A variance reduction framework for stable feature selection

Yue Han
Department of Computer Science
Binghamton University, State University of New York
Binghamton, NY 13902-6000, USA
yhan1@binghamton.edu

Lei Yu
Department of Computer Science
Binghamton University, State University of New York
Binghamton, NY 13902-6000, USA
lyu@cs.binghamton.edu

Abstract

Stability of feature selection is an important but under-addressed issue in knowledge discovery from high-dimensional data. In this study, we present a theoretical framework about the relationship between the stability and the accuracy of feature selection based on a formal bias-variance decomposition of feature selection error. The framework also reveals the connection between stability and sample size and suggests a variance reduction approach for improving the stability of feature selection algorithms under small sample size. Following the theoretical framework, we propose an empirical variance reduction framework, margin based instance weighting, which weights training instances according to their importance to feature evaluation. Our extensive experimental study first verifies the theoretical and empirical frameworks based on synthetic data sets and a popular feature selection algorithm SVM-RFE. Experiments based on real-world microarray data sets further verify that the empirical framework is effective at reducing the variance and improving the subset stability of two representative feature selection algorithms, SVM-RFE and ReliefF, while maintaining comparable predictive accuracy based on the selected features. The proposed instance weighting framework is also shown to be more effective and efficient than the ensemble framework at improving the subset stability of the feature selection algorithms under study.

1 Introduction

Feature selection plays an increasingly important role in knowledge discovery from many application domains with high-dimensional data such as microarray analysis [10], biomedical imaging [21], and text categorization [7]. A large body of feature selection algorithms have been developed with a focus on improving predictive accuracy while reducing dimensionality and model complexity [12, 19]. Besides high accuracy, another important issue is the *stability of feature selection* - the insensitivity of the result of a feature selection algorithm to variations to the training set. This issue is particularly critical for applications where feature selection is used as a knowledge discovery tool for identifying characteristic markers to explain the observed phenomena. For example, in microarray analysis, biologists are interested in finding a small number of features (genes or proteins) that explain the mechanisms driving different behaviors of microarray samples [23]. Biologists instinctively have high confidence in the result of an algorithm that selects similar sets

of genes under some variations to the microarray samples. However, a common problem of existing feature selection algorithms is that an algorithm often selects largely different subsets of features under variations to the training data, although most of these subsets are as good as each other in terms of predictive performance [5, 15, 20]. Such instability dampens the confidence of domain experts in experimentally validating the selected features.

The stability of feature selection is a complicated issue. It can be affected by a number of factors such as data distribution, sample size, and mechanism of feature selection. Moreover, the stability of feature selection should be investigated together with the predictive performance of the selected features. Domain experts will not be interested in a strategy (e.g., arbitrarily selecting the same subset of features regardless of the input instances) that yields very stable feature subsets but bad predictive models. Currently, there exist no theoretical studies on why and how various factors affect the stability of feature selection or the relationship between the stability and predictive performance of feature selection.

In this study, we present a theoretical framework about feature selection stability based on a formal bias-variance decomposition of feature selection error. The theoretical framework explains the relationship between the stability and the accuracy of feature selection, and guides the development of stable feature selection algorithms. It suggests that one does not have to sacrifice predictive accuracy in order to get more stable feature selection results. A better tradeoff between the bias and the variance of feature selection can lead to more stable results while maintaining or even improving predictive accuracy based on the selected features. The framework also reveals the dependency of feature selection stability on the number of instances in a training set (or sample size), and suggests a variance reduction approach for improving the stability of feature selection algorithms under small sample size.

Furthermore, we propose an empirical variance reduction framework - margin based instance weighting. The framework first weights each instance in a training set according to its importance to feature evaluation, and then provides the weighted training set to a feature selection algorithm. The idea of instance weighting is motivated by the theory of importance sampling for variance reduction. Intuitively, instance weighting aims to assign higher weights to instances from regions which contribute more to the aggregate feature weights and assign lower weights to instances from other less important (or outlying) regions. To this end, we introduce a novel concept, margin vector feature space, which enables the the estimation of the importance of instances with respect to (w.r.t.) feature evaluation.

The proposed theoretical and empirical frameworks are validated through an extensive set of experiments. Experiments on synthetic data sets demonstrate the bias-variance decomposition of feature selection error and the effects of sample size on feature selection, using the SVM-RFE algorithm. These experiments also verify the effectiveness of the proposed instance weighting framework at reducing the variance of feature weighting by SVM-RFE, and in turn improving the stability and the predictive accuracy of the selected features by SVM-RFE. Experiments on real-world microarray data sets further verify that the instance weighting framework is effective at reducing the variance of feature weighting and improving the subset stability for two representative feature selection algorithms, SVM-RFE and ReliefF, while maintaining comparable predictive accuracy based on the selected features. Moreover, the instance weighting framework is shown to be more effective and efficient than a recently proposed ensemble framework for stable feature selection.

The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 introduces our theoretical framework on feature selection stability. Section 4 proposes an empirical framework of margin based instance weighting and an efficient algorithm developed under this framework. Section 5 evaluates the theoretical and empirical frameworks based on synthetic and microarray data. Section 6 concludes the paper and outlines future research directions.

2 Related work

Feature selection, the problem of searching for an optimal subset of features guided by some evaluation measures, has been extensively studied in the past (see [12] and [19] for reviews). Various feature selection algorithms can be broadly categorized into filter, wrapper, and embedded algorithms according to whether and how learning algorithms are involved in feature evaluation. Filter algorithms [30] evaluate the quality of features based on the intrinsic data characteristics of the training data, independent of any learning algorithm. Wrapper algorithms [16] rely on the performance of a predefined learning algorithm to evaluate the goodness of feature subsets. Embedded algorithms [14, 24], similar to wrapper algorithms, also involve a learning algorithm for feature evaluation. However, rather than repeatedly applying the learning algorithm on every candidate feature subset, embedded algorithms incorporate feature selection as part of the model training process. These algorithms have been developed with a focus on improving predictive accuracy while reducing dimensionality and model complexity.

There exist a few studies on feature selection stability. Early studies on this topic focused on stability measures and empirical evaluation of the stability of feature selection algorithms [15, 18]. More recently, Loscalzo, et al. proposed a group-based stable feature selection algorithm which exploits the intrinsic correlations among a large number of features to identify consensus feature groups and then selects relevant features based on feature groups [20]. Saey, et al. studied bagging-based ensemble feature selection which aggregates the results from a conventional feature selection algorithm repeatedly applied on a number of bootstrapped samples of the same training set [27]. In contrast to existing studies on stable feature selection, our study provides a theoretical framework which explains the relationship between the stability and the accuracy of feature selection and the dependency of feature selection stability on sample size. In addition, our study proposes an instance weighting framework for improving the stability of feature selection algorithms.

Another line of closely related research is margin based feature selection. Several studies have developed feature selection algorithms under the large margin principles, such as SVM-based feature selection [13, 14] and the Relief family of algorithms [9, 25]. These studies have shown both nice theoretical properties and good generalization performance of margin based feature selection algorithms, but have not yet addressed the stability issue of feature selection. Our study also employs the concept of margins in the proposed margin based instance weighting framework. In contrast with margin based feature selection algorithms (e.g., ReliefF) which directly use margins to weight *features*, our framework exploits the discrepancies among the margins at various instances to weight *instances*. Our framework acts as a preprocessing step to produce a weighted training set which can be input to any feature selection algorithm capable of handling weighted instances.

A problem related to the stability of feature selection is the stability of learning algorithms [2]. It is well known that the generalization error of a learning algorithm can be decomposed into bias, variance, and noise. The variance component quantifies the *instability* of a learning algorithm w.r.t. the predictions of models produced on different training sets. Previous studies on the bias-variance tradeoff [6, 22] explain the relationship between the stability and the accuracy of learning algorithms. Our theoretical framework reveals the relationship between the stability and the accuracy of feature selection algorithms, where stability is measured w.r.t. the results (feature weights or subsets) of a feature selection algorithm from different training sets.

3 Theoretical framework

In Section 3.1, we formally define the stability of feature selection from a sample variance perspective, present a bias-variance decomposition of feature selection error, and discuss the relationship between the stability and the accuracy of feature selection based on this decomposition. In Section 3.2, we further show that for feature selection algorithms which can be viewed as Monte Carlo estimators, their stability depends on the sample size and can be improved by variance reduction techniques such as importance sampling.

3.1 Bias-variance decomposition of feature selection error

Let $D = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$ be a training set of n labeled instances, where $\mathbf{x} \in \Re^d$, defined by d features $X_1, ..., X_d$, and y is the value of the class variable Y. In general, the result of a feature selection algorithm $\mathcal{F}$ on a training set D can be viewed as a vector $\mathbf{r} = (r_1, ..., r_d)$, where r_j $(1 \leq j \leq d)$ is the *estimated* relevance score of feature X_j assigned by $\mathcal{F}$. Let $\mathbf{r}^* = (r_1^*, ..., r_d^*)$ be a vector indicating the *true* relevance score of each feature to the class. In this paper, we focus our discussion on feature weighting algorithms, and adopt the commonly used squared loss function $(r_j^* - r_j)^2$ to measure the error made by $\mathcal{F}$ on feature X_j. When there is no risk of ambiguity, we will drop the subscript $_j$ and use r^* or r to represent the true or estimated relevance score of any feature X, respectively.

For the same feature X, a feature selection algorithm $\mathcal{F}$ in general produces different estimated relevance scores r based on different training sets D. Therefore, we can speak of D as a random variable and use $r(D)$ to represent the estimated relevance score of feature X based on a given training set. $r(D)$ can be viewed as a Monte Carlo estimate of r^* for feature weighting algorithms which decide the relevance score of each feature based on aggregating the scores over all instances in a training set (e.g., the Relief family of algorithms and SVM-based algorithms). To evaluate the overall performance of $\mathcal{F}$, the quantity of interest is the *expected loss (or error)*, $EL(X)$, defined as:

$$EL(X) = \mathbb{E}[(r^* - r(D))^2] = \sum_{D \in \mathscr{D}} (r^* - r(D))^2 p(D) \ , \tag{1}$$

where $\mathscr{D}$ is the set of all possible training sets of size n drawn from the same underlying data distribution, and $p(D)$ is the probability mass function on $\mathscr{D}$.

Let $\mathbb{E}(r(D)) = \sum_{D \in \mathscr{D}} r(D)p(D)$ be the expected value of the estimates for feature X over $\mathscr{D}$. The *bias* of a feature selection algorithm $\mathcal{F}$ on a feature X is defined as:

$$Bias(X) = [r^* - \mathbb{E}(r(D))]^2 \ . \tag{2}$$

The *variance* of a feature selection algorithm $\mathcal{F}$ on a feature X is defined as:

$$Var(X) = \mathbb{E}[r(D) - \mathbb{E}(r(D))]^2 = \sum_{D \in \mathscr{D}} [r(D) - \mathbb{E}(r(D))]^2 p(D) \ . \tag{3}$$

Following the above definitions on the expected loss, bias, and variance, for any feature X, we have the following standard decomposition of the expected loss:

$$EL(X) = Bias(X) + Var(X) \ . \tag{4}$$

Intuitively, the bias reflects the loss incurred by the central tendency of $\mathcal{F}$, while the variance reflects the loss incurred by the fluctuations around the central tendency in response to different training sets. Extending the above definitions to the entire set of features, we can speak of the

average loss, average bias, and average variance, and have the following decomposition among the three:

$$\frac{1}{d}\sum_{j=1}^{d} EL(X_j) = \frac{1}{d}\sum_{j=1}^{d} Bias(X_j) + \frac{1}{d}\sum_{j=1}^{d} Var(X_j) \ . \tag{5}$$

The average variance component naturally quantifies the sensitivity or *instability* of a feature selection algorithm under training data variations; lower average variance means higher stability of the algorithm. We will use the average variance as one of the stability measures in our empirical study. The above bias-variance decomposition is for feature weighting algorithms under squared loss function, and can be extended to feature subset selection algorithms under zero-one loss function in future study.

The above bias-variance decomposition reveals the relationship between the stability (the opposite of variance) and the accuracy (the opposite of error) of feature selection. Reducing either the bias or the variance alone does not necessarily reduce the error, but a better tradeoff between the bias and the variance does. One thing to note at this point is that the error of feature selection in the above decomposition is measured w.r.t. the true relevance of features, not the generalization error of the model learned based on the selected features. The former, in theory, is consistent with the latter; a perfect weighting of the features leads to an optimal feature set and hence an optimal Bayesian classifier [17]. However, in practice, the generalization error depends on both the error of feature selection and the bias-variance properties of the learning algorithm itself. An in-depth study of how the stability of feature selection affects the bias-variance properties of various learning algorithms would be interesting, but is out of the scope of this paper.

The framework presented here also sheds lights on the relationship between the stability of feature selection and the predictive accuracy based on the selected features. Existing studies on stable feature selection [15, 27] showed that different feature selection algorithms performed differently w.r.t. stability and predictive accuracy, and there was no clear winner in terms of both measures. They suggested a tradeoff between the stability and predictive accuracy. To pick the best algorithm for a given data set, a user could use a joint measure which weights the two criteria based on the user's preference on higher accuracy or higher stability. In contrast to the previous studies, our theoretical framework suggests that one does not have to sacrifice predictive accuracy in order to get more stable feature selection results. A better tradeoff between the bias and variance of feature selection can lead to more stable feature selection results, while maintaining or even improving predictive accuracy based on selected features. In the next section, we present a general principle to achieve such a better tradeoff for feature weighting algorithms.

3.2 Variance reduction via importance sampling

Consider a (possibly multidimensional) random variable X having probability density function $f(x)$ on a set of values $\mathscr{X}$. Then the expected value of a function g of X is

$$\mathbb{E}(g(X)) = \int_{x\in\mathscr{X}} g(x)f(x)dx \ .$$

The variance of $g(X)$ is

$$Var(g(X)) = \int_{x\in\mathscr{X}} [g(x) - \mathbb{E}(g(X))]^2 f(x)dx \ .$$

If we were to randomly take an n-sample of X's and compute the mean of $g(x)$ over the sample, then we would have the Monte Carlo *estimate*

$$\widetilde{g_n}(x) = \frac{1}{n}\sum_{i=1}^{n} g(x_i)$$

of $\mathbb{E}(g(X))$. We could, alternatively, speak of the random variable

$$\widetilde{g_n}(X) = \frac{1}{n}\sum_{i=1}^{n} g(X_i)$$

as the Monte Carlo *estimator* of $\mathbb{E}(g(X))$. Note that $\widetilde{g_n}(X)$ is unbiased for $\mathbb{E}(g(X))$:

$$\mathbb{E}(\widetilde{g_n}(X)) = \mathbb{E}(g(X)) \ .$$

The variance of $\widetilde{g_n}(X)$ is:

$$Var(\widetilde{g_n}(X)) = \frac{Var(g(X))}{n} \ .$$

Many feature weighting algorithms such as the Relief family of algorithms and SVM-based algorithms decide the relevance score for each feature based on aggregating the scores over all instances in a training set. We can view such algorithm as a Monte Carlo estimator in the above context. Let $g(X)$ be the weighting function associated with a feature weighting algorithm $\mathcal{F}$, which assigns a relevance score to feature X based on each sampled value of X. $\mathbb{E}(g(X))$ represents the relevance score of X assigned by $\mathcal{F}$ based on the entire distribution. $\widetilde{g_n}(x)$, the Monte Carlo estimate of $\mathbb{E}(g(X))$, represents the estimated relevance score of X assigned by $\mathcal{F}$ based on a training set D of size n. To measure the expected loss, bias, and variance of $\mathcal{F}$, Eqs. (1), (2), and (3) defined in the previous section can be respectively rewritten as follows:

$$EL(X) = \mathbb{E}[(r^* - \widetilde{g_n}(X))^2] \ , \tag{6}$$

$$Bias(X) = [r^* - \mathbb{E}(\widetilde{g_n}(X))]^2 = [r^* - \mathbb{E}(g(X))]^2 \ , \tag{7}$$

$$Var(X) = \mathbb{E}[\widetilde{g_n}(X) - \mathbb{E}(\widetilde{g_n}(X))]^2 = \frac{Var(g(X))}{n} \ . \tag{8}$$

The above formulations show that given a data distribution, $Bias(X)$ depends on the feature selection algorithm, while $Var(X)$ depends on both the feature selection algorithm and the sample size of the training set. Different feature selection algorithms may have different bias and variance properties, and hence different errors. For the same feature selection algorithm, increasing the sample size leads to a reduction of both variance and the expected error of feature selection. In reality, increasing the sample size could be impractical or very costly in many applications. For example, in gene expression microarray data analysis, each instance corresponds to a tissue sample of a patient, which is usually hard to obtain and costly to perform experiments on. Fortunately, there are numerous ways to reduce the variance of a Monte Carlo estimator without increasing the sample size.

Importance sampling is one of the commonly used variance reduction techniques [26]. Intuitively, importance sampling is about choosing a good distribution from which to simulate one's random variables. It can be formally stated by Theorem 1 below (See [26] p. 123 for the proof).

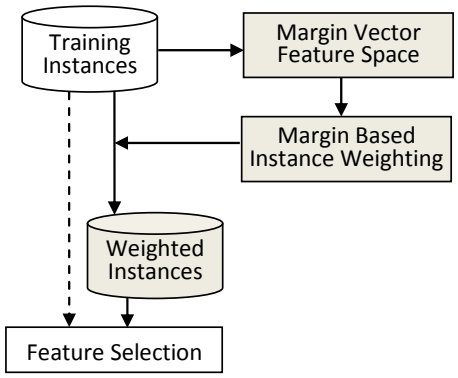


Figure 1: An empirical framework of margin based instance weighting for stable feature selection, consisting of three key components connected by solid arrows

Theorem 1. *Let $h(x)$ be the probability density function for the random variable X which takes values only in $\mathscr{A}$ so that $\int_{x\in\mathscr{A}} h(x)dx = 1$. Then*

$$\int_{x\in\mathscr{A}} g(x)dx = \int_{x\in\mathscr{A}} \frac{g(x)}{h(x)} h(x)dx = \mathbb{E}_h\left(\frac{g(X)}{h(X)}\right) \ ,$$

so long as $h(x) \neq 0$ for any $x \in \mathscr{A}$ for which $g(x) \neq 0$. This gives a Monte Carlo estimator:

$$\widetilde{g_n^h}(X) = \frac{1}{n}\sum_{i=1}^{n} \frac{g(X_i)}{h(X_i)} \quad where \quad X_i \sim h(x).$$

$Var(\widetilde{g_n^h}(X))$ is minimized when $h(x) \propto |g(x)|$.

A good importance sampling function $h(x)$ will be one that is as close as possible to being proportional to $|g(x)|$. In practice, it is rather difficult to find such a function, especially in high-dimensional spaces. Nevertheless, the theory of importance sampling suggests that in order to reduce the variance of a Monte Carlo estimator, instead of performing i.i.d. sampling, we should increase the number of instances taken from regions which contribute more to the quantity of interest and decrease the number of instances taken from other regions. When given only the empirical distribution in a training set, we cannot redo the sampling process. However, we can simulate the effect of importance sampling by estimating the importance of instances w.r.t. feature evaluation and increasing the weights of more important instances and decreasing the weights of others. Therefore, the problem of variance reduction for feature selection boils down to finding an empirical solution of instance weighting.

4 Empirical framework: margin based instance weighting

Motivated by the theoretical framework in Section 3, we propose an empirical framework of instance weighting for variance reduction. As shown in Figure 1, the proposed framework differs from conventional feature selection which directly works on a given training set, consisting of three key components: (1) transforming the original feature space into the margin vector feature space which enables the estimation of the importance of instances w.r.t. feature evaluation; (2) weighting each training instance according to its importance in the margin vector feature space; and (3) applying a given feature selection algorithm on the original feature space with weighted instances. Each of these three key components will be elaborated in turn. Section 4.1 introduces the main ideas and technical details of margin vector feature space. Section 4.2 explains the instance weighting

scheme under the transformed space. Section 4.3 combines components (1) and (2) into an efficient algorithm of margin based instance weighting. In component (3), how to exploit weighted instances in feature selection depends on the specific choice of a feature selection algorithm, and will be discussed later in Section 5.1 where two representative feature selection algorithms are extended to take into account instance weights.

4.1 Margin vector feature space

Margins measure the confidence of a classifier w.r.t. its decisions, and have been used both for theoretical generalization bounds and as guidelines for algorithm design [3]. There are two natural ways of defining the margin of an instance w.r.t. a classifier [4]. *Sample margin* measures the distance between an instance and the decision boundary of a classifier. Support Vector Machine (SVM) [3], for example, uses this type of margin; it finds the separating hyper-plane with the largest sample margin for support vectors. An alternative definition, *hypothesis margin*, measures the distance between the hypothesis of an instance and the closest hypothesis that assigns alternative label to the instance. Hypothesis margin requires a distance measure between hypotheses. For example, AdaBoost [8] uses this type of margin with the L_1-norm as the distance measure between hypotheses. Feature selection algorithms developed under the large margin principles [13, 9] evaluate the relevance of features according to their respective contributions to the margins.

For 1-Nearest Neighbor (1NN) classifier, [4] proved that (i) the hypothesis margin lower bounds the sample margin; and (ii) the hypothesis margin of an instance $\mathbf{x}$ w.r.t. a training set D can be computed by the following formula:

$$\theta_D(\mathbf{x}) = \frac{1}{2}\left(||\mathbf{x} - \mathbf{x}^M|| - ||\mathbf{x} - \mathbf{x}^H||\right) \ ,$$

where $\mathbf{x}^H$ and $\mathbf{x}^M$ represent the nearest instances (called Hit and Miss) to $\mathbf{x}$ in D with the same and opposite class labels, respectively.

Since hypothesis margin is easy to compute and large hypothesis margin ensures large sample margin, we focus on hypothesis margin in this paper. We will speak of hypothesis margin simply as margin in the rest of this paper. In our framework of instance weighting, we employ the concept of margin in a different way. By decomposing the margin of an instance along each dimension, the instance in the original feature space can be represented by a new vector (called *margin vector*) in the *margin vector feature space* defined as follows.

Let $\mathbf{x} = (x_1, ..., x_d)$ be an instance in the original feature space $\Re^d$, and $\mathbf{x}^H$ and $\mathbf{x}^M$ represent the nearest instances to $\mathbf{x}$ with the same and opposite class labels, respectively. For each $\mathbf{x} \in \Re^d$, $\mathbf{x}$ can be mapped to $\mathbf{x}'$ in a new feature space $\Re'^d$ according to:

$$x'_j = |x_j - x_j^M| - |x_j - x_j^H| \ , \tag{9}$$

where x'_j is the jth coordinate of $\mathbf{x}'$ in the new feature space $\Re'^d$, and x_j, x_j^M, or x_j^H is the jth coordinate of $\mathbf{x}$, $\mathbf{x}^H$ or $\mathbf{x}^M$ in $\Re^d$, respectively. Vector $\mathbf{x}'$ is called the margin vector of $\mathbf{x}$, and $\Re'^d$ is called the margin vector feature space. In essence, $\mathbf{x}'$ captures the local profile of feature relevance for all features at $\mathbf{x}$. The larger the value of x'_j, the more feature X_j contributes to the margin of instance $\mathbf{x}$. Thus, the margin vector feature space captures local feature relevance profiles (margin vectors) for all instances in the original feature space.

Figure 2 illustrates the concept of margin vector feature space through a 2-d example. Each instance in the original feature space is projected into the margin vector feature space according to Eq. (9). We can clearly see that the three instances highlighted by circles exhibit largely different

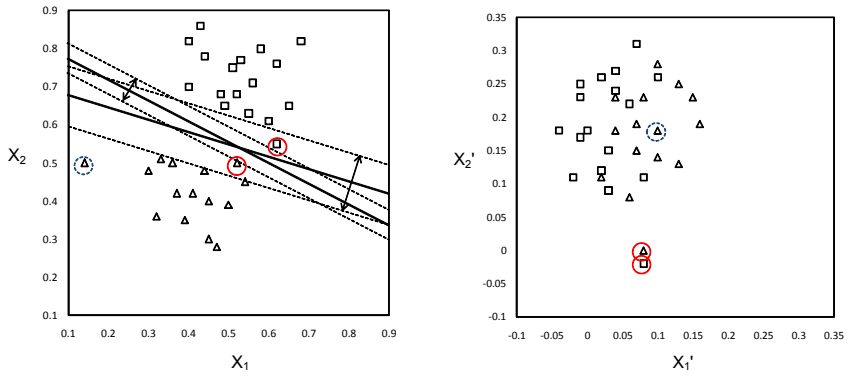


Figure 2: An example for Margin Vector Feature Space. Each data point in the original feature space (left) is projected to the margin vector feature space (right) according to a decomposition of its hypothesis margin along each dimension in the original feature space.

outlying degrees in the two feature spaces. Specifically, the two instances in the solid circles are close to the center of all instances in the original space, but are far apart from the center in the transformed space. The one in the dashed circle shows the opposite trend; it appears as an outlier in the original feature space, but becomes close to the center in the transformed space. Overall, the margin vector feature space captures the distance among instances w.r.t. their margin vectors (instead of feature values in the original space), and enables the detection of instances that largely deviate from others in this respect.

To weight the importance of features X_1 and X_2 in the example, one intuitive idea is to aggregate all margin vectors along each dimension, as adopted by the well-known Relief algorithm [25]. Since the two instances in the solid circles exhibit distinct margin vectors from the rest of the instances, the presence or absence of these instances will contribute to large variance of the aggregate feature weights under training data variations. Another approach, as used by SVM-RFE [13], is to weight features based on the magnitude of feature coefficients in the optimal decision boundary of a linear SVM classifier. As shown in the example, the decision boundaries with or without the two instances in the solid circles show different slopes, resulting in different weights of features X_1 and X_2. Therefore, such instances also cause variance in feature weighting by SVM. By identifying and reducing the emphasis on these outlying instances, more stable results can be produced from a feature selection algorithm. In the next section, we will discuss how to exploit the margin vector feature space to weight instances in order to alleviate the affect of training data variations on feature selection results. In Section 5.1, we will further discuss how to incorporate weighted instances by ReliefF and SVM-RFE.

The above definition and example of margin vector feature space only consider one nearest neighbor from each class. To reduce the affect of noise or outliers in the training set on the transformed feature space, multiple nearest neighbors from each class can be used to compute the margin vector of an instance. In this work, we consider all neighbors from each class for a given instance. Eq. (9) can then be extended to:

$$x'_j = \sum_{l=1}^{m} |x_j - x_j^{M_l}| - \sum_{l=1}^{h} |x_j - x_j^{H_l}| \ , \tag{10}$$

where $x_j^{H_l}$ or $x_j^{M_l}$ denotes the jth component of the lth neighbor to $\mathbf{x}$ with the same or different class labels, respectively. m or h represents the total number of misses or hits ($m + h$ equals the total number of instances in the training set excluding the given instance).

4.2 Margin based instance weighting

Recall the intuitions of importance sampling for variance reduction discussed in Section 3.2. In order to reduce the variance of a Monte Carlo estimator, we should increase the number of instances taken from regions which contribute more to the quantity of interest and decrease the number of instances taken from other regions. When given only an empirical distribution, importance sampling can be simulated by instance weighting. In the context of feature selection, in order to reduce the variance in feature weighting, we should increase the weights of instances from regions which contribute more to the aggregate feature weights and decrease the weights of instances from other less important (or outlying) regions.

The margin vector feature space introduced above enables instance weighting w.r.t. the contribution of each instance in feature weighting. As illustrated in Figure 2, in the margin vector feature space, the central mass region containing most of the instances is clearly more important in deciding the aggregate feature weight for X_1 and X_2 than the outlying region with a couple of outliers. Following the ideas of importance sampling, instances with higher outlying degrees should be assigned lower instance weights in order to reduce the variance of feature weighting under training data variations. To quantitatively evaluate the outlying degree of each instance $\mathbf{x}$ based on its margin vector $\mathbf{x}'$, we measure the average distance of $\mathbf{x}'$ to all other margin vectors. Obviously, instances in sparse regions will have greater average distance, while instances in dense regions will have shorter average distance. Other alternative weighting schemes can be explored, although this simple heuristic works well according to our empirical study. Specifically, the weight for an instance $\mathbf{x}$ is determined by the normalized inverse average distance of its margin vector $\mathbf{x}'$ to all other margin vectors, as in the following formula:

$$w(\mathbf{x}) = \frac{1/\overline{d}(\mathbf{x}')}{\sum_{i=1}^{n} 1/\overline{d}(\mathbf{x}'_i)} \ , \tag{11}$$

where

$$\overline{d}(\mathbf{x}') = \frac{1}{n-1} \sum_{p=1, \mathbf{x}'_p \neq \mathbf{x}'}^{n-1} ||\mathbf{x}' - \mathbf{x}'_p|| \ .$$

4.3 Algorithm

Algorithm 1 combines the processes of margin vector feature space transformation and margin based instance weighting into a single algorithm. Given a training data set, the algorithm first projects all instances in the original feature space to their margin vectors in the margin vector feature space according to Eq. (10). It then weights instances according to Eq. (11) based on the transformed feature space. The algorithm outputs instance weights for all instances, which can be incorporated by a feature selection algorithm.

Both feature space transformation and instance weighting involve distance computation along all features for all pairs of instances: the former in the original feature space, and the latter in the margin vector feature space. Since these computations dominate the time complexity of the algorithm, the overall time complexity of the algorithm is $O(n^2 * d)$, where n is the sample size and d is the number of features in a training set. Therefore, the algorithm is very efficient for high-dimensional data with small sample size (i.e., $n << d$).

Algorithm 1 Margin based instance weighting

Input: training data $D = \{\mathbf{x}_i\}_{i=1}^{n}$
Output: weight vector $\mathbf{w} = (w_1, ..., w_n)$ for all instances in D
// Margin vector feature space transformation
for $i = 1$ **to** n **do**
  for $j = 1$ **to** d **do**
    For $\mathbf{x}_i$, compute $x'_{i,j}$ according to Eq. (10)
  end for
end for
// Margin based instance weighting
Calculate and store pair-wise Euclidean distances among all margin vectors $\mathbf{x}'_i$
for $i = 1$ **to** n **do**
  For $\mathbf{x}_i$, compute its weight $w(\mathbf{x}_i)$ according to Eq. (11)
end for

5 Empirical study

The objective of our empirical study is threefold: (1) to demonstrate the bias-variance decomposition proposed in Section 3; (2) to verify the effectiveness of the proposed instance weighting framework on variance reduction; and (3) to study the impacts of variance reduction on the stability and predictive performance of the selected subsets. Section 5.1 describes the methods in comparison. Section 5.2 introduces the subset stability measure used in our experiments. In Section 5.3, using synthetic data with prior knowledge of the true relevance of features, we demonstrate the bias-variance decomposition based on the widely adopted SVM-RFE algorithm. We further show that the proposed instance weighting framework significantly reduces the variance of feature weights assigned by SVM-RFE, and consequently, improves both the stability and the predictive accuracy of the selected feature subsets. Moreover, we study the effects of sample size on feature selection and the proposed instance weighting framework. In Section 5.4, using real-world microarray data sets, we verify the effectiveness of the instance weighting framework on variance reduction and stability improvement for both SVM-RFE and ReliefF algorithms. Furthermore, we show that the instance weighting framework is more effective and efficient than the ensemble feature selection framework.

5.1 Methods in comparison

We choose SVM-RFE and ReliefF as the baseline algorithms for experimental study. We evaluate the effectiveness of the proposed instance weighting framework for these two algorithms. Furthermore, we compare the instance weighting framework with the ensemble framework using SVM-RFE and ReliefF as the base algorithms.

5.1.1 Baseline algorithms: SVM-RFE and ReliefF

SVM-RFE [13] is chosen as a baseline algorithm because of its wide adoption in high-dimensional data analysis. The main process of SVM-RFE is to recursively eliminate features of lowest ranks, using SVM to rank features. Starting from the full set of features, at each iteration, the algorithm trains a linear SVM classifier based on the remaining set of features, ranks features according to the magnitude of feature coefficients in the optimal hyperplane, and eliminates one or more features with the lowest ranks. This recursive feature elimination (RFE) process stops until all features have

been removed or a desired number of features is reached. Our implementation of SVM-RFE is based on Weka's [29] implementation of soft-margin SVM using linear kernel and default C parameter. As in its debut work, 10 percent of the remaining features are eliminated at each iteration to speed up the RFE process.

ReliefF [25] is chosen as another representative algorithm for margin based feature selection. It is a simple and efficient feature weighting algorithm which considers all features together in evaluating the relevance of features. The main idea of ReliefF is to weight features according to how well their values distinguish between instances that are similar to each other. Specifically, for a two-class problem, the weight for each feature X_j is determined as follows:

$$W(X_j) = \frac{1}{nK} \sum_{i=1}^{n} \sum_{l=1}^{K} (|x_{i,j} - x_{i,j}^{M_l}| - |x_{i,j} - x_{i,j}^{H_l}|) \ , \tag{12}$$

where $x_{i,j}$, $x_{i,j}^{M_l}$, or $x_{i,j}^{H_l}$ denotes the jth component of instance $\mathbf{x}_i$, its lth closest Miss $\mathbf{x}_i^{M_l}$, or its lth closest Hit $\mathbf{x}_i^{H_l}$, respectively. n is the total number of instances, and K is the number of Hits or Misses considered for each instance. We used Weka's implementation of ReliefF with the default setting $K = 10$.

ReliefF appears similar to our proposed instance weighting algorithm since both algorithms involve distance calculation between an instance and its Hits or Misses along each feature for all instances. However, the two algorithms are intrinsically different. ReliefF is a feature weighting algorithm; it produces *feature* weights according to Eq. (12) which does not explicitly construct the margin vector for each instance, but takes an average of the margins over all instances. Our instance weighting algorithm produces *instance* weights by explicitly projecting each instance to its margin vector in the margin vector feature space (based on Eq. (10)) and a successive instance weighting procedure in the margin vector feature space. Our instance weighting algorithm can be used as a preprocessing step for any feature selection algorithms which can be extended to incorporate instance weights.

5.1.2 Instance weighting SVM-RFE and instance weighting ReliefF

Given a training set, SVM-RFE and ReliefF select features based on the original training set where every instance is equally weighted. They can be extended to work on a weighted training set produced by our instance weighting algorithm. We refer to this version of SVM-RFE or ReliefF as instance weighting (**IW**) SVM-RFE or instance weighting (**IW**) ReliefF, respectively. We next explain how instance weights are incorporated into each algorithm.

For SVM-RFE, feature weights are determined based on the final chosen hyperplane of soft-margin SVM which is decided by the trade-off between maximizing the margin and minimizing the training error [3]. With an instance weight $w_i > 0$ assigned to each instance, the original objective function of soft-margin SVM is extended as follows:

$$\min \quad \frac{1}{2}\|\mathrm{w}\|^2 + C \sum_{i=1}^{n} w_i \xi_i \ , \tag{13}$$

where the first component is the opposite of the margin, and ξ_i (value of the slack variable) and C in the second component respectively capture the error of each instance caused by the hyperplane and the error penalty. For instances with $\xi_i > 0$, increased or decreased instance weight influences the error term (and hence the chosen hyperplane) by amplifying or reducing the effect of ξ_i. When

all instances have equal weight, Eq. (13) becomes the original objective function of soft-margin SVM.

To see how the extension in Eq. (13) affects the variance of feature weighting by SVM, let us revisit the example in Figure 2. Without instance weighting, the optimal decision boundary is sensitive to the presence or absence of the two support vectors highlighted by the solid circles (assuming a reasonably large penalty term C). With instance weighting, these two instances will receive much lower weights than the rest of the instances. Given the same penalty term C, the optimal decision boundary will be determined by alternative support vectors which lie much closer to the center of all instances in the margin vector feature space, and hence less sensitive to training data variations.

For ReliefF, feature weights are determined based on the weighting function in Eq. (12). With an instance weight $w_i > 0$ assigned to each instance, the original weighting function is extended as follows:

$$W(X_j) = \sum_{i=1}^{n} w_i \sum_{l=1}^{K} (w_i^{M_l} |x_{i,j} - x_{i,j}^{M_l}| - w_i^{H_l} |x_{i,j} - x_{i,j}^{H_l}|) \tag{14}$$

where w_i, $w_i^{M_l}$, or $w_i^{H_l}$ denotes the weight of instance $\mathbf{x}_i$, its lth closest Miss $\mathbf{x}_i^{M_l}$, or its lth closest Hit $\mathbf{x}_i^{H_l}$, respectively. Intuitively, instances with higher weights will have bigger influence on deciding the feature weights, and vice versa. Eq. (14) becomes the original weighting function Eq. (12) when all instances have equal weight $1/n$, and all Hits and Misses have equal weight $1/K$.

5.1.3 Ensemble SVM-RFE and ensemble ReliefF

Given a training set, the bagging ensemble framework [27] first generates a number of bootstrapped training sets, and then repeatedly applies a base feature selection algorithm (e.g., SVM-RFE or ReliefF) on each of the newly created training sets to generate a number of feature rankings. To aggregate the different rankings into a final consensus ranking, the complete linear aggregation scheme sums the ranks of a feature decided based on all bootstrapped training sets. We refer to the ensemble version of SVM-RFE or ReliefF as ensemble SVM-RFE or ensemble ReliefF, respectively. In our implementation, we use 20 bootstrapped training sets to construct each ensemble.

5.2 Subset stability measure

The variance defined in Section 3 naturally quantifies the instability of a feature selection algorithm w.r.t. feature weights. The stability of a feature selection algorithm can also be measured w.r.t. the selected subsets. Following [15] and [20], we take a similarity based approach where the stability of a feature selection algorithm is measured by the average over all pairwise similarity comparisons among all feature subsets obtained by the same algorithm from different subsamplings of a data set. Let $\{D_i\}_{i=1}^{q}$ be a set of subsamplings of a data set of the same size, and S_i be the subset selected by a feature selection algorithm $\mathcal{F}$ on the subsampling D_i. The stability of $\mathcal{F}$ is given by:

$$\overline{Sim} = \frac{2 \sum_{i=1}^{q-1} \sum_{j=i+1}^{q} Sim(S_i, S_j)}{q(q-1)} , \tag{15}$$

where $Sim(S_i, S_j)$ represents a similarity measure between two subsets.

The stability of $\mathcal{F}$ depends on the specific choice of the similarity measure $Sim(S_i, S_j)$. Simple measures such as the percentage of overlap or Jaccard index can be applied as in [15]. These measures tend to produce higher values for larger subsets due to the increased bias of selecting

overlapping features by chance. To correct this bias, [18] suggested the use of the Kuncheva index, defined as follows:

$$Sim(S_i, S_j) = \frac{|S_i \cap S_j| - (k^2/d)}{k - (k^2/d)} \ , \tag{16}$$

where d denotes the total number of features in a data set, and $k = |S_i| = |S_j|$ denotes the size of the selected subsets. The Kuncheva index takes values in [-1,1], with larger value indicating larger number of common features in both subsets. The k^2/d term corrects a bias due to the chance of selecting common features between two randomly chosen subsets. An index close to zero reflects that the overlap between two subsets is mostly due to chance.

5.3 Experiments on synthetic data

5.3.1 Experimental setup

The data distribution used to generate training and test sets consists of 1000 random variables (features) from a mixture of two multivariate normal distributions: $\mathcal{N}_1(\mu_1, \Sigma)$ and $\mathcal{N}_2(\mu_2, \Sigma)$, with means

$$\mu_1 = (\underbrace{0.5, ..., 0.5}_{50}, \underbrace{0, ..., 0}_{950}) \ ,$$

$$\mu_2 = (\underbrace{-0.5, ..., -0.5}_{50}, \underbrace{0, ..., 0}_{950}) \ ,$$

and covariance

$$\Sigma = \begin{bmatrix} \Sigma_1 & 0 & \cdots & 0 \\ 0 & \Sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Sigma_{100} \end{bmatrix} \ ,$$

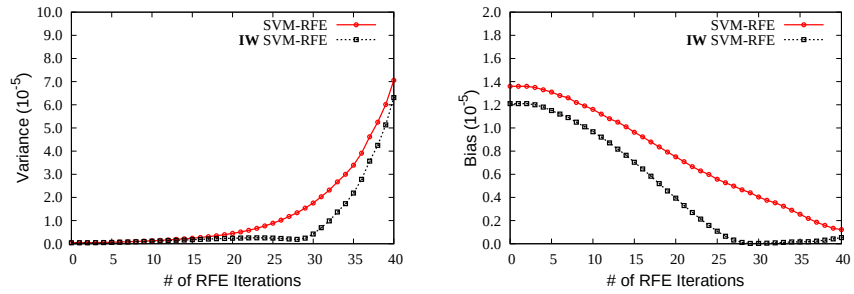
where Σ is a block diagonal matrix, and Σ_i is a 10×10 square matrix with elements 1 along its diagonal and 0.8 off its diagonal. So, there are 100 correlated groups with 10 features per group. The class label of each instance from this distribution is decided by the sign of a linear combination of all feature values according to the optimal weight vector

$$\mathbf{r}^* = (\underbrace{0.02, ..., 0.02}_{50}, \underbrace{0, ..., 0}_{950}) \ .$$

Note that the weights of all features sums to 1. Features in the first 5 groups are equally relevant, and features in the other groups are irrelevant.

To measure the variance, bias, and error of a given feature selection algorithm according to the definitions in Section 3, we simulate $\mathscr{D}$, the distribution of all possible training sets, by 500 training sets randomly drawn from the above data distribution. Each training set consists of 100 instances with 50 from $\mathcal{N}_1$ and 50 from $\mathcal{N}_2$. To measure the predictive performance of the selected features, we also randomly draw a test set of 5000 instances.

For experiments with synthetic data, we focus on the SVM-RFE algorithm. To measure the variance, bias, and error of SVM-RFE, we alternatively view the RFE process as an iterative feature weighting process, and associate a normalized weight vector $\mathbf{r} = (r_1, ..., r_d)$ ($\sum_{j=1}^{d} r_j = 1, r_j \geq 0$) to the full set of d features. At each iteration of the RFE process, the weight of each feature is determined according to $r_j = \frac{|w_j|}{\sum_{j=1}^{d} |w_j|}$, where $w_j = 0$ for the eliminated features, and w_j equals the weight of feature j in the current optimal hyperplane for the remaining features.



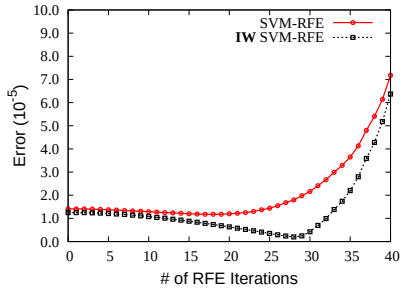
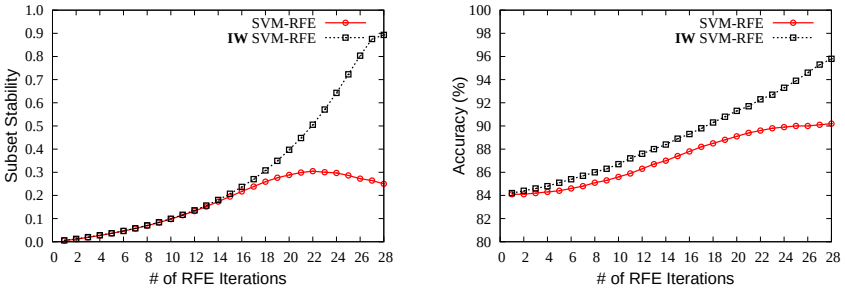


Figure 3: Variance (a), Bias (b), and Error (c) of the feature weights assigned by the conventional and Instance Weighting (IW) versions of SVM-RFE at each iteration of the RFE process for synthetic data

5.3.2 Bias-variance decomposition and variance reduction w.r.t. feature weights

Given the 500 training sets described above, SVM-RFE is applied on each training set, and the resulting normalized weights for all features are recorded at each iteration of the RFE process. The variance, bias, and error over all features (as defined in Eqs. (1)-(5)) are then calculated at each iteration. To verify the effect of instance weighting on variance reduction, the proposed instance weighting algorithm is also applied on each training set to produce its weighted version. SVM-RFE is then repeatedly applied on the 500 weighted training sets in order to measure the variance, bias, and error for instance weighting (**IW**) SVM-RFE.

Figure 3 reports the variance, bias, and error of SVM-RFE and **IW** SVM-RFE across the RFE process (until 10 features remain at the 40th iteration). We can observe the following three major trends. First, for both versions of SVM-RFE, at any iteration, the error is always equal to the sum of the variance and the bias, which is consistent with the bias-variance decomposition of error shown in Eq. (5). Second, for both versions of SVM-RFE, the error is first dominated by the bias during the early iterations when many irrelevant features are assigned non-zero weights, and then becomes dominated by the variance during the later iterations when some relevant features are assigned zero weights. In particular, the error of **IW** SVM-RFE reaches to almost zero at the 28th iteration when the number of remaining features is closest to 50 (the number of truly relevant features). Before or after that point, its error almost solely results from its bias or variance, respectively. Third, **IW** SVM-RFE exhibits significantly lower variance and bias (hence, lower error) than SVM-RFE when the number of remaining features approaches to 50. Note that the magnitude of variance, bias, and error reported in Figure 3 is very small due to the small magnitude of the normalized feature weights.



(a) (b)

Figure 4: Stability (a) and predictive performance (b) of the selected subsets by the conventional and Instance Weighting (IW) versions of SVM-RFE at each iteration of the RFE process for synthetic data

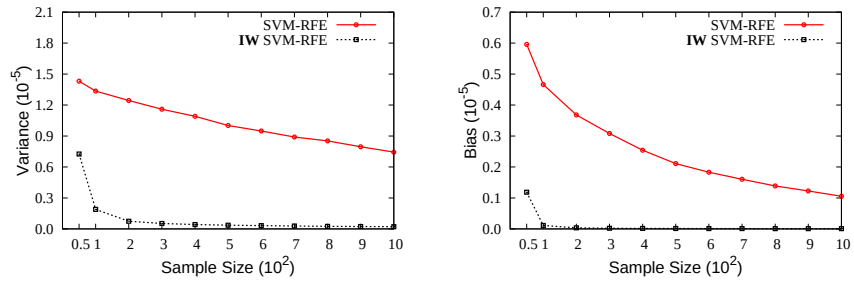
5.3.3 Stability and predictive performance w.r.t. selected subsets

We next study the impacts of variance reduction on the stability and the predictive performance of the selected subsets. Figure 4 (left) compares the subset stability (by Kuncheva index) of SVM-RFE and **IW** SVM-RFE across the RFE process (until about 50 features remain at the 28th iteration). To measure predictive performance, for each training set, a linear SVM classifier is trained based on the selected subset at each RFE iteration and tested on the independent test set. Figure 4 (right) compares the average predictive accuracy (over the 500 training/test trials) of linear SVM at each RFE iteration.

From Figure 4 (left), we can observe that the stability of the subsets selected by **IW** SVM-RFE becomes significantly higher than those selected by SVM-RFE as the number of selected features approaches to the number of truly relevant features at the 28th iteration. Examining the trend of subset stability together with the trend of variance (in Figure 3), we can see that the reduction of variance by instance weighting goes in parallel with the improvement of subset stability, except for the early iterations when irrelevant features are eliminated largely by chance. Note that both versions of SVM-RFE exhibit very low stability during the early iterations, because of the inclusion of the correction term in the Kuncheva index. From Figure 4 (right), we can observe that the subsets selected by **IW** SVM-RFE also result in higher predictive accuracy than those selected by SVM-RFE. The difference is particularly significant during iterations when **IW** SVM-RFE exhibits much higher stability than SVM-RFE. Overall, results from Figures 3 and 4 demonstrate that variance reduction by instance weighting, an approach for a better bias-variance tradeoff, can lead to increased subset stability as well as improved predictive accuracy based on the selected features.

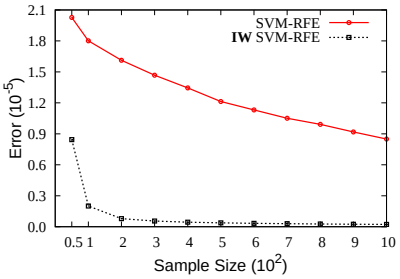
5.3.4 Sample size effects on feature selection and instance weighting

Results reported so far are based on the synthetic data distribution described in Section 5.3.1, with 100 training instances in each training set. We now study the effects of training sample size on SVM-RFE and **IW** SVM-RFE. We first look into how increasing sample size affects the variance, bias, and error of the two versions of SVM-RFE. Instead of examining the three metrics across the entire RFE process, we focus on the iteration when 50 features are selected. As shown in Figure 5, the performance of SVM-RFE clearly depends on the sample size. Both the variance and the bias (hence the error) of SVM-RFE consistently decrease as the sample size increases from 50 to 1000. In contrast, the performance of **IW** SVM-RFE exhibits a much less dependency on the sample size.



(a)

(b)



(c)

Figure 5: Variance (a), Bias (b), and Error (c) of the feature weights assigned by the conventional and Instance Weighting (IW) versions of SVM-RFE when 50 features selected on synthetic data with increasing sample size

Its three metrics quickly converge to near zero when the sample size approaches to 200. Matching trends can be observed from Figure 6 which depicts the stability and the predictive performance of the selected subsets by the two algorithms under increasing sample size. Both subset stability and predictive performance of SVM-RFE consistently improve with the increase of sample size, while the two metrics of **IW** SVM-RFE converge to almost perfect values as soon as the sample size reaches 200.

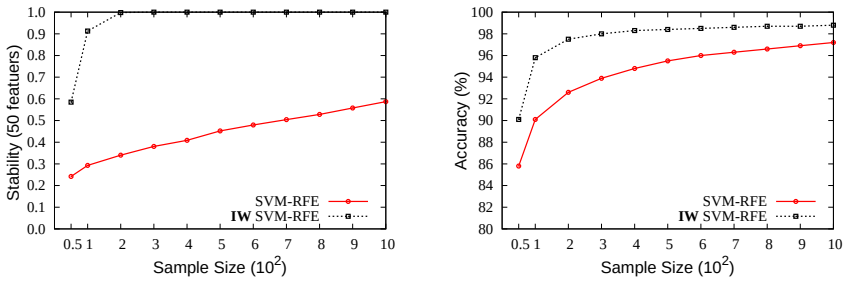
Overall, results from Figures 5 and 6 demonstrate: (i) the sample size dependency of the performance of feature selection algorithms such as SVM-RFE and (ii) the effectiveness of instance weighting on alleviating such dependency. It is worthy of noting that the performance of **IW** SVM-RFE at sample size 200 is significantly better than that of SVM-RFE at sample size 1000. Such observation indicates that the proposed instance weighting framework is an effective alternative to increasing sample size for improving the stability and predictive performance of feature selection algorithms. Recall that in many real-world applications like microarray data analysis, increasing the number of training samples could be impractical or very costly.

5.4 Experiments on real-world data

5.4.1 Experimental setup

We experimented with four frequently studied microarray data sets characterized in Table 1. For the Lung data set, we applied a standard t-test to the original data set and only kept the top 5000 features in order to make the experiments more manageable.

In this section, we verify the effectiveness of the instance weighting framework for both SVM-RFE and ReliefF algorithms. Furthermore, we compare the instance weighting framework with the ensemble feature selection framework. To do so, we evaluate the performance of the original, ensemble, and instance weighting (IW) versions of SVM-RFE and ReliefF on each data set, using the 10×10 folds cross-validation (CV) procedure. Given a data set and a random 10-fold partition



(a) (b)

Figure 6: Stability (a) and predictive performance (b) of the selected subsets by the conventional and Instance Weighting (IW) versions of SVM-RFE when 50 features selected on synthetic data with increasing sample size

Table 1: Summary of microarray data sets

Data Set	# Features	# Instances	Source
Colon	2000	62	[1]
Leukemia	7129	72	[10]
Prostate	6034	102	[28]
Lung	12533	181	[11]

of it, the three versions of SVM-RFE and ReliefF are repeatedly applied to 9 out of the 10 folds to produce feature weights and select subsets of features at various sizes, while a different fold is hold out each time. For each selected subset, both a linear SVM and a KNN (K=1) classifiers are trained based on the selected features and the training set, and then tested on the corresponding hold-out test set. The variance and subset stability of each algorithm is measured in the same way as for synthetic data. The predictive performance of each algorithm is measured based on the CV accuracies of the linear SVM and KNN classifiers. This entire procedure is repeated 10 times, and the average over 10 runs is reported for each performance metric.

5.4.2 Variance reduction w.r.t. feature weights

Since the true relevance of features is usually unknown for real-world data, it is infeasible to measure the bias and error of feature selection and study the effect of instance weighting on them for real-world data. Nevertheless, we can still evaluate the effect of instance weighting on the variance of SVM-RFE following a similar procedure as used for synthetic data. Figure 7 (a)-(d) report the variance of SVM-RFE and **IW** SVM-RFE across the RFE process for each of the four microarray data sets. Since these data sets contain various numbers of features, to make all figures comparable along the horizontal axis, each variance curve is made to show 40 iterations starting from when about 1000 features remain until when about 10 features remain (the same range shown in Figure 3 for synthetic data). As shown from Figure 7, the variance for both versions of SVM-RFE remains almost zero in the early iterations. However, in the later iterations, the variance of SVM-RFE increases sharply as the number of remaining features approaches to 10, while the variance of **IW** SVM-RFE shows a significantly slower rate of increase than SVM-RFE.

The variance curves for SVM-RFE and **IW** SVM-RFE appear to be indistinguishable during the early iterations because the variance values in the beginning are several orders of magnitude smaller than those at the end of the RFE process. To illustrate the significant difference of perfor-

mance between the two algorithms, we provide a zoomed-in view on the variance of SVM-RFE and **IW** SVM-RFE at the first indexed iteration for each data set in sub-graph (e), where an error bar shows the standard deviation over the 10 random runs. Clearly, instance weighting significantly reduces the variance on all data sets (except Leukemia) even in the beginning of the RFE process, in spite of the small magnitude of the variance values (due to the normalization applied to feature weight vectors). For the iterations close to the end of the RFE process, the error bars remain much smaller compared to the gap between the two curves. For clarify of presentation, the error bars are omitted from sub-figures (a)-(d).

We now examine the effectiveness of instance weighting on variance reduction for ReliefF. Note that ReliefF does not involve an RFE procedure used in SVM-RFE. Figure 8 reports the variance of ReliefF and **IW** ReliefF for all four data sets. Except Prostate, instance weighting significantly reduces the variance of ReliefF. Results in this section demonstrate the effect of instance weighting on the variance of two representative algorithms based on real-world data. We next examine how it impacts feature subset stability and predictive performance of these algorithms.

5.4.3 Stability and predictive performance w.r.t. selected subsets

Figure 9 reports the subset stability across different numbers of selected features for the conventional, Ensemble (**En**), and Instance Weighting (**IW**) versions of SVM-RFE on four data sets. Instance weighting significantly improves the stability of SVM-RFE, which is consistent with the variance reduction effect of instance weighting observed above. Moreover, a comparison of the stability of **IW** SVM-RFE and **En** SVM-RFE indicates that instance weighting is more effective than ensemble for improving the stability of SVM-RFE.

Figure 10 reports the subset stability across different numbers of selected features for ReliefF in three versions on four data sets. Comparing Figure 10 with Figure 9, we can observe that the stability of ReliefF is consistently higher than SVM-RFE under the same stability measure. Although ReliefF shows relatively more stable results, instance weighting in general improves its stability for all data sets. It is worth mentioning that for the Lung data set, **IW** ReliefF exhibits almost perfect subset stability. In contrast to instance weighting, the ensemble method exhibits a negative effect on the stability of ReliefF.

Tables 2 and 3 report the predictive accuracy (average value $\pm$ standard deviation) of linear SVM and 1NN based on the selected features by the three versions of SVM-RFE and ReliefF, respectively. For both SVM-RFE and ReliefF, the three versions in general lead to very similar predictive accuracy. Except for a few cases, the differences in the average accuracy values produced by the three versions are insignificant given the standard deviations. The accuracy results in Tables 2 and 3 verify that the increased stability resulted from instance weighting (as shown in Figures 9 and 10) is not at the price of accuracy.

Observations from this section indicate that different feature selection algorithms can lead to similarly good classification results, while their stability performance can largely vary. The difficulty in distinguishing feature selection algorithms in terms of predictive accuracy mainly lies in the small sample size of the test sets in microarray data as opposed to synthetic data used in Section 5.3. Studying the stability of feature selection provides a new perspective to domain experts in choosing a feature selection algorithm and validating the selected features.

5.4.4 Algorithm efficiency

Figure 11 compares the running time of the three versions of SVM-RFE and ReliefF on the entire data set for each microarray data set. **En** SVM-RFE is almost 20 times slower than SVM-RFE,

Table 2: Predictive accuracy of the selected subsets by the conventional, Ensemble (**En**), and Instance Weighting (**IW**) versions of SVM-RFE for four data sets

Data Set	Classifier	Selector	Number of Selected Features				
			10	20	30	40	50
Colon	SVM	SVM-RFE	82.1±3.5	82.1±3.8	81.9±4.9	82.4±3.6	82.1±3.3
		En SVM-RFE	82.1±4.5	83.9±2.8	83.2±4.5	82.5±4.0	83.2±4.0
		IW SVM-RFE	82.8±2.3	86.6±1.3	86.3±3.1	85.6±3.6	84.6±2.6
	1NN	SVM-RFE	76.8±3.8	78.7±3.9	79.5±3.7	81.0±3.0	81.8±3.5
		En SVM-RFE	76.5±4.5	80.3±2.5	79.0±3.1	79.2±3.1	80.2±3.6
		IW SVM-RFE	76.4±4.0	77.6±5.6	77.7±3.3	78.8±2.4	79.7±2.6
Leukemia	SVM	SVM-RFE	95.0±2.0	96.0±1.4	96.7±1.2	96.8±0.7	97.1±0.8
		En SVM-RFE	94.4±1.3	96.0±1.0	96.2±0.9	95.8±1.1	96.8±1.3
		IW SVM-RFE	92.9±1.2	94.7±1.8	96.0±1.5	96.4±1.2	96.5±0.7
	1NN	SVM-RFE	93.6±2.2	95.3±1.2	95.8±1.6	95.7±1.0	96.5±1.8
		En SVM-RFE	93.3±1.9	94.2±2.2	95.1±1.8	95.4±3.0	95.7±2.4
		IW SVM-RFE	92.8±1.9	95.1±1.3	95.3±1.4	94.7±1.4	95.7±1.8
Prostate	SVM	SVM-RFE	91.9±2.3	92.3±2.0	93.0±1.6	92.6±1.6	93.8±0.9
		En SVM-RFE	93.0±2.5	92.9±1.3	93.8±1.9	94.4±1.7	94.1±1.2
		IW SVM-RFE	93.0±1.3	92.0±1.1	91.3±1.6	91.2±1.7	91.2±1.2
	1NN	SVM-RFE	90.3±3.1	90.5±3.9	91.7±2.7	91.7±2.5	92.3±1.2
		En SVM-RFE	89.7±2.7	91.7±2.7	91.6±2.3	92.1±2.2	92.3±1.8
		IW SVM-RFE	91.0±1.7	90.9±1.3	90.5±2.2	90.2±2.4	91.6±2.3
Lung	SVM	SVM-RFE	98.3±0.4	98.8±0.3	99.0±0.3	99.0±0.3	98.9±0.0
		En SVM-RFE	98.8±0.5	98.8±0.2	98.8±0.2	99.0±0.2	98.9±0.0
		IW SVM-RFE	98.5±0.5	98.8±0.2	98.9±0.0	99.1±0.3	99.0±0.2
	1NN	SVM-RFE	98.2±0.4	98.5±0.4	98.4±0.6	98.6±0.4	98.7±0.3
		En SVM-RFE	98.8±0.2	98.5±0.4	98.6±0.5	98.7±0.3	98.5±0.3
		IW SVM-RFE	98.8±0.6	98.5±0.6	98.8±0.2	98.9±0.0	98.9±0.0

while **IW** SVM-RFE is only slightly slower than SVM-RFE. The same trend applies to the three versions of ReliefF. The efficiency of **IW** SVM-RFE and **IW** ReliefF lies in the fact that the instance weighting process acts as a preprocessing step which is executed only once. Such slight extra cost of instance weighting leads to significantly increased stability for the base algorithms.

6 Conclusion

In this paper, we have presented a theoretical framework which reveals the relationship between the stability and accuracy of feature selection and the dependency of stability on sample size. We have also developed an empirical instance weighting framework for variance reduction and a margin based instance weighting algorithm. Our empirical study has verified that instance weighting is an effective and efficient approach to reduce the variance and improve the stability of feature selection algorithms without sacrificing the predictive accuracy based on the selected features.

The specific algorithm developed under the instance weighting framework was meant to demonstrate the effectiveness of the framework, and can be improved in various ways. In the future, we plan to investigate alternative methods for weighting instances according to margin vectors and study the effectiveness of the instance weighting framework for other feature selection algorithms. Along the theoretical framework, an interesting direction would be to investigate how the stability of feature selection affects the bias-variance properties of various learning algorithms.

References

[1] U. Alon, N. Barkai, D. A. Notterman, K. Gishdagger, S. Ybarradagger, D. Mackdagger, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and

Table 3: Predictive accuracy of the selected subsets by the conventional, Ensemble (**En**), and Instance Weighting (**IW**) versions of ReliefF for four data sets

Data Set	Classifier	Selector	Number of Selected Features				
			10	20	30	40	50
Colon	SVM	ReliefF	84.0±2.0	84.8±1.2	84.0±3.1	84.2±2.7	85.2±3.6
		En ReliefF	84.2±2.8	84.5±2.6	85.0±2.5	84.3±2.9	84.5±2.6
		IW ReliefF	83.9±2.6	83.7±2.2	84.4±1.3	84.2±2.5	83.9±4.3
	1NN	ReliefF	75.3±3.5	75.0±2.3	75.8±3.4	75.5±3.4	74.6±4.3
		En ReliefF	77.0±3.0	75.5±2.9	76.1±3.6	76.8±3.1	76.2±2.8
		IW ReliefF	74.7±4.6	76.9±3.4	75.7±2.6	75.8±3.2	76.1±2.6
Leukemia	SVM	ReliefF	92.1±1.5	94.7±1.1	95.1±1.3	96.0±1.2	96.0±0.8
		En ReliefF	92.6±0.9	93.9±1.3	94.4±1.6	95.3±1.5	95.4±1.5
		IW ReliefF	92.5±1.5	93.9±1.2	94.9±0.9	95.4±0.7	95.3±1.5
	1NN	ReliefF	91.9±1.4	91.9±1.7	91.7±2.1	92.4±2.3	91.9±2.6
		En ReliefF	91.8±1.8	91.9±3.0	91.1±1.8	91.7±2.5	91.8±2.1
		IW ReliefF	92.9±1.0	92.1±2.1	91.3±2.5	91.5±2.2	91.0±2.3
Prostate	SVM	ReliefF	92.3±1.1	92.9±1.4	92.1±1.6	92.4±1.8	91.2±1.3
		En ReliefF	92.1±1.4	92.5±2.3	91.7±1.1	92.0±1.1	91.4±1.7
		IW ReliefF	92.5±1.1	91.8±2.2	92.0±1.8	91.5±1.5	91.1±0.9
	1NN	ReliefF	88.3±2.2	87.5±2.5	86.9±2.8	87.3±2.1	87.2±2.6
		En ReliefF	86.4±2.9	86.5±2.3	85.5±3.1	85.7±2.0	86.4±2.0
		IW ReliefF	89.5±1.4	89.3±2.3	88.0±1.5	87.5±1.9	87.1±1.4
Lung	SVM	ReliefF	98.6±0.3	98.8±0.2	98.9±0.0	99.1±0.3	99.0±0.2
		En ReliefF	98.7±0.3	98.9±0.0	98.8±0.2	98.8±0.2	99.0±0.2
		IW ReliefF	98.6±0.5	98.8±0.2	98.9±0.0	98.8±0.2	98.8±0.2
	1NN	ReliefF	98.7±0.5	98.7±0.3	98.8±0.2	98.7±0.3	98.6±0.4
		En ReliefF	98.5±0.7	98.6±0.4	98.7±0.3	98.7±0.3	98.5±0.4
		IW ReliefF	98.8±0.6	98.7±0.4	98.5±0.5	98.6±0.3	98.5±0.5

normal colon tissues probed by oligonucleotide arrays. *Proc. Natl Acad. Sci. USA*, 96:6745–6750, 1999.

[2] O. Bousquet and A. Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

[3] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

[4] K. Crammer, R. Gilad-Bachrach, and A. Navot. Margin analysis of the LVQ algorithm. In *Proceedings of the 17th Conference on Neural Information Processing Systems*, pages 462–469, 2002.

[5] C. A. Davis, F. Gerick, V. Hintermair, C. C. Friedel, K. Fundel, R. Kffner, and R. Zimmer. Reliable gene signatures for microarray classification: assessment of stability and performance. *Bioinformatics*, 22:2356–2363, 2006.

[6] P. Domingos. A unified bias-variance decomposition and its applications. In *Proceedings of the Seventeenth Internationl Conference on Machine Learning*, pages 231–238, 2000.

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

[8] Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Computer Systems and Science*, 55(1):119 – 139, 1997.

[9] R. Gilad-Bachrach, A. Navot, and N. Tishby. Margin based feature selection: theory and algorithms. In *Proceedings of the 21st International Conference on Machine learning*, 2004.

[10] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E.S. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.

[11] G. J. Gordon, R. V. Jensen, L. Hsiaoand, S. R. Gullans, J. E. Blumenstock, S. Ramaswamy, W. G. Richards, D. J. Sugarbaker, and R. Bueno. Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. *Cancer Research*, 62:4963–4967, 2002.

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

[13] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.

[14] T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu. The entire regularization path for the support vector machine. *The Journal of Machine Learning Research*, 5:1391–1415, 2004.

[15] A. Kalousis, J. Prados, and M. Hilario. Stability of feature selection algorithms: a study on high-dimensional spaces. *Knowledge and Information Systems*, 12:95–116, 2007.

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

[17] D. Koller and M. Sahami. Toward optimal feature selection. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 284–292, 1996.

[18] L. Kuncheva. A stability index for feature selection. In *Proceedings of the 25th International Multi-Conference on Artificial Intelligence and Applications*, pages 390–395, 2007.

[19] H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 17(4):491–502, 2005.

[20] S. Loscalzo, L. Yu, and C. Ding. Consensus group based stable feature selection. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-09)*, pages 567–576, 2009.

[21] T. M. Mitchell, R. Hutchinson, R. S. Niculescu, F. Pereira, X. Wang, M. Just, and S. Newman. Learning to decode cognitive states from brain images. *Machine Learning*, 57(1-2):145–175, 2004.

[22] M. A. Munson and R. Caruana. On feature selection, bias-variance, and bagging. In *Proceedings of the 20th European Conference on Machine Learning*, pages 144–159, 2009.

[23] M. S. Pepe, R. Etzioni, Z. Feng, J. D. Potter, M. L. Thompson, M. Thornquist, M. Winget, and Y. Yasui. Phases of biomarker development for early detection of cancer. *J Natl Cancer Inst*, 93:1054–1060, 2001.

[24] 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.

[25] M. Robnik-Sikonja and I. Kononenko. Theoretical and empirical analysis of Relief and ReliefF. *Machine Learning*, 53:23–69, 2003.

[26] B. Y. Rubinstein. *Simulation and the Monte Carlo Method*. John Wiley & Sons, New York, 1981.

[27] Y Saeys, T. Abeel, and Y. V. Peer. Robust feature selection using ensemble feature selection techniques. In *Proceedings of the ECML Confernce*, pages 313–325, 2008.

[28] D. Singh, P. G. Febbo, K. Ross, D. G. Jackson, J. Manola, C. Ladd, P. Tamayo, A. A. Renshaw, A. V. D'Amico, J. P. Richie, E. S. Lander, M. Loda, P. W. Kantoff, T. R. Golub, and W. R. Sellers. Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell.*, 2:203–209, 2002.

[29] I. H. Witten and E. Frank. *Data Mining - Pracitcal Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers, 2005.

[30] L. Yu and H. Liu. Feature selection for high-dimensional data: a fast correlation-based filter solution. In *Proceedings of the twentieth International Conference on Machine Learning (ICML)*, pages 856–863, 2003.

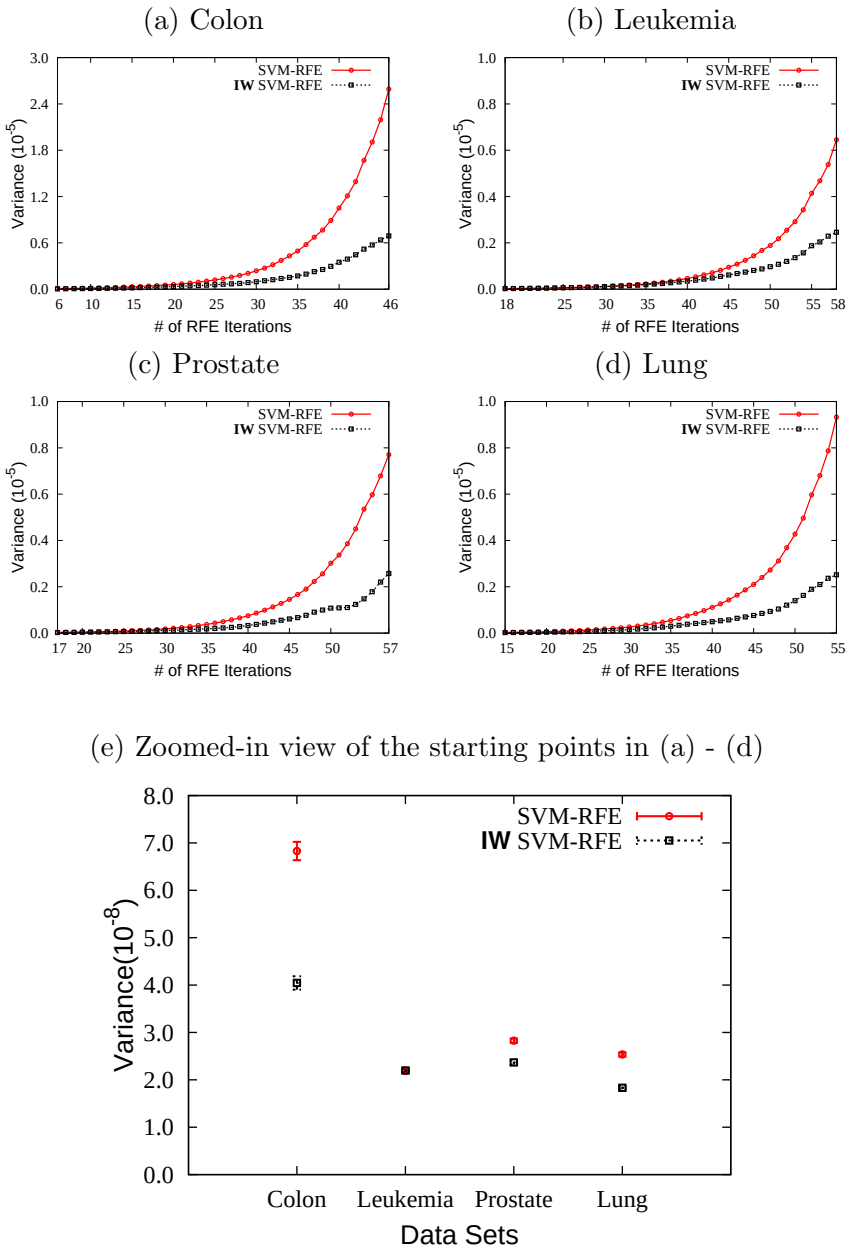


Figure 7: Variance of the feature weights assigned by the conventional and Instance Weighting (**IW**) versions of SVM-RFE at each iteration of the RFE process for four data sets (a)-(d). A zoomed-in view of the starting points in each figure is shown in (e), where an error bar shows the standard deviation over 10 runs.

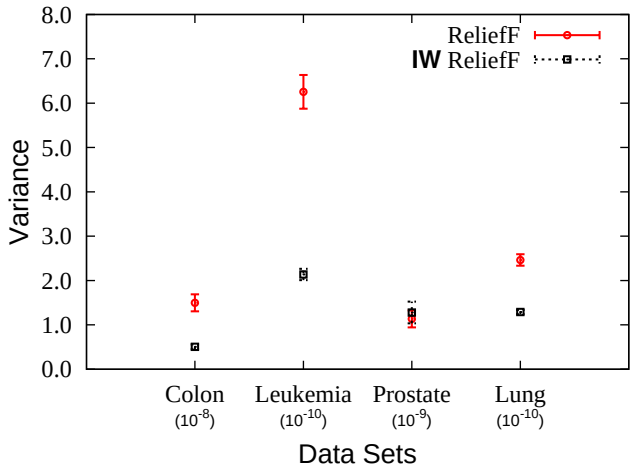


Figure 8: Variance of the feature weights assigned by the conventional and Instance Weighting (**IW**) versions of ReliefF for four data sets. Scales of variance values are shown in parentheses.

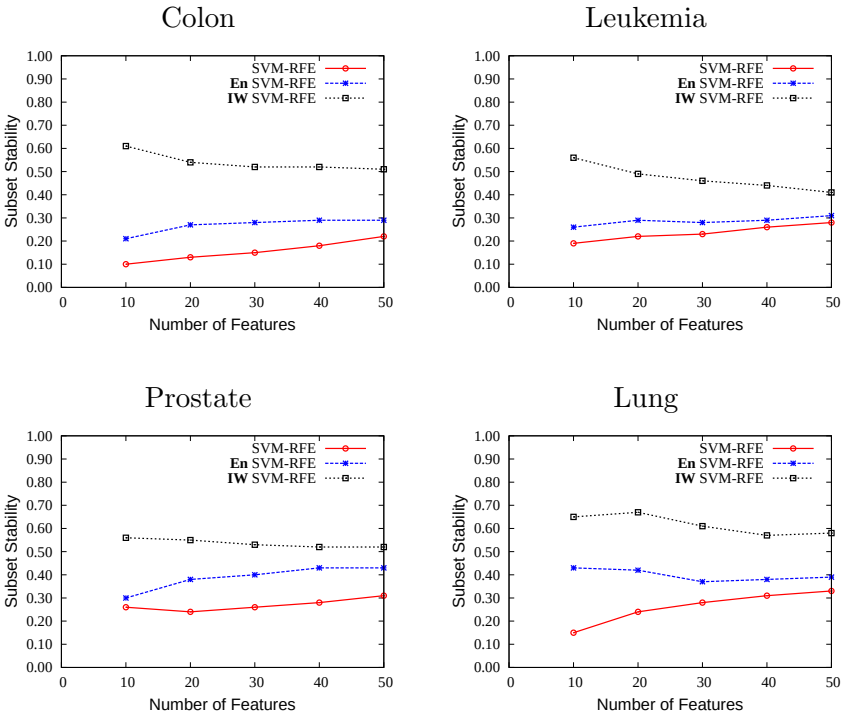


Figure 9: Stability of the selected subsets by the conventional, Ensemble (**En**), and Instance Weighting (**IW**) versions of SVM-RFE for four data sets

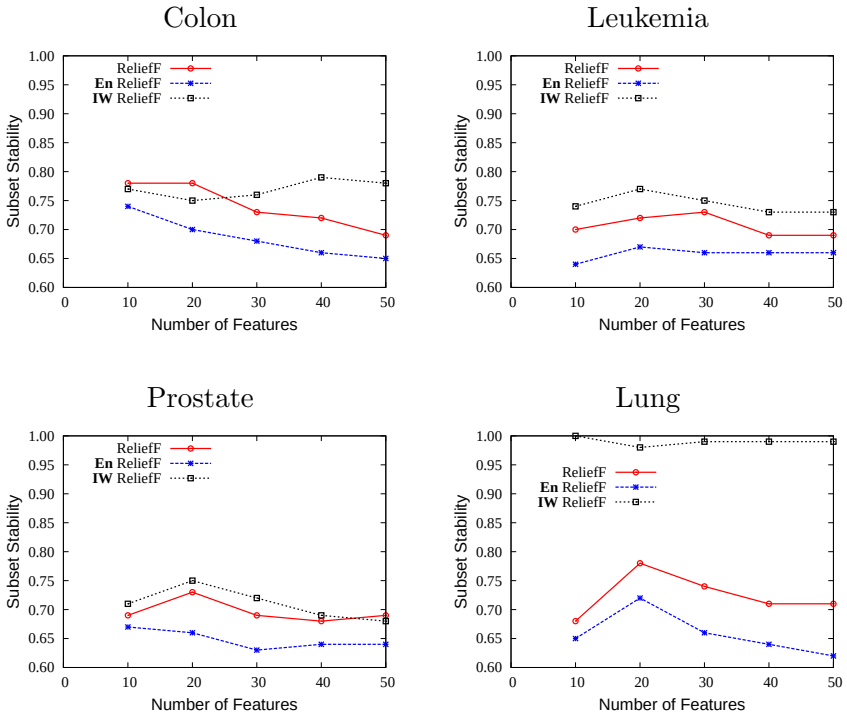


Figure 10: Stability of the selected subsets by the conventional, Ensemble (**En**), and Instance Weighting (**IW**) versions of ReliefF for four data sets

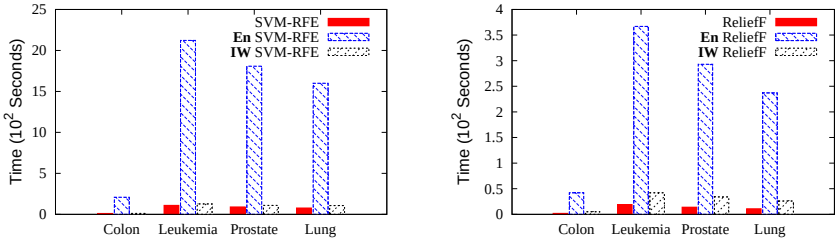


Figure 11: Running time for the conventional, Ensemble (**En**), and Instance Weighting (**IW**) versions of SVM-RFE (a) and ReliefF (b) for four data sets