Chapter 4
A Geometric Approach to Feature Ranking Based Upon Results of Effective Decision Boundary Feature Matrix

Claudia Diamantini, Alberto Gemelli and Domenico Potena

Abstract This chapter presents a new method of *Feature Ranking* (FR) that calculates the *relative weight of features* in their original domain with an algorithmic procedure. The method supports information selection of real world features and is useful when the number of features has costs implications. The *Feature Extraction* (FE) techniques, although accurate, provide the weights of artificial features whereas it is important to weight the real features to have readable models. The accuracy of the ranking is also an important aspect; the *heuristics methods*, another major family of ranking methods based on generate-and-test procedures, are by definition approximate although they produce readable models. The ranking method proposed here combines the advantages of older methods, it has at its core a feature extraction technique based on *Effective Decision Boundary Feature Matrix* (EDBFM), which is extended to calculate the total *weight* of the real features through a procedure geometrically justified. The modular design of the new method allows to include any FE technique referable to the EDBFM model; a thorough benchmarking of the various solutions has been conducted.

Keywords Feature ranking · Feature weight · Effective decision boundary feature matrix · Classification

4.1 Introduction

The recent developments of information technology dramatically increased the capability of gathering information. This information is described by a high number of attributes, observations or measures, generically called *features*. On the one hand this

C. Diamantini · A. Gemelli (✉) · D. Potena
Dipartimento di Ingegneria Dell'Informazione, Università Politecnica Delle Marche,
via Brecce Bianche, 60131 Ancona, Italy
e-mail: albertogemelli@hotmail.com

C. Diamantini
e-mail: diamantini@dii.univpm.it

D. Potena
e-mail: potena@dii.univpm.it

U. Stańczyk and L.C. Jain (eds.), *Feature Selection for Data and Pattern Recognition*, Studies in Computational Intelligence 584,
DOI 10.1007/978-3-662-45620-0_4

improves our ability to study real phenomena, but on the other hand huge amounts of data produce an "informative overload", raising data acquisition and processing costs without effective exploitation of information. What is more, most of machine learning techniques suffer from the so called "curse of dimensionality" effect, and human interpretation of models generated by these techniques can be difficult on high dimensional spaces. To address these issues, the adoption of *Feature Selection* (FS) in processes is observing increasing interest and expansion.

Decision making and operations in the modern production contexts require a FS method which is generally valid for all applications, therefore robust and flexible, able to operate interactively in a dynamic information environment, dealing effectively with challenges posed by data heterogeneity, data bandwidth and real-time requirements. The large availability of information represents also a challenge because of the exponential growth of data acquisition costs and, last but not least, energy consumption by computers and acquisition sensor systems. The FS process represents a complex decisional mechanism in which the accuracy of results is equally important as usability, fastness, robustness and scalability. In the scientific literature, the current approaches to FS in the machine learning process show distinct solutions which address specific issues and highlight opposite vintages, though many practical issues have arisen around applications in productive contexts that have never been considered on the whole. This is the context that inspires the invention and validation of our novel Feature Ranking (FR) method that supports the FS. This chapter proposes an innovative approach to FR that detains vintages otherwise dispersed over a variety of distinct methods. Our research is articulated over two main objectives, the first is to obtain feature ranking leading to high accuracy in machine learning goals achievement, the second is to provide an algorithm capable to actively consider cost functions in supporting decision making. These issues have been studied in relation to a machine learning process among the most known: the classification.

4.2 Feature Ranking for Classification: The Background Picture

4.2.1 Intrinsic Discriminant Dimension of a Classification Task

In the literature FS refers to the problem of selecting a subset of relevant features for building robust learning models [19, 27]. The concept of *optimal feature subset* has been refined during the years by the comprehension of the dataset properties that condition the classification performance. As it happens in generic data collections, many of the features are insignificant to reach a learning objective. A definition of *relevant feature* is provided by [3]: a feature $\mathbf{x_i}$ is strongly relevant to dataset X if there exist examples A and B in X that differ only in their assignment to $\mathbf{x_i}$ and have different labels. A feature $\mathbf{x_i}$ is weakly relevant to classification accuracy if it is possible to remove a subset of the features so that becomes strongly relevant.

In a classification task, the FS is used to predict the so called *intrinsic discriminant dimension* of the dataset, which has been defined by Lee and Landgrebe [24] as the smallest dimensional subspace wherein the same classification accuracy can be obtained as could be obtained in the original space. Effects of FS on accuracy have more recently been studied by Sima et al. [34]. In [21, 35], the problem of FS is seen as trade-off between generalization and specialization or, equivalently, a trade-off between bias and variance of the inductive process. A classification algorithm partitions the instance space into regions; when the number of features is relatively small, regions are too large, that causes the partitioning of the instances to be poor in terms of generalization and therefore accuracy decreases, this phenomenon is called *bias*. When the number of features is high, the probability that individual regions are labeled with the wrong class is increased too. This effect is called *variance*. Decision tree and neural network classifiers are particularly sensible to variance. There emerges the concept of irrelevant/redundant features that might cause the classification algorithms loosing efficiency and accuracy, whereas the subset of features that improves the performance of learning algorithms is defined *optimal subset*. All the aspects of the learning algorithm sensitivity to the dataset dimensionality, have been generally named as the *curse of dimensionality* by Kira and Rendell [20].

The optimal subset can be detected on a feature evaluation function [8]. When doing classification, an *Evaluation Function* (EF) expresses for each feature subset its ability to discriminate between classes. The effectiveness of the EF in highlighting the relative importance of feature depends on the search strategy by which the space of all possible subsets is explored, and it has measurable properties: accuracy (how accurate is the prediction of the EF), generality (how suitable is the EF for different classifiers) and time complexity (time taken to calculate the EF). A selection based on classification accuracy can be considered effective if the classifier error rate does not significantly decrease after selection. The authors indicate the *1NN classifier* as a convenient algorithm to build the evaluation function since it appears to always provide a reasonable classification performance in most applications.

4.2.2 Classical Feature Selection Strategies

The FS process is divided generally into two phases: FR and FS in the strict sense. It is necessary to rank the relative importance of features before proceeding to an optimal selection and then learning a classification model, although these two phases can be integrated in different modes as it will be discussed in this section. The progress in scientific research almost coincides for ranking and selection. As in the survey of [2, 15], the FS methods are categorized in two main categories: (i) methods that explore the space of possible subsets, searching an optimal subset of features by using heuristics to limit computational complexity, (ii) methods that rank features individually based on properties that good features are presumed to have, such as their contribution to class separability. In the classification learning process, input dataset is arranged in a n by m matrix where each row, or record, represents an

object belonging to a class, and each column represents a characterizing feature. In a geometric sight, the objects can be thought as points positioned in an m-dimensional space of features. The solution to a problem of classification can be thought as the procedure that finds the hyperplanes that, in the feature space, separate the classes of points.

A broad group of FS techniques is based on the construction and ranking of new features [11, 16]. The *Feature Extraction* (FE) process is based on a transformation of the original set of real features by a linear combination of these, by which the power to discriminate among classes is concentrated on a reduced number of extracted features. The relevancy of each individual feature is evaluated, in fact the set of *eigenvalues*, always associated with the transformation process, represents the relative relevancy of each extracted feature and allows ranking them. It is important to notice, however, that FE methodology was conceived primarily to do data compression, therefore it effectively reduces the size of the initial volume of data, but it implies the entire dataset to be available to construct each extracted feature; clearly the FE approach is of no help in an application where the containment of data acquisition costs is important. Furthermore the FE model is very application specific since extracted features are uniquely associated with a dataset.

For FS, three modes of application have been identified by [6, 16, 29] in relation to the dependence on the classification algorithm: in the *wrapper* mode, selection and classification are iterated to refine the selection of features up to achieving an optimal performance of the classification algorithm. The exploration of the solution space can be conducted either with the brute force or the heuristic approaches. The wrapper mode is supervised and is not suitable for applications in real-time, although some solutions have been proposed that increase its performance whilst avoiding its procedural complexity [28]. By contrast, in the *filter* mode the features that respond to a general criterion of relevancy for a classification process are selected. The filter method is applied in a unique step independently of the classification algorithm. In the *Embedded* mode, FS is part of the model training process, and features relevancy is obtained by analyzing their utility for optimizing the objective function of the learning model; an application example is in [30]. From a productive point of view these three methods represent different levels of trade-off between ease of execution and accuracy of the results.

When *heuristic* methods are used in feature selection the search of the optimal subset is done by attempts, by which there is built an evaluation function that provides for each subset of real features its ability to discriminate between classes [8, 36]. The results depend sensibly on the heuristic adopted and the amount of points effectively explored of the solution space. Because of the underlying subjective assumption, the heuristic approach is not fully reliable [1, 33], however it has the vintage to produce a rank model for real world features, therefore retaining a human interpretability. Among the heuristic strategies we would like to describe briefly the following: *Gain Ratio, One-Rule and Relief-F*.

The *Gain Ratio* algorithm [32] uses information entropy to find out how well a feature separates instances. The goodness of each individual feature depends of how broadly and uniformly it splits the considered data. Features are sorted from the most

relevant (the one with the highest gain ratio) to the least relevant (the one with the lowest gain ratio). Then, a decision tree is created starting with the most relevant feature. This method is computationally efficient because it tests at most a number of cases equal to the number of features. The danger is that if none of the features is significantly better than the others then the method may fail to find a good subset, by contrast if there is a strongly relevant feature the method gives reasonably good results.

The *One-Rule* algorithm [18] ranks the attributes according to the error rate. This method is sensibly affected by overfitting.

The *Relief* algorithm uses a nearest-neighbor approach [20]. The algorithm updates iteratively a *relevance vector* of length equal to the number of features, initially set to zero. In a two-class problem, for a randomly chosen sample, one nearest point is chosen in the same class and one in the opposite class. The squared component distances of these two closest examples are component-wise subtracted from (or added to) the relevance vector depending on whether the closest example was in the same (or different) class. This procedure is repeated for m (a given parameter) times, and those features whose relevance weight, thus computed, are above a certain threshold are selected. An improvement of the basic algorithm is *Relief-F* [23] that uses M, instead of just one, nearest hits and ensures greater robustness of the algorithm against noise.

The development in scientific research currently focuses on topics related to *data explosion* phenomenon such as FS for ultrahigh dimensional data [30], and multi-source FS [38]. In [13] there is a case study on feature selection techniques applied to geographic information systems and geospatial decision support, an application domain where the growing availability of data poses several challenges along with important perspectives. There is a growing interest to consider the FS as something more than just a routine to improve machine learning accuracy; the FR model is by itself a knowledge model holding important semantic aspects of the information environment. There have been attempts to further enrich the concept of relevant feature with semantic meanings, such as the contribution of a feature to the knowledge of the physical process underlying the generation of the data. The usefulness of the FR in selecting the variables for modelling dynamic systems has been studied in [5]. A *causal feature selection* is proposed in [17], where the FS is driven by the detection of cause-effect relationships observed in time. This kind of selection process explicitly associates the concept of relevant feature with the concept of control variable. One step forward to the contribution of FS to the modelling of a real system is provided by [12, 33], which in the selection process take into account the interaction of features, acknowledging the fact that features exhibit group properties that cannot be detected on individual features, as they were actual components of a system. More recently there have been attempts to integrate the FS with preexisting basis of knowledge such as ontologies and association rules [7].

4.2.3 A Multiple-Challenge Case Study for Feature Ranking

The issues identified in the previous sections have been dealt in the scientific literature in separate ways, but in reality they constitute a complex of challenges to be addressed in an integrated manner, especially when pursuing goals of efficiency and effectiveness as it is in real applications and in production environments. The problem on which we focus our interest is to obtain a new model for ranking features which combines effective FE methods to a representation model that is humanly understandable and can be integrated in domain knowledge. It is also an objective to explore how generalizable is the efficacy of this new method and how it benefits from a modular architecture that allows to choose between alternative methods of feature extraction depending on restrictions imposed by specific applications. In order to compare the quality of the new model, and its possible variants, to the classical methods it is necessary to identify suitable performance metrics and a benchmarking methodology that uses reference datasets. At the same time there has to be explored the possibility to obtain cost-benefit functions of the features for use in decision-making.

4.3 Focus on Feature Extraction Based Ranking

4.3.1 Linear Models

Many known techniques of Feature Extraction (FE) differ in the principle underlying the detection of an optimal new set of features. However, all of them show an underlying unity in the calculation of geometric transformation, algebraically expressed as projection (or mapping) matrix.

In *Linear Discriminant Analysis* (LDA), where a linear separability of classes is assumed, the principle underlying the detection of a new feature is that of maximising the ratio of the *between-class variance* to the *within-class variance* on this feature. Therefore a set of new features are obtained by maximizing the ratio of the between-class covariance matrix $\mathbf{S_b}$ to the within-class covariance matrix $\mathbf{S_w}$. The projection matrix is the eigenvector matrix $\mathbf{U}$ obtained by solving the generalized eigenvalue problem: $\mathbf{S_b} \cdot \mathbf{U} = \mathbf{S_w} \cdot \mathbf{U} \cdot \Lambda$, where Λ is a diagonal matrix whose entries are the *eigenvalues* of $\mathbf{U}$. Each eigenvalue λ_i measures the relative capability of each new feature $\mathbf{u_i}$ of separating classes.

A limitation of the classic LDA algorithm is that both $\mathbf{S_w}$ and $\mathbf{S_b}$ matrices must be non-singular in order to preserve the orthonormality of the mapping. For this reason several variants of the classic algorithm have been proposed in order to overcome the singularity problem. In particular in this work, we consider the *Orthogonal Linear Discriminant Analysis* (*OLDA*) algorithm [37]. This algorithm uses *Singular Value Decomposition* to obtain a non-singular approximation of $\mathbf{S_w}^{-1} \cdot \mathbf{S_b}$. When $\mathbf{S_w}$ and $\mathbf{S_b}$ matrices are non-singular, OLDA and classic LDA give identical results.

4.3.2 Feature Extraction Based on Decision Boundary

Another family of FE techniques is based on the properties of decision border [10]. Classes are statistically characterized by the *class-conditional probability density function* (cpdf) $p_{X|Y}(\mathbf{x}|y_i)$, where the continuous random vector $\boldsymbol{X}$ takes values in $\mathscr{R}^N$ and the discrete random variable Y takes value in y. The cumulative probability density function of the random vector $\boldsymbol{X}$ is:

$$p_X(\mathbf{x}) = \sum_{i=1}^{C} P_Y(y_i) p_{X|Y}(\mathbf{x}|y_i), \tag{4.1}$$

where $P_Y(y_i)$ is the a-priori probability of class y_i.

Therefore, a classification or decision rule is a mapping $\Psi : \mathscr{R}^N \rightarrow Y$ that assigns a class label to data on the basis of the observation of its feature vector. A classification rule determines a partition of the feature space in C *decision regions* $D_1, \ldots, D_C$ such that $D_i = \{\mathbf{x} \in \mathscr{R}^N \mid \Psi(\mathbf{x}) = y_i\}$. The boundary separating decision regions is called the *decision boundary*. Figure 4.1 illustrates an example of decision rule for two Gaussian classes (symbolized by '$*$' and 'o'). The straight line represents the decision boundary: all points at the left of it are assigned by the decision rule to '$*$' class, and those at the right to 'o' class.

Among all possible classification rules, the rule achieving the minimum *error probability*

$$\varepsilon = \int \sum_{y_i \neq \Psi(\mathbf{x})} p(\mathbf{x}|y_i) P(y_i) d\mathbf{x} \tag{4.2}$$

is the Bayes rule $\Psi_B(\mathbf{x}) = arg\, MAX_{y_i}[p(\mathbf{x}|y_i)P(y_i)]$. The corresponding decision boundary is consistently called *Bayes boundary*, which is the theoretically optimal solution that every classification method aims to achieve.

The geometry of the decision boundary has been used in the discriminative feature extraction approach known as *Decision Boundary Feature Extraction* (DBFE) [25]

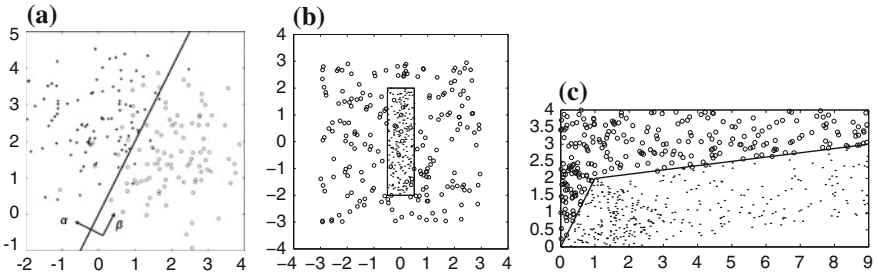


Fig. 4.1 Examples of two-classes classification problems in a 2-dimensional space. **a** Linear boundary. α and β represent the informative direction and the redundant direction respectively, **b** Closed boundary, **c** Piecewise linear boundary

to recognize those informative features allowing to achieve the same classification accuracy as in the original space. The basic idea of DBFE is that moving along the direction of the decision boundary, the classification of each observation will remain unchanged (see Fig. 4.1a). Hence, the direction of the decision boundary is redundant. In contrast, while moving along the direction normal to the decision boundary the classification changes, hence it represents an informative direction. Moreover, the effectiveness of a direction is directly proportional to the area of decision boundary with the same normal vector. To discuss this statement, consider Fig. 4.1b. There, the border is a rectangle parallel to the axes, so the informative directions defined by normal vectors to the border are the x and y axes themselves. Although both directions are informative, it is simple to see that the x-axis is more important since projecting data on it results in less class overlapping than projecting data on the y-axis.

The idea is formalized by the notion of *Effective Decision Boundary Feature Matrix* (EDBFM):

$$\Sigma_{EDBFM} = \frac{1}{\int_{S'} p(\mathbf{x}) d\mathbf{x}} \int_{S'} \mathbf{N}^T(\mathbf{x}) \mathbf{N}(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}, \tag{4.3}$$

where $\mathbf{N}(\mathbf{x})$ is the normal vector at a point $\mathbf{x}$, $\mathbf{N}^T(\mathbf{x})$ denotes the transposed normal vector and S' is the portion of decision boundary containing most of the training data (the effective decision boundary). It has been proved [25] that:

- the rank of the EDBFM represents the *intrinsic discriminant dimension*, that is the minimum number of feature vectors needed to achieve the same Bayes error probability as in the original space;
- the eigenvectors of EDBFM corresponding to nonzero eigenvalues are the necessary feature vectors.

In order to construct a Bayes decision border, in [25] there has been proposed *SVM Decision Boundary Analysis*, a method that combines DBFE principle and Support Vector Machine algorithm. In [14] the use of *Analytical Decision Boundary Feature Extraction* (ADBFE) is introduced, where the normal vectors are calculated analytically from the equations of the decision border. All methods produce an EDBFM that represents a data projection matrix onto a new feature space.

4.4 Feature Ranking Based on Effective Decision Boundary Feature Matrix

4.4.1 Geometric Considerations

As it has been introduced in previous sections, it is desirable to obtain a ranking of real features on the basis of information contained in EDBFM. The idea is intuitively

explained by referring again to the examples in Fig. 4.1. Let us consider decision boundaries formed by a unique line, like line β in Fig. 4.1a. In these cases none of the features is redundant, however it is apparent that the relevance of a feature can be stated in terms of the line slope. In order to apply the DBFE method, let us observe that the decision boundary has the form $y = mx + k$, hence the normal vector is $N = [m, -1]$. The calculus of equation (4.3) is straightforward since the normal vector is constant along S' and the equation becomes:

$$\Sigma_{EDBFM} = \frac{\mathbf{N}^T \mathbf{N} \int_{S'} p(\mathbf{x}) d\mathbf{x}}{\int_{S'} p(\mathbf{x}) d\mathbf{x}} = \mathbf{N}^T \mathbf{N} = \begin{pmatrix} m^2 & -m \\ -m & 1 \end{pmatrix}. \tag{4.4}$$

Eigenvalues and related eigenvectors are $\lambda_1 = 0$, $\lambda_2 = m^2 + 1$, $v_1 = [1, m]$, $v_2 = [-m, 1]$, and only the second eigenvector v_2 is the informative direction. In this case the eigenvector components define the relevance of the real features. For instance, when $m = 0$ (boundary parallel to the x-axis) the only informative real feature is the y-axis, when $m = 1$ (boundary $y = x$) both features are equally important, finally as $m \to \infty$ (boundary tends to the y-axis) the relevance of x-axis grows. As a second case, let us consider the border in Fig. 4.1b. In this case, *cpdfs* are taken constant along the boundary and EDBFM is

$$\Sigma_{EDBFM} = \begin{pmatrix} 8 & 0 \\ 0 & 2 \end{pmatrix},$$

with $\lambda_1 = 8,\ \lambda_2 = 2, v_1 = [1, 0],\ v_2 = [0, 1]$. This case is somewhat complementary to the former: now, since new features coincide with the real ones, the relevance of the latter is fully expressed by eigenvalues. From the analysis of these two cases we can derive that in the DBFE approach the eigenvector components represent the weight of every real feature locally to the new feature, whereas the eigenvalues represent the discriminative power of each new feature. Hence we can combine these two characteristics in order to define a global ranking of the real features as it is in the objective of the present work. Firstly eigenvectors are weighted by multiplying them by the respective eigenvalues, and then the corresponding components of weighed eigenvectors are summed (in the absolute values). Resulting values are the individual contributions (or weights) of every real feature into the transformation, and represent the discriminative power of each real feature and its relative position in a *rank model*.

Formally, let $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_N}$ be the eigenvectors of the EDBFM matrix, λ_1, λ_2, …, λ_N the corresponding eigenvalues, and u_{ij} the jth component of the eigenvector $\mathbf{u_i}$. The weights of real features are computed as follows:

$$w_j = \sum_{i=1}^{N} \lambda_i |u_{ij}|, j = 1, \ldots, N, \tag{4.5}$$

$w_j > w_k \Rightarrow$ feature f_j is more important than f_k.

As a numeric example, let us consider Fig. 4.1c. The equation of the border is $y = 2x$ for $x \in [0, 1]$, $y = x/8 + 15/8$ for $x \in [1, 9]$. The *cpdfs* are taken constant along the boundary. It turns out that

$$\Sigma_{EDBFM} = \begin{pmatrix} 1.913 & -1.887 \\ -1.887 & 8.385 \end{pmatrix},$$

$\lambda_1 = 1.4$, $\lambda_2 = 8.89$, $v_1 = [0.965, 0.261]$, $v_2 = [-0.261, 0.965]$. The ranking method leads to the following weights: $w_1 = 3.68$, $w_2 = 8.95$, hence the real feature y turns out to be more discriminant than x as the figure suggests, since the first piece of boundary is shorter than the second one which is almost parallel to the x-axis.

4.4.2 The Algorithm

The presented method is based on the calculus of the EDBFM, which in turn needs the knowledge of the decision boundary. In order to apply it to real cases, where the decision boundary, as well as *cpdfs* are typically unknown, non-parametric approaches will be considered. In non-parametric approaches we are given a set of instances of the true phenomenon (training data) only, and no assumption on the form of *cpdfs* is made. In this work we propose the use of *Labeled Vector Quantizer* (LVQ) architectures and the *Bayes Vector Quantizer* (BVQ) learning algorithm. The reason for the choice of BVQ is twofold: (1) it has demonstrated to drive an LVQ toward a (locally) optimal approximation of the Bayes boundary [10]; (2) the approximation is piecewise linear, thus simplifying the calculus of the normal vectors.

An Euclidean nearest neighbor Vector Quantizer (VQ) of dimension N and order Q is a function $\Omega : \mathscr{R}^N \rightarrow \mathscr{M}$, $\mathscr{M} = \{\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_Q\}$, $\mathbf{m}_i \in \mathscr{R}^N$, $\mathbf{m}_i \neq \mathbf{m}_j$, which defines a partition of $\mathscr{R}^N$ into Q regions $\mathscr{V}_1, \mathscr{V}_2, \ldots, \mathscr{V}_Q$, such that

$$\mathscr{V}_i = \{\mathbf{x} \in \mathscr{R}^N : \| \mathbf{x} - \mathbf{m}_i \|^2 < \| \mathbf{x} - \mathbf{m}_j \|^2,\ j \neq i\}. \tag{4.6}$$

Elements of $\mathscr{M}$ are called *code vectors*. The region $\mathscr{V}_i$ defined by (4.6) is called the *Voronoi region* of the code vector $\mathbf{m}_i$. Note that the Voronoi region is completely defined by $\mathscr{M}$. In particular, the boundary of Voronoi region $\mathscr{V}_i$ is defined by the intersection of a finite set of hyperplanes $\mathscr{S}_{i,j}$ with equation

$$(\mathbf{m}_i - \mathbf{m}_j) \cdot (\mathbf{x} - \frac{\mathbf{m}_i + \mathbf{m}_j}{2}) = 0,$$

where $\mathbf{m}_j$ is a neighbor code vector to $\mathbf{m}_i$. The definition of normal vectors to these hyperplanes is thus straightforward and it is $\mathbf{N}_{ij} = \mathbf{m}_i - \mathbf{m}_j$ (see Fig. 4.2).

By associating with each code vector a class we can define a decision rule. A Labeled Vector Quantizer (LVQ) is a pair $LVQ =< \Omega, \mathscr{L} >$, where $\Omega : \mathscr{R}^N \rightarrow \mathscr{M}$ is a vector quantizer, and $\mathscr{L} : \mathscr{M} \rightarrow \dagger$ is a labeling function, assigning to each

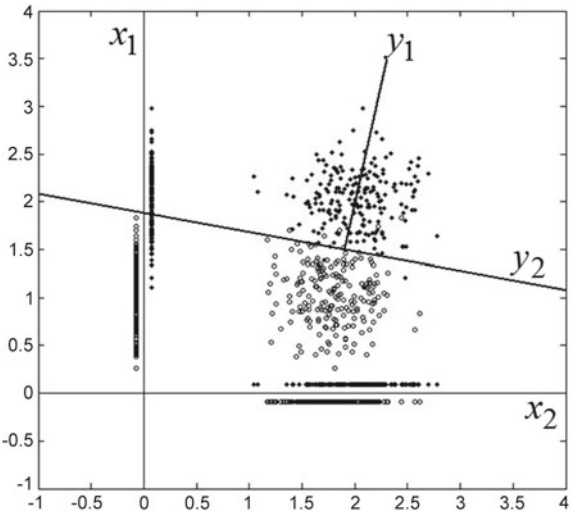


Fig. 4.2 A piece of true decision boundary, its linear approximation and the local discriminative direction $\mathbf{N}_{ij} = \mathbf{m}_i - \mathbf{m}_j$

code vector in $\mathscr{M}$ a class label. The classification rule associated with an LVQ is: $\Psi_{LVQ} : \mathscr{R}^N \rightarrow y, \mathbf{x} \mapsto \mathscr{L}(\Omega(\mathbf{x}))$.

Note the Nearest Neighbor nature of this classification rule: each vector in $\mathscr{R}^N$ is assigned to the same class as its nearest code vector. Thus, decision regions are defined by the union of Voronoi regions of code vectors with the same label. Note also that the decision boundary is defined only by those hyperplanes $\mathscr{S}_{i,j}$ such that $\mathbf{m}_i$ and $\mathbf{m}_j$ have different labels.

An LVQ can be trained to find an approximation of the Bayes boundary. LVQ training algorithms have been originally proposed by Kohonen [22]. Here we use a more recent algorithm known as Bayes VQ (BVQ), formally defined as a gradient descent algorithm for the minimization of the error probability. It strongly resembles Kohonen's LVQ2.1, however, formal derivation introduces also some modifications that improve performances and robustness. The BVQ algorithm is an iterative punishing-rewarding adaptation schema. At each iteration, the algorithm considers a sample randomly picked from the training set. If the sample turns out to fall "on" the decision boundary, then the position of the two code vectors determining the boundary is updated, moving the code vector with the same label of the sample towards the sample itself and moving away that with a different label. Since the decision boundary is a null measure subspace of the feature space, we have zero probability to get samples falling exactly on it. Thus, an approximation of the decision boundary is made, considering those samples falling close to it. Due to lack of space we cannot report the BVQ algorithm here. The algorithm is described in [9].

Having a trained LVQ, the calculus of the feature rank is straightforward and is given by the following BVQ-based Feature Ranking (BVQ-FR) Algorithm 1.

Algorithm 1 BVQ-FR algorithm

1: Train the LVQ $\{(\mathbf{m}_1, l_1), \ldots, (\mathbf{m}_Q, l_Q)\}$, $\mathbf{m}_i \in \mathscr{R}^N$, $l_i \in y$ by using the BVQ algorithm;
2: Set the elements of the matrix Σ_{BVQFM} to 0;
3: $w_{tot} = 0$;
4: For each training sample t_k
    1: Find the two code vectors m_i, m_j nearest to t_k;
    2: If $l_i \neq l_j$ and t_k falls at a distance less than Δ from the border S_{ij} then
        1: Calculate the unit normal vector to the decision boundary as: $\mathbf{N}_{ij} = \frac{(\mathbf{m}_i - \mathbf{m}_j)}{\|\mathbf{m}_i - \mathbf{m}_j\|}$;
        2: $\Sigma_{BVQFM} = \Sigma_{BVQFM} + \mathbf{N}_{ij}^T * \mathbf{N}_{ij}$;
        3: $w_{tot} = w_{tot} + 1$;
5: $\Sigma_{BVQFM} = \frac{\Sigma_{BVQFM}}{w_{tot}}$;
6: Calculate eigenvectors $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_N}$ and related eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_N$ of Σ_{BVQFM};
7: Set $\mathbf{W} = \sum_{z=1}^{N} \lambda_i |\mathbf{u_i}|$;
8: Sort features with respect to $\mathbf{W}$ components.

The core of the BVQ-FR algorithm is at point 4. There, finding the two nearest code vectors to each training sample allows us to define the effective decision boundary of the LVQ. As a matter of fact, testing whether labels are different guarantees that the piece of Voronoi boundary S_{ij} is actually a part of the decision boundary. Secondly, incrementing the Σ_{BVQFM} each time a pair of code vectors is selected, allows to weight the normal vector $\mathbf{N}_{ij}$ by the number of samples falling at a distance less than Δ from S_{ij}. It can be proved that this is equivalent to a Parzen estimate of the integral $\int_{S_{ij}} p(\mathbf{x})$, while the final value of w_{tot} represents the Parzen estimate of $\int_{S'} p(\mathbf{x})$ in Eq. (4.3) [10].

It should be noted that the algorithm BVQ-FR can be transformed by replacing BVQ with other FE algorithm that produces a transformation matrix EDBFM-like. For example there can be used OLDA, SVM and ADBFE algorithms. In the next section an experimental comparison between these alternatives will be made.

4.5 Experiments

4.5.1 Experimental Setting

This section is devoted to experimental evaluation of the EDBFM-based feature weighting method. In particular in the present subsection we propose a synthetic experiment which allows us to illustrate the properties of the method. We also describe both the experimental procedure and the evaluation criteria that will be used for all subsequent experiments. In the next subsection various implementations of the method will be tested over real-world datasets and compared with well-known feature weighting algorithms.

As synthetic experiment we draw a dataset from a 22 dimensions two-class problem. The first two dimensions are drawn from the classical XOR problem, while the remaining 20 dimensions are drawn from the normal distribution. The first two dimensions are useful to classify the two classes (i.e. informative dimensions), while the remaining dimensions are noise. The dataset contains 1,000 samples equally distributed over the two classes. In this experiment, as well as in all experiments of the following subsection, we followed a 10-fold cross-validation procedure: in each fold the 90 % of the samples are used to build the EDBFM matrix and to weight the original features; the remaining samples are used to evaluate the performance of the method. In particular, for each fold a weight model is calculated on an incrementing number of features taken in the rank order from the test set to extract a projection along the first informative features. Hence we firstly obtain two datasets with the most important feature, then two datasets with the first two most important features, and so forth until the full-dimensional datasets (i.e. the original ones) are returned. For each of these pairs of datasets the Nearest Neighbor algorithm is used to estimate the accuracy. After the tenth fold repetition, the weights and the accuracies are averaged by rank, and curves are built, which represent the average accuracy that the method achieves over all folds as a function of the most informative features.

The experimental work-flow is depicted in Fig. 4.3a, it consists of two phases: first the appliance of the EDBFM based ranking method to the multivariate dataset in the filter mode of [26], and then the validation procedure. The process is sketched in the following pseudocode.

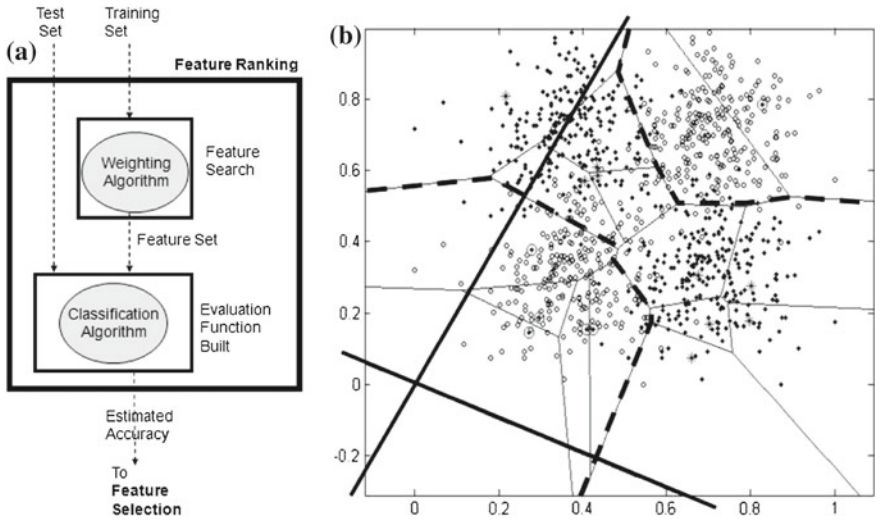


Fig. 4.3 **a** General work-flow of feature selection techniques. **b** Example of two classes classification problems. *Piecewise lines* represent the approximation of the Bayes boundary found by BVQ. y_1 and y_2 represent the two most important extracted features

Algorithm 2 *First phase*: a FE algorithm (BVQ in this example) is applied to the training set, and then the feature ranking algorithm is executed

1: Let $X = \{x_1, x_2, ..., x_m\}$ be the m-dimensional normalized dataset.
2: Apply the BVQ algorithm to X. Let $Y = \{y_1, y_2, ..., y_n\}$ be the extracted eigenfeatures.
3: Compute the contributive weight w_i of each feature x_i to the eigenfeatures of Y.
4: Sort the features of X such that $x_a < x_b$ if $w_a < w_b$. Let $X^s = \{x_1^s, x_2^s, ..., x_m^s\}$ be the sorted dataset and m the rank index.

Algorithm 3 *Second phase*: on an incrementing number of features, taken in the rank order from the test set, the 1NN classification process is run and the accuracy calculated

1: The dataset X^s is input.
2: Apply 1NN to whole X^s, let A_m be the returned accuracy.
3: For rank i=1 to m (where $m = 22$ for this dataset):
- let $X_i^s = \{x_1^s, x_2^s, ..., x_i^s\}$ be a subset of X^s with selected features up to rank i.
- compute accuracy A_i using 1NN with 10-fold cross-validation.

For the first fold, the decision boundary depicted by BVQ is reported in Fig. 4.3b, altogether with features extracted on the basis of the DBFE method.

The *BVQ* setting: Optimal values for Δ and local region r have been found by a manually conducted search assuming the classification error rate as objective function. The parameters were fixed to $\Delta = 0.4$ and $r = 0.5$; 16 code vectors have been detected. The choice of the classification algorithm is unimportant to our purpose since we are interested only in study of the relative performance of ranking algorithms. The *1NN* is a non-parametric classifier among the simplest of all machine learning algorithms, the object is simply assigned to the class of its nearest neighbour on the basis of the Euclidean distance, it does not require settings. In [21] the 1*NN* classifier is indicated as a convenient algorithm to build the evaluation function, since it appears to always provide a reasonable classification performance in most applications.

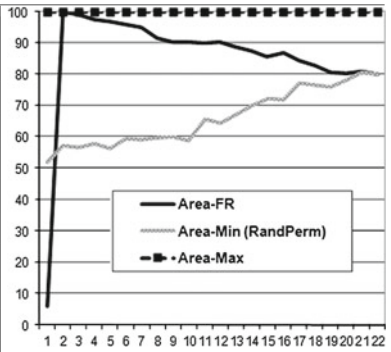
For this experiment the resulting accuracies, in the order they were calculated, are reported in Table 4.1 and plotted aside. The curve shows a steep rise which expresses the high contribution to classification accuracy by the two highest rank features. Beyond a critical point, which in this example occurs on the second feature, the curve tends to decrease because irrelevant features (low rank) are added, which only cause curse of dimensionality. By a way of comparison, a random sorting of features has been used, to which the same validation procedure is applied. The fifth column of Table 4.1 and the corresponding plot represent the average accuracy achieved by 20 different 1NN classifiers, where the features are selected according to 20 different ranks obtained by means of trivial random permutations.

As a figure of merit to characterize the performance of the ranking method we define an empirical *Performance Index* ϕ:

$$\textit{Performance Index}(\phi) = \frac{\text{AreaFR} - \text{AreaRP}}{\text{AreaMax} - \text{AreaRP}} \tag{4.7}$$

Table 4.1 The features (first column) sorted by weight (second column); cumulative percentage of weight (third column); the accuracy, by subset, of EDBFM method (fourth column); the accuracy, by subset, on a random weight model (fifth column)

Rank	Weight	Weight %Cum.	1NN Acc.% Cum. Norm (BVQ-FR)	1NN Acc.% Cum. Norm (rand. rank)
Feat. 1	0.557	27.41	6.08	51.79
1 to 2	0.461	50.10	100	57.27
1 to 3	0.081	54.07	99.09	56.57
1 to 4	0.075	57.76	97.40	57.82
1 to 5	0.063	60.82	95.94	56.26
1 to 6	0.062	63.88	95.15	59.22
1 to 7	0.058	66.74	91.54	59.16
1 to 8	0.057	69.53	90.30	59.54
1 to 9	0.055	72.18	90.41	58.88
1 to 10	0.054	74.83	88.61	58.66
…	…	…	…	
1 to 22	0.035	100.00	79.93	79.93



The accuracies of fourth and fifth columns, normalized to 100 %, are also plotted aside

where *AreaFR* is the area underneath the accuracy curve relative to BVQ-FR, obtained by summing the accuracy values at each feature subset, namely the accuracy value in the BVQ-FR column of Table 4.1. Analogously *AreaRP* is the area underneath the curve obtained by random permutation of features. The *AreaMax* is the area underneath a theoretical curve reaching the 100 % possible accuracy with the top rank feature, thereafter remaining constant up to full dataset. The *AreaFR* is expected to be geometrically bounded between the other two curves, mathematically $0 \leq \phi \leq 1$, where ϕ represents a *relative area*. When AreaFR approximates AreaMax , $\phi = 1$, the ranking model approximates an ideal order of the features, where the first feature is the most significant and contains all the weight to discriminate between classes. Conversely, when AreaFR approximates AreaRP, $\phi = 0$, the ranking model approximates to a random ordering of the features and is therefore useless.

4.5.2 Benchmarking the EDBFM Ranking Method

In this section the EDBFM ranking method is tested on complex and real world datasets and the rank models are compared to other methods. Testing includes two phases:

- studying EDBFM performance when different FE algorithms are included;
- comparing EDBFM ranking and heuristic methods.

As data testbed of the experiments, 13 multivariate datasets have been considered. Eight of these datasets (*Heart, HeartStat, Australian, Ionosphere, Waveform, Segment, CoverType, Letter*) have been drawn from the UCI repository [31], selected for their large number of instances, classes and features as it is appropriate when testing ranking algorithms. Five more datasets (*Urban, Wildfire, Landslide, Corine, Gottigen*) have been extracted from large geographic data collections. These datasets, which include both discrete and continuous variables, are heterogeneous collections of data, excellent to challenge the selective capability of our method and to highlight the properties of the ranking model. The datasets: *Urban, Wildfire, Landslide, Corine* originated from the same data collection, they differ from each other by a different feature chosen as class attribute. *Urban, Wildfire* and *Landslide* have balanced classes, namely in these datasets all classes are represented by an equal number of instances. The geographic dataset named *Gottingen* comes from a different collection [4], its features correspond to Earth observation imagery from satellite on different wavelength band. The characteristics of all the datasets are resumed in Table 4.2, where the datasets are sorted by number of classes, then by number of features, and by number of instances. Such a sorting also represents an increasing complexity of dataset, ranging from a simple two-class perfectly balanced dataset with relatively few instances, such is the Urban, up to the Corine dataset which is a 26 class large dataset. All datasets have gone through a common preprocessing step where each feature has been normalized in the range [0; 1], to give equal importance to each feature during learning.

The first set of experiments aims to highlight how the FR algorithm performance varies when different FE built-in algorithms are used. As already mentioned, the algorithm BVQ-FR can be transformed by changing the FE algorithm,

Table 4.2 Testbed datasets

Origin	Dataset name	# Classes	# Features	# Instances
UCI	HeartStat	2	13	270
UCI	Heart	2	13	293
UCI	Australian	2	14	690
GIS	Urban	2	18	3,972
GIS	Wildfires	2	18	5,359
GIS	Landslides	2	18	23,663
UCI	Ionosphere	2	34	351
UCI	Waveform	3	40	5,000
UCI	CoverType	7	12	58,104
UCI	Segment	7	19	2,310
GIS	Gottigen	14	8	28,083
UCI	Letter	26	16	20,000
GIS	Corine	26	18	48,379

Datasets are sorted by *number of classes*, by *number of features*, and finally by *number of instances*

e.g. using *OLDA, SVM, ADB*. In subsequent experiments we will denote these variants respectively with the acronyms *OLDA-FR, SVM-FR, ADBFE-FR*. These algorithms, along with BVQ-FR, will be tested on the datasets listed above, following the experimental procedure described in the previous section. Notice in the parameter setting for BVQ-FR, the number of code vectors has been set to a multiple of the number of classes in the dataset, with 200000 BVQ iterations, whereas Δ and r come from a manual refinement in three steps. In SVM-FR, we employ a *Gaussian radial basis kernel* to train the SVM, and we set r to 0.2.

In the Fig. 4.4, the accuracy curves are grouped by dataset to compare the performance of EDBFM Ranking algorithms. For each dataset the accuracy curve obtained by means of random permutation of features is also displayed. Notice the curves of EDBFM ranking are always located above the random ranking curve, that reveals the general efficacy of EDBFM ranking. The qualitative comparison between the curves is difficult because of the irregular pattern and their overlaps. The Performance Index ϕ is of help in the analysis. In Table 4.3 ϕ calculated for each curve is shown. Note that missing values in the Table 4.3 are due to the impossibility to perform computationally expensive algorithms, such as SVM and ADBFE, on datasets with large number of classes and instances. We can observe in Table 4.3, where rows are sorted by increasing complexity of the dataset, OLDA-FR and BVQ-FR have, together, a dominance in the values of ϕ when applied to datasets with two classes *Heart-Stat, Heart, Australian, Urban, Wildfire, Landslide* whereas BVQ-FR has a relative dominance on complex datasets *Ionosphere, Waveform, CoverType, Segment, Gottigen, Letter, Corine*. This is due to the fact that BVQ-FR, based on nonparametric model, has a superior performance when working on non-linearly separable classes of objects.

In the second set of experiments we compare the performance of OLDA-FR and BVQ-FR with other ranking methods known in literature such as *Relief, Gain Ratio and One-Rule*. Also heuristic methods calculate a weight for each individual real feature, which allows to rearrange the features by decreasing weights and to submit dataset to the 1NN classification algorithm using the same procedure as for the models based on EDBFM. Accuracies calculated in the previous experiment for OLDA-FR and BVQ-FR are now compared with accuracies obtained using the Relief, Gain Ratio and One Rule. The accuracy curves gathered by dataset are shown in Fig. 4.5. The criterion of comparison of curves is the same than in the previous experiment.

The general picture of performances is rather complex, but trends are evidenced by the analysis of the index ϕ. For each dataset the best ranker is highlighted in the Table 4.4; there are also reported some statistics of the Performance Index: the mean value of ϕ for BVQ-FR is the highest, and the variance has the lowest value. The statistics indicate a low dispersion of ϕ for BVQ-FR algorithm, that reveals a relatively stable behaviour in comparison to Relief, GainRatio and One-Rule rankers and OLDA-FR as well.

In the star plot (see Fig. 4.6) the index values are shown as a radial line from a common centre point. Points corresponding to the same algorithm are connected by a common-style line. In the clockwise direction the datasets are sorted by increasing complexity. Notice in the part of the diagram where two-classes datasets are

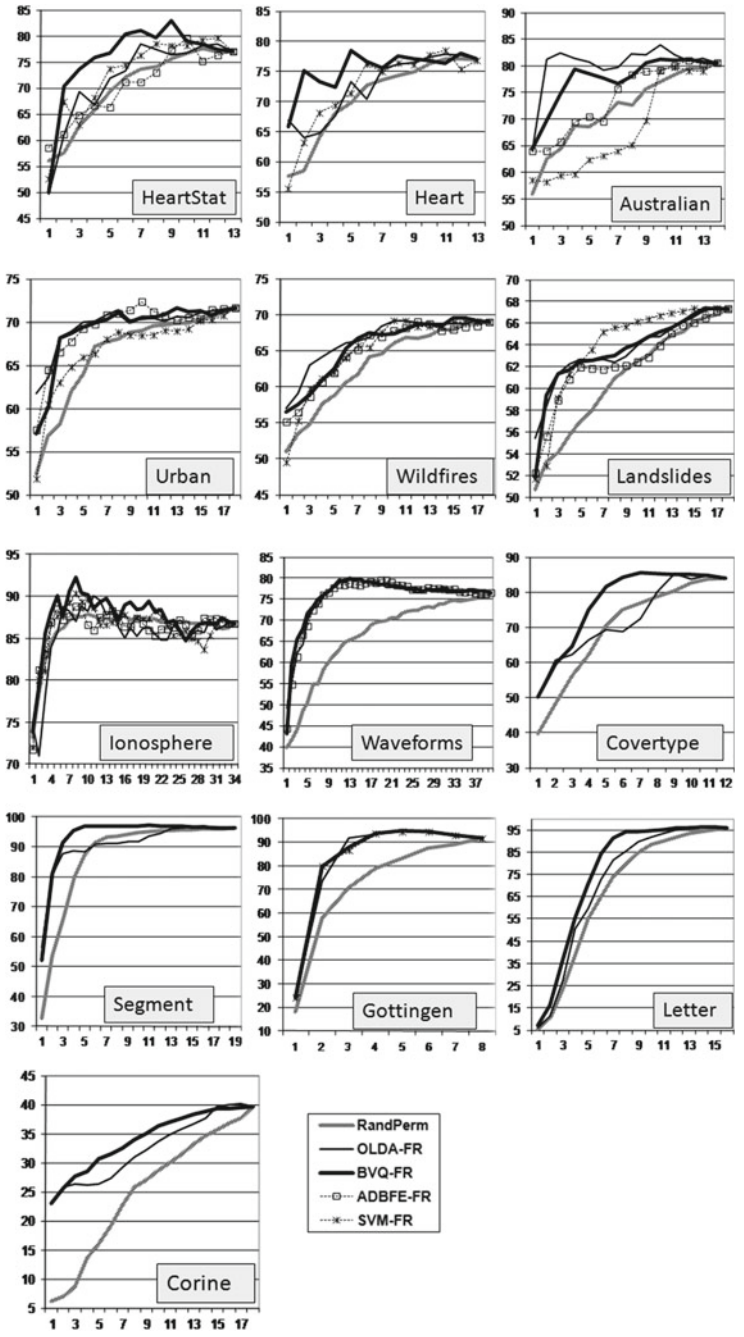


Fig. 4.4 Feature Ranking experiments. Comparing the performance of FE filter algorithms. On the *horizontal* axis the features sorted by rank and in *vertical* the percentage of accuracy

Table 4.3 EDBFM ranking: comparison of filter FE algorithms

	OLDA-FR (ϕ)	BVQ-FR (ϕ)	ADBFE-FR (ϕ)	SVM-FR (ϕ)
HeartStat	0.116	*0.411*	0.015	0.188
Heart	0.187	*0.471*	–	0.144
Australian	*0.685*	0.463	0.180	−0.298
Urban	*0.543*	0.465	0.502	0.075
Wildfires	*0.474*	0.354	0.277	0.265
Landslides	0.400	0.388	0.224	*0.446*
Ionosphere	−0.096	*0.207*	0.019	0.017
Waveform	0.651	*0.670*	0.641	–
CoverType	0.125	*0.373*	–	–
Segment	0.348	*0.595*	–	–
Gottigen	0.445	*0.456*	–	0.447
Letter	0.133	*0.287*	–	–
Corine	0.484	*0.592*	–	–

The *Performance Index* (ϕ) for each of the accuracy curves in Fig. 4.4

concentrated, from Heart to Ionosphere, there is an evident superiority of One Rule over the other rankers. By contrast where more complex datasets are concentrated, from Waveform to Corine, BVQ-FR tends to outperform the other rankers whose performance decreases more rapidly as the dataset complexity increases.

Another comparative indicator of performance is the number of features needed to reach 90 % of total accuracy, see Table 4.5. This indicator represents a relative measure of the steepness of the curve; it indicates the ranker's ability to lead to higher accuracies with relatively small subsets. On this indicator BVQ-FR outperforms all other rankers.

Let us observe in more detail a ranking model to highlight its usefulness in supporting cost-benefit informed decision making. In Fig. 4.7 left, for the Wildfire dataset, the curve of accuracy obtained for BVQ-FR is overlaid with the curve of cumulative weights, the horizontal axis represents the features sorted by rank. Notice that the first nine features, which are 50 % of total, represent half the cost of the entire dataset, but detain over 70 % of the total weight and over 98 % of the total accuracy achievable. Analogously, in Fig. 4.7 right, for CoverType dataset, the first feature holds 17 % of the total weight of the features, whereas the first six features (50 % of total features) detain over 70 % of the total weight and over 80 % of the accuracy achievable on the full dataset. If the individual costs of the features are given, it is possible to construct a detailed cost function. As a consequence, it is evident that the proposed methodology can guarantee the best ratio between cost of features acquisition and informative power.

As it was described above, the index ϕ has been used to compare the relative performance of ranking algorithms on a dataset. To assess the overall performance for each algorithm the number of times that the algorithm has had the highest ϕ was counted. These aleatory results, however, require a test of statistical significance

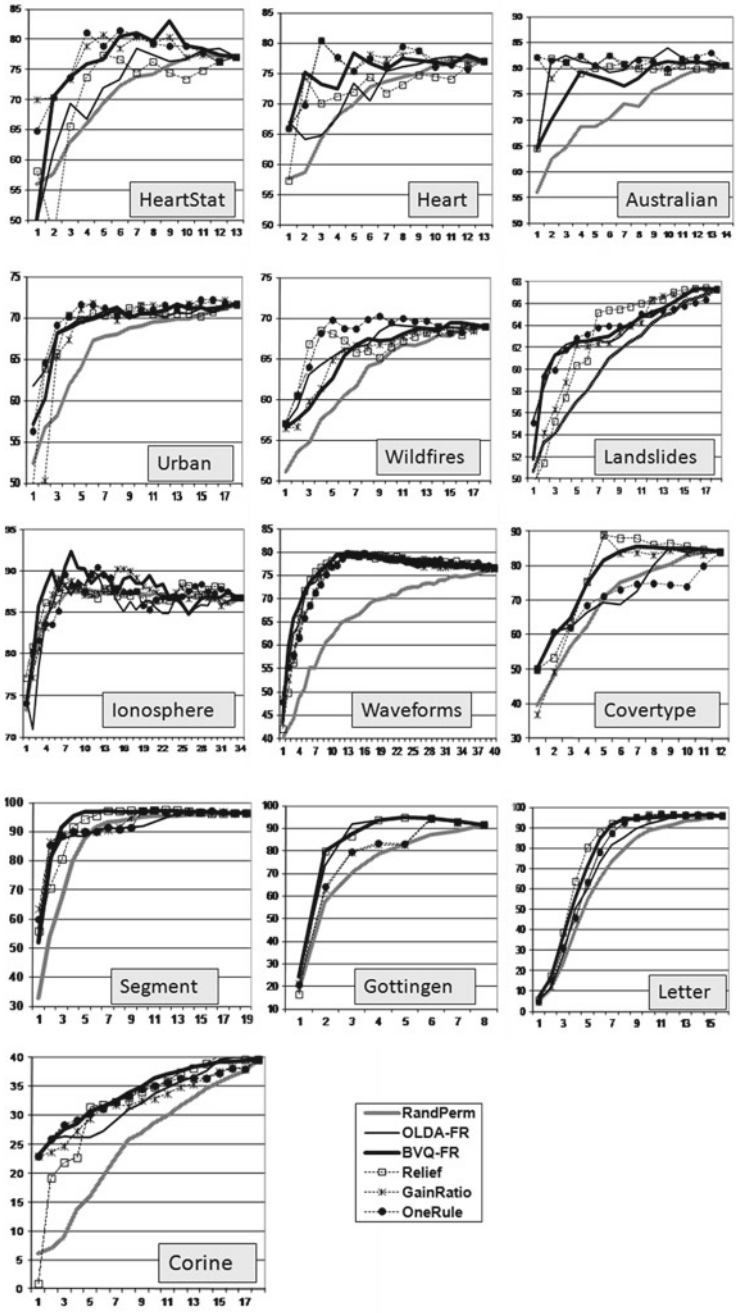


Fig. 4.5 Feature Ranking experiments. Comparing the performance of Ranking Algorithms. On the *horizontal axis* the features are sorted by rank and in *vertical* the percentage of accuracy

Table 4.4 The EDBFM ranking is compared to other methods (Relief, Gain Ratio, OneRule); the *Performance Index* (ϕ) is calculated for each of the accuracy curves in Fig. 4.5

	Goal oriented ranking				
	OLDA-FR (ϕ)	BVQ-FR (ϕ)	Relief (ϕ)	Gain-ratio (ϕ)	One-rule (ϕ)
HeartStat	0.116	0.411	0.059	*0.516*	0.513
Heart	0.187	0.471	0.151	*0.548*	0.521
Australian	0.685	0.463	0.602	0.744	*0.806*
Urban	0.543	0.465	0.408	0.331	*0.609*
Wildfires	0.474	0.354	0.483	0.336	*0.647*
Landslides	0.400	0.388	0.292	0.251	*0.402*
Ionosphere	−0.096	*0.207*	0.083	0.075	0.072
Waveform	0.651	*0.670*	0.654	0.614	0.621
CoverType	0.125	0.373	*0.418*	0.264	0.040
Segment	0.348	*0.595*	0.483	0.501	0.467
Gottigen	0.445	*0.456*	0.407	0.169	0.187
Letter	0.133	0.287	*0.349*	0.198	0.217
Corine	0.484	*0.592*	0.438	0.474	0.545
Mean	0.346	0.441	0.371	0.386	0.434
Variance	0.055	0.016	0.034	0.039	0.056

The two bottom rows are descriptive statistics of the Performance Index computed values

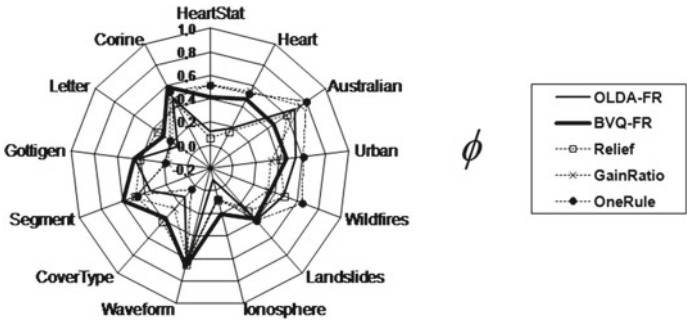


Fig. 4.6 Each piecewise line represents a method of ranking, each *radial line* represents a dataset. Datasets are radially ordered by increasing complexity. The intersections represent the values of performance index ϕ

to support the observations made. The probability of success of an algorithm over another is calculated with the *binomial distribution*, from the count of victories and defeats, or the number of times that the algorithm outperformed the others on the basis of the index of performance. Assuming the *null hypothesis* is that the frequency of success of the two algorithms is the same, with the *two-tailed test* it can be seen how much we deviate from "null hypothesis" assumption.

Table 4.5 Number of features needed to reach 90 % of total accuracy

	Rand rank	Goal oriented ranking				
		OLDA-FR	BVQ-FR	Relief	Gain-ratio	One-rule
HeartStat	9	7	*4*	11	*4*	*4*
Heart	6	5	*2*	8	3	3
Australian	9	*2*	4	*2*	N/D	N/D
Urban	6	*3*	*3*	*3*	*3*	*3*
Wildfires	8	4	6	*3*	5	*3*
Landslides	8	*3*	*3*	6	5	4
Ionosphere	4	4	*3*	*3*	4	4
Waveforms	24	*6*	*6*	*6*	8	8
Covertype	9	9	*5*	*5*	*5*	12
Segment	6	4	*3*	4	*3*	*3*
Gottingen	6	*3*	*3*	*3*	6	6
Letter	10	9	7	*6*	7	7
Corine	16	13	*10*	12	14	12

Dataset sorted by increasing complexity

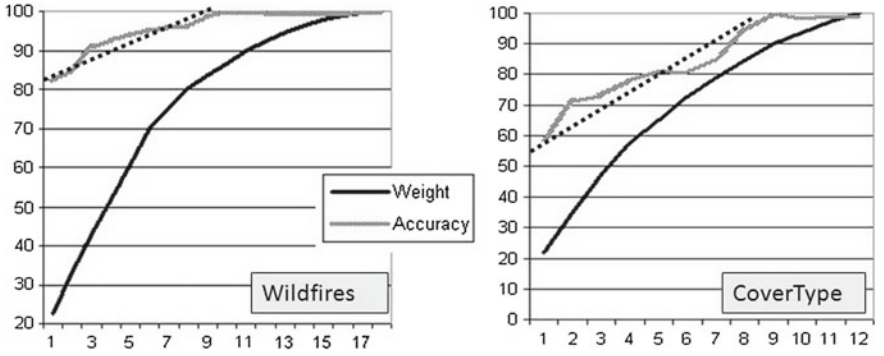


Fig. 4.7 *Performance Indices* cost-benefit of features of EDBFM based ranking. *Left* Wildfire dataset. *Right* CoverType dataset. On the *horizontal* axis the features sorted by rank and in *vertical* the values in percentage normalized to 100 %

In the Table 4.6 (top), the significance test is performed on all ranking experiments. The table does not allow us to assert the superiority of a method over another; pointing out that more experiments are needed. However, we have that BVQ-FR overcomes Relief and Gain Ratio with statistical significance greater than 0.9, while there is condition of parity with One Rule that is expressed by null statistic significance. It is noteworthy that BVQ-FR results have been obtained without stressing the parameters setup of the BVQ algorithm.

Table 4.6 Overall comparison of the five algorithms

	Sign test—all datasets				
ϕ	BVQ-FR	Relief	G.Ratio	Onerule	
	4/9	7/6	6/7	3/10	w/l
OLDA-FR	0.733	0.0	0.0	0.908	P
		10/3	10/3	7/6	w/l
BVQ-FR		0.908	0.908	0.0	P
			8/5	6/7	w/l
Relief			0.419	0.0	P
				5/8	w/l
GainRatio				0.419	P

4.6 Conclusions

This chapter focuses on a novel ranking procedure. We considered that the premise for integrating the feature ranking models into domain knowledge is their representation in terms of real world features. This principle is the fundamental premise of the study conducted which leads to a computational model that is accurate and humanly understandable. A new approach to Feature Ranking (FR) based on features extraction (FE) and properties of the decision border has been discussed. This method uses Effective Decision Boundary Feature Matrix (EDBFM) to measure the relevance of the real world features thus maintaining the readability of the knowledge model extracted. The method has been tested on classification problems and cost-benefit analysis of features. While maintaining the geometric procedure which yields the ranking of features, this method allows to choose between alternative core FE algorithms, such as BVQ, when extracting the EDBFM, that allows to optimize the method application on datasets with different complexity. In particular BVQ-FR has proven to be more effective in applications to dataset of non-linearly separable points. Benchmarking tests, supported by the calculation of index of performance, show that BVQ-FR and OLDA-FR are generally more effective than other solutions. Furthermore, the comparison with known heuristic techniques of ranking confirms the robustness and the superiority of the EDBFM based method on complex dataset.

References

1. Alelyani, S., Liu, H., Wang, L.: The effect of the characteristics of the dataset on the selection stability. In: Proceedings of the 23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI), pp. 970–977 (2011)
2. Arauzo-Azofra, A., Aznarte, A.L., Benitez, J.M.: Empirical study of feature selection methods based on individual feature evaluation for classification problems. Expert Syst. Appl. **37**(3), 8170–8177 (2011)

3. Blum, A., Langley, P.: Selection of relevant features and examples in machine learning. Artif. Intell. **97**(1), 245–271 (1997)
4. Bock, M., Bohner, J., Conrad, O., Kothe, R., Ringler, A.: Saga, system for automated geoscientific analysis. Technical Report Saga Users Group Association, University of Gottingen, http://www.saga-gis.org (2000)
5. Cantu-Paz, E., Newsam, S., Kamath, C.: Feature selection in scientific applications. In: Proceedings of the 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 788–793 (2004)
6. Chandrashekar, G., Sahin, F.: A survey on feature selection methods. Comput. Electr. Eng. **40**(1), 16–28 (2014)
7. Chawla, S.: Feature selection, association rules network and theory building. In: Proceedings of the Fourth Workshop on Feature Selection in Data Mining, pp. 14–21 (2010)
8. Dash, M., Liu, H.: Feature selection for classification. Intell. Data Anal. **97**(11), 131–156 (1997)
9. Diamantini, C., Panti, M.: An efficient and scalable data compression approach to classification. ACM SIGKDD Explor. **2**(2), 54–60 (2000)
10. Diamantini, C., Potena, D.: A study of feature extraction techniques based on decision border estimate. In: Liu, H., Motoda, H. (eds.) Computational Methods of Feature Selection, pp. 109–129. Chapman & Hall/CRC, Boca Raton (2007)
11. Ding, S., Zhu, H., Jia, W., Su, C.: A survey on feature extraction for pattern recognition. Artif. Intell. Rev. **37**(3), 169–180 (2012)
12. Escalante, H.J., Montes, M., Sucar, E.: An energy-based model for feature selection. In: Proceedings of the 2008 IEEE World Congress on Computational Intelligence (WCCI), pp. 1–8 (2008)
13. Gemelli, A., Mancini, A., Diamantini, C., Longhi, S.: GIS to Support Cost-Effective Decisions on Renewable Sources: Applications for Low Temperature Geothermal Energy. Springer, New York (2013)
14. Go, J., Lee, C.: Analytical decision boundary feature extraction for neural networks. In: Proceedings of the IEEE 2000 International Geoscience and Remote Sensing, pp. 3072–3074 (2000)
15. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. **2003**(3), 1157–1182 (2003)
16. Guyon, I., Elisseeff, A.: An introduction to feature extraction. In: Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L. (eds.) Feature Extraction, Foundations and Applications, pp. 1–25. Springer, New York (2006)
17. Guyon, I., Aliferis, C., Elisseeff, A.: Causal feature selection. In: Liu, H., Motoda, H. (eds.) Computational Methods of Feature Selection, pp. 1–40. Chapman and Hall, London (2007)
18. Holte, R.C.: Very simple classification rules perform well on most commonly used datasets. Mach. Learn. **11**(1), 63–90 (1993)
19. John, G.H., Kohavi, R., Pfleger, K.: Irrelevant features and the subset selection problem. In: Proceedings of the 11th International Conference on Machine Learning, pp. 121–129 (1994)
20. Kira, K., Rendell, L.A.: The feature selection problem: traditional methods and a new algorithm. In: Proceedings of the Ninth National Conference on Artificial Intelligence, pp. 129–132 (1992)
21. Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artif. Intell. **97**(1), 273–324 (1997)
22. Kohonen, T.: The self-organizing map. Proc. IEEE **78**(9), 1464–1480 (1990)
23. Kononenko, P.C.: Estimating attributes: analysis and extensions of relief. In: Proceedings of the European Conference on Machine Learning '94, pp. 171–182 (1994)
24. Lee, C., Landgrebe, D.A.: Feature selection based on decision boundaries. In: Proceedings of the IEEE 1991 International in Geoscience and Remote Sensing Symposium—IGARSS, pp. 1471–1474 (1991)
25. Lee, C., Landgrebe, D.A.: Feature extraction based on decision boundaries. IEEE Trans. Pattern Anal. Mach. Intell. **15**(4), 388–400 (1993)
26. Liu, H., Yu, L.: Toward integrating feature selection algorithms for classification and clustering. IEEE Trans. Knowl. Data Eng. **17**(4), 491–502 (2005)

27. Liu, H., Motoda, H.: Less is more. In: Liu, H., Motoda, H. (eds.) Computational Methods of Feature Selection, pp. 3–12. Chapman and Hall, London (2007)
28. Liu, H., Suna, J., Liu, L., Zhang, H.: Feature selection with dynamic mutual information. Pattern Recognit. **42**(7), 1330–1339 (2009)
29. Liu, H., Motoda, H., Setiono, R., Zhao, Z.: Feature selection: an ever evolving frontier in data mining. J. Mach. Learn. Res.- Proc. **10**(1), 4–13 (2010)
30. Monteiro, S.T., Murphy, R.J.: Embedded feature selection of hyperspectral bands with boosted decision trees. In: Proceedings of the IEEE 2011 International in Geoscience and Remote Sensing Symposium, IGARSS, pp. 2361–2364 (2011)
31. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: Repository of machine learning databases. University of California, Technical Report (1998)
32. Quinlan, J.R.: Improved use of continuous attributes in C4.5. J. Artif. Intell. Res. **4**(1), 77–90 (1996)
33. Senoussi, H., Chebel-Morello, B.: A new contextual based feature selection. In: Proceedings of the 2008 International Joint Conference on Neural Networks (IJCNN), pp. 1265–1272 (2008)
34. Sima, C., Attoor, S., Brag-Neto, U., Lowey, J., Suh, E., Dougherty, E.R.: Impact of error estimation on feature selection. Pattern Recognit. **38**(12), 2472–2482 (2005)
35. Singhi, K.S., Liu, H.: Feature subset selection bias for classification learning. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 849–856 (2006)
36. Wang, L., Zhou, N., Chu, F.: A general wrapper approach to selection of class-dependent features. IEEE Trans. Neural Netw. **19**(7), 1267–1278 (2008)
37. Ye, J.: Characterization of a family of algorithms for generalized discriminant analysis on undersampled problems. J. Mach. Learn. Res. **6**, 483–502 (2005)
38. Zhao, Z., Wang, J., Sharma, S., Agarwal, N., Liu, H., Chang, Y.: An integrative approach to identifying biologically relevant genes. In: Proceedings of SIAM International Conference on Data Mining, pp. 838–849 (2010)