Information Theoretic Learning and local modeling for binary and multiclass classification

Iago Porto-Díaz · David Martínez-Rego ·
Amparo Alonso-Betanzos · Oscar Fontenla-Romero

Received: 14 September 2012 / Accepted: 29 September 2012 / Published online: 16 October 2012
© Springer-Verlag Berlin Heidelberg 2012

Abstract In this paper, a learning model for binary and multiclass classification based on local modeling and Information Theoretic Learning (ITL) is described. The training algorithm for the model works on two stages: first, a set of nodes are placed on the frontiers between classes using a modified clustering algorithm based on ITL. Each of these nodes defines a local model. Second, several one-layer neural networks, associated with these local models, are trained to locally classify the points in its proximity. The method is successfully applied to problems with a large amount of instances and high dimension like intrusion detection and microarray gene expression.

Keywords Machine learning · Classification · FVQIT · Information theoretic learning · Local modeling

1 Introduction

Pattern classification in highly non-linear and multimodal problems has been a challenge for machine learning algorithms through the years. Several previous researchers [28] have analyzed the difficulties found when facing these kinds of problems by both classical statistical classifiers (such as Fisher Linear Discriminant [20] and its variations) and machine learning methods (such as artificial neural networks [4] and decision trees like ID3 [46] or C4.5 [47]). Over the last years, more sophisticated models have come out. These models try to mitigate the weaknesses of classical algorithms in order to being able to deal with more complex classification

problems. One of the latest and more well-known approaches are Support Vector Machines (SVMs) [13]. These models convert a complex non-linear, non-separable problem into a linear problem, by means of a transformation to a higher dimensional space.

Most classifiers are global methods. A *global method* attempts to solve a problem by means of adjusting a single model for the whole feature space. However, there exists another approach to the classification problem, the *combination of classifiers* [30]. This is a relatively recent technique that can be considered a meta-algorithm in the sense that it combines a set of component classifiers in order to obtain a more precise and stable model. The two most important strategies to combine classifiers are fusion and selection. On *fusion of classifiers*, each of the classifiers has knowledge of the totality of the feature space. On the other hand, on *selection of classifiers*, each classifier knows only a part of the feature space.

The methods based on *fusion of classifiers* are also known as *ensemble methods*. The most popular strategies are *Boosting*, *Bagging* and *Stacking*:

- *Boosting* is based on the question enunciated by Kearns [27]: “can a set of weak learners create a single strong learner?” They consist of training several weak classifiers iteratively and adding them to a final strong classifier. After a weak learner is added, data are weighted: misclassified samples gain weight and correctly classified ones lose weight. In this manner, newly added weak learners focus more on previously misclassified samples. Algorithms of this family are, e.g. AdaBoost [22] and its variants AdaBoost.M1 and M2 [21], and AdaBoostR [37].
- *Bagging* [8] randomly generates several data sets from the original one with replacement. The models are trained and combined using voting.

I. Porto-Díaz (✉) · D. Martínez-Rego · A. Alonso-Betanzos ·
O. Fontenla-Romero
Department of Computer Science, Facultade de Informática,
University of A Coruña, Campus de Elviña s/n, Coruña, Spain
e-mail: iporto@udc.es

- *Stacking* [55] utilizes an extra classifier that learns to combine the outputs of the base classifiers in order to generate a common final output.

The methods based on *selection of classifiers* are also known as *local methods*. The idea of using different classifiers for different inputs was suggested by Dasarathy and Sheela [14], who combined a linear classifier and a *k-Nearest Neighbor* (k-NN). Rastrigin [48] already proposed a methodology for selection of classifiers that is virtually similar to the one used these days. The philosophy of local methods consists of splitting up the feature space in several subspaces and adjusting a model for each of these subspaces. Each subproblem is supposed to be simpler than the original model and may be solved with simpler classification models, i.e. linear ones. In this manner, large and complex problems, like the ones dealt in this paper, are more approachable. Therefore, a correct division of the original problem is very important for the correct operation of the system. The most straightforward way of splitting up the data is a division in regular regions, which is possible, but it may happen that some of them contain few or no data at all. In order to insure that the regions always contain some patterns, it is usual to employ a clustering algorithm to split up the data [29,35].

In this paper, a local classification method called Frontier Vector Quantization based on Information Theoretic concepts (FVQIT) is presented. The algorithm performs classification based on the combination of neural networks by means of local modeling and techniques based on Information Theoretic Learning (ITL) [44]. ITL is a framework used to construct algorithms which is able to model information extracted from data. It has obtained analytically tractable cost functions for Machine Learning problems through the expression of classical Information Theory concepts such as divergence, mutual information, etc. in terms of Renyi's entropy and nonparametric Probability Density Function (PDF) estimators. These cost functions can substitute classical ones such as Mean Square Error (MSE) in widely known models such as linear filters, Multi-Layer Perceptron (MLP), etc. It can be proven that, using these cost functions, models are more robust to noise and able to obtain optimal parameters in a more accurate way in scenarios where Gaussianity of output noise cannot be assumed [44]. FVQIT is based on Vector Quantization using Information Theoretic concepts (VQIT) [31], an information theoretic clustering algorithm that is able to distribute a set of nodes in such a way that mutual information between the nodes and the data set is maximized. The result of this self-organizing task can be subsequently used for clustering or quantization purposes.

There exist two versions of FVQIT, one for binary (two-class) problems and the other one for multiclass problems. The method, in its binary version, has been applied to several high-dimensional problems, both in samples and fea-

tures, such as intrusion detection [43] and binary microarray gene expression [41,42]. Moreover, in this paper, the multiclass version of FVQIT has also been studied over several multiclass microarray gene expression data sets, and a comparative study with other classifiers has been carried out.

The rest of the paper is organized as follows. Section 2 describes the learning model and the training algorithm for the binary version of the method. Section 3 describes the multiclass version of FVQIT. In Sect. 4, the application of the binary version of the method to intrusion detection and microarray gene expression is exposed. Section 5 thoroughly exposes the study of the multiclass version of the method on the multiclass microarray gene expression problem. Finally, Sect. 6 states the conclusions extracted.

2 Learning model for binary classification problems

The binary version of the model utilizes supervised learning and is based on local modeling and ITL [36]. The method is composed of two stages. First, a set of nodes, which are points placed in the same space as data, are moved from their initial random positions to the frontier between classes. This part of the algorithm is a modification of VQIT algorithm [31]. Second, a set of local models, associated with the nodes, based on one-layer neural networks are trained using the efficient algorithm described in [10], in such a way that a piecewise borderline between the classes is built. Therefore, the final system consists of a set of local experts, each of which will be trained to solve a subproblem of the original. In this manner, the method benefits from a finer adaptation to the characteristics of the training set. The following subsections describe both stages in detail.

2.1 Selection of local models

The VQIT algorithm, upon which FVQIT is based, was developed for vector quantization, that is, for representing a large data set with a smaller number of vectors in an appropriate way [31]. However, in our approach, the original algorithm has been modified in order to be able to build a piecewise representation of the borderline between classes in a classification problem. Therefore, the objective is placing a set of nodes on the frontier between the two classes, in such a way that each node will represent a local model.

The algorithm minimizes the energy function that calculates the divergence between the Parzen estimator of the distribution of data points and the estimator of the distribution of the nodes. Under this premise, a physical interpretation can be made. Both data points and nodes are considered two kinds of particles associated with a potential field. These fields induce repulsive and attractive interactions between particles, depending on its sign. In the original VQIT algorithm, data and nodes had different signs. In FVQIT, data

Fig. 1 Evolution of a node from a random position to a position on the frontier between classes

particles belonging to different classes have different signs. In this manner, a series of forces converge upon each node. Training patterns of a class exert an attractive force on a node, and training patterns of the other class induce a repulsive force on it. Which class attracts and which class repels is decided using the Euclidean distance and k-NN [11] as a rule of thumb. The closest class to the node (called ‘own class’) repels it and the furthest one attracts it. These roles alternate during the iterations as nodes move. An example of the movement of a node until it reaches its stability point can be seen in Fig. 1. Moreover, there exists a third force of repulsion between the nodes, which favors a better distribution, avoiding the accumulation of several nodes on a point.

In this context, the Parzen density estimators of the distribution of data points $f(\mathbf{x})$ and nodes $g(\mathbf{x})$ are:

$$f(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^N K(\mathbf{x} - \mathbf{x}_i, \sigma_f^2) \quad (1)$$

$$g(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^N K(\mathbf{x} - \mathbf{w}_i, \sigma_g^2)$$

where N is the number of data points, K is any kernel function, σ_f^2 and σ_g^2 are the variances of the kernel functions, $\mathbf{x}_i \in \mathbb{R}^n$ are data points, and $\mathbf{w}_i \in \mathbb{R}^n$ are the weights associated to the nodes.

The function of energy $J(\mathbf{w})$ that calculates the divergence between the Parzen estimators is:

$$J(\mathbf{w}) = \log \int f^2(\mathbf{x}) d\mathbf{x} + 2 \log \int f^+(\mathbf{x})g(\mathbf{x}) d\mathbf{x} - 2 \log \int f^-(\mathbf{x})g(\mathbf{x}) d\mathbf{x} + \log \int g^2(\mathbf{x}) d\mathbf{x} \quad (2)$$

where $f^+(\mathbf{x})$ and $f^-(\mathbf{x})$ are the estimators of the distributions of data for each of the classes.

According to [45], the first term of (2) is the information potential among data. Since data are stationary during the learning process, this term will not be considered from now on. The second and third terms are the crossed correlations between the distributions of data and nodes. The fourth term is the information potential of the nodes. Note that $H(\mathbf{x}) = -\log \int g^2(\mathbf{x}) d\mathbf{x}$ is the Renyi quadratic entropy of the nodes. Consequently, minimizing the divergence between $f(\mathbf{x})$ and $g(\mathbf{x})$ is equivalent to maximizing the sum of the entropy of the nodes and the cross-information potentials between the distribution of data and nodes.

Assuming this formulation, when the nodes are placed on the minimum of the energy function $J(\mathbf{w})$, they are situated on a frontier area. Therefore, we utilize the gradient descent method to move the nodes toward such situation. To develop this, the derivative of (2) is calculated. For simplicity, the derivation of $J(\mathbf{w})$ is divided in three parts: (a) calculation of the contribution of the data from the own class (the closest one), (b) calculation of the contribution of the data from the other class (the furthest one) and (c) calculation of the contribution of the interactions between nodes.

Developing the last three terms in (2):

– Data from the own class:

$$C_+ = \int f^+(\mathbf{x})g(\mathbf{x}) d\mathbf{x} = \frac{1}{MN_+} \int \sum_i^{N_+} G(\mathbf{x} - \mathbf{x}_i^+, \sigma_f^2) \sum_j^M G(\mathbf{x} - \mathbf{w}_j, \sigma_g^2) d\mathbf{x} = \frac{1}{MN_+} \sum_i^{N_+} \sum_j^M \int G(\mathbf{x} - \mathbf{x}_i^+, \sigma_f^2) G(\mathbf{x} - \mathbf{w}_j, \sigma_g^2) d\mathbf{x} = \frac{1}{MN_+} \sum_j^M \sum_i^{N_+} G(\mathbf{w}_j - \mathbf{x}_i^+, \sigma_a^2) \quad (3)$$

where M is the number of nodes, N_+ is the number of objects from the class of the node, $\mathbf{x}_i^+$ are the data from the own class, $\mathbf{w}_j$ are the weights of the nodes and the covariance of the Gaussian after integration is $\sigma_a^2 = \sigma_f^2 + \sigma_g^2$.

– Data from the other class:

$$C_- = \int f^-(\mathbf{x})g(\mathbf{x}) d\mathbf{x} = \frac{1}{MN_-} \int \sum_i^{N_-} G(\mathbf{x} - \mathbf{x}_i^-, \sigma_f^2) \sum_j^M G(\mathbf{x} - \mathbf{w}_j, \sigma_g^2) d\mathbf{x} = \frac{1}{MN_-} \sum_i^{N_-} \sum_j^M \int G(\mathbf{x} - \mathbf{x}_i^-, \sigma_f^2) G(\mathbf{x} - \mathbf{w}_j, \sigma_g^2) d\mathbf{x} = \frac{1}{MN_-} \sum_j^M \sum_i^{N_-} G(\mathbf{w}_j - \mathbf{x}_i^-, \sigma_a^2) \quad (4)$$

where N_- is the number of objects from the class of the node, $\mathbf{x}_i^-$ are the data from the other class, $\mathbf{w}_j$ are the weights of the nodes and the covariance of the Gaussian after integration is $\sigma_a^2 = \sigma_{f-}^2 + \sigma_g^2$.

– Interactions between nodes (entropy):

$$V = \int g(\mathbf{x})^2 d\mathbf{x} = \frac{1}{M^2} \sum_i^M \sum_j^M G(\mathbf{w}_i - \mathbf{w}_j, \sqrt{2}\sigma_g) \quad (5)$$

where $\mathbf{w}_i$ and $\mathbf{w}_j$ are the weights of the nodes.

The contributions to the gradient update for each of the previous terms in an iteration are:

– Data from the own class:

$$\frac{\partial}{\partial \mathbf{w}_k} 2 \log C_+ = -2 \frac{\nabla C_+}{C_+} \quad (6)$$

where the term ∇C_+ denotes the derivative of C_+ with respect to $\mathbf{w}_k$.

$$\nabla C_+ = -\frac{1}{MN_+} \sum_i^{N_+} G(\mathbf{w}_k - \mathbf{x}_i^+, \sigma_a) \sigma_a^{-1} (\mathbf{w}_k - \mathbf{x}_i^+) \quad (7)$$

– Data from the other class:

$$\frac{\partial}{\partial \mathbf{w}_k} 2 \log C_- = -2 \frac{\nabla C_-}{C_-} \quad (8)$$

where the term ∇C_- denotes the derivative of C_- with respect to $\mathbf{w}_k$.

$$\nabla C_- = -\frac{1}{MN_-} \sum_i^{N_-} G(\mathbf{w}_k - \mathbf{x}_i^-, \sigma_a) \sigma_a^{-1} (\mathbf{w}_k - \mathbf{x}_i^-) \quad (9)$$

– Interactions between nodes (entropy):

$$\frac{\partial}{\partial \mathbf{w}_k} 2 \log V = \frac{\nabla V}{V} \quad (10)$$

where the term ∇V denotes the derivative of V with respect to $\mathbf{w}_k$.

$$\nabla V = -\frac{1}{M^2} \sum_j^M G(\mathbf{w}_j - \mathbf{w}_k, \sqrt{2}\sigma_g) \sigma_g^{-1} (\mathbf{w}_k - \mathbf{w}_j) \quad (11)$$

Therefore, using Eqs. (6), (8) and (10), and through gradient descent, the weight update rule for the node $\mathbf{w}_k$ becomes:

$$\mathbf{w}_k(n+1) = \mathbf{w}_k(n) - \eta \left(\frac{\nabla V}{V} + \frac{\nabla C_+}{C_+} - \frac{\nabla C_-}{C_-} \right) \quad (12)$$

where n is the iteration and η is the step size.

As with self-organizing maps, a good starting point is to choose high-variance kernels and a large η parameter such that all particles interact with one another. This allows a fast distribution of nodes along the feature space. Gradually, in order to obtain stability and a smooth convergence, the variances of the kernels and the parameter η are decreased or annealed at each step.

Once FVQIT is trained, the nodes, ideally, will find themselves well distributed on the frontiers between classes. Each node defines a region, a local model in the feature space which is in charge of classifying the data inside. Those models are defined by proximity: the local model associated to each node is composed of the nearest points (according to Euclidean distance) in the feature space, independently of their class. Therefore, data from both classes could coexist in the same local model. Algorithm 1 summarizes the pseudocode of the training process of FVQIT.

Algorithm 1 Training algorithm for the binary version of FVQIT.

Inputs: Training set, number of nodes M , learning rate η , covariance matrices σ_f and σ_g , annealing rates η_{dec} and σ_{dec} , maximum number of iterations p , number of neighbors k .

1. Initialize the weights of the M nodes randomly in the data range.
 2. Calculate which class repels the node and which class attracts it by calculating the Euclidean distances from each node to every data point and using the k -NN (k -Nearest Neighbor) rule.
 3. Evaluate the cross information potential C_+ between each node and the data from the repelling class, as in (3).
 4. Calculate the cross information potential C_- between each node and the data from the attracting class, using (4).
 5. Evaluate the entropy V between nodes as described in (5).
 6. Calculate the derivatives ∇C_+ , ∇C_- y ∇V , utilizing (7), (9) and (11), respectively.
 7. Evaluate the weight update for each PE using (12).
 8. Reduce learning rate η in the proportion shown by η_{dec} .
 9. Reduce σ_f and σ_g in the proportion shown by σ_{dec} .
 10. Repeat from 2 until the predefined maximum number of iterations p is reached.
-

The method employs several input parameters. Some of them can be assigned to a standard value or do not significantly affect the final performance of the method. The covariance matrices σ_f and σ_g are assigned to the covariance matrices of the patterns in the training set. This assignment is derived from the work in [31] and has obtained good results in the experiments in [36]. The parameter k of the k -NN rule does not present a great impact on performance as its effect when the nodes are near the frontier between classes is compensated due to the subsequent

moves of the nodes. It may take any typical value between 1 and 10. The parameter η controls the magnitude of node movements in each learning step. With high values, a significant oscillation of the nodes in the first learning steps will be observed and it will take longer time to converge to a stable situation in the frontier. This parameter usually takes values in the interval $[\text{range}(X)/2, \text{range}(X)]$, where $\text{range}(X) = \text{abs}(\max(D) - \min(D))$ and X , the training set. η_{dec} and σ_{dec} control the smoothness of the convergence to the frontier. They may take a value in the interval $(0, 1)$, although they typically take values close to 1 to ensure a smooth evolution. The maximum number of iterations p is a stopping condition added to the method. If a poor performance is observed, it can be increased to let the method converge to the frontier. The number of nodes M is usually selected using cross validation.

2.2 Adjustment of local models

In the first stage, a set of local models was constructed. Since each local model covers the closest points to the position of its associated node, the input space is completely filled, as input data are always assigned to a local model. In this second stage, the goal is to construct a classifier for each local model. This classifier will be in charge of classifying points in the region assigned to its local model and will be trained only with the points of the training set in this region.

As local modeling algorithms may suffer from temporal efficiency problems, caused by the process of training several local classifiers, we have decided to use a lightweight classifier. We have chosen one-layer neural networks, trained with the efficient algorithm presented in [10], as it has proven suitable in [36]. This algorithm allows rapid supervised training for one-layer feed-forward neural networks. The key idea is to measure the error prior to the non-linear activation functions. In this manner, it is proven in [10] that the minimization based on the MSE can be rewritten in equivalent fashion in terms of the error committed prior to the application of the activation function, which produces a system of equations with $I + 1$ equations and unknowns. This kind of systems can be solved computationally with a complexity of $O(M^2)$, where $M = I + 1$ is the number of weights of the network. Thus, it requires much less computational resources than classic methods.

2.3 Operation of the model

After the training process, when a new pattern arrives to be classified, the method first calculates the closest node $\mathbf{w}_k$ to a new pattern $\mathbf{x}_n$ using the Euclidean distance and then classifies it using the neural network associated with the local model $\mathbf{w}_k$.

Fig. 2 Example of operation of FVQIT. Local models and frontier between classes

In Fig. 2, a simple two-class bi-dimensional example is displayed. Data from one class are displayed with 'x' mark and data from the other class with circles. The FVQIT nodes are represented with squares. The division in local models is shown with dotted lines and the solid lines depict the decision regions defined by each neural network.

3 Learning model for multiclass classification problems

The training process of multiclass FVQIT is very similar to the binary one. In the first stage of the training process of the binary version, the closest class to each node in each iteration repelled the node and the other class attracted it. In the multiclass version, for each node, the two nearest classes are chosen using the same k-NN rule of thumb. From among them, the closest one repels the node; the second closest one attracts it (Algorithm 2). The rest of the training of the first stage is the same as in binary FVQIT, employing the two closest classes in order to generate the crossed information potentials (see Sect. 2).

Algorithm 2 Mechanism of selection of the classes that attract and repel.

Inputs: Training set, number of classes, distance from each node to each data point, number of neighbors k .

1. For each node $\mathbf{w}_i$,
 - (a) Sort the data points by increasing Euclidean distance to the node.
 - (b) Take the classes of the k closest points and calculate its mode. The mode will be the repelling class for that node.
 - (c) Take the classes of the k closest points to each node that do not belong to the repelling class and calculate its mode. The mode will be the attracting class for that node.
-

In the second stage of the training of binary FVQIT, a one-layer neural network was trained in each local model. In the multiclass version, instead of just one neural network, we will have several one-layer neural networks in each local model, each of them associated with one of the classes of the problem. Thereafter, the training is performed following a one-versus-rest strategy, that is to say, each neural network is trained to recognize the patterns of “its” class against the points of the rest of classes.

Once the model is trained, when a new pattern needs to be classified, in binary FVQIT the pattern was assigned to the nearest local model (using Euclidean distance) and the associated network classified it into one of the two classes. In multiclass FVQIT, the pattern is assigned to a local model in the same manner. However, after that, the outputs of the one-layer neural networks associated to this local model are evaluated. The pattern is classified in the class associated to the network that produces the highest output ($c_i = \arg \max_j net_j$).

4 Applications of the binary version

The binary version of FVQIT has been studied over several high-dimensional problems. In this section, two studies are described. First, the study on intrusion detection, particularly the KDD Cup 99 data set, which has a very large amount of data. Second, the method is applied on microarray gene expression data sets. These data sets have a very large amount of features (in the order of the thousands) and very few samples (in the order of the tens).

4.1 Experimental study over intrusion detection

The KDD Cup 99 data set is a processed version of the DARPA 1998 data set, which was constructed from a simulation performed by the Defense Advanced Research Projects Agency (DARPA) through the Intrusion Detection Evaluation Program (IDEP) in 1998. The KDD Cup 99 data set was released for a classifier learning contest, whose task was to distinguish between legitimate and illegitimate connections in a computer network [18], at the Knowledge Discovery and Data Mining (KDD) Conference in 1999. The training data set consists of about 5 million connection records (although a reduced training data set containing around 500,000 records exists) [32]. Each record contains values of 41 variables which describe different aspects of the connection, and the value of the class label (either normal, either the specific attack type). The test data set comprises 300,000 records and its data are not from the same probability distribution as training data.

Following the KDD Cup contest, the data set has been extensively used as a benchmark for developing machine learning algorithms for intrusion detection systems. The data

set is very demanding not only because of its size but also due to the great inner variability among features. For those reasons, the KDD Cup 99 data set is one of the most challenging classification problems nowadays. Despite that KDD Cup 99 is a multiclass data set, it can be treated as a binary data set, simply by considering attack or no attack, instead of the different attack types. This approach is interesting in the sense that, most of the time, it is enough to distinguish between normal connections and attacks. This transformation has been carried out by other authors [3,23], and there exist several results in the literature which are utilized as part of the comparative study.

The experimental study performed involves applying the proposed FVQIT algorithm to the binary version of the KDD Cup 99 data set [43]. As a preliminary stage, discretization and feature selection were both performed on the data set. The motivation for using discretization is that some features of the KDD Cup 99 data set present high imbalance and variability. This situation may cause a malfunction in most feature selection methods and classifiers. The problem is softened up using discretization methods. In substance, the process of discretization involves putting continuous values into groups, by means of a number of discrete intervals. Two discretization methods will be employed in this study: Proportional k-Interval Discretization (PKID) [56] and Entropy Minimization Discretization (EMD) [19].

In order to reduce input dimensionality and improve the computational efficiency of the classifier, feature selection was performed. Filter methods were chosen because they are computationally cheaper than wrapper methods, and computational efficiency is a desirable feature, given the large size of the data set [6]. The filters that will be used in this study are INTERACT [57] and Consistency-based Filter [15].

The discretization methods (PKID and EMD) are considered in combination with the above-named filters (INTERACT and Consistency based). Thus, four combinations of discretizator plus filter are analyzed in order to check which subset of features works best with the FVQIT method.

The model is trained with the KDD Cup 99 reduced training data set—494,021 samples—and is tested using the standard KDD Cup 99 test data set of 311,029 samples. Three performance measures are employed:

- Test error (TE): indicates the overall percentage error rate for both classes (Normal and Attack).
- True positive rate (TP): shows the overall percentage of detected attacks.
- False positive rate (FP): indicates the percentage of normal patterns classified as attacks.

The results of the proposed method are compared with those obtained by other authors [3,6,18,23], as can be seen in Table 1. Specifically, the classification methods to be

Table 1 Results obtained by the four versions of the proposed method and by other authors

Method	TE (%)	TP (%)	FP (%)	NF
PKID+Cons+FVQIT	5.95	92.73	0.48	6
EMD+INT+FVQIT	5.40	93.50	0.85	7
EMD+Cons+FVQIT	4.73	94.50	1.54	7
PKID+INT+FVQIT	5.68	93.61	2.75	7
KDD Winner	6.70	91.80	0.55	41
PKID+Cons+C4.5	5.14	94.08	1.92	6
EMD+INT+C4.5	6.69	91.81	0.49	7
FNs_poly	6.48	92.45	0.86	41
FNs_fourier	6.69	92.72	0.75	41
FNs_exp	6.70	92.75	0.75	41
SVM Linear	6.89	91.83	1.62	41
SVM RBF	6.86	91.83	1.43	41
ANOVA ens.	6.88	91.67	0.90	41
LP 2cl.	6.90	91.80	1.52	41

compared are decision trees (C4.5), functional networks (FN), SVMs, ANalysis Of VAriance (ANOVA) (ANOVA ens.) and linear perceptrons (LP). Font in boldface indicates best results considering all three measures altogether. Table columns show the test error (TE), the true positive rate (TP), the false positive rate (FP) and the number of features employed (NF). Both error and rates are shown in percentage (%). These measures (TE, TP and FP) are typical in the field of intrusion detection.

As can be seen in Table 1, the combination PKID+Cons+FVQIT obtains the best result as it improves the performance obtained by the KDD Cup Winner in all three measures used, using a considerably reduced number of features (6 instead of the 41 original features).

In addition, this combination outperforms all other results included in this study. Despite the fact that individual values of error and TP for the combination EMD+Cons+FVQIT are better than those for the above-mentioned combination—4.73 versus 5.95 and 94.50 versus 92.73—it must be noted that the variations in percentage between these quantities are quite small—20 and 2 % respectively—in contrast to the variation between the values of FP—1.54 versus 0.48 (300%)—. On the other hand, error and TP for EMD+INT+FVQIT, EMD+Cons+FVQIT, and PKID+INT+FVQIT are good, but unfortunately at the expense of FP, which happens to be high for all of them.

4.2 Experimental study over microarray gene expression

In this experimental study, the FVQIT classifier is employed to classify 12 DNA microarray gene expression data sets of different kinds of cancer. These data sets present features of

Table 2 Description of the data sets

Data set	No. of features	Total samples
Brain [38]	12,625	21
Breast [53]	24,481	97
CNS [40]	7,129	60
Colon [2]	2,000	62
DLBCL [1]	4,026	47
GLI [7]	22,283	85
Leukemia [24]	7,129	72
Lung [25]	12,533	181
Myeloma [52]	12,625	173
Ovarian [39]	15,154	253
Prostate [50]	12,600	136
SMK [51]	19,993	187

the order of thousands and very few samples (tens or hundreds). A comparative study with other well-known classifiers is carried out [41,42]. The number of features and samples for each data set are shown in Table 2.

Since the number of input features of these kinds of data sets is huge, as can be seen on Table 2, feature selection is applied again, as in the previous problem [49]. Two different kinds of filter methods are employed: subset filters and rankers. Subset filters provide a subset of selected features, while rankers make use of a scoring function in order to build a feature ranking, where all features of the data set are sorted in decreasing relevance order. In the first experiment (subset filters), the performance of the method is tested. The aim of the second experiment (ranker methods), is to check

the stability of the performance reached by FVQIT, independently of the number of features selected.

4.2.1 Experiment 1: study of performance using subset filters

In the first experimental setting, the FVQIT method is compared with other classifiers with the objective of finding out which classifier obtains the best performance. Thus, five well-known machine learning classifiers—naive Bayes (NB), k-NN, C4.5, SVMs, and MLP—are also applied over the filtered data sets. The implementation of these methods can be found in [33], except for MLP, where the Matlab Neural Networks Toolbox was used. Three filters have been chosen in order to consider different behaviors. In previous works, values obtained by filters were shown to be influenced by discretization [5], thus, as a consequence, we are using two discretizers – EMD [19] and PKID [56]—in combination with the subset filters Correlation-based Feature Selection (CFS) [26], Consistency-based Filter [15] and INTERACT [57], which can be found in the Weka tool [54].

The data sets have been divided using 2/3 for training and 1/3 for test. A tenfold cross validation has been performed on the training sets, in order to estimate the validation error to choose a good configuration of parameters. The results of FVQIT have been compared with those obtained by other classifiers. Table 3 shows the estimated test errors (TE in the table) as well as the sensitivity (Se) and specificity (Sp) rates—in percentage—and the number of features (NF) selected by each method tested. Moreover, the ranking is displayed between parentheses. The ranking assigns a position between 1 and 6 to each method in each data set, taking into account the ties among them. Also, the best error obtained for each data set is emphasized in bold font. Despite having executed all six combinations of discretizer + filter, only the best result for each classifier in each data set is shown.

As can be seen in Table 3, the FVQIT method obtains good performance on all data sets, with an adequate number of selected features. Specially remarkable are the results obtained on the data sets DLBCL and Leukemia, where the FVQIT classifier is the only method able to achieve 0 % of test error. The result obtained on the Prostate data set is also important. Its test set is unbalanced (26 % of one class and 74 % of the other). C4.5, naive Bayes and k-NN are assigning all the samples to the majority class and SVM is assigning all the samples to the minority class, whereas FVQIT is able to do something different and better, which results in a lower test error.

In Table 4, the average rankings (obtained from the rankings displayed in Table 3 between parentheses) are shown. In average, the proposed method is clearly preferable above the other methods studied. It is shown that the proposed method is the most specific (it correctly identifies most of the

negatives) and the most sensitive (it correctly identifies most of the positives). Therefore, in light of the above, we can conclude that FVQIT classifier is suitable to be combined with discretizers and filters to deal with problems with a much higher number of features than instances, such as DNA microarray gene expression problems.

4.2.2 Experiment 2: study of performance stability using rankers

When using feature selection, sometimes it is difficult to compare performance between classifiers because there are two variables involved: test error and number of features selected. Depending on the application, sometimes it may be desirable to choose the minimum test error regardless of the number of features, but sometimes a somewhat larger error may be accepted in the interest of a smaller number of features. In this context, the aim of the second experiment is to check the stability of the performance reached by the FVQIT classification method independently of the number of features selected. Therefore, in this case, it is advisable to utilize rankers, so as to compare the performance of the classifiers in a wide range of selected features. Four rankers have been chosen in order to consider different behaviors. The ranker methods that we have chosen and whose implementation can be found in [33] are the following: Fisher Score [17], Chi-square [34], Information Gain [12], and Minimal Redundancy Maximal Relevance (mRMR) [16].

Since ranker methods provide a sorted list of features according to a score, there is a decision to make regarding the number of features to be selected. As of this, in this experiment, we are going to test the classifiers with different numbers of features. Thus, we are going to select the first 1, 3, 5, 10, 15, 20, 30, 40, 50 and 100 features from the sorted list of features that the rankers provide.

First, the overall results of the comparative study for each data set are presented and then we focus on the overall results for each feature number. As the number of experimental results is very large (all the combinations of 4 rankers, 7 classifiers and 10 different feature numbers over 12 data sets account for 3,360 experiments), some summary of results needs to be used. In a similar way as in the first part of the experimental section, the methods are sorted in a table using a ranking in which ties have been taken into consideration. The average rankings of test error, sensitivity and specificity for all 12 data sets are represented in Table 5.

As can be seen in Table 5, the FVQIT method is the classifier that obtains the best average performance for all data sets, as well as the best sensitivity and specificity. However, FVQIT does not obtain the best performance in every data set. Since these data sets are a difficult challenge, obtaining the best result in average is an important achievement for FVQIT,

Table 3 Best estimated test errors (TE), sensitivity (Se), specificity (Sp) and number of features selected (NF)

Data set	FVQIT	SVM	NB	MLP	k-NN	C4.5
Brain						
TE	0.00 (1)	14.29 (4)	14.29 (4)	28.57 (6)	0.00 (1)	0.00 (1)
Se	100.00 (1)	100.00 (1)	100.00 (1)	0.00 (6)	100.00 (1)	100.00 (1)
Sp	100.00 (1)	83.33 (4)	83.33 (4)	71.43 (6)	100.00 (1)	100.00 (1)
NF	1	45	45	1	1	45
Breast						
TE	21.05 (1)	21.05 (1)	26.32 (5)	21.05 (1)	26.32 (5)	21.05 (1)
Se	75.00 (5)	83.33 (1)	83.33 (1)	83.33 (1)	83.33 (1)	66.70 (6)
Sp	85.71 (2)	71.43 (3)	57.10 (5)	71.43 (3)	57.10 (5)	100 (1)
NF	17	119	5	17	5	3
CNS						
TE	25.00 (1)	35.00 (3)	25.00 (1)	35.00 (3)	35.00 (3)	35.00(3)
Se	69.20 (3)	71.43 (2)	69.20 (3)	68.75 (6)	69.20 (3)	76.90 (1)
Sp	85.70 (1)	50.00 (4)	85.70 (1)	50.00 (4)	57.10 (3)	42.90 (6)
NF	4	60	4	60	4	47
Colon						
TE	10.00 (1)	10.00 (1)	15.00 (3)	40.00 (6)	15.00 (3)	15.00 (3)
Se	80.00 (4)	80.00 (4)	87.50 (1)	50.00 (6)	87.50 (1)	87.50 (1)
Sp	100.00 (1)	100.00 (1)	83.30 (3)	61.11 (6)	83.30 (3)	83.30 (3)
NF	12	12	3	12	3	3
DLBCL						
TE	0.00 (1)	6.67 (2)	6.67 (2)	6.67 (2)	6.67 (2)	13.33 (6)
Se	100.00 (1)	100.00 (1)	85.70 (4)	100.00 (1)	85.70 (4)	85.70 (4)
Sp	100.00 (1)	88.89 (4)	100.00 (1)	88.89 (4)	100.00 (1)	87.50 (6)
NF	36	36	36	47	36	2
GLI						
TE	10.71 (1)	14.29 (3)	10.71 (1)	17.86 (5)	14.29 (3)	21.43 (6)
Se	85.71 (1)	85.00 (3)	85.71 (1)	78.26 (5)	81.82 (4)	75.00 (6)
Sp	100.00 (1)	87.50 (6)	100.00 (1)	100.00 (1)	100.00 (1)	100.00 (1)
NF	113	23	23	23	122	3
Leukemia						
TE	0.00 (1)	2.94 (2)	5.88 (3)	5.88 (3)	8.82 (6)	5.88 (3)
Se	100.00 (1)	100.00 (1)	100.00 (1)	92.86 (5)	100.00 (1)	92.86 (5)
Sp	100.00 (1)	95.24 (2)	90.00 (5)	95.00 (3)	90.00 (5)	95.00 (3)
NF	2	18	18	2	1	2
Lung						
TE	0.67 (2)	1.34 (4)	4.70 (5)	0.67 (2)	0.00 (1)	18.12 (6)
Se	100.00 (1)	99.26 (3)	94.80 (5)	99.26 (3)	100.00 (1)	82.80 (6)
Sp	93.75 (4)	93.33 (5)	100.00 (1)	100.00 (1)	100.00 (1)	73.30 (6)
NF	40	40	1	40	40	1
Myelomas						
TE	21.05 (2)	21.05 (2)	21.05 (2)	21.05 (2)	29.82 (6)	19.30 (1)
Se	84.00 (1)	81.48 (3)	81.48 (3)	80.36 (6)	82.20 (2)	80.70 (5)
Sp	42.86 (1)	33.33 (2)	33.33 (2)	0.00 (5)	25.00 (4)	0.00 (5)
NF	2	40	2	2	2	2

Table 3 continued

Data set	FVQIT	SVM	NB	MLP	k-NN	C4.5
Ovarian						
TE	0.00 (1)	0.00 (1)	0.00 (1)	0.00 (1)	0.00 (1)	1.19 (6)
Se	100.00 (1)	100.00 (1)	100.00 (1)	100.00 (1)	100.00 (1)	98.10 (6)
Sp	100.00 (1)	100.00 (1)	100.00 (1)	100.00 (1)	100.00 (1)	100.00 (1)
NF	3	3	3	17	3	
Prostate						
TE	20.59 (1)	73.53 (6)	26.47 (3)	23.53 (2)	26.47 (3)	26.47 (3)
Se	56.25 (2)	26.47 (3)	0.00 (4)	100.00 (1)	0.00 (4)	0.00 (4)
Sp	100.00 (1)	0.00 (6)	100.00 (1)	75.76 (5)	100.00 (1)	100.00 (1)
NF	64	3	2	3	2	2
SMK						
TE	25.81 (1)	33.87 (3)	40.32 (6)	32.26 (2)	33.87 (3)	33.87 (3)
Se	78.79 (2)	71.88 (4)	67.85 (6)	89.47 (1)	75.00 (3)	68.42 (5)
Sp	68.97 (1)	60.00 (3)	52.94 (6)	58.14 (5)	58.82 (4)	62.50 (2)
NF	21	3	3	21	21	3

The rankings are displayed between parentheses

Table 4 Average rankings of error, sensitivity and specificity for all data sets

Measure	FVQIT	SVM	NB	MLP	k-NN	C4.5
TE	1.17	2.67	3.00	2.92	3.08	3.50
Sensitivity	1.92	2.25	2.58	3.50	2.17	4.17
Specificity	1.33	3.42	2.58	3.67	2.92	3.00

Table 5 Average ranking of test error (TE), sensitivity (Se) and specificity (Sp) for all data sets

Data set	FVQIT			SVM			NB			MLP			k-NN			C4.5		
	TE	Se	Sp	TE	Se	Sp	TE	Se	Sp	TE	Se	Sp	TE	Se	Sp	TE	Se	Sp
Brain	2.1	1.4	2.6	2.3	2.4	2.5	2.9	3.3	2.5	3.7	3.4	3.2	3.9	4.6	3.9	1.3	3.5	1.0
Breast	2.3	2.6	2.4	2.4	2.6	2.8	3.3	3.7	3.7	4.0	4.4	3.9	3.5	3.8	3.6	2.9	2.8	2.9
CNS	1.6	2.2	3.0	2.9	5.4	1.0	4.6	3.8	4.9	3.3	3.5	3.9	2.4	2.8	3.5	2.3	2.6	3.5
Colon	2.0	3.3	2.1	6.0	1.0	6.0	1.4	2.6	1.4	2.0	3.0	2.3	3.5	4.4	3.6	2.0	3.1	2.0
DLBCL	1.4	1.0	1.9	2.1	1.1	2.3	1.7	1.7	1.7	1.4	1.7	1.4	2.7	2.0	3.3	3.6	4.2	3.6
GLI	1.1	1.4	2.2	6.0	6.0	1.0	1.5	1.8	2.9	2.2	2.7	2.5	2.0	2.4	2.8	4.5	4.5	5.9
Leukemia	1.3	2.3	1.3	6.0	1.0	6.0	1.8	3.3	1.4	3.3	2.5	3.8	1.2	2.1	1.6	4.2	5.6	3.9
Lung	1.9	2.3	2.6	4.5	6.0	1.0	2.0	1.2	2.9	2.4	2.1	3.1	2.3	1.8	3.2	5.2	5.0	5.9
Myeloma	3.9	2.5	3.4	1.5	3.3	2.1	5.3	5.5	5.3	1.6	3.5	2.6	4.5	3.1	4.3	2.5	2.6	2.5
Ovarian	2.2	1.3	2.1	1.1	1.2	1.0	3.8	2.0	3.6	1.5	1.5	1.2	4.7	2.3	4.7	2.2	2.0	1.5
Prostate	1.6	2.5	1.9	3.1	1.3	3.9	4.9	4.8	3.4	1.9	3.5	2.4	1.8	3.5	2.6	4.0	4.9	2.9
SMK	3.7	3.8	3.9	1.7	2.7	2.1	2.5	3.0	3.0	4.0	2.4	4.3	3.4	4.1	3.8	3.9	4.8	3.7
Average	2.09	2.22	2.45	3.30	2.83	2.64	2.98	3.06	3.06	2.61	2.85	2.88	2.99	3.07	3.41	3.22	3.80	3.28

especially when comparing it with popular and well-tested methods such as the ones employed in this work.

In a second step, the results in function of the number of features are analyzed. Again, the same processing is made, in such a way that the average ranking of test error, sensitivity and specificity for all features are represented in Table 6.

From Table 6, it can be observed how the FVQIT classifier outperforms the other methods for all feature numbers except for 100 features, where it obtains the second best result, behind MLP. In light of the above, it can be concluded that FVQIT is the most stable classifier because it obtains good results both with few and many features, in

Table 6 Average ranking of test error (TE), sensitivity (Se) and specificity (Sp) for all features

Data set	FVQIT			SVM			NB			MLP			k-NN			C4.5		
	TE	Se	Sp	TE	Se	Sp	TE	Se	Sp	TE	Se	Sp	TE	Se	Sp	TE	Se	Sp
1	1.7	2.0	1.8	2.9	2.7	2.4	2.4	2.1	2.6	1.9	2.7	2.7	3.6	4.0	3.7	2.3	3.4	2.1
3	2.0	3.1	2.5	2.9	2.9	2.2	2.8	2.8	2.8	2.3	2.7	2.4	3.6	2.9	3.3	2.6	3.6	2.9
5	2.3	1.9	2.6	3.3	3.3	3.1	2.7	3.3	2.8	2.4	3.4	2.4	3.6	3.8	3.8	2.4	3.3	2.8
10	2.3	2.1	2.7	3.3	3.0	2.4	3.0	2.9	3.3	2.8	3.1	3.3	3.2	3.4	3.9	3.2	3.3	3.4
15	2.3	2.8	2.5	3.3	2.9	2.5	3.2	3.3	3.4	2.8	2.7	3.2	3.1	2.7	3.8	3.4	3.2	3.6
20	2.2	2.2	2.6	3.8	2.8	3.2	3.0	3.3	2.8	3.0	3.2	3.2	2.8	2.8	3.3	3.1	3.8	3.1
30	1.8	1.8	2.2	3.3	2.6	2.7	3.3	3.8	3.5	3.2	2.4	3.3	2.6	3.0	3.2	3.7	4.0	3.7
40	2.2	2.4	2.3	3.4	2.8	2.8	2.9	2.8	3.0	2.7	2.7	3.2	2.3	2.7	2.8	3.5	4.2	3.3
50	1.8	1.4	2.6	3.2	2.7	2.5	3.2	3.1	3.2	2.8	3.3	3.2	2.1	2.8	2.6	4.2	5.0	4.1
100	2.3	2.5	2.8	3.5	2.8	2.8	3.3	3.4	3.4	2.1	2.5	2.2	3.1	2.8	3.6	3.8	4.3	3.9
Average	2.09	2.22	2.45	3.30	2.83	2.64	2.98	3.06	3.06	2.61	2.85	2.88	2.99	3.08	3.41	3.22	3.80	3.28

Table 7 Multiclass DNA microarray data sets employed in the experiment

Data set	No. of samples	No. of features	No. of classes
CLL-SUB	74	11,340	3
GCM	144	16,063	14
GLA-BRA	120	49,151	4
Lymphoma	64	4,026	9
TOX	114	5,748	4

Table 8 Number of features selected by the INTERACT filter

Data set	No. of features
CLL-SUB	61
GCM	78
GLA-BRA	150
Lymphoma	160
TOX	80

contrast with other classifiers. For instance, k-NN performs correctly between 15 and 50 features but it does not obtain good results with smaller numbers (<15) and higher ones (100). On the other hand, C4.5 performs adequately with few features but its performance decreases when the number of features increases. Finally, MLP shows stable behavior for all the feature numbers (although it is better for few features), but, on average, FVQIT performs better. Besides, the FVQIT method is the most sensitive and specific in average. For further details, please refer [42].

5 Applications of the multiclass version

The multiclass version of FVQIT has been applied over several multiclass microarray gene expression data sets. In this

study, five multiclass DNA microarray data sets have been chosen. The main characteristics of these data sets are shown in Table 7. Three of them (CLL-SUB, GLA-BRA and TOX) have been obtained from the Web site of feature selection of the Arizona State University [33]. The remaining data sets (GCM and Lymphoma) are available at the Broad Institute Cancer Program Data Sets Repository [9]. The methods compared with FVQIT are: MLP, SVM—note that a one-versus-all strategy has been used), k-NN, Naive Bayes (NB), and C4.5.

As can be seen, the multiclass DNA microarray data sets also present many more features than instances. Therefore, again, feature selection methods are utilized. For this experiment, the INTERACT filter [57] is applied to those data sets as a preprocessing step in order to make them manageable. This filter has been previously utilized with success on binary microarray data sets [41]. The number of features selected for each data set is displayed in Table 8. The data sets have been divided using 2/3 for training and 1/3 for testing. Table 9 shows the estimated test errors (in percentage) for each classifier and data set.

A tenfold cross validation is performed upon the training sets in order to choose a good configuration of parameters. The k in the k-NN method ranges from 1 to 5. The SVM utilizes a Radial Basis Function kernel and its parameters C and γ range from 1 to 10,000 and 0.1 to 40, respectively. The MLP has one hidden layer which contains neurons between 3 and 50. The FVQIT utilizes between 10 and 40 nodes, 100 iterations, initial η between 1 and 5 and η decrement between 0.7 and 0.99.

From Table 9, it can be observed that the FVQIT obtains the best test errors in four of the five data sets. In Table 10, a ranking of the performance results for all the compared methods is shown. The ranking assigns a position between 1 and 6 to each method for each data set. The proposed method

Table 9 Error committed (%) by each method on each multiclass DNA microarray data set

Classifier	CLL-SUB	GCM	GLA-BRA	Lymphoma	TOX	Average
FVQIT	21.62	45.65	33.33	12.50	12.28	26.41
k-NN	29.73	54.35	41.67	15.63	22.81	32.84
Naive Bayes	27.03	50.00	36.67	40.63	26.32	36.13
SVM	37.84	73.91	48.33	25.00	15.79	40.17
MLP	45.95	39.13	35.00	43.75	38.60	40.49
C4.5	43.24	63.04	55.00	46.88	52.63	52.16

Table 10 Ranking for each method on the comparative study of multiclass DNA microarray data sets

Classifier	CLL-SUB	GCM	GLA-BRA	Lymphoma	TOX	Average
FVQIT	1st	2nd	1st	1st	1st	1.2
Naive Bayes	2nd	3rd	3rd	4th	4th	3.2
k-NN	3rd	4th	4th	2nd	3rd	3.2
MLP	6th	1st	2nd	5th	5th	3.8
SVM	4th	6th	5th	3rd	2nd	4
C4.5	5th	5th	6th	6th	6th	5.6

is clearly preferable, as it obtains an average ranking of 1.2 as opposed to the ranking of 3.2 of the second classifier.

6 Conclusions

In this paper, a local classifier based on ITL is presented. The classifier is able to obtain complex classification models via a two-step process that first defines local models by means of a modified clustering algorithm and, subsequently, several one-layer neural networks, assigned to the local models, construct a piecewise borderline between classes. Two versions of the method are detailed: binary (two-class problems) and multiclass. Using the divide-and-conquer approach, it has been shown that the proposed method is able to successfully classify complex and unbalanced data sets, high-dimensional data samples and/or features, achieving good average results. Several experiments have been performed over the complex domains of intrusion detection and microarray gene expression. The intrusion detection data set employed is KDD Cup 99. It is very large (5 million samples), highly unbalanced and has 41 features. The most important contribution of the method is the considerable reduction in the number of false positives (an important measure in this field of application), with a considerable reduction in the number of features used (6 vs. 41) in comparison with the KDD Winner and the results obtained by other authors. On the other hand, the microarray data sets have a large amount of features (thousands or tens of thousands) but very few samples (tens or hundreds), which is a difficult challenge for most machine learning methods. In this case, the method has been compared with several

state-of-the-art classifiers, achieving the best average values of all the performance measurements used, exhibiting an important difference with the second best method, both in the binary and the multiclass experiments. Furthermore, as different feature selection methods can select different features, the stability of the proposed method has also been tested for different ranges of features, again showing the best behavior compared with the other classifiers.

Acknowledgments This work was supported in part by Xunta de Galicia under Project Code CN2011/007, and by Spanish Ministerio de Ciencia e Innovación under Project Code TIN2009-10748, both partially supported by the European Union ERDF. Furthermore, Iago Porto-Díaz is supported by University of A Coruña pre-doctoral grant and David Martínez-Rego is supported by the Spanish Ministry of Education FPU grant.

References

1. Alizadeh, A.A., Eisen, M.B., Davis, R.E., Ma, C., Lossos, I.S., Rosenwald, A., Boldrick, J.C., Sabet, H., Tran, T., Yu, X., et al.: Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature* **403**(6769), 503–511 (2000)
2. Alon, U., Barkai, N., Notterman, D.A., Gish, K., Ybarra, S., Mack, D.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl. Acad. Sci. USA* **96**(12), 6745–6750 (1999)
3. Alonso-Betanzos, A., Sanchez-Marono, N., Carballal-Fortes, F.M., Suarez-Romero, J., Perez-Sanchez, B.: Classification of computer intrusions using functional networks: a comparative study. In: *Proceedings of the ESANN*, pp. 25–27 (2007)
4. Bishop, C.M.: *Neural networks for pattern recognition*. Clarendon Press, Oxford (1995)
5. Bolón-Canedo, V., Sánchez-Marono, N., Alonso-Betanzos, A.: On the effectiveness of discretization on gene selection of microarray

- data, pp. 3167–3174. In: Proceedings of International Joint Conference on Neural Networks, IJCNN (2010)
6. Bolón-Canedo, V., Sanchez-Maroo, N., Alonso-Betanzos, A.: A combination of discretization and filter methods for improving classification performance in kdd cup 99 dataset. In: International Joint Conference on Neural Networks, IJCNN 2009, pp. 359–366. IEEE (2009)
7. Brandy Hamill. freij-affy-human-91666. <http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE4412> (2006). Accessed Sept 2012
8. Breiman, L.: Bagging predictors. *Mach. Learn.* **24**(2), 123–140 (1996)
9. Broad Institute. Broad Institute Cancer Program Data Sets. <http://www.broadinstitute.org/cgi-bin/cancer/datasets.cgi>. Accessed Sept 2012
10. Castillo, E., Fontenla-Romero, O., Guijarro-Berdiñas, B., Alonso-Betanzos, A.: A global optimum approach for one-layer neural networks. *Neural Comput.* **14**(6), 1429–1449 (2002)
11. Cover, T., Hart, P.: Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory* **13**(1), 21–27 (1967)
12. Cover, T.M., Thomas, J.A.: Elements of information theory. Wiley, London (1991)
13. Cristianini, N., Shawe-Taylor, J.: An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press, Cambridge (2000)
14. Dasarathy, B.V., Sheela, B.V.: A composite classifier system design: concepts and methodology. *Proc. IEEE* **67**(5), 708–713 (1979)
15. Dash, M., Liu, H.: Consistency-based search in feature selection. *Artif. Intell.* **151**(1), 155–176 (2003)
16. Ding, C., Peng, H.: Minimum redundancy feature selection from microarray gene expression data. *J. Bioinformatics Comput. Biol.* **3**(2), 185–206 (2005)
17. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern classification, 2nd edn. Wiley, New York (2001)
18. Elkan, C.: Results of the kdd'99 classifier learning. *ACM SIGKDD Explor. Newsl.* **1**(2), 63–64 (2000)
19. Fayyad, U., Irani, K.: Multi-interval discretization of continuous-valued attributes for classification, learning (1993)
20. Fisher, R.A.: The use of multiple measurements in taxonomic problems. *Annals Hum. Genet.* **7**(2), 179–188 (1936)
21. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: International Workshop then Conference on Machine Learning, pp. 148–156. Morgan Kaufmann Publishers, Inc. (1996)
22. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of online learning and an application to boosting. *J. Comput. System Sci.* **55**(1), 119–139 (1997)
23. Fugate, M., Gattiker, J.R.: Computer intrusion detection with classification and anomaly detection, using SVMs. *Int. J. Pattern Recognit. Artif. Intell.* **17**(3), 441–458 (2003)
24. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science* **286**(5439), 531–537 (1999)
25. Gordon, G.J., Jensen, R.V., Hsiao, L.L., Gullans, S.R., Blumenstock, J.E., Ramaswamy, S., Richards, W.G., Sugarbaker, D.J., Bueno, R.: Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. *Cancer Res.* **62**(17), 4963–4971 (2002)
26. Hall, M.A.: Correlation-based feature selection for machine learning. PhD thesis, University of Waikato, Hamilton, New Zealand (1999)
27. Kearns, M.J.: Thoughts on hypothesis boosting. *ML Class Project* **319**, 320 (1988)
28. Kiang, M.Y.: A comparative assessment of classification methods. *Decis. Support Systems* **35**(4), 441–454 (2003)
29. Kuncheva, L.I.: Clustering-and-selection model for classifier combination. In: Proceedings of the Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies, 2000, vol. 1, pp. 185–188. IEEE (2000)
30. Kuncheva, L.I.: Combining pattern classifiers: methods and algorithms (2004)
31. Lehn-Schiøler, T., Hegde, A., Erdogmus, D., Principe, J.C.: Vector quantization using information theoretic concepts. *Nat. Comput.* **4**(1), 39–51 (2005)
32. Levin, I.: Kdd-99 classifier learning contest: Lloft's results overview. *SIGKDD Explor.* **1**(2), 67–75 (2000)
33. Liu, H.: Feature selection at Arizona State University, Data Mining and Machine Learning Laboratory. <http://featureselection.asu.edu/index.php> (2010). Accessed Sept 2012
34. Liu, H., Setiono, R.: Chi2: feature selection and discretization of numeric attributes. In: Proceedings of the Seventh IEEE International Conference on Tools with Artificial Intelligence, November 5–8, 1995, pp. 388–391. IEEE Comput. Soc. (1995)
35. Liu, R., Yuan, B.: Multiple classifiers combination by clustering and selection. *Inf. Fusion* **2**(3), 163–168 (2001)
36. Martinez-Rego, D., Fontenla-Romero, O., Porto-Díaz, I., Alonso-Betanzos, A.: A new supervised local modelling classifier based on information theory. In: International Joint Conference on Neural Networks, 2009. IJCNN 2009, pp. 2014–2020. IEEE (2009)
37. Nock, R., Nielsen, F.: A real generalization of discrete adaboost. In: Proceeding of the 2006 Conference on ECAI 2006: 17th European Conference on Artificial Intelligence, August 29–September 1, 2006, Riva del Garda, Italy, pp. 509–515. IOS Press (2006)
38. Nutt, C.L., Mani, D.R., Betensky, R.A., Tamayo, P., Cairncross, J.G., Ladd, C., Pohl, U., Hartmann, C., McLaughlin, M.E., Batchelor, T.T.: Gene expression-based classification of malignant gliomas correlates better with survival than histological classification. *Cancer Res.* **63**(7), 1602–1610 (2003)
39. Petricoin III, E.F., Ardekani, A.M., Hitt, B.A., Levine, P.J., Fusaro, V.A., Steinberg, S.M., Mills, G.B., Simone, C., Fishman, D.A., Kohn, E.C., et al.: Use of proteomic patterns in serum to identify ovarian cancer. *Lancet* **359**(9306), 572–577 (2002)
40. Pomeroy, S.L., Tamayo, P., Gaasenbeek, M., Sturla, L.M., Angelo, M., McLaughlin, M.E., Kim, J.Y.H., Goumnerova, L.C., Black, P.M., Lau, C., et al.: Prediction of central nervous system embryonal tumour outcome based on gene expression. *Nature* **415**(6870), 436–442 (2002)
41. Porto-Díaz, I., Bolón-Canedo, V., Alonso-Betanzos, A., Fontenla-Romero, O.: Local modeling classifier for microarray gene-expression data. In: Artificial Neural Networks, ICANN 2010, pp. 11–20 (2010)
42. Porto-Díaz, I., Bolón-Canedo, V., Alonso-Betanzos, A., Fontenla-Romero, O.: A study of performance on microarray data sets for a classifier based on information theoretic learning. *Neural Netw.* **24**(8), 888–896 (2011)
43. Porto-Díaz, I., Martínez-Rego, D., Alonso-Betanzos, A., Fontenla-Romero, O.: Combining feature selection and local modelling in the kdd cup 99 dataset. In: Artificial Neural Networks, ICANN 2009, pp. 824–833 (2009)
44. Principe, J.C.: Information theoretic learning: Renyi's entropy and kernel perspectives. Springer (2010)
45. Principe, J.C., Xu, D., Zhao, Q., Fisher, J.W.: Learning from examples with information theoretic criteria. *J. VLSI Signal Process.* **26**(1), 61–77 (2000)
46. Quinlan, J.R.: Induction of decision trees. *Mach. Learn.* **1**(1), 81–106 (1986)
47. Quinlan, J.R.: C4. 5: programs for machine learning. Morgan kaufmann (1993)
48. Rastrigin, L.A., Erenstein, R.H.: Method of collective recognition. Energoizdat, Moscow (1981)

49. Saeys, Y., Inza, I., Larrañaga, P.: A review of feature selection techniques in bioinformatics. *Bioinformatics* **23**(19), 2507–2517 (2007)
50. Singh, D., Febbo, P.G., Ross, K., Jackson, D.G., Manola, J., Ladd, C., Tamayo, P., Renshaw, A.A., D'Amico, A.V., Richie, J.P., et al.: Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell* **1**(2), 203–209 (2002)
51. Spira, A., Beane, J.E., Shah, V., Steiling, K., Liu, G., Schembri, F., Gilman, S., Dumas, Y.M., Calner, P., Sebastiani, P., et al.: Airway epithelial gene expression in the diagnostic evaluation of smokers with suspect lung cancer. *Nature Med.* **13**(3), 361–366 (2007)
52. Tian, E., Zhan, F., Walker, R., Rasmussen, E., Ma, Y., Barlogie, B., Shaughnessy, Jr., J.D.: The role of the Wnt-signaling antagonist DKK1 in the development of osteolytic lesions in multiple myeloma. *N. Engl. J. Med.* **349**(26), 2483–2494 (2003)
53. van't Veer, V., Laura, J., Dai, H., Van de Vijver, M.J., He, Y.D., Hart, A.A.M.: Gene expression profiling predicts clinical outcome of breast cancer. *Nature* **415**(6871), 530–536 (2002)
54. Witten I.H., Frank, E.: Data mining: practical machine learning tools and techniques. Morgan Kaufmann Pub. <http://www.cs.waikato.ac.nz/ml/weka/> (2005). Accessed Sept 2012
55. Wolpert, D.H.: Stacked generalization. *Neural Netw.* **5**(2), 241–259 (1992)
56. Yang, Y., Webb, G.: Proportional k-interval discretization for naive-bayes classifiers. In: Machine Learning: ECML 2001, pp. 564–575 (2001)
57. Zhao, Z., Liu, H.: Searching for interacting features. In: Proceedings of the 20th International Joint Conference on Artificial intelligence, pp. 1156–1161. Morgan Kaufmann Publishers Inc. (2007)