and l_P belonging to the class P, and say $f(\mathbf{p_i})$ and $f(\mathbf{n_j})$ the output of a classifier on the ith positive sample $\mathbf{p_i}$ and on the jth negative sample $\mathbf{n_j}$, we have

$$R = \frac{\sum_{i=1}^{l_P}\sum_{j=1}^{l_N} I(f(\mathbf{p_i}), f(\mathbf{n_j}))}{l_P l_N}, \tag{3}$$

where $I(a, b)$ is an indicator function defined as

$$I(a,b) = \begin{cases} 1 & \text{if } a > b, \\ 0.5 & \text{if } a = b, \\ 0 & \text{if } a < b. \end{cases}$$

In this way, it is possible to evaluate AUC of f directly through Eq. (3) without explicitly plotting the ROC curve and estimating the area with a numerical integration. Several papers try to maximize AUC suggesting an approximation approach to the WMW statistic. For example in Refs. [20,12] a continuous function is used so that it is possible to use the gradient methods to solve the optimization problem.

Hence AUC also represents a measure of the quality of the ranking. It is worth noting that when we evaluate the ranking we do not need to evaluate a threshold. In fact, to have $f(\mathbf{p_i}) > f(\mathbf{n_j})$ with $\mathbf{p_i} = (p_i^1\ p_i^2)$ and $\mathbf{n_j} = (n_j^1\ n_j^2)$, it is sufficient that $w_1 p_i^1 + w_2 p_i^2 > w_1 n_j^1 + w_2 n_j^2$. As a result, the WMW statistic (and thus AUC) is independent of the value of the threshold.

Let us now consider the decision function in Eq. (2) and the decision boundary ϕ. Given two points in feature space $\mathbf{p} = (p_1, p_2)$ and $\mathbf{n} = (n_1, n_2)$ drawn, respectively, from class P and N their signed Euclidean distance from the decision boundary are

$$d(\mathbf{p}, \phi) = \frac{w_1 p_1 + w_2 p_2 + w_0}{\sqrt{w_1^2 + w_2^2}},$$
$$d(\mathbf{n}, \phi) = \frac{w_1 n_1 + w_2 n_2 + w_0}{\sqrt{w_1^2 + w_2^2}}.$$

A correct ranking for the pair $\mathbf{p}$ and $\mathbf{n}$ thus means that the signed distance of the positive point is higher than the signed distance of the negative point; in other words, the positive point follows the negative point on the line orthogonal to the decision boundary. As a consequence, the classification cannot be wrong for both samples: in the worst case, if the threshold is not adequately chosen, both points lie on the same side of the decision boundary. However, a suitable shifting of the decision boundary allows the two points to be correctly classified. Hence, if $w_1 p_1 + w_2 p_2 > w_1 n_1 + w_2 n_2$, we can choose a threshold w_0^* such as

$$-(w_1 p_1 + w_2 p_2) \leqslant w_0^* \leqslant -(w_1 n_1 + w_2 n_2)$$
$$\Downarrow$$
$$w_1 p_1 + w_2 p_2 + w_0^* \geqslant 0 \quad \text{and} \quad w_1 n_1 + w_2 n_2 + w_0^* \leqslant 0.$$

On the other hand, if the pairs $\mathbf{p}$ and $\mathbf{n}$ are not correctly ranked, there is no value for the threshold w_0 which can correctly classify both of the points. In summary, the slope maximizing AUC

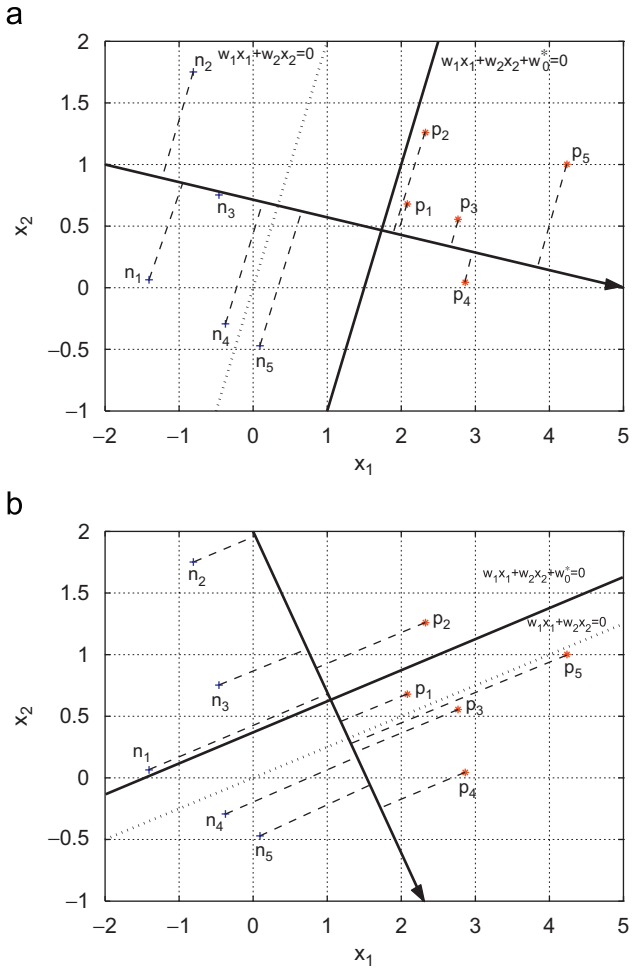

Fig. 2. Two-dimensional problem with five positive samples (asterisks) and five negative samples (plus signs). The decision boundary (dotted line) leading to a correct ranking, i.e. maximum AUC, is shown in (a) while in (b) an imperfect ranking that means that no suitable threshold can be chosen to linearly separate the two classes is reported.

is the slope for which there is the maximum number of pairs correctly ranked, i.e. of the pairs that could be correctly classified with a suitable choice of the threshold.

A two-dimensional example is shown in Fig. 2. Five samples for the positive class and five for the negative class, i.e. twenty-five possible pairs, are plotted. In Fig. 2(a) the decision boundary ϕ is shown with a slope that maximizes the AUC. As an evidence, we consider the perpendicular to ϕ and project all the samples on that line; fixing a way to move along the line, it is possible to obtain a correct ranking among all the possible pairs (AUC = 1). If we choose a threshold on this line it is possible to build a linear classifier that is able to assign all the samples to the correct class. In Fig. 2(b) we have a similar situation but in this case the slope of the decision boundary does not lead to a perfect ranking (AUC = $\frac{22}{25}$) and this reflects in errors in the class assignment.

3. AUC maximization in two-dimensional feature space

In the previous section we discussed the suitability of AUC for a linear classifier. Let us now focus on an approach to combine features in a linear way, so as to maximize AUC of the resulting linear classifier. To this aim, we first consider how to find a suitable slope for the linear combination of two features that maximizes the WMW statistic. Let X be the set of samples as defined before and let us consider two generic features x_h and x_k. Let us consider the values of the hth and kth features on the ith positive sample p_i and the jth negative sample n_j: p_i^h n_j^h p_i^k n_j^k and the relative ranking measure for the two features measured by

$$R_h = \frac{\sum_{i=1}^{l_P}\sum_{j=1}^{l_N} I(p_i^h, n_j^h)}{l_P l_N},$$
$$R_k = \frac{\sum_{i=1}^{l_P}\sum_{j=1}^{l_N} I(p_i^k, n_j^k)}{l_P l_N}.$$

Since we want to maximize AUC we are independent of the threshold. Hence, let us consider a linear combination of the two features

$$x_{lc} = \alpha x_h + (1-\alpha)x_k,$$

where $\alpha/(1-\alpha)$ is the relative weight of the features x_k with respect to x_h. The value of x_{lc} on the positive and negative sample will be:

$$p_i^{lc} = \alpha p_i^h + (1-\alpha)p_i^k,$$
$$n_j^{lc} = \alpha n_j^h + (1-\alpha)n_j^k.$$

According to the WMW statistic the quality of the ranking of x_{lc} can be measured by

$$R_{lc} = \frac{\sum_{i=1}^{l_P}\sum_{j=1}^{l_N} I(p_i^{lc}, n_j^{lc})}{l_P l_N} \tag{4}$$

and depends on the value of the weight α. We want to maximize AUC relative to the pair of features; to do that we have to analyze the term $I(p_i^{lc}, n_j^{lc})$ that depends on the value of $I(p_i^h, n_j^h)$ and $I(p_i^k, n_j^k)$. We can distinguish three different important cases[1]:

- $I(p_i^h, n_j^h)=1$ and $I(p_i^k, n_j^k)=1$ means that according to both features the two samples are correctly ranked, i.e. $p_i^h > n_j^h$ and $p_i^k > n_j^k$ and therefore $p_i^{lc} > n_j^{lc} \Rightarrow I(p_i^{lc}, n_j^{lc}) = 1$ whichever is the value of α.
- $I(p_i^h, n_j^h) = 0$ and $I(p_i^k, n_j^k) = 0$ means that neither feature correctly ranks the two samples i.e. $p_i^h < n_j^h$ and $p_i^k < n_j^k$ and therefore $p_i^{lc} < n_j^{lc} \Rightarrow I(p_i^{lc}, n_j^{lc}) = 0$ independently of α.
- $I(p_i^h, n_j^h) = 1$ and $I(p_i^k, n_j^k) = 0$ or $I(p_i^h, n_j^h) = 0$ and $I(p_i^k, n_j^k)=1$ means that only one feature correctly ranks the two samples and the value of $I(p_i^{lc}, n_j^{lc})$ is dependent on α.

[1] It is worth noting that a tie is only possible for discrete features and therefore we assume $I(a, b)=0$ when $a=b$ as well. This implies a negligible underestimate of the ranking, but it sensibly simplifies the following analysis.

According to these cases we can subdivide the set of samples in four different subsets defined as

$$X_{mn} = \{(i, j) | I(p_i^h, n_j^h) = m \text{ and } I(p_i^k, n_j^k) = n\}.$$

As a consequence the expression of the ranking of the combined features will be

$$R_{lc} = \frac{1}{l_P l_N}\left[\sum_{(i,j)\in X_{00}} I(p_i^{lc}, n_j^{lc}) + \sum_{(i,j)\in X_{11}} I(p_i^{lc}, n_j^{lc}) + \sum_{(i,j)\in X_{10}\cup X_{01}} I(p_i^{lc}, n_j^{lc})\right],$$

where $\sum_{(i,j)\in X_{00}} I(p_i^{lc}, n_j^{lc})$ is equal to zero and $\sum_{(i,j)\in X_{11}} I(p_i^{lc}, n_j^{lc})$ is the cardinality of the set X_{11}. Moreover, if we define as $v(\alpha)$ the quantity $\sum_{(i,j)\in X_{10}\cup X_{01}} I(p_i^{lc}, n_j^{lc})$ that is dependent on the weight, we have

$$R_{lc} = \frac{1}{l_P l_N}[\text{card}(X_{11}) + v(\alpha)].$$

So we have to focus on the pairs on which the features are differently ranked, i.e. on the sets X_{10} and X_{01}. In order to find the value of α that maximizes the ranking we have to study the term $\sum_{(i,j)\in X_{10}\cup X_{01}} I(\xi_i, \eta_j)$ looking at the weight for which $I(\xi_i, \eta_j) = 1 \Rightarrow \xi_i > \eta_j$, i.e.:

$$\begin{aligned} &\alpha p_i^h + (1-\alpha) p_i^k > \alpha n_j^h + (1-\alpha) n_j^k \\ &\Downarrow \\ &\alpha \Delta_{ij}^h + (1-\alpha)\Delta_{ij}^k > 0, \end{aligned} \tag{5}$$

where $\Delta_{ij}^h = p_i^h - n_j^h$ and $\Delta_{ij}^k = p_i^k - n_j^k$. From Eq. (5) we can obtain two different constraints on α for the two sets we are considering; we have

$$\alpha < \frac{\Delta_{ij}^k}{\Delta_{ij}^k - \Delta_{ij}^h} \quad \text{if } (i, j) \in X_{10}, \tag{6a}$$

$$\alpha > \frac{\Delta_{ij}^k}{\Delta_{ij}^k - \Delta_{ij}^h} \quad \text{if } (i, j) \in X_{01}. \tag{6b}$$

If Eq. (6) is verified for each pair $(i, j) \in X_{10} \cup X_{01}$, it is possible to obtain the maximum value for the function $v(\alpha)$, i.e. the cardinality of $X_{10} \cup X_{01}$. In this case, we can find an optimum α:

$$\max_{(i,j)\in X_{01}} \frac{\Delta_{ij}^k}{\Delta_{ij}^k - \Delta_{ij}^h} < \alpha_{opt} < \min_{(i,j)\in X_{10}} \frac{\Delta_{ij}^k}{\Delta_{ij}^k - \Delta_{ij}^h}. \tag{7}$$

This condition is verified only if the two sets are completely disjoint, i.e. if the two features are highly complementary in the ranking evaluation. So when the two sets are not separated we have to evaluate the weight α using the cumulative distributions

$$F_{10}(\alpha) = \text{card}\left((i, j) \in X_{10} \left| \frac{\Delta_{ij}^k}{\Delta_{ij}^k - \Delta_{ij}^h} > \alpha \right.\right),$$

$$F_{01}(\alpha) = \text{card}\left((i, j) \in X_{01} \left| \frac{\Delta_{ij}^k}{\Delta_{ij}^k - \Delta_{ij}^h} < \alpha \right.\right).$$

Hence, the function that has to be maximized is

$$v(\alpha) = F_{10}(\alpha) + F_{01}(\alpha)$$

and the optimal value of α can be found by means of a sequential search.

An example of the real distributions of the ratios $\Delta_{ij}^k/(\Delta_{ij}^k - \Delta_{ij}^h)$ in the two sets X_{10} and X_{01} is shown in Fig. 3 while the function $v(\alpha)$ obtained by these two distributions is plotted in Fig. 4.

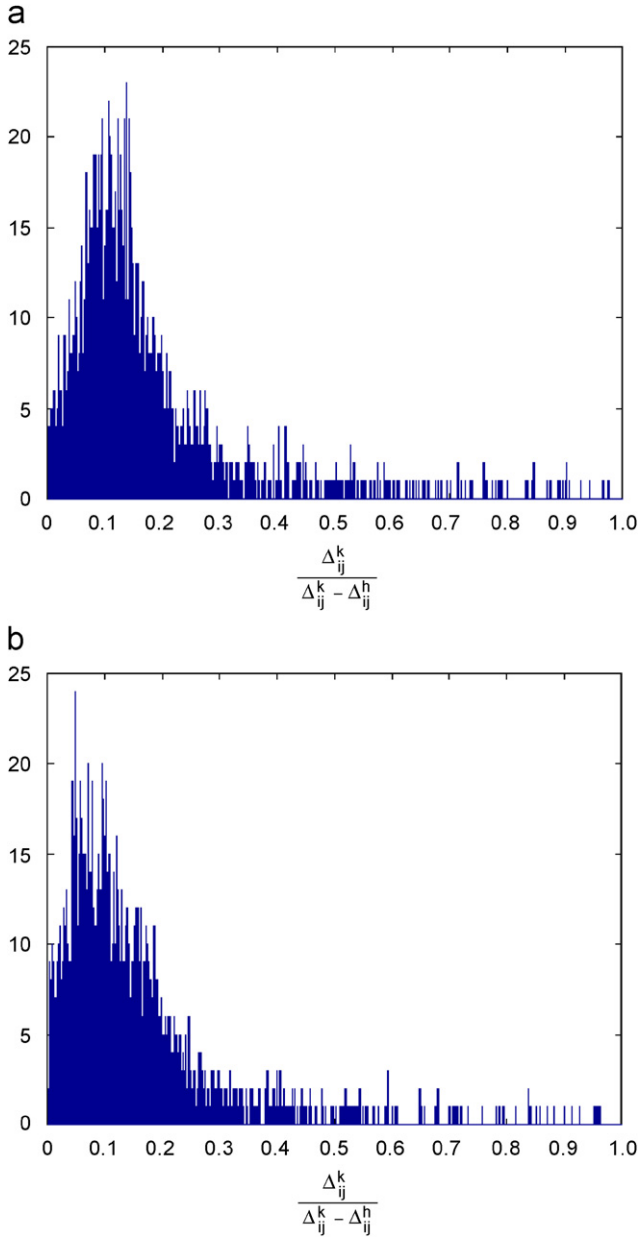

Fig. 3. Histograms of the occurrences of the ratio $\Delta_{ij}^k/(\Delta_{ij}^k - \Delta_{ij}^h)$ evaluated on the sets X_{10} (a) and X_{01} (b).

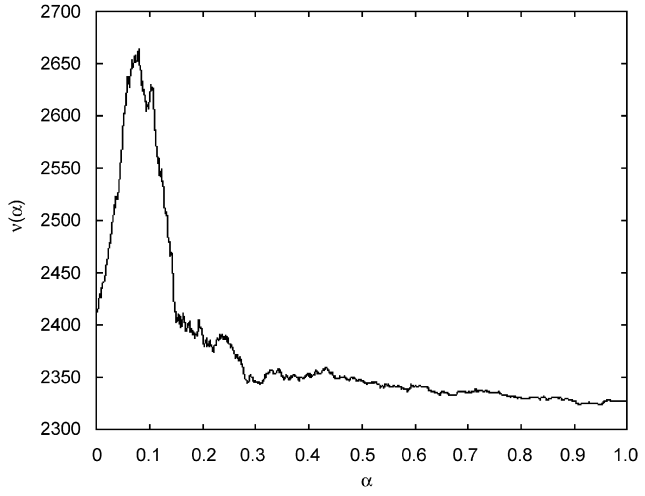

Fig. 4. The trend of the function $v(\alpha)$ obtained by the two distributions shown in Fig. 3.

4. AUC maximization in multidimensional feature space

The next step of our method consists in extending the procedure described in the previous section to a higher number of features. To this aim, let us consider Q features $x_1 \dots x_Q$ and their linear combination:

$$x_{lc} = \alpha_1 x_1 + \cdots + \alpha_Q x_Q = \sum_{i=1}^{Q} \alpha_i x_i = \boldsymbol{\alpha}^{\mathrm{T}} \mathbf{x}.$$

The goal is to find the weight vector

$$\boldsymbol{\alpha}_{opt} = (\alpha_1 \dots \alpha_Q),$$

maximizing the WMW statistic associated with the ranker described by x_{lc}. However, it is not possible to extend our approach to this case since the direct optimization of the function in Eq. (4) is intractable due to the computational problem connected to the evaluation of Eq. (5).

Therefore, a suboptimal algorithm that approximates the solution using a greedy approach has been adopted. Greedy methods build solutions piece by piece. Each step increases the size of the partial solution and is based on local optimization: the selected choice is the one that produces the largest immediate gain, i.e. the best ranking, maintaining the feasibility of the problem. In our case, instead of finding a weight vector in one step we iteratively find the optimal weight of the linear combination of two features (as described in the previous section) so as to evaluate all the combination weights in at most $Q-1$ steps.

In this context an important role is played by the order of combination, i.e. which pair of features should be combined at first, to avoid considering every possible combination. From Eq. (7) we know that the more separated the two distributions are relative to the sets X_{10} and X_{01} the greater is the improvement to the ranking of the combined features. Therefore, it is possible to combine the features choosing the pair that exhibits the maximum diversity in the ranking, i.e. the minimum rank correlation coefficient between features [21]. To this aim, we choose Spearman's rank correlation coefficient, a nonparametric measure of correlation that assesses how well an arbitrary monotonic function could describe the relationship between two features without making any assumptions about the frequency distribution [22]. It is defined as

$$\rho_{hk} = 1 - 6\frac{\sum_{i=1}^{L}(r_i^h - r_i^k)^2}{L(L^2-1)},$$

where r_i^h and r_i^k are the rankings on the two considered features h and k and $L = l_P + l_N$.

Algorithm 1 (The Maximum AUC Linear Classifier (MALC)).

Input: A L-dimensional matrix representing the training set with Q features $\mathbf{x}_1, \dots, \mathbf{x}_Q$; $l_P > 0$ and $l_N > 0$, the number of positive and negative samples with $L = l_P + l_N$.
Output: $\boldsymbol{\alpha}$, the weight vector of the linear combination of features.

for $h = 1$ to Q **do**
  $r_h^{(0)} \leftarrow x_h^{(0)}$ sorted by decreasing values and ranked
end for
for $h = 1$ to $Q-1$ **do**
  for $k = h+1$ to Q **do**
    /*evaluate the rank coefficient matrix at step 0*/
    $\rho_{h,k}^{(0)} = 1 - \frac{6}{L(L^2-1)}\sum_{i=1}^{L}(r_i^h - r_i^k)^2$
  end for
end for
for $m = 1$ to $Q-1$ **do**
  /*find the pair of classifiers with the minimum rank coefficient */
  $(\sigma, \tau) \leftarrow \mathrm{argmin}_{h,k}\boldsymbol{\rho}^{(m-1)}$
  for $i = 1$ to l_P **do**
    for $j = 1$ to l_N **do**
      $X_{rs} \leftarrow \{(i,j) | I(p_i^\sigma, n_j^\sigma) = r \text{ and } I(p_i^\tau, n_j^\tau) = s\}$ with $r, s = 0, 1$
      $\Delta_{i,j}^\sigma = p_i^\sigma - n_j^\sigma$
      $\Delta_{i,j}^\tau = p_i^\tau - n_j^\tau$
    end for
  end for
  evaluate $F_{10}^{\sigma\tau}$ and $F_{01}^{\sigma\tau}$
  $v(\alpha) \leftarrow F_{10}^{\sigma\tau} + F_{01}^{\sigma\tau}$
  $\alpha_{opt} \leftarrow \max_\alpha v(\alpha)$
  update the combination tree
  $r_{\sigma+\tau}^{(m)} \leftarrow \alpha x_\sigma + (1-\alpha)x_\tau$ sorted by decreasing values and ranked
  if $m < Q-1$ **then**
    for $n = 2$ to $Q-m$ **do**
      $\rho_{\sigma+\tau,n}^{(m)} = 1 - \frac{6}{L(L^2-1)}\sum_{i=1}^{L}(r_i^{\sigma+\tau} - r_i^n)^2$
    end for
  end if
end for
/*update the rank correlation matrix*/
$\boldsymbol{\rho}^{(m)} \leftarrow \boldsymbol{\rho}^{(m-1)}$ eliminating $\rho_\sigma^{(m-1)}$ and $\rho_\tau^{(m-1)}$ and adding $\rho_{\sigma+\tau}^{(m)}$
evaluate $\boldsymbol{\alpha}$ by multiplying the values on the edges of the tree

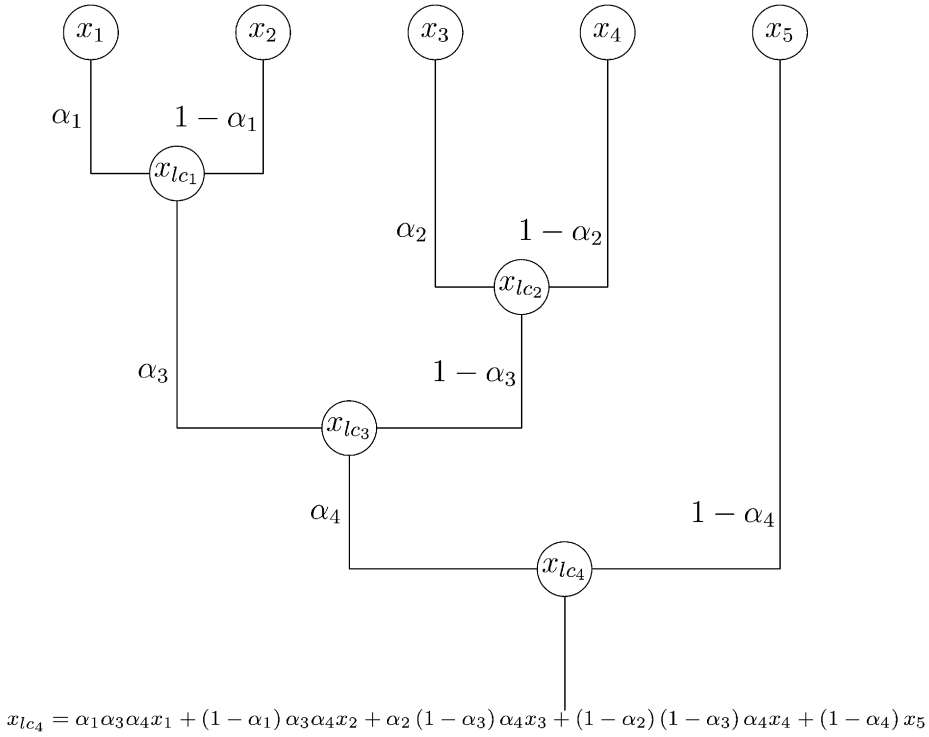

Fig. 5. Example of the tree used to rebuild the weight vector. Traversing the tree from each leave to the root and multiplying the values on the edges it is possible to recover the weight associated to each feature.

Once the procedure has been repeated until a single feature is obtained (i.e. $Q-1$ times), it is necessary to recover the weight for each of the features to be combined. To this aim, a combination tree is built during the evaluation of the weight vector. The original features are the leaves of the tree and a parent node is added when a pair of features is combined. The edges are labelled with the weights assigned to each feature in each step. Multiplying the values found on the edges by traversing the tree (from the leaves down to the root) it is possible to recover the weight for each single feature.

Fig. 5 shows an example of such a tree for a classification problems with five features. In the first step the pair (x_1, x_2) is combined obtaining a new feature $x_{lc_1} = \alpha_1 x_1 + (1-\alpha_1)x_2$; in the second step x_3 and x_4 are combined and so on until we obtain a single feature x_{lc_4}. To recover the weights associated to $x_1 \ldots x_5$, we multiply the weights on the edges that we encounter on the path from the single features to x_{lc_4}. As an example, moving from the feature x_3 towards x_{lc_4} we encounter the weights α_3, $1-\alpha_2$ and α_4 and so the weight relative to x_3 is $\alpha_3(1-\alpha_2)\alpha_4$.

A pseudo code of the whole algorithm is reported in Algorithm 1. Hereafter, we will refer to our approach as Maximum AUC Linear Classifier (MALC).

5. Experiments

In this section some experiments are reported to assess the quality of the proposed method. To this aim, a comparison with other classifiers that work on ranking has been conducted. In particular, three classifiers (described in Section 5.1) have been used: SVM [23], RankBoost [9] and AUC–LPC [16]. All the classifiers have been implemented by means of PRTools [24] toolbox.

To evaluate the performance of our approach experiments on both artificial and real data sets have been performed. The former approach (reported in Section 5.2) has been used to show the behavior of the MALC on known data distributions while experiments on real data (see Section 5.3) have been accomplished to verify the utility of our method even when dealing with real problems. In particular, the admissibility of MALC on some data sets has been proved in a statistical way with respect to the employed methods.

To avoid any bias in the comparison a 10-fold cross validation procedure has been performed on all data sets. In each run 9 folds have been used as training set to train the classifiers and the remaining fold as test set to evaluate the classifiers performance.

It is worth recalling that the comparison has been computed in terms of AUC since we are aiming at the maximization of the ranking quality of the classifier and not at the evaluation of the error rate (or other measures depending on a threshold value). Hence, in our experiments only the value of AUC has been evaluated using the WMW statistic according to Eq. (3).

5.1. Learning algorithms based on ranking

AUC has become a very important topic in the recent literature and therefore several learning algorithms based on ranking

maximization have been proposed. In our experiments a comparison with RankBoost and SV-based rankers has been performed.

5.1.1. RankBoost

A well-performing learning algorithm is proposed in Ref. [9] where a method to combine rankings based on the boosting approach has been introduced. Boosting is a method to produce highly accurate prediction rules by combining many weak rules which may be only moderately accurate [25]. Like all boosting algorithms, RankBoost operates in rounds assuming access to a separate procedure called weak learner that, on each round, is used to produce a weak ranking. Intuitively, RankBoost assigns different weights to different parts of the training data. A high weight assigned to a pair of samples indicates a great importance that the weak learner orders that pair correctly. Weak rankings are usually based on the given ranking features. In particular, in Ref. [9] a weak ranking from the ranking of the feature x_i is derived by comparing the score of x_i on a given instance to a threshold and assigning a default score to instances not correctly ranked by x_i. Then, the weak rankings are used in the boosting algorithm to update the weight distribution concentrating on the pairs whose relative ranking is hardest to determine. The final ranking H is a weighted sum of the weak rankings h:

$$H(\mathbf{x}) = \sum_{\tau=1}^{T} \alpha_\tau h_\tau(\mathbf{x}),$$

where T is the number of rounds.

Even though RankBoost is a multistage non-linear approach a comparison with this ranker is performed as it is one of the best performing methods in the literature. In our experiments a binary version has been used, where the weak learners only threshold a single feature. In total, 10 weak classifiers are employed.

5.1.2. SVM and SV-based rankers

SVM is a well-known classifier not directly built to maximize the ranking performance, but to minimize the error rate; nevertheless, in literature it is considered to be a good ranker among classifiers methods. In our experiments a linear kernel has been considered due to the linear behavior of the proposed approach.

Recently, rank optimizing classifiers based on SVMs have came into focus. In Ref. [13] a minimization of the l_2-norm of the weight vector $\boldsymbol{\alpha}$ with constraints on the ordering of the objects. The optimization problem is defined as follows:

$$\begin{aligned} \min \quad & \|\boldsymbol{\alpha}^2\| + C \sum_{i=1}^{m_P} \sum_{j=1}^{m_N} \xi_{ij} \\ \text{s.t.} \quad & \forall i, j : f(\mathbf{p}_i) - f(\mathbf{n}_j) \geqslant 1 - \xi_{ij}, \quad \xi_{ij} \geqslant 0, \end{aligned}$$

where C is a trade off parameter between the two parts of the objective and the slack variables ξ_{ij} are used to approximate the indicator function in the WMW statistic. By introducing a kernel, thanks to the self duality of the l_2-norm, the above formulation remains valid for nonlinear SVM.

A similar linear weighting of features has been developed in Ref. [16] (called AUC linear programming classifier (AUC–LPC)) and successfully applied to the interstitial lung disease. In this case the optimization problem for a linear classifier can be written as

$$\begin{aligned} \min \quad & \|\boldsymbol{\alpha}_1\| + C \sum_{i=1}^{m_P} \sum_{j=1}^{m_N} \xi_{ij} \\ \text{s.t.} \quad & \forall i, j : \boldsymbol{\alpha}^{\mathrm{T}}(\mathbf{p}_i - \mathbf{n}_j) \geqslant 1 - \xi_{ij}, \quad \xi_{ij} \geqslant 0. \end{aligned}$$

Following Ref. [16], this can be rewritten in a linear programming formulation as

$$\begin{aligned} \min \quad & \sum_h (u_h + v_h) + C \sum_{i=1}^{m_P} \sum_{j=1}^{m_N} \xi_{ij} \\ \text{s.t.} \quad & \forall i, j : (\mathbf{u}^{\mathrm{T}} - \mathbf{v}^{\mathrm{T}})(\mathbf{p}_i - \mathbf{n}_j) \geqslant 1 - \xi_{ij}, \quad \xi_{ij} \geqslant 0, \\ & \forall h : u_h \geqslant 0, \quad v_h \geqslant 0. \end{aligned}$$

Since using the slack variables ξ_{ij} to approximate the indicator function in the WMW statistic has serious drawback (the number of constraints is quadratic in the number of the objects), a strategy to speed up the algorithm is also used based on a randomly sub-sample of the constraints.

The choice of these classifiers for the comparison has been made for different reasons. We used SVMs since as we said before they are the basis for several recent approaches for AUC maximization procedures. Moreover, we employed AUC–LPC since it represents a simplified version of the rank optimizing SVM [13] approach with lower computational complexity; we did not compare with rank SVM since its computational complexity is too high. Since SVM and AUC–LPC are parametric classifiers, different architectures of these algorithms have been employed. In particular, we have varied the C parameter for both SVM and AUC–LPC between 0.1 and 1000. For the sake of readability, in the following tables we only report the best results obtained for these classifiers on each of the employed data sets.

5.2. The artificial data

The first part of our experiments focuses on synthetic data: in particular, some data sets have been built to analyze the characteristics of the classifier. To this aim, a Gaussian distribution for both classes has been used to generate 500 samples varying the number of features Q to show the behavior of the greedy approach for high dimensionality problems. A first set consists of two Gaussian spherical distributed data classes with equal identity covariance matrices and difference Δ_μ between the means of the two classes distributions chosen equal to 0.3 and 1. The second set is a correlated Gaussian data set consisting of two classes with equal covariances matrices. The mean of the first class is equal to zero for all features. The mean of the second class has been chosen equal to 1 and 3 for the first feature and equal to 0 for all the other features. The two co-

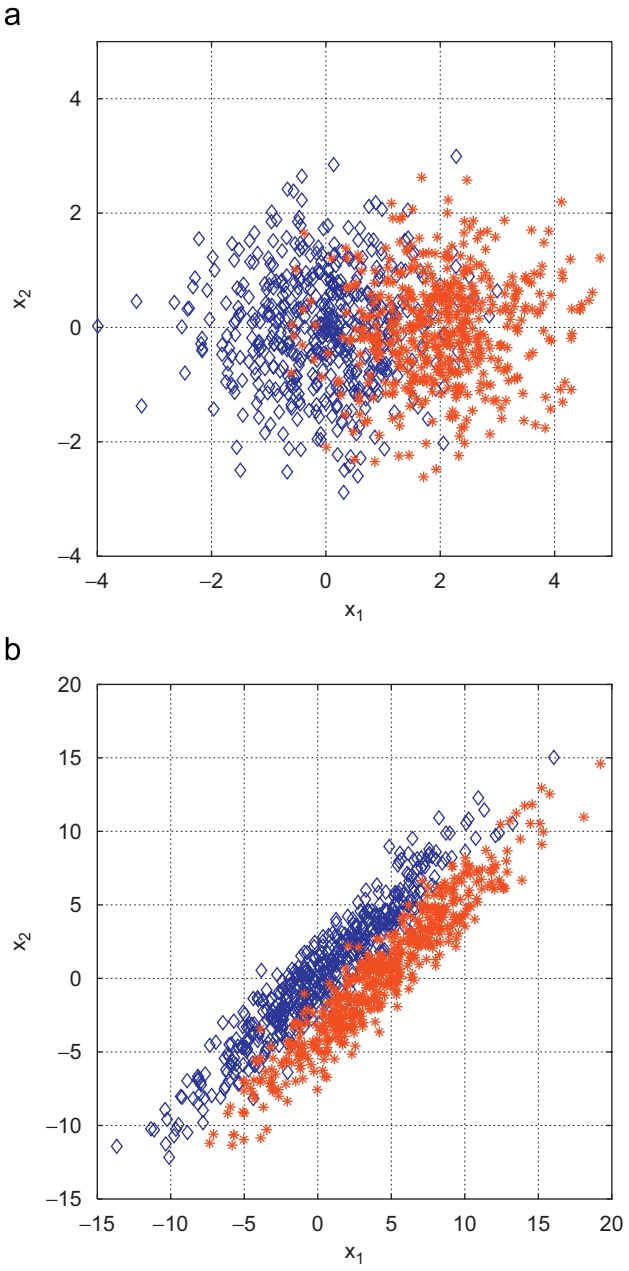

Fig. 6. Scatter plot of the first two features for the 10-dimensional Gaussian uncorrelated data (a) with $\Delta_\mu = 1$ and correlated data (b) with $\Delta_\mu = 3$.

variance matrices are equal diagonal matrices with a variance of 40 for the second feature and a unit variance for all the other features. The data set is rotated in the subspace spanned by the first two features of 45° to construct a strong correlation. A plot of a two-dimensional projection of the first two features is presented in Fig. 6, respectively, for uncorrelated and correlated data.

The results of the comparison of the four employed classifiers are presented in Tables 1 and 2 for uncorrelated data and in Tables 3 and 4 for correlated class distributions. Each cell of the tables contains a value corresponding to the mean (and the standard deviation in parentheses) of AUC relative to the performance of each classifier on each data set for the relative number of features.

Table 1
Results on the test set for a Gaussian data set with uncorrelated class distributions, Δ_μ equal to 0.3 and variable number of features

Feature size	Classifiers			
	MALC	SVM	AUC–LPC	RankBoost
5	0.621 (0.073)	0.615 (0.074)	0.615 (0.073)	0.620 (0.068)
10	0.595 (0.054)	0.552 (0.045)	0.562 (0.060)	0.591 (0.027)
30	0.584 (0.095)	0.559 (0.082)	0.570 (0.110)	0.580 (0.047)
50	0.555 (0.067)	0.550 (0.072)	0.548 (0.048)	0.596 (0.061)
75	0.540 (0.042)	0.538 (0.068)	0.522 (0.071)	0.589 (0.064)
100	0.533 (0.044)	0.528 (0.068)	0.512 (0.047)	0.567 (0.043)

Table 2
Results on the test set for a Gaussian data set with uncorrelated class distributions, Δ_μ equal to 1 and variable number of features

Feature size	Classifiers			
	MALC	SVM	AUC–LPC	RankBoost
5	0.939 (0.019)	0.937 (0.023)	0.936 (0.022)	0.931 (0.022)
10	0.920 (0.047)	0.914 (0.047)	0.910 (0.049)	0.913 (0.047)
30	0.929 (0.032)	0.921 (0.038)	0.918 (0.037)	0.931 (0.025)
50	0.915 (0.028)	0.880 (0.038)	0.888 (0.024)	0.916 (0.027)
75	0.741 (0.059)	0.726 (0.079)	0.701 (0.074)	0.800 (0.048)
100	0.886 (0.037)	0.843 (0.058)	0.841 (0.051)	0.908 (0.022)

Table 3
Results on the test set for a Gaussian data set with correlated class distributions, Δ_μ equal to 1 and variable number of features

Feature size	Classifiers			
	MALC	SVM	AUC–LPC	RankBoost
5	0.809 (0.055)	0.771 (0.052)	0.767 (0.053)	0.717 (0.070)
10	0.796 (0.015)	0.773 (0.077)	0.771 (0.069)	0.708 (0.026)
30	0.756 (0.041)	0.740 (0.052)	0.733 (0.054)	0.667 (0.065)
50	0.655 (0.501)	0.707 (0.065)	0.726 (0.061)	0.603 (0.090)
75	0.634 (0.080)	0.686 (0.067)	0.666 (0.057)	0.589 (0.068)
100	0.632 (0.076)	0.633 (0.060)	0.641 (0.064)	0.565 (0.042)

Table 4
Results on the test set for a Gaussian data set with correlated class distributions, Δ_μ equal to 3 and variable number of features

Feature size	Classifiers			
	MALC	SVM	AUC–LPC	RankBoost
5	0.980 (0.024)	0.979 (0.023)	0.979 (0.023)	0.843 (0.047)
10	0.985 (0.011)	0.984 (0.015)	0.985 (0.014)	0.862 (0.034)
30	0.975 (0.014)	0.975 (0.018)	0.971 (0.018)	0.835 (0.053)
50	0.972 (0.011)	0.969 (0.021)	0.972 (0.016)	0.844 (0.042)
75	0.979 (0.015)	0.968 (0.019)	0.970 (0.021)	0.825 (0.069)
100	0.970 (0.055)	0.963 (0.022)	0.977 (0.019)	0.847 (0.061)

Firstly, let us analyze the results obtained on uncorrelated data. In this case, it is possible to highlight the good behavior of MALC when the dimensionality of the feature space is not high. In fact, MALC gives the highest mean value among the

four classifiers on all the employed data sets until the value of Q is lower than 50. When Q becomes greater than 50 RankBoost exhibits the best performance but MALC is still better than SVM and AUC–LPC. The reason for the good performance of RankBoost in high dimension space can be found in its characteristics since it is a multistage approach that trains different classifiers on different samples in each round of its procedure. On the contrary, the three linear classifiers have a different behavior on this data: if we look at Tables 1 and 2, we, we can observe that MALC loses at least 5% in AUC when passing from 30 to 100 features and a similar behavior is shown by SVM and AUC–LPC that, as reported by the same authors [23,16], suffer the high dimensionality of data. Also in our approach, some problems can occur when dealing with high number of features since we are using a greedy approach based on an iterative pairwise coupling that do not recover errors that could be generated in each step of the algorithm.

For correlated data the situation is more complicated to analyze since there is no clear dominance of one method above the others. In this case, RankBoost does not exhibit good performance in comparison with the other rankers probably due to the correlation among the weak classifiers that are used to perform the boosting approach. Moreover, for a low number of features it is possible to show a better behavior of MALC with respect to the other two linear rankers, at least until Q is lower than 30. When we increase the number of features, MALC exhibits a different behavior according to the overlapping of the classes. If the two classes are well separated (see Table 4) MALC is the best classifier except one case ($Q = 100$ where AUC–LPC performs better). On the other hand, if the data are almost completely overlapping (see Table 3) the performance of MALC quickly decreases when the feature number grows and even for $Q = 50$ it exhibits a lower AUC value than SVM and AUC–LPC.

Another type of experiment has been performed to analyze the robustness of our ranker with respect to a monotonic scaling of the features of each data set. We scaled data both in a linear and a nonlinear way. The first model has been used to show the behavior of classifiers with respect to data normalization even though a linear scaling can be undone by a simple pre-processing step. The nonlinear way, instead, cannot be corrected in a pre-processing step (to do that we should know the scaling model) and so it has been performed to study the scale dependence of some classifiers. In summary, three different types of scaling have been evaluated for each sample $\mathbf{x}$: a linear scaling $a\mathbf{x} + b$ with a and b randomly chosen, an exponential scaling $\exp(\mathbf{x})$ and a random exponential scaling $a^{\mathbf{x}}$ with a randomly chosen.

In Tables 5–8 we report the results relative to the employed artificial data sets. Comparing these tables with the corresponding results on unscaled data set we can conclude that SVM and AUC–LPC are not entirely stable towards the considered scaling while RankBoost is completely invariant. MALC shows a very robust (almost invariant) behavior for a linear scaling probably due to the linear approach employed for the combination weight search. However, our approach is not so stable for other transformations even if it maintains good performance on all the scaling transformations for a number of features lower than 50.

In summary, the analysis of artificial data has shown that the proposed method performs well for data with low dimensionality (less than 50 features) both for correlated and uncorrelated data distributions, i.e. our ranker is admissible in comparison with well-known methods in literature. It has also been shown that MALC seems to be robust towards scaling variations but is not as robust as RankBoost.

5.3. Experiments on real data sets

In this section we propose another type of experiment based on real data sets. Our goal here is to verify if MALC is an admissible classifier, i.e. if there exist some real cases in which no other classifier performs equal to or better than MALC.

To this aim, the proposed method has been tested on several data sets publicly available at the UCI machine learning repository [26]. All of them have two classes[2] and a variable number of numerical input features.

In order to have a statistical validation of the obtained results, some tests have been performed. Statistics offers powerful specialized procedures for testing the significance of differences between multiple means and, in this case, ANOVA [27] is the typical choice. However, it is worth noting that ANOVA is based on two assumptions, i.e. the samples are drawn from normal distributions and the distributions have equal variance. Since this is not assured in our experiments, we employed the Friedman test [28], a nonparametric equivalent of the ANOVA that results to be a more general test even if it is less powerful than ANOVA when ANOVA's assumptions are met [29]. In our case, the null hypothesis for the Friedman test corresponds to a no statistically significant difference between the mean AUC of the employed methods. Therefore, when the null hypothesis is rejected there is a statistical difference among the classifiers.

In this case, we can proceed with a post hoc test to find out which classifiers exhibits a statistically different behavior. Two different situations can be evaluated: to compare all the classifiers between each other or to compare all classifiers with a control method. Since the power of a post hoc test is much greater in the second case when all classifiers are compared with a single method and since we are testing if MALC gives better performance than the existing methods, we focus on one of this procedure: the Holm step-down test [29,30].

The results obtained for the four classifiers are reported in Table 9 on 27 data sets, six of which (Breast, Monks1, Monks2, Monks3, Votes and TicTacToe) contain only nominal features. Each cell of the table contains a value corresponding to the mean (and the standard deviation in parentheses) of AUC relative to the performance of each classifier on each data set. A bold value in the table indicates that the corresponding method on that data set has a statistically significant lower performance than MALC according to the Holm procedure. If the value

[2] Three multiclass data sets (Glass, Waveform and Wine) have also been used. In this case, a One vs. All approach has been applied to select one of the classes from the multiclass data set.

Table 5
Results on the test set for a Gaussian data set with uncorrelated class distributions, Δ_μ equal to 0.3 and variable number of features for different scaling of the features

Feature size	MALC	SVM	AUC–LPC	RankBoost
Linear scaling				
5	0.621 (0.070)	0.616 (0.073)	0.614 (0.069)	0.620 (0.068)
10	0.594 (0.031)	0.564 (0.026)	0.575 (0.044)	0.591 (0.027)
30	0.583 (0.054)	0.596 (0.050)	0.615 (0.061)	0.580 (0.047)
50	0.560 (0.052)	0.579 (0.061)	0.562 (0.034)	0.596 (0.061)
75	0.552 (0.028)	0.555 (0.041)	0.551 (0.024)	0.589 (0.064)
100	0.540 (0.026)	0.560 (0.039)	0.545 (0.024)	0.567 (0.043)
Exponential scaling				
5	0.624 (0.073)	0.624 (0.070)	0.609 (0.073)	0.620 (0.068)
10	0.616 (0.061)	0.595 (0.051)	0.596 (0.051)	0.591 (0.027)
30	0.606 (0.061)	0.591 (0.064)	0.595 (0.074)	0.580 (0.047)
50	0.572 (0.041)	0.565 (0.041)	0.558 (0.038)	0.595 (0.062)
75	0.565 (0.040)	0.558 (0.045)	0.581 (0.042)	0.589 (0.064)
100	0.579 (0.054)	0.583 (0.050)	0.549 (0.038)	0.567 (0.043)
Random exponential scaling				
5	0.632 (0.061)	0.577 (0.051)	0.578 (0.040)	0.620 (0.068)
10	0.600 (0.074)	0.578 (0.051)	0.603 (0.058)	0.591 (0.027)
30	0.603 (0.072)	0.591 (0.063)	0.606 (0.073)	0.580 (0.047)
50	0.580 (0.056)	0.560 (0.048)	0.575 (0.058)	0.595 (0.062)
75	0.556 (0.029)	0.574 (0.057)	0.576 (0.068)	0.589 (0.064)
100	0.564 (0.057)	0.533 (0.025)	0.545 (0.023)	0.567 (0.043)

Table 6
Results on the test set for a Gaussian data set with uncorrelated class distributions, Δ_μ equal to 1 and variable number of features for different scaling of the features

Feature size	MALC	SVM	AUC–LPC	RankBoost
Linearscaling				
5	0.939 (0.020)	0.937 (0.023)	0.939 (0.021)	0.931 (0.022)
10	0.919 (0.047)	0.915 (0.047)	0.909 (0.049)	0.913 (0.047)
30	0.932 (0.028)	0.922 (0.037)	0.905 (0.052)	0.931 (0.025)
50	0.919 (0.026)	0.882 (0.037)	0.873 (0.026)	0.916 (0.027)
75	0.732 (0.049)	0.724 (0.082)	0.702 (0.074)	0.800 (0.048)
100	0.888 (0.037)	0.835 (0.053)	0.828 (0.061)	0.908 (0.022)
Exponential scaling				
5	0.939 (0.019)	0.938 (0.023)	0.936 (0.023)	0.931 (0.022)
10	0.922 (0.046)	0.914 (0.047)	0.911 (0.051)	0.913 (0.047)
30	0.934 (0.027)	0.918 (0.038)	0.913 (0.048)	0.931 (0.025)
50	0.924 (0.027)	0.866 (0.023)	0.869 (0.027)	0.916 (0.027)
75	0.783 (0.060)	0.735 (0.072)	0.697 (0.057)	0.800 (0.048)
100	0.897 (0.035)	0.828 (0.064)	0.827 (0.053)	0.908 (0.022)
Random exponential scaling				
5	0.938 (0.019)	0.931 (0.022)	0.925 (0.029)	0.931 (0.022)
10	0.922 (0.047)	0.915 (0.048)	0.886 (0.059)	0.913 (0.047)
30	0.932 (0.028)	0.908 (0.049)	0.886 (0.048)	0.931 (0.025)
50	0.924 (0.025)	0.856 (0.046)	0.826 (0.036)	0.916 (0.027)
75	0.747 (0.068)	0.681 (0.069)	0.657 (0.081)	0.800 (0.048)
100	0.911 (0.034)	0.804 (0.076)	0.791 (0.067)	0.908 (0.022)

is underlined MALC exhibits lower performance compared to that method while if the value is in normal style it means that the corresponding method has undistinguishable performance from MALC. When the values in a row of the table are signed with an asterisk there is no statistical difference according to the Friedman test (i.e. the null hypothesis cannot be rejected). All the tests (both the Friedman and the Holm test) have been performed with a level of significance equal to 0.05.

From these results we can see that there is a statistical difference among the employed methods according to the Friedman test in 18 of the 27 data sets. In two cases (i.e. Breast and Monks2 data sets) the null hypothesis is rejected according to

Table 7
Results on the test set for a Gaussian data set with correlated class distributions, Δ_μ equal to 1 and variable number of features for different scaling of the features

Feature size	MALC	SVM	AUC–LPC	RankBoost
Linear scaling				
5	0.726 (0.080)	0.771 (0.052)	0.766 (0.053)	0.717 (0.070)
10	0.714 (0.129)	0.774 (0.075)	0.765 (0.072)	0.708 (0.026)
30	0.733 (0.043)	0.739 (0.050)	0.727 (0.052)	0.667 (0.065)
50	0.594 (0.061)	0.708 (0.063)	0.709 (0.064)	0.603 (0.090)
75	0.595 (0.052)	0.689 (0.065)	0.663 (0.066)	0.589 (0.068)
100	0.555 (0.066)	0.630 (0.068)	0.627 (0.045)	0.565 (0.042)
Exponential Scaling				
5	0.721 (0.060)	0.627 (0.082)	0.701 (0.073)	0.717 (0.070)
10	0.695 (0.113)	0.580 (0.046)	0.666 (0.107)	0.708 (0.026)
30	0.726 (0.052)	0.564 (0.038)	0.667 (0.036)	0.667 (0.065)
50	0.596 (0.054)	0.568 (0.054)	0.630 (0.060)	0.603 (0.090)
75	0.559 (0.039)	0.561 (0.049)	0.615 (0.059)	0.589 (0.068)
100	0.559 (0.038)	0.553 (0.030)	0.605 (0.092)	0.565 (0.042)
Random exponential scaling				
5	0.594 (0.058)	0.607 (0.059)	0.612 (0.069)	0.717 (0.070)
10	0.622 (0.098)	0.555 (0.035)	0.645 (0.102)	0.708 (0.026)
30	0.581 (0.051)	0.587 (0.059)	0.587 (0.055)	0.667 (0.065)
50	0.586 (0.068)	0.577 (0.057)	0.618 (0.075)	0.603 (0.090)
75	0.555 (0.051)	0.687 (0.070)	0.636 (0.050)	0.589 (0.068)
100	0.570 (0.048)	0.530 (0.043)	0.549 (0.067)	0.565 (0.042)

Table 8
Results on the test set for a Gaussian data set with correlated class distributions, Δ_μ equal to 3 and variable number of features for different scaling of the features

Feature size	MALC	SVM	AUC–LPC	RankBoost
Linear scaling				
5	0.980 (0.023)	0.980 (0.023)	0.973 (0.026)	0.843 (0.047)
10	0.985 (0.011)	0.984 (0.014)	0.974 (0.020)	0.862 (0.034)
30	0.936 (0.136)	0.977 (0.017)	0.957 (0.024)	0.835 (0.053)
50	0.942 (0.106)	0.969 (0.021)	0.955 (0.033)	0.844 (0.042)
75	0.980 (0.015)	0.963 (0.018)	0.952 (0.033)	0.825 (0.069)
100	0.697 (0.057)	0.980 (0.021)	0.961 (0.024)	0.847 (0.061)
Exponential scaling				
5	0.961 (0.035)	0.653 (0.046)	0.930 (0.044)	0.843 (0.047)
10	0.944 (0.043)	0.775 (0.036)	0.927 (0.028)	0.862 (0.034)
30	0.897 (0.124)	0.790 (0.068)	0.902 (0.050)	0.835 (0.053)
50	0.897 (0.082)	0.702 (0.086)	0.871 (0.040)	0.844 (0.042)
75	0.934 (0.032)	0.615 (0.076)	0.834 (0.057)	0.825 (0.069)
100	0.698 (0.075)	0.859 (0.043)	0.868 (0.043)	0.847 (0.061)
Random exponential scaling				
5	0.955 (0.033)	0.686 (0.111)	0.882 (0.036)	0.843 (0.047)
10	0.772 (0.036)	0.727 (0.048)	0.748 (0.032)	0.862 (0.034)
30	0.857 (0.104)	0.719 (0.059)	0.797 (0.058)	0.835 (0.053)
50	0.724 (0.053)	0.577 (0.064)	0.610 (0.063)	0.844 (0.042)
75	0.700 (0.081)	0.694 (0.086)	0.709 (0.089)	0.825 (0.069)
100	0.638 (0.073)	0.957 (0.031)	0.842 (0.043)	0.847 (0.061)

the Friedman test but the post hoc test fails to detect which classifiers are statistically different due to the lower power of the post hoc test with respect to Friedman test (in such a case the only thing that we can say is that some of the algorithms differ but no other conclusion can be drawn). On the 18 data sets for which a statistical difference is found according to Friedman test we can consider that only in two cases (Glass2 and Thyroidsub) MALC is worse than one of the other methods (in these cases RankBoost) while the proposed method results in five cases better than SVM, in seven cases better than AUC–LPC and in nine cases better than RankBoost.

In conclusion, we have shown that also on real data the proposed approach can be profitably used to maximize AUC on the analyzed problem. In fact, the reported results show the

Table 9
Results in terms of AUC obtained in the experiments performed on real data set (see text for notation)

Data sets	Classifiers			
	MALC	SVM	AUC–LPC	RankBoost
Arrhythmia	0.783 (0.066)	**0.720** (**0.097**)	0.765 (0.095)	0.736 (0.070)
Biomed	0.968 (0.044)	0.958 (0.041)	0.961 (0.040)	**0.927** (**0.069**)
Breast	0.996 (0.005)	0.995* (0.006*)	0.994* (0.006*)	0.990* (0.008*)
Cancer_wpbc	0.762 (0.041)	0.780 (0.055)	0.762 (0.047)	0.741 (0.046)
Diabetes	0.811 (0.038)	0.826 (0.035)	0.821 (0.042)	0.834 (0.058)
Glass1	0.841 (0.084)	**0.840** (**0.093**)	**0.827** (**0.095**)	0.870 (0.056)
Glass2	0.660 (0.048)	0.627 (0.046)	0.714 (0.059)	0.748 (0.038)
Glass3	0.838 (0.045)	0.804 (0.054)	0.782 (0.082)	0.802 (0.031)
Glass4	0.943 (0.086)	0.940 (0.031)	0.925 (0.057)	0.937 (0.025)
Glass5	0.921 (0.029)	0.920 (0.033)	0.948 (0.023)	0.912 (0.058)
Heart	0.895 (0.057)	0.895 (0.057)	0.902 (0.060)	**0.542** (**0.058**)
Hepatitis	0.855 (0.048)	**0.784** (**0.074**)	**0.802** (**0.067**)	**0.790** (**0.053**)
Ionosphere	0.893 (0.087)	0.895 (0.056)	0.880 (0.070)	**0.701** (**0.120**)
Liver	0.720 (0.069)	0.714 (0.044)	0.718 (0.063)	0.720 (0.085)
Monks1	0.612 (0.064)	0.612 (0.064)	0.611 (0.057)	**0.545** (**0.057**)
Monks2	0.552 (0.046)	0.551* (0.036*)	0.555* (0.038*)	0.553* (0.053*)
Monks3	0.654 (0.057)	0.661 (0.044)	0.618 (0.068)	**0.560** (**0.062**)
Sonar	0.821 (0.046)	0.845 (0.042)	0.85 (0.050)	0.830 (0.030)
TicTacToe	0.649 (0.063)	**0.614** (**0.048**)	0.626 (0.049)	0.683 (0.050)
Thyroidsub	0.987 (0.013)	**0.973** (**0.017**)	0.984 (0.014)	0.998 (0.001)
Votes	0.983 (0.012)	0.987 (0.012)	0.989 (0.012)	0.971 (0.032)
Waveform1	0.937 (0.029)	0.937 (0.027)	**0.931** (**0.030**)	**0.921** (**0.027**)
Waveform2	0.940 (0.037)	0.922 (0.027)	**0.919** (**0.030**)	0.935 (0.031)
Waveform3	0.953 (0.036)	0.953 (0.037)	**0.948** (**0.036**)	**0.942** (**0.033**)
Wine1	0.996 (0.013)	1.000 (0.000)	**0.971** (**0.042**)	0.999 (0.004)
Wine2	0.994 (0.014)	0.993 (0.014)	**0.977** (**0.037**)	**0.990** (**0.017**)
Wine3	0.999 (0.005)	0.999 (0.005)	0.980 (0.063)	0.995 (0.010)

admissibility of the classifier on some of the considered data sets and therefore, our ranker is able to compete with other well-known methods proposed in literature.

6. Conclusions

Since AUC is independent on priors and misclassification costs it is often a more suitable measure than the classification error when dealing with problems where imbalanced class priors or misclassification costs are often present. Moreover, there are real situations in which the ordering is more important than classification. In these situations classifiers directly built to optimize measures related to the error rate such as accuracy or RMSE are not advantageous. In Ref. [7] it has been proved that algorithms designed to minimize the error rate may not lead to the best possible AUC thus motivating the use of algorithms and combiners directly optimizing AUC.

As no trick is known for directly optimizing AUC, the approaches proposed in literature are based on the optimization of a related measure (as SVM does) or the application of heuristics to reduce the dependence on the number of samples. For example, AUC–LPC reduces the complexity by randomly subsampling the constraints of the minimization problem while RankBoost uses a multistage approach applying several times a weak ranker on a part of the original problem and combining the results obtained by each weak ranker.

The approach proposed in this paper, instead, after an analysis of the linear discriminant functions in the light of the ROC analysis, we have proposed a nonparametric classifier that performs a linear combination of features directly maximizing AUC. The approach relies on the Wilcoxon–Mann–Whitney statistic and the pairwise feature evaluation provides a new procedure essentially different from other techniques performing AUC optimization. The greedy approach let us reduce the computational complexity of the algorithm that results in a faster implementation than AUC–LPC (and thus than all the other methods based on SVMs). With respect to RankBoost, MALC has comparable processing times, but our model is easily interpretable in feature space since at each step it can be seen as the weighted projection of samples from a plane to a straight line. This could allow the use of kernel functions to generalize the approach in the case of data that cannot be linearly ordered, which will be enhanced in a future research work.

The extensive experiments confirmed that the proposed rule exhibits good performance in comparison to the other well-known methods from the literature using both artificial and real data sets. In particular, it seems to be advantageous when analyzing data with a low number of features and, even if it is not as robust against scaling variations as RankBoost, it gives very good classifiers in real world problems.

References

[1] F. Provost, T. Fawcett, Robust classification for imprecise environments, Mach. Learn. 42 (3) (2001) 203–231.

[2] T. Fawcett, An introduction to ROC analysis, Pattern Recognition Lett. 27 (2006) 861–874.
[3] P.A. Flach, The geometry of roc space: understanding machine learning metrics through roc isometrics, Proceedings of the 20th International Conference on Machine Learning, 2003.
[4] F. Tortorella, A roc-based reject rule for dichotomizers, Pattern Recognition Lett. 26 (2005) 167–180.
[5] A.P. Bradley, The use of the area under the *roc* curve in the evaluation of machine learning algorithms, Pattern Recognition 30 (1997) 1145–1159.
[6] J.A. Hanley, B.J. McNeil, The meaning and use of the area under a receiver operating characteristic (roc) curve, Radiology 143 (1982) 29–36.
[7] C. Cortes, M. Mohri, Auc optimization vs. error rate minimization, Advances in Neural Information Processing Systems (NIPS 2003).
[8] J. Huang, C.X. Ling, Using auc and accuracy in evaluating learning algorithms, IEEE Trans. Knowledge Data Eng. 17 (2005) 299–310.
[9] Y. Freund, R. Iyer, R.E. Schapire, Y. Singer, An efficient boosting algorithm for combining preferences, J. Mach. Learn. Res. 4 (2003) 933–969.
[10] C. Rudin, C. Cortes, M. Mohri, R. Schapire, Margin-based ranking meets boosting in the middle, Proceedings of the 18th Annual Conference on Computational Learning Theory, 2005.
[11] C. Ferri, P. Flach, J. Hernandez-Orallo, Learning decision trees using the area under the roc curve, Proceedings of the 19th International Conference on Machine Learning, 2002.
[12] A. Herschtal, B. Raskutti, Optimising area under the roc curve using gradient descent, Proceedings of the 21st International Conference on Machine Learning, 2004.
[13] A. Rakotomamonjy, Optimizing area under roc curve with svms, Proceedings of the First Workshop on ROC Analysis and Artificial Intelligence, 2004.
[14] U. Brefeld, T. Scheffer, Auc maximizing support vector learning, Proceedings of the 22nd International Conference on Machine Learning—Workshop on ROC Analysis in Machine Learning, 2005.
[15] K. Ataman, N. Street, Y. Zhang, Learning to rank by maximizing auc with linear programming, Proceedings of the IEEE International Joint Conference on Neural Networks, 2006, pp. 123–129.
[16] D.J.M. Tax, R.P.W. Duin, Y. Arzhaeva, Linear model combining by optimizing the area under the roc curve, Proceedings of the 18th IEEE International Conference on Pattern Recognition, 2006, pp. 119–122.
[17] C. Marrocco, M. Molinara, F. Tortorella, Exploiting auc for optimal linear combination of dichotomizers, Pattern Recognition Lett. 27 (8) (2006) 900–907.
[18] D.J. Hand, R.J. Till, A simple generalization of the area under the roc curve to multiple class classification problems, Mach. Learn. 45 (2001) 171–186.
[19] H.B. Mann, D.R. Whitney, On a tests whether one of two random variables is stochastically larger than the other, Ann. Math. Statist. 18 (1947) 50–60.
[20] L. Yan, R. Dodier, M.C. Mozer, R. Wolniewicz, Optimizing classifier performance via the Wilcoxon–Mann–Whitney statistics, Proceedings of the 20th International Conference on Machine Learning, 2003, pp. 848–855.
[21] L.I. Kuncheva, C.J. Whitaker, C.A. Shipp, R.P.W. Duin, Is independence good for combining classifiers?, Proceedings of the 15th IEEE International Conference on Pattern Recognition, 2000, pp. 168–171.
[22] E.L. Lehmann, H.J.M. D'Abrera, Nonparametrics. Statistical Methods Based on Ranks, McGraw Hill International Book Company, 1975.
[23] V.N. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.
[24] R.P.W. Duin, Prtools version 3.0, a matlab toolbox for pattern recognition, 2000 ⟨http://www.prtools.org⟩.
[25] Y. Freund, R.E. Schapire, A decision-theoretic generalization of online learning and an application to boosting, J. Comput. System Sci. 55 (1) (1997) 119.
[26] C. Blake, E. Keogh, C.J. Merz, Uci repository of machine learning databases, 1998 ⟨www.ics.uci.edu/∼mlearn/MLRepository.html⟩.
[27] R.A. Fisher, Statistical Methods and Scientific Inference, second ed., Hafner Publishing Co., 1959.
[28] M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, J. Amer. Statist. Assoc. 32 (1937) 675–701.
[29] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[30] S. Holm, A simple sequentially rejective multiple test procedure, Scand. J. Statist. 6 (1979) 65–70.

About the Author—CLAUDIO MARROCCO received the M.Sc. degree in Telecommunications Engineering in 2003 and the Ph.D. degree in Information and Electrical Engineering in 2007, both from University of Cassino, Italy. Currently, he is in a postdoctoral position. His research interests include statistical learning, multiple expert systems and medical image analysis and classification. He is a member of the International Association for Pattern Recognition (IAPR).

About the Author—ROBERT P.W. DUIN studied applied physics at Delft University of Technology in the Netherlands. In 1978, he received the Ph.D. degree for a thesis on the accuracy of statistical pattern recognizers. In his research, he included various aspects of the automatic interpretation of measurements, learning systems, and classifiers. Between 1980 and 1990, he studied and developed hardware architectures and software configurations for interactive image analysis. After this period, his interest was redirected to neural networks and pattern recognition. At this moment, he is an associate professor in the Faculty of Electrical Engineering, Mathematics, and Computer Science at Delft University of Technology. His present research is in the design, evaluation, and application of algorithms that learn from examples. This includes neural network classifiers, support vector machines, and classifier combining strategies. Recently, he started to study alternative object representations for classification and became thereby interested in the use of relational methods for pattern recognition and in the possibilities to learn domain descriptions. He is a member of the IEEE and the IEEE Computer Society.

About the Author—FRANCESCO TORTORELLA received the M.Sc. degree with honors in Electronic Engineering and the Ph.D. Degree in Electronic and Computer Engineering in 1995, both from the University of Naples "Federico II", Italy. From 1991 to 1998 he was a member of the research group on image processing and understanding in the Dipartimento di Informatica e Sistemistica at the Universitu of Naples "Federico II". In 1998 he joined the Dipartimento di Automazione, Elettromagnetismo, Ingegneria dell' Informazione e Matematica Industriale, University of Cassino, where he is now Full Professor of Computer Architecture and Image Understanding. Prof. Tortorella has authored over 70 research papers in international journals and conference proceedings and has served as referee for many international journals. His current research interests include classification techniques, statistical learning, neural networks, medical image analysis and interpretation, document processing and understanding. Prof. Tortorella is a member of the IEEE Computer Society and of the IEEE Computational Intelligence Society. He is also member of the International Association for Pattern Recognition (IAPR).