Knowledge-Based Systems 54 (2013) 180–198

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Robust feature selection based on regularized brownboost loss

Pan Wei, Qinghua Hu *, Peijun Ma, Xiaohong Su

School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China

ARTICLE INFO

Article history:
Received 10 September 2012
Received in revised form 22 July 2013
Accepted 5 September 2013
Available online 24 September 2013

Keywords:
Feature selection
Margin
Robustness
Brownboost loss
Regularization

ABSTRACT

Feature selection is an important preprocessing step in machine learning and pattern recognition. It is also a data mining task in some real-world applications. Feature quality evaluation is a key issue when designing an algorithm for feature selection. The classification margin has been used widely to evaluate feature quality in recent years. In this study, we introduce a robust loss function, called Brownboost loss, which computes the feature quality and selects the optimal feature subsets to enhance robustness. We compute the classification loss in a feature space with hypothesis-margin and minimize the loss by optimizing the weights of features. An algorithm is developed based on gradient descent using L_2-norm regularization techniques. The proposed algorithm is tested using UCI datasets and gene expression datasets, respectively. The experimental results show that the proposed algorithm is effective in improving the classification robustness.

1. Introduction

The availability of information is increasing explosively, so the problem of focusing machine learning algorithms on the key information is becoming more and more important. Attribute reduction and feature selection are considered to be effective approaches for overcoming the problem of information overload. In general, attribute reduction aims to find a mapping from a high-dimensional space to a lower-dimensional space, while feature selection tries to pick a subset of features from the raw data. Feature selection has two advantages: it retains the original semantics of the selected features, which are useful for understanding the data, and it improves the modeling performance [1,2]. In recent years, feature selection techniques have been applied to gene data analysis [3,4], medical image processing [5], predictive analysis [6], and other applications.

It is well known that feature evaluation is a crucial aspect of the design of feature selection algorithms. The strategies used for feature evaluation are divided into two categories. The first category evaluates the candidate features directly based on their classification accuracy. However, these techniques are generally computationally expensive [1]. The second category evaluates the quality of different features using independent functions, including margin [7,8], consistency [9,10], mutual information [11–13], correlation [14,15], and dependency [16,17].

Margin is a representative independent function, which has become a hot research topic in recent years. Margin was first introduced by Vapnik for training support vector machines (SVM), where it maximizes the classification margin between different classes [18]. In 1999, Shawe Taylor and Cristianini stated the upper bound of the SVM generalization error and showed that the bound is related to the sample size and classification margin [19]. In 2002, Crammer et al. discussed the generalization error of the margin, which was used in AdaBoost, and showed that generalization error is independent of input dimension size, while the VC dimension grows with an increasing number of base classifiers [20]. In 2004, Gilad Bachrach et al. developed three feature selection methods based on margin and proved the infinite sample generalization bound for 1NN using the large margin theory [7]. In general, we desire samples that produce large margins so the classification has higher levels of confidence and reliability.

Many margin-based learning algorithms have been developed, including margin-based feature selection [7,8,21,22], classifier training [23,24] and ensemble learning [25–27]. All of these algorithms use margin-based classification loss functions to obtain optimal solutions. Classification loss functions decrease monotonically with the margin. When the margin of a sample is larger than zero, the sample is correctly classified and the classification loss is small; otherwise, it is misclassified and the loss is relatively large. There are several loss functions, including the hinge loss used in SVM training [18], the squared loss applied in regression and forecasting analysis [28,29], the Exponential Loss function [30,31] applied in AdaBoost, and the logistic loss used in regression learning and ensemble learning [32–34]. In 2001, Freund et al.

* Corresponding author. Tel.: +86 22 27401839.
E-mail address: huqinghua@tju.edu.cn (Q. Hu).

0950-7051/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.knosys.2013.09.005

recognized that the behavior of boosting methods is closely related to Brownian motion in a noisy environment and designed a robust boosting algorithm known as Brownboost, then derived the Brownboost loss function [26].

Loss functions can be used as the optimization objectives in classification and regression, so they can also be employed for evaluating and selecting features. Margin-based feature selection algorithms have been discussed extensively. In general, margin-based feature selection can be divided into three categories. The first method category maximizes the margin for feature selection directly, such as Relief [35], Simba [7], and ReliefF [36]. Relief and its extended algorithms compute the margin in the feature space and use the margin as the weights of the features. Simba possesses some improvements relative to Relief. It calculates the weight of each feature using gradient descent and adds the samples to update the weights iteratively.

The second class of algorithms minimize the margin-induced loss functions to compute the weights of features. In 1992, Tibshirani et al. proposed the least absolute shrinkage and selection operator (LASSO), which minimizes the sum of the squared residuals with a constraint on the L_1-norm of the coefficient vector [37]. Zhao et al. showed that LASSO was an efficient method for variable selection by jointly minimizing the empirical error and the L_1 penalty [38]. Previous studies [39,40] presented feature selection methods that obtain sparse solutions for LASSO-penalized logistic regression and applied them to SVMs. Kim et al. stated that the optimization problem of LASSO can be considered to be an extension of L_1 boosting and that both are consistent with learning theory. In addition, Chen et al. proposed a feature selection method that used linear programming, which was based on a maximum margin criterion, where the hinge loss and sample distances were combined to learn the weights of features [21]. Later, Hu et al. presented a method for sample selection based on the margin loss and applied it to a nearest neighbor classifier [22]. Pan et al. presented a large margin feature selection method based on Brownboost loss and L_1 regularization for SVM to obtain sparse feature weightings [41].

The third class trains the weights of features using a support vector machine. In 2002, Guyon et al. proposed a method for feature selection that utilized a support vector machine-based recursive feature elimination (SVM-RFE) method and good experimental results were obtained using gene data [42]. Recent variants or extensions of SVM-RFE included multiclass extensions of SVM-RFE [4], a variant of SVM-RFE that uses simulated annealing to eliminate a number of features at a time [43], and a two-stage feature selection algorithm based on SVM-RFE [44].

Thus, many margin-based feature selection methods have been proposed. However, the above mentioned methods quantify the margin based on convex losses, which has a drawback when applied to classification and regression. They use a large value to penalize large negative margins, which may make the algorithm sensitive to noise. Brownboost loss was introduced to overcome this drawback during boosting [26], which assigns an upper-bounded penalty to a sample with a large negative margin. Given that noise is widespread in real-world applications, robust methods for feature evaluation and selection are highly desirable. In this study, we developed a feature selection method based on Brownboost loss and L_2-norm regularization, and we compared its robustness with some classical methods. In our experiments, the comparative analysis showed that our proposed method was efficient and robust to attribute noise and classification noise.

The remainder of this paper is organized as follows. In Section 2, we introduce related studies that address margin-based feature selection. In Section 3, we explain the L_2-norm regularization-based Brownboost loss function and present a method for feature weight learning using gradient descent, which we compare with L_1-norm regularization. In Section 4, we present the experimental analysis. Finally, we conclude this study in Section 5.

2. Related work

Several margin-based feature selection algorithms have been developed and we introduce related algorithms.

2.1. Relief series

The Relief [35] algorithm is a feature selection method that maximizes the hypothesis-margin directly when estimating attributes. The key concept employed by the Relief algorithm is to estimate the quality of attributes based on how well their values can distinguish instances that are very similar. During each iteration, a sample x is selected randomly and the algorithm searches for its two nearest neighbors: one from the same class (the nearest hit or NH) and another from a different class (the nearest miss or NM). Next, the weight of the ith feature is updated, as follows:

$$w_i = w_i + \|x_i - NM(x)_i\|^2 - \|x_i - NH(x)_i\|^2, \tag{1}$$

where $\|x_i - NM(x)_i\|^2 - \|x_i - NH(x)_i\|^2$ is the hypothesis margin of sample x and $\|\cdot\|$ is the Euclidean distance.

However, Relief is not robust and the nearest neighbors defined in the original feature space are unlikely to be those in the weighted space. Thus, some improvements were developed, including Simba [7] and ReliefF [36].

ReliefF is a more robust algorithm for feature selection and it can deal with multi-class problems. Similar to Relief, ReliefF selects a random instance x and searches for k of its nearest neighbors from the same class, as well as k nearest neighbors from each of the different classes. It updates w based on the average contribution of the k nearest hits and the k nearest misses for all attributes, depending on their values for x.

Algorithm 1. ReliefF

set all weights $w \leftarrow (0, 0, \ldots, 0)$.
for $t = 1$ to N **do**
  select a random instance x from S.
  find k nearest hits H_t.
  for each class $C \neq class(x)$ **do**
    from class C find k nearest misses M_t.
  end for
  for attribute $i = 1$ to M **do**
    $$w_i \leftarrow w_i + \sum_{C \neq class(x)} \frac{P(C)}{1 - P(class(x))} \frac{\sum_{\forall \tilde{x} \in M_t} \|x_i - \tilde{x}_i\|^2}{k} - \frac{\sum_{\forall \tilde{x} \in H_t} \|x_i - \tilde{x}_i\|^2}{k}$$
  end for
end for

Simba estimates the weight of each feature based on gradient descent of the hypothesis margin with respect to the weights of the features for multi-class problems. During each iteration, the weights of the features are updated based on a randomly selected sample x:

$$\begin{aligned} w_i &= w_i + \frac{\partial \theta_h^w}{\partial w_i} \\ &= w_i + \frac{1}{2}\left(\frac{(x_i - NM(x)_i)^2}{\|x - NM(x)\|_w} - \frac{(x_i - NH(x)_i)^2}{\|x - NH(x)\|_w} \right) \cdot w_i, \end{aligned} \tag{2}$$

where θ_h^w is the hypothesis margin with respect to the weights of features, $\theta_h^w = \frac{1}{2}(\|x - NM(x)\|_w - \|x - NH(x)\|_w)$ and $\|z\|_w = \sqrt{\sum_i w_i^2 \cdot z_i^2}$.

Algorithm 2. Simba

1: initialize $w \leftarrow (1,1,\ldots,1)$
2: **for** $t = 1$ to T **do**
3: select a random instance x form S.
4: calculate $NM(x)$ and $NH(x)$ with respect to $S \backslash x$ and the weight vector w.
5: **for** $i = 1$ to N **do**
6: $\triangle_i \leftarrow \frac{1}{2}\left(\frac{(x_i - NM(x)_i)^2}{\|x - NM(x)\|_w} - \frac{(x_i - NH(x)_i)^2}{\|x - NH(x)\|_w}\right) \cdot w_i$
7: **end for**
8: $w \leftarrow w + \triangle$
9: **end for**
10: $w \leftarrow \frac{w^2}{\|w^2\|_\infty}$ where $(w^2)_i := (w_i)^2$

2.2. SVM-RFE methods

SVM-RFE [42] was developed based on SVMs for reducing the dimensions of gene data, where this method conducts gene selection using a backward elimination procedure. SVM-RFE was initially proposed for binary problems. The squared coefficients $w_j^2 (j = 1, 2, \ldots, p)$ of the weight vectors w are used as feature ranking criteria. This concept is derived from the Optimal Brain Damage (OBD) algorithm [45].

Consider a binary classification problem with training samples $\{x_i, y_i\}_{i=1}^n$, where $x_i \in \Re^d$ and $y_i \in \{+1, -1\}$. The objective function of a SVM is usually written as:

$$J = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} L(y_i f(x_i)), \tag{3}$$

where $f(x)$ is the decision function of a SVM, which has the form of $f(x) = w \cdot x + b$, $L(z)$ is the Hinge loss, and $L(z) = max(0, 1 - z)$, while $y_i f(x_i)$ is referred to as the sample margin of x_i.

If the jth feature is discarded in binary SVMs, we drop the offset b for the sake of simplicity. The criterion J can be expanded to a second-order Taylor series, as follows.

$$\Delta J(j) = \frac{\partial J}{\partial w_j}(\Delta w_j) + \frac{\partial^2 J}{\partial^2 w_j}(\Delta w_j)^2 + \Theta(\Delta w_j)^3. \tag{4}$$

In (4), the first-order term can be neglected, which becomes $\Delta J(j) \approx (\Delta w_j)^2$. If we denote the value of J by $J(j)$ after the jth feature is removed, it follows that:

$$J(j) \approx J + w_j^2. \tag{5}$$

Therefore, removing the feature with the smallest w_j^2 will lead to the lowest rise in J, which also increases the generalization performance.

In 2006, Zhou et al. presented the multi-class SVM-RFE (MSVM-RFE) algorithm [4], which is based on multiple binary SVMs and 'all-together' methods. This method starts with all of the features and removes one or a few features at each iteration. In addition, the coefficients of the weight vectors of linear SVMs are used to rank the features, before removing the features with the smallest score $c_i = \sum_r w_{ri}^2$, where w_i represents the corresponding ith feature in the weight vector and r is the class number.

Algorithm 3. Multi-class SVM-RFE (MSVM-RFE)

1: Initialize S for the full feature set: //S is the set of selected features;
2: p = the number of features in Set S;
3: **while** $p \neq m$ **do**
4: Train a multiclass SVM using the features in Set S;
5: $w_r = [w_{r1}, w_{r2}, \ldots, w_{rp}]^T$;
6: Calculate the ranking criteria for set $S : c_i = \sum_r w_{ri}^2$;
7: Identify the feature with the smallest ranking criterion;
8: Remove the identified feature from set S;
9: p = the number of features in set S
10: **end while**
11: return feature set S;

2.3. LASSO methods

LASSO is the least absolute shrinkage and selection operator for linear regression, which minimizes the sum of squared loss with a constraint on the L_1-norm of the coefficient vector [37]. Thus, LASSO is a computationally feasible method for variable selection and sparse learning. LASSO estimators solve the following optimization problem.

$$\min_{\alpha 1,\ldots,\alpha_m} \quad \frac{1}{2}\sum_i \left(y_i - \sum_{j=1}^{m}\alpha_i x_{ij}\right)^2, \quad s.t. \sum_{j=1}^{N}|\alpha_j| \leqslant t,\ t \geqslant 0. \tag{6}$$

The objective function of LASSO is not smooth, so special optimization techniques are necessary. Tibshirani et al. used the quadratic program (QP) for least square regressions [37]. Osborne et al. proposed a faster QP algorithm for LASSO [46]. In 2003, Kim and Kim proposed a gradient descent algorithm for LASSO, which was based on L_1 boosting for large datasets [47]. They also noted that the LASSO optimization problem can be transformed to minimize $\sum_i \left(1 - \sum_{j=1}^m y_i^T \alpha_i x_{ij}\right)^2$ subject to $\sum_{j=1}^N |\alpha_j| \leqslant t, t \geqslant 0$ and by considering $y_i^T \alpha_i x_{ij}$ as the margin in boosting algorithms.

Logistic regression has been discussed widely in the context of classification and regression [39], and it has a direct probabilistic interpretation. One of the advantages of logistic regression is that it provides the user with explicit probabilities for classification, apart from the class label information. Moreover, it can be readily extended to the multi-category classification problem.

Logistic regression can be formulated as the following optimization problem based on the LASSO penalty:

$$\min_\beta \rho = \sum_i g(-y_i f(x_i)), s, t \quad \sum_{j=0}^{n}|\beta_j| \leqslant t. \tag{7}$$

where $t \geqslant 0$ is the tuned parameter, $f(x)$ is a linear model $\left(f(x_i) = \sum_{j=0}^n \beta_j x_{ij}\right)$, and the function g is given by:

$$g(\gamma) = \log(1 + \exp(\gamma)), \tag{8}$$

which is the negative log-likelihood function associated with the probabilistic model. This is usually referred to as the logistic loss in machine learning and $y_i f(x_i)$ is known as the sample margin of x_i.

Shevade et al. noted that the above problem is equivalent to the following unconstrained optimization problem using optimality conditions [39] and proposed an algorithm based on the Gauss–Seidel method for gene selection. Later, Liu et al. formulated the problem as the L_1-ball constrained smooth convex optimization and proposed the solution of the problem using Nesterovs method [40].

Thus, various loss functions have been used to search for the optimal classification functions during machine learning. We now summarize some of these loss functions.

The first is the squared loss $l(\theta) = (1-\theta)^2$, which is used for least squares regression. The second is the logistic loss $l(\theta) = \log(1 + exp(-\theta))$, which is used widely in regression analysis. The third is the hinge loss $l(\theta) = max(0, 1-\theta)$, which is used in SVMs. In addition, the Exponential Loss $l(\theta) = \exp(-\theta)$ is used in the Adaboost algorithm.

Some of these margin-based loss functions are not sufficiently robust to deal with noisy data, such as squared and Exponential Loss functions. Noise is widespread in real-world applications, so robust loss functions and algorithms are highly desirable.

3. Robust loss function and feature weight learning

3.1. Robust loss function

In this section, we introduce a robust loss function referred to as Brownboost loss, which is based on the boost-by-majority algorithm (BBM) and the infinite horizon concept [26].

Next, we describe the details of the Brownboost loss function. In our setup, the time variable range was $0 \leqslant t \leqslant 1$ and we denote the margin by θ, so we define the potential function as:

$$\begin{aligned} &\phi(\theta,t) = \frac{1}{2}\left\{1 - \mathbf{erf}\left(\frac{\theta-\alpha(t)}{\beta(t)}\right)\right\}, \\ &where\ \mathbf{erf}(a) = \frac{2}{\pi}\int_0^a \exp(-x^2)dx, \end{aligned} \tag{9}$$

erf is the error function, and $\alpha(t)$ and $\beta(t)$ are defined by the equations

$$\alpha(t) = (\eta - 2\rho)\exp(1-t) + 2\rho, \tag{10}$$

$$\beta(t) = \sqrt{(\beta_0^2+1)\exp(2(1-t)) - 1}, \tag{11}$$

where $\eta > 0, \beta_0^2 \geqslant 0$ and $\rho > 0$ are the parameters of the algorithm.

In Eqs. (10) and (11), $\alpha(t)$ and $\beta(t)$ do not depend on margin θ, so we can view them as positive integers and obtain the following equation:

$$\begin{aligned} &Loss_{Brownboost}(\theta) = \frac{1}{2}\{1 - \mathbf{erf}(p\theta + q)\}, \\ &where\ \mathbf{erf}(a) = \frac{2}{\pi}\int_0^a \exp(-x^2)dx, \end{aligned} \tag{12}$$

where $p = \frac{1}{\beta(t)}$ and $q = \frac{-\alpha(t)}{\beta(t)}$.

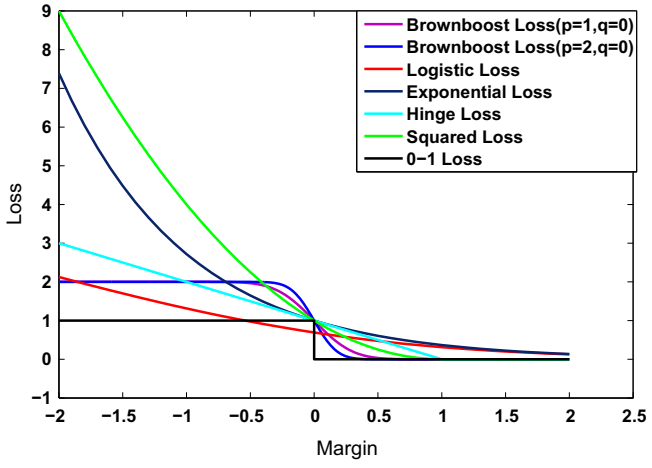

Fig. 1. Comparison of the classification loss.

The existing margin loss is the upper boundary on the generalization error of a zero-one loss. As shown in Fig. 1, the margin loss is a monotonic function that decreases with the margin. When the margin of a sample is more than zero, the sample will be classified correctly and the margin loss is a small value, otherwise it will be misclassified and the margin loss received a larger penalty value.

Clearly, the total loss will be large if most of the instances have negative margins. However, the Exponential Loss, hinge loss, and squared loss output are larger than the Brownboost loss if the instance has a large negative margin. In general, the Brownboost loss rejects instances that are located deep on the incorrect side of the boundary, which are unlikely to be classified correctly at the end. Thus, this method is expected to be more robust than the other functions. The Brownboost loss assigns the same value to penalize negative margins if they are far from the boundary.

3.2. Feature weight learning for the robust loss function

The margin-based Brownboost loss function allows us to minimize the loss of the hypothesis margin using feature weight learning.

To obtain the margin loss of a dataset, we provide the following definition.

Definition 1. Given a set of samples $S = \{x_1, x_2, \ldots, x_n\}$, we define the loss function as

$$L(S) = \frac{1}{n}\sum_{x_i \in S}\left(l\left(\theta_h^w(x)\right)\right), \tag{13}$$

where $L(S)$ is the average loss of S, $l(\theta_h^w(x))$ is the hypothesis margin of sample x with its weighted distance, and $\theta_h^w(x)$ is the hypothesis margin in terms of the distance.

Next, we present the objective function of the Brownboost loss for the sample set.

$$\psi(w) = \frac{1}{n}\sum_{x_i \in S}\frac{1}{2}\left\{1 - \frac{2}{\pi}\int_0^{p\theta_h^w(x)+q}\exp(-x^2)dx\right\} \tag{14}$$

In Eq. (14), we can see that the Brownboost loss is a non-convex function. The gradient descent method is the most efficient means of solving it, but sometimes it fails to find the best solution [37]. A popular and successful approach for statistical learning is the use of regularization penalties in the optimization function [38]. By jointly minimizing the loss function and penalty, we can search for a good and simple solution, which avoids large variations. L_1 and L_2 regularization are used widely at present. L_1 regularization penalizes the weight vector for its L_1-norm, whereas L_2 regularization uses its L_2-norm. In general, L_1 regularization has a significant advantage for sparse representation, but L_2 regularization may be more robust to outliers than L_1 regularization and it is more efficient [48]. In our proposed optimization function, we tested the use of a L_2 regularization penalty for robust feature weight learning and compared it with L_1 regularization.

First, we give the optimization function based on the L_2 regularization penalty.

$$\begin{aligned} &\phi_{L_2}(w) = \frac{1}{n}\sum_{x_i \in S}\frac{1}{2}\left\{1 - \frac{2}{\pi}\int_0^{p\theta_h^w(x)+q}\exp(-x^2)dx\right\} + \lambda\|w\|_2 \\ &s.t.\ w_i \geqslant 0, \\ &where\ \lambda > 0, \lambda\ is\ a\ tune\ parameter. \end{aligned} \tag{15}$$

This function can be differentiated to yield a feature weight vector, so we can use the gradient descent method to minimize $\phi_{L_2}(w)$. The gradient of $\phi_{L_2}(w)$ is evaluated as

Table 1
Datasets.

Number	Data	Samples	Features	Classes
1	Breast	84	9216	5
2	Crx	690	15	2
3	DLBCL	88	4026	6
4	German	1000	24	2
5	Iono	351	34	2
6	Leukemial	72	7129	3
7	Sick	2800	29	2
8	Sonar	208	60	2
9	Soybean	683	35	19
10	Spam	4601	57	2
11	SRBCT	88	2308	5
12	Wdbc	569	30	2
13	Wine	178	13	3
14	Zoo	101	16	7

$$\begin{aligned}(\nabla\phi_{L_2}(w))_i &= \frac{\partial\phi(w)}{\partial w} = \frac{\partial\psi}{\partial\theta_h^w}\cdot\frac{\partial\theta_h^w}{\partial w} + \lambda\nabla(\|w\|_2)_w \\ &= \frac{p}{2n\pi}\sum_{x\in S}\exp\left\{-(p\theta_h^w+q)^2\right\}\times\nabla(\theta_h^w) + \lambda\frac{w_i}{\|w\|_2} \\ \text{where } \nabla(\theta_h^w) &= w_i\times\left(\frac{x_i - NH(x_i)}{\|x-NH(x)\|_w} - \frac{x_i - NM(x_i)}{\|x-NM(x)\|_w}\right)\end{aligned} \tag{16}$$

Next, we give the optimization function based on the L_1 regularization penalty.

$$\phi_{L_1}(w) = \frac{1}{n}\sum_{x_i\in S}\frac{1}{2}\left\{1 - \frac{2}{\pi}\int_0^{p\theta_h^w(x)+q}\exp(-x^2)dx\right\} + \upsilon\|w\|_1 \tag{17}$$

$$s.t.\ w_i \geqslant 0$$

where $\upsilon > 0, \upsilon$ *is a regularization factor*.

In Eq. (17), when the weight is zero, $\|w\|_1$ is not smooth so L_1 regularization is considered to be a difficult problem [39,40,58]. Many researchers have proposed effective methods to address this problem, such as Gauss–Seidel [39], Grafting [49], Shooting [50], and Stochastic Gradient Descent (SGD) [51]. A previous study [41] used the SGD method to address this problem and obtained good results. This approach was based on the iterative shrinkage thresholding (IST) technique [52] where the smooth objective function in Eq. (14) converges to a minimum if the selected threshold υ is sufficiently large. The weight updating process is divided into two steps. First, the weight is updated without considering the L_1 penalty term and we can obtain $\bar{w}_i$. Next, the L_1 penalty is applied to the weight so it does not change the *sign* value. In other words, the weight is removed when it crosses zero. The weight updating procedure is as follows.

$$\bar{w}_i^{k+1} = w_i^k + \upsilon\rho_i\frac{\partial\psi(w)}{\partial w_i}$$

where

$$\frac{\partial\psi(w)}{\partial w_i} = \frac{p}{2n\pi}\sum_{x\in S}\exp\left\{-(p\theta_h^w+q)^2\right\}\times\nabla(\theta_h^w); \tag{18}$$

$$\nabla(\theta_h^w) = w_i\times\left(\frac{x_i - NH(x_i)}{\|x-NH(x)\|_w} - \frac{x_i - NM(x_i)}{\|x-NM(x)\|_w}\right)$$

$$w_{k+1} = \begin{cases} max(0, \bar{w}_i^{k+1} - \upsilon\rho_i) & \text{if } \bar{w}_i^{k+1} > 0, \\ min(0, \bar{w}_i^{k+1} + \upsilon\rho_i) & \text{if } \bar{w}_i^{k+1} < 0, \end{cases} \tag{19}$$

In Eqs. (18) and (19), $\upsilon > 0$,υ is a regularization factor and the weights of the features become smooth if υ is large, while ρ is the learning factor and k is the kth iteration.

3.3. Algorithms

Next, we compare two algorithms that minimize the regularized Brownboost loss functions and discuss their computational complexity. One is our proposed method based on L_2 regularization and the other is based on L_1 regularization, as described previously [41]. Both of these methods can cope with multi-class tasks.

In FWL-L_2 (Algorithm 4), we use the gradient descent method to solve the problem defined in (15). In Algorithm 4, we set γ_0 as a small constant, MaxIterNum is the number of iterations, and ϱ is the learning factor that determines how the parameters change during each iteration of the gradient descent method. In the inner loop from Step 6 to Step 13, we first compute the hypothesis margin of sample x, which is taken as the objective function in (16), before we use $\bigtriangledown(\phi_{L_2}(w))_i$ to respond to the change in the direction of the ith feature and update the weight of the ith feature. Finally, if the ith feature of sample x is less than zero, we set it as zero. When the stop condition ($\|\triangle w\| < \varepsilon,\varepsilon$) of 0.005 is

Table 2
1NN Performance comparison of classical methods using the raw datasets (%).

DataSet	InfoGain	Consistency	Simba	ReliefF	MSVM-RFE
Breast	98.8 ± 4.0(60)	87.5 ± 8.3(4)	98.8 ± 4.0(62)	97.1 ± 6.8(205)	100 ± 0.0(10)
Crx	80.6 ± 12.9(6)	79.7 ± 13.5(12)	79.6 ± 11.4(9)	79.6 ± 11.4(9)	81.9 ± 8.3(10)
DLBCL	97.3 ± 5.8(55)	84.0 ± 12.1(4)	100 ± 0.0(191)	98.3 ± 5.4(76)	100 ± 0.0(12)
German	70.6 ± 1.5(2)	66.0 ± 3.4(15)	71.2 ± 2.8(10)	70.0 ± 1.2(1)	70.1 ± 3.7(8)
Iono	91.2 ± 5.4(7)	88.4 ± 5.6(7)	90.7 ± 5.2(18)	91.8 ± 6.5(7)	92.4 ± 3.7(8)
Leukemial	97.2 ± 5.7(24)	81.1 ± 12.5(4)	98.6 ± 4.5(158)	97.3 ± 5.7(22)	100 ± 0.0(16)
Sick	97.3 ± 1.1(6)	96.6 ± 1.3(9)	96.6 ± 1.4(11)	97.3 ± 0.9(11)	97.2 ± 1.4(7)
Sonar	87.5 ± 7.3(44)	82.2 ± 5.1(14)	87.6 ± 6.8(24)	88.5 ± 6.8(15)	88.9 ± 4.7(51)
Soybean	93.6 ± 4.1(28)	88.0 ± 4.4(14)	91.1 ± 4.6(35)	92.7 ± 4.1(32)	94.5 ± 4.3(22)
Spam	89.7 ± 3.0(33)	87.0 ± 3.1(25)	88.6 ± 3.9(57)	88.6 ± 3.9(57)	90.8 ± 2.4(18)
SRBCT	81.9 ± 17.9(223)	71.3 ± 12.8(6)	88.8 ± 13.9(187)	84.7 ± 17.6(56)	91.4 ± 10.9(38)
Wdbc	96.0 ± 2.9(20)	94.2 ± 3.7(7)	96.8 ± 2.5(25)	96.3 ± 2.5(10)	96.1 ± 2.7(5)
Wine	97.6 ± 4.3(8)	95.5 ± 4.4(5)	98.3 ± 2.7(7)	96.5 ± 5.0(11)	97.6 ± 4.3(7)
Zoo	95.4 ± 8.4(11)	92.1 ± 9.4(5)	95.4 ± 8.4(15)	95.4 ± 8.4(15)	95.4 ± 8.4(10)
Ave.	91.1(37.6)	85.3(9.4)	91.6(57.8)	91.0(37.6)	92.6(15.9)

Table 3
1NN Performance comparison of margin-based techniques using the raw datasets (%).

DataSet	Logistic-LASSO	LASSO	Exponential Loss	BBL (FWL-L_2)	BBL (FWL-L_1)
Breast	93.8 ± 10.6(30)	89.2 ± 12.9(103)	99.2 ± 2.6(93)	100 ± 0.0(106)	98.8 ± 4.0(76)
Crx	79.6 ± 13.7(13)	78.8 ± 11.6(13)	78.7 ± 11.2(15)	84.0 ± 9.7(10)	76.6 ± 13.8(7)
DLBCL	98.0 ± 4.2(199)	96.1 ± 5.2(65)	100 ± 0.0(255)	100 ± 0.0(52)	98.3 ± 5.3(55)
German	70.6 ± 3.3(2)	70.0 ± 3.6(1)	70.0 ± 1.2(1)	72.6 ± 3.3(17)	70.2 ± 1.3(5)
Iono	91.8 ± 5.1(8)	86.4 ± 6.8(7)	91.0 ± 4.0(16)	92.6 ± 3.4(14)	91.8 ± 5.7(12)
Leukemial	97.3 ± 5.7(8)	95.9 ± 6.6(36)	100 ± 0.0(158)	100 ± 0.0(49)	98.8 ± 4.0(41)
Sick	96.3 ± 0.9(23)	95.6 ± 0.7(22)	96.2 ± 1.1(24)	97.5 ± 0.8(9)	97.1 ± 1.4(4)
Sonar	87.1 ± 7.6(60)	87.1 ± 7.6(60)	87.5 ± 4.5(26)	91.4 ± 7.7(33)	88.9 ± 6.4(31)
Soybean	91.2 ± 4.4(26)	91.1 ± 4.6(34)	93.7 ± 4.3(31)	95.0 ± 3.4(19)	94.8 ± 3.8(15)
Spam	88.6 ± 3.8(52)	88.6 ± 3.8(57)	88.6 ± 3.6(36)	89.2 ± 2.1(34)	88.6 ± 3.9(57)
SRBCT	79.7 ± 16.8(11)	83.2 ± 20.1(14)	70.1 ± 15.7(280)	93.3 ± 9.4(108)	88.0 ± 14.9(87)
Wdbc	96.7 ± 2.9(9)	95.4 ± 2.9(29)	96.8 ± 2.6(24)	97.2 ± 1.9(11)	96.5 ± 3.2(15)
Wine	96.6 ± 3.1(7)	94.9 ± 5.1(13)	98.3 ± 2.8(7)	98.9 ± 2.3(6)	98.3 ± 3.7(7)
Zoo	95.4 ± 8.4(16)	95.4 ± 8.4(16)	95.4 ± 8.3(11)	96.4 ± 8.3(8)	95.4 ± 8.4(16)
Ave.	90.2(33.1)	89.1(33.6)	90.4(69.8)	93.4(34)	91.6(30)

Table 4
SVM-RBF performance comparison of classical methods using the raw datasets (%).

DataSet	InfoGain	Consistency	Simba	ReliefF	MSVM-RFE
Breast	96.3 ± 6.1(54)	77.9 ± 9.2(4)	96.7 ± 5.5(33)	91.7 ± 8.4(77)	100 ± 0.0(9)
Crx	85.5 ± 18.5(1)	84.1 ± 17.5(12)	85.5 ± 18.5(2)	85.5 ± 18.4(1)	85.6 ± 18.5(5)
DLBCL	98.0 ± 4.2(28)	81.0 ± 13.2(4)	92.6 ± 7.1(21)	95.0 ± 3.3(42)	100 ± 0.0(13)
German	75.9 ± 4(8)	74.5 ± 2.5(15)	74.6 ± 3.7(7)	76.0 ± 4.7(18)	76.1 ± 3.8(8)
Iono	95.8 ± 3.6(23)	92.6 ± 3.7(7)	95.2 ± 3.8(14)	95.7 ± 3.4(16)	94.9 ± 3.9(30)
Leukemial	97.3 ± 5.7(31)	74.5 ± 7.3(4)	97.3 ± 5.7(15)	98.6 ± 4.5(9)	100 ± 0.0(14)
Sick	93.9 ± 0.1(1)	93.9 ± 0.1(9)	93.9 ± 0.1(1)	93.9 ± 0.1(1)	94.0 ± 0.2(16)
Sonar	87.0 ± 6.8(49)	82.3 ± 7.0(14)	88.9 ± 5.7(39)	88.5 ± 6.1(55)	87.5 ± 6.9(31)
Soybean	93.6 ± 3.7(28)	90.8 ± 3.6(14)	90.4 ± 4.7(34)	93.6 ± 3.7(30)	93.9 ± 4.6(11)
Spam	92.1 ± 2.9(57)	90.0 ± 2.3(25)	92.2 ± 2.8(56)	92.1 ± 2.9(57)	92.2 ± 2.8(50)
SRBCT	82.1 ± 26.8(51)	62.5 ± 18.7(6)	87.3 ± 16.9(40)	82.5 ± 25.2(33)	94.4 ± 8.1(22)
Wdbc	98.1 ± 2.3(26)	96.5 ± 2.6(7)	98.1 ± 2.3(30)	98.1 ± 2.3(23)	98.1 ± 2.2(16)
Wine	98.9 ± 2.3(12)	97.2 ± 4.0(5)	98.9 ± 2.3(6)	98.9 ± 2.3(13)	99.4 ± 1.8(9)
Zoo	95.4 ± 8.4(12)	87.4 ± 11.4(5)	94.4 ± 8.4(15)	94.4 ± 8.4(13)	95.4 ± 8.4(6)
Ave.	92.1(27.2)	84.7(9.4)	91.9(22.4)	91.8(27.7)	93.7(17.1)

Table 5
SVM-RBF performance comparison of margin-based techniques using the raw datasets (%).

DataSet	Logistic-LASSO	LASSO	Exponential Loss	BBL (FWL-L_2)	BBL (FWL-L_1)
Breast	93.8 ± 10.6(22)	83.1 ± 10.5(46)	96.7 ± 5.5(41)	99.2 ± 2.6(31)	100 ± 0.0(22)
Crx	85.5 ± 18.5(3)	85.2 ± 18.3(12)	85.5 ± 18.5(2)	85.5 ± 18.5(3)	85.5 ± 18.5(5)
DLBCL	98.0 ± 4.2(23)	90.0 ± 9.4(27)	93.6 ± 5.7(21)	99.0 ± 3.2(25)	97.0 ± 4.8(31)
German	76.4 ± 3.3(16)	74.0 ± 3.6(23)	73.4 ± 2.6(24)	77.0 ± 2.2(11)	74.8 ± 3.5(12)
Iono	95.2 ± 4.2(25)	94.9 ± 4.2(28)	95.2 ± 3.8(15)	96.0 ± 3.6(18)	95.2 ± 3.8(28)
Leukemial	97.3 ± 5.7(10)	94.4 ± 7.3(48)	97.5 ± 5.4(32)	100 ± 0.0(17)	97.3 ± 5.7(9)
Sick	94.0 ± 0.2(22)	93.9 ± 0.1(1)	93.9 ± 0.2(20)	94.0 ± 0.8(14)	93.9 ± 0.2(8)
Sonar	88.9 ± 7.2(50)	88.0 ± 7.9(50)	89.0 ± 6.4(35)	89.9 ± 4.7(40)	88.9 ± 7.2(43)
Soybean	91.5 ± 5.9(20)	91.7 ± 4.7(25)	94.2 ± 3.7(27)	95.0 ± 3.4(24)	94.2 ± 3.8(28)
Spam	92.2 ± 2.8(52)	92.1 ± 3.0(55)	92.2 ± 2.9(56)	92.2 ± 2.8(53)	92.1 ± 2.9(57)
SRBCT	83.3 ± 22.4(14)	85.4 ± 19.2(36)	77.6 ± 24.3(23)	88.0 ± 9.5(29)	81.0 ± 23.5(6)
Wdbc	98.1 ± 2.2(17)	98.1 ± 2.2(30)	98.1 ± 2.3(21)	98.1 ± 2.2(27)	98.1 ± 2.3(25)
Wine	98.9 ± 2.3(13)	98.9 ± 2.3(13)	99.4 ± 1.8(7)	99.4 ± 2.3(6)	98.9 ± 2.3(6)
Zoo	94.4 ± 8.4(13)	94.4 ± 8.4(12)	94.4 ± 8.3(10)	95.4 ± 8.4(7)	95.4 ± 8.4(8)
Ave.	92.0(21)	90.3(29)	91.5(23.9)	93.5(21.8)	92.3(20.6)

satisfied, the algorithm quits the outer loop and returns the weights of the features. In this case, we consider the weights when the features converge.

Note that the most complex operations are to select the nearest hit (NH) and miss (NM) because we have to compute the distances between x and all other instances, which require $\Theta(NM)$ comparisons. The computational complexity of Algorithm 4 is $\Theta(tNM)$, where N is the number of features, M is the size of the sample set S, and t is the number of iterations. Algorithm 4 will converge within a finite number of steps.

Algorithm 4. Feature weight learning using gradient descent and L_2 norm Regularization (FWL-L_2)

procedure FeatureWeightLearningGD$L_2(S,p,q,MaxIterNum)$
  initialize feature vector $w \leftarrow (1,1,\ldots,1)^T$
  for $t = 1$ to *MaxIterNum* **do**
    $w_0 \leftarrow w$;
    $\forall x \in S$, find $NM(x)$ and $NH(x)$;
    for $i = 1$ to N **do**
      $\theta_h^w(x) \leftarrow \frac{1}{2}(\|x - NM(x)\|_w - \|x - NH(x)\|_w)$;
      $\phi_{L2}^x(w) \leftarrow \frac{1}{2}\left\{1 - \frac{2}{\pi}\int_0^{p\theta_h^w(x)+q} \exp(-x^2)dx\right\} + \frac{\lambda}{M}\|w\|_2$;
      $w_i \leftarrow w_i - \gamma_0 \varrho \frac{\partial \phi_{L2}^x(w)}{\partial w_i}$;
      if $w_i < 0$ **then**
        $w_i \leftarrow 0$;
      end if
    end for
    $w \leftarrow \frac{w}{\|w\|_\infty}$;
    $\Delta w \leftarrow w - w_0$;
    if $\|\Delta w\| < \varepsilon$ **then**
      break;
    end if
  end for
  rank the features in descending order according to w;
  return w
end procedure

In FWL-L_1 (Algorithm 5), we use the SGD method to solve the problems defined in (18) and (19). MaxIterNum is the number of iterations and υ is the regularization factor, which determines the sparseness degree of the weights. In general, we set υ as a positive constant. ρ is the learning factor that determines how the parameters change during each iteration of the gradient descent method. In this study, we set $\rho_i = \rho_0 e^{-i}$ as the learning factor, where i is proportional to the number of samples. From Step 6 to Step 9 (see Algorithm 5), Eq. (18) is used to update the weight of the ith feature. From Step 10 to Step 14 (see Algorithm 5), we use the threshold $\upsilon \cdot \rho_i$ to adjust the weights of the features in (19). The computational complexity of Algorithm 5 (FWL-L_1) is $\Theta(tNM)$, where N is the number of features, M is the number of samples S, and t is the number of iterations.

Algorithm 5. Feature weight learning using gradient descent and L_1-norm regularization (FWL-L_1)

procedure FeatureWeightLearning$L_1(S,p,q,\upsilon,MaxIterNum)$
  initialize feature vector $w \leftarrow (1,1,\ldots,1)^T$;
  for $t = 1$ to *MaxIterNum* **do**
    $\forall x \in S$, find $NM(x), NH(x)$;
    for $i = 1$ to N **do**
      $\theta_h^w(x) \leftarrow \frac{1}{2}(\|x - NM(x)\|_w - \|x - NH(x)\|_w)$;
      $\psi^x(w) \leftarrow \frac{1}{2}\left\{1 - \frac{2}{\pi}\int_0^{p\theta_h^w(x)+q} \exp(-x^2)dx\right\}$;
      $w_i \leftarrow w_i - \upsilon * \rho_i \frac{\partial \psi^x(w)}{\partial w_i}$;
      $\bar{w}_i \leftarrow w_i$;
      if $\bar{w}_i \geqslant 0$ **then**
        $w_i \leftarrow max(0, \bar{w}_i - \upsilon \cdot \rho_i)$;
      else if $\bar{w}_i < 0$ **then**
        $w_i \leftarrow min(0, \bar{w}_i + \upsilon \cdot \rho_i)$;
      end if
      if $w_i < 0$ **then**
        $w_i \leftarrow 0$;
      end if
    end for
  end for
  return w
end procedure

Based on comparisons with the algorithms described in Section 2, we found that our proposed algorithms are categories of filter models for feature selection. Filter techniques are computationally simple and fast, while they are also independent of the classification algorithm so they can easily be scaled to very high-dimensional datasets.

4. Experimental analysis

4.1. Data description

In the experiments, we use ten UCI datasets [53] and four gene datasets to test the performance of the algorithms. The number of samples in the datasets ranged from tens to thousands, while the feature dimensions ranged from dozens to over 9000. A summary of these datasets is provided in Table 1.

Next, we provide detailed information about the gene expression data. Breast [54] was extracted from the database as a single table, where each row, column, and cell represents an array element, a hybridization, and the observed fluorescent ratio for the array element in the appropriate hybridization, respectively. This table contained 9216 rows and 84 columns. DLBCL [55] was a dataset used to record 88 measurements related to diffuse large B-cell lymphoma. This dataset contained 4026 array elements. Leukemial [56] was a collection of 72 expression measurements, which contained a training set of 27 samples related to acute lymphoblastic leukemia (ALL), 11 samples related to acute myeloblastic leukemia (AML), and an independent test set that contained 20 ALL and 14AML samples, where each sample was analyzed using 7129 probes of 6817 human genes. SRBCT [57] covered five different childhood tumors where the similarities in their appearance during routine histology make the correct clinical diagnosis extremely challenging.

4.2. Experimental methods and parameter setting

We compared our techniques with five representative algorithms based on the margin, i.e., MSVM-RFE [4], ReliefF [35], Simba [7], LASSO [37], Logistic-LASSO (Logistic loss using the LASSO penalty) [40], and Exponential Loss. We also compared our methods with InfoGain [11] and Consistency [9], which are used for feature selection. InfoGain is a feature selection method based on information entropy. Consistency is a feature selection technique that tries to retain the discriminatory power of the data defined by its original features. Our proposed method is referred to as BBL (FWL-L_2) and BBL (FWL-L_1) [41].

All of the methods were implemented using MATLAB 2010b. The implementations were run on an Intel i5 2.8 GHz CPU machine with 4 GB of memory and a Windows 7 32-bit operating system. In the experiment, we first learned the weights of the features using the methods described above, before ranking the features in descending order and adding the candidate features one by one. After adding a new feature, we used SVM and 1NN to compute the classification accuracy. Finally, the features with the highest accuracy were selected.

In our proposed method, the two parameters p and q were critical for the classification performance. Thus, we conducted external cross-validation using a sequence of given values for p and q (p had values from [1,5], and q was set as zero). In FWL-L_2, we set λ as 0.02, but for FWL-L_1, we simply selected the features with weights greater than zero and used the cross-validation method interval to obtain values of υ in the range [0.001, 0.08].

4.3. Experimental setup

The objective of our experiment was to compare the classification accuracy, data reduction rate, and the robustness of the methods. To test the robustness of the algorithms, we consider the attribute noise and classification noise. We generated class noises by randomly relabeling the class labels of some samples and added the samples as noisy data in the raw data. We also generate attribute noise by adding Gaussian noise to the raw data. In the experiments, we add 5% and 10% noise, and generate ten datasets with different noise levels, before computing the average classification accuracy. We also used SVM-RBF and 1NN to compute the classification accuracy for the raw data and the reduced data.

In the feature selection task, two main criteria were used to assess the quality of feature selection methods, apart from the time complexities of the algorithms. These were the classification accuracy and the number of features. In general, a feature selection method is considered to be better if it has a fewer number of

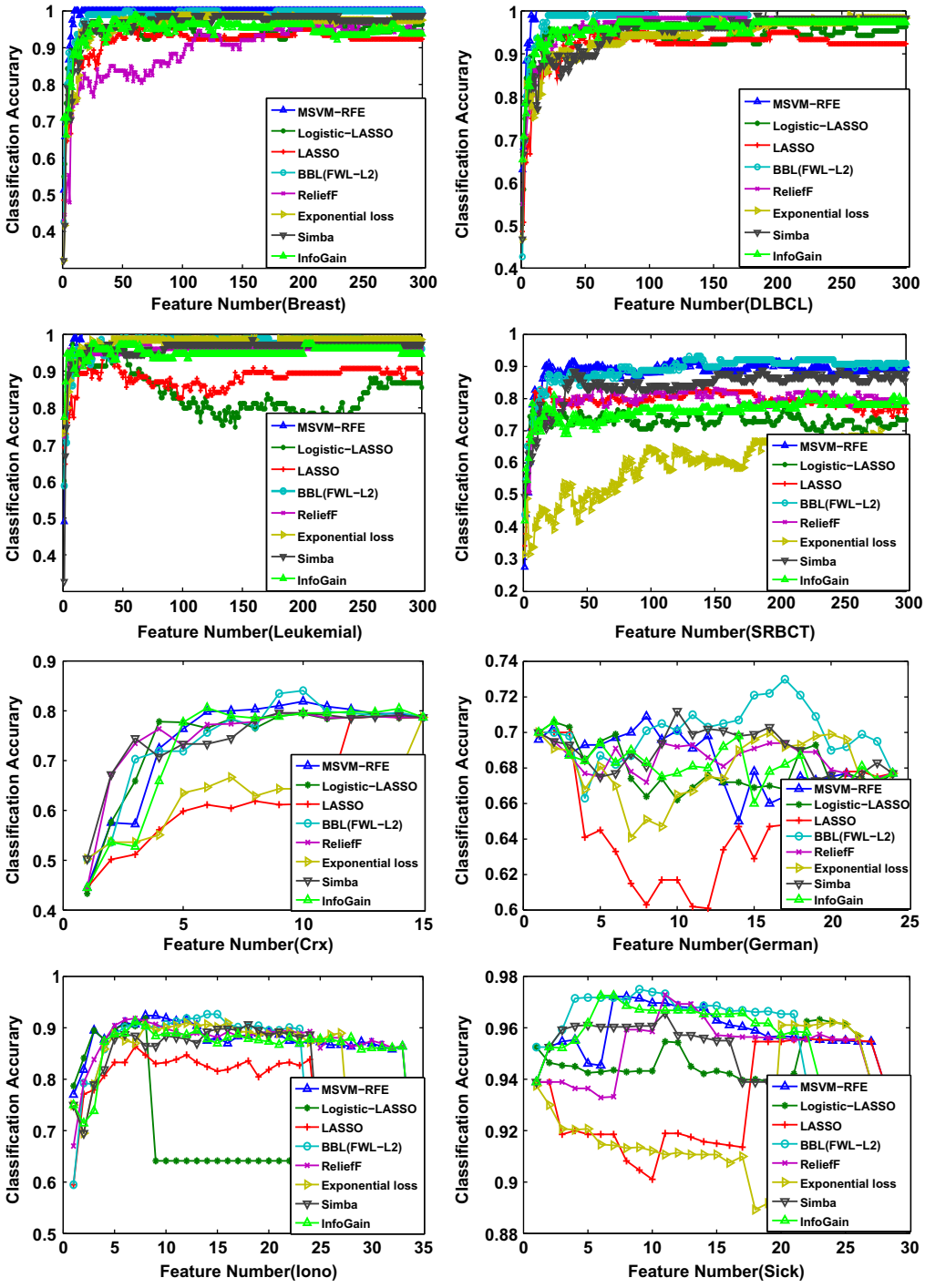

Fig. 2. 1NN classification accuracy.

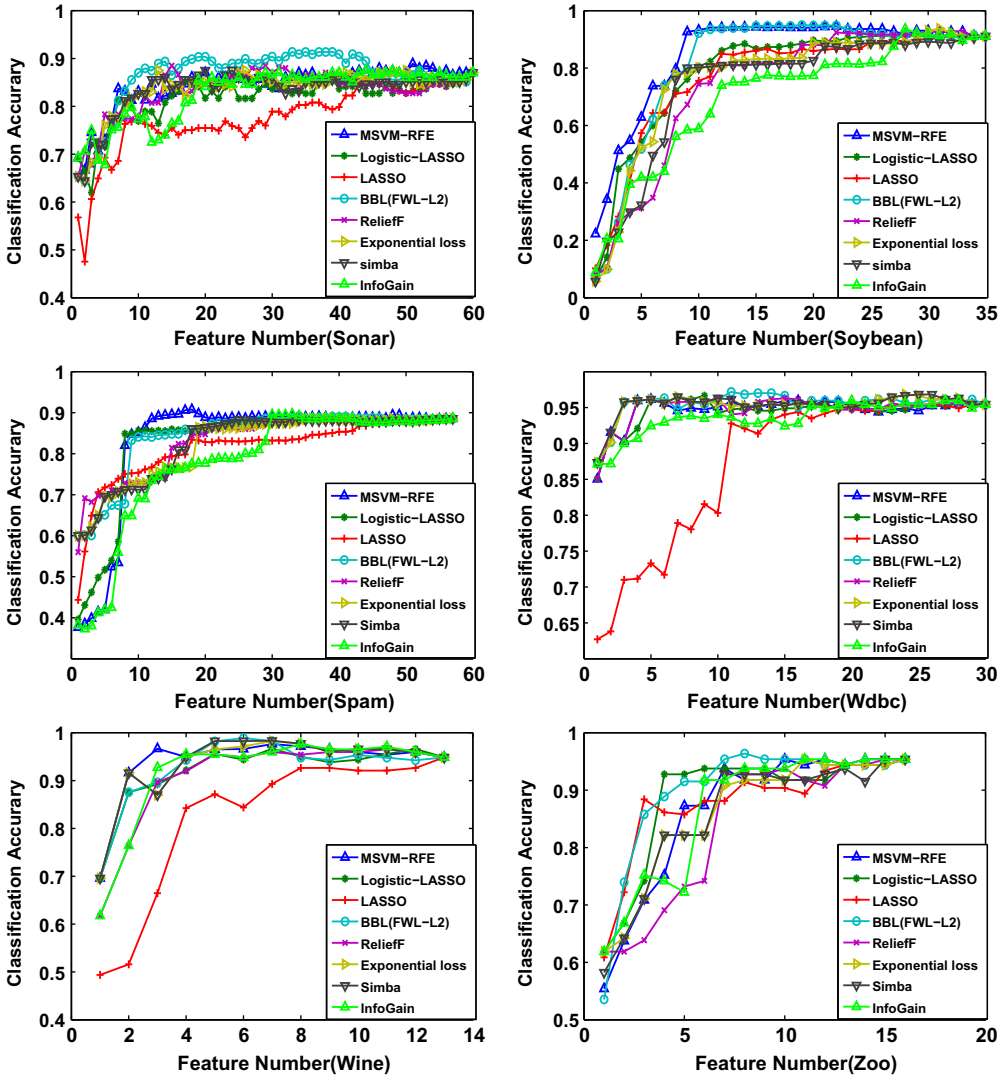

Fig. 2 (*continued*)

features and it obtains a higher level of classification precision compared with other methods. In addition, the variation in the classification accuracy can be measured based on the quality of robustness with different noise levels. It is very easy to know that a lower level of variation indicates the higher robustness of the algorithm as the noise level increases. In summary, these three criteria were used as the corresponding measure in our experiments. In general, if a feature selection method selects a fewer number of features and obtains a higher level of classification precision compared with other methods, it also has a lower variation in its classification accuracy so it can be considered to be a better method. Our final goal was classification, so a method was considered to perform badly if it had lower classification accuracy than other methods.

To describe the noise type in the experiments, C indicated the class noise and F represented the attribute noise. To measure the changes in the classification accuracy with different noise levels, $\triangle_1$ represented the difference between the raw datasets and the 5% noisy data while $\triangle_2$ was the difference between the results obtained with the raw data and the 10% noisy data.

4.4. Results and discussion

This section presents an analysis of the experimental results. First, we compared the performance of the feature selection methods using the raw data. The comparative results with 1NN are shown in Tables 2 and 3. The results show that BBL (FWL-L_2) performed much better than the other methods in terms of the classification accuracy. It was 0.8% higher compared with the other methods. Moreover, the improvement was as high as 8.1% compared with Consistency. However, Consistency selected the least features. MSVM-RFE selected fewer than the other methods apart form Consistency in terms of the number of features, but Simba, Exponential Loss and BBL (FWL-L_1) selected more than the other methods.

We compared the performance of the SVM-RBF classifiers using the raw data. As shown in Tables 4 and 5, BBL (FWL-L_2) delivered a better performance than the other methods in terms of the classification accuracy and the number of features selected. However, BBL (FWL-L_2) and MSVM-RFE had almost the same performance, and Consistency still had the worst performance as the SVM-RBF classifier, but the other methods do not exist remarkable differences.

To illustrate this problem further, we present the accuracy curves for different algorithms where variable numbers of features were selected from the raw data, which are shown in Figs. 2 and 3. The gene expression data had high dimensionality, so we only considered the first 300 features in the figures. The two figures show that BBL (FWL-L_2) had the highest classification accuracy and it obtained fewer

features than the other methods, except with the gene data. MSVM-RFE was most effective for gene data classification, while LASSO delivered the worst performance with some datasets.

We compared the performance of 1NN in a noisy environment. Tables 6 and 7 show the results with class noise, which demonstrate that the classification accuracy of each algorithm declined with the noise levels. However, the classification accuracy of BBL (FWL-L_2) declined much more slowly than that of the other methods. Using $C\triangle_1$ and $C\triangle_2$, we can see that BBL (FWL-L_2) had the same performance as ReliefF. Both of them obtain the best performance, which was at least 0.7% lower than Simba, Logistic-LASSO and MSVM-RFE. Simba, Logistic-LASSO and MSVM-RFE ranked second. They were at best 0.8% lower than the other methods. Consistency had the worst performance and it was 3.3% higher than BBL (FWL-L_2). Furthermore, in terms of classification accuracy and the number of features, BBL (FWL-L_2) had the highest classification accuracy and it obtained fewer features than the other methods. MSVM-RFE ranked second because it and BBL (FWL-L_2) had the same reduction ability but BBL (FWL-L_2) was at least 1.3% higher than MSVM-RFE in terms of the classification accuracy. Simba,

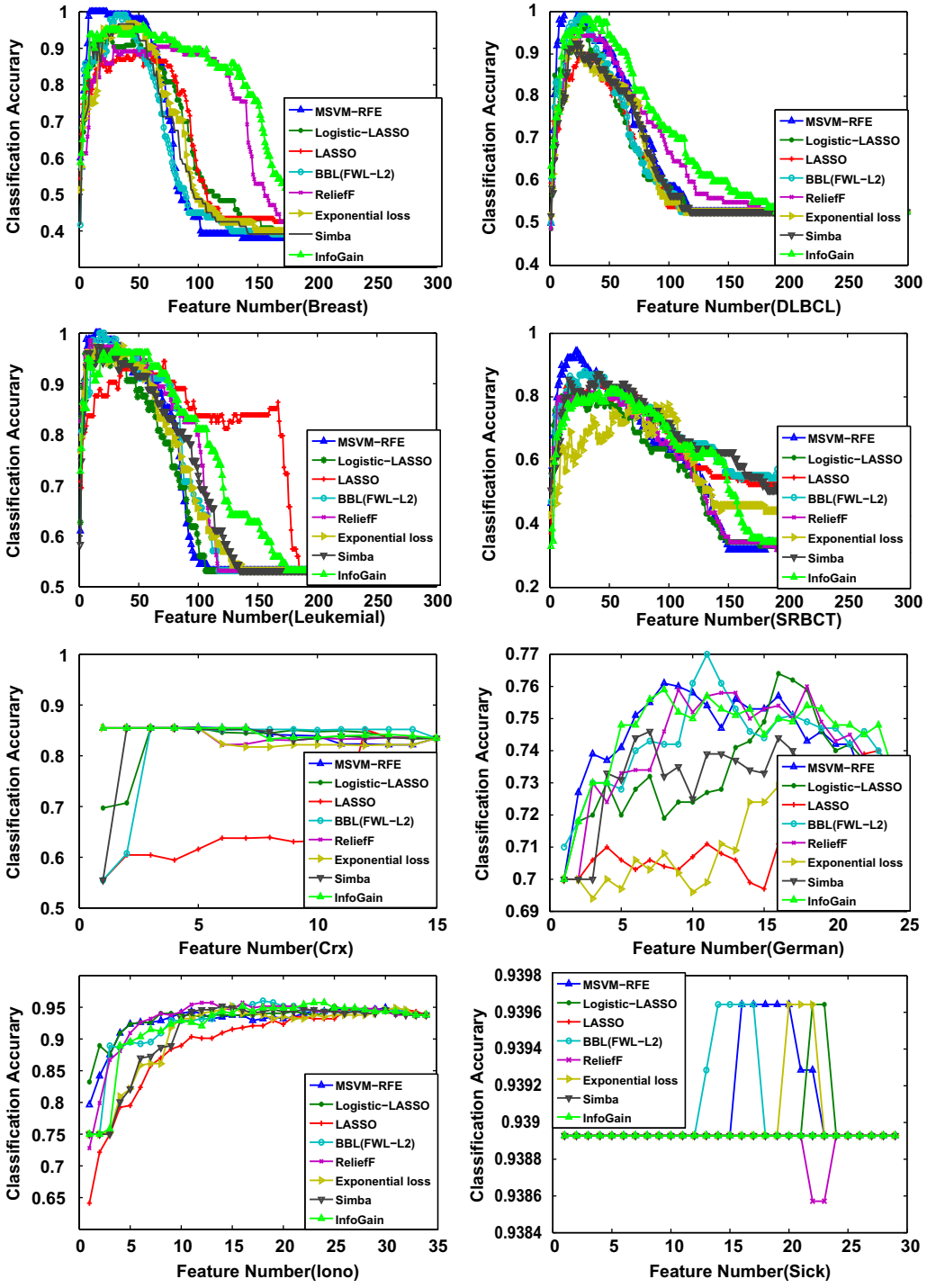

Fig. 3. SVM-RBF classification accuracy.

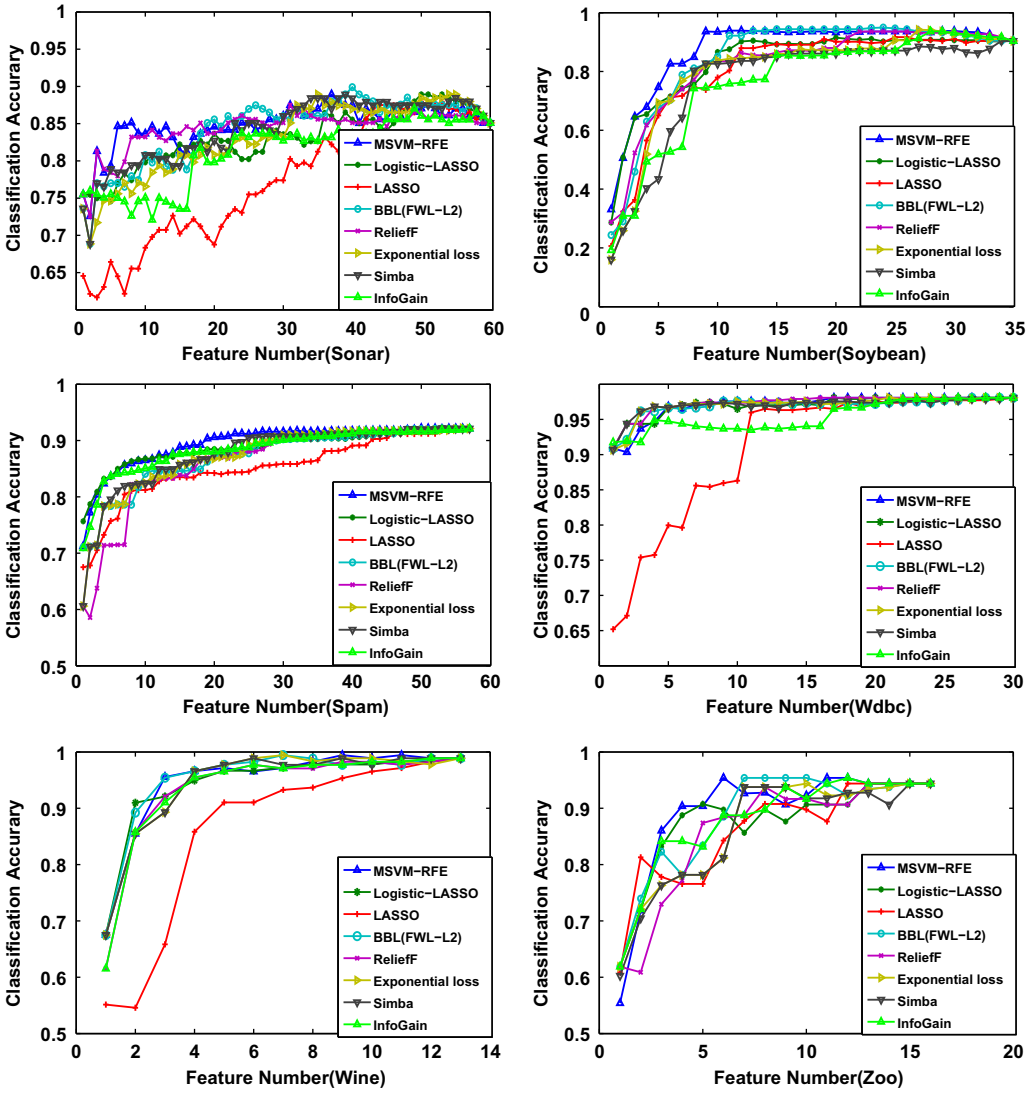

Fig. 3 (*continued*)

Table 6
1NN performance comparison for classical methods with class noisy data (%).

DataSet	Noise	InfoGain	Consistency	Simba	ReliefF	MSVM-RFE
Breast	0%	98.8 ± 4.0(60)	87.5 ± 8.3(4)	98.8 ± 4.0(62)	97.1 ± 6.8(205)	100 ± 0.0(10)
	C5%	91.3 ± 11.9(29)	83.1 ± 11.8(5)	94.1 ± 11.6(142)	96.8 ± 19.9(145)	97.5 ± 7.9(75)
	C10%	90.5 ± 15.8(154)	76.1 ± 9.2(5)	90.0 ± 9.9(217)	95.8 ± 12.4(211)	96.6 ± 7.5(42)
Crx	0%	80.6 ± 12.9(6)	79.7 ± 13.5(12)	79.6 ± 11.4(9)	79.6 ± 11.4(9)	81.9 ± 8.3(10)
	C5%	77.9 ± 16.4(13)	76.2 ± 15.2(11)	77.3 ± 11.9(9)	77.3 ± 11.9(9)	76.6 ± 16.5(10)
	C10%	76.4 ± 18.4(7)	65.1 ± 16.8(1)	75.5 ± 19.1(11)	74.5 ± 18.2(13)	74.3 ± 19.0(13)
DLBCL	0%	97.3 ± 5.8(55)	84.0 ± 12.1(4)	100 ± 0.0(191)	98.3 ± 5.4(76)	100 ± 0.0(12)
	C5%	93.4 ± 7.7(181)	80.3 ± 10.6(7)	94.4 ± 5.9(245)	95.3 ± 6.8(68)	95.6 ± 5.7(51)
	C10%	89.1 ± 10.5(18)	78.2 ± 16.4(6)	92.3 ± 9.1(263)	94.6 ± 7.2(117)	94.6 ± 7.8(30)
German	0%	70.6 ± 1.5(2)	66.0 ± 3.4(15)	71.2 ± 2.8(10)	70.0 ± 1.2(1)	70.1 ± 3.7(8)
	C5%	69.3 ± 5.9(11)	61.4 ± 5.9(11)	69.0 ± 1.7(2)	69.0 ± 1.7(2)	69.7 ± 3.6(9)
	C10%	68.2 ± 4.4(4)	58.3 ± 5.8(9)	68.6 ± 2.8(2)	68.7 ± 2.8(1)	68.8 ± 4.8(11)
Iono	0%	91.2 ± 5.4(7)	88.4 ± 5.6(7)	90.7 ± 5.2(18)	91.8 ± 6.5(7)	92.4 ± 3.7(8)
	C5%	88.1 ± 10.3(8)	84.1 ± 12.6(13)	86.7 ± 11.5(10)	87.1 ± 10.1(9)	87.8 ± 9.3(11)
	C10%	82.9 ± 14.6(29)	82.1 ± 9.8(11)	83.2 ± 12.1(9)	85.5 ± 11.5(10)	84.2 ± 14.7(6)
Leukemial	0%	97.3 ± 5.7(24)	81.1 ± 12.5(4)	98.6 ± 4.5(158)	97.3 ± 5.7(22)	100 ± 0.0(16)
	C5%	91.9 ± 21.1(154)	73.8 ± 20.9(5)	93.0 ± 18.1(93)	91.8 ± 17.4(49)	97.5 ± 13.7(35)
	C10%	90.4 ± 17.8(15)	63.6 ± 13.4(4)	91.9 ± 21.1(142)	87.7 ± 18.2(65)	95.3 ± 18.5(144)
Sick	0%	97.3 ± 1.1(6)	96.6 ± 1.3(9)	96.6 ± 1.4(11)	97.3 ± 0.9(11)	97.2 ± 1.4(7)
	C5%	95.0 ± 2.0(24)	94.4 ± 3.2(12)	95.2 ± 2.6(6)	96.0 ± 3.1(11)	95.6 ± 4.2(10)

Table 6 (*continued*)

DataSet	Noise	InfoGain	Consistency	Simba	ReliefF	MSVM-RFE
	C10%	91.9 ± 6.7(24)	91.4 ± 8.3(12)	91.9 ± 6.1(26)	92.5 ± 9.1(11)	92.6 ± 7.6(21)
Sonar	0%	87.5 ± 7.3(44)	82.2 ± 5.1(14)	87.6 ± 6.8(24)	88.5 ± 6.8(15)	88.9 ± 4.7(51)
	C5%	85.4 ± 14.2(28)	79.0 ± 14.3(16)	84.9 ± 6.4(18)	87.2 ± 9.3(24)	84.5 ± 10.7(34)
	C10%	84.6 ± 12.6(57)	77.9 ± 14.9(13)	83.6 ± 10.5(14)	86.4 ± 14.5(29)	81.3 ± 16.6(47)
Soybean	0%	93.6 ± 4.1(28)	88.0 ± 4.4(14)	91.1 ± 4.6(35)	92.7 ± 4.1(32)	94.5 ± 4.3(22)
	C5%	89.9 ± 9.3(27)	85.9 ± 8.9(14)	87.2 ± 8.1(35)	88.6 ± 8.2(30)	90.3 ± 9.6(22)
	C10%	87.7 ± 14.3(27)	83.3 ± 13.6(15)	83.9 ± 13.8(35)	85.6 ± 14.1(30)	88.5 ± 14.3(20)
Spam	0%	89.7 ± 3.0(33)	87.0 ± 3.1(25)	88.6 ± 3.9(57)	88.6 ± 3.9(57)	90.8 ± 2.4(18)
	C5%	86.1 ± 9.3(57)	84.5 ± 8.7(26)	86.3 ± 8.9(50)	86.1 ± 9.4(56)	87.0 ± 8.8(23)
	C10%	84.8 ± 11.3(20)	83.4 ± 11.1(25)	84.7 ± 11.7(48)	84.6 ± 12.6(50)	85.2 ± 11.5(36)
SRBCT	0%	81.9 ± 17.9(223)	71.3 ± 12.8(6)	88.8 ± 13.9(187)	84.7 ± 17.6(56)	91.4 ± 10.9(38)
	C5%	78.9 ± 16.6(278)	66.8 ± 25.4(7)	85.5 ± 14.9(109)	79.5 ± 22.7(107)	87.6 ± 12.5(99)
	C10%	66.2 ± 22.8(203)	56.2 ± 17.7(9)	83.5 ± 12.7(223)	74.8 ± 27.3(129)	78.4 ± 13.5(124)
Wdbc	0%	96.0 ± 2.9(20)	94.2 ± 3.7(7)	96.8 ± 2.5(25)	96.3 ± 2.5(10)	96.1 ± 2.7(5)
	C5%	91.0 ± 15.1(30)	89.5 ± 14.1(10)	91.8 ± 15.4(22)	91.8 ± 13.8(8)	91.1 ± 15.1(30)
	C10%	88.4 ± 24.3(19)	87.5 ± 22.9(9)	89.1 ± 25.2(9)	89.2 ± 23.5(13)	89.1 ± 21.2(14)
Wine	0%	97.6 ± 4.3(8)	95.5 ± 4.4(5)	98.3 ± 2.7(7)	96.5 ± 5.0(11)	97.6 ± 4.3(7)
	C5%	90.5 ± 15.7(9)	90.5 ± 21.2(7)	91.1 ± 17.7(5)	89.5 ± 14.4(6)	91.0 ± 15.9(7)
	C10%	89.5 ± 22.5(10)	85.8 ± 12.1(7)	89.9 ± 16.4(7)	89.0 ± 18.9(5)	90.5 ± 16.5(8)
Zoo	0%	95.4 ± 8.4(11)	92.1 ± 9.4(5)	95.4 ± 8.4(15)	95.4 ± 8.4(15)	95.4 ± 8.4(10)
	C5%	91.1 ± 12.1(12)	90.0 ± 12.3(5)	91.1 ± 12.1(11)	91.1 ± 12.1(12)	91.1 ± 12.1(12)
	C10%	86.9 ± 10.3(16)	87.9 ± 11.9(7)	88.8 ± 10.5(13)	88.8 ± 8.9(14)	89.0 ± 11.2(6)
Ave.	0%	91.1(37.6)	85.3(9.4)	91.6(57.8)	91.0(37.6)	92.6(15.9)
	C5%	87.1(61.5)	81.4(10.6)	87.7(54.1)	87.7(38.3)	88.8(30.6)
	C10%	84.1(43.1)	76.9(9.5)	85.5(72.8)	85.6(49.9)	86.4(37.3)
	$C\triangle_1$	3.9	3.9	3.9	3.3	3.8
$\triangle$	$C\triangle_2$	7.0	8.4	6.1	5.4	6.2

Table 7
1NN performance comparison of margin-based techniques with class noisy data (%).

DataSet	Noise	Logistic-LASSO	LASSO	Exponential Loss	BBL (FWL-L_2)	BBL (FWL-L_1)
Breast	0%	93.8 ± 10.6(30)	89.2 ± 12.9(103)	99.2 ± 2.6(93)	100 ± 0.0(106)	98.8 ± 4.0(76)
	C5%	93.7 ± 8.8(40)	85.9 ± 21.7(115)	96.6 ± 8.8(106)	98.1 ± 5.9(86)	92.6 ± 15.8(46)
	C10%	90.5 ± 20.5(68)	83.8 ± 15.7(94)	93.8 ± 15.8(284)	96.6 ± 7.5(107)	88.8 ± 12.4(66)
Crx	0%	79.6 ± 13.7(13)	78.8 ± 11.6(13)	78.7 ± 11.2(15)	84.0 ± 9.7(10)	76.6 ± 13.8(7)
	C5%	76.5 ± 16.2(13)	76.3 ± 13.6(13)	76.2 ± 14.7(13)	78.1 ± 14.3(8)	73.8 ± 15.5(7)
	C10%	74.5 ± 19.6(9)	73.4 ± 19.5(12)	73.7 ± 19.1(10)	76.6 ± 17.8(10)	71.8 ± 18.1(8)
DLBCL	0%	98.0 ± 4.2(199)	96.0 ± 5.2(65)	100 ± 0.0(255)	100 ± 0.0(52)	98.3 ± 5.3(55)
	C5%	93.4 ± 7.7(127)	88.5 ± 10.4(161)	93.7 ± 11.9(283)	96.9 ± 5.1(61)	93.5 ± 5.7(46)
	C10%	89.1 ± 10.5(18)	78.2 ± 16.4(6)	92.3 ± 9.1(263)	94.6 ± 7.2(62)	94.6 ± 7.8(49)
German	0%	70.6 ± 3.3(2)	70.0 ± 3.6(1)	70.0 ± 1.2(1)	72.6 ± 3.2(17)	70.2 ± 1.3(5)
	C5%	69.6 ± 3.6(2)	68.9 ± 1.4(2)	69.6 ± 3.6(2)	70.7 ± 4.4(14)	68.8 ± 0.6(2)
	C10%	68.9 ± 2.5(3)	67.7 ± 4.2(1)	68.4 ± 3.2(1)	70.5 ± 6.8(14)	68.3 ± 3.5(4)
Iono	0%	91.8 ± 5.1(8)	86.4 ± 6.8(7)	91.0 ± 4.0(16)	92.6 ± 3.3(14)	91.8 ± 5.7(12)
	C5%	87.0 ± 8.6(11)	83.5 ± 10.4(7)	87.8 ± 12.2(6)	90.5 ± 8.9(5)	87.6 ± 7.3(7)
	C10%	84.2 ± 17.7(8)	80.0 ± 12.3(32)	84.9 ± 13.3(7)	88.3 ± 12.3(15)	84.7 ± 10.6(8)
Leukemial	0%	97.3 ± 5.7(8)	95.9 ± 6.6(36)	100 ± 0.0(158)	100 ± 0.0(49)	98.8 ± 4.0(41)
	C5%	93.1 ± 14.1(34)	89.3 ± 13.4(54)	94.3 ± 18.1(72)	95.6 ± 14.1(56)	91.9 ± 21.1(25)
	C10%	88.9 ± 17.5(39)	81.3 ± 14.0(40)	91.9 ± 21.1(47)	94.3 ± 13.8(87)	88.1 ± 18.0(86)
Sick	0%	96.3 ± 0.9(23)	95.6 ± 0.7(22)	96.2 ± 1.1(24)	97.5 ± 0.8(9)	97.1 ± 1.4(4)
	C5%	94.7 ± 3.1(25)	94.3 ± 2.4(19)	94.9 ± 2.9(25)	96.3 ± 2.2(11)	95.7 ± 3.2(4)
	C10%	91.9 ± 4.8(26)	91.7 ± 8.9(13)	92.1 ± 4.8(26)	92.1 ± 8.9(8)	91.8 ± 11.1(4)
Sonar	0%	87.1 ± 7.6(60)	87.1 ± 7.6(60)	87.5 ± 4.5(26)	91.4 ± 7.7(33)	88.9 ± 6.4(31)
	C5%	84.9 ± 9.1(38)	83.3 ± 13.4(60)	87.2 ± 4.7(16)	89.7 ± 9.3(20)	84.9 ± 8.9(14)
	C10%	82.2 ± 8.9(45)	81.2 ± 13.1(60)	83.3 ± 13.4(60)	88.5 ± 9.7(30)	83.7 ± 12.4(57)
Soybean	0%	91.2 ± 4.4(26)	91.1 ± 4.6(34)	93.7 ± 4.3(31)	95.0 ± 3.4(19)	94.8 ± 3.8(15)
	C5%	87.2 ± 8.3(30)	87.2 ± 8.1(35)	87.1 ± 7.9(30)	90.7 ± 9.0(20)	89.9 ± 9.4(20)
	C10%	84.2 ± 13.4(27)	84.0 ± 13.7(34)	84.3 ± 13.9(34)	88.7 ± 14.9(22)	88.0 ± 14.6(17)
Spam	0%	88.6 ± 3.8(52)	88.6 ± 3.8(57)	88.6 ± 3.6(36)	89.2 ± 2.1(34)	88.6 ± 3.9(50)
	C5%	86.2 ± 9.3(52)	86.1 ± 9.3(57)	86.4 ± 9.5(41)	86.4 ± 8.8(46)	86.1 ± 9.3(56)
	C10%	84.5 ± 12.4(55)	84.5 ± 12.4(56)	84.5 ± 12.6(53)	85.6 ± 11.6(36)	84.5 ± 12.5(53)
SRBCT	0%	79.7 ± 16.8(11)	83.2 ± 20.0(14)	70.1 ± 15.7(280)	93.3 ± 9.5(108)	88.0 ± 14.9(87)
	C5%	78.5 ± 19.2(89)	82.1 ± 13.7(59)	63.8 ± 24.5(103)	91.8 ± 9.4(86)	85.1 ± 17.4(75)
	C10%	70.5 ± 16.1(120)	74.7 ± 16.7(122)	53.2 ± 20.3(183)	85.7 ± 10.1(105)	76.9 ± 12.1(78)

(*continued on next page*)

Table 7 (*continued*)

DataSet	Noise	Logistic-LASSO	LASSO	Exponential Loss	BBL (FWL-L_2)	BBL (FWL-L_1)
Wdbc	0%	96.7 ± 2.6(9)	95.4 ± 2.9(29)	96.8 ± 2.6(24)	97.2 ± 1.9(11)	96.5 ± 3.2(15)
	C5%	91.5 ± 15.9(12)	91.2 ± 15.2(29)	91.7 ± 15.3(21)	92.3 ± 14.9(15)	92.2 ± 14.9(25)
	C10%	88.3 ± 23.1(14)	88.4 ± 24.3(27)	89.1 ± 23.4(22)	90.7 ± 21.7(12)	88.6 ± 26.2(10)
Wine	0%	96.7 ± 2.9(7)	94.9 ± 5.1(13)	98.3 ± 2.8(7)	98.9 ± 2.3(6)	98.3 ± 3.7(7)
	C5%	91.0 ± 16.4(7)	88.9 ± 15.2(13)	91.1 ± 17.7(5)	92.6 ± 16.0(6)	92.6 ± 15.9(7)
	C10%	89.0 ± 20.5(8)	86.4 ± 25.7(13)	89.9 ± 15.5(10)	91.6 ± 17.9(7)	89.9 ± 15.5(10)
Zoo	0%	95.4 ± 8.4(16)	95.4 ± 8.4(16)	95.4 ± 8.4(11)	96.4 ± 8.3(8)	95.4 ± 8.4(16)
	C5%	91.1 ± 12.1(12)	90.0 ± 12.3(5)	91.1 ± 12.1(11)	91.1 ± 12.1(12)	91.1 ± 12.1(12)
	C10%	86.9 ± 10.3(16)	87.9 ± 11.9(7)	88.8 ± 10.5(13)	88.8 ± 8.9(14)	89.0 ± 11.2(6)
Ave.	0%	90.2(33.1)	89.1(33.6)	90.4(69.8)	93.4(34)	91.6(30)
	C5%	87.0(34.6)	85.3(45.5)	86.5(52.6)	90.1(31.6)	87.3(25)
	C10%	84.0(40.9)	81.9(48.7)	82.8(71.2)	88.3(37.9)	84.5(33)
	C$\triangle_1$	3.2	3.8	3.9	3.3	4.3
$\triangle$	C$\triangle_2$	6.2	7.2	7.6	5.1	7.1

Table 8
1NN performance comparison of classical methods with attribute noisy data (%).

DataSet	Noise	InfoGain	Consistency	Simba	ReliefF	MSVM-RFE
Breast	0%	98.8 ± 4.0(60)	87.5 ± 8.3(4)	98.8 ± 4.0(62)	97.1 ± 6.8(205)	100 ± 0.0(10)
	F5%	96.3 ± 6.0(272)	73.3 ± 9.8(6)	98.6 ± 4.3(271)	95.8 ± 6.9(192)	100 ± 0.0(29)
	F10%	89.2 ± 7.1(150)	64.6 ± 14.3(8)	89.6 ± 7.2(166)	92.5 ± 8.7(243)	100 ± 0.0(40)
Crx	0%	80.6 ± 12.9(6)	79.7 ± 13.5(12)	79.6 ± 11.4(9)	79.6 ± 11.4(9)	81.9 ± 8.3(10)
	F5%	78.4 ± 12.2(14)	76.7 ± 14.8(1)	79.3 ± 11.8(12)	78.5 ± 11.1(8)	79.7 ± 12.2(3)
	F10%	74.6 ± 9.6(5)	71.1 ± 14.1(1)	73.8 ± 11.2(7)	76.8 ± 9.8(10)	76.1 ± 10.4(4)
DLBCL	0%	97.3 ± 5.8(55)	84.0 ± 12.1(4)	100 ± 0.0(191)	98.3 ± 5.4(76)	100 ± 0.0(12)
	F5%	97.0 ± 3.2(30)	81.4 ± 9.1(5)	99.0 ± 3.2(112)	98.3 ± 5.3(60)	100 ± 0.0(20)
	F10%	95.3 ± 6.2(93)	67.2 ± 7.6(10)	96.3 ± 4.4(244)	98.3 ± 5.3(160)	100 ± 0.0(36)
German	0%	70.6 ± 1.5(2)	66.0 ± 3.4(15)	71.2 ± 2.8(10)	70.0 ± 1.2(1)	70.1 ± 3.7(8)
	F5%	69.9 ± 3.8(10)	64.4 ± 3.3(5)	70.7 ± 3.3(15)	68.2 ± 4.2(18)	68.3 ± 4.3(13)
	F10%	66.2 ± 4.0(20)	62.9 ± 4.0(6)	66.5 ± 4.2(12)	65.3 ± 5.1(16)	64.5 ± 3.9(3)
Iono	0%	91.2 ± 5.4(7)	88.4 ± 5.6(7)	90.7 ± 5.2(18)	91.8 ± 6.5(7)	92.4 ± 3.7(8)
	F5%	87.8 ± 4.8(13)	83.3 ± 5.3(17)	86.1 ± 6.7(22)	87.8 ± 3.2(12)	89.5 ± 4.7(5)
	F10%	74.0 ± 7.9(6)	73.2 ± 5.7(10)	75.5 ± 7.2(27)	77.1 ± 7.4(26)	74.2 ± 10.6(29)
Leukemial	0%	97.3 ± 5.7(24)	81.1 ± 12.5(4)	98.6 ± 4.5(158)	97.3 ± 5.7(22)	100 ± 0.0(16)
	F5%	97.3 ± 5.7(62)	80.1 ± 11.1(4)	98.6 ± 3.9(104)	91.2 ± 0.9(25)	100 ± 0.0(23)
	F10%	97.3 ± 5.7(62)	80.1 ± 11.1(4)	98.5 ± 3.9(104)	91.2 ± 0.9(25)	100 ± 0.0(23)
Sick	0%	97.3 ± 1.1(6)	96.6 ± 1.3(9)	96.6 ± 1.4(11)	97.3 ± 0.9(11)	97.2 ± 1.4(7)
	F5%	90.9 ± 0.8(6)	90.0 ± 1.1(8)	91.1 ± 1.6(12)	91.2 ± 0.9(8)	90.8 ± 1.1(26)
	F10%	89.4 ± 1.3(24)	89.1 ± 1.5(8)	89.2 ± 2.1(28)	89.2 ± 1.6(26)	89.6 ± 1.6(17)
Sonar	0%	87.5 ± 7.3(44)	82.2 ± 5.1(14)	87.6 ± 6.8(24)	88.5 ± 6.8(15)	88.9 ± 4.7(51)
	F5%	84.2 ± 10.0(58)	75.5 ± 7.7(10)	86.9 ± 9.5(55)	86.9 ± 9.8(37)	84.6 ± 7.4(42)
	F10%	69.8 ± 11.6(36)	58.6 ± 12.8(6)	75.5 ± 10.1(37)	73.1 ± 9.4(22)	70.7 ± 7.8(20)
Soybean	0%	93.6 ± 4.1(28)	88.0 ± 4.4(14)	91.1 ± 4.6(35)	92.7 ± 4.1(32)	94.5 ± 4.3(22)
	F5%	90.9 ± 4.7(34)	81.6 ± 6.6(14)	89.2 ± 4.2(29)	91.7 ± 3.9(30)	92.4 ± 4.1(31)
	F10%	72.1 ± 6.1(30)	73.2 ± 8.2(25)	75.4 ± 9.6(21)	73.1 ± 7.8(30)	75.9 ± 7.3(21)
Spam	0%	89.7 ± 3.0(33)	87.0 ± 3.1(25)	88.6 ± 3.9(57)	88.6 ± 3.9(57)	90.8 ± 2.4(18)
	F5%	75.1 ± 3.8(43)	64.3 ± 4.4(34)	65.9 ± 4.0(36)	66.4 ± 5.0(42)	66.7 ± 4.8(16)
	F10%	55.1 ± 22.3(44)	54.3 ± 23.8(57)	55.5 ± 21.3(55)	55.6 ± 23.4(51)	56.0 ± 23.4(53)
SRBCT	0%	81.9 ± 17.9(223)	71.3 ± 12.8(6)	88.8 ± 13.9(187)	84.7 ± 17.6(56)	91.4 ± 10.9(38)
	F5%	80.1 ± 15.1(13)	64.0 ± 17.1(8)	86.9 ± 14.4(245)	84.7 ± 17.7(54)	91.0 ± 12.7(86)
	F10%	60.5 ± 20.8(197)	51.1 ± 20.7(13)	78.1 ± 22.3(258)	83.5 ± 19.6(98)	87.6 ± 12.7(168)
Wdbc	0%	96.0 ± 2.9(20)	94.2 ± 3.7(7)	96.8 ± 2.5(25)	96.3 ± 2.5(10)	96.1 ± 2.7(5)
	F5%	89.3 ± 5.6(8)	87.1 ± 4.5(7)	91.0 ± 7.1(12)	89.3 ± 6.7(9)	91.0 ± 7.1(12)
	F10%	63.5 ± 10.2(8)	62.3 ± 9.3(10)	65.8 ± 8.2(9)	65.2 ± 7.9(12)	65.2 ± 8.0(12)
Wine	0%	97.6 ± 4.3(8)	95.5 ± 4.4(5)	98.3 ± 2.7(7)	96.5 ± 5.0(11)	97.6 ± 4.3(7)
	F5%	90.5 ± 15.7(9)	90.5 ± 21.2(7)	91.1 ± 17.7(5)	89.5 ± 14.4(6)	91.0 ± 15.9(7)
	F10%	89.5 ± 22.5(10)	85.8 ± 12.1(7)	89.9 ± 16.4(7)	89.0 ± 18.9(5)	90.5 ± 16.5(8)
Zoo	0%	95.4 ± 8.4(11)	92.1 ± 9.4(5)	95.4 ± 8.4(15)	95.4 ± 8.4(15)	95.4 ± 8.4(10)
	F5%	91.1 ± 12.1(12)	90.0 ± 12.3(5)	91.1 ± 12.1(11)	91.1 ± 12.1(12)	91.1 ± 12.1(12)
	F10%	86.9 ± 10.3(16)	87.9 ± 11.9(7)	88.8 ± 10.5(13)	88.8 ± 8.9(14)	89.0 ± 11.2(6)
Ave.	0%	91.1(37.6)	85.3(9.4)	91.6(57.8)	91.0(37.6)	92.6(15.9)
	F5%	87.4(43.2)	78.8(9.6)	87.9(67.4)	86.9(37)	88.7(23.3)
	F10%	77.3(59.6)	68.8(12.9)	79.0(73.3)	79.5(55.5)	80.9(34.6)
	F$\triangle_1$	3.7	6.5	3.7	4.1	3.9
$\triangle$	F$\triangle_2$	13.8	16.5	12.6	11.5	11.7

Table 9
1NN performance comparison of margin-based techniques with attribute noisy data (%).

DataSet	Noise	Logistic-LASSO	LASSO	Exponential Loss	BBL (FWL-L_2)	BBL (FWL-L_1)
Breast	0%	93.8 ± 10.6(30)	89.2 ± 12.9(103)	99.2 ± 2.6(93)	100 ± 0.0(106)	98.8 ± 4.0(76)
	F5%	92.7 ± 5.5(53)	87.9 ± 13.7(103)	95.3 ± 9.2(237)	100 ± 0.0(110)	98.6 ± 4.2(86)
	F10%	90.8 ± 10.5(80)	87.6 ± 6.0(246)	85.0 ± 11.5(259)	93.8 ± 7.9(83)	91.7 ± 8.3(56)
Crx	0%	79.6 ± 13.7(13)	78.8 ± 11.6(13)	78.7 ± 11.2(15)	84.0 ± 9.7(10)	76.6 ± 13.8(7)
	F5%	79.0 ± 12.5(5)	78.7 ± 12.1(13)	76.7 ± 9.9(12)	82.2 ± 9.9(11)	75.9 ± 13.7(6)
	F10%	76.0 ± 10.6(7)	70.9 ± 8.6(15)	75.9 ± 8.9(9)	77.8 ± 11.0(7)	72.9 ± 10.7(5)
DLBCL	0%	98.0 ± 4.2(199)	96.0 ± 5.2(65)	100 ± 0.0(255)	100 ± 0.0(52)	98.3 ± 5.3(55)
	F5%	96.3 ± 6.2(160)	96.0 ± 5.2(154)	98.3 ± 5.3(229)	100 ± 0.0(83)	98.3 ± 5.3(70)
	F10%	93.3 ± 7.7(257)	89.6 ± 9.5(248)	95.6 ± 5.9(182)	100 ± 0.0(64)	96.7 ± 10.5(47)
German	0%	70.6 ± 3.3(2)	70.0 ± 3.6(1)	70.0 ± 1.2(1)	72.6 ± 3.2(17)	70.2 ± 1.3(5)
	F5%	67.7 ± 3.4(17)	67.2 ± 2.9(23)	69.0 ± 3.3(16)	71.8 ± 5.4(13)	67.1 ± 3.6(24)
	F10%	65.0 ± 5.1(4)	62.4 ± 3.2(11)	67.4 ± 5.3(10)	67.6 ± 3.3(10)	60.6 ± 4.1(24)
Iono	0%	91.8 ± 5.1(8)	86.4 ± 6.8(7)	91.0 ± 4.0(16)	92.6 ± 3.3(14)	91.8 ± 5.7(12)
	F5%	86.9 ± 6.4(4)	83.8 ± 6.2(30)	87.0 ± 6.8(15)	89.2 ± 4.5(10)	85.8 ± 7.2(19)
	F10%	75.3 ± 8.3(31)	73.9 ± 5.4(34)	77.3 ± 6.1(28)	78.4 ± 9.1(20)	75.5 ± 4.7(22)
Leukemial	0%	97.3 ± 5.7(8)	95.9 ± 6.6(36)	100 ± 0.0(158)	100 ± 0.0(49)	98.8 ± 4.0(41)
	F5%	97.1 ± 9.1(22)	93.4 ± 7.0(111)	98.8 ± 3.9(61)	100 ± 0.0(51)	98.8 ± 4.0(73)
	F10%	96.1 ± 6.3(43)	92.5 ± 5.3(213)	93.6 ± 9.1(126)	93.1 ± 6.3(60)	93.2 ± 11.5(44)
Sick	0%	96.3 ± 0.9(23)	95.6 ± 0.7(22)	96.2 ± 1.1(24)	97.5 ± 0.8(9)	97.1 ± 1.4(4)
	F5%	90.4 ± 1.3(24)	91.1 ± 1.5(27)	90.6 ± 1.5(26)	91.6 ± 1.2(13)	90.3 ± 1.8(4)
	F10%	89.6 ± 1.0(27)	89.7 ± 1.5(27)	89.6 ± 1.0(27)	89.7 ± 1.4(7)	89.9 ± 0.9(29)
Sonar	0%	87.1 ± 7.6(60)	87.1 ± 7.6(60)	87.5 ± 4.5(26)	91.4 ± 7.7(33)	88.9 ± 6.4(31)
	F5%	84.2 ± 12.3(53)	84.2 ± 13.2(55)	85.5 ± 11.2(36)	90.9 ± 6.6(39)	88.0 ± 9.8(49)
	F10%	68.7 ± 8.3(50)	64.9 ± 15.0(54)	74.1 ± 10.7(37)	78.8 ± 8.4(34)	71.2 ± 10.2(27)
Soybean	0%	91.2 ± 4.4(26)	91.1 ± 4.6(34)	93.7 ± 4.3(31)	95.0 ± 3.4(19)	94.8 ± 3.8(15)
	F5%	90.2 ± 4.8(32)	89.8 ± 4.4(34)	92.5 ± 4.0(28)	93.7 ± 2.1(22)	92.1 ± 4.6(23)
	F10%	72.6 ± 7.1(32)	71.4 ± 7.2(35)	72.4 ± 6.6(27)	77.3 ± 8.6(28)	71.9 ± 5.8(35)
Spam	0%	88.6 ± 3.8(52)	88.6 ± 3.8(57)	88.6 ± 3.6(36)	89.2 ± 2.1(34)	88.6 ± 3.9(50)
	F5%	65.9 ± 2.8(11)	63.7 ± 3.7(56)	66.1 ± 4.3(49)	65.5 ± 3.4(48)	65.2 ± 3.7(46)
	F10%	55.9 ± 1.2(25)	55.3 ± 2.8(51)	54.7 ± 4.1(55)	55.3 ± 3.1(50)	55.3 ± 2.8(52)
SRBCT	0%	79.7 ± 16.8(11)	83.2 ± 20.0(14)	70.1 ± 15.7(280)	93.3 ± 9.5(108)	88.0 ± 14.9(87)
	F5%	79.2 ± 21.7(128)	81.5 ± 19.2(147)	69.4 ± 14.5(295)	92.4 ± 8.1(111)	87.7 ± 14.8(84)
	F10%	69.1 ± 24.2(214)	70.7 ± 17.6(297)	68.6 ± 15.2(265)	86.0 ± 12.2(106)	85.8 ± 13.5(86)
Wdbc	0%	96.7 ± 2.6(9)	95.4 ± 2.9(29)	96.8 ± 2.6(24)	97.2 ± 1.9(11)	96.5 ± 3.2(15)
	F5%	92.8 ± 5.9(25)	91.9 ± 3.2(30)	93.3 ± 2.9(16)	93.7 ± 3.2(10)	92.5 ± 3.3(9)
	F10%	81.7 ± 5.6(20)	78.2 ± 6.5(30)	80.7 ± 5.8(23)	82.4 ± 4.1(21)	78.4 ± 7.7(12)
Wine	0%	96.7 ± 2.9(7)	94.9 ± 5.1(13)	98.3 ± 2.8(7)	98.9 ± 2.3(6)	98.3 ± 3.7(7)
	F5%	91.0 ± 5.8(9)	88.7 ± 8.1(13)	91.0 ± 7.1(12)	92.1 ± 4.8(10)	91.0 ± 7.1(12)
	F10%	68.6 ± 10.9(10)	61.3 ± 9.1(13)	65.8 ± 9.4(9)	68.6 ± 11.0(10)	64.5 ± 9.1(8)
Zoo	0%	95.4 ± 8.4(16)	95.4 ± 8.4(16)	95.4 ± 8.4(11)	96.4 ± 8.3(8)	95.4 ± 8.4(16)
	F5%	94.4 ± 9.1(13)	94.4 ± 9.3(14)	95.4 ± 8.4(11)	95.4 ± 8.4(10)	95.4 ± 8.4(12)
	F10%	91.7 ± 9.6(11)	90.4 ± 8.3(16)	92.7 ± 9.9(12)	92.9 ± 9.8(10)	92.8 ± 9.9(12)
Ave.	0%	90.2(33.1)	89.1(33.6)	90.4(69.8)	93.4(34)	91.6(30)
	F5%	86.3(39.7)	85.2(57.9)	86.4(74.5)	89.9(38.6)	87.6(37)
	F10%	78.2(57.9)	75.6(92.1)	78.1(76.4)	81.6(36.4)	78.6(33)
	F$\triangle_1$	3.9	4.0	4.0	3.5	4.0
$\triangle$	F$\triangle_2$	12.0	13.5	12.3	11.8	13.0

LASSO, Exponential Loss and Consistency showed worse performance compare with InfoGian, Logistic-LASSO and BBL (FWL-L_1).

Next, we compare the performance of 1NN in terms of the attribute noise. The results are described in Tables 8 and 9. Using F$\triangle_2$, it be see that the performances of BBL (FWL-L_2), ReliefF and MSVM-RFE were similar in terms of their robustness. These three ranked the highest and they were slightly better than Simba, Logistic-LASSO, and Exponential Loss. There were no significant differences between InfoGain and LASSO, and these two techniques were worse than Simba, Logistic-LASSO, and Exponential Loss. Consistency remained the worst. In terms of classification accuracy and reduction ability, BBL (FWL-L_2) showed much better performance than the other methods and MSVM-RFE rank second. They were at least 0.8% higher than the other methods in terms of their classification accuracy. Furthermore, they used fewer features than other methods, with the exception Consistency, which ranked the worst.

Next, we compared the performance of the SVM-RBF classifier in a noisy environment. The results with class noise are given in Tables 10 and 11. Using C$\triangle_1$ and C$\triangle_2$, we found that BBL (FWL-L_2) was slightly lower than the other methods. Simba performed worse than the others, but there were no obvious differences among the other methods. In terms of the classification accuracy and reduction ability, BBL (FWL-L_2) and MSVM-RFE delivered almost the same performance and they were at least 0.9% higher than the other methods in terms of the classification accuracy. Consistency and Exponential Loss performed poorly and the rest of methods delivered almost the same performance.

Tables 12 and 13 show the results when attribute noise was added. Using C$\triangle_1$ and C$\triangle_2$, it show that MSVM-RFE performed

Table 10
SVM-RBF performance comparison of classical methods with class noisy data (%).

DataSet	Noise	InfoGain	Consistency	Simba	ReliefF	MSVM-RFE
Breast	0%	96.3 ± 6.1(54)	77.9 ± 9.2(4)	96.7 ± 5.5(33)	91.7 ± 8.4(77)	100 ± 0.0(9)
	C5%	92.5 ± 8.7(18)	74.9 ± 11.7(5)	91.3 ± 13.2(38)	88.0 ± 12.9(74)	97.5 ± 7.9(17)
	C10%	90.9 ± 11.5(58)	70.1 ± 14.8(5)	89.8 ± 11.2(44)	85.6 ± 15.3(93)	94.1 ± 11.5(15)
Crx	0%	85.5 ± 18.5(1)	84.1 ± 17.5(12)	85.5 ± 18.5(2)	85.5 ± 18.4(1)	85.6 ± 18.5(5)
	C5%	81.6 ± 21.3(1)	79.0 ± 20.0(11)	81.6 ± 21.3(2)	81.6 ± 21.3(1)	81.6 ± 21.3(1)
	C10%	79.5 ± 26.6(5)	78.9 ± 27.8(1)	78.9 ± 27.8(2)	78.9 ± 27.8(1)	79.6 ± 26.3(5)
DLBCL	0%	98.0 ± 4.2(28)	81.0 ± 13.2(4)	92.6 ± 7.1(21)	95.0 ± 3.3(42)	100 ± 0.0(13)
	C5%	88.2 ± 9.3(15)	74.2 ± 11.5(7)	85.2 ± 12.8(27)	91.3 ± 8.7(19)	95.6 ± 7.8(21)
	C10%	87.8 ± 11.1(23)	66.5 ± 14.8(6)	80.2 ± 19.4(27)	87.9 ± 14.3(20)	89.0 ± 12.8(23)
German	0%	75.9 ± 4(8)	74.5 ± 2.5(15)	74.6 ± 3.7(7)	76.0 ± 4.7(18)	76.1 ± 3.8(8)
	C5%	72.4 ± 7.2(15)	72.0 ± 6.9(11)	72.3 ± 2.9(6)	71.4 ± 8.3(12)	73.6 ± 4.9(8)
	C10%	70.5 ± 9.2(23)	68.8 ± 7.4(9)	70.1 ± 8.4(23)	70.2 ± 9.0(18)	70.7 ± 9.6(16)
Iono	0%	95.8 ± 3.6(23)	92.6 ± 3.7(7)	95.2 ± 3.8(14)	95.7 ± 3.4(16)	94.9 ± 3.9(30)
	C5%	91.3 ± 14.3(21)	89.2 ± 13.8(13)	89.7 ± 14.9(33)	92.2 ± 12.6(16)	90.0 ± 14.9(31)
	C10%	89.1 ± 17.8(23)	87.8 ± 17.5(11)	89.3 ± 15.1(28)	89.6 ± 13.6(19)	88.3 ± 16.7(33)
Leukemial	0%	97.3 ± 5.7(31)	74.5 ± 7.3(4)	97.3 ± 5.7(15)	98.6 ± 4.5(9)	100 ± 0.0(14)
	C5%	91.6 ± 18.1(25)	71.5 ± 17.5(5)	93.0 ± 18.1(28)	91.9 ± 21.1(15)	97.8 ± 7.0(14)
	C10%	90.9 ± 21.1(46)	66.4 ± 14.2(4)	91.9 ± 17.5(44)	91.6 ± 18.2(24)	94.5 ± 13.7(20)
Sick	0%	93.9 ± 0.1(1)	93.9 ± 0.1(9)	93.9 ± 0.1(1)	93.9 ± 0.1(1)	94.0 ± 0.2(16)
	C5%	93.5 ± 2.8(23)	92.8 ± 1.5(12)	93.7 ± 3.1(11)	93.8 ± 3.2(4)	93.3 ± 2.5(24)
	C10%	90.1 ± 0.5(1)	88.4 ± 6.3(12)	90.1 ± 0.3(1)	90.2 ± 0.2(7)	89.7 ± 1.7(1)
Sonar	0%	87.0 ± 6.8(49)	82.3 ± 7.0(14)	88.9 ± 5.7(39)	88.5 ± 6.1(55)	87.5 ± 6.9(31)
	C5%	84.5 ± 16.3(48)	79.9 ± 16.3(16)	83.1 ± 15.9(45)	84.5 ± 16.0(46)	85.4 ± 12.2(28)
	C10%	80.6 ± 11.9(58)	73.8 ± 17.3(13)	81.3 ± 11.9(60)	83.3 ± 11.9(60)	84.6 ± 11.8(48)
Soybean	0%	93.6 ± 3.7(28)	90.8 ± 3.6(14)	90.4 ± 4.7(34)	93.6 ± 3.7(30)	93.9 ± 4.6(11)
	C5%	90.3 ± 9.5(29)	86.1 ± 9.1(14)	88.4 ± 9.1(35)	90.2 ± 9.4(23)	90.3 ± 9.2(29)
	C10%	88.7 ± 13.9(29)	85.3 ± 14.4(15)	86.9 ± 13.3(35)	88.1 ± 13.6(28)	88.6 ± 13.5(29)
Spam	0%	92.1 ± 2.9(57)	90.0 ± 2.3(25)	92.2 ± 2.8(56)	92.1 ± 2.9(57)	92.2 ± 2.8(50)
	C5%	87.7 ± 8.7(57)	84.8 ± 8.0(26)	87.7 ± 8.7(56)	87.7 ± 8.7(57)	87.7 ± 8.7(57)
	C10%	83.7 ± 10.7(57)	80.9 ± 9.6(25)	83.7 ± 10.7(57)	83.7 ± 10.6(57)	83.7 ± 10.7(56)
SRBCT	0%	82.1 ± 26.8(51)	62.5 ± 18.7(6)	87.3 ± 16.9(40)	82.5 ± 25.0(33)	94.4 ± 8.1(22)
	C5%	79.2 ± 26.1(17)	60.8 ± 19.8(7)	82.6 ± 21.2(23)	77.4 ± 26.7(14)	88.2 ± 13.9(20)
	C10%	72.6 ± 22.8(203)	60.3 ± 19.4(9)	71.6 ± 19.9(25)	71.8 ± 27.7(31)	80.8 ± 11.9(19)
Wdbc	0%	98.1 ± 2.3(26)	96.5 ± 2.6(7)	98.1 ± 2.3(30)	98.1 ± 2.3(23)	98.1 ± 2.2(16)
	C5%	93.0 ± 15.2(24)	92.3 ± 14.9(10)	93.0 ± 15.2(27)	93.2 ± 15.2(21)	92.8 ± 15.2(26)
	C10%	88.9 ± 24.5(22)	88.6 ± 22.2(9)	88.7 ± 24.9(30)	88.7 ± 25.0(29)	89.2 ± 19.6(14)
Wine	0%	98.9 ± 2.3(12)	97.2 ± 4.0(5)	98.9 ± 2.3(6)	98.9 ± 2.3(13)	99.4 ± 1.8(9)
	C5%	94.2 ± 16.6(11)	91.0 ± 16.4(7)	94.2 ± 16.6(13)	94.2 ± 16.6(12)	94.2 ± 16.6(9)
	C10%	91.1 ± 21.2(11)	87.9 ± 22.3(7)	90.5 ± 22.9(12)	90.0 ± 24.5(13)	92.1 ± 14.5(6)
Zoo	0%	95.4 ± 8.4(12)	87.4 ± 11.5(5)	94.4 ± 8.4(15)	94.4 ± 8.4(13)	95.4 ± 8.4(6)
	C5%	92.1 ± 11.5(10)	85.0 ± 13.0(5)	92.1 ± 11.4(11)	91.1 ± 11.1(12)	92.1 ± 11.5(10)
	C10%	88.8 ± 10.7(12)	83.2 ± 11.7(7)	88.5 ± 11.9(11)	88.8 ± 10.7(12)	87.9 ± 10.0(8)
Ave.	0%	92.1(27.2)	84.7(9.4)	91.9(22.4)	91.8(27.7)	93.7(17.1)
	C5%	88.0(22.4)	81.0(10.6)	87.7(25.4)	87.8(23.3)	90.0(21.1)
	C10%	85.2(29.2)	77.6(9.5)	84.4(28.5)	84.9(29.4)	86.6(21)
	$C\triangle_1$	4.1	3.7	4.2	4.0	3.7
$\triangle$	$C\triangle_2$	6.9	7.1	7.5	6.9	7.1

better than other methods in terms of the robustness. BBL (FWL-L_2) and Logistic-LASSO ranked below MSVM-RFE. Simba performed the worst with attribute noise, but the other methods did not differ significantly. MSVM-RFE performed the best in terms of the classification accuracy and reduction ability. Clearly, BBL (FWL-L_2) had a better classification performance and reduction capacity than the other methods, except MSVM-RFE. Surprisingly, we found that Logistic-LASSO and BBL (FWL-L_2) had almost the same robustness. However, Logistic-LASSO ranked below BBL (FWL-L_2) in terms of the classification accuracy and the number of features. In the rest of methods, ReliefF and BBL (FWL-L_1) rank below Logistic-LASSO regardless of robustness and the performance of feature selection. Consistency still performed the worst in the feature selection task with attribute noise.

The time complexities of the algorithms are shown in Table 14. We assume that the dataset was S, which included M samples and N attributes, P was the class number, and t was the number of iterations. For ReliefF, k is the number of nearest neighbors selected. Table 14 shows that InfoGain had a lower time complexity than the other methods, whereas MSVM-RFE and Consistency had higher time complexities than the other methods. This was because MSVM-RFE was based on wrapper model for feature reduction, while Consistency was considered to be an enumeration method. The remaining methods differed little so we selected some representative methods and compared their runtimes to assess the validity and practicality of the algorithms.

To compare the runtime of the algorithms, we only show the runtime for seven relatively large datasets: Breast, DLBCL, German,

Table 11
SVM-RBF performance comparison of margin-based techniques with class noisy data (%).

DataSet	Noise	Logistic-LASSO	LASSO	Exponential Loss	BBL (FWL-L_2)	BBL (FWL-L_1)
Breast	0%	93.8 ± 10.6(22)	83.0 ± 10.5(46)	96.7 ± 5.5(41)	99.2 ± 2.6(31)	100 ± 0.0(22)
	C5%	91.3 ± 11.8(29)	80.5 ± 8.7(31)	95.0 ± 12.1(23)	96.8 ± 9.8(44)	94.1 ± 11.5(44)
	C10%	89.7 ± 11.3(28)	78.9 ± 10.4(38)	91.8 ± 11.2(37)	91.3 ± 11.5(40)	90.0 ± 11.8(52)
Crx	0%	85.5 ± 18.5(3)	85.2 ± 18.3(12)	85.5 ± 18.5(2)	85.5 ± 18.5(3)	85.5 ± 18.5(5)
	C5%	81.6 ± 21.3(2)	80.9 ± 20.9(12)	82.6 ± 15.1(5)	81.8 ± 21.3(3)	81.6 ± 21.4(5)
	C10%	79.7 ± 26.0(6)	78.4 ± 27.3(12)	76.1 ± 3.4(12)	80.5 ± 26.6(10)	79.1 ± 27.5(5)
DLBCL	0%	98.0 ± 4.2(23)	90.0 ± 9.4(27)	93.6 ± 5.7(21)	99.0 ± 3.2(25)	97.0 ± 4.8(31)
	C5%	93.5 ± 5.6(24)	85.6 ± 12.9(25)	88.7 ± 14.5(37)	95.2 ± 7.5(22)	92.7 ± 13.7(30)
	C10%	91.2 ± 8.7(15)	82.3 ± 13.1(26)	82.2 ± 16.1(61)	93.7 ± 10.5(24)	86.4 ± 12.1(32)
German	0%	76.4 ± 3.3(16)	74.0 ± 3.6(23)	73.4 ± 2.6(24)	77.0 ± 2.2(11)	74.8 ± 3.5(12)
	C5%	72.0 ± 7.4(17)	70.6 ± 7.5(22)	72.5 ± 6.2(9)	73.4 ± 4.7(9)	72.2 ± 3.0(14)
	C10%	70.6 ± 8.7(21)	69.8 ± 9.2(24)	71.0 ± 9.5(18)	71.7 ± 8.7(18)	69.8 ± 7.4(14)
Iono	0%	95.2 ± 4.2(25)	94.9 ± 4.2(28)	95.2 ± 3.8(15)	96.0 ± 3.6(18)	95.2 ± 3.8(28)
	C5%	90.3 ± 14.5(14)	90.3 ± 12.4(21)	89.7 ± 14.8(13)	90.8 ± 14.1(25)	90.0 ± 14.9(30)
	C10%	88.3 ± 16.7(34)	88.3 ± 16.9(33)	88.5 ± 16.9(32)	89.1 ± 15.9(26)	89.6 ± 16.1(32)
Leukemial	0%	97.3 ± 5.7(10)	94.4 ± 7.3(48)	97.5 ± 5.3(32)	100 ± 0.0(17)	97.3 ± 5.7(9)
	C5%	93.2 ± 17.6(23)	90.3 ± 17.4(41)	93.0 ± 17.7(32)	95.6 ± 14.0(32)	93.8 ± 17.5(30)
	C10%	91.6 ± 18.0(9)	87.7 ± 17.1(31)	87.9 ± 17.1(44)	93.9 ± 18.1(21)	90.1 ± 15.1(4)
Sick	0%	94.0 ± 0.2(22)	93.9 ± 0.1(1)	93.9 ± 0.2(20)	94.0 ± 0.8(14)	93.9 ± 0.2(8)
	C5%	93.4 ± 2.8(26)	93.8 ± 3.2(6)	93.4 ± 2.6(21)	93.6 ± 3.0(18)	92.8 ± 1.5(3)
	C10%	90.1 ± 0.3(1)	90.1 ± 0.3(6)	90.1 ± 0.6(1)	90.1 ± 0.3(1)	90.0 ± 0.6(3)
Sonar	0%	88.9 ± 7.2(50)	88.0 ± 7.9(50)	89.0 ± 6.4(35)	89.9 ± 4.7(40)	88.9 ± 7.2(43)
	C5%	83.1 ± 14.3(48)	84.0 ± 15.6(57)	83.6 ± 15.4(55)	87.2 ± 15.8(32)	85.4 ± 10.3(42)
	C10%	81.7 ± 10.9(59)	81.6 ± 12.1(57)	83.2 ± 11.9(60)	85.9 ± 10.6(45)	84.0 ± 15.2(49)
Soybean	0%	91.5 ± 5.9(20)	91.7 ± 4.7(25)	94.2 ± 3.7(27)	95.0 ± 3.4(24)	94.2 ± 3.8(28)
	C5%	88.5 ± 9.6(30)	88.6 ± 9.2(30)	88.5 ± 9.6(30)	90.6 ± 9.5(20)	89.6 ± 9.2(29)
	C10%	86.9 ± 13.3(35)	87.0 ± 12.9(28)	86.9 ± 13.3(35)	89.1 ± 13.9(27)	88.5 ± 13.6(25)
Spam	0%	92.2 ± 2.8(52)	92.1 ± 3.0(55)	92.2 ± 2.9(56)	92.2 ± 2.8(53)	92.1 ± 2.8(57)
	C5%	87.7 ± 8.8(54)	87.7 ± 8.8(50)	87.7 ± 8.8(57)	87.7 ± 8.7(57)	87.7 ± 8.6(56)
	C10%	83.7 ± 10.7(57)	83.7 ± 10.7(57)	83.7 ± 10.6(55)	83.8 ± 10.6(45)	83.6 ± 10.8(55)
SRBCT	0%	83.3 ± 22.4(14)	85.4 ± 19.2(36)	77.7 ± 24.3(23)	88.0 ± 9.5(29)	81.0 ± 23.5(6)
	C5%	81.8 ± 18.9(15)	81.2 ± 16.4(18)	74.3 ± 20.6(26)	87.4 ± 12.3(17)	75.8 ± 9.9(7)
	C10%	71.2 ± 22.1(15)	77.3 ± 14.5(37)	70.6 ± 23.1(100)	81.7 ± 19.7(36)	65.7 ± 20.9(4)
Wdbc	0%	98.1 ± 2.2(17)	98.1 ± 2.2(30)	98.1 ± 2.3(21)	98.1 ± 2.2(27)	98.1 ± 2.3(25)
	C5%	93.3 ± 15.3(19)	92.8 ± 15.2(27)	93.0 ± 15.2(25)	93.2 ± 15.2(20)	93.0 ± 15.2(27)
	C10%	89.1 ± 22.4(14)	88.9 ± 25.0(29)	88.7 ± 24.9(30)	89.1 ± 21.2(9)	91.6 ± 17.6(30)
Wine	0%	98.9 ± 2.3(13)	98.9 ± 2.3(13)	99.4 ± 1.8(7)	99.4 ± 2.3(6)	98.9 ± 2.3(6)
	C5%	94.2 ± 16.6(10)	94.2 ± 16.6(13)	94.2 ± 16.6(13)	94.2 ± 16.6(13)	94.2 ± 16.6(13)
	C10%	90.0 ± 24.5(13)	90.0 ± 24.6(13)	90.5 ± 22.9(12)	92.1 ± 18.0(9)	93.1 ± 14.9(8)
Zoo	0%	94.4 ± 8.4(13)	94.4 ± 8.4(12)	94.4 ± 8.4(10)	95.4 ± 8.4(7)	95.4 ± 8.4(8)
	C5%	90.0 ± 11.9(13)	91.1 ± 11.1(12)	91.1 ± 11.1(11)	92.1 ± 11.5(12)	92.1 ± 11.5(16)
	C10%	86.8 ± 13.1(13)	86.8 ± 11.9(14)	88.5 ± 11.9(10)	88.8 ± 10.7(12)	87.9 ± 10.0(16)
Ave.	0%	92.0(21)	90.3(29)	91.5(23.9)	93.5(21.8)	92.3(20.6)
	C5%	88.1(23)	86.5(26)	87.7(25.5)	90.0(23.1)	88.2(24.7)
	C10%	85.0(23)	83.6(29)	84.3(36.2)	87.2(23.0)	85.0(23.5)
	C$\triangle_1$	3.8	3.8	3.8	3.5	4.1
$\triangle$	C$\triangle_2$	6.9	6.7	7.2	6.3	7.3

Table 12
SVM-RBF performance comparison of classical methods with attribute noisy data (%).

DataSet	Noise	InfoGain	Consistency	Simba	ReliefF	MSVM-RFE
Breast	0%	96.3 ± 6.1(54)	77.9 ± 9.2(4)	96.7 ± 5.5(33)	91.7 ± 8.4(77)	100 ± 0.0(9)
	F5%	87.5 ± 5.9(23)	76.7 ± 16.5(6)	93.3 ± 9.2(29)	86.6 ± 9.8(42)	100 ± 0.0(10)
	F10%	78.8 ± 8.8(26)	72.5 ± 18.5(8)	66.8 ± 10.5(11)	81.8 ± 17.1(33)	96.3 ± 6.0(28)
Crx	0%	85.5 ± 18.5(1)	84.1 ± 17.5(12)	85.5 ± 18.5(2)	85.5 ± 18.4(1)	85.6 ± 18.5(5)
	F5%	85.5 ± 18.5(1)	83.5 ± 18.5(1)	85.5 ± 18.5(2)	85.5 ± 18.5(1)	85.5 ± 18.5(1)
	F10%	82.0 ± 16.4(7)	81.7 ± 16.5(1)	82.6 ± 15.5(7)	82.8 ± 15.3(11)	82.3 ± 15.3(10)
DLBCL	0%	98.0 ± 4.2(28)	81.0 ± 13.2(4)	92.6 ± 7.1(21)	95.0 ± 3.3(42)	100 ± 0.0(13)
	F5%	98.0 ± 4.2(25)	77.3 ± 13.4(5)	90.9 ± 7.1(25)	94.0 ± 7.0(30)	100 ± 0.0(16)
	F10%	89.8 ± 8.2(30)	65.2 ± 11.3(10)	70.4 ± 14.0(15)	85.3 ± 10.3(27)	96.0 ± 8.4(16)
German	0%	75.9 ± 4(8)	74.5 ± 2.5(15)	74.6 ± 3.7(7)	76.0 ± 4.7(18)	76.1 ± 3.8(8)
	F5%	75.1 ± 4.2(23)	74.2 ± 3.3(5)	75.1 ± 3.7(14)	75.7 ± 4.8(19)	76.1 ± 4.0(9)

(*continued on next page*)

Table 12 (*continued*)

DataSet	Noise	InfoGain	Consistency	Simba	ReliefF	MSVM-RFE
	F10%	72.9 ± 2.6(24)	70.8 ± 1.6(6)	73.1 ± 2.3(19)	72.9 ± 2.6(24)	73.6 ± 2.6(18)
Iono	0%	95.8 ± 3.6(23)	92.6 ± 3.7(7)	95.2 ± 3.8(14)	95.7 ± 3.4(16)	94.9 ± 3.9(30)
	F5%	94.9 ± 4.8(34)	90.9 ± 7.4(17)	94.9 ± 4.8(34)	94.9 ± 5.2(22)	94.9 ± 4.8(33)
	F10%	87.0 ± 7.8(33)	82.4 ± 6.8(10)	87.6 ± 8.3(25)	86.1 ± 7.0(29)	87.3 ± 6.8(32)
Leukemial	0%	97.3 ± 5.7(31)	74.5 ± 7.3(4)	97.3 ± 5.7(15)	98.6 ± 4.5(9)	100 ± 0.0(14)
	F5%	97.3 ± 5.3(51)	73.2 ± 8.5(4)	97.3 ± 5.7(13)	98.6 ± 4.5(16)	100 ± 0.0(11)
	F10%	93.6 ± 9.0(32)	68.4 ± 12.9(6)	85.9 ± 12.0(21)	87.8 ± 10.2(9)	100 ± 0.0(18)
Sick	0%	93.9 ± 0.1(1)	93.9 ± 0.1(9)	93.9 ± 0.1(1)	93.9 ± 0.1(1)	94.0 ± 0.2(16)
	F5%	93.9 ± 0.1(1)	93.9 ± 0.1(8)	93.8 ± 0.1(1)	93.8 ± 0.1(1)	93.8 ± 0.1(1)
	F10%	93.9 ± 0.1(1)	93.9 ± 0.1(8)	93.8 ± 0.1(1)	93.8 ± 0.1(1)	93.7 ± 0.1(1)
Sonar	0%	87.0 ± 6.8(49)	82.3 ± 7.0(14)	88.9 ± 5.7(39)	88.5 ± 6.1(55)	87.5 ± 6.9(31)
	F5%	84.6 ± 6.0(35)	78.4 ± 6.0(10)	84.2 ± 7.1(33)	84.1 ± 7.9(53)	86.1 ± 8.1(38)
	F10%	76.4 ± 7.2(36)	67.8 ± 10.7(6)	77.9 ± 8.5(41)	78.3 ± 9.5(27)	77.4 ± 9.8(47)
Soybean	0%	93.6 ± 3.7(28)	90.8 ± 3.6(14)	90.4 ± 4.7(34)	93.6 ± 3.7(30)	93.9 ± 4.6(11)
	F5%	93.3 ± 4.6(31)	87.1 ± 5.0(14)	88.1 ± 10.2(27)	93.2 ± 3.5(32)	90.1 ± 7.3(21)
	F10%	83.1 ± 5.6(34)	82.0 ± 6.5(25)	83.2 ± 5.0(26)	83.3 ± 5.6(34)	83.1 ± 5.2(29)
Spam	0%	92.1 ± 2.9(57)	90.0 ± 2.3(25)	92.2 ± 2.8(56)	92.1 ± 2.9(57)	92.2 ± 2.8(50)
	F5%	76.9 ± 5.3(38)	76.5 ± 5.4(34)	76.8 ± 5.6(52)	76.7 ± 5.4(51)	76.7 ± 5.3(51)
	F10%	63.4 ± 2.5(33)	62.8 ± 1.9(57)	62.9 ± 2.1(44)	63.1 ± 17.9(47)	63.8 ± 21.9(24)
SRBCT	0%	82.1 ± 26.8(51)	62.5 ± 18.7(6)	87.3 ± 16.9(40)	82.5 ± 25.0(33)	94.4 ± 8.1(22)
	F5%	82.1 ± 25.4(12)	62.3 ± 24.2(8)	78.1 ± 31.8(31)	82.4 ± 20.9(18)	94.4 ± 9.6(26)
	F10%	65.5 ± 22.4(11)	56.6 ± 26.7(13)	66.6 ± 19.6(22)	79.7 ± 19.6(15)	93.3 ± 12.1(22)
Wdbc	0%	98.1 ± 2.3(26)	96.5 ± 2.6(7)	98.1 ± 2.3(30)	98.1 ± 2.3(23)	98.1 ± 2.2(16)
	F5%	95.3 ± 3.0(24)	93.8 ± 3.8(9)	95.6 ± 2.9(23)	95.4 ± 2.5(14)	95.9 ± 2.6(10)
	F10%	87.7 ± 3.3(30)	85.6 ± 4.8(13)	87.7 ± 4.2(28)	88.1 ± 4.1(26)	87.9 ± 3.5(21)
Wine	0%	98.9 ± 2.3(12)	97.2 ± 4.0(5)	98.9 ± 2.3(6)	98.9 ± 2.3(13)	99.4 ± 1.8(9)
	F5%	93.1 ± 5.6(10)	89.3 ± 7.2(7)	93.7 ± 6.0(12)	93.1 ± 5.6(11)	93.7 ± 6.0(12)
	F10%	76.9 ± 7.6(13)	73.6 ± 9.0(10)	76.9 ± 7.6(12)	77.4 ± 6.8(8)	76.9 ± 7.6(13)
Zoo	0%	95.4 ± 8.4(12)	87.4 ± 11.5(5)	94.4 ± 8.4(15)	94.4 ± 8.4(13)	95.4 ± 8.4(6)
	F5%	94.3 ± 8.5(12)	85.4 ± 8.3(6)	94.3 ± 8.4(9)	93.8 ± 10.1(8)	94.4 ± 8.5(8)
	F10%	92.3 ± 8.0(11)	83.7 ± 10.2(7)	92.3 ± 8.0(12)	92.3 ± 8.0(12)	92.3 ± 8.0(11)
Ave.	0%	92.1(27.2)	84.7(9.4)	91.9(22.4)	91.8(27.7)	93.7(17.1)
	F5%	89.4(22.9)	81.6(9.6)	88.7(21.8)	89.1(22.7)	91.5(17.6)
	F10%	81.7(22.9)	74.8(12.9)	79.1(20.3)	82.3(21.6)	86.0(20.7)
	$F\triangle_1$	2.7	3.1	3.2	2.6	2.1
$\triangle$	$F\triangle_2$	10.5	9.9	12.7	9.4	7.7

Table 13
SVM-RBF performance comparison of margin-based techniques with attribute noisy data (%).

DataSet	Noise	Logistic-LASSO	LASSO	Exponential Loss	BBL (FWL-L_2)	BBL (FWL-L_1)
Breast	0%	93.8 ± 10.6(22)	83.0 ± 10.5(46)	96.7 ± 5.5(41)	99.2 ± 2.6(31)	100 ± 0.0(22)
	F5%	93.8 ± 6.6(25)	82.5 ± 10.5(33)	91.3 ± 11.9(38)	95.4 ± 6.0(21)	92.1 ± 9.1(30)
	F10%	86.7 ± 7.3(24)	76.3 ± 10.3(28)	84.6 ± 11.7(29)	94.2 ± 8.8(14)	88.7 ± 8.8(18)
Crx	0%	85.5 ± 18.5(3)	85.2 ± 18.3(12)	85.5 ± 18.5(2)	85.5 ± 18.5(3)	85.5 ± 18.5(5)
	F5%	85.4 ± 18.3(1)	85.2 ± 18.4(12)	81.6 ± 21.3(2)	85.5 ± 18.3(3)	85.5 ± 18.4(4)
	F10%	82.0 ± 16.6(2)	82.5 ± 15.0(14)	79.1 ± 27.4(9)	83.2 ± 15.9(7)	82.6 ± 16.6(8)
DLBCL	0%	98.0 ± 4.2(23)	90.0 ± 9.4(27)	93.6 ± 5.7(21)	99.0 ± 3.2(25)	97.0 ± 4.8(31)
	F5%	94.6 ± 5.9(9)	87.6 ± 10.3(32)	90.1 ± 7.1(18)	97.3 ± 7.0(29)	97.0 ± 4.8(19)
	F10%	87.1 ± 7.9(24)	71.1 ± 9.2(34)	81.3 ± 14.1(18)	92.6 ± 12.2(21)	86.7 ± 9.8(19)
German	0%	76.4 ± 3.3(16)	74.0 ± 3.6(23)	73.4 ± 2.6(24)	77.0 ± 2.2(11)	74.8 ± 3.5(12)
	F5%	75.0 ± 4.8(20)	73.1 ± 4.2(23)	73.3 ± 3.6(13)	76.5 ± 4.2(17)	74.8 ± 3.5(24)
	F10%	70.6 ± 8.7(21)	69.8 ± 9.2(24)	71.0 ± 9.5(18)	71.7 ± 8.7(18)	69.8 ± 7.4(14)
Iono	0%	95.2 ± 4.2(25)	94.9 ± 4.2(28)	95.2 ± 3.8(15)	96.0 ± 3.6(18)	95.2 ± 3.8(28)
	F5%	94.9 ± 4.8(34)	94.9 ± 4.8(34)	94.9 ± 4.8(34)	95.0 ± 4.7(30)	94.9 ± 4.2(31)
	F10%	86.7 ± 7.8(30)	86.4 ± 7.9(32)	87.6 ± 7.2(27)	88.1 ± 8.4(31)	86.7 ± 6.5(20)
Leukemial	0%	97.3 ± 5.7(10)	94.4 ± 7.3(48)	97.5 ± 5.3(32)	100 ± 0.0(17)	97.3 ± 5.7(9)
	F5%	97.2 ± 4.5(13)	91.7 ± 11.5(41)	95.8 ± 3.9(33)	100 ± 0.0(20)	95.0 ± 8.7(30)
	F10%	97.1 ± 6.2(17)	87.8 ± 14.5(30)	86.8 ± 9.0(16)	96.9 ± 10.5(17)	93.8 ± 10.8(11)
Sick	0%	94.0 ± 0.2(22)	93.9 ± 0.1(1)	93.9 ± 0.2(20)	94.0 ± 0.8(14)	93.9 ± 0.2(8)
	F5%	93.8 ± 0.1(1)	93.8 ± 0.1(1)	93.9 ± 0.1(1)	93.8 ± 0.1(1)	93.9 ± 0.2(1)
	F10%	93.8 ± 0.1(1)	93.8 ± 0.1(1)	93.9 ± 0.1(1)	93.8 ± 0.1(1)	93.9 ± 0.2(1)
Sonar	0%	88.9 ± 7.2(50)	88.0 ± 7.9(50)	89.0 ± 6.4(35)	89.9 ± 4.7(40)	88.9 ± 7.2(43)
	F5%	84.1 ± 8.5(50)	82.2 ± 8.3(60)	84.6 ± 6.7(34)	86.1 ± 6.9(33)	84.6 ± 8.3(38)

Table 13 (*continued*)

DataSet	Noise	Logistic-LASSO	LASSO	Exponential Loss	BBL (FWL-L_2)	BBL (FWL-L_1)
	F10%	76.4 ± 9.5(50)	73.5 ± 8.1(56)	76.0 ± 8.0(49)	79.8 ± 5.6(39)	75.0 ± 9.8(29)
Soybean	0%	91.5 ± 5.9(20)	91.7 ± 4.7(25)	94.2 ± 3.7(27)	95.0 ± 3.4(24)	94.2 ± 3.8(28)
	F5%	89.9 ± 9.9(27)	89.6 ± 10.7(35)	94.1 ± 3.6(28)	94.9 ± 3.6(26)	94.1 ± 3.5(26)
	F10%	83.4 ± 6.2(33)	82.7 ± 6.3(35)	82.7 ± 6.3(35)	83.6 ± 5.9(32)	83.6 ± 5.3(21)
Spam	0%	92.2 ± 2.8(52)	92.1 ± 3.0(55)	92.2 ± 2.9(56)	92.2 ± 2.8(53)	92.1 ± 2.8(57)
	F5%	76.6 ± 5.6(56)	76.6 ± 5.4(57)	76.6 ± 5.4(53)	76.7 ± 5.5(49)	72.2 ± 5.8(26)
	F10%	62.9 ± 21.7(26)	63.0 ± 21.8(56)	63.1 ± 21.8(55)	62.9 ± 22.1(52)	62.6 ± 17.8(32)
SRBCT	0%	83.3 ± 22.4(14)	85.4 ± 19.2(36)	77.7 ± 24.3(23)	88.0 ± 9.5(29)	81.0 ± 23.5(6)
	F5%	83.3 ± 22.4(16)	83.5 ± 22.0(44)	77.6 ± 24.7(56)	85.8 ± 18.7(36)	79.0 ± 22.1(42)
	F10%	79.9 ± 22.9(24)	75.5 ± 12.9(38)	74.7 ± 18.4(54)	79.3 ± 20.4(13)	77.4 ± 13.9(25)
Wdbc	0%	98.1 ± 2.2(17)	98.1 ± 2.2(30)	98.1 ± 2.3(21)	98.1 ± 2.2(27)	98.1 ± 2.3(25)
	F5%	95.4 ± 2.5(12)	94.7 ± 2.9(30)	95.4 ± 3.0(25)	96.3 ± 2.5(10)	95.6 ± 2.8(20)
	F10%	88.4 ± 3.3(28)	87.7 ± 3.3(30)	87.7 ± 3.3(30)	89.1 ± 3.7(26)	87.9 ± 5.4(24)
Wine	0%	98.9 ± 2.3(13)	98.9 ± 2.3(13)	99.4 ± 1.8(7)	99.4 ± 2.3(6)	98.9 ± 2.3(6)
	F5%	93.2 ± 3.7(10)	92.0 ± 5.8(13)	93.8 ± 4.3(10)	94.2 ± 6.3(11)	93.1 ± 6.2(13)
	F10%	76.9 ± 7.6(13)	76.9 ± 7.6(13)	76.9 ± 9.3(12)	79.2 ± 7.6(11)	76.3 ± 7.5(11)
Zoo	0%	94.4 ± 8.4(13)	94.4 ± 8.4(12)	94.4 ± 8.4(10)	95.4 ± 8.4(7)	95.4 ± 8.4(8)
	F5%	93.4 ± 8.2(10)	92.8 ± 9.9(14)	93.8 ± 10.1(10)	95.4 ± 8.5(8)	93.8 ± 8.3(16)
	F10%	92.3 ± 8.1(11)	91.4 ± 7.5(14)	92.3 ± 8.0(12)	92.4 ± 7.9(9)	92.3 ± 8.1(16)
Ave.	0%	92.0(21)	90.3(29)	91.5(23.9)	93.5(21.8)	92.3(20.6)
	F5%	89.3(20)	87.2(31)	88.3(25.4)	90.9(21.2)	88.9(22.9)
	F10%	83.3(22)	80.1(29)	81.4(26.3)	84.9(20.9)	82.8(18.5)
	F$\triangle_1$	2.7	3.1	3.2	2.6	3.4
$\triangle$	F$\triangle_2$	8.7	10.2	10.1	8.6	9.5

Table 14
Comparison of the time complexity.

Name	Time complexity	Name	Time complexity
ReliefF	$\Theta(tMN\log k)$	Consistency	$\Theta(77N^5)$
Simba	$\Theta(tMN)$	InfoGain	$\Theta(MN)$
LASSO	$\Theta(MN\min\{M,N\})$	Exponential Loss	$\Theta(tMN)$
Logistic-LASSO	$\Theta(MN\min\{M,N\})$	BBL (FWL-L_1)	$\Theta(tMN)$
MSVM-RFE	$\Theta(PMN^3)$	BBL (FWL-L_2)	$\Theta(tMN)$

Table 15
Comparison of the runtime (seconds).

DataSet	Simba	ReliefF	LASSO	BBL (FWL-L_1)	BBL (FWL-L_2)
Breast	53.42	81.81	35.82	943.35	44.78
DLBCL	28.26	36.13	47.23	475.13	27.48
German	78.13	178.02	53.55	1.42×10^3	90.21
Leukemial	37.41	49.49	26.20	645.45	38.34
Sick	246.36	482.25	355.23	1.01×10^4	258.17
Spam	2.92×10^3	8.45×10^3	5.38×10^3	2.78×10^4	3.05×10^3
SRBCT	16.51	22.41	6.83	246.53	17.06

Leukemial, Sick, Spam, and SRBCT, where Spam had a relatively high number of samples, while Breast, DLBCL, Leukemial, and SRBCT had a relatively high numbers of features. For the ReliefF algorithm, we set k as 10 and calculated the average margin. For BBL (FWL-L_1) and BBL (FWL-L_2), we assumed that the parameter p was set to the optimal value. Table 15 shows the runtime for feature selection using the five methods. BBL (FWL-L_1) was much slower than the other methods because the sparse parameter υ was obtained based on the cross-validation of the classification accuracy. The other methods had similar runtime.

5. Conclusions

Noise is widespread in real-world data, so robust feature selection algorithms are highly desirable. Brownboost loss is a margin-induced evaluation function, which is used for classification learning that is considered more robust than other loss functions. In this study, we developed a algorithm for feature selection based on L_2-norm regularized Brownboost loss with gradient descent techniques. We compared our algorithm with some representative feature selection methods in terms of the classification performance, dimensionality reduction capacity, and robustness.

After extensive experimental analyses, we reached the following conclusions. With 1NN and SVM-RBF, BBL (FWL-L_2) performed better than the other methods using the raw datasets, with the exception of MSVM-RFE. In a noisy environment, BBL (FWL-L_2) was more robust than the other methods to class noise and attribute noise, although it was slightly weaker than MSVM-RFE when only attribute noise was considered. However, MSVM-RFE is a wrapper technique so it requires a much longer runtime. Thus, BBL (FWL-L_2) was still the most suitable for real-world applications. BBL (FWL-L_2) performed the best of the nine filter techniques. We also found that BBL

(FWL-L_2) performed much better than BBL (FWL-L_1) in terms of its robustness and classification performance.

Acknowledgments

This work was supported by the National Key Basic Research Program under Grant 2013CB329304 and the National Natural Science Foundation of China through Grant Nos. 61222210 and 61175027 and New Century Excellent Talents in University under Grant NCET-12-0399.

References

[1] M. Dash, H. Liu, Feature selection for classification, Intelligent Data Analysis 1 (1997) 131–156.
[2] M. Dash, K. Choi, P. Scheuermann, H. Liu, Feature selection for clustering a filter solution, in: Proceeding of Second International Conference Data Mining, 2002, pp. 115–122.
[3] I. Guyaon, J. Weston, S. Barnhill, Gene selection for cancer classification using support vector machines, Machine Learning 46 (2002) 389–422.
[4] X. Zhou, P. David, MSVM-RFE: extensions of SVM-RFE for multiclass gene selection on DNA microarray data, Bioinformatics 23 (2007) 1106–1114.
[5] S. Francisco, Improving the ranking quality of medical image retrieval using a genetic feature selection method, Decision Support Systems 51 (4) (2011) 810–820.
[6] C.F. Tsai, Feature selection in bankruptcy prediction, Knowledge-Based Systems 22 (2) (2009) 120–127.
[7] R. Gilad-Bachrach, A. Navot, N. Tishby, Margin based feature selection–theory and algorithms, in: proceedings of the 21st International Conference on Machine Learning, 2004, pp. 40–48.
[8] Q.H. Hu, W.W. Pan, Y.P. Song, D. Yu, Large-margin feature selection for monotonic classification, Knowledge-Based Systems 31 (2012) 8–18.
[9] H. Liu, R. Setiono, A probabilistic approach to feature selection – A filter solution, in: The 13th International Conference on Machine Learning 1996, pp. 319–327.
[10] M. Dash, H. Liu, Consistency-based search in feature selection, Artifical Intelligence 151 (2003) 155–176.
[11] D. Huang, T.W. Chow, Effective feature selection scheme using mutual information, Neurocomputing 63 (2005) 325–343.
[12] H.W. Liu, J.G. Sun, L. Liu, Feature selection with dynamic mutual information, Pattern Recognition 42 (2009) 1330–1339.
[13] H.C. Peng, F.H. Long, C. Ding, Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy, IEEE Transactions on Pattern Analysis and Machine Intelligence 8 (2005) 1226–1236.
[14] L. Yu, H. Liu, Feature selection for high-dimensional data: a fast correlation-based filter solution, in: Proceedings of the Twentieth International Conference on Machine Learning, 2003, pp. 856–863.
[15] G. Forman, An extensive empirical study of feature selection metrics for text classification, Journal of Machine Learning Research 3 (2003) 1289–1306.
[16] T. Li, D. Ruan, W. Geert, J. Song, Y. Xu, A rough sets based characteristic relation approach for dynamic attribute generalization in data mining, Knowledge-Based Systems 5 (20) (2007) 485–494.
[17] Q.H. Hu, S. An, D.R. Yu, Soft fuzzy dependency for robust feature evaluation, Information Sciences (22) (2010) 4384–4400.
[18] V.N. Vapnik, Statistical Learning Theory, New York, 1998.
[19] P.L. Bartlett, J. Shawe-Taylor, Generalization performance of support vector machines and other pattern classifiers, in: Advances in Kernel Methods: Support Vector Learning, MIT Press, Cambridge, MA, 1999, pp. 43–54.
[20] K. Crammer, R. Gilad-Bachrach, A. Navot, Margin analysis of the LVQ algorithm, in: Proc. 17th Conference on Neural Information Processing Systems, 2002.
[21] B. Chen, H. Liu, J. Chai, Z. Bao, Large margin feature weighting method via linear programming, IEEE Transactions on Knowledge and Data Engineering 10 (2009) 1475–1486.
[22] Q.H. Hu, P.F. Zhu, Y. Yang, D.R. Yu, Large-margin nearest neighbor classifiers via sample weight learning, Neurocomputing 74 (2011) 656–660.
[23] A. Garg, D. Roth, Margin distribution and learning algorithms, in: Proceedings of the Twentieth International Conference on Machine Learning, 2003, pp. 210–217.
[24] X. Nguyen, M.J. Wainwright, M.I. Jordan, On surrogate loss functions and f-divergences, The Annals of Statistics 2 (2009) 876–904.
[25] C. Rudin, R.E. Schapire, I. Daubechies, Analysis of boosting algorithms using the smooth margin function, The Annals of Statistics 6 (2007) 2723–2768.
[26] Y. Freund, An adaptive version of the boost by majority algorithm, International Conference on Machine Learning 43 (2001) 293–318.
[27] J.H. Friedman, T. Hastie, R. Tibshirani, Additive logistic regression: a statistical view of boosting, Annals of Statistics 28 (2000) 337–407.
[28] J.A.K. Suykens, J. Vandewalle, Least squares support vector machine classifiers, Neural Processing Letters 9 (3) (1999) 293–300.
[29] J. Henseler, C.M. Ringle, R. Sinkovics, The use of partial least squares path modeling in international marketing, Advances in International Marketing (2009) 277–319.
[30] M.H. Yang, S. Belongie, Visual tracking with online multiple instance learning, Computer Vision and Pattern Recognition (2009) 983–990.
[31] J.H. Friedman, B.E. Popescu, Predictive learning via rule ensembles, The Annals of Applied Statistics 2 (3) (2008) 916–954.
[32] S.Y. Park, Y.F. Liu, Robust penalized logistic regression with truncated loss functions, Canadian Journal of Statistics 39 (2) (2011) 300–323.
[33] S. Shalev-Shwartz, A. Tewari, Stochastic methods for L_1 regularized loss minimization, Journal of Machine Learning Research 12 (2011) 1865–1892.
[34] C. Parka, J.Y. Koob, P.T. Kimc, J.W. Leeb, Stepwise feature selection using generalized logistic loss, Computational Statistics and Data Analysis 52 (7) (2008) 3709–3718.
[35] I. Kononenko, Estimating attributes: analysis and extensions of RELIEF, in: Proceedings European Conference Machine Learning, 1994, pp. 171–182.
[36] Y. Sun, Iterative RELIEF for feature weighting: algorithms, theories, and applications, IEEE Transations on Pattern Analysis and Machine Intelligence 6 (2007) 1–17.
[37] R. Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society. Series B 58 (1996) 267–288.
[38] P. Zhao, B. Yu, On model selection consistency of lasso, Journal of Machine Learning Research 7 (2006) 2541–2563.
[39] S.K. Shevade, S.S. Keerthi, A simple and efficient algorithm for gene selection using sparse logistic regression, Bioinformatics 17 (19) (2003) 2246–2253.
[40] J. Liu, J. Chen, J. Ye, Large-scale sparse logistic regression, KDD09, 2009.
[41] P. Wei, P.J. Ma, X.H. Su, Large margin feature selection for support vector machine, Applied Mechanics and Materials 274 (2013) 161–164.
[42] I. Guyon, Gene selection for cancer classification using support vector machines, Machine Learning 46 (2002) 389–422.
[43] Y. Ding, D. Wilkins, Improving the performance of SVM-RFE to select genes in microarray data, BMC Bioinformatics 7 (2006) 12–20.
[44] Y. Tang, Y.Q. Zhang, Z. Huang, Development of two-stage SVM-RFE gene selection stategy for microarray expression data analysis, IEEE Transaction on Computational Biology and Bioinformatics 3 (2007) 1545–1580.
[45] Y.L. Cun, Optimal brain damage, in: Advances in Neural Information Processing Systems II, Morgan Kaufman Publishers, 1990.
[46] M.R. Osborne, B. Presnell, B.A. Turlach, A new approach to variable selection in least squares problems, IMA Journal of Numerical Analysis 20 (2000) 389–403.
[47] Y. Kim, J. Kim, Gradient LASSO for feature selection, in: Proceedings of the 21 st International Conference on Machine Learning, 2004.
[48] F.P. Nie, H. Huang, X. Cai, C. Ding, Efficient and Robust Feature Selection via Joint L2, 1-Norms Minimization, NIPS2010.
[49] S. Perkins, K. Lacker, J. Theiler, Grafting: Fast incremental feature selection by gradient descent in function space, Journal of Machine Learning Research 3 (2003) 1333–1356.
[50] S.I. Lee, H. Lee, P. Abbeel, A.Y. Ng, Effcient L_1 Regularized Logistic Regression, AAAI, 2006.
[51] Y. Tsuruoka, J. Tsujii, S. Ananiadou, Stochastic Gradient Descent Training for L1-regularized Log-linear Models with Cumulative Penalty, in: Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP, 2009, pp. 477–485.
[52] A. Beck, M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, SIAM Journal of Imaging Sciences 1 (2) (2009) 183–202.
[53] C.J. Merz, P. Merphy, UCI repository of machine learning databases [OB/OL]. <http://www.ics.uci.edu/mlearn/>.
[54] P. CM, T. Solie, E. MB, Molecular portraits of human breast tumours, MLRRepository.html, Nature 48 (2000) 747–752.
[55] A. Alizadeh et al., Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling, Nature (2000) 503–511.
[56] T. Golub, Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, Science (2000) 531–537.
[57] J. Khan, J.S. Weil, M. Ringne, L.H. Saall, M. Ladanyi, Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks, Nature Medicine 7 (2001) 673–679.
[58] J. Liu, S.W. Ji, J.P. Ye, SLEP: Sparse Learning with Efficient Projections, Arizona State University, 2009.