中央研究院
資訊科學研究所
Institute of Information Science, Academia Sinica • Taipei, Taiwan, ROC

TR-IIS-09-010

An Adaptive Multiple Feature Subset Method for Feature Ranking and Feature Selection

Fu Chang and Jen-Cheng Chen

December 22 2009 || Technical Report No. TR-IIS-09-010
http://www.iis.sinica.edu.tw/page/library/LIB/TechReport/tr2009/tr09.html

An Adaptive Multiple Feature Subset Method for Feature Ranking and Feature Selection

Fu Chang and Jen-Cheng Chen

Institute of Information Science, Academia Sinica
128 Academia Road, Taipei, 115, Taiwan
{fchang, clement}@iis.sinica.edu.tw

Abstract

In this paper, we propose a new feature evaluation method that forms the basis for feature ranking and feature selection. The method starts by generating a number of feature subsets in a random fashion and evaluates features based on the derived subsets. It then proceeds in a number of stages. In each stage, it inputs the features whose ranks in the previous stage were above the median rank and re-evaluates those features in the same fashion as it did in the first stage. When the number of features is high, the method has a computational advantage over recursive feature elimination (RFE), a state-of-art method that ranks features by identifying the least valuable feature in each stage. It also achieves better results than RFE in terms of classification accuracy and some other measures introduced in this paper, especially when the size of the training data is small or the number of irrelevant features is large.

Keywords: adaptive multiple feature subset method, correlation method**,** curse of dimensionality, essential feature, feature ranking, feature selection, random subset method, recursive feature elimination

1. Introduction

The performance of learning machines depends to a large extent on the quality of the training data. The presence of irrelevant features is a factor that can affect the test accuracy of trained classifiers significantly. Although some effective learning methods, such as support vector machines (SVMs), can tolerate a few redundant features, their generalization power can be compromised by a large number of such features. This so-called "curse of dimensionality" (COD) occurs because one needs to sample a lot more data points to gain insight into a high-dimensional space compared to a low-dimensional one. In the case of SVM learning, the insufficiency of training samples may trick the learning process into believing that a larger margin exists in an incorrect high-dimensional space than in the correct low-dimensional one.

The selection of features is often based on a process that evaluates the usefulness of features. In some methods, the evaluation process determines the features that are selected. In other methods, features are ranked before they are selected. We discuss the two selection methods in more detail at the end of this section.

In this paper, feature selection is considered as a means of extracting a subset of features from the set of full features. We do not address those methods that transform features (as linear combinations or clusters of original features, for example) before reducing the number of them. Under this restriction, feature selection methods can be generally categorized into three types: filters, wrappers, and embedded methods (Blum and Langley, 1997; Guyon and Elisseeff, 2003).

Filters evaluate features individually according to some statistical criteria (Golub et al., 1999; Hall, 2000) or information-theoretic criteria (Lewis, 1992; Koller and Sahami, 1996; Singh and Provan, 1996; Peng et al., 2005). Forman (2003) reviews and compares many such criteria for binary variables in text categorization applications. After evaluating the features, one can use some statistical methods to eliminate those that are irrelevant (Almuallim and Dietterich, 1991; Kira and Rendell, 1992; Konenko, 1994). An alternative procedure is to use some classification methods to select a subset comprised of a number of top-ranked features (see, for example, Lewis, 1992; Forman, 2003).

Wrappers (Kohavi and John, 1997) evaluate feature subsets using certain search strategies to find the locally best subset. They proceed in this manner until no better feature subset can be found. Various strategies can be used for the local search, e.g., sequential backward selection (Marill and Green 1963), branch-and-bound (Narendra and Fukunaga, 1977; Yu and Yuan, 1993), beam search and bidirectional search (Siedlecki and Sklansky, 1988), best-first (Xu et al., 1989), genetic algorithms (Vafaie and De Jong, 1992; Vafaie and De Jong, 1993), sequential floating search (Pudil et al., 1994), and simulated annealing (Meiri and Zahavi, 2006). The evaluation is often performed with the help of a certain learning machine and some validation data sets.

Embedded methods (Lal et al., 2006) rely on learning machines to evaluate the usefulness of features. Recursive feature elimination (RFE) is a well-known embedded method that uses either linear SVMs (Guyon et al., 2002), non-linear (i.e., kernel-based) SVMs (Rakotomamonjy, 2003), or penalized logistic regression (Zhu and Hastie, 2004) for learning. RFE determines the usefulness of a feature by estimating the change in the objective function. There are various ways to estimate such changes, e.g., finite difference calculation, quadratic approximation of the cost function, sensitivity of the objective-function calculation, and the generalization error bound (LeCun et al., 1990; Guyon and Elisseeff, 2003; Rakotomamonjy, 2003; Weston et al., 2003). In the original version of RFE, features are eliminated one at a time; they can also be eliminated one group at a time, as required by some applications (Lal et al., 2004).

In addition to RFE, Bi et al. (2003), Perkins et al. (2003), and Weston et al. (2003) formulate feature selection as an optimization problem. This approach adds certain regularization terms to the original hard-margin or soft-margin optimization problem. However, the proposed algorithms only work for problems with linear objectives. An alternative approach designed to handle non-linear objectives transforms the feature selection problem into a feature scaling problem by assigning a weight to each feature. In the latter approach, the selected features are those that attain significantly large weights (Jebara and Jaakkola, 2000; Weston et al., 2000; Chapelle et al., 2002; Grandvalet and Canu, 2002). Note that, although feature selection can be formulated as an optimization problem, the problem is NP-hard (Amaldi and Kann, 1998) under such a formulation. Thus, in practice, some approximations or greedy algorithms have to be adopted to solve the problem.

Rather than use SVMs as learning machines, some embedded methods use decision trees (Breiman et al. 1984; Quinlan, 1986; Cardie, 1993; Schlimmer, 1993), linear least-square predictors (Chen et al., 1989), polynomial classifiers (Rivals and Personnaz, 2003; Stoppiglia, 2003), or perceptrons (Gentile, 2004) to evaluate the usefulness of features.

Feature selection is a highly developed area. The above review only highlights some well-known methods. For further information, readers may consult Blum and Langley

(1997), Kohavi and John (1997), Guyon and Elisseeff (2003), Lal et al. (2006), and a special issue of the *Journal of Machine Learning Research* (Guyon and Elisseeff editors, 2003).

In this paper, we propose a method that evaluates features based on a number of feature subsets that are generated in an adaptive fashion. For this reason, we call our method the *adaptive multiple feature subset* (AMFES) method. AMFES adopts a very useful technique from the RFE method. RFE starts with the set of all features and eliminates the feature with the lowest score. A feature's score is defined as the difference in the SVM's objective function with and without the feature. RFE performs the same operation on the set of remaining features until it is left with only one feature. By so doing, it ranks all the features in the order they are eliminated during the procedure.

AMFES performs feature ranking in a number of stages. In the initial stage, it generates several feature subsets. When a feature subset is given, AMFES assigns the same scores to the features as those allotted by RFE. It then ranks features according to their strengths, where the strength of a feature *f* is the sum of the scores assigned to *f*, re-scaled by the number of subsets that contain *f*. In each subsequent stage, AMFES inputs the features whose ranks in the previous stage were above the medium rank. It then re-computes the ranks of those features in the same way as it does in the first stage.

The random subset method (RSM), proposed by Lai et al. (2006), is a precursor to AMFES that ranks features in a non-adaptive fashion. Motivated by the methodology of random decision forests or random forests (Ho, 1995, 1998; Breiman, 2001), RSM generates feature subsets all at once and ranks features based on the scores computed on the subsets. Since RSM does not re-compute feature ranks, it behaves like AMFES in the initial stage. Our experiments demonstrate that ranking features in an adaptive fashion is more effective than a non-adaptive approach.

AMFES performs well for the following reason. Initially, AMFES may encounter a huge number of redundant features. Because of the COD effect, it may require a huge number of training samples to obtain a good feature ranking. Despite the shortage of training data, AMFES can move most, if not all, essential features to the top ranks, thereby reducing the number of irrelevant features in those ranks. In the next stage, AMFES deals with the features that were top-ranked in the previous stage, so there are fewer irrelevant features. Thus, it can improve the feature ranking by moving essential features closer to the top of the ranked list and pushing irrelevant features closer to the bottom. In subsequent stages, AMFES continues to enhance the feature ranking until the end of the procedure.

Let d be the total number of features. In each stage of the feature ranking procedure, AMFES adjusts half of the features inherited from the previous stage and generates m feature subsets for the adjustment; hence, it generates $m\log_2 d$ feature subsets and trains the same number of SVMs during the whole procedure. RFE, on the other hand, eliminates one feature at a time and needs to train d SVMs during the whole procedure. The computational efficiency of AMFES over RFE for large values of d is thus very clear.

In addition to the feature ranking procedure, we propose a feature selection procedure that divides a data set into various pairs of training and validation components (called training-validation pairs hereafter). From these pairs, we derive a set of selected features that performs better than a set derived from only one training-validation pair.

To evaluate the performance of AMFES and other methods, we adopt two approaches. The first examines the effect of each method's feature ranking procedure. Specifically, we compute the accuracy rates of the classifiers that are built on k top-ranked features, where k runs from 1 to d. The second approach evaluates each method's feature selection procedure. For this purpose, we compute the accuracy rate of the classifier that is built on a set of selected features. In both approaches, AMFES outperforms the compared methods.

To evaluate a feature ranking procedure, we use *benchmark* data sets; and to assess a feature selection procedure, we use *synthetic* data sets. Benchmark data sets are useful for evaluation purposes, but they have the following limitations. First, we do not know which features in the data sets are essential. Second, we cannot determine how the compared methods cope with the COD effect. To compensate for these deficiencies, we design a few synthetic data sets in which the essential features are generated according to a multi-normal distribution. We then add some irrelevant features to the essential ones to complete the formation of our data points. The advantage of generating synthetic data is that we can control the number of samples and the number of features at our disposal.

On synthetic data sets, we can evaluate a method in terms of several measures, such as the test accuracy, the number of selected features, and the number of essential features captured by the selection procedure. We can also observe how these measures vary with the number of features and the number of training samples.

In addition to AMFES and RFE, we also compare the performance of the correlation (CORR) method, which evaluates individual features in terms of the correlation between class labels (labels, for short) and feature values. Although CORR computes feature ranks rapidly, it is not as efficient as AMFES or RFE. Our experiments on synthetic data help explain why this is so. CORR can easily pick essential features that manifest a high correlation with labels; however, it is not efficient in identifying features that correlate weakly with labels but strongly with other essential features.

Although the compared methods can utilize both linear SVM (LSVM) and non-linear SVM (NSVM) as learning machines, we only employ LSVM in our experiments because it is a lot more efficient than NSVM. Moreover, as Guyon et al. (2002) observed, the soft-margin version of SVM can find an adequate solution for LSVM. The soft-margin assumption requires that the cost factor C has a low value. For this reason, we set the value of C at 1.

At this point, we would like to remark on the two paths that lead to a set of selected features. On the first path, the search strategy starts from a feature subset and tries to find a more promising subset to rank above it. By contrast, on the second path, features are ranked first; then, a subset comprising the top-k ranked features is selected. The first path has 2^d feature subsets to select from; thus, overfitting may be an issue (Ng, 1998). It has been shown, for example, that a more complicated search strategy may perform weakly compared to a simpler one, although the former yields a seemingly better feature subset (Reunanen, 2003). The second path has d nested subsets of ranked features to choose from; hence, it is more robust against overfitting (Guyon and Elisseeff, 2003).

AMFES proceeds via the second path. From our experiment results on synthetic data, we observe that AMFES may fail to select some essential features when a training data set is small. This is an inevitable consequence of using a small amount of training data. However, although AMFES does not select some features, it places them before all irrelevant features on the ranked list. Moreover, it places more essential features towards the top of the ranked list than RFE and CORR. This is an encouraging result because, in many applications, users rely on machine learning to suggest potentially useful factors (features), rather than specify exactly what they are. Therefore, it is important that those useful factors are ranked higher than irrelevant ones. AMFES satisfies this requirement rather well, and is more effective than the compared methods in this respect.

The remainder of the paper is organized as follows. In Section 2, we describe the basic components of AMFES, namely, the feature-strength measure, as well as the feature ranking and feature selection procedures, which are based on this measure. In Section 3, we present our experimental setting and the experiment results. There are a number of goals to accomplish in the experiments, so we divide this section into three subsections. We describe the experimental data sets in Subsection 3.1; we use the first approach to compare AMFES with the alternative methods in Subsection 3.2; and we use the second approach for comparison in Section 3.3. Section 4 contains some concluding remarks.

2. The AMFES Method

We assume that a set of training samples $\mathrm{X}_n = \{(\mathbf{x}_1, y_1), (\mathbf{x}_n, y_n)\}$ is given, where $\mathbf{x}_i \in \mathbf{R}^d$ is a data point and y_i is the label of $\mathbf{x}_i$ for $i = 1, n$. In this section, we describe our feature ranking procedure followed by our feature selection procedure. The implementations of both procedures are available at

http://ocrlnx03.iis.sinica.edu.tw/~dar/Download%20area/amfes.php3.

2.1 The Feature Ranking Procedure

The ranking procedure of AMFES proceeds in stages, as shown in Figure 1. In each stage, AMFES takes half of features from the previous stage and adjusts their ranks; the ranks of the remaining features are the same in all subsequent stages. In the initial stage, AMFES starts with the set of all d features, computes their strengths (defined below), and ranks them accordingly. At the end of this stage, AMFES outputs the features and their ranks. Let κ_{t-1} be the number of features output in stage t-1. In stage t, AMFES takes the κ_t top-ranked features as input, where $\kappa_t = \kappa_{t-1}/2$ (if κ_{t-1} is an odd integer, it is understood that κ_t will take the integer part of $\kappa_{t-1}/2$); it then computes the strengths of the κ_t features and ranks them accordingly. Finally, it outputs the κ_t features and their ranks. This procedure stops in stage T when $\kappa_T \leqq 3$.

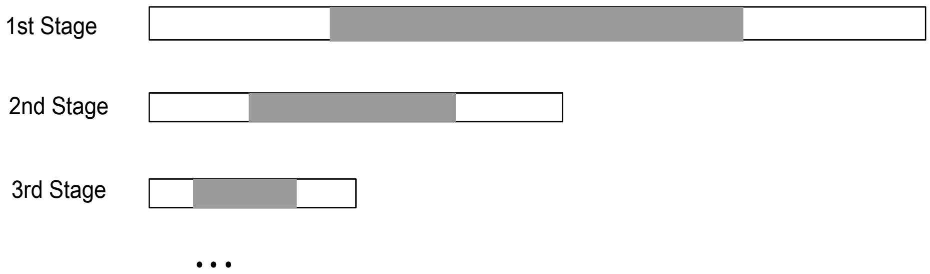

Figure 1. In each stage, half of the features, depicted by a white box, are taken as input from the previous stage. The ranks of those features are adjusted based on 100 feature subsets. Each subset comprises half of the input features, depicted by a gray box.

Next, we explain how to compute the strengths of features and how to rank the features in each stage. Let κ be the number of features taken as input in a given stage. For convenience, we assume that the features are indexed from 1 to κ. We start by generating a number of feature subsets of size λ, where $\lambda = \kappa/2$. Each feature subset S induces a transformation; for example, if S = {2, 4, 6}, the transformation induced by S converts $\mathbf{x} = (x_1, \ldots, x_\kappa)$ to $\mathbf{z} = (x_2, x_4, x_6)$. The steps of the procedure are as follows.

I. Generate m independent subsets $S_1, \ldots, S_m$, where each S_i consists of λ elements drawn randomly and independently from the set of input features. To ensure that each input feature has a good chance of being included in several feature subsets, we set $m = 100$.

II. Each S_i induces a transformation that converts $\mathbf{x}_1, \ldots, \mathbf{x}_n$ to $\mathbf{z}_{i1}, \ldots, \mathbf{z}_{in}$. We construct an SVM $\mathcal{C}_i$ based on $\mathbf{z}_{i1}, \ldots, \mathbf{z}_{in}$. Then, for each $f \in S_i$, we compute the change in the $\mathcal{C}_i$-objective function due to f (specified below), and assign that quantity, denoted as $score_i(f)$, to f.

III. We compute the *strength* of each input feature f by summing all the scores that have been assigned to f, and divide the sum by the number of times each score has been assigned to f. The result is denoted by $\theta(f)$:

$$\theta(f) = \frac{\sum_{i=1}^{m} \sum_{f \in S_i} score_i(f)}{\sum_{i=1}^{m} I_{\{f \in S_i\}}},$$

where I is an indicator function such that $I_{prop} = 1$ if *prop* is true; otherwise, $I_{prop} = 0$.

IV. Rank all the input features in descending order of $\theta(\cdot)$.

With regard to the score assigned to each feature f in S_i in Step II, a classifier $\mathcal{C}_i$ is trained on $\mathbf{z}_{i1}, \ldots, \mathbf{z}_{in}$ and a score is assigned to each feature f as the change in the $\mathcal{C}_i$-objective function due to f. $\mathcal{C}_i$ is an SVM whose objective function is

$$obj = \sum_{j,k} y_j \alpha_j y_k \alpha_k k(\mathbf{x}_j, \mathbf{x}_k),$$

where $\mathbf{x}_j$ and $\mathbf{x}_k$ are support vectors in $\mathcal{C}_i$, y_j and y_k are their respective labels, α_j and α_k are their respective weights, and $k(\mathbf{x}_j, \mathbf{x}_k)$ is a kernel function that expresses the similarity relation between $\mathbf{x}_j$ and $\mathbf{x}_k$. In the case of LSVM, the kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$ is reduced to the inner product of $\mathbf{x}_i$ and $\mathbf{x}_j$. The change in the SVM's objective function due to a feature f is

$$\left| obj - obj^{(f)} \right| = \left| \sum_{j,k} y_j \alpha_j y_k \alpha_k k(\mathbf{x}_j, \mathbf{x}_k) - \sum_{j,k} y_j \alpha_j y_k \alpha_k k(\mathbf{x}_j^{(f)}, \mathbf{x}_k^{(f)}) \right|$$

(Rakotomamonjy, 2003). Here, if $\mathbf{x}$ is a p-dimensional feature vector, then $\mathbf{x}^{(f)}$ is the $(p-1)$-dimensional vector obtained from $\mathbf{x}$ by dropping the feature f. For example, if $\mathbf{x} = (x_1, \ldots, x_p)$, then $\mathbf{x}^{(1)} = (x_2, \ldots, x_p)$. In the case of the LSVM,

$$\left| obj - obj^{(f)} \right| = w_f^2,$$

where w_f is the f^{th}-component of the vector

$$\mathbf{w} = \sum_j y_j \alpha_j \mathbf{x}_j$$

(Guyon et al., 2002).

We now summarize the parameters involved in the ranking procedure and assess their sensitivities. First, the cost factor C is set at 1, indicating that a soft-margin SVM is used as the learning machine. Next, in each stage, AMFES inputs the κ features that were top-ranked in the previous stage. Moreover, it generates *m* subsets, each comprised of μ features. In our experiments, we set κ at half the number of input features in the previous stage, *m* at 100, and μ at $\kappa/2$.

The soft-margin version of SVM plays an important role in AMFES. Our method has to generate various feature subsets in order to compute the strengths of the features, which are derived from the sizes of the margins. A soft-margin SVM makes it easy for AMFES to find a subset that produces a non-trivial soft margin. In fact, any subset comprised of some essential features has a good chance of producing such a margin. For this reason, AMFES's performance is not very sensitive toμ, the size of each feature subset, so long as μ is not close to two bad values, 1 and κ. We set μ to $\kappa/2$, because this value is farthest from these two values.

AMFES has to generate *m* feature subsets in each stage. Since the size of each subset is a function of κ, *m* does not have to be related to κ. However, *m* must be large enough to ensure that each feature is sampled a sufficient number of times. This is the reason that we set *m* to 100. The probability that a given feature will fall in a subset is 0.5; therefore, the expected number of the subsets that contain a given feature is 50.

In each stage, AMFES takes 1/2 of the features from the previous stage and re-ranks them. This is a surprisingly robust strategy. If we replace '1/2' by a more conservative '2/3' or '90%', we do not gain any significant improvement in performance; in fact, it increases the computation time. We even replaced the '1/2' strategy with a much more costly strategy that inputs a variable, rather than a constant, proportion of features in each stage, but there was still no significant improvement. On the other hand, replacing '1/2' with a more radical '1/3' of the features may cause the performance to deteriorate; hence, this strategy is not recommended.

2.2 The Feature Selection Procedure

To select features, we divide our data set into a *training* component and a *validation* component, which are then used to build SVMs and select a set of features respectively.

Feature selection is difficult because of the small number of data points compared to the number of irrelevant features. As a result, the number of data points distributed among the training and validation components is also small, so the obtained validation accuracy may *not* be stable. In other words, if we vary the members of the validation component, we may change both the number and the content of selected features drastically.

One solution to this problem is to divide the data set in more than one way. For example, if we divide the data in Π ways, we can extract Π training-validation pairs. Each pair p, $p = 1, \ldots, \Pi$, produces a feature ranking and a set $\{v_p(k): k = 1, \ldots, d\}$, where $v_p(k)$ represents the validation accuracy rates associated with the k top-ranked features. We then take the average of the accuracy rates to obtain

$$v_\Pi(k) = \sum\nolimits_{p=1}^{\Pi} v_p(k) / \Pi \,, \tag{1}$$

for $k = 1, \ldots, d$. Based on $v_\Pi(\cdot)$, we define

$$\sigma_\Pi = \arg\max_k v_\Pi(k)\,. \tag{2}$$

To obtain a set of features from the Π training-validation pairs, for $p = 1, \ldots, \Pi$, we define

$$\sigma_p = \arg\max_k v_p(k)\,,$$

$$credit_p(f) = \begin{cases} 1, \text{ if } \sigma_p \geq f\text{'s rank in the ranking list for } p, \\ 0, \text{ otherwise,} \end{cases}$$

and

$$credit_\Pi(f) = \sum\nolimits_{p=1}^{\Pi} credit_p(f)\,. \tag{3}$$

We then rank all the features in *descending* order of $credit_\Pi(\cdot)$. Finally, we take the σ_Π top-ranked features as the selected features. This procedure produces a single set, rather than Π sets, of selected features from the Π training-validation pairs. In the next section, we demonstrate the merits of the procedure via experiments.

3. Experiment Settings and Results

In our experiments, we compare the performance of AMFES with that of other methods. We also investigate the effect of COD on the compared methods and use it as a measure for comparison. Finally, we demonstrate the merits of the proposed feature selection procedure.

To compare AMFES with other methods, we adopt two approaches. The first assesses the performance of each method on the benchmark data sets. We begin by extracting Π training-validation pairs from each data set. Then, for each compared method, we compute $v_{\Pi}(k)$ as defined in (1), where k runs from 1 to d. Finally, we form the curve $\{(k, v_{\Pi}(k)): k = 1, \ldots, d\}$, called the *v-curve* hereafter.

The above approach only examines the effect of a feature ranking procedure. To evaluate a feature selection procedure, we must use another approach. In this case, we conduct our experiment on synthetic data sets, instead of the benchmark data sets. There are two reasons for this. First, evaluation based the benchmark data sets is a lot more costly. Second, the synthetic data sets allow us to gain better insights into the feature selection procedures we evaluate. We discuss these points further in Subsection 3.3. Under the second approach, we form Π training-validation pairs out of each synthetic data set and derive a set of selected features from those pairs. Then, we build a classifier on the selected features and compute its accuracy rate on an independent test data set.

Next, we describe the data sets used in our experiments.

3.1 Experimental Data Sets

We use both benchmark data sets and synthetic data sets in the experiments. Each data point in a dataset is associated with one of two labels, denoted as 1 and -1 respectively. Table 1 shows the benchmark data sets, their properties, and the sources. Note that the original “GINA”, “REGED”, “LUCAP”, and “MARTI” datasets in the source repositories contain some samples whose labels have not been disclosed to the public. Therefore, we exclude those samples from our experiments.

Table 1. The benchmark data sets used in the experiments, where N_1 = the number of features, and N_2 = the number of samples.

Dataset	N_1	N_2	Source
Colon	2,000	62	U. Alon et al., 1999; http://www.kyb.tuebingen.mpg.de/bs/people/weston/l0/
Lymphoma	4,026	96	A. A. Alizadeh et al., 2000; http://www.kyb.tuebingen.mpg.de/bs/people/weston/l0/
Leukemia	7,129	72	T. R. Golub, et al., 1999; http://www.broadinstitute.org/cgi-bin/cancer/publications/pub_paper.cgi?mode=view&paper_id=43
GINA	970	3,468	WCCI 2006, Model Selection workshop and performance prediction challenge; http://clopinet.com/isabelle/Projects/modelselect
REGED	999	500	WCCI 2008, Causation and Prediction Challenge; http://www.causality.inf.ethz.ch/challenge.php
LUCAP	143	2,000	
MARTI	1,024	500	

To construct synthetic data sets, we follow the experiment design described by Guyon (2003) closely. The synthetic data sets comprise data points $\mathbf{x} = (x_1, \ldots, x_c, x_{c+1}, \ldots, x_d)$, where $x_1, \ldots, x_c$ are essential features and $x_{c+1}, \ldots, x_d$ are irrelevant features. The latter are independent random variables that are uniformly distributed in the closed interval [-1, 1]. The essential features $x_1, \ldots, x_c$ are generated by the following linear model.

$$\mathbf{v} = \mathbf{u}\mathbf{W} + \boldsymbol{\mu} \qquad (4)$$

where $\mathbf{v} = (x_1, \ldots, x_c)$, $\mathbf{W}$ is a $c \times c$ matrix whose entry at row i and column j is w_{ij}, $\mathbf{u} = (u_1, \ldots, u_c)$, and $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_c)$. The vector $\mathbf{u}$ is composed of c independent unit normal random variables (i.e., their mean is 0 and the standard deviation is 1). The matrix $\mathbf{W}$ comprises entries w_{ij}, which are uniformly distributed in the closed interval [-1, 1]. The vector $\boldsymbol{\mu}$ is composed of c random numbers whose value is 1 or -1 with a probability of 1/2. In summary, $\mathbf{v}$ is a multi-normal random vector whose mean vector is $\boldsymbol{\mu}$ and whose covariance matrix is $\mathbf{W}^T\mathbf{W}$, where $\mathbf{W}^T$ is the transpose of $\mathbf{W}$.

To avoid the generation of outliers by (4), we adjust the value of x_i, $i = 1, \ldots, c$, by the following truncation procedure:

$$x_i \mapsto \max(\min(x_i, 3), -3)\,. \qquad (5)$$

Labeled data points are derived by generating two sets of matrices and mean vectors $\{\mathbf{W}_i, \boldsymbol{\mu}_i\}$, $i = 1, 2$; the first set is associated with label 1 and the second set with label -1.

To obtain a data point with label 1, we generate c independent unit normal random variables $u_1, \ldots, u_c$, and then generate $x_1, \ldots, x_c$ by means of (4) and (5), where $\mathbf{W} = \mathbf{W}_1$ and $\boldsymbol{\mu} = \boldsymbol{\mu}_1$. The remaining $x_{c+1}, \ldots, x_d$ are generated independently and uniformly from the closed interval [-1, 1]. A data point with label -1 is generated in a similar fashion. We also stipulate that, in each synthetic data set, there must be equal numbers of samples carrying label 1 and label -1.

In the experiments, we create two groups of synthetic data sets. The common dimensionality of Group I is 200 and that of Group II is 2,000; and the number of essential features in each group is 15. The data sets in each group are *nested* in the sense that if A_1, A_2, A_3, … are the sets in a group that are ordered according to their sizes, then $A_1 \subseteq A_2 \subseteq A_3 \subseteq \ldots$, as shown in Table 2.

Table 2. The synthetic data sets used in the experiments.

Data Set	I-1	I-2	I-3	II-1	II-2	II-3
Number of Features	200			2,000		
Number of Essential Features	15			15		
Size of Data Set	500	1,000	1,500	500	1,000	1,500

Finally, when building SVMs on a benchmark or a synthetic data set, we normalize all the feature values to a real number between 0 and 1. We do this by transforming each value v of a feature f into $(v\text{-}f_{min})/(f_{max}\text{-}f_{min})$, where f_{max} and f_{min} are the maximum and minimum values of f respectively.

3.2 Comparison of Methods: The First Approach

In this sub-section, we compare the performance of a number of methods, using the v-curve as the means of comparison. In this experiment, we only use the benchmark data sets listed in Table 1. We begin by extracting Π training-validation pairs from each data set. All pairs are randomly and independently generated. In each pair, the ratio of the training component to the validation component is 4:1. Details are given in Table 3.

Table 3. For each benchmark data set, we show the size of the data set and the size of each training and validation component.

	Colon	Lymphoma	Leukemia	GINA	REGED	LUCAP	MARTI
Size of Data Set	62	96	72	3,468	500	2,000	500
Size of Training Component	50	77	58	2,774	400	1,600	400
Size of Validation Component	12	19	14	694	100	400	100

To compute the *v*-curves, we use LSVM as the learning machine. The only parameter involved in LSVM is the cost factor *C*. As mentioned earlier, we set *C* at 1 to allow a rather high degree of tolerance for training errors. For LSVM training, we adopt the linear version of LIBSVM (Fan et al., 2005). LIBLINEAR (Fan et al., 2008; Chang and Lin, 2008), a specialized tool kit for solving LSVM, may also be used. For the data sets in our experiments, we find that LIBSVM is as efficient as LIBLINEAR.

Next, we describe how to find an appropriate Π for each data set. From the Π training-validation pairs, we want to derive $v_{\Pi}(\sigma_{\Pi})$, where $v_{\Pi}(\cdot)$ and σ_{Π} are defined in (1) and (2) respectively. Furthermore, we want Π to be sufficiently large so that $v_{\Pi}(\sigma_{\Pi})$ is stable. For this reason, we require that $\Pi \geq \Gamma$, where Γ is the smallest T such that $|v_M(\sigma_M) - v_N(\sigma_N)| < 0.1\%$ for any integers M and N in the interval [T, T+4]. Table 4 shows the size of Γ determined by AMFES and the size of Π that we choose for each benchmark data set.

Table 4. The size of Γ determined by AMFES and the size of Π that we chose for each benchmark data set.

	Colon	Lymphoma	Leukemia	GINA	REGED	LUCAP	MARTI
Γ	64	62	53	16	8	13	36
Π	100	100	100	20	20	20	40

3.2.1 Adaptive Version versus Non-adaptive Version

First, we compare the adaptive and non-adaptive versions of AMFES. The adaptive version is the method that we propose in this paper. Its algorithm is specified in Steps I to IV in Section 2.1. The non-adaptive version corresponds to the initial stage of the adaptive version. To ensure that the comparison is fair, we require that both versions contain the same number of feature subsets. Since the adaptive version must generate $100 \times \log_2 d$ feature subsets, we let the non-adaptive version generate the same number of subsets. In the adaptive version, the subsets vary in size, whereas all subsets in the non-adaptive version comprise $d/2$ features.

Figures 1 to 7 show the v-curves produced by the adaptive and non-adaptive versions of AMFES. Due to space limitations, we only display each v-curve in the range [0, k_0], where $k_0 = \min(d, 200)$. In each figure, the horizontal line that passes through the vertical axis at the level $v_\Pi(d)$ is called the *baseline*. Table 5 shows σ_Π, $v_\Pi(\sigma_\Pi)$ and the standard deviation of $v_\Pi(\sigma_\Pi)$ derived by the two versions. The standard deviation of $v_\Pi(\sigma_\Pi)$ is defined as

$$std_\Pi = \sqrt{\sum_{p=1}^{\Pi}\left[v_p(\sigma_\Pi) - v_\Pi(\sigma_\Pi)\right]^2 / \Pi} \, .$$

Note that $v_\Pi(\sigma_\Pi)$ is the maximum accuracy rate and σ_Π is the point at which the maximum occurs. We call σ_Π the *peak location* and $v_\Pi(\sigma_\Pi)$ the *peak value*.

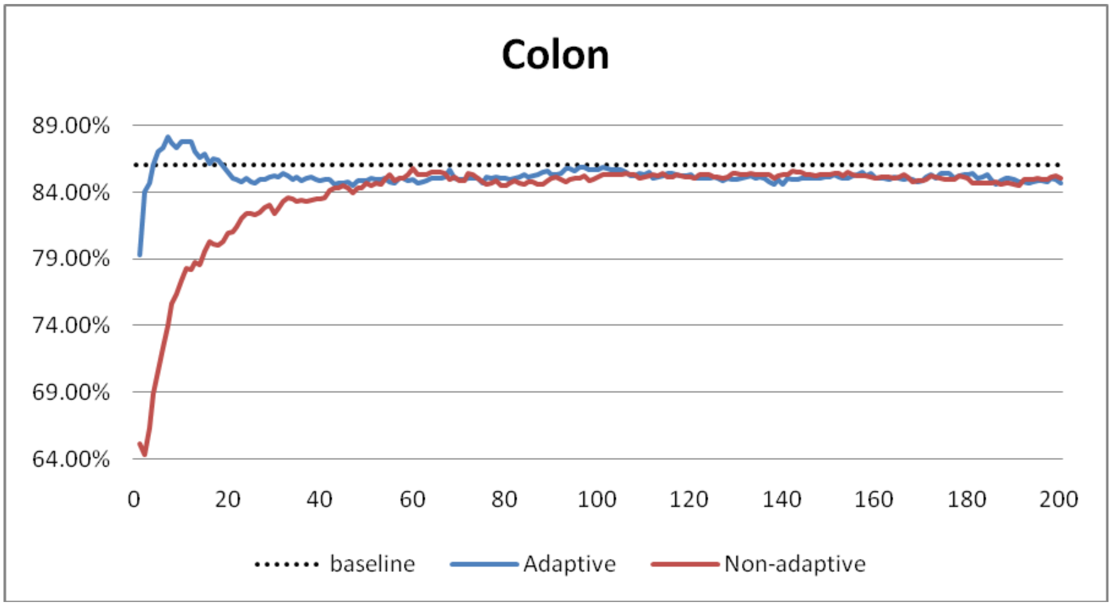

Figure 1. The v-curves produced by the adaptive and non-adaptive versions for the "Colon" data set.

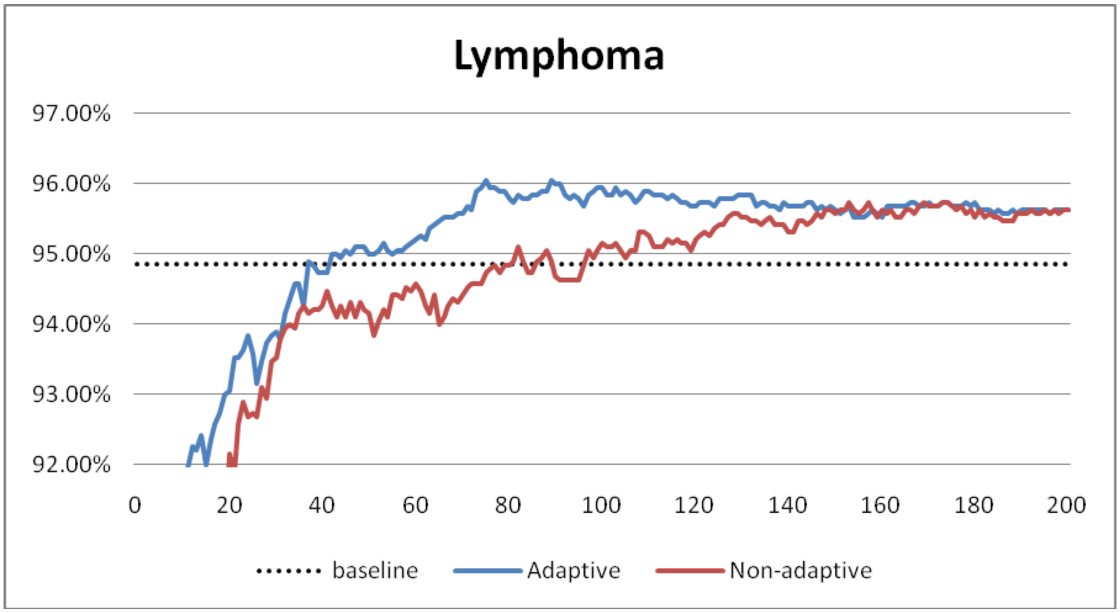

Figure 2. The *v*-curves produced by the adaptive and non-adaptive versions for the "Lymphoma" data set.

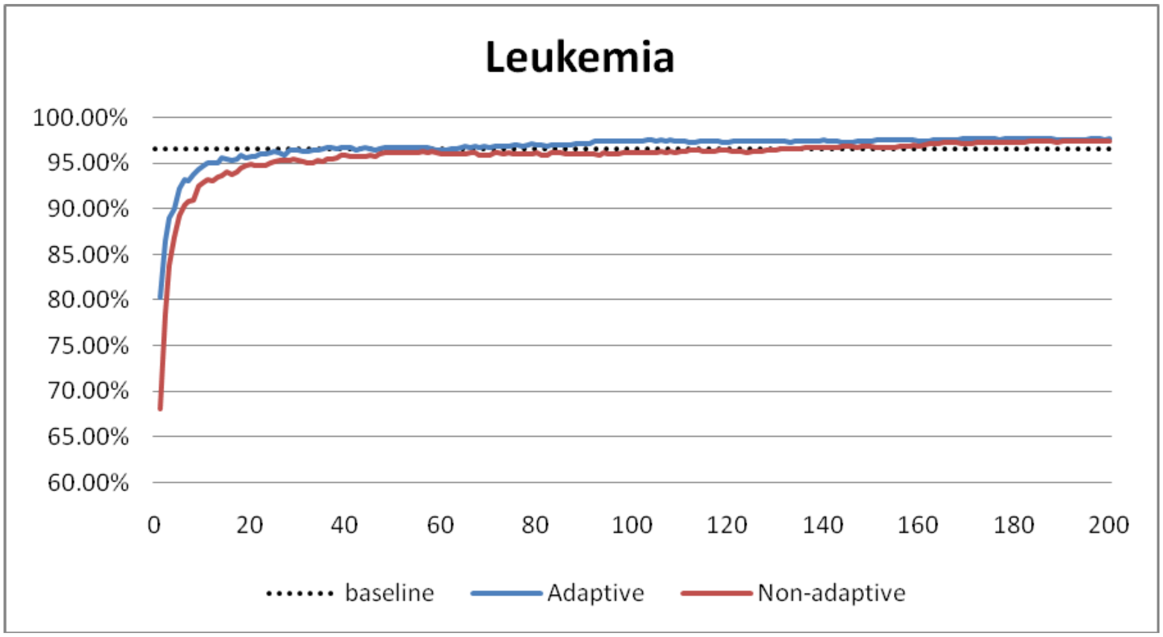

Figure 3. The *v*-curves produced by the adaptive and non-adaptive versions for the "Leukemia" data set.

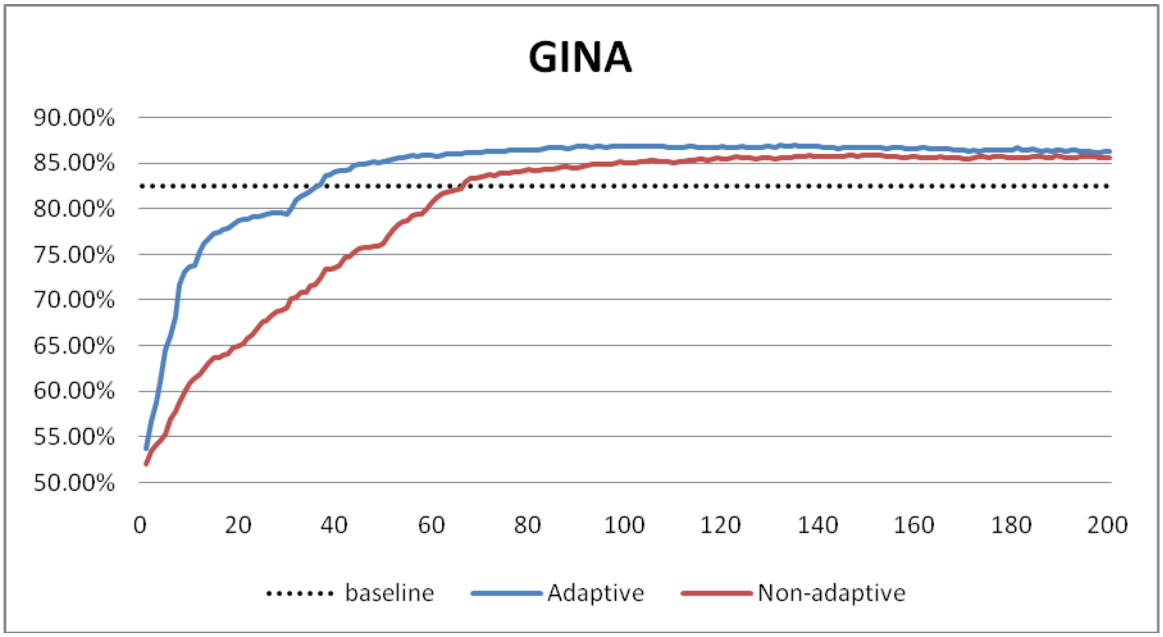

Figure 4. The *v*-curves produced by the adaptive and non-adaptive versions for the "GINA" data set.

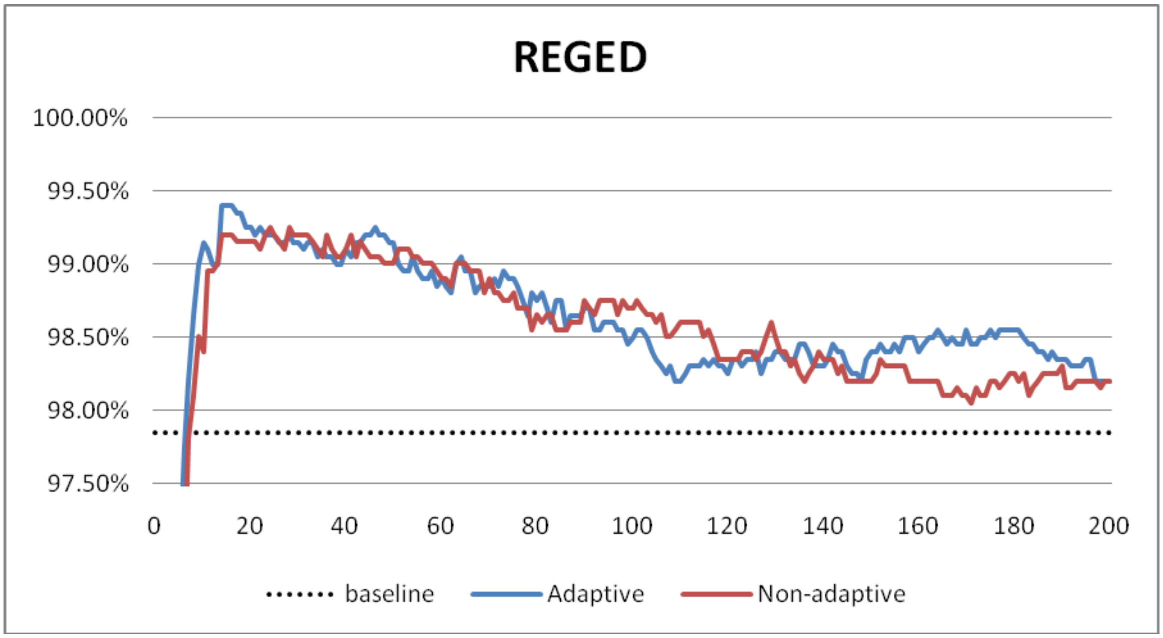

Figure 5. The *v*-curves produced by the adaptive and non-adaptive versions for the "REGED" data set.

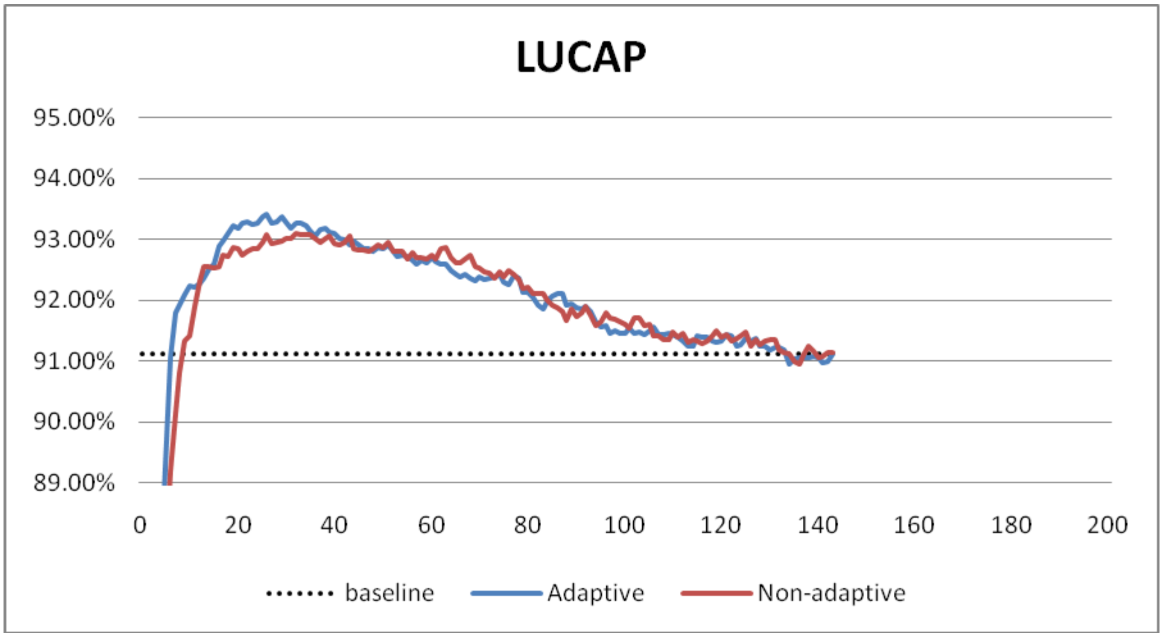

Figure 6. The *v*-curves produced by the adaptive and non-adaptive versions for the “LUCAP” data set.

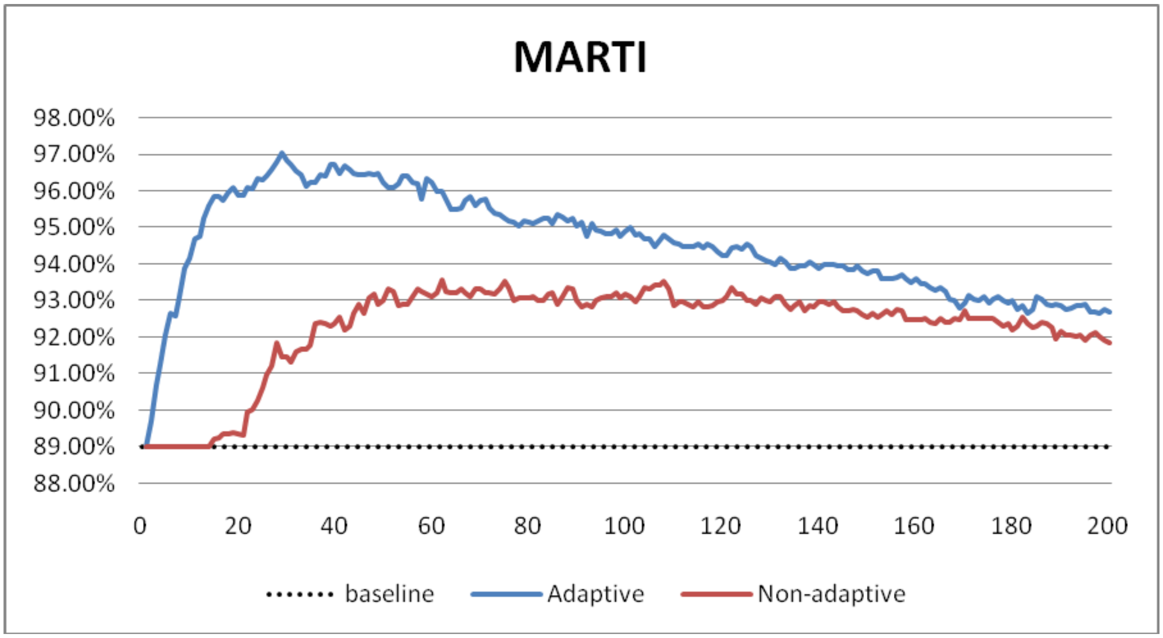

Figure 7. The *v*-curves produced by the adaptive and non-adaptive versions for the “MARTI” data set.

Table 5. The peak location σ_Π, the peak value $v_\Pi(\sigma_\Pi)$, and the standard deviation std_Π of the peak value derived by each approach, where $v_\Pi(\sigma_\Pi)$ and std_Π are expressed in percentages. We also show the baseline accuracy rates for reference.

		Colon	Lymphoma	Leukemia	GINA	REGED	LUCAP	MARTI
Adaptive	σ_Π	7	89	170	135	14	26	29
	$v_\Pi(\sigma_\Pi)$	**88.17**	**96.05**	**97.73**	**87.04**	**99.40**	**93.41**	**97.05**
	std_Π	8.26	4.61	3.80	1.14	0.58	1.16	1.56
Non-Adaptive	σ_Π	60	173	195	149	28	32	62
	$v_\Pi(\sigma_\Pi)$	85.75	95.74	97.40	85.90	99.25	93.08	93.55
	std_Π	9.30	4.92	4.41	1.25	0.70	1.19	2.46
Baseline	$v_\Pi(d)$	86.08	94.26	96.60	82.51	97.85	91.13	89.00

The v-curves obtained by the adaptive and non-adaptive versions are denoted as A-curves and N-curves respectively. We let σ_A and σ_N denote the peak of an A-curve and of an N-curve respectively. From Table 5 and Figures 1 to 7, we observe the following facts about each data set: (i) $\sigma_A < \sigma_N$, i.e., the peak of each A-curve occurs earlier than that of the corresponding N-curve; (ii) $v_\Pi(\sigma_A) > v_\Pi(\sigma_N)$, i.e., the peak value of each A-curve is higher than that of the corresponding N-curve; and (iii) each A-curve lies above the corresponding N-curve before σ_A is reached. The above facts demonstrate the superiority of the adaptive version over the non-adaptive version.

3.2.2 Comparing AMFES with RFE and CORR

We now compare the performances of the three methods, AMFES, RFE, and CORR, on the benchmark data sets, using the v-curve as the means of comparison. The CORR method ranks features in descending order of the following score.

$$corr(f) = \frac{\sum_{i=1}^{n}\left[\mathbf{x}_{i,f} - mean(f)\right]\left[y_i - mean(y)\right]}{\sqrt{\sum_{i=1}^{n}\left[\mathbf{x}_{i,f} - mean(f)\right]^2 \left[y_i - mean(y)\right]^2}},$$

where $\mathbf{x}_{i,f}$ is the value of a feature f for $\mathbf{x}_i$, mean(f) is the average value of f, y_i is the label of $\mathbf{x}_i$, and mean(y) is the average value of the label.

Figures 8 to 14 show the v-curves derived by the three methods based on the training-validation pairs listed in Table 6. The table also shows the peak location σ_Π, the peak value $v_\Pi(\sigma_\Pi)$, and the standard deviation of $v_\Pi(\sigma_\Pi)$ produced by each method.

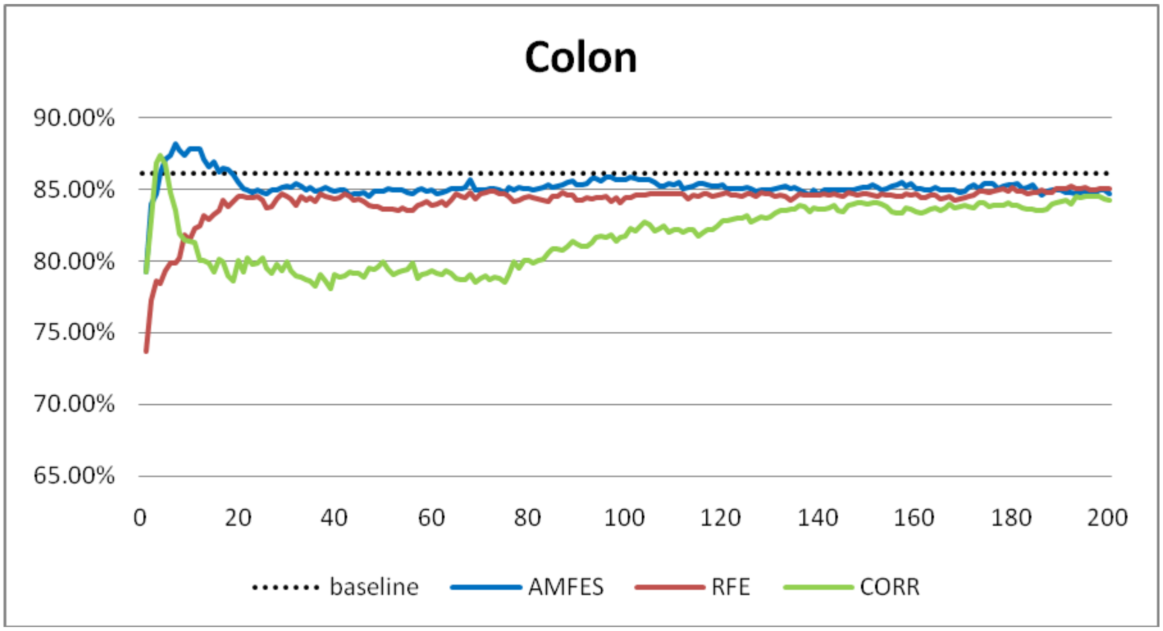

Figure 8 The *v*-curves derived by the three methods for the “Colon” data set.

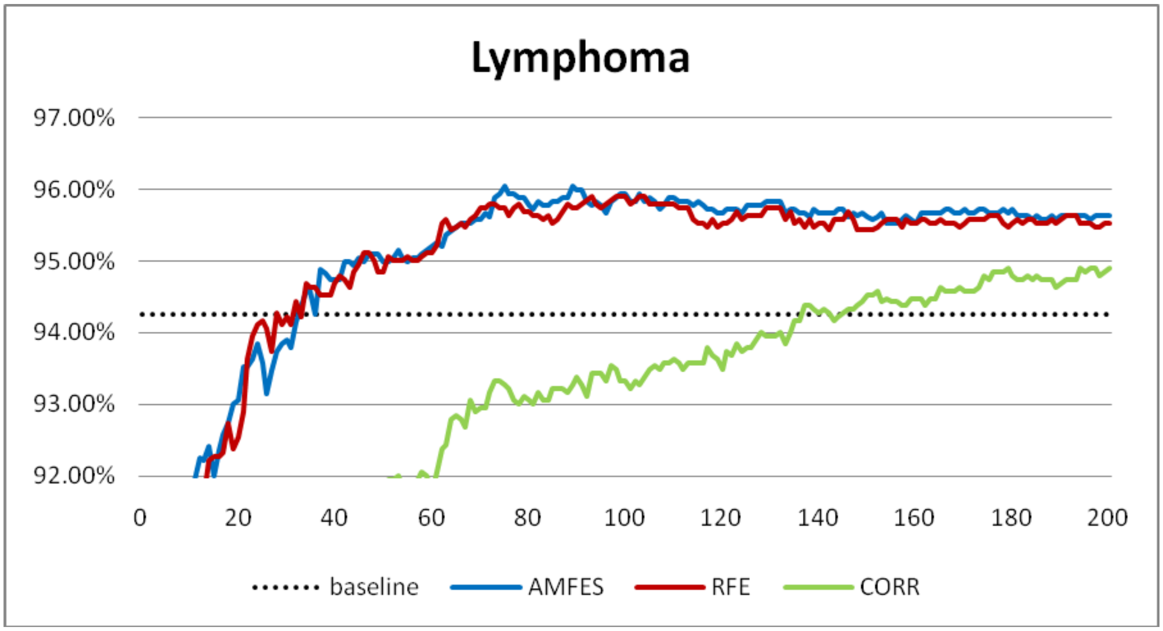

Figure 9. The *v*-curves derived by the three methods for the “Lymphoma” data set.

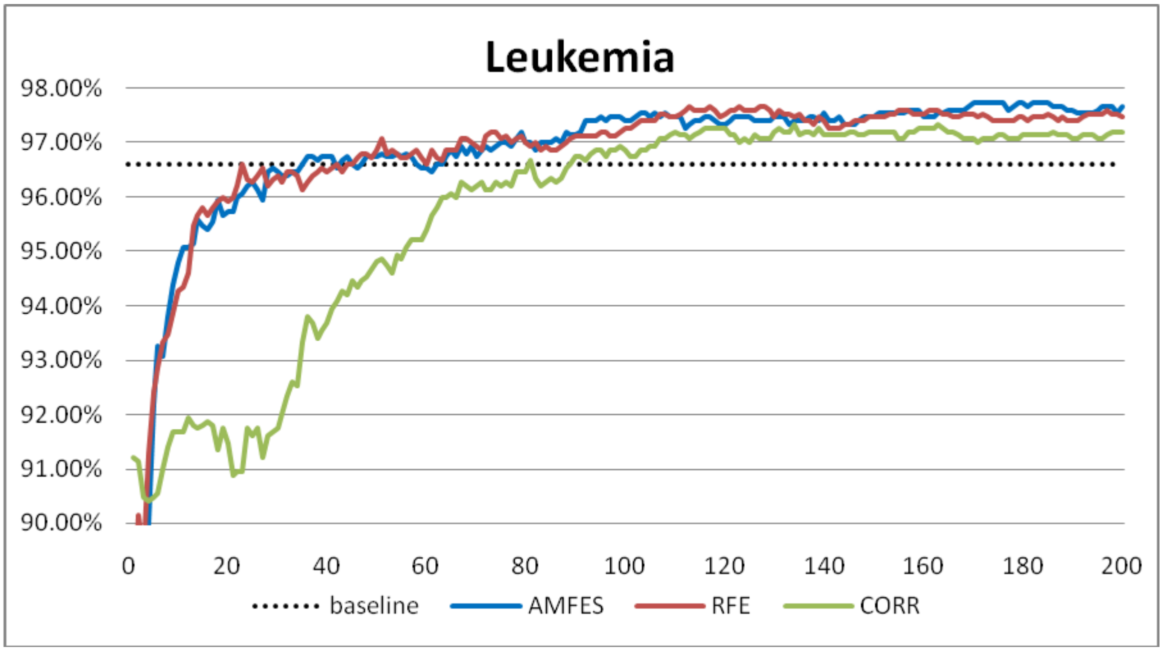

Figure 10. The *v*-curves derived by the three methods for the "Leukemia" data set.

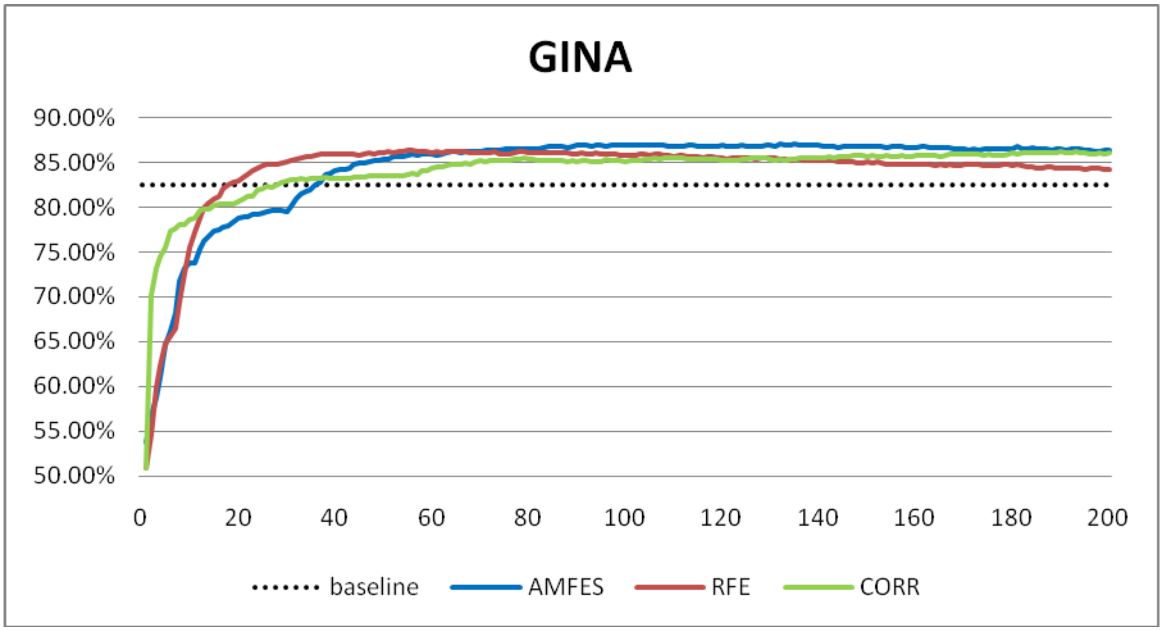

Figure 11. The *v*-curves derived by the three methods for the "GINA" data set.

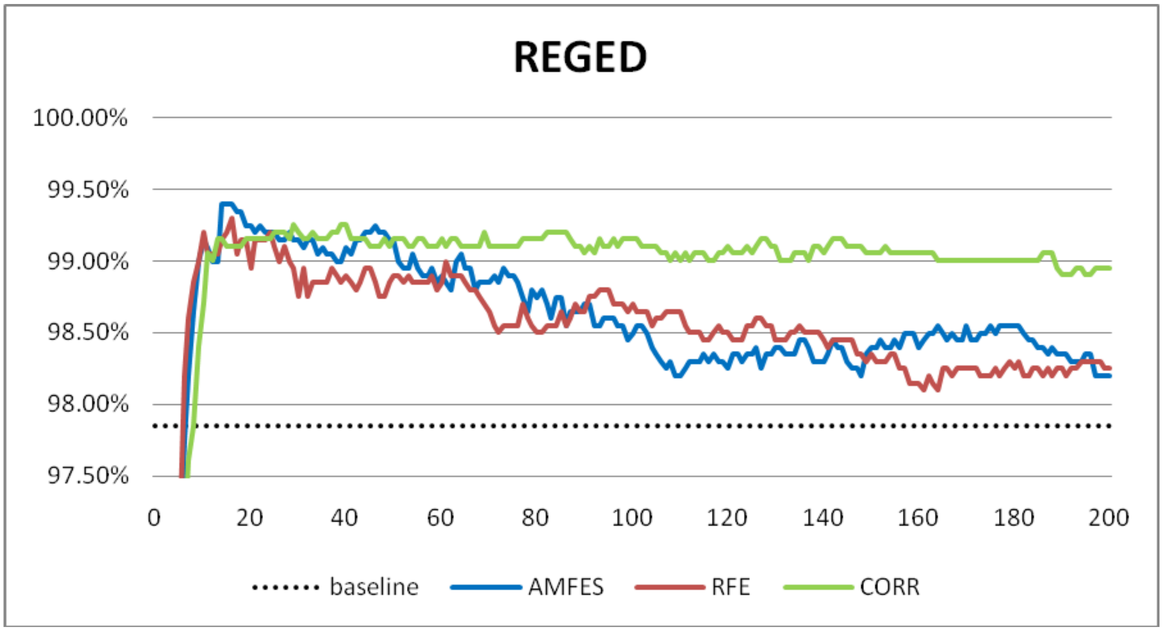

Figure 12. The *v*-curves derived by the three methods for the “REGED” data set.

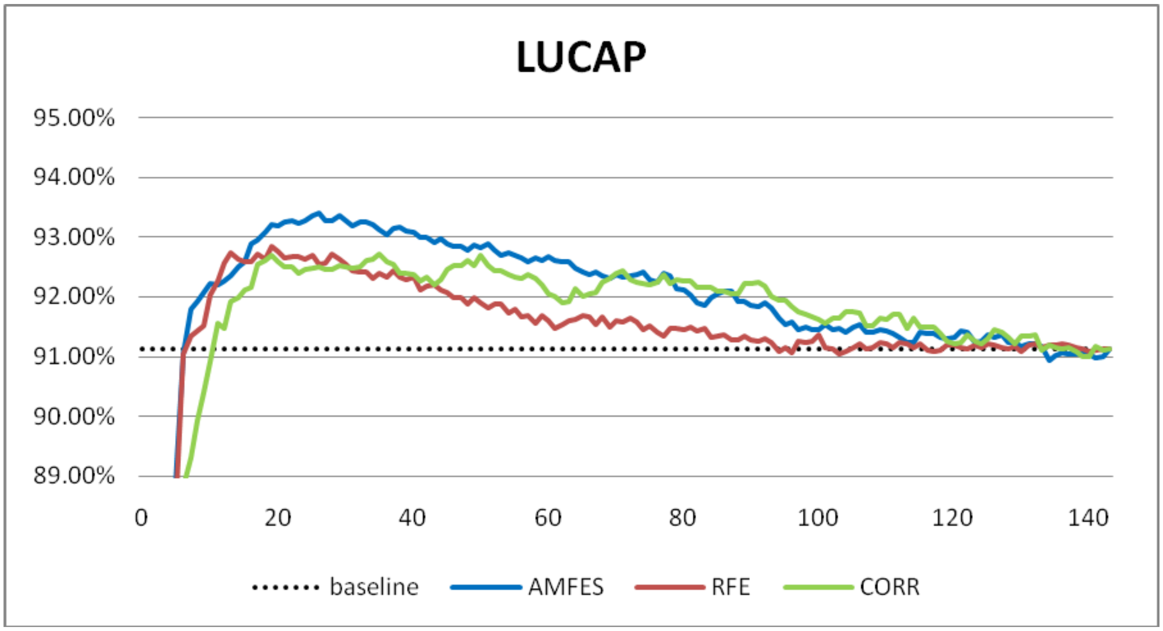

Figure 13. The *v*-curves derived by the three methods for the “LUCAP” data set.

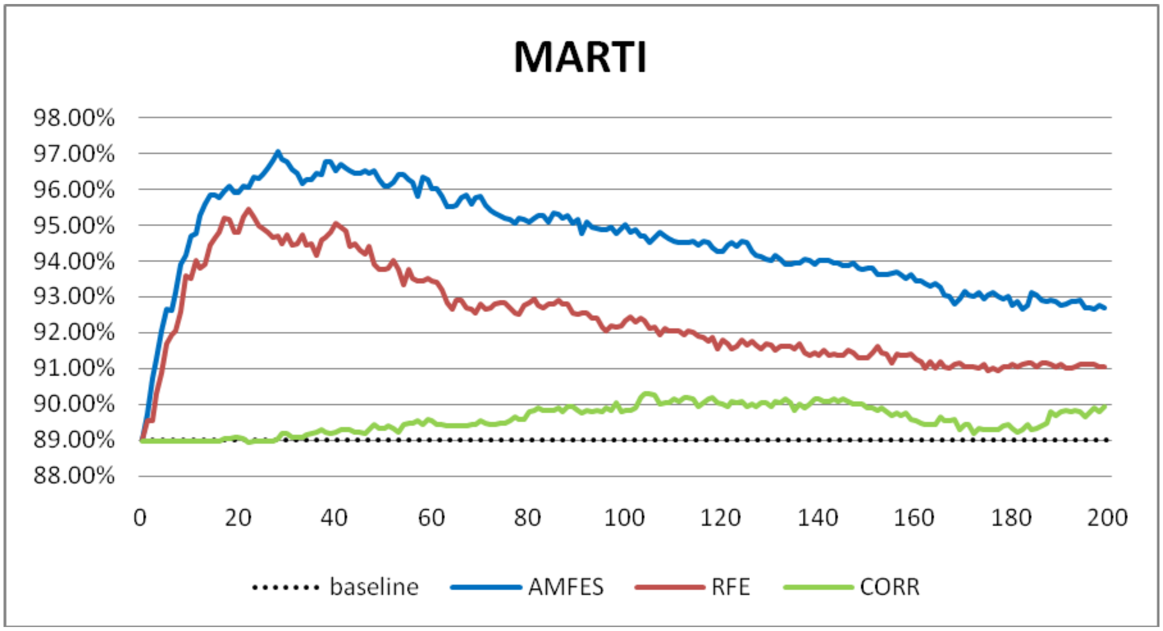

Figure 14. The *v*-curves derived by the three methods for the “MARTI” data set.

Table 6. The peak location σ_Π, the peak value $v_\Pi(\sigma_\Pi)$, and the standard deviation std_Π of the peak value derived by each method; $v_\Pi(\sigma_\Pi)$ and std_Π are expressed in percentages. We also include the baseline accuracy rates for reference.

		Colon	Lymphoma	Leukemia	GINA	REGED	LUCAP	MARTI
AMFES	σ_Π	7	89	170	135	14	26	29
	$v_\Pi(\sigma_\Pi)$	**88.17**	**96.05**	**97.73**	**87.04**	**99.40**	**93.41**	**97.05**
	std_Π	8.26	4.61	3.80	1.14	0.58	1.16	1.56
RFE	σ_Π	192	103	117	55	16	19	23
	$v_\Pi(\sigma_\Pi)$	85.25	95.90	97.67	86.31	99.30	92.84	95.45
	std_Π	8.73	4.83	3.93	1.14	0.70	1.30	2.25
CORR	σ_Π	4	179	134	193	39	35	105
	$v_\Pi(\sigma_\Pi)$	87.33	94.89	97.33	86.28	99.25	92.71	90.30
	std_Π	9.46	4.99	4.00	1.17	0.68	1.15	1.76
Baseline	$v_\Pi(d)$	86.08	94.26	96.60	82.51	97.85	91.13	89.00

The results in Table 6 and Figures 8 to 14 show that, in terms of the peak values, AMFES outperforms the other two methods, and CORR performs rather weakly on all the data sets, except “Colon”. RFE’s performance is comparable to or weaker than that of AMFES on all the data sets.

Table 7. The average time (in seconds) required by each method to complete the ranking procedure.

	Colon	Lymphoma	Leukemia	GINA	REGED	LUCAP	MARTI
AMFES	1.92	6.62	10.88	1,586.22	24.01	96.98	794.34
RFE	23.12	244.97	635.45	10,862.18	211.69	66.52	1797.32
CORR	**0.01**	**0.04**	**0.02**	**0.36**	**0.06**	**0.16**	**0.08**

Table 7 shows the time required by each method to complete the ranking procedure. All the experiments were conducted on an Intel Xeon E5345 CPU 2.33 GHz with a 2GB RAM. Since we extracted Π training-validation pairs from each data set, we conducted Π ranking procedures for each set. The table shows the average time that each method requires for one ranking procedure. CORR involves the simplest calculation, so it is the fastest method; however, in most cases, its performance in terms of the peak values is not acceptable, as shown in Table 7. On the other hand, AMFES is much faster than RFE on all data sets, except for "LUCAP," whose dimensionality is only 143. Hence, AMFES is more effective than the other two methods.

3.3 Comparison of the Methods: The Second Approach

In this sub-section, we examine the effect of the feature selection procedure. Since we demonstrated the advantage of the adaptive version of AMFES over the non-adaptive version in Section 3.2.1, we only consider the adaptive version in this subsection. Moreover, we use *synthetic* data sets to evaluate this version of AMFES as well as RFE and CORR.

We use the two groups of synthetic data sets described in Table 2. Both groups comprise three nested data sets. From each data set, we extract Π training-validation pairs The ratio of the training component to the validation component is 4:1. To compute the accuracy rate of the classifier built on the selected features, we generate a test data set for each group that is independent of all the data sets in that group. Furthermore, Group I and Group II are generated Ω times so that we obtain an average result from Ω procedures. In each procedure, we generate nested data sets, perform feature selection on the sets, and compute the test accuracy of the selected features. We show all the relevant quantities in Table 8.

Table 8. The values of Ω and Π, and the sizes of the training components, validation components, and test data sets.

Data Set	I-1	I-2	I-3	II-1	II-2	II-3
Total Number of Features	200			2,000		
Ω	20			20		
Π	20			20		
Size of Data Set	500	1,000	1,500	500	1,000	1,500
Size of Training Component	400	800	1,200	400	800	1,200
Size of Validation Component	100	200	300	100	200	300
Size of Test Data Set	1,000			1,000		

By using synthetic data sets in the experiment, we can generate as many training, validation, and test samples as we wish. When we conduct the same experiment on the benchmark data sets, we have to employ a double-loop experiment design, where the inner loop selects features and the outer loop computes their test accuracy. This design forces us to generate at least Γ^2 training-validation pairs. As a result, we have to generate at least 64^2 such pairs for "Colon" and 62^2 such pairs for "Lymphoma" (cf. Table 4). Using synthetic data sets, on the other hand, allows us to generate $\Omega \cdot \Pi$ training-validation pairs, which are 20×20 pairs, for all the data sets (cf. Table 8). The synthetic data sets obviously incur a much smaller computational cost than the benchmark data sets. Another advantage of using synthetic data sets is that we can specify accurate ground truths for them. As a result, we can define many performance measures that we cannot define on benchmark data sets. We can then use those measures to compare AMFES with other methods.

As mentioned above, we repeat the procedure Ω times. In the following, we assume that the ω^{th} procedure is being implemented. From the given data set, we derive a set of selected features. Let the number of selected features be σ_ω. We then train a classifier on the data set using the σ_ω selected features as its features and apply the classifier to the test data set in order to compute its test accuracy rate t_ω. Finally, let the number of essential features captured by the selection procedure be ε_ω. We define

$$t_\Omega = \sum\nolimits_{\omega=1}^{\Omega} t_\omega / \Omega ,$$

$$\sigma_{\Omega} = \sum_{\omega=1}^{\Omega} \sigma_{\omega} / \Omega \text{, and}$$

$$\varepsilon_{\Omega} = \sum_{\omega=1}^{\Omega} \varepsilon_{\omega} / \Omega .$$

The above quantities are, respectively, the average test accuracy rate, the average number of selected features, and the average number of essential features captured by the selection procedure.

In addition to these quantities, for $p = 1, \ldots, \Omega$, we define

$$P_p = \frac{\varepsilon_p}{\sigma_{\mathrm{p}}},$$

$$R_p = \frac{\varepsilon_p}{\text{number of essential features}} \text{, and}$$

$$F_p = 2 \cdot \frac{P_p \cdot R_p}{P_p + R_p} .$$

P_p and R_p are, respectively, the precision and recall rates of essential features captured by the selection procedure. F_p is the F_1-measure (van Rijsbergen, 1979) for the captured essential features for pair p. Finally, let P_{Ω}, R_{Ω}, and F_{Ω} be the averages of P_p, R_p, and F_p respectively.

Our feature selection procedure derives a set of selected features from Π training-validation pairs. We compare this procedure with an alternative procedure that derives a set of selected features from *each* training-validation pair. There are $\Omega \cdot \Pi$ such pairs. From each pair q, we derive a set of selected features and train a classifier on the learning component using the selected features as its features. We then apply the classifier to the corresponding test data set to obtain t_q. We define

$$t_{\Omega \cdot \Pi} = \sum_{q=1}^{\Omega \cdot \Pi} t_q / (\Omega \cdot \Pi),$$

$$\sigma_{\Omega \cdot \Pi} = \sum_{q=1}^{\Omega \cdot \Pi} \sigma_q / (\Omega \cdot \Pi) \text{, and}$$

$$\varepsilon_{\Omega \cdot \Pi} = \sum_{q=1}^{\Omega \cdot \Pi} \varepsilon_q / (\Omega \cdot \Pi) .$$

For $q = 1, \ldots, \Omega\cdot\Pi$, let P_q and R_q be the precision and recall rates respectively; and let F_q be the F_1-measure for the captured essential features. Finally, let $P_{\Omega\cdot\Pi}$, $R_{\Omega\cdot\Pi}$, and $F_{\Omega\cdot\Pi}$ be the averages of P_q, R_q, and F_q respectively.

We call the proposed feature selection procedure the *Ω procedure* and the alternative procedure the *Ω·Π procedure*.

Next, we discuss the results of applying AMFES, RFE, and CORR to all the synthetic data sets. Table 9 shows the average test accuracy rates of the Ω procedure and the Ω·Π procedure. We include the baseline test accuracy rates in the table for reference. For a synthetic data set D, we obtain the baseline test accuracy rate t as follows. First, we build a classifier on D by using all the features as its features, and then apply the classifier to the test data set to compute t. Table 10 shows the average numbers of selected features; Table 11 shows the average numbers of essential features captured by the selection procedure: and Table 12 shows the average F_1-measures for the captured essential features.

Table 9. The results of applying the three compared methods to the synthetic data sets. The table shows the average test accuracy rates and their standard deviations (in parentheses), derived by the baseline method, the Ω procedure, and the Ω·Π procedure.

	Baseline	AMFES		RFE		CORR	
	t	t_{Ω}	$t_{\Omega\cdot\Pi}$	t_{Ω}	$t_{\Omega\cdot\Pi}$	t_{Ω}	$t_{\Omega\cdot\Pi}$
I-1	86.00	**96.29** (0.35)	95.70 (0.67)	95.55 (0.69)	95.16 (0.81)	95.28 (0.22)	95.06 (0.58)
I-2	94.10	**97.02** (0.24)	96.61 (0.67)	96.90 (0.19)	96.40 (0.73)	95.37 (0.17)	95.31 (0.50)
I-3	95.50	**97.32** (0.22)	97.12 (0.41)	97.27 (0.20)	97.10 (0.35)	95.35 (0.22)	95.47 (0.48)
II-1	55.30	**95.04** (0.40)	94.79 (0.66)	94.88 (0.48)	94.58 (0.66)	94.94 (0.44)	94.72 (0.77)
II-2	60.10	**96.24** (0.62)	95.93 (0.67)	95.36 (0.15)	95.07 (0.49)	95.35 (0.16)	95.20 (0.46)
II-3	63.60	**97.31** (0.20)	97.09 (0.49)	95.89 (0.44)	95.47 (0.54)	95.26 (0.17)	95.15 (0.29)

Table 10. The results of applying the three compared methods to the synthetic data sets. The table shows the average numbers of selected features and their standard deviations (in parentheses), derived by the Ω procedure and the Ω·Π procedure.

	AMFES		RFE		CORR	
	σ_{Ω}	$\sigma_{\Omega\cdot\Pi}$	σ_{Ω}	$\sigma_{\Omega\cdot\Pi}$	σ_{Ω}	$\sigma_{\Omega\cdot\Pi}$
I-1	12.90 (2.88)	11.83 (7.98)	8.05 (1.43)	9.29 (4.61)	7.15 (0.36)	11.36 (10.53)
I-2	16.45 (3.23)	17.09 (14.27)	11.60 (1.20)	12.90 (9.71)	7.15 (0.36)	19.05 (29.87)
I-3	14.25 (1.81)	17.04 (11.17)	13.95 (1.75)	13.94 (5.93)	7.90 (1.26)	38.33 (45.21)
II-1	7.60 (1.02)	8.62 (2.47)	7.35 (0.79)	7.47 (1.90)	7.75 (0.89)	9.21 (3.59)
II-2	10.60 (1.77)	11.49 (5.03)	7.00 (0.00)	7.76 (2.24)	7.20 (0.51)	10.42 (7.79)
II-3	14.05 (1.75)	14.74 (6.17)	7.95 (0.80)	9.10 (3.30)	7.25 (0.89)	12.13 (8.38)

Table 11. The results of applying the three methods to the synthetic data sets. The table shows the average numbers of essential features captured by the selection procedure and their standard deviations (in parenthesis), derived by the Ω procedure and the Ω·Π procedure.

	AMFES		RFE		CORR	
	ε_{Ω}	$\varepsilon_{\Omega\cdot\Pi}$	ε_{Ω}	$\varepsilon_{\Omega\cdot\Pi}$	ε_{Ω}	$\varepsilon_{\Omega\cdot\Pi}$
I-1	12.65 (2.65)	9.99 (10.90)	7.85 (1.06)	7.58 (1.21)	7.05 (0.22)	7.47 (0.95)
I-2	14.40 (1.07)	12.45 (14.65)	11.15 (1.06)	10.07 (2.07)	7.10 (0.30)	8.15 (1.76)
I-3	13.85 (1.39)	13.02 (2.27)	13.00 (1.14)	11.93 (1.90)	7.60 (0.73)	9.27 (2.31)
II-1	7.15 (0.48)	7.31 (0.70)	6.85 (0.36)	6.65 (0.48)	7.00 (0.00)	7.05 (0.27)
II-2	10.50 (1.57)	9.88 (2.22)	7.00 (0.00)	6.88 (0.38)	7.00 (0.00)	7.09 (0.38)
II-3	13.70 (1.31)	12.79 (2.36)	7.95 (0.80)	7.46 (0.53)	7.00 (0.00)	7.30 (0.52)

Table 12. The results of applying the three methods to the synthetic data sets. The table shows the F_1-measure for the captured essential features and their standard deviations (in parentheses), derived by the Ω procedure and the Ω·Π procedure.

	AMFES		RFE		CORR	
	F_{Ω}	$F_{\Omega\cdot\Pi}$	F_{Ω}	$F_{\Omega\cdot\Pi}$	F_{Ω}	$F_{\Omega\cdot\Pi}$
I-1	**89.60** (11.12)	73.72 (12.85)	67.82 (5.47)	62.88 (6.69)	63.66 (1.59)	59.52 (7.99)
I-2	**92.00** (6.47)	81.08 (15.38)	83.68 (4.73)	73.81 (10.74)	64.09 (1.92)	57.71 (12.53)
I-3	**94.49** (4.68)	84.46 (13.68)	89.67 (3.38)	83.11 (8.85)	66.25 (3.27)	48.87 (18.13)
II-1	**63.25** (2.19)	62.17 (5.03)	61.29 (2.30)	59.33 (3.83)	61.63 (2.32)	59.12 (6.02)
II-2	**81.57** (7.33)	74.43 (9.78)	63.64 (0.00)	60.74 (4.34)	63.09 (1.37)	58.24 (8.70)
II-3	**94.14** (4.25)	86.49 (10.48)	67.07 (2.67)	62.54 (6.04)	63.01 (2.19)	56.70 (9.83)

Based on the test accuracy results shown in Table 9, we make the following observations:

(1) Under all three methods, t_{Ω} and $t_{\Omega \cdot \Pi}$ increase with the size of the training data. Moreover, t_{Ω} and $t_{\Omega \cdot \Pi}$ are generally higher for Group I compared to Group II; that is, t_{Ω} and $t_{\Omega \cdot \Pi}$ for I-1 are higher than those for II-1, etc.

(2) For AMFES and RFE, t_{Ω} is uniformly better than $t_{\Omega \cdot \Pi}$. For CORR, t_{Ω} and $t_{\Omega \cdot \Pi}$ are about the same.

(3) When we compare the three methods in terms of t_{Ω} and $t_{\Omega \cdot \Pi}$, we find that AMFES performs better than RFE, and significantly better than CORR.

Next, we consider the feature selection results shown in Tables 10 to 12. It is rather difficult to compare the numbers of selected features and the numbers of captured essential features. This is because a method may select a lot of features, but it may only capture a few essential features. On the other hand, the F_1-measure for the captured essential features provides a more balanced view. Thus, we focus on this measure, as shown in Table 12. We make the following observations:

(i) For AMFES and RFE, F_{Ω} and $F_{\Omega \cdot \Pi}$ increase with the size of the training data, although there are exceptions due to statistical fluctuations; however, the increasing trend does not exist in CORR. Furthermore, Group I has uniformly higher F_{Ω} and $F_{\Omega \cdot \Pi}$ than Group II.

(ii) For all three methods, F_{Ω} is uniformly better than $F_{\Omega \cdot \Pi}$.

(iii)When we compare the three methods in terms of F_{Ω} and $F_{\Omega \cdot \Pi}$, we find that AMFES outperforms RFE by a significant margin, especially when the size of the training data is small or the number of irrelevant features is large; CORR is the weakest method among the three.

The last result is as expected because CORR evaluates features according to their correlations with labels. From the way mean vectors are generated (cf. Section 3.1), we infer that $\boldsymbol{\mu}_1$ differs from $\boldsymbol{\mu}_2$ in about half of the features, which is 7.5. The number of essential features found by CORR is close to this figure.

We conclude this sub-section by examining another performance measure. For a given ranking procedure, we define φ to be the largest k such that all top-k features are essential features. This measure does not evaluate how many essential features have been selected; instead, it considers how many of them are ranked above all irrelevant features. Once again, let φ_{Ω} and $\varphi_{\Omega\cdot\Pi}$ be the average number of such features in the Ω procedure and the Ω·Π procedure, respectively. We show all the relevant results in Table 13.

Table 13. Applying the three methods to the synthetic data sets, we show the number of essential features that are ranked above all irrelevant features, produced by the Ω procedure or the Ω·Π procedure.

	AMFES		RFE		CORR	
	φ_{Ω}	$\varphi_{\Omega\cdot\Pi}$	φ_{Ω}	$\varphi_{\Omega\cdot\Pi}$	φ_{Ω}	$\varphi_{\Omega\cdot\Pi}$
I-1	14.75 (0.62)	**14.85** (0.48)	8.45 (0.59)	8.44 (0.62)	7.55 (0.50)	7.26 (0.47)
I-2	**15.00** (0.00)	**15.00** (0.00)	11.50 (1.24)	11.23 (1.18)	7.45 (0.59)	7.26 (0.50)
I-3	**15.00** (0.00)	**15.00** (0.00)	13.40 (0.49)	13.41 (0.69)	8.20 (0.40)	7.78 (0.55)
II-1	7.10 (0.30)	**7.63** (0.76)	7.00 (0.00)	7.00 (0.00)	7.00 (0.00)	7.01 (0.11)
II-2	**13.25** (0.77)	11.77 (1.28)	7.05 (0.22)	7.05 (0.21)	7.00 (0.00)	7.02 (0.15)
II-3	**15.00** (0.00)	14.98 (0.15)	8.05 (0.22)	7.59 (0.52)	7.40 (0.49)	7.17 (0.37)

For all three methods, φ_{Ω} is comparable to $\varphi_{\Omega\cdot\Pi}$; therefore, we only discuss the results for φ_{Ω}. We observe that, for AMFES, $\varphi_{\Omega} > \varepsilon_{\Omega}$ in all cases, although the gap gets smaller as the size of the data set increases. This means that AMFES places more essential features before irrelevant features than it actually takes as selected features. The same is true of RFE, although to a lesser degree. On the other hand, CORR places about the same number of essential features before irrelevant features as it takes as selected features. Moreover, AMFES has generally higher φ_{Ω} than RFE and CORR, especially when the size of the training data is small or the number of irrelevant features is large.

We summarize the findings in Tables 9 to 13 as follows.

(a) The Ω procedure, which ranks and selects features according to Π training-validation pairs, outperforms the Ω·Π procedure, in which features are ranked and selected based on only one training-validation pair.

(b) AMFES's feature selection procedure outperforms RFE and CORR in terms of the test accuracy rate and the F_1-measure for the captured essential features.

(c) The feature selection procedures of AMFES and RFE manifest the COD effect in that their performance measures increase with the training data size, and decrease with the number of irrelevant features. CORR's performance measures are relatively invariant to the training data size, but they are usually inferior to those of AMFES and RFE.

(d) AMFES places more essential features in the top ranks (i.e., above all irrelevant features) than it actually takes as selected features. The same is true of RFE, although to a lesser degree. CORR places about the same number of essential features in the top ranks as it takes as selected features. Moreover, AMFES outperforms RFE and CORR in terms of the number of essential features placed in the top ranks.

4. Conclusion

The proposed feature evaluation method, AMFES, ranks features according to their strengths, which are derived from multiple feature subsets. Initially, AMFES ranks all the features; then, in each subsequent stage, it takes the features whose ranks in the previous stage were above the median rank and re-ranks them in the same fashion as it did in the first stage. The complexity of our method is thus O($\log_2 d$) This is lower than the complexity of RFE, which is O(d). To cope with the COD effect, we propose a procedure that derives a set of selected features from various training-validation pairs, rather than from one pair. By so doing, we obtain a more stable and more effective set of selected features than those derived from a single pair. In the experiments conducted to compare AMFES with the other methods, AMFES outperformed RFE and CORR in terms of the feature-ranking performance, as expressed in the v-curve. AMFES's performance was also superior in terms of feature selection, as expressed in the test accuracy rate, the F_1-measure for captured essential features, and the number of features ranked above all irrelevant features. Finally, the superior performance of AMFES is particularly noticeable on data sets with a small number of training samples or a large number of irrelevant features, as manifested in the experiments on synthetic data.

References

A. A. Alizadeh et al. Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, 403:503-511, 2000.

H. Almuallim and T. G. Dietterich. Learning with many irrelevant features. In *9th National Conference on Artificial Intelligence*, San Jose, CA, 547-552, 1991.

U. Alon et al. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissue probed by oligonucleotide arrays. *Proceedings of the National Academy of Science*, 96(12):6745-6750, 1999.

E. Amaldi and V. Kann. On the approximability of minimizing non-zero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, 209(1):237-260, 1998.

J. Bi, K. Bennett, M. Embrechts, C. Breneman, and M. Song. Dimensionality reduction via sparse support vector machines. *Journal of Machine Learning Research*, 3:1229–1243, 2003.

A. L. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2):245-271, 1997.

L. Breiman. Random Forests. *Machine Learning*, 45(1):5-32, 2001.

L. Breiman, J. H. Freidman, R. A. Olsen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, 1984.

C. Cardie. Using decision trees to improve case-based learning. In *10th International Conference on Machine Learning*, Amherst, MA, 25-32, 1993.

Y.-W. Chang and C.-J. Lin. Feature ranking using linear SVM. In *JMLR Workshop and Conference Proceedings: Causation and Prediction Challenge*, 3:53-64, 2008.

O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3):131-159, 2002.

S. Chen, S. A. Billings, and W. Luo. Orthogonal least squares and their applications to non-linear system identification. *International Journal of Control*, 50:1873-1896, 1989.

R.-E. Fan, P.-H. Chen, and C.-J. Lin. Working set selection using second order information for training SVM. *Journal of Machine Learning Research*, 6:1889-1918, 2005.

R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871-1874, 2008.

G. Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3:1289-1305, 2003.

C. Gentile. Fast feature selection from microarray expression data via multiplicative large margin algorithms. In S. Thrun, L. Saul, and B. Schölkopf (editors), *Advances in Neural Information Processing Systems*, MIT Press, Cambridge, MA, 2004.

T. R. Golub, et al. Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. *Science*, 286(5439):531-537, 1999.

Y. Grandvalet and S. Canu. Adaptive scaling for feature selection in SVMs. In *Neural Information Processing Systems*, Cambridge, MA, 2002.

I. Guyon. Design of experiments of the NIPS 2003 variable selection benchmark. http://www.nipsfsc.ecs.soton.ac.uk/papers/Datasets.pdf, 2003.

I. Guyon and A. Elisseeff (editors). JMRL special Issue on variable and feature selection. *Journal of Machine Learning Research*, 3, 2003.

I. Guyon, A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157-1182, 2003.

I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3): 389-422, 2002.

M. A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In 1*7th International Conference on Machine Learning*, 359-366, 2000.

T. K. Ho. Random decision forests. In *3rd International Conference on Document Analysis and Recognition*, 278−282, 1995.

T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832−844, 1998.

T. Jebara and T. Jaakkola. Feature selection and dualities in maximum entropy discrimination. In *16th Annual Conference on Uncertainty in Artificial Intelligence*, 2000.

K. Kira and L. A. Rendell. A practical approach to feature selection. In *9th International Workshop on Machine Learning*, Aberdeen, Scotland, 249-256, 1992.

R. Kohavi, G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2): 273-324, 1997.

D. Koller and M. Sahami. Toward optimal feature selection. In *13th International Conference on Machine Learning*, Bari, Italy, 284-292, 1996.

I. Konenko. Estimating attributes: Analysis and extensions of RELIEF. In *7th European Conference on Machine Learning*, 171-182, Catania, Italy, 1994.

C. Lai, M. J. T. Reinders, and L. Wessels. Random subspace method for multivariate feature selection. *Pattern Recognition Letters*, 27(10):1067-1076, 2006.

T. N. Lal, O. Chapelle, J. Weston, and A. Elisseeff. Embedded methods. In I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh (editors) *Feature Extraction: Foundations and Applications*, Springer, Berlin, 137-165, 2006.

T. N. Lal, M. Schröder, T. Hinterberger, J. Weston, M. Bogdan, N. Birbaumer, and B. Schölkopf. Support vector channel selection in BCI. *IEEE Transactions on Biomedical Engineering. Special Issue on Brain-Computer Interfaces*, 51(6):1003–1010, 2004.

Y. LeCun, J. S. Denker, and S. A. Solla. Optimal brain damage. In D. S. Touretzky (ed.) *Advances in Neural Information Processing Systems II*, San Mateo, CA, 598-605, 1990.

D. D. Lewis. Feature selection and feature extraction for text categorization. In *Workshop on Speech and Natural Language*, San Francisco, CA, 212-217, 1992.

T. Marill and D. Green. On the effectiveness of receptors in recognition systems. *IEEE Transactions on Information Theory*, 9:11-17, 1963.

R. Meiri and J. Zahavi. Using simulated annealing to optimize the feature selection problem in marketing applications. *European Journal of Operational Research*, 171(3):842-858, 2006.

P. M. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. *IEEE Transactions on Computers*, C-26(9):917-922, 1977.

A. Y. Ng. On feature selection: learning with exponentially many irrelevant features as training examples. In *15th International Conference on Machine Learning*, San Francisco, CA, 404-412, 1998.

H. Peng, F. Long, and 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*, 27(8):1226-1238, 2005.

S. Perkins, K. Lacker, and J. Theiler. Grafting: Fast incremental feature selection by gradient descent in function space. *Journal of Machine Learning Research*, 3:1333–1356, 2003.

P. Pudil, J. Novovičová, and J. Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15: 1119-1125, 1994.

J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81-106, 1986.

A. Rakotomamonjy. Variable selection using SVM-based criteria. *Journal of Machine Learning Research*, 3:1357-1370, 2003.

J. Reunanen. Overfitting in making comparisons between variable selection methods. *Journal of Machine Learning Research*, 3:1371-1382, 2003.

I. Rivals and L. Personnaz. MLPs (mono-layer polynomials and multi-layer perceptrons) for nonlinear modeling. *Journal of Machine Learning Research*, 3:1383-1398, 2003.

J. C. Schlimmer. Efficiently inducing determinations: A complete and systematic search algorithm that uses optimal pruning. In *10th International Conference on Machine Learning*, San Mateo, CA, 284-290, 1993.

W. Siedlecki and J. Sklansky. On automatic feature selection. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(2):197-220, 1988.

M. Singh and G. M. Provan. Efficient learning of selective Bayesian network classifiers. In *13th International Conference on Machine Learning*, Bari, Italy, 453-461, 1996.

H. Stoppiglia, G. Dreyfus, R. Dubois, Y. Oussar. Ranking a random feature for variable and feature selection. *Journal of Machine Learning Research*, 3:1399-1414, 2003.

H. Vafaie and K. De Jong. Genetic algorithms as a tool for feature selection in machine learning. In *4th International Conference on Tools with Artificial Intelligence*, IEEE Computer Society, 200-203, 1992.

H. Vafaie and K. De Jong. Robust feature selection algorithms. In *5th International Conference on Tools with Artificial Intelligence*, IEEE Computer Society, 356-363, 1993.

C. J. van Rijsbergen. *Information Retrieval*, Butterworth-Heinemann, 2nd Edition, 1979.

J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of the zero norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439-1461, 2003.

J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In *Neural Information Processing Systems*, Cambridge, MA, 2000.

L. Xu, P. Yan, and T. Chang. Best first strategy for feature selection. In *9th International Conference on Pattern Recognition*, 706-708, 1989.

B. Yu and B. Yuan. A more efficient branch and bound algorithm for feature selection. *Pattern Recognition*, 26(6):883-889, 1993.

J. Zhu and T. Hastie. Classification of gene microarrays by penalized logistic regression. *Biostatistics*, 5(3):427-443, 2004.