Noname manuscript No.
(will be inserted by the editor)

Linear components of quadratic classifiers

José R. Berrendero · Javier Cárcamo

Received: date / Accepted: date

Abstract We obtain a decomposition of any quadratic classifier in terms of products of hyperplanes. These hyperplanes can be viewed as relevant linear components of the quadratic rule (with respect to the underlying classification problem). As an application, we introduce the associated *multidirectional classifier*; a piecewise linear classification rule induced by the approximating products. Such a classifier is useful to determine linear combinations of the predictor variables with ability to discriminate. We also show that this classifier can be used as a tool to reduce the dimension of the data and helps identify the most important variables to classify new elements. Finally, we illustrate with a real data set the use of these linear components to construct oblique classification trees.

Keywords Supervised classification · Fisher linear discriminant analysis · Quadratic discriminant analysis · Reduction of the dimension · Feature extraction · Oblique classification trees

Mathematics Subject Classification (2000) 62H30

1 Introduction and motivation

We consider the binary supervised classification problem, that is, we seek to classify an object in one of two possible and well defined groups using a vector of predictor

This research was supported by the Spanish MCyT grant MTM2016-78751-P.

José R. Berrendero
Departamento de Matemáticas, Universidad Autónoma de Madrid, 28049 Madrid (Spain)
Tel.: +0034914976690
E-mail: joser.berrendero@uam.es

Javier Cárcamo
Departamento de Matemáticas, Universidad Autónoma de Madrid, 28049 Madrid (Spain)
Tel.: +0034914977635
E-mail: javier.carcamo@uam.es

variables. To build a classification or discriminant rule, it is assumed that the true membership is known for a training sample of observations. Although many procedures have been developed to face this problem, the simplest and most popular ones are those based on linear and quadratic functions of the predictors, which lead to the so called *linear discriminant analysis* (LDA) and *quadratic discriminant analysis* (QDA), respectively.

LDA has many desirable features in comparison with more complex methods: linear classifiers are extremely easy to interpret and allow the practitioner to identify the most relevant variables to classify new elements. It is also optimal when the predictor vector is normally distributed and the covariance matrices in both groups coincide (homoscedasticity). Moreover, despite its simplicity, it has been shown that in practice LDA achieves a good performance in a great variety of scenarios (see Hand (2006)).

QDA is arguably the most important extension of LDA and can be motivated from different perspectives. For instance, it is theoretically optimal when the predictor vector is normally distributed in both groups and each group has a different covariance structure (heteroscedasticity). In such a case, it is expected that QDA outperforms LDA. However, as QDA requires the estimation of more parameters than LDA, the practical improvement is only observed when the training sample size is large compared with the dimension of the predictor vector (see Wald and Kronmal (1977)). It is also clear that quadratic rules are much more difficult to interpret in terms of the original variables than the linear ones because products of the predictors take part in the procedure. Devroye et al (1996) or Hastie et al (2009) are two helpful general references addressing further advantages and drawbacks of different methodologies in supervised classification.

The aim of this paper is to provide an approximation of any quadratic classifier in terms of certain products of hyperplanes that can be regarded as relevant linear components of the quadratic rule. The outcome of the procedure is a collection of linear combinations of the original variables that are potentially relevant for classifying. One of the possible and natural applications of these linear combinations is to build new classifiers (related to the initial quadratic rule) as easily interpretable as LDA, improving over LDA in the situations in which QDA is preferable. The procedure to obtain these linear components is straightforward from a computational point of view and with an elementary implementation because we only need to diagonalize an appropriate symmetric matrix that can be easily estimated with the training sample. Further, in general, some of these approximating hyperplanes are not significant for classification purposes and can be therefore disregarded. In this way, the most important linear combinations (for the classification task) of the predictor variables can be detected. Consequently, the proposed methodology can be applied to reduce the dimension of the problem at hand by considering those combinations with more discriminative information. In some occasions, the output also allows identifying the most relevant variables to classify new observations. Other potential applications, such as the construction of oblique decision trees, that is, classification trees that test a linear combination of the attributes at each internal node, can also be easily implemented from the ideas of this paper. Finally, from a theoretical point of view, a piecewise linear decomposition of a quadratic rule provides an idea of its geometric complexity,

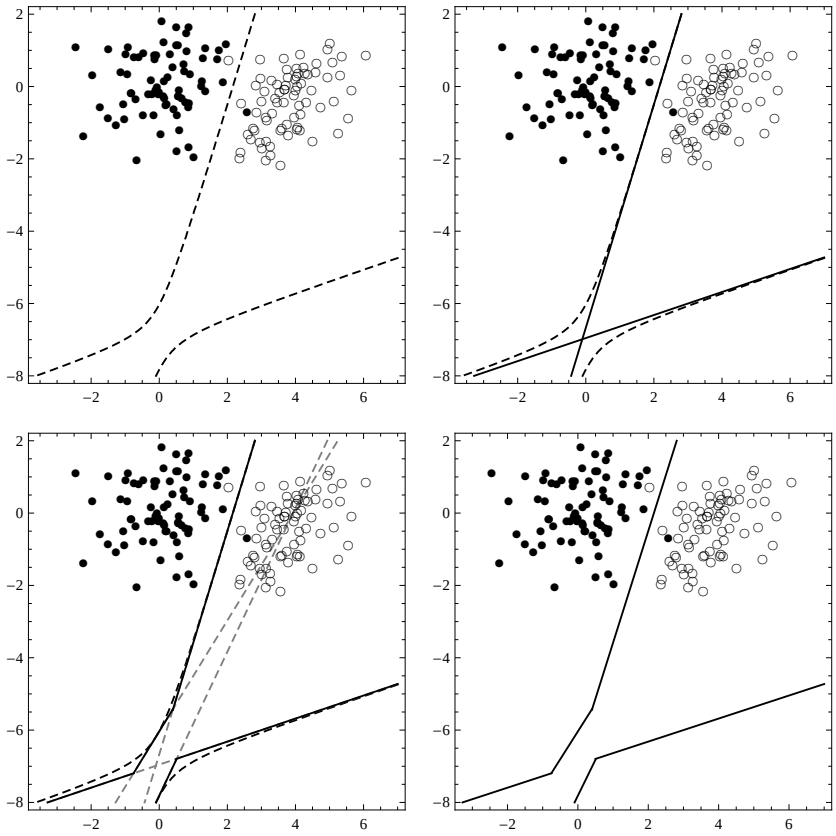

Fig. 1 The training sample and the optimal rule (upper-left panel); the approximation with one product of hyperplanes in solid lines (upper-right panel); the approximation with two products of hyperplanes (lower left panel); and the multidirectional classifier with two products (lower-right panel).

because we find out the number of hyperplanes required to wrap up the classification boundary.

To illustrate the potential applications of our results, as well as the spirit behind them, we consider three simple examples corresponding to heteroscedastic two-dimensional normal training samples (see Figures 1–3). These examples roughly cover all possible situations in $\mathbb{R}^2$. Although in $\mathbb{R}^d$ for $d \geq 3$ there are obviously much more possibilities, the patterns in higher dimensions essentially resemble what is observed in these figures. In all cases we display the optimal rule (dashed black lines) and the decomposition generated by products of hyperplanes (solid lines), together with the training sample.

Figure 1: In the upper-left panel we observe that although the optimal rule is quadratic, it is approximately linear in the area where the two groups are closer, that is, in the relevant region for classification purposes. Consequently, the quadratic classifier can be replaced with a linear one without loss. As we will see, the procedure

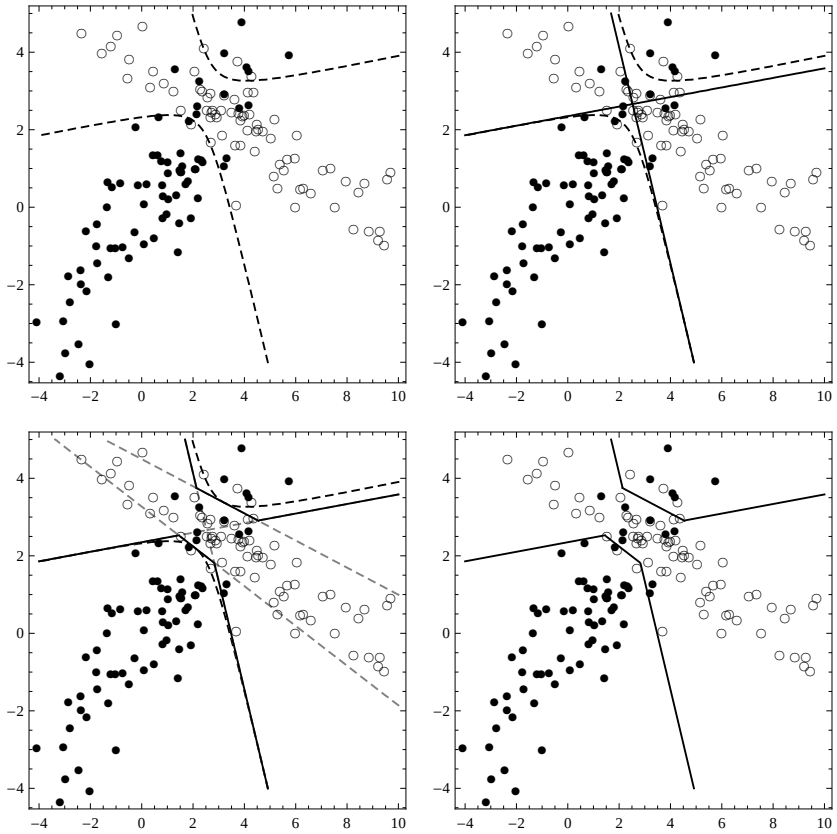

Fig. 2 The training sample and the optimal rule (upper-left panel); the approximation with one product of hyperplanes in solid lines (upper-right panel); the approximation with two products of hyperplanes (lower left panel); and the multidirectional classifier with two products (lower-right panel).

described in this paper allows us to identify this situation with ease, even in high dimension. In terms of our decomposition, this would be the case in which one product of hyperplanes provides a good approximation of the quadratic rule and the remainder is negligible (upper-right panel). Actually, in this example only one hyperplane of the product is meaningful for the classification problem. Note that a more precise approximation with two products of hyperplanes (lower panels) does not produce any significant improvement for the classification task.

Figure 2: In the upper panels we see that the optimal quadratic rule is far from being linear, but it can be accurately approximated by a product of two hyperplanes. In this example, we can replace the optimal quadratic rule with another one depending only on a few linear combinations of the original variables. Therefore, this decomposition is useful to determine linear combinations of the predictor variables that are important to discriminate. In the lower panels we also see that we can obtain a very precise approximation of the quadratic classifier by using a second product of hyperplanes. In

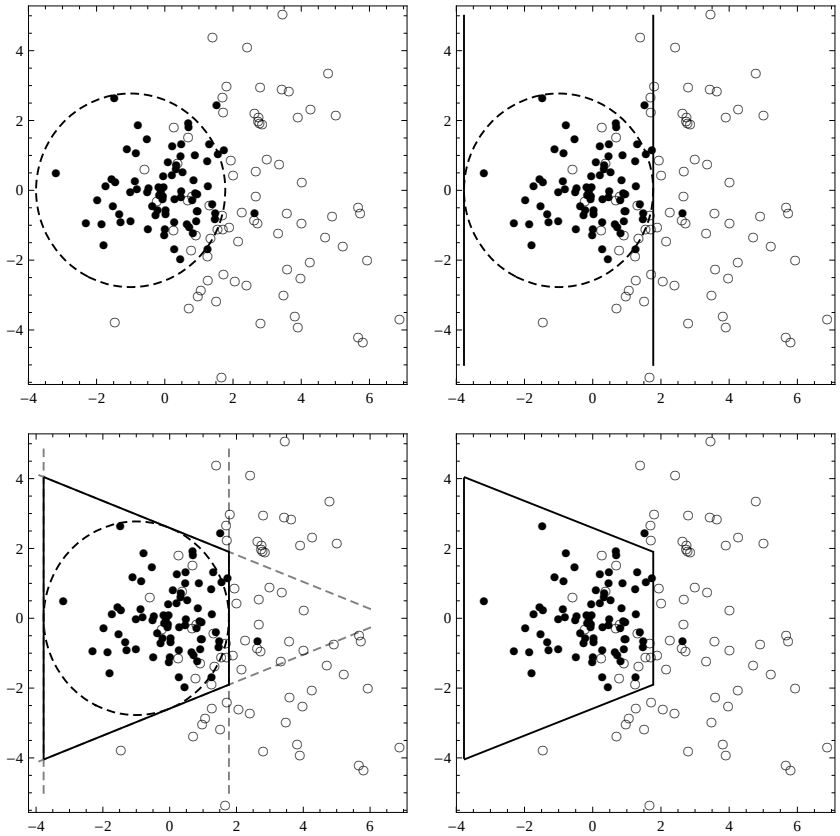

Fig. 3 The training sample and the optimal rule (upper-left panel); the approximation with one product of hyperplanes in solid lines (upper-right panel); the approximation with two products of hyperplanes (lower left panel); and the multidirectional classifier with two products (lower-right panel).

this way, a piecewise linear classifier with a clear interpretation can be constructed (lower-right panel).

Figure 3: This quadratic classifier (upper-left panel) is the most difficult to be approximated by hyperplanes. The first product's approximation (upper-right panel) separates well the two groups, but a second product (lower panels) is required to account for the shape and curvature of the quadratic rule. Observe that the hyperplanes approximate better the quadratic rule in the most important area for the classification, that is, between the two centers, whereas the approximation is cruder in less important parts.

The ideas in this paper are related to those in Huang et al (2012), where the authors look for two discriminative directions by solving an optimization problem subject to certain restrictions to deal with distinct subpopulations in the groups. Though the present approach is completely different, the methodology proposed in this paper can

also work under the presence of different clusters within the groups (see the example in Section 4).

We have only considered two classes as a multiclass classification problem can always be decomposed into various binary problems. Once the binary case is solved, we can use the majority voting principle (a combined decision from the classifiers) to predict the label of a new observation. In this direction, two simple but effective approaches are commonly applied in practice: the one-vs-all classification (see Rifkin and Klautau (2004)) and all-vs-all classification (see Park and Fürnkranz (2007)). The one-vs-all approach constructs one binary classifier for each class, where one training group consist in one class and the other is formed by the union of the rest of the classes. In contrast, in the all-vs-all approach $K(K-1)/2$ classifiers are computed, where K is the number of classes, separating each pair of classes.

The paper is organized as follows: In Section 2, we recall some important quadratic classifiers and we establish the basic results to decompose any quadratic rule in products of hyperplanes. Such a decomposition is used in Section 3 to derive the *multidirectional classifier*, a piecewise linear classifier constructed with the main linear components of a quadratic rule. We also note that we can arrange the products in terms of their ability to discriminate and obtain a classifier only including the most discriminative hyperplanes. In Section 4, we discuss the potential applications of this methodology to feature extraction by means of a toy example. The information contained in the multidirectional classifier can also be used to construct oblique classification trees. This is illustrated with a real data set in Section 5. In Section 6, we carry out a simulation study to assess the practical performance of our proposal with small samples. Finally, Section 7 collects some technical results regarding various theoretical aspects of the approximation.

2 Decomposition of a quadratic rule by products of hyperplanes

In this section, we enumerate some well-known quadratic classifiers and we specify the aforementioned decomposition of a quadratic classification rule by products of hyperplanes.

2.1 Quadratic classifiers

Throughout the paper, $\mathbf{x}$ denotes the predictor vector taking values in $\mathbb{R}^d$ and $G \in \{0,1\}$ is the categorical response variable representing the class memberships. The goal is to predict G using the knowledge of $\mathbf{x}$. In this work, we focus on quadratic classifiers, that is, the classification rule can be written in the form

$$\eta_Q(\mathbf{x}) = \mathbb{I}_{\{Q>0\}}(\mathbf{x}), \tag{1}$$

where $\mathbb{I}_R$ stands for the indicator function of the set R and Q is a *quadratic classification function* given by

$$Q(\mathbf{x}) = \mathbf{x}^\top \mathbf{A}\mathbf{x} + 2\mathbf{a}^\top \mathbf{x} + c, \qquad \mathbf{x} \in \mathbb{R}^d, \tag{2}$$

for a $d \times d$ symmetric matrix $\mathbf{A}$, a vector $\mathbf{a} \in \mathbb{R}^d$ and a scalar $c \in \mathbb{R}$. In other words, the curve (or surface) defined by the equation $Q(\mathbf{x}) = 0$ is the boundary of the classification rule. Consequently, this notation means that we assign $G = 1$ if and only if $\eta_Q(\mathbf{x}) = 1$, that is, whenever $Q(\mathbf{x}) > 0$. Note that we consider column vectors and $\mathbf{v}^\top$, $\mathbf{M}^\top$ are the transposes of the vector $\mathbf{v}$ and the matrix $\mathbf{M}$, respectively. Observe also that the assumption that $\mathbf{A}$ is symmetric does not represent any restriction on Q. Actually, if this is not the case, we can always substitute $\mathbf{A}$ for its symmetrization, the matrix $(\mathbf{A}+\mathbf{A}^\top)/2$, which defines exactly the same quadratic form.

The expectations and covariance matrices of $\mathbf{x}$ in both groups are denoted by μ_0, μ_1, Σ_0 and Σ_1, respectively. The covariance matrices can be assumed to be positive definite. In practice, the construction of the classifiers described in this section (which in some cases requires the estimation of μ_0, μ_1, Σ_0 and Σ_1) relies on the data available in the training sample, whose group memberships are known.

With this notation, it can be easily checked that some important and well-known classification rules can be expressed as η_Q, for different choices of the parameters of Q (the symmetric matrix $\mathbf{A}$, the vector $\mathbf{a}$ and the constant c).

Mahalanobis classifier: This rule assigns an observation $\mathbf{x}$ to the group $G = 0$ whenever $\mathbf{x}$ is closer (in terms of the Mahalanobis distance) to μ_0 than to μ_1. This example corresponds to η_Q, with

$$\mathbf{A} = \Sigma_0^{-1} - \Sigma_1^{-1}, \quad \mathbf{a} = \Sigma_1^{-1}\mu_1 - \Sigma_0^{-1}\mu_0 \quad \text{and} \quad c = \mu_0^\top \Sigma_0^{-1}\mu_0 - \mu_1^\top \Sigma_1^{-1}\mu_1. \quad (3)$$

Bayes classifier under normality: The Bayes (optimal) rule under normality coincides with η_Q with $\mathbf{A}$ and $\mathbf{a}$ as in (3), but

$$c = \mu_0^\top \Sigma_0^{-1}\mu_0 - \mu_1^\top \Sigma_1^{-1}\mu_1 + 2\log\left(\frac{\pi_1}{\pi_0}\right) + \log\left(\frac{|\Sigma_0|}{|\Sigma_1|}\right), \quad (4)$$

where $\pi_i = \mathbb{P}(G = i)$, $i = 0, 1$, are the prior probabilities of the groups and $|\mathbf{M}|$ stands for the determinant of $\mathbf{M}$. The constant c may also incorporate information regarding the misclassification costs: $c(i|1-i) =$ the cost of classifying an observation with membership $G = 1 - i$ as $G = i$ ($i = 0, 1$). This can be included in the classifier by adding the quantity $2\log(c(1|0)/c(0|1))$ to the constant c in (4).

Fisher classifier: The Fisher rule is a particular (and degenerate) case of the previous quadratic classifiers under homoscedasticity (i.e., when $\Sigma = \Sigma_0 = \Sigma_1$). It can be therefore expressed as η_Q, with

$$\mathbf{A} = \mathbf{0}, \quad \mathbf{a} = \Sigma^{-1}(\mu_1 - \mu_0) \quad \text{and} \quad c = (\mu_0 - \mu_1)^\top \Sigma^{-1}(\mu_0 + \mu_1). \quad (5)$$

(Here $\mathbf{0}$ stands for the matrix with all its entries equal to zero.) In this example, $\mathbf{a}$ in (5) is the well-known *(Fisher) linear discriminant vector*.

Support vector machines (SVM) with a quadratic kernel: A popular choice for the kernel of the SVM classifier is a polinomial of degree 2, that is, the kernel is $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^\top \mathbf{y} + 1)^2$ ($\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$). Therefore, ultimately the SVM classifier is η_Q, with

$$Q(\mathbf{x}) = \beta_0 + \sum_{\mathbf{x}_i \in S} \alpha_i (\mathbf{x}^\top \mathbf{x}_i + 1)^2, \qquad \mathbf{x} \in \mathbb{R}^d,$$

where S is the set of support vectors and the constants β_0 and α_i are the output of the SVM convex optimization problem. Accordingly, in this example the parameters of the quadratic classification function are

$$\mathbf{A} = \sum_{\mathbf{x}_i \in S} \alpha_i \mathbf{x}_i \mathbf{x}_i^\top, \quad \mathbf{a} = \sum_{\mathbf{x}_i \in S} \alpha_i \mathbf{x}_i \quad \text{and} \quad c = \beta_0 + \sum_{\mathbf{x}_i \in S} \alpha_i.$$

Quadratic classifiers for high dimensional data: There are various quadratic classifiers designed to deal with high dimensional data. For instance, *regularized discriminant analysis* (see Friedman (1989)) provides a quadratic classifier depending on two tuning parameters with a good behaviour in settings for which sample sizes are small and the number of measurement variables is large. Therefore, this regularized quadratic classifier can be computed even when the dimension of the predictor vector is greater than the number of observations. More recently, Fan et al (2015) propose a supervised dimension reduction method called QUADRO based on the optimization of the Rayleigh quotient for quadratic functions. By solving a convex optimization problem, they find the quadratic function that maximizes the Rayleigh quotient for elliptic models.

The ideas of this paper allow us to find the main linear components of any quadratic rule. When the considered classifier is no longer quadratic (such as SVM with a polynomial kernel of degree greater than 2 or gaussian kernel) we might still use the developed framework. This would require to approximate the non-quadratic classifier at different points by quadratic forms. We plan to further develop these ideas in a future research work.

2.2 The approximation of a quadratic classifier by products of hyperplanes

Let us consider the generic quadratic classifier η_Q in (1), with Q defined in (2). Our approach consists in decomposing Q as a product of two hyperplanes plus a remainder term. Hence, each decomposition only depends on two linear combinations of the original predictor variables, i.e., on two discriminative directions. Whenever the remainder is not important, such a decomposition yields a good approximation of Q in terms of a much simpler quadratic form; a product of two hyperplanes (see Figures 1 and 2). If that is not the case, we can use more products to obtain a more accurate approximation of the initial rule (see Figure 3).

Our starting point is the following basic proposition. Without loss of generality, in the sequel we assume that $c \neq 0$.

Proposition 1 *Let Q be as in (2) with $c \neq 0$. We have that*

$$cQ(\mathbf{x}) = (\mathbf{a}^\top \mathbf{x} + c)^2 - \sum_{i=1}^{r} \lambda_i (\mathbf{v}_i^\top \mathbf{x})^2, \qquad \mathbf{x} \in \mathbb{R}^d, \tag{6}$$

where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r$ and $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r$ are the eigenvalues and the associated unit eigenvectors of the matrix

$$\mathbf{B} = \mathbf{a}\mathbf{a}^\top - c\mathbf{A}. \tag{7}$$

Proof Some simple computations show the identity $cQ(\mathbf{x}) = (\mathbf{a}^\top \mathbf{x} + c)^2 - \mathbf{x}^\top \mathbf{B}\mathbf{x}$. As $\mathbf{A}$ is symmetric, $\mathbf{B}$ is also symmetric and therefore diagonalizable. Finally, (6) follows by the spectral decomposition of $\mathbf{B}$.

It turns out that the (spectral decomposition of the) matrix $\mathbf{B}$ in (7) plays a crucial role hereafter. In general, $\mathbf{B}$ may have negative eigenvalues. However, if we assume that there are points which are classified in different groups, that is, if there exist $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ such that $Q(\mathbf{x})Q(\mathbf{y}) < 0$, then $\mathbf{B}$ necessarily has at least one strictly positive eigenvalue. Note that if all its eigenvalues were negative or zero, from (6) we would have $cQ(\mathbf{x}) \geq 0$, for all $\mathbf{x} \in \mathbb{R}^d$.

We next analyze the decompositions of Q obtained using positive eigenvalues of $\mathbf{B}$. Negative eigenvalues are considered in Subsection 7.2. Let us denote by $r = \text{rank}(\mathbf{B})$ the rank of $\mathbf{B}$ and $\text{sg}(\mathbf{B}) = (r_+, r_-)$ its signature. Then r_+ and r_- are the number of positive and negative eigenvalues of $\mathbf{B}$, respectively. Note that $r_+ + r_- = r$ and, as argued before, in general $r_+ \geq 1$.

Corollary 1 *For each eigenvalue $\lambda_i > 0$ ($i = 1, \ldots, r_+$) of $\mathbf{B}$, we have that $Q = P_i + R_i$, where $cP_i = L_{i,1}L_{i,2}$ is the product of two hyperplanes given by*

$$L_{i,1}(\mathbf{x}) = (\mathbf{a} + \mathbf{b}_i)^\top \mathbf{x} + c, \quad L_{i,2}(\mathbf{x}) = (\mathbf{a} - \mathbf{b}_i)^\top \mathbf{x} + c, \qquad \mathbf{x} \in \mathbb{R}^d, \tag{8}$$

with $\mathbf{b}_i = \sqrt{\lambda_i}\mathbf{v}_i$, and R_i is a remainder term that can be expressed as

$$R_i(\mathbf{x}) = -\frac{1}{c} \sum_{i \neq \ell = 1}^{r} \lambda_\ell (\mathbf{v}_\ell^\top \mathbf{x})^2, \qquad \mathbf{x} \in \mathbb{R}^d. \tag{9}$$

Proof From (6), we directly obtain

$$\begin{aligned} cQ(\mathbf{x}) &= (\mathbf{a}^\top \mathbf{x} + c)^2 - \lambda_i (\mathbf{v}_i^\top \mathbf{x})^2 - \sum_{i \neq \ell = 1}^{r} \lambda_\ell (\mathbf{v}_\ell^\top \mathbf{x})^2 \\ &= (\mathbf{a}^\top \mathbf{x} + c)^2 - (\mathbf{b}_i^\top \mathbf{x})^2 - \sum_{i \neq \ell = 1}^{r} \lambda_\ell (\mathbf{v}_\ell^\top \mathbf{x})^2 \\ &= L_{i,1}(\mathbf{x}) L_{i,2}(\mathbf{x}) + cR_i(\mathbf{x}). \end{aligned}$$

In case that $\mathbf{B}$ had only one large positive eigenvalue, λ_1, whereas the rest of the eigenvalues are negligible, we would obtain $R_1(\mathbf{x}) \approx 0$, for all "important" $\mathbf{x} \in \mathbb{R}^d$, and therefore

$$Q(\mathbf{x}) \approx P_1(\mathbf{x}) = \frac{1}{c} \left[(\mathbf{a} + \mathbf{b}_1)^\top \mathbf{x} + c \right] \left[(\mathbf{a} - \mathbf{b}_1)^\top \mathbf{x} + c \right].$$

This is the situation underlying the examples in Figures 1 and 2 and explains why in both cases a product of two hyperplanes yields a good approximation of the quadratic rule. In fact (see Subsection 7.1), Q is a product of two hyperplanes if and only if $\mathbf{B}$ has exactly one strictly positive eigenvalue. For instance, this holds for the Bayes rule under normality and homoscedasticity. In such a special case, $\mathbf{B} = \mathbf{a}\mathbf{a}^\top$ and the quadratic rule obviously reduces to the Fisher linear one, which is a well-known fact.

However, in other occasions, particulary when d is large, the linear structures behind a quadratic rule may be hidden. The spectral analysis of $\mathbf{B}$ provides an easy way to uncover them, as shown in the examples of the introduction.

It is natural to ask about the points for which the product P_i in Corollary 1 provides a good approximation of Q. From (9), we see that $R_i(\mathbf{x}) = 0$, for all $\mathbf{x} \in \text{Span}\{\mathbf{v}_1, \ldots, \mathbf{v}_{i-1}, \mathbf{v}_{i+1}, \ldots, \mathbf{v}_r\}^{\perp}$ (the orthogonal complement of the vector subspace generated by $\{\mathbf{v}_1, \ldots, \mathbf{v}_{i-1}, \mathbf{v}_{i+1}, \ldots, \mathbf{v}_r\}$), and hence Q and P_i coincide on this subspace. In particular, we always have that $Q(\alpha \mathbf{v}_i) = P_i(\alpha \mathbf{v}_i)$, for all $\alpha \in \mathbb{R}$, so the approximation of P_i is indeed exact along the direction of the eigenvector $\mathbf{v}_i$. This implies that $Q(\mathbf{0}) = P_i(\mathbf{0})$ (with $\mathbf{0} = (0, \ldots, 0)^{\top} \in \mathbb{R}^d$) and it is easy to check that the gradients of P_i and Q are also equal at this point, that is, $\nabla Q(\mathbf{0}) = \nabla P_i(\mathbf{0})$. Therefore, P_i provides a good approximation of Q around the origin. For the supervised classification problem it is therefore convenient to center the sample points so that one of the estimated group means coincides with the origin, or make an appropriate change of variable in Q (see Section 3 for details).

The hyperplanes defined by the equations $L_{i,1}(\mathbf{x}) = 0$ and $L_{i,2}(\mathbf{x}) = 0$ can also be viewed as linear approximations of the classification boundary $Q(\mathbf{x}) = 0$. Indeed, these two hyperplanes are tangent to $Q(\mathbf{x}) = 0$ and the tangent points can be explicitly computed (see Proposition 2 in Subsection 7.3). Therefore, they provide a piecewise linear approximation of the boundary of the classification rule generated by Q.

We remark that, as the vectors $\mathbf{b}_i = \sqrt{\lambda_i}\mathbf{v}_i$ (for $i = 1, \ldots, r_+$) are orthogonal, each product P_i in Corollary 1 reflects a different direction to "look at" Q. For this reason, the second product approximation in Figures 2 and 3 nicely completes and complements the work done by the first product. Further, we also note that $\mathbf{a}$ in (3) can be seen as a version of the Fisher linear discriminant vector (see (5)) because they coincide under homoscedasticity. The orthogonal $\mathbf{b}_i$-s in the normal vectors of the hyperplanes in (8) are hence (uncorrelated) corrections of $\mathbf{a}$ to take into account heteroscedasticity.

3 The multidirectional classifier

Given the quadratic classification function Q in (2), in this section we describe the construction of the associated multidirectional classifier (plotted for various examples in the lower-right panels of Figures 1–3). We also show how we can use this piecewise linear rule to reduce the dimension of the classification problem in some situations.

3.1 The choice of the approximating point

As argued in Subsection 2.2, each product P_i in Corollary 1 approximates the quadratic classification function Q in (2) around the origin. This fact does not represent any drawback because, at least initially, the approximation of Q can be carried out around any preselected point $\mathbf{x}_0 \in \mathbb{R}^d$. It is enough to rewrite Q as

$$Q(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_0)^{\top}\mathbf{A}(\mathbf{x} - \mathbf{x}_0) + 2\,\mathbf{a}_{\mathbf{x}_0}^{\top}(\mathbf{x} - \mathbf{x}_0) + c_{\mathbf{x}_0}, \qquad \mathbf{x} \in \mathbb{R}^d,$$

where $\mathbf{a}_{\mathbf{x}_0} = \mathbf{a} + \mathbf{A}\mathbf{x}_0$ and $c_{\mathbf{x}_0} = Q(\mathbf{x}_0)$ (and make the change of variable $\mathbf{y} = \mathbf{x} - \mathbf{x}_0$). The results of the Section 2 are valid by just substituting $\mathbf{a}$ and c by $\mathbf{a}_{\mathbf{x}_0}$ and $c_{\mathbf{x}_0}$, respectively.

However, for classification purposes a natural and sensible choice for $\mathbf{x}_0$ is one of the centers of the groups, that is, μ_0 or μ_1. In such cases, the corresponding matrix $\mathbf{B}$ in (7) is respectively transformed into

$$\mathbf{B}_i = \mathbf{a}_i \mathbf{a}_i^\top - c_i \mathbf{A}, \qquad i = 0, 1, \tag{10}$$

where $\mathbf{a}_i = \mathbf{a} + \mathbf{A}\mu_i$ and $c_i = Q(\mu_i)$. In other words, if we approximate Q around μ_0 (respectively, μ_1), we have to analyze the matrix $\mathbf{B}_0$ (respectively, $\mathbf{B}_1$).

To clarify the differences between choosing μ_0 or μ_1 to approximate Q, we briefly discuss an example in $\mathbb{R}^3$. We consider a training sample coming from two normal distributions with parameters $\mu_0 = (0,0,0)^\top$, $\Sigma_0 = \mathrm{diag}(2,2,0.5)$ (the diagonal matrix with $(2,2,0.5)^\top$ as main diagonal) and $\mu_1 = (3,0,0)^\top$, $\Sigma_1 = \mathrm{diag}(1,1,1)$, respectively. The Bayes rule in this example is described in Subsection 2.1 and the matrix $\mathbf{B}_0$ has eigenvalues $\lambda_1 \approx 8.31$, $\lambda_2 \approx 4.85$ and $\lambda_3 \approx -4.15$. From Corollary 1, we therefore have two products of hyperplanes that approximate the optimal rule (around μ_0): P_1 and P_2.

In Figure 4, we have plotted the training sample (cubes correspond to $G = 0$ and spheres to $G = 1$) and some of the approximating products of the Bayes rule. We first see the quadratic boundary of the Bayes rule (upper-left panel) and that the product P_2 separates well the two groups (upper-right panel). The combination of the products generated by the positive eigenvalues of $\mathbf{B}$, P_1 and P_2, provides a better approximation of the Bayes rule (lower-left panel). The approximating hyperplanes generate a piecewise linear surface separating the groups and wrapping up μ_0, the point around which we approximate the Bayes rule (lower-right panel).

If we want to approximate Q around μ_1, we have to compute the eigenvalues of the matrix $\mathbf{B}_1$ in (10). These eigenvalues are $\lambda_1^{(1)} \approx 4.85$, $\lambda_2^{(1)} \approx 2.60$ and $\lambda_3^{(1)} \approx -5.19$. Therefore, we again have two approximating products, say $P_1^{(1)}$ and $P_2^{(1)}$. In Figure 5, we have displayed the result of the approximation of Q around μ_1. In this case, the combination of $P_1^{(1)}$ and $P_2^{(1)}$ provides a piecewise linear surface separating the groups and wrapping up μ_1.

From Figures 4-5, we observe that the result of the procedure depends on the approximating point. As we only consider the positive eigenvalues of the matrices $\mathbf{B}_0$ and $\mathbf{B}_1$, the resulting products collect the convex structure of the underlying quadratic rule around the approximating point. As in the considered example the quadratic boundary of the Bayes rule is not convex, we clearly observe the difference in the generated classifiers (lower-right panel in Figure 4 and right panel in Figure 5).

3.2 Multidirectional classification

Let us assume that we have a training sample at our disposal and we want to construct the multidirectional classifier associated to a certain quadratic rule Q.

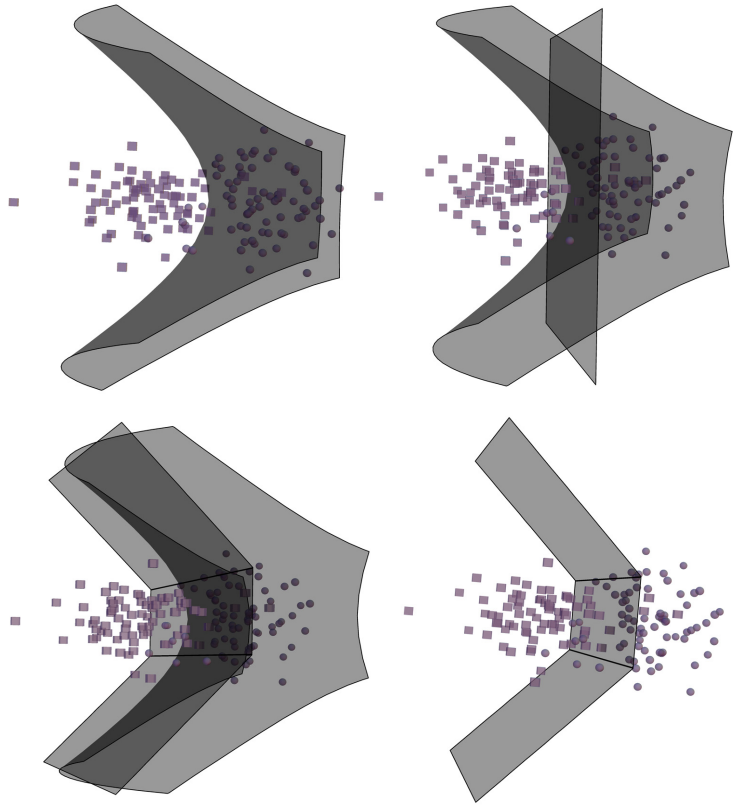

Fig. 4 The training sample and the Bayes rule (upper-left panel); approximation by P_2 (upper-middle panel); approximation by P_1 and P_2 (lower-left panel); and the multidirectional classifier corresponding to P_1 and P_2 (lower-right panel).

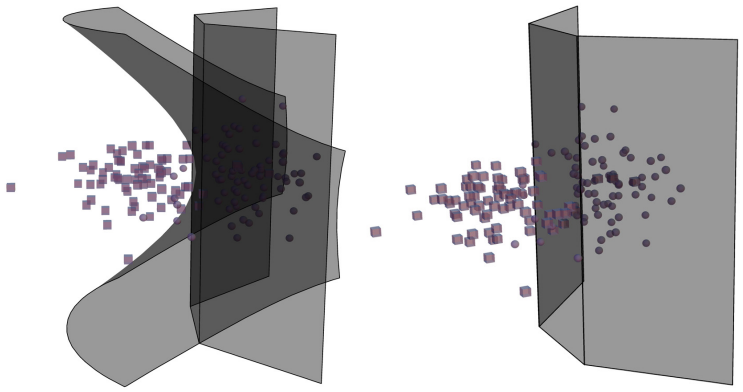

Fig. 5 The approximation around μ_1 generated by $P_1^{(1)}$ and $P_2^{(1)}$ (left panel); and the associated multidirectional classifier (right panel).

First, let us assume that we select μ_0 as the approximation point. With the training sample we can estimate Q and the matrix $\mathbf{B}_0$ in (10) to carry out its spectral analysis. In practice, the rank of the estimated matrix of $\mathbf{B}_0$ will be equal to d (the number of predictor variables), with probability 1. This means that if r_+ eigenvalues are strictly positive, applying Corollary 1, we obtain r_+ approximating products of hyperplanes. For each approximating product P of the (estimated) classification function Q, we can substitute the quadratic rule η_Q in (1) by the simpler one η_P (the classifier induced by P). However, it remains to show how we can merge the information of all products to obtain a combined classifier, that is, a unique piecewise linear classification rule.

As explained in Subsection 3.1, the combination of products generated by positive eigenvalues of $\mathbf{B}_0$ produces a convex piecewise linear classification boundary wrapping up μ_0. Therefore, with the products $P_1,\ldots,P_{r_+}$ described in Corollary 1 we classify a new observation $\mathbf{x}\in\mathbb{R}^d$ in the group $G=0$ if and only if $\eta_{P_1}(\mathbf{x})=\cdots=\eta_{P_{r_+}}(\mathbf{x})=0$, in other words, if $P_1(\mathbf{x})\leq 0,\ldots,P_{r_+}(\mathbf{x})\leq 0$, simultaneously. Alternatively, $\mathbf{x}$ is assigned to $G=1$ if $\eta_{P_i}(\mathbf{x})=1$, for some $i\in\{1,\ldots,r_+\}$, that is, if $P_i(\mathbf{x})>0$, for some $i\in\{1,\ldots,r_+\}$. The *multidirectional classifier (associated with Q and) centered at μ_0* is hence the rule defined by

$$\eta^0_{\mathrm{MD}}=\max\{\eta_{P_1},\ldots,\eta_{P_{r_+}}\}=1-\prod_{i=1}^{r_+}(1-\eta_{P_i}). \tag{11}$$

Observe that this classifier can also be expressed as the indicator of a set; $\eta^0_{\mathrm{MD}}=\mathbb{I}_{A_0}$, where $A_0=\cup_{i=1}^{r_+}\{P_i>0\}$.

Analogously, if we approximate Q around μ_1, we have to consider the matrix $\mathbf{B}_1$ in (10) and the corresponding *multidirectional classifier (associated with Q and) centered at μ_1* is

$$\eta^1_{\mathrm{MD}}=\min\{\eta_{P^1_1},\ldots,\eta_{P^1_{s_+}}\}=\prod_{i=1}^{s_+}\eta_{P^1_i}, \tag{12}$$

where s_+ is the number of positive eigenvalues of $\mathbf{B}_1$ and $P^1_1,\ldots,P^1_{s_+}$ are the resulting products of the approximation of Q around μ_1 as given in Corollary 1. In this case, we have that $\eta^1_{\mathrm{MD}}=\mathbb{I}_{A_1}$, where $A_1=\cap_{i=1}^{s_+}\{P^1_i>0\}$.

We remark that, in general, $r_+\neq s_+$ and $\eta^0_{\mathrm{MD}}\neq\eta^1_{\mathrm{MD}}$. For instance, in the example discussed in Subsection 3.1, η^0_{MD} is plotted in the lower-right panel of Figure 4, whereas η^1_{MD} is displayed in the right panel of Figure 5. We therefore propose to define the *multidirectional classifier (associated with Q)* as the rule $\eta_{\mathrm{MD}}=\eta^0_{\mathrm{MD}}$ whenever $r_+>s_+$ and $\eta_{\mathrm{MD}}=\eta^1_{\mathrm{MD}}$ if $r_+<s_+$. When $r_+=s_+$, we can use the classifier for which the sum of positive eigenvalues of $\mathbf{B}$ is greater. For instance, according to this definition, in the example of Subsection 3.1 the classifier η_{MD} is plotted in the lower-right panel of Figure 4.

As discussed in Subsection 7.4, in practice we usually have that the following inequality holds: $\lceil d/2\rceil\leq\max\{r_+,s_+\}\leq d$, where $\lceil\cdot\rceil$ is the ceiling function. Therefore, the multidirectional classifier includes at least d (and at most $2d$) hyperplanes.

3.3 Reduction of the dimension through the multidirectional classifier

Let us assume that $\eta_{\mathrm{MD}} = \eta^0_{\mathrm{MD}}$ in (11). (The case $\eta_{\mathrm{MD}} = \eta^1_{\mathrm{MD}}$ in (12) is completely analogous.) Some of the r_+ products included in η_{MD} will contain useful information to classify new observations and others will not. To eliminate the noninformative combinations, the first task is to rank them according to their relevance for the classification problem. For that purpose we can simply use the empirical classification error of the n observations in the training sample or any other estimation of the error obtained by cross-validation. The *empirical error rate* of a classifier η is denoted by

$$\mathrm{R}(\eta) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}_{\{\eta(\mathbf{x}_i)\neq G_i\}},$$

where G_i is the true group of the observation $\mathbf{x}_i$ of the training sample. It is well-known that $\mathrm{R}(\eta)$ is too optimistic to estimate the probability of misclassification of η. However, that is not an important issue here because we are only interested in the relative performance of the products. We denote the r_+ approximating products by $P_{(1)},\ldots,P_{(r_+)}$ after ranking them according to R or any other sensible estimation of the error. This means that if $\mathrm{R}_{(i)} = \mathrm{R}(\eta_{P_{(i)}})$, then $\mathrm{R}_{(1)} \leq \cdots \leq \mathrm{R}_{(r_+)}$.

For each $k = 1,\ldots,r_+$, we can define the *multidirectional classifier with the main k products* as the rule

$$\eta_{\mathrm{MD}(k)} = \max\{\eta_{P_{(1)}},\ldots,\eta_{P_{(k)}}\} = 1 - \prod_{i=1}^{k}(1-\eta_{P_{(i)}}). \tag{13}$$

In practice, to select an appropriate value for k, we recommend to use an elbow method similar to the one employed in a standard principal component analysis. We can depict the number of products (ordered according to any criterion) on the x-axis and the estimated percentage of correct classification with that number of products on the y-axis. We can choose the minimum number of products for which there is no substantial improvement in the error rate classification. When $\eta_{\mathrm{MD}(k)}$ achieves an acceptable misclassification error for a relatively small value of k, we can discard the rest of the products, $P_{(k+1)},\ldots,P_{(r_+)}$.

We therefore obtain $2k$ linear combinations of the original variables in $\eta_{\mathrm{MD}(k)}$ which are important to discriminate. If the number of relevant linear combinations is much smaller than the number of original variables, the dimensionality of the classification problem can be substantially reduced. Using regularization techniques, the new rules can be constructed even when the number of variables is greater than the number of observations. In Section 4 we discuss in depth these ideas with an example.

4 Feature extraction and dimension reduction

In classification with high-dimensional data it is common to face the problem of dealing with noisy variables. That is, within the predictor vector many variables contain no information regarding the class membership. In such a case, the misclassification error may increase substantially because the noninformative variables are incorporated

into the models adding extra variability without reducing the error rate. This is specially noticeable when the dimension of the data is high and the estimation of the parameters is more delicate. Regarding this problem, the multidirectional classifier defined in Section 3 has a desirable property: Let us assume that only the first k variables (out of the d) of the predictor vector are informative. In the population model and for the quadratic rules in (3)-(4), this is the case in which the parameters are given by

$$\mu_i = \left[\begin{array}{c} \nu_i \\ \hline \nu \end{array}\right] \quad \text{and} \quad \Sigma_i = \left[\begin{array}{c|c} \Lambda_i & \mathbf{0} \\ \hline \mathbf{0} & \Lambda \end{array}\right] \qquad (i = 0, 1),$$

where $\nu_i \in \mathbb{R}^k$, $\nu \in \mathbb{R}^{d-k}$, Λ_i is a $k \times k$ real matrix, Λ is a $(d-k) \times (d-k)$ real matrix and $\mathbf{0}$ is the zero matrix. In this situation, it is straightforward to check that the normal vectors of the hyperplanes appearing in equation (8) of Corollary 1 can be expressed as $\mathbf{a} \pm \mathbf{b}_i = [\tilde{\mathbf{a}} \pm \tilde{\mathbf{b}}_i | \mathbf{0}]^\top$ $(i = 1, \ldots, r_+)$, where $\tilde{\mathbf{a}}$, $\tilde{\mathbf{b}}_i \in \mathbb{R}^k$. Therefore, in this case noisy variables have zero weight in all the normal vectors of the hyperplanes generated by the multidirectional classifier.

In this section, we present a toy example to illustrate how the linear combinations of the predictor variables corresponding to the multidirectional classifier can be used to detect the variables with more ability to discriminate. Further, this classifier can also be used to reduce the dimension of the data by removing those combinations with no relevance. Note that the results are potentially useful even when the dimension of the predictor vector is high compared with the sample sizes of the training sample.

4.1 Toy example in low dimension ($n_0 = n_1 = 100$, $d = 2$)

We consider a bidimensional training sample with two groups of size 100. The first one has been generated from a bivariate normal distribution with mean vector $(0,3)^\top$ and covariance the diagonal matrix $\mathrm{diag}(1,4)$. The second one consists on two clusters of size 50, each of them coming from bivariate normal distributions centered at $(-3,0)^\top$ and $(3,0)^\top$, respectively, and with the same covariance matrix, $\mathrm{diag}(1,4)$. In Figure 6, we display the quadratic support vector machine rule (see Subsection 2.1) and the underlying multidirectional classifier (see Section 3). Finally, for the sake of comparison, we have also plotted the Fisher linear rule.

Observe that the quadratic rule, despite not being optimal for these particular populations, separates well the two groups in the training sample and copes satisfactorily with the presence of two clusters in the second group. Also, the multidirectional classifier provides a good approximation of the quadratic rule and finds the directions that separate the groups best.

In order to compare the efficiency of these and other rules quantitatively, we have generated independent test samples of size 1000 from the initial distributions and we have classified them using the following rules: Fisher (LDA), linear SVM (LSVM), quadratic Bayes rule under normality (QDA), quadratic SVM (QSVM), Friedman's regularized classifier (see Friedman (1989)) which is a quadratic rule depending on two tuning parameters with a good behaviour for high-dimensional data (QRDA), and a tree classifier (TREE). To carry out the computations we have called the function `train` of the `R` package `caret` (see R Core Team (2016) and Kuhn (2008)) using

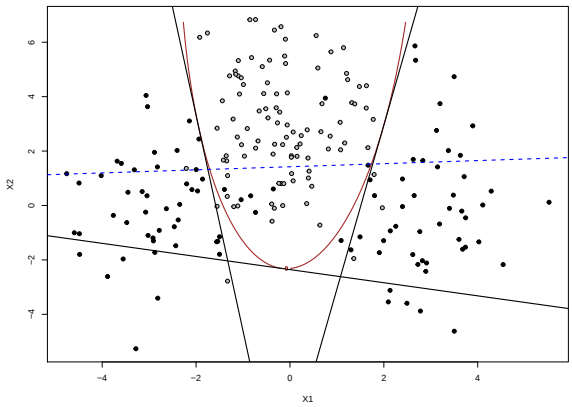

Fig. 6 The training sample ($G = 0$ in white and $G = 1$ in black), Fisher linear classifier (dashed line), quadratic SVM rule (solid curve) and the corresponding multidirectional classifier based on the best two products (solid lines).

standard 10-fold cross validation to tune the models when necessary. We have also applied the results introduced in the previous sections to the quadratic rules to obtain, in the obvious way, the corresponding *mutidirectional discriminant analysis* based on one product of hyperplanes (the best according to misclassifications rate) denoted by QDA-M, QSVM-M and QRDA-M. The results are displayed in the first row of Table 1.

Observe that both QDA-M and QSVM-M are simple piecewise linear classifiers which improve significantly the behavior of the Fisher linear one, and whose performances are also slighly better than those of their corresponding quadratic rules, which are more difficult to interpret. Theses classifiers depend only on two linear combinations of the original variables.

4.2 Adding high-dimensional noisy data

Here we show how to apply the multidirectional classifier when the sample sizes of the training sample are relatively small in comparison with the dimension of the explanatory vector. In such a case, the covariance matrices estimates are unstable and the performance of the quadratic classifiers that rely on these estimations is usually poor.

We continue the example of Subsection 4.1 by adding to the training sample in Figure 6 other $d-2$ independent standard normal variables, playing the role of non-informative noise, for several values of d. Then, we end up with a classification problem for which the sample size (100 for both groups in the training sample) with d up to 200, but only the first two variables contain information. For certain values of d,

d	LDA	LSVM	QDA	QDA-M	QRDA	QRDA-M	QSVM	QSVM-M	TREE
2	0.24	0.24	0.09	0.07	0.20	0.23	**0.06**	**0.06**	0.20
25	0.24	0.27	0.24	0.26	0.23	0.25	0.13	**0.09**	**0.09**
50	0.30	0.31	-	-	0.30	0.30	0.15	**0.10**	0.12
100	0.36	0.36	-	-	0.34	0.34	0.16	**0.11**	0.17
150	-	0.36	-	-	0.35	0.35	0.19	**0.11**	0.19
200	-	0.36	-	-	0.41	0.40	0.20	**0.12**	0.20

Table 1 Classification errors for noisy data with increasing dimension and nine different classifiers. Missing values are due to the fact that LDA and QDA are not feasible for certain values of the dimension. The lowest misclassification rates in bold.

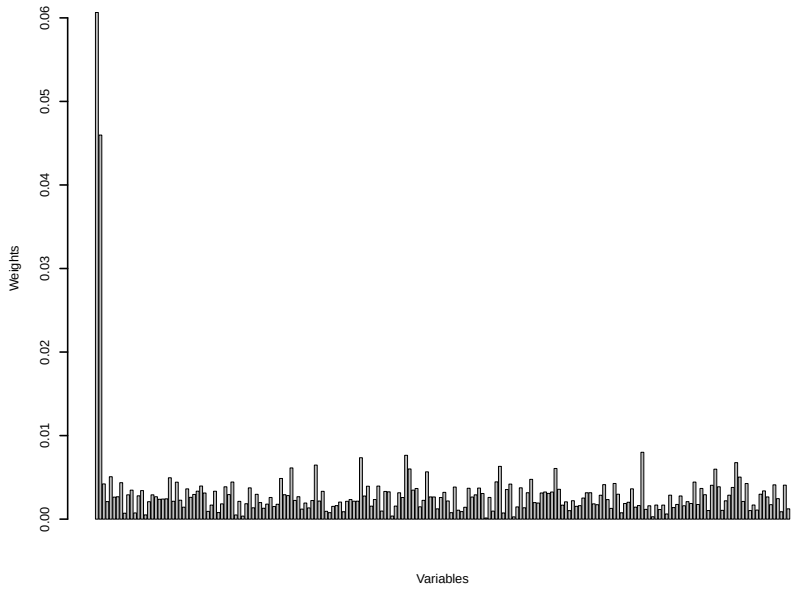

Fig. 7 Average absolute weights given to the 200 variables by the first product of hyperplanes corresponding to QSVM-M. The two first variables are clearly identified as the most informative ones.

the quadratic classifiers defined by the parameters in (3)–(4) cannot be even computed because the estimated covariance matrices are singular. The results are displayed in Table 1.

Among the quadratic rules the best one is QSVM, but notice that in this example QSVM-M achieves an even better error rate using just two linear combinations of the original variables. Figure 7 corresponds to the average weights (in absolute value) given to each variable by the two discriminant directions of the first product when $d = 200$. From this picture, we identify the first two variables as the relevant ones among the 200 variables of the predictor vector.

In Figure 8 we see the projections of the training sample over the plane spanned by the two directions corresponding to QSVM-M. Although there is a large number

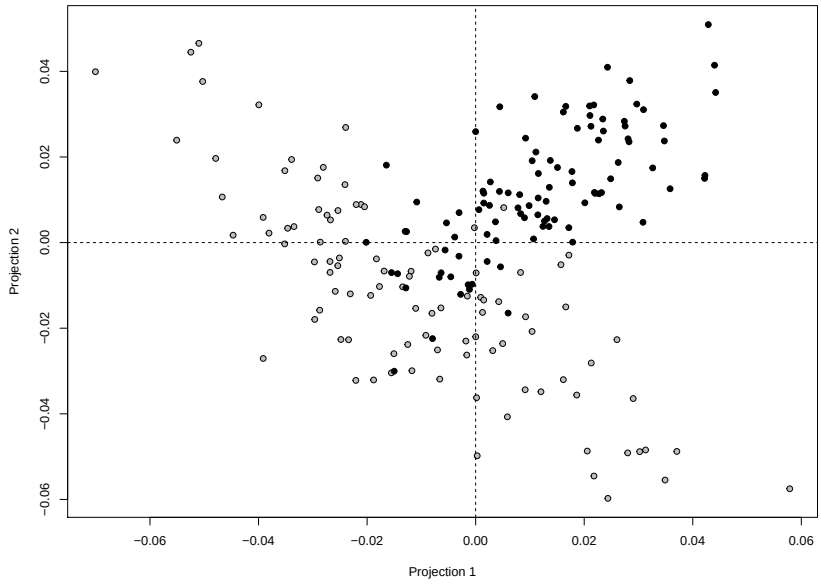

Fig. 8 Projections of 200-dimensional training data on the two directions corresponding to the best product of hyperplanes (QSVM-M).

of noisy variables, the two samples can be neatly distinguished using only these two projections, what yields a substantial reduction of the dimension of the supervised classification problem. The multidirectional rule classifies a new observation as black whenever both projections are positive or negative. This rule achieves an error rate (for independent test data) of 0.12 using only two linear combinations of the variables.

4.3 Conclusions

We learn in the two dimensional case that quadratic classifiers perform well in comparison with the linear ones (LDA and LSVM). However, the combinations of the two variables obtained with multidirectional classifiers provide a better result. This is somehow natural because one group has clusters and it has already been observed in Huang et al (2012) that products of two hyperplanes might produce good results in this situation.

When the dimension of the classification problem increases, the multidirectional classifier (based on QSVM) still manages to find the best two directions to look at the data. Further, we can neatly distinguish the first two variables as the important ones to discriminate.

5 An application to oblique classification trees

In this section we illustrate with a real data set how we can use the output of the multidirectional classifier to construct oblique classification trees.

5.1 Oblique classification trees

Classification and regression trees (CART) are generally considered to be an easily interpretable classification tool. However, its interpretability diminishes as they become larger (with more splits). Oblique classification trees (OCT), that is, trees in which splits are based on the values of linear combinations of the original variables, may be useful to obtain more compact classification rules than those obtained with classical trees, which only use axis-parallel splits. Moreover, in OCT the first/upper combinations in the splits can be seen as indicators or markers which are relevant for the classification problem.

Several criteria have been proposed in order to build OCT (see for instance Truong (2009) and references therein for a survey on this subject). Roughly speaking, an *oblique split dictionary* is a set of linear combinations that divide the data in two sets. There are mainly two different ways to grow OCT. A *direct search approach* looks for the best oblique split in the whole dictionary with the intention of stopping early. Obviously, finding good oblique splits at each step could be a computationally difficult task. By contrast, *indirect search approaches* restrict the attention to a certain subset of the oblique split dictionary. In general, such a subset is usually based on some linear classifiers that provide good directions to look at the data.

The multidirectional classifier introduced in Section 3 provides a set of linear combinations of the predictors that can be used as a suitable subset of the oblique split dictionary to grow the OCT. To be more precise, we can project our data in the directions given by the normal vectors of the hyperplanes included in the multidirectional classifier and afterwards use the CART algorithm to obtain an OCT. In the next subsection we illustrate this idea with the analysis of a real data set.

5.2 Breast Cancer Wisconsin Data

Data were gathered to assess a method to diagnose breast cancer. The method combines fine needle aspiration with computer imaging. The needle is used to obtain a fluid sample from a patient's breast lump. Then, the sample is stained in order to analyze the nuclei of the cells. Images of the nuclei are processed with a computer so that the exact boundaries of the nuclei are determined. Ten features are computed reflecting different aspects of the size and shape of the nuclei. Each sample consists of several tens of cells so that the mean, standard deviation and the largest value of the ten features are registered for each sample. This procedure yields a total of 30 variables. For instance, variable 1 is the radius average, variable 11 is the radius standard deviation and variable 21 is the largest radius. (A complete description of the features can be found at the UCI Machine Learning Repository, Bache and Lichman (2013).) The data

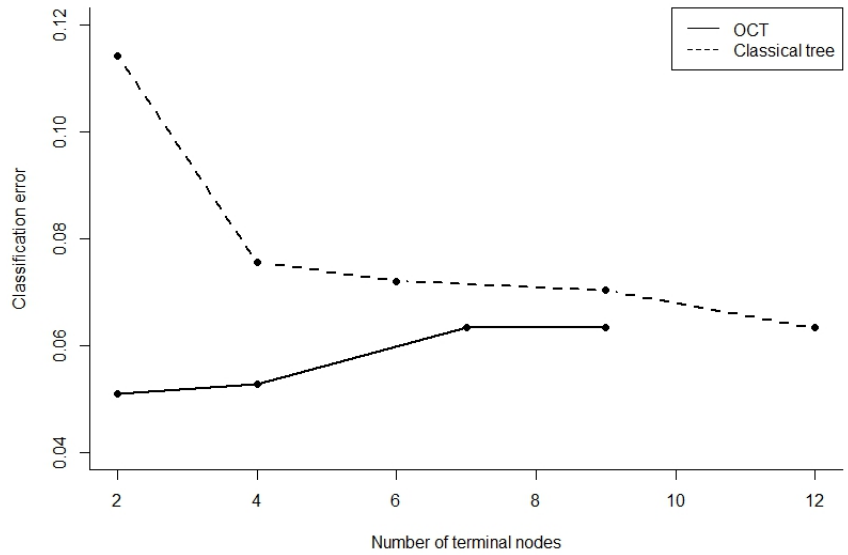

Fig. 9 Classification error rates (estimated by 10-fold cross validation) for the classical tree and the oblique tree generated by the multidirectional classifier, as a function of the number of terminal nodes.

set consists of 569 observations (samples), of which 212 were diagnosed as malignant ($G = 1$) and 357 as benign ($G = 0$).

First, we have built a classical tree using the `R` package `tree` (see R Core Team (2016) and Ripley (2014)). After growing an initial tree, we have applied a cost-complexity pruning in order to obtain a sequence of optimal subtrees as a function of the penalization parameter. We have used 10-fold cross validation to estimate the classification error rate. The dashed line in Figure 9 represents the classification error rates for the optimal subtrees as a function of the complexity (number of terminal nodes).

The final optimal tree (taking into account both the classification error rate and the complexity penalization on the number of terminal nodes) is displayed in Figure 10. Six variables were actually used to build this tree, with 9 terminal nodes. The maximum perimeter seems to be an important variable as it is used to define the first split. Generally speaking, large values of the variables have worse prognosis regarding the nature of breast lumps.

Next, we have estimated the quadratic Bayes rule under normality and computed the multidirectional classifier. This procedure yields 10 products of hyperplanes and 20 potentially interesting directions to look at the data. We have projected the training sample on these directions and built an initial OCT using the projections. Then, we

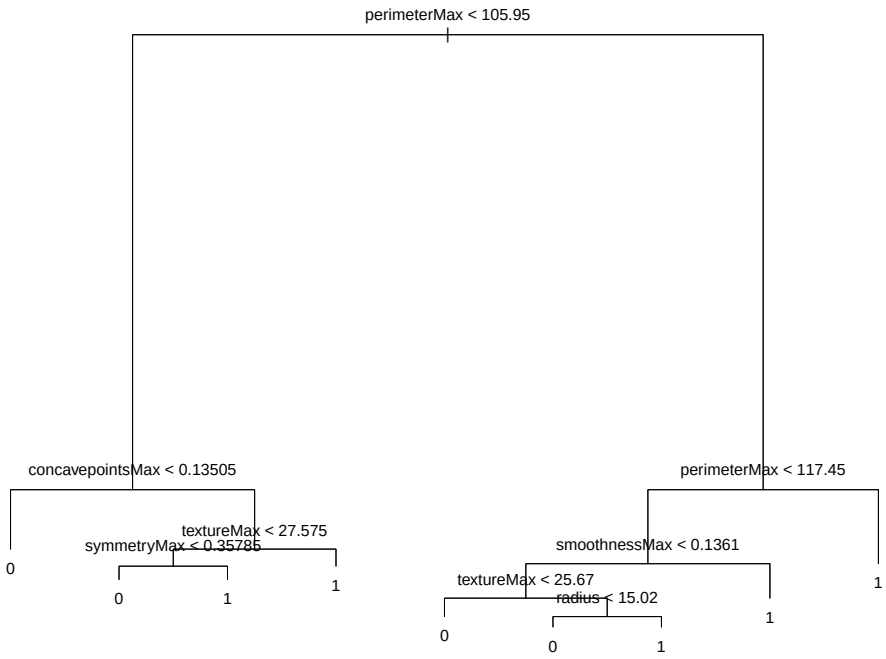

Fig. 10 A classical tree for breast cancer Wisconsin data.

have pruned this tree in the same way as we did when using the original variables. The solid line in Figure 9 represents the classification error rates for the sequence of optimal subtrees based on projections. Observe that, for all the levels of complexity, the error rates are always lower than those corresponding to the trees built with the individual variables.

The optimal OCT based on projections has only two terminal nodes, meaning that only one linear combination of the variables is used in the classification. The estimated classification error rate of this rule is about 5% whereas the estimated classification error rate of the tree in Figure 10 is about 7%.

6 Simulation study

In this section we report the results of a simulation study to discuss the difference between a quadratic rule and its approximation in terms of a few products of hyperplanes. There is a large number of possible scenarios (relative position of means, scale differences between the populations, shape of the cloud points, different clusters within the

classes, an so on) that could be considered. In order to carry out unbiased experiments that cover quite different settings, we have decided to follow very closely the study designed by Friedman (1989). As all models in Friedman (1989) are Gaussian, we have also incorporated another model with two clusters in one of the classes.

Each experiment consists of 100 replications of the following procedure: first, two training samples of size 100 are drawn, one from each class of the model (see below the list of models). Using these samples, a set of 13 different classification rules are trained. Then, independent test data sets of size 100 from the same models are generated and classified using all the rules, thereby obtaining an estimate of their probabilities of misclassification. Reported tables contain the averages of the misclassification rates and the corresponding standard errors (over the 100 replications).

Whenever a classifier is quadratic, we have computed the multidirectional rules based on one and two products of hyperplanes as defined in Equation (13). This is the complete list of classification rules:

LDA: Linear discriminant analysis (Fisher rule).
LSVM: Linear support vector machine.
QDA: Quadratic discriminant analysis, the Bayes rule for heteroskedastic Gaussian classes.
QDA-M1 and **QDA-M2**: Multidirectional rules based on QDA with one and two products of hyperplanes, respectively.
RDA: Regularized discriminant analysis, as described by Friedman (1989).
RDA-M1 and **RDA-M2**: Multidirectional rules based on RDA with one and two products of hyperplanes, respectively.
QSVM: Support vector machine with a quadratic kernel.
QSVM-M1 and **QSVM-M2**: Multidirectional rules based on QSVM with one and two products of hyperplanes, respectively.
GSVM: Support vector machine with a Gaussian kernel.
TREE: Classification tree.

To carry out the computations we have used the function `train` of the `R` package `caret` (see R Core Team (2016) and Kuhn (2008)) using standard 10-fold cross validation to select the tuning constants of the classifiers when necessary.

We have considered six different Gaussian scenarios and another non-Gaussian. For each situation, simulations are run for dimensions $d = 6, 10, 20$ and 40. Next, we describe the considered models:

M1: Equal spherical covariance matrix. In this case $\Sigma_0 = \Sigma_1 = \mathbf{I}$, the identity matrix. The population means are $\mu_0 = (0,\ldots,0)^\top$, $\mu_1 = (3,\ldots,3)^\top \in \mathbb{R}^d$.
M2: Unequal spherical covariance matrix. In this case $\Sigma_0 = \mathbb{I}$ and $\Sigma_1 = 2\mathbf{I}$. The population means are $\mu_0 = (0,\ldots,0)^\top$, $\mu_1 = (3,\ldots,3)^\top \in \mathbb{R}^d$.
M3: Equal highly ellipsoidal covariance matrix. In this case $\Sigma_0 = \Sigma_1 = \mathrm{diag}(e_1,\ldots,e_d)$, where $e_i = (9(i-1)/(d-1)+1)^2$, $i = 1,\ldots,d$.
M3.1: Classes concentrated in low variance subspace. The population means are $\mu_0 = (0,\ldots,0)^\top$ and $\mu_1 = (\mu_{1,1},\ldots,\mu_{1,d})^\top \in \mathbb{R}^d$, where $\mu_{1,i} = 2.5\sqrt{e_i/d}(d-i)(d/2-i)^{-1}$, $i = 1,\ldots,d$.

M3.2: Classes concentrated in high variance subspace. The population means are $\mu_0 = (0,\ldots,0)^\top$, $\mu_1 = (\mu_{1,1},\ldots,\mu_{1,d})^\top \in \mathbb{R}^d$, where $\mu_{1,i} = 2.5\sqrt{e_i/d}(i-1)(d/2-i)^{-1}$, $i = 1,\ldots,d$.

M4: Unequal highly ellipsoidal covariance matrix. Under this model the covariance matrices are $\Sigma_0 = \text{diag}(a_1,\ldots,a_d)$ and $\Sigma_1 = \text{diag}(b_1,\ldots,b_d)$, where $a_i = [9(d-i)/(d-1)+1]^2$ and $b_i = \{9[i-(d-1)/2]/(d-1)\}^2$, $i = 1,\ldots,d$.

M4.1: Equal population means. In this case, $\mu_0 = \mu_1 = (0,\ldots,0)^\top \in \mathbb{R}^d$.

M4.2: Unequal population means. This case corresponds to $\mu_0 = (0,\ldots,0)^\top$, $\mu_1 = (14,\ldots,14)^\top/\sqrt{d} \in \mathbb{R}^d$.

M5: Clusters within a class. In this case, the first class is Gaussian with $\mu_0 = (0,3,0,\ldots,0) \in \mathbb{R}^d$ and $\Sigma_0 = \text{diag}(1,4,1,\ldots,1)$. For the second class, 50% of the observations are Gaussian with $\mu_{1,A} = (-3,0,0,\ldots,0) \in \mathbb{R}^d$ and $\Sigma_1 = \Sigma_0$, and the other 50% are Gaussian with $\mu_{1,B} = (3,0,0,\ldots,0) \in \mathbb{R}^d$ and $\Sigma_1 = \Sigma_0$.

Observe that under M1 and M3 the Bayes rule is actually LDA, whereas under M2 and M4 the Bayes rule is QDA. Notice that the design of the simulation study does not particularly favor multidirectional rules, whose behavior tends to be better when there are clusters within the populations as illustrated by the toy example in Section 4 and pointed out in Huang et al (2012).

In Tables 2, 3 and 4 we report the results for dimensions $d = 10$, $d = 20$ and $d = 40$, respectively. The results for $d = 6$ are similar so they have been omitted.

In those situations where linear rules are optimal (M1 and M3) using one or two products of hyperplanes is enough to approximate the classification errors of the original quadratic rule. Even in some cases the multidirectional classifiers provide a better result than the approximated quadratic rules. This is reasonable because the approximation of the quadratic rule might be closer to the optimal linear rule. In M2 one covariance matrix is a dilatation of the other one and the boundary of the Bayes rule is a sphere surrounding one of the centers. In dimension 2, this situation corresponds to Figure 3. This example is therefore difficult to approximate with just a few number of products of hyperplanes in high dimensions. In model M4.1 and M4.2, half of the eigenvalues of the corresponding matrix $\mathbf{B}$ in (7) are positive and the other half negative. Therefore, it is not posible to obtain a good approximation of the quadratic rule by considering only half of the spectrum of the quadratic structure of the classifier. Finally, M5 is the prototypical example in which our approach works well as there are only two informative variables plus additional noise. In this case, one product of hyperplanes is able to capture all the relevant information of the classification problem.

6.1 Some computational issues

The decomposition procedure proposed in this paper relies on the diagonalization of matrix $\mathbf{B}$ in (7). For symmetric matrices with dimensions from 1000×1000 to 10000×10000, the computation of the eigenvalues and eigenvectors does not represent an important problem from a computational point of view. Most of the software programs use QR iteration (see Golub and Van Loan (2013)). Though computational

	M1	M2	M3.1	M3.2	M4.1	M4.2	M5
LDA	7.0 (1.9)	12.0 (2.2)	5.2 (1.4)	5.3 (1.6)	47.8 (3.8)	2.6 (1.3)	24.0 (3.4)
LSVM	7.3 (2.0)	12.3 (2.2)	6.1 (2.0)	6.0 (1.8)	41.4 (5.5)	2.9 (1.4)	23.8 (3.2)
QDA	8.0 (1.9)	9.9 (2.0)	6.2 (1.7)	6.0 (1.6)	1.1 (0.6)	0.1 (0.2)	12.4 (2.2)
QDA-M1	7.8 (2.2)	27.8 (5.4)	6.1 (2.0)	6.7 (2.1)	15.6 (2.8)	5.3 (1.7)	14.6 (8.5)
QDA-M2	7.5 (2.0)	21.9 (4.2)	6.0 (1.7)	6.0 (1.8)	5.5 (1.7)	1.4 (1.0)	15.0 (8.2)
RDA	7.0 (2.0)	10.8 (2.2)	5.9 (1.6)	5.2 (1.6)	10.2 (2.5)	1.7 (1.0)	21.3 (3.1)
RDA-M1	7.1 (1.9)	12.7 (2.4)	5.9 (1.6)	5.2 (1.6)	44.1 (2.3)	3.3 (1.6)	24.4 (3.4)
RDA-M2	7.1 (1.9)	12.6 (2.4)	5.9 (1.6)	5.2 (1.6)	39.8 (2.5)	3.3 (1.6)	23.4 (3.2)
QSVM	14.3 (3.4)	16.6 (3.6)	16.5 (3.4)	11.6 (3.3)	9.2 (3.3)	3.0 (1.3)	13.2 (2.8)
QSVM-M1	15.4 (5.8)	25.2 (6.4)	20.1 (6.6)	15.8 (6.0)	32.6 (5.2)	13.7 (6.2)	13.0 (6.9)
QSVM-M2	13.5 (4.0)	20.0 (5.5)	16.5 (4.1)	14.1 (4.8)	24.6 (6.0)	9.9 (3.8)	13.0 (6.4)
GSVM	14.3 (3.4)	16.6 (3.6)	16.5 (3.4)	11.6 (3.3)	9.2 (3.3)	3.0 (1.3)	13.2 (2.8)
TREE	6.8 (1.9)	11.7 (2.5)	17.1 (3.4)	17.0 (3.5)	11.4 (2.8)	5.6 (2.8)	16.0 (4.7)

Table 2 Averages (standard errors) of the misclassification rates for dimension 10.

	M1	M2	M3.1	M3.2	M4.1	M4.2	M5
LDA	8.1 (2.2)	12.7 (2.3)	7.3 (1.8)	7.4 (1.8)	46.6 (4.3)	3.8 (1.7)	26.0 (3.1)
LSVM	9.2 (2.3)	13.2 (2.5)	8.9 (2.1)	9.5 (2.3)	41.0 (4.9)	4.8 (2.0)	26.2 (3.2)
QDA	12.8 (2.7)	11.9 (2.5)	12.1 (2.5)	12.5 (2.6)	0.1 (0.2)	0.0 (0.1)	20.0 (3.0)
QDA-M1	11.5 (3.7)	37.5 (4.8)	9.9 (3.1)	11.4 (3.6)	9.6 (2.0)	6.4 (2.1)	23.3 (9.1)
QDA-M2	10.1 (2.5)	32.0 (4.5)	9.1 (2.3)	9.5 (2.4)	2.0 (0.9)	1.0 (0.7)	22.7 (9.0)
RDA	8.1 (2.3)	10.5 (2.2)	7.9 (1.9)	7.2 (1.8)	3.8 (1.7)	1.4 (1.0)	24.2 (2.9)
RDA-M1	8.0 (2.2)	15.0 (2.8)	7.9 (1.8)	7.2 (1.8)	47.9 (1.6)	6.7 (2.2)	25.8 (3.1)
RDA-M2	8.0 (2.2)	14.9 (2.7)	7.9 (1.8)	7.2 (1.8)	46.3 (1.9)	6.6 (2.2)	25.6 (3.0)
QSVM	12.2 (2.8)	13.3 (2.5)	27.8 (3.5)	15.4 (3.0)	6.2 (1.9)	3.6 (1.4)	13.4 (2.5)
QSVM-M1	15.2 (6.1)	31.9 (6.3)	26.6 (6.8)	27.1 (7.0)	44.6 (2.2)	27.4 (5.3)	10.7 (3.9)
QSVM-M2	13.1 (4.1)	29.0 (5.3)	26.6 (5.0)	23.9 (5.5)	40.6 (3.2)	24.6 (4.1)	11.4 (4.2)
GSVM	12.2 (2.8)	13.3 (2.5)	27.8 (3.5)	15.4 (3.0)	6.2 (1.9)	3.6 (1.4)	13.4 (2.5)
TREE	7.5 (2.2)	11.4 (2.6)	26.2 (3.4)	26.9 (3.6)	6.0 (2.0)	6.1 (2.5)	15.5 (4.4)

Table 3 Averages (standard errors) of the misclassification rates for dimension 20.

cost grows as d^3, where d is the dimension of the matrix, for moderately large matrices (up to size 10000 approximately) this task can be done in a few minutes. The QR algorithm can be seen as a sophisticated variation of the basic *power eigenvalue algorithm*, which enables us to find the eigenvectors associated with the largest eigenvalues. The power eigenvalue algorithm procedure costs of order d^2 and can be used to compute only the largest eigenvalues and their corresponding eigenvectors in those cases in which the computational cost of computing the whole decomposition of the matrix $\mathbf{B}$ is too large.

Once the spectral decomposition of the matrix $\mathbf{B}$ has been carried out, another issue that strongly affects the computational cost of the proposed methodology is arranging the products of hyperplanes according to any given criterion, such as the training or the cross-validation error of the rule generated by each of them. Finding this ordering among the products of hyperplanes is crucial to obtain the principal linear components of the underlying quadratic classifier. In this way we can reduce the dimension of the classification problem through the multidirectional classifier as proposed in Section 3.3 and illustrated in Section 4. However, if we just want to use the generated hyperplanes as a dictionary of relevant directions (for example, to construct

	M1	M2	M3.1	M3.2	M4.1	M4.2	M5
LDA	10.0 (2.3)	14.7 (2.5)	10.3 (2.2)	10.0 (2.1)	46.1 (3.6)	5.6 (1.7)	27.5 (3.4)
LSVM	11.8 (2.5)	15.9 (3.1)	14.2 (3.0)	12.6 (2.6)	42.3 (4.5)	6.4 (1.9)	28.1 (3.6)
QDA	23.8 (3.6)	23.4 (3.5)	25.0 (3.9)	24.7 (3.8)	0.2 (0.3)	0.0 (0.2)	31.5 (3.7)
QDA-M1	20.9 (4.9)	40.3 (3.3)	19.5 (4.7)	20.1 (5.3)	5.8 (1.7)	4.4 (1.5)	35.0 (9.2)
QDA-M2	17.6 (3.8)	35.1 (3.4)	17.1 (3.5)	18.0 (4.2)	0.8 (0.7)	0.4 (0.4)	33.8 (8.6)
RDA	9.8 (2.0)	11.1 (2.3)	10.9 (2.2)	9.3 (2.1)	0.8 (0.7)	0.8 (0.6)	26.8 (3.4)
RDA-M1	9.7 (2.2)	19.6 (2.8)	10.9 (2.2)	9.3 (2.1)	49.4 (0.9)	12.7 (2.8)	27.3 (3.3)
RDA-M2	9.7 (2.2)	19.5 (2.8)	10.9 (2.2)	9.2 (2.1)	49.1 (1.1)	12.5 (2.7)	27.1 (3.3)
QSVM	12.9 (2.6)	16.5 (2.4)	37.7 (3.6)	16.0 (2.7)	7.5 (2.0)	5.5 (1.7)	13.1 (2.5)
QSVM-M1	11.0 (2.7)	39.7 (2.6)	30.1 (5.8)	19.5 (6.4)	49.1 (0.8)	41.1 (2.8)	9.5 (3.2)
QSVM-M2	11.1 (2.6)	39.5 (2.6)	31.4 (4.7)	19.2 (5.2)	48.5 (1.0)	40.4 (2.6)	10.2 (3.2)
GSVM	12.9 (2.6)	16.5 (2.4)	37.7 (3.6)	16.0 (2.7)	7.5 (2.0)	5.5 (1.7)	13.1 (2.5)
TREE	7.5 (1.8)	11.5 (2.4)	34.2 (4.1)	33.9 (3.9)	4.9 (4.0)	4.8 (3.5)	16.2 (4.2)

Table 4 Averages (standard errors) of the misclassification rates for dimension 40.

oblique classification trees as in Section 5.2), the arrangement of the hyperplanes is no longer necessary and the overall computation cost can be reduced substantially.

7 Technical appendix: Some theoretic aspects of the approximation

In this section we collect some theoretical results regarding the approximation of a quadratic classifier by products of hyperplanes. We first remark that Q in (2) can be expressed in matricial form in a unique way as

$$Q(\mathbf{x}) = \widetilde{\mathbf{x}}^\top \mathbf{A}_Q \widetilde{\mathbf{x}}, \qquad \mathbf{x} \in \mathbb{R}^d, \tag{14}$$

where $\widetilde{\mathbf{x}} = [1, \mathbf{x}]^\top$ and $\mathbf{A}_Q$ is the *associated matrix* with Q, that is, the symmetric matrix

$$\mathbf{A}_Q = \left[\begin{array}{c|c} c & \mathbf{a}^\top \\ \hline \mathbf{a} & \mathbf{A} \end{array}\right]. \tag{15}$$

7.1 When is Q exactly a product of two hyperplanes?

In the following theorem we give a simple and concise necessary and sufficient condition so that a quadratic form is a product of two hyperplanes.

Theorem 1 *Let Q be as in (2) with $c \neq 0$. Then, Q is a product of hyperplanes if and only if there exists $\mathbf{b} \in \mathbb{R}^d$ such that $\mathbf{B} = \mathbf{b}\mathbf{b}^\top$, where $\mathbf{B}$ is defined in (7). In such a case,*

$$Q(\mathbf{x}) = \frac{1}{c}\left[(\mathbf{a}+\mathbf{b})^\top \mathbf{x} + c\right]\left[(\mathbf{a}-\mathbf{b})^\top \mathbf{x} + c\right], \qquad \mathbf{x} \in \mathbb{R}^d. \tag{16}$$

Proof Let us assume that Q is a product of two hyperplanes. Then, there exist $\mathbf{v}$, $\mathbf{w} \in \mathbb{R}^d$ such that $Q(\mathbf{x}) = (\mathbf{v}^\top \mathbf{x} + c)(\mathbf{w}^\top \mathbf{x} + 1)$. Therefore,

$$\begin{aligned} Q(\mathbf{x}) &= \mathbf{x}^\top (\mathbf{v}\mathbf{w}^\top)\mathbf{x} + (\mathbf{v} + c\mathbf{w})^\top \mathbf{x} + c \\ &= \mathbf{x}^\top \left(\frac{\mathbf{v}\mathbf{w}^\top + \mathbf{w}\mathbf{v}^\top}{2}\right)\mathbf{x} + (\mathbf{v} + c\mathbf{w})^\top \mathbf{x} + c. \end{aligned} \tag{17}$$

As every quadratic form can be expressed in a unique way by a symmetric matrix, from (2) and (17) we conclude that

$$\mathbf{v}\mathbf{w}^\top + \mathbf{w}\mathbf{v}^\top = 2\mathbf{A} \qquad \text{and} \qquad \mathbf{v} + c\mathbf{w} = 2\mathbf{a}.$$

We then have that $\mathbf{a}\mathbf{w}^\top + \mathbf{w}\mathbf{a}^\top - c\mathbf{w}\mathbf{w}^\top = \mathbf{A}$. Hence, $\mathbf{B} = (\mathbf{a} - c\mathbf{w})(\mathbf{a} - c\mathbf{w})^\top$, and we can take $\mathbf{b} = \mathbf{a} - c\mathbf{w}$.

On the other hand, let us now assume that there exists $\mathbf{b} \in \mathbb{R}^d$ satisfying $\mathbf{B} = \mathbf{b}\mathbf{b}^\top$. We then have

$$c\mathbf{A} = \mathbf{a}\mathbf{a}^\top - \mathbf{b}\mathbf{b}^\top = \frac{(\mathbf{a}+\mathbf{b})(\mathbf{a}-\mathbf{b})^\top + (\mathbf{a}-\mathbf{b})(\mathbf{a}+\mathbf{b})^\top}{2}. \tag{18}$$

From (18), it is easy to show that (16) holds and the proof is complete.

For the sake of completeness, we include the following result for the case $c = 0$. The proof is similar to that of Theorem 1 and it is therefore omitted.

Theorem 2 *Let Q be as in (2) with $c = 0$.*

(a) If $\mathbf{a} \neq \mathbf{0}$, Q is a product of two hyperplanes if and only if there exists $\mathbf{b} \in \mathbb{R}^d$ such that $\mathbf{A} = \mathbf{a}\mathbf{b}^\top + \mathbf{b}\mathbf{a}^\top$. In such a case, $Q(\mathbf{x}) = 2\,(\mathbf{a}^\top\mathbf{x})(\mathbf{b}^\top\mathbf{x} + 1)$, for $\mathbf{x} \in \mathbb{R}^d$.

(b) If $\mathbf{a} = \mathbf{0}$, Q is a product of two hyperplanes if and only if there exist $\mathbf{v}, \mathbf{w} \in \mathbb{R}^d$ such that $\mathbf{A} = \mathbf{v}\mathbf{w}^\top + \mathbf{w}\mathbf{v}^\top$. In such a case, $Q(\mathbf{x}) = 2\,(\mathbf{v}^\top\mathbf{x})(\mathbf{w}^\top\mathbf{x})$, for $\mathbf{x} \in \mathbb{R}^d$.

If the condition in Theorem 1 is satisfied, then Q is a product of two secant hyperplanes if and only if $\mathbf{a}$ and $\mathbf{b}$ are linearly independent. The following corollaries provide some results regarding the normal vectors of the hyperplanes in (16). In what follows $\mathrm{tr}(\mathbf{M})$ stands for the trace of the matrix $\mathbf{M}$.

Corollary 2 *If Theorem 1 holds, then $(\mathbf{a}+\mathbf{b})^\top(\mathbf{a}-\mathbf{b}) = c\,\mathrm{tr}(\mathbf{A})$. In particular, the normal vectors of the hyperplanes in (16) are orthogonal if and only if $\mathrm{tr}(\mathbf{A}) = 0$.*

Proof The conclusion follows from Theorem 1 and the following identities,

$$\begin{aligned}(\mathbf{a}+\mathbf{b})^\top(\mathbf{a}-\mathbf{b}) &= \mathbf{a}^\top\mathbf{a} - \mathbf{b}^\top\mathbf{b} \\ &= \mathrm{tr}(\mathbf{a}\mathbf{a}^\top - \mathbf{b}\mathbf{b}^\top) \\ &= \mathrm{tr}(c\mathbf{A}).\end{aligned}$$

Corollary 3 *Let Q be as in (2) with $c \neq 0$. Then, Q is a product of two secant hyperplanes if and only if $\mathrm{sg}(\mathbf{A}) = \mathrm{sg}(\mathbf{A}_Q) = (1,1)$.*

Proof Let us assume that Q is a product of two secant hyperplanes. From Theorem 1, we obtain that there exists $\mathbf{b} \in \mathbb{R}^d$ such that $\mathbf{B} = \mathbf{b}\mathbf{b}^\top$ and $c\mathbf{A} = \mathbf{a}\mathbf{a}^\top - \mathbf{b}\mathbf{b}^\top$, with $\mathbf{a}$ and $\mathbf{b}$ linearly independent. Thus, we have that $\mathrm{sg}(\mathbf{B}) = (1,0)$ and $\mathrm{sg}(\mathbf{A}) = (1,1)$. On the other hand, $\mathbf{A}_Q$ fulfills

$$\mathbf{M}\mathbf{A}_Q\mathbf{M}^\top = \frac{-1}{c}\left[\begin{array}{c|c} -c^2 & \mathbf{0}^\top \\ \hline \mathbf{0} & \mathbf{B} \end{array}\right], \quad \text{with} \quad \mathbf{M} = \left[\begin{array}{c|c} 1 & \mathbf{0}^\top \\ \hline -\mathbf{a}/c & \mathbf{I} \end{array}\right], \tag{19}$$

where $\mathbf{0} = (0,\ldots,0)^\top \in \mathbb{R}^d$ and $\mathbf{I}$ is the $d \times d$ identity matrix. Therefore, the matrix $\mathbf{A}_Q$ has the same rank and signature as $\mathbf{M}\mathbf{A}_Q\mathbf{M}^\top$ and then $\mathrm{sg}(A_Q) = (1,1)$. The other implication can be easily deduced from (19).

The proof of the previous corollary can also be derived from known results on the classification of hyperquadratic forms. If $d = 2$, the conditions in the last corollary are equivalent to $|\mathbf{A}| < 0$ and $|\mathbf{A}_Q| = 0$.

7.2 Negative eigenvalues of $\mathbf{B}$

The negative eigenvalues of $\mathbf{B}$ in (7) can also be used to approximate Q as the following corollary shows.

Corollary 4 *For each pair λ_i, λ_j of eigenvalues of $\mathbf{B}$ such that $\lambda_i > 0$ and $\lambda_j < 0$, we have that $Q = P_{i,j} + R_{i,j}$, where $cP_{i,j} = L_{i,j,1}L_{i,j,2}$ is the product of two hyperplanes given by*

$$L_{i,j,1}(\mathbf{x}) = (\mathbf{b}_j + \mathbf{b}_i)^\top \mathbf{x}, \quad L_{i,j,2}(\mathbf{x}) = (\mathbf{b}_j - \mathbf{b}_i)^\top \mathbf{x}, \qquad \mathbf{x} \in \mathbb{R}^d, \tag{20}$$

with $\mathbf{b}_i = \sqrt{\lambda_i}\mathbf{v}_i$, $\mathbf{b}_j = \sqrt{|\lambda_j|}\mathbf{v}_j$ and $R_{i,j}$ is a remainder term that can be expressed as

$$R_{i,j}(\mathbf{x}) = \frac{1}{c}(\mathbf{a}^\top \mathbf{x} + c)^2 - \frac{1}{c}\sum_{i,j\neq\ell=1}^{r} \lambda_\ell (\mathbf{v}_\ell^\top \mathbf{x})^2$$

Proof From (6), we obtain

$$\begin{aligned} cQ(\mathbf{x}) =& (\mathbf{b}_j^\top \mathbf{x})^2 - (\mathbf{b}_i^\top \mathbf{x})^2 + cR_{i,j}(\mathbf{x}) \\ =& L_{i,j,1}(\mathbf{x})L_{i,j,2}(\mathbf{x}) + cR_{i,j}(\mathbf{x}). \end{aligned}$$

In this case, $L_{i,j,1}(\mathbf{0}) = L_{i,j,2}(\mathbf{0}) = 0 \neq Q(\mathbf{0})$ and, in general, $\nabla P_{i,j}(\mathbf{0}) = \mathbf{0} \neq \nabla Q(\mathbf{0})$. However, the hyperplanes defined by equations $L_{i,j,1}(\mathbf{x}) = 0$ and $L_{i,j,2}(\mathbf{x}) = 0$ are also linear approximations of the classification boundary $Q(\mathbf{x}) = 0$ because they are tangent to the surface defined by this equation. Explicit expressions for the tangent points can be found in Proposition 3 of Subsection 7.3. These products take into account other aspects of the shape of the boundary $Q(\mathbf{x}) = 0$.

Further, the classifier obtained with the combination of the products in Corollary 4 with respect to the matrix $\mathbf{B}_0$ is very similar to the one obtained with the positive eigenvalues of the matrix $\mathbf{B}_1$. In other words, the relevant products obtained with negative eigenvalues (around μ_0) play a similar role as the products obtained with positive eigenvalues, but approximating Q around μ_1.

We finally observe that if $\mathrm{sg}(\mathbf{B}) = (r_+, r_-)$, from Corollaries 1 and 4, it is possible to obtain $m = r_+(1 + d - r_+)$ approximating products of hyperplanes. Moreover, as $r_+ \geq 1$, it holds that $m \geq d$. The quantity m can be hence viewed as the number of hyperplanes required to wrap up the boundary of Q. Therefore, it provides information on the geometric complexity of the quadratic classifier.

7.3 Tangent points of the approximating hyperplanes

Here we provide explicit expressions for the tangent points of the hyperplanes in (8) and (20).

Proposition 2 *Let λ_i be a positive eigenvalue of* $\mathbf{B}$ *and let us assume that* $\mathbf{a}^\top \mathbf{v}_i \pm \sqrt{\lambda_i} \neq 0$. *The hyperplanes defined by the equations $L_{i,1}(\mathbf{x}) = 0$ and $L_{i,2}(\mathbf{x}) = 0$ are tangent to the surface $Q(\mathbf{x}) = 0$ at the points $\mathbf{x}_{i,j} = \alpha_{i,j}\mathbf{v}_i$ ($j = 1,2$), respectively, where $\alpha_{i,1} = -c/(\mathbf{a}^\top \mathbf{v}_i + \sqrt{\lambda_i})$ and $\alpha_{i,2} = -c/(\mathbf{a}^\top \mathbf{v}_i - \sqrt{\lambda_i})$.*

Proof From Corollary 1, it is easy to check that $Q(\mathbf{x}_{i,1}) = L_{i,1}(\mathbf{x}_{i,1}) = 0$ and we also have

$$c\nabla Q(\mathbf{x}) = L_{i,1}(\mathbf{x})\nabla L_{i,2}(\mathbf{x}) + L_{i,2}(\mathbf{x})\nabla L_{i,1}(\mathbf{x}) - 2\sum_{i\neq\ell=1}^{r} \lambda_\ell \left(\mathbf{v}_\ell^\top \mathbf{x}\right)\mathbf{v}_\ell.$$

Therefore, $c\nabla Q(\mathbf{x}_{i,1}) = L_{i,2}(\mathbf{x}_{i,1})\nabla L_{i,1}(\mathbf{x}_{i,1})$, and this shows that the hyperplane $L_{i,1}(\mathbf{x}) = 0$ is tangent to $Q(\mathbf{x}) = 0$ at the point $\mathbf{x}_{i,1}$. The corresponding proof for $L_{i,2}$ is analogous and it is therefore omitted.

Proposition 3 *Let $\lambda_i > 0$ and $\lambda_j < 0$ be two eigenvalues of* $\mathbf{B}$ *and let us assume that* $\mathbf{a}^\top\left(\sqrt{|\lambda_j|}\mathbf{v}_i \pm \sqrt{\lambda_i}\mathbf{v}_j\right) \neq 0$. *The hyperplanes defined by the equations $L_{i,j,1}(\mathbf{x}) = 0$ and $L_{i,j,2}(\mathbf{x}) = 0$ are tangent to the surface $Q(\mathbf{x}) = 0$ at the points $\mathbf{x}_{i,j,k} = \alpha_{i,j,k}\mathbf{v}_i + \beta_{i,j,k}\mathbf{v}_j$ ($k = 1,2$), respectively, where*

$$\alpha_{i,j,1} = \frac{-c\sqrt{|\lambda_j|}}{\mathbf{a}^\top(\sqrt{|\lambda_j|}\mathbf{v}_i - \sqrt{\lambda_i}\mathbf{v}_j)}, \qquad \beta_{i,j,1} = \frac{c\sqrt{\lambda_i}}{\mathbf{a}^\top(\sqrt{|\lambda_j|}\mathbf{v}_i - \sqrt{\lambda_i}\mathbf{v}_j)},$$
$$\alpha_{i,j,2} = \frac{-c\sqrt{|\lambda_j|}}{\mathbf{a}^\top(\sqrt{|\lambda_j|}\mathbf{v}_i + \sqrt{\lambda_i}\mathbf{v}_j)}, \qquad \beta_{i,j,2} = \frac{-c\sqrt{\lambda_i}}{\mathbf{a}^\top(\sqrt{|\lambda_j|}\mathbf{v}_i + \sqrt{\lambda_i}\mathbf{v}_j)}.$$

Proof We just give the proof for the hyperplane $L_{i,j,1}(\mathbf{x}) = 0$ because the corresponding for $L_{i,j,2}$ is analogous. From Corollary 4, we have that

$$cQ(\mathbf{x}) = cL_{i,j,1}(\mathbf{x})L_{i,j,2}(\mathbf{x}) + (\mathbf{a}^\top\mathbf{x} + c)^2 - \sum_{i,j\neq\ell=1}^{r} \lambda_\ell(\mathbf{v}_\ell^\top\mathbf{x})^2. \tag{21}$$

It can be easily seen that $L_{i,j,1}(\mathbf{x}_{i,j,1}) = \mathbf{a}^\top\mathbf{x}_{i,j,1} + c = 0$. Also, from (21) and the orthonormality of the vectors $\{\mathbf{v}_\ell\}_{\ell=1}^r$, we obtain that $Q(\mathbf{x}_{i,j,1}) = 0$. Finally, from (21), we have

$$c\nabla Q(\mathbf{x}) = L_{i,j,1}(\mathbf{x})\nabla L_{i,j,2}(\mathbf{x}) + L_{i,j,2}(\mathbf{x})\nabla L_{i,j,1}(\mathbf{x}) + 2(\mathbf{a}^\top\mathbf{x} + c)\mathbf{a} - 2\sum_{i,j\neq\ell=1}^{r} \lambda_\ell \left(\mathbf{v}_\ell^\top\mathbf{x}\right)\mathbf{v}_\ell,$$

and this yields $c\nabla Q(\mathbf{x}_{i,j,1}) = L_{i,j,2}(\mathbf{x}_{i,j,1})\nabla L_{i,j,1}(\mathbf{x}_{i,j,1})$, and the proof is complete.

7.4 The number of positive eigenvalues of $\mathbf{B}_0$ and $\mathbf{B}_1$

The following result provides information about the number of positive eigenvalues of $\mathbf{B}_0$ and $\mathbf{B}_1$ in (10). In what follows, $r_+(\mathbf{M})$ and $r_-(\mathbf{M})$ indicate the number of positive and negative eigenvalues of the matrix $\mathbf{M}$, respectively.

Proposition 4 *Let us consider the matrices* $\mathbf{B}_0$ *and* $\mathbf{B}_1$ *defined in (10). We have:*

(a) If $c_0c_1 < 0$, *then* $\operatorname{rank}(\mathbf{A}) \le r_+(\mathbf{B}_0) + r_+(\mathbf{B}_1) \le \operatorname{rank}(\mathbf{A}) + 2$.
(b) If $c_0, c_1 > 0$, *then* $2r_-(\mathbf{A}) \le r_+(\mathbf{B}_0) + r_+(\mathbf{B}_1) \le 2(1 + r_-(\mathbf{A}))$.
(c) If $c_0, c_1 < 0$, *then* $2r_+(\mathbf{A}) \le r_+(\mathbf{B}_0) + r_+(\mathbf{B}_1) \le 2(1 + r_+(\mathbf{A}))$.

Moreover, all the previous inequalities are sharp in the sense that they are actually equalities in some situations.

Proof Let us assume that $c_0 < 0$ and $c_1 > 0$. The rest of the cases can be treated in a similar way and the proofs are left to the reader. First, from (19), we see that the matrices

$$\mathbf{A}_i = \left[\begin{array}{c|c} c_i & \mathbf{a}_i^\top \\ \hline \mathbf{a}_i & \mathbf{A} \end{array}\right] \quad \text{and} \quad \mathbf{M}_i = \left[\begin{array}{c|c} c_i & \mathbf{0}^\top \\ \hline \mathbf{0} & -\mathbf{B}_i/c_i \end{array}\right] \qquad (i = 0, 1)$$

have the same signature. As c_i is an eigenvalue of the matrix $\mathbf{M}_i$ $(i = 0, 1)$, we conclude that $r_+(\mathbf{B}_0) = r_+(\mathbf{A}_0)$ and $r_+(\mathbf{B}_1) = r_-(-\mathbf{B}_1) = r_-(\mathbf{A}_1)$. We then obtain

$$r_+(\mathbf{B}_0) + r_+(\mathbf{B}_1) = r_+(\mathbf{A}_0) + r_-(\mathbf{A}_1). \tag{22}$$

Finally, we observe that, by the Sylvester's criterion to compute the signature of real and symmetric matrices, we have

$$\begin{aligned} r_+(\mathbf{A}) \le r_+(\mathbf{A}_0) \le r_+(\mathbf{A}) + 1, \\ r_-(\mathbf{A}) \le r_-(\mathbf{A}_1) \le r_-(\mathbf{A}) + 1. \end{aligned} \tag{23}$$

Therefore, the inequalities in (a) follow by (22) and (23). Finally, it is easy to check that the left-hand side inequalities in (a)–(d) are in fact equalities if $\mathbf{A} = \mathbf{I}$ (the $d \times d$ identity matrix) and $\mathbf{a}_0 = \mathbf{a}_1 = \mathbf{0}$. The right-hand side inequalities in (a)–(d) are also equalities when $\mathbf{A} = \mathbf{0}$ (the matrix whose elements are 0) and $\mathbf{a}_0 \neq \mathbf{0} \neq \mathbf{a}_1$.

Observe that the condition $c_0c_1 < 0$ in Proposition 4 (a) amounts to saying, in terms of the classification problem, that μ_0 and μ_1 are classified in different groups by η_Q. Therefore, this is a common situation in practice. In such a case, if one of the vectors $\mathbf{a}_0$ or $\mathbf{a}_1$ is non zero, then

$$r_+(\mathbf{B}_0) + r_+(\mathbf{B}_1) \ge \max\{1, \operatorname{rank}(\mathbf{A})\}.$$

In particular, when we estimate the parameters of Q with the training sample, we usually obtain that $\operatorname{rank}(\mathbf{A}) = d$ (with probability 1), and we thus have

$$\left\lceil \frac{d}{2} \right\rceil \le \max\{r_+(\mathbf{B}_0), r_+(\mathbf{B}_1)\} \le d,$$

where $\lceil \cdot \rceil$ is the ceiling function.

Acknowledgements The authors are grateful to the reviewers and the associate editor for their insightful comments which have improved the presentation of the paper. We would like to thank Jesús María Arregui (University of the Basque Country) and Jesús Gonzalo (Universidad Autnoma de Madrid) with whom we have shared illuminating conversations on quadratic forms.

References

Bache K, Lichman M (2013) UCI Machine Learning Repository. URL `http://archive.ics.uci.edu/ml`

Devroye L, Györfi L, Lugosi G (1996) A probabilistic theory of pattern recognition, Applications of Mathematics (New York), vol 31. Springer-Verlag, New York

Fan J, Ke ZT, Liu H, Xia L (2015) QUADRO: a supervised dimension reduction method via Rayleigh quotient optimization. Ann Statist 43(4):1498–1534

Friedman JH (1989) Regularized discriminant analysis. J Amer Statist Assoc 84(405):165–175

Golub GH, Van Loan CF (2013) Matrix computations, 4th edn. Johns Hopkins Studies in the Mathematical Sciences, Johns Hopkins University Press, Baltimore, MD

Hand DJ (2006) Classifier technology and the illusion of progress. Statist Sci 21(1):1–34, with comments and a rejoinder by the author

Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning, 2nd edn. Springer Series in Statistics, Springer, New York

Huang H, Liu Y, Marron JS (2012) Bidirectional discrimination with application to data visualization. Biometrika 99(4):851–864

Kuhn M (2008) Building predictive models in R using the caret package. J Stat Softw 28:1–26

Park SH, Fürnkranz J (2007) Efficient pairwise classification. In: European Conference on Machine Learning, Springer, pp 658–665

R Core Team (2016) R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria, URL `http://www.R-project.org/`

Rifkin R, Klautau A (2004) In defense of one-vs-all classification. J Mach Learn Res 5:101–141

Ripley B (2014) tree: Classification and regression trees. R package version 1.0-35, URL `http://CRAN.R-project.org/package=tree`

Truong A (2009) Fast growing and interpretable oblique trees via logistic regression models. Doctoral dissertation, University of Oxford

Wald PW, Kronmal R (1977) Discriminant functions when covariances are unequal and sample sizes are moderate. Biometrics 33:479–484