Data & Knowledge Engineering 81–82 (2012) 67–103

Contents lists available at SciVerse ScienceDirect

Data & Knowledge Engineering

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

DBFS: An effective Density Based Feature Selection scheme for small sample size and high dimensional imbalanced data sets

Mina Alibeigi, Sattar Hashemi *, Ali Hamzeh

CSE and IT Dept., Engineering Campus Number 2, Mollasadra Ave., Shiraz, Iran

ARTICLE INFO

Article history:
Received 31 October 2010
Received in revised form 5 August 2012
Accepted 6 August 2012
Available online 18 August 2012

Keywords:
Feature selection
Imbalanced data set
Probability density function (PDF)

ABSTRACT

Nowadays, imbalanced data sets are pervasive in real world human practices, and hence, become a very interesting research area within machine learning communities. Imbalanced data sets introduce a significant reduction in performance of standard classifiers when they are invoked to learn data underlying concepts. The problem becomes even more sever when imbalanced data sets are involved with high dimensions.

This paper presents a novel feature ranking approach based on the probability density estimation to cope with these issues. The idea behind our approach, named Density Based Feature Selection (DBFS), is that features' distributions over classes can bring significant benefits to feature selection algorithms. In other words, to explore the contribution of each attribute and assign it an appropriate rank, DBFS takes into account features' corresponding distributions over all classes along with their correlations.

To show the effectiveness of the presented approach, well-known feature ranking methods are implemented and compared with our approach across varieties of small sample size and high dimensional data sets from microarray, mass spectrometry and text mining domains. Our theoretical analysis and experimental observations reveal that our approach is the method of choice by offering a simple yet effective feature ranking method based on well-known statistical evaluation measures.

1. Introduction

The class imbalance problem refers to the issue that occurs when one or more classes of a data set have significantly more number of instances than other classes [1]. Nowadays, imbalanced data sets, also known as skew data sets, have received a great deal of attention by researchers due to their importance in many real world human practices such as biological data analysis [2], text classification [3–5], image classification [6] and fraud detection [7,8] among many others.

Despite the prevalence of imbalanced data sets, the performance of many classification algorithms like Naïve Bayes [9], Nearest Neighbor [9], Support Vector Machines [9] and C4.5 [10] degrades significantly when they are applied on these types of data sets [11–13]. This poor performance can be attributed to the fact that almost all classifiers return a simple yet accurate hypothesis to avoid over-fitting the data. These standard algorithms assume or expect balanced class distributions or equal miss classification costs for different classes [1,3,14,66,67]. It is not surprising to see that when presented with complex imbalanced data sets, such hypothesis simply returns the majority class as the result of classification to satisfy the simplicity and accuracy trade off [1,3]. Nonetheless, when dealing with imbalanced data sets, we would prefer that classifiers perform well on the minority class, even at the expense of misclassifying instances of the majority class due to the importance of the minority class [14]. Also, it is worth mentioning that imbalanced data sets usually tend to suffer from class overlapping, lack of representative

* Corresponding author at: P.O. Box: 71348 51154. Tel.: +98 711 613 3544; fax: +98 711 647 4605.
E-mail addresses: alibeigi@cse.shirazu.ac.ir (M. Alibeigi), s_hashemi@shirazu.ac.ir (S. Hashemi), ali@cse.shirazu.ac.ir (A. Hamzeh).

0169-023X/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.datak.2012.08.001

data (rare instances), small disjuncts or presence of noisy and borderline instances that make the learning process of classifiers difficult [13,20,66–70,90].

Generally, existing approaches to imbalanced data classification fall into three categories: sampling, algorithmic methods and feature selection approaches [3,13–16]. The vast majority of studies on imbalanced data sets are devoted to sampling methods where algorithmic methods are less frequently explored, and feature selection approaches have especially become the center of focus in recent years. The following subsections present the most prominent studies in each category.

1.1. Sampling methods for imbalanced learning

The first category i.e. sampling methods modify the imbalanced data set by some mechanisms to rebalance the data distribution in order to reduce the effect of the skewed class imbalance in the learning process [12,18,24,67]. Sampling methods are further divided into three categories: oversampling, undersampling and finally, hybrids that combine both sampling methods. Oversampling approaches aim at converting data sets to balanced ones by the means of instance replication or by creating new synthetic instances of the minority class [17–20], whereas, undersampling approaches do the same by cutting down that instances from the majority class [21–23]. Many sampling methods are proposed in literature. The simplest oversampling (undersampling) methods are random oversampling (random undersampling) methods that as their name shows, randomly select some instances of the minority (majority) class and replicate and add (remove) them to (from) the original data set; while these methods can result in improvements of the classification performance over the original data sets, they suffer from some key issues. Random undersampling methods may eliminate some valuable instances from being considered by the classifier entirely. In contrast, random oversampling methods may cause the classifier to overfit the data by duplicating existing instances [24,25]. Many sampling approaches are introduced to alleviate the shortcomings of random undersampling (oversampling) techniques. One family of these techniques is the synthetic minority oversampling techniques (SMOTE) [18]. Modified SMOTE (MSMOTE) [71], Borderline SMOTE [19] and Adaptive Synthetic Sampling (ADASYN) [62] algorithms are some of the most prominent methods of this family. Another group of sampling methods is the data cleaning techniques. Some representative works in this area include the one-sided sampling method (OSS) [21], the neighborhood cleaning rule (NCL) [72], the condensed nearest neighbor rule and Tomek links (CNN + Tomek links) integration method [73], edited nearest neighbor rule (ENN) [73] and the integration of SMOTE with ENN (SMOTE + ENN) and SMOTE with Tomek links (SMOTE + Tomek) [73]. Also, some hybrid sampling methods are presented such as selective preprocessing of imbalanced data (SPIDER) [74] that combines local oversampling of the minority class with filtering difficult instances from the majority class.

1.2. Algorithmic methods for imbalanced learning

Algorithmic methods as the second category of imbalanced learning methods, are designed to develop a learning approach that is intrinsically insensitive to class skewness in the training data [26]. These approaches create or modify the existing algorithms to consider the importance of the minority instances into consideration [67]. A wide variety of algorithmic strategies have been proposed to combat the class imbalance problem, including one-class learners (novelty detection methods) [27,28], ensemble methods [29–31,63,64] and cost-sensitive approaches [25,32–35,59,65].

One-class learners refer to those methods that recognize instances from a given class and reject instances from all other classes. These methods achieve the goal by merely learning from positive instances with no background information [27,28]. One-class SVMs [35,75,76] and the autoassociator (or autoencoder) methods [77,78] are some of the most prominent learners of this family. One-class learners are not likely the best approach, unless one has only training instances from one class with no other background information [14,28]. The interested reader may find more useful discussions and references to this category of algorithmic methods in [66].

Ensemble methods combine the predictions of a set of classifiers to predict the class label of an instance based on the prediction of individual classifiers. Each classifier is trained using a randomly selected subset of the full set of instances (a.k.a bootstrap). The idea behind ensemble methods is that, in many cases, the performance of ensemble is much better than the performance of any individual classifier in ensemble [33–35] if the individual classifiers perform better than chance. Ensemble methods are divided into boosting and bagging based on the relation between individual classifiers [36]. AdaBoost [31] and Bagging [79] are the most common ensemble learning algorithms among many other different ensemble approaches [67]. In Bagging, an ensemble is created by independently training individual classifiers on bootstrap instances of the training set, and fusing the results of individual classifiers with a combination rule [36]. Conversely, component classifiers in boosting (AdaBoost) are built sequentially and instances that are misclassified by previous components are chosen more often for contributing in the training set of the next classifier [36]. Both standard bagging and boosting methods have high accuracy in general but poor performance on the minority class when applied on imbalanced data sets [30]. Combination of ensemble methods with other techniques to tackle the class imbalance problem has led to several new methods in literature which show improved results. Some of these proposed methods, including but not limited to SMOTEBoost [29], MSMOTEBoost [71], RUSBoost [80], OverBagging [81], UnderOverBagging [81], IIVotes [82] and balance random forest [30] and weighted random forest [30]. For more useful references and more information on these methods, interested readers may refer to this review paper on ensemble methods for class imbalance problems [67].

Another category of algorithmic methods are cost-sensitive approaches. These methods are motivated by the real world applications such as the problem of cancer recognition for which the misclassification costs are not uniform [32]. Cost-sensitive learners are those that try to optimize a loss function associated with a data set that favors the minority class instead of

maximizing the overall accuracy of predictions. The performance of cost-sensitive methods significantly depends on the chosen cost matrix [32]. MetaCost [32] and cost-sensitive boosting methods such as AdaCost [83], CSB1, CSB2 [84] and RareBoost [85] are examples of this category of algorithmic methods among many others.

1.3. Feature selection methods for imbalanced learning

In the recent decade, the class imbalance problem is commonly accompanied by the issue of high dimensionality of the data set and small sample size of the data set [14,24,66]. Some specific examples include but are not limited to gene expression data analysis (microarray and mass spectrometry data), text mining, face recognition and fraud detection [38,66]. Small sample size problem may cause a classifier not to generalize characteristics of the data very well, also, the classifier may overfit the training data and make wrong predictions on unseen (test) data [14]. Traditionally, the small sample size problem has been studied extensively in literature [66,86]. Dimensionality reduction algorithms such as principal component analysis (PCA) and its extensions [87] have a key solution to this problem [66] due to the fact that a good choice of action to increase the generalization potential of a classifier is feature selection [1,3]. Dealing with imbalanced data sets, combination of imbalanced data and the small sample size present new challenges to the community [88,66] which are discussed in [66] to some degree of detail. Some of the different approaches used to tackle the class imbalance problem could make the problems with learning on a small data set even worse [14]. Since, the class imbalance problem is commonly accompanied by the issue of high dimensionality of the data set [14,24], applying feature selection approaches is a necessary step [14]. Ingenious sampling and algorithmic approaches may not be enough to combat the high dimensional class imbalance problem. According to Putten and Someren [39], Forman [3] and Wasikowski and Chen [14] and others [1,3–5,91], in high dimensional imbalanced data sets, feature selection may alone combat the class imbalance problem; however, in his study [37], Elkan found that feature ranking methods are not sufficient to tackle this problem and the interaction between different features must be considered in the selection process of features. He noted that the most prominent weakness of most of the applied feature selection methods is that they did not consider selecting highly correlated features because they were thought to be redundant. Also, Guyon [42] gave a strong theoretical analysis about the limits of feature ranking methods. She stated that those features which are useless (irrelevant) by themselves, can be useful in conjunction with other features [42]. The run time for finding the best feature subset among possible feature subsets is of order $O(2^n)$ when n is the number of features of the data set but this run time is intractable when dealing with high dimensional data sets [9,14]. Also, feature subset selection methods like wrappers and embedded ones that consider the interaction between features in the subset selection phase, may find the feature subset that overfits the training data [14]; however, feature ranking methods do not suffer these issues in dealing with high dimensional data sets [14] and even when feature ranking methods are not optimal, they may be preferred due to their linear run time in the size of features of the data sets [14,42].

Based on these observations, this paper suggests the use of feature ranking in imbalanced or skew data sets to combat the small sample size and high dimensional data sets. Our approach is based on a novel feature ranking approach based on the probability density estimation of features. At the heart of our proposed feature ranking method, named Density Based Feature Selection (DBFS), is a heuristic for evaluating the merit of a feature. The assumption based on the heuristic driven, is that a good feature is the one whose values for each class have minimum overlap with the rest of classes, namely, instances of each class are as apart as possible from instances of other classes according to the feature's values. To explore the contribution of each feature and assign it an appropriate rank, DBFS takes into account features' corresponding distributions over all classes along with their correlations. Experimental results show the effectiveness of the proposed feature ranking method to combat the high dimensional class imbalance problem. The results show that in both well-known biological and text mining domains, DBFS selects the best set of features regardless of the classifier used along with considering each classifier separately.

The rest of this paper is organized as follows. Section 2 discusses the related feature selection methods introduced to overcome the class imbalance problem. Section 3 explains the proposed method. Our experimental results are given in Section 4. The computational complexities of rival feature ranking methods are analyzed in Section 5 and Section 6 concludes the paper by a conclusion part and presents the future work.

2. Related work

For the sake of its numerous benefits to learning algorithms, such as avoiding overfitting, resisting noise and strengthening prediction performance [40], feature selection is a key step for many machine learning algorithms especially for high dimensional data sets such as microarray and mass spectrometry data sets with thousands of features [38] and text mining problems with words exceeding by more than an order of magnitude that of the documents [3].

Feature selection methods can be broadly divided into three categories: filter, wrapper and embedded approaches [14,40]. Filter approaches choose features from the original feature space according to pre-specified evaluation criterions, which are independent of the specified learning algorithms. Conversely, wrapper approaches select features with higher prediction performance according to specified learning algorithms. Thus wrapper approaches can achieve better performance than filter ones. However, wrapper approaches are less common than filter ones because they need higher computational resources and are often intractable for large scale problems [41]. Like wrappers, embedded methods select a subset of features with the best prediction power. In the embedded model, feature selection is integrated into the learning process of an algorithm. This restriction severely limits the number of available embedded methods [42]. One of the typical embedded methods is C4.5 [10]. Due to its computational efficiency, linear run time in the size of the feature set and the independency to any specified learning

algorithm, filter approaches (or feature ranking methods) are more popular and common for high dimensional data sets [30] than wrapper and embedded techniques.

Although feature selection has been studied extensively [40,42–45], its importance in resolving the high dimensional class imbalance problem was recently realized [14]. Mladenic and Grobelnik [4] examined the performance of eleven feature ranking methods on Yahoo Web hierarchies [36]. They examined the classification power of the selected features using the Naïve Bayes classifier and showed that the best results were nearly universally achieved by Odds Ratio and its variants according to evaluation measures including F1 [3–5], F2 [3–5], precision [3–5] and recall [3–5]. They concluded that good feature ranking methods which significantly improve the classification performance are those that favor common features and consider domain and algorithm characteristics. Moreover, Forman [3] examined the performance of twelve feature ranking methods over a number of text mining data sets, focusing on support vector machines and binary class problems with high skewness in data sets. He evaluated the performance of the trained linear SVMs according to multiple evaluation measures using accuracy, F1, precision and recall. The results show that "Bi-Normal Sepration" (BNS) [3] has the best performance. Based on Forman's general finding, the best performing feature ranking methods are those that select features so that separate the minority class from the majority class well.

Zheng et al. [5] investigated that their proposed feature selection framework which explicitly controls the combination of positive features (features indicating membership in a class) and negative features (features indicating non-membership in a class), is more useful than one-sided feature ranking methods that solely select positive features based on their score and two-sided feature ranking methods that implicitly combine positive and negative features. Thus both positive and negative features are important to achieve the best possible classification power.

Jing et al. [89] proposed a general feature selection framework for text categorization. They deduce the distribution characteristics of features contributive to text categorization from the rough set theory. They show that their framework generalizes some of the state-of-the-art feature ranking methods including ECE (Expected Cross Entropy), MI (Mutual Information), IG (Information Gain), CHI (Chi-Square), OR (Odds Ratio) and OCFS (Optimal Orthogonal Centrio Feature Selection) [89]. This framework is useful to analyze the capability of different feature ranking methods and finally select suitable feature ranking methods for specific domains. They proposed a weighted version of this framework which is suitable for imbalanced data sets. Their experiments show that this framework is more effective than CHI, IG and OCFS on both balanced and imbalanced data sets [89].

Chen and Wasikowski [1] proposed a feature ranking method named "Feature Assessment by Sliding Thresholds" (FAST) based on the area under the ROC curve [1,14], named AUC, which is generated by moving the decision boundaries for a single feature classifier to thresholds selected according to an even-bin distribution of feature instances. In an even-bin distribution, feature instances are divided into a number of bins with the same number of instances in each. The thresholds are the mean of instances in each bin. They showed that when the number of bins is equal to 10, the estimated AUC is very close to the exact value of AUC considering all possible thresholds whereas, the FAST algorithm was nearly ten times faster. This method is a non-parametric and two-sided feature ranking method that is directly applicable to continues data sets. Their experimental results showed that FAST outperforms both RELIEF [46] and correlation coefficient [3,42] feature ranking methods on text mining, mass spectrometry and microarray data sets, especially when a small number of features is preferred. Moreover, in another study [14], they proposed another feature ranking method named "Feature Assessment by Information Retrieval" (FAIR) which is similar to FAST except that it uses the area under the precision-recall curve instead of the area under the ROC curve. Results show that the performance of this method is less than other compared methods.

In one point of view, feature ranking methods are either one-sided or two-sided based on whether they select only positive features or a combination of positive and negative features [5]. In another viewpoint, feature ranking methods are either binary or continuous. Binary feature ranking methods can handle only binary or nominal data. For instance, Chi-Square (CHI) [42], Information Gain (IG) [42] and Odds Ratio (OR) [42] belong to the group of binary feature ranking methods. To apply these methods on continuous data sets, a binarization threshold is needed that determines its proper value becomes an important challenge. Therefore, continuous feature ranking methods are designed to handle continuous data without any required preprocessing. Pearson Correlation Coefficient (PCC) [42], Signal to Noise correlation coefficient (S2N) [42] and Feature Assessment by Sliding Thresholds (FAST) [1] are examples of this category. Table 1 gives the formulas of the well-known state of the art binary and continues feature ranking methods which are used in the previous studies as a solution to class imbalance problem [1,3–5,14,42]. CHI, IG, OR, PCC and S2N are standard feature ranking methods that are commonly used in literature for imbalanced data sets as well; however, FAST and FAIR are both feature ranking methods that are specially designed to solve the small sample size and high dimensional class imbalance problem. The following subsections give a brief introduction to each of these feature ranking methods.

2.1. Binary feature ranking methods

2.1.1. CHI

Chi-Square is a statistical test that measures the independence of a feature from the class label. It is a two-sided binary feature ranking method which generalizes well for nominal data but fails on continues data. Forman noted that this test behaves erratically when there are small expected counts of features which are common in text classification problems [3].

2.1.2. IG

Information Gain measures the difference between the entropy of the class label and the conditional entropy of class label when a feature is given. It is also a two-sided binary feature selection which generalizes for nominal data but breaks down on continues data. Like CHI, IG is applicable to multi-class problems.

Table 1
Different feature ranking methods used on imbalanced data sets.

Name	Formula
CHI	$t(tp,(tp+fp)P_{pos})+t(fn,(fn+tn)P_{pos})+t(fp,(tp+fp)P_{neg})+t(tn,(fn+tn)P_{neg})$ where $t(c,e)=(c-e)^2/e$
IG	$e(pos,neg)-\left(\left(\frac{tp+fp}{N}\right)e(tp,fp)+\left(\frac{tn+fn}{N}\right)e(fn,tn)\right)$ where $e(x,y)=-\frac{x}{x+y}\log_2\frac{x}{x+y}-\frac{y}{x+y}\log_2\frac{y}{x+y}$
OR	$\log\left(\frac{tp^*tn}{fp^*fn}\right)$
PCC	$\frac{1}{N}\sum\left(\frac{X-\mu_X}{\sigma_X}\right)\left(\frac{Y-\mu_Y}{\sigma_Y}\right)$
S2N	$\frac{\mu_1-\mu_{-1}}{\sigma_1+\sigma_{-1}}$
FAST	Area under ROC
FAIR	Area under PRC

Notations:
tp: true positive, *fn*: false negative.
fp: false positive, *tn*: true negative.
pos: number of positive cases = *tp* + *fn*.
neg: number of negative cases = *fp* + *tn*.
P_{pos} = *pos*/*all*, P_{neg} = *neg*/*all*.

2.1.3. OR

Odds Ratio computes the probability of a feature occurring in the positive class normalized by the probability of the feature occurring in the negative class. This method is one-sided. These metric is designed to work on binary data sets.

2.2. Continues feature ranking methods

2.2.1. PCC

Pearson Correlation Coefficient measures the linear dependency between a feature and the class label. Correlation coefficients can range from −1 to +1. The absolute value of the coefficients gives the strength of the relationship and the sign of coefficients gives the direction of the relationship. PCC is a one-sided method which can be converted to a two-sided one by squaring the feature scores.

2.2.2. S2N

Signal-to-Noise Correlation Coefficient is a concept in electrical engineering. It measures the ratio of a signal's power to the power of the background noise in the signal. S2N is a similar measurement in machine learning. It compares the ratio of the difference between the class means to the sum of the class standard deviations. If for a given feature, the two class means are distant, there is less probability for an instance being misclassified. The sum of standard deviations scales the distance appropriately. It is a one-sided feature ranking method.

2.2.3. FAST

Feature Assessment by Sliding Thresholds is the method proposed by Chen and Wasikowski [1] based on the area under the ROC curve generated by moving the decision boundaries of a single feature classifier with thresholds given based on an even-bin distribution.

2.2.4. FAIR

Feature Assessment by Information Retrieval is the same as the FAST method except that it uses the area under the precision-recall curve (PRC) to evaluate each single feature classifier.

3. DBFS: Density Based Feature Selection

This section gives a more elaborate view into how our proposed method, called Density Based Feature Selection (DBFS), works. To begin with, the overall picture of DBFS algorithm is outlined in Table 2.

At the heart of our proposed feature ranking method is a heuristic for evaluating the merit of a feature. The assumption based on the heuristic driven, is that a good feature is the one whose values for each class have minimum overlap with the rest of classes, namely, instances of each class are as apart as possible from instances of other classes according to the feature's values. In other words, the instances of each class do not spread into the instances from other classes.

To explore the contribution of each feature and to assign it an appropriate rank, DBFS takes into account features' corresponding distributions over all classes along with their correlations. Because features' distributions over classes bring significant benefit to our feature selection algorithm, the first step of the proposed method is to estimate the probability density function (PDF) of each feature in each class separately. In the following, the methods for estimating PDF are introduced and then the steps of the DBFS method are described in detail.

Table 2
The proposed Density Based Feature Selection method (DBFS).

Input: D={d_1, d_2, ..., d_N} // A data set containing N labeled instances
Input: F={f_1, f_2, ..., f_m} // A data set containing m features
Input: CL={cl_1,cl_2,...,cl_N} // A set containing all class labels
Output: $F^{(ranked)}$ // List of ranked features (desired features are receiving lower ranks)

for $f = 1$ **to** $num\_features$ **do** // num_features is the original number of features of a data set except the class label

Step 1. Estimate the probability density function of feature f in each class as $PDF(cl_i)$, $1 \le i \le num\_classes$

Step 2. for $cl = 1$ **to** $num\_classes$ **do** // num_classes is the number of classes in a data set
Step 2.1. Compute the overlapping area of feature f in class cl which is the overlap between PDF of class cl with PDF of other classes, according to following formula:

$$Overlapping(f, cl) = \int Min\left(PDF(cl), Max\left(PDF(cl_j)\right)\right)$$

$$\text{where } 1 \le j \le num\_classes \text{ and } j \ne cl$$

Step 2.2. Compute the nonoverlapping area of feature f in class cl via the following formula which is a good indication for the discriminant ability:

$$DiscriminantAbility(f, cl) = \left(1 - Overlapping(f, cl)\right)$$

// Because the area under PDF curve over all instance space is always equal to one
end for

Step 3. Enumerate the number of changes as $numChanges$. Number of changes refers to the number of times that instances' labels toggle from one class to another class along the PDF of a particular feature. For a given PDF, instance's label is simply the class having maximum probability (PDF value) in that point.

Step 4. Determine the score of feature f based on the following formula:

$$\text{Score}(f) = \frac{\frac{1}{num\_classes}\sum_{cl=1}^{num\_classes} DiscriminantAbility(f, cl)}{numChanges}$$

end for

Sort features according to their scores in descending order

3.1. Probability density estimation

The popular methods for estimating PDF can be categorized into parametric and nonparametric approaches [47]. The parametric methods assume that data follow a known distribution like Gaussian and hence density estimation problem is merely to determine appropriate values for mean and variance of the distribution. In contrast, nonparametric methods have no prior assumption about the shape of the density function; rather they compute the density directly from the instances. It is worth noting that in most pattern recognition applications there is no prescribed formal structure for estimating the density of the underlying data. Hence, the common parametric forms rarely fit the densities actually encountered in practice. In particular, all of the classical parametric densities are unimodal (have a single local maximum), whereas many practical problems involve multimodal densities [9]. Conversely, nonparametric procedures can be used with arbitrary distributions without the assumption that the forms of the underlying densities are known [9]. This is why nonparametric procedures are of more interest [9] and employed by our approach. The general form of a nonparametric estimation of PDF is according to the following equation:

$$p(x) \cong {}^{k}\!/_{N^{*}V} \tag{1}$$

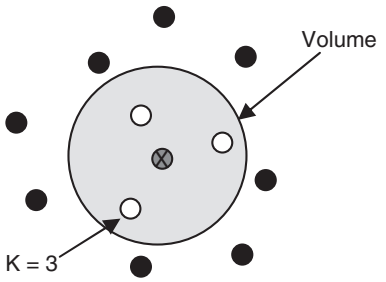

Fig. 1. Non-parametric estimation of probability density function for instance *x*.

where, $p(x)$ is the value of the estimated PDF for instance x, V is the volume surrounding x, N is the total number of instances and k is the number of instances inside V. These concepts are visually depicted in Fig. 1. The PDF estimation becomes more accurate as N increases and volume V shrinks [48]. Since, in practice the total number of instances are fixed (N), to improve the accuracy of the estimated PDF for instance x ($p(x)$), we could let volume V approach zero but then it would be so small that it would enclose no instances. This means that, in practice (with a fix number of instances), by finding a compromise value for the volume V, with even a small number of instances, an admissible probability density would be estimated [48].

Two basic approaches can be adapted to practical nonparametric density estimation methods based on the status of k and V. Fixing the value of k and determining the corresponding volume V from the data, lead to the methods commonly referred to as *K Nearest Neighbor* (*KNN*) methods. On the other hand, when the value of the volume V is chosen to be fixed and k is determined from the data, the nonparametric estimation method is called *Kernel Density Estimation* (*KDE*). It can be shown that both *KNN* and *KDE* density estimators converge to the true probability density in the limit $N \to \infty$ provided that V shrinks suitably with N and k grows with N [48]. Generally, the obtained estimation with the *KNN* approaches is not very satisfactory because of some drawbacks. They are prone to the local noise with very heavy tails. Moreover, the resulting density is not a true probability density since its integral over all the instance space diverges [9]. Furthermore, the estimated probability is not continuous. In contrast, *KDE* methods do not have these shortcomings. Many kernel functions are proposed to be used in *KDE* techniques which lead to different estimation methods. One of the simplest and basic *KDE* approaches is parzen window [48,49]. In the parzen method, the fix volume of V is a unit hypercube centered at the origin x. The kernel function assigns 1 to instances of which their differences with the origin instance x are less than or equal to 0.5 in all dimensions and assigns zero, otherwise. The parzen window has several drawbacks. This method yields density estimations that have discontinuities. Also, it weights equally the entire instances in the volume V surrounding instance x, regardless of their distance to the point x. However, more distant instances should contribute less. It is easy to overcome some of these difficulties by generalizing the parzen window with a smooth kernel function such as Gaussian function. Inspired by these observations, in this study, we estimate PDF through the *KDE* method with Gaussian kernel. It is worth mentioning that our proposed algorithm is not dependent on the use of any particular estimation method. However, using more accurate estimation methods causes the algorithm to perform more efficiently. Fig. 2 illustrates the estimated PDFs of the 2322nd feature of CNS2 data set [50] used in our experiments. The dashed line shows the majority class distribution while the solid line demonstrates that of the minority class.

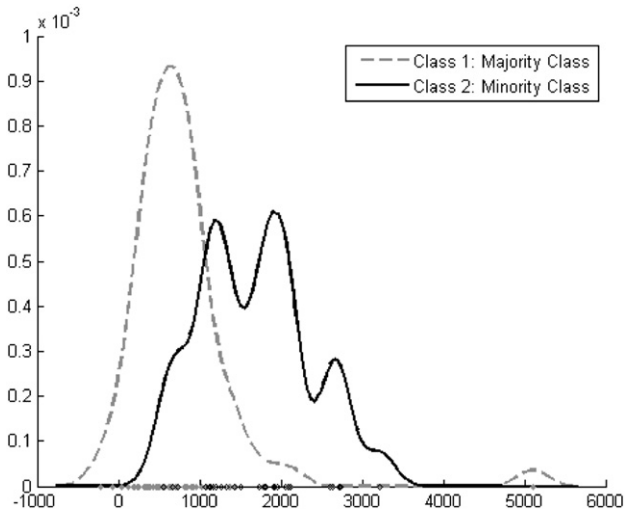

Fig. 2. The PDF of 2322nd feature of CNS2 data set in both majority and minority classes.

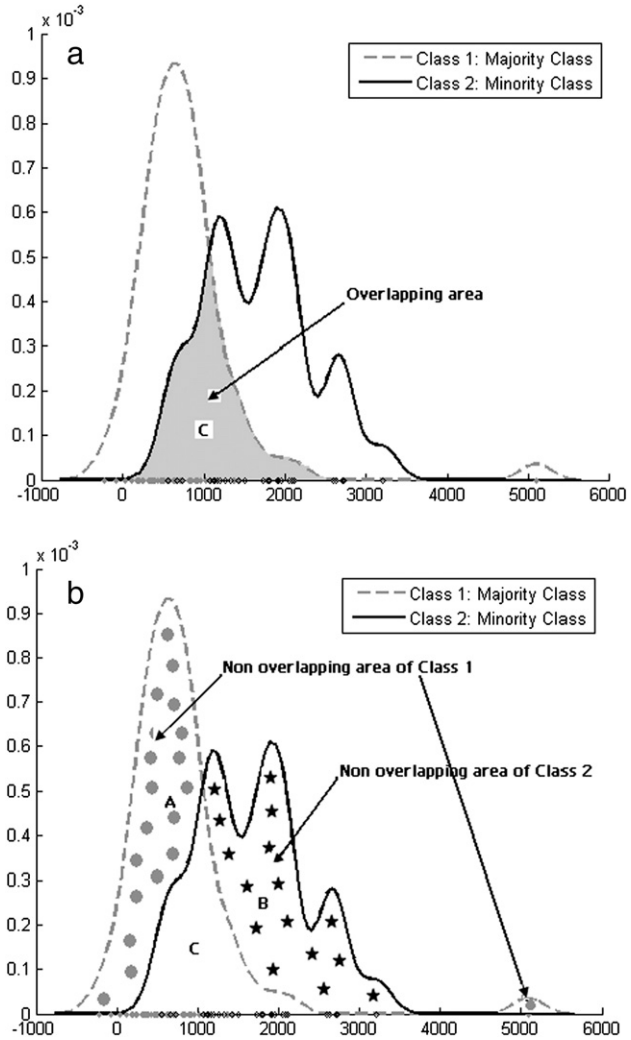

Fig. 3. (a) Overlapping area of 2322nd feature of CNS2 data set, (b) non-overlapping areas of the same feature in each class.

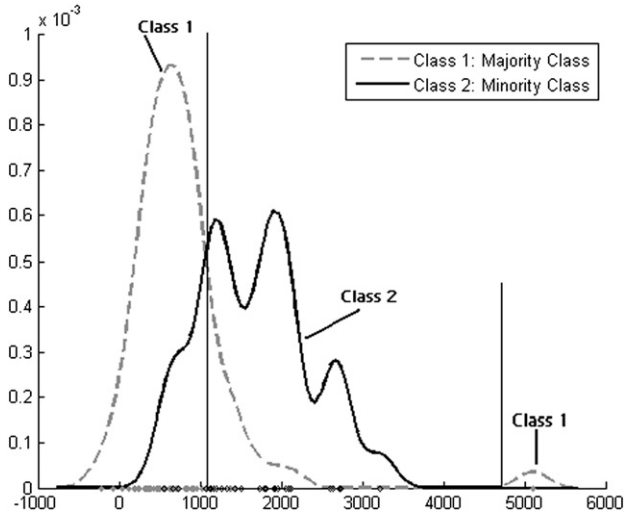

Fig. 4. Each region is tagged by the label of the maximum PDF. For this feature *numChanges* is equal to three.

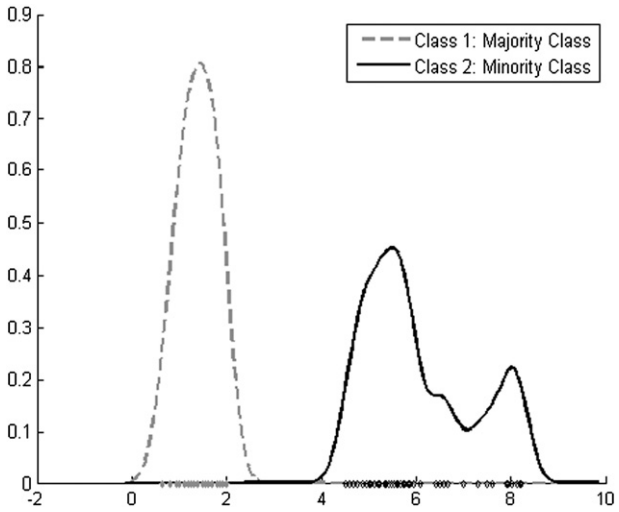

Fig. 5. PDF of a sample best feature. This feature has the *DiscriminantAbility* equal to one and *numChanges* equal to two. This feature is given the highest score via DBFS, equal to 0.5 and is ranked first.

3.2. Feature ranking procedure

As was mentioned earlier, effective methods on imbalanced data sets are those that take the importance of the minority class into consideration because when dealing with imbalanced data sets, classifying the instances of the minority class is highly preferred even at the expense of misclassifying instances of the majority class. DBFS addresses this issue by estimating the PDF in each class. It is known that the area under the PDF curve over the whole instance space is equal to one [47]. Since the areas under the estimated PDF curves for both classes are equal to one and the minority class has fewer instances than the majority class, instances of the minority class are implicitly given higher importance compared to those of the majority class.

The second step after estimating the PDF in each class is to discover the worth of the feature based on its estimated PDFs over classes. As was stated before, a good feature is the one whose values for each class have minimum overlap with the remaining classes. It means finding a feature where, considering its given values, instances of each class are as apart as possible from instances of other classes. In order to estimate the amount of overlap value between instances of two particular classes for a specific feature, we use PDF estimations for each feature and every class label. As Fig. 3(a) illustrates, instances that belong to both classes are in the region marked as C. The estimated area of region C can be considered as the probability that an instance whose feature values lie in region C belongs to both classes. Once the overlapping area for a feature has increased, its importance for

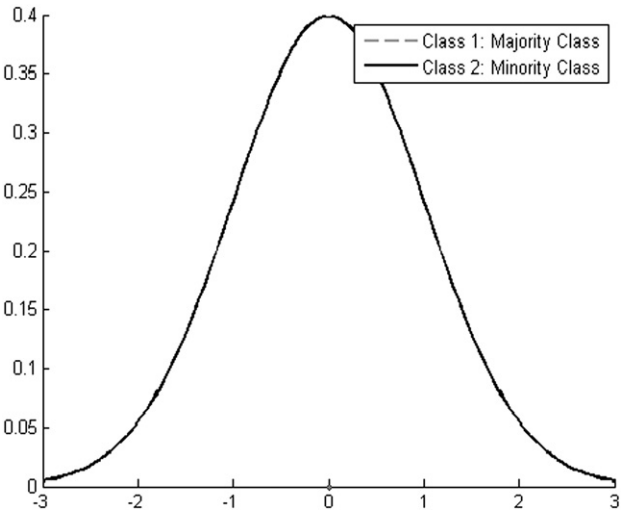

Fig. 6. PDF of a sample worst feature. PDF of both classes are the same. This feature has the *DiscriminantAbility* equal to zero and *numChanges* equal to one. This feature is given the lowest score via DBFS, equal to 0 which gives it the highest possible rank.

Table 3
Characteristics of biological data sets used in experiments.

Name	Samples	Samples per each class (imbalance ratio)	Features	Description
CNS1	40	30, 10 (3:1)	7129	Central nervous system embryonal tumor data set [50].
CNS2	90	60, 30 (2:1)	7129	
Leukemia	73	48, 25 (2:1)	7129	Leukemia molecular data set [51].
Lymphoma_1	77	45,32 (1.4:1)	7129	Diffuse large B-cell lymphoma data set [52].
Lymphoma_2	77	51, 26 (2:1)	7129	
Prostate	89	63, 26 (2.4:1)	15,154	Prostate cancer data set [55].
Lung	180	150, 30 (5:1)	12,533	Lung cancer data set [56].
Ovarian_1	116	100, 16 (6:1)	15,154	Ovarian cancer data set [53].
Ovarian_2	116	100, 16 (6:1)	15,154	

predicting the class label would decrease since considering it as the decision attribute leads to misclassification of a large mass of instances. Overlapping value for a feature, f, in class cl is calculated according to the following formula:

$$Overlapping\ (f, cl) = \int Min\left(PDF(cl), Max\left(PDF\left(cl_j\right)\right)\right) \quad \text{where } 1 \le j \le num_{classes} \text{ and } j \ne cl \tag{2}$$

where $PDF(cl)$ is the estimated PDF for feature f in class cl and $num\_classes$ is the number of classes existing in the data set.

For a given instance, the instance label is simply the class label having maximum probability (PDF value) for that instance. Taking this point into account, the non-overlapping area for feature f in each class which is a good indication for discriminant ability of that feature, is defined as follows:

$$DiscriminantAbility(f, cl) = (1 - Overlapping(f, cl)) \tag{3}$$

The value of *DiscriminantAbility* for each feature in each class conveys the meaning as to how reliable the feature is to safely classify instances of that class. Fig. 3(b) shows the non-overlapping regions for the 2322nd feature of the CNS2 data set [50] in each class. It can be seen that the areas with labels A and B are the non-overlapping areas for the majority and minority class, respectively.

The overall *DiscriminantAbility* of a feature is the mean of its *DiscriminantAbility* values in each class. It can be said that a feature with larger non-overlapping areas or with higher mean of *DiscriminantAbilitiy* values, is able to classify instances more accurately. Furthermore, as was mentioned before, a feature is assumed as a good one if according to the corresponding values of a feature, instances belonging to one class do not spread into the other classes. To take this point into consideration, we incorporate the *numChanges* as a regularization term into our method. The *numChanges* is the proportion of the number of times that the label of regions has changed to the total number of possible of changes along the feature space. To understand this term more precisely, Fig. 4 shows the 2322nd feature of the CNS2 data set [50] which has *numChanges* equal to three. The score of a feature is calculated with respect to the value of its *DiscriminantAbility* and the value of *numChanges* via Eq. (4). The higher the score of a feature, the lower its rank is.

$$\text{Score}(f) = \frac{\frac{1}{num\_classes}\sum_{cl=1}^{num\_classes} DiscriminantAbility(f, cl)}{numChanges} \tag{4}$$

Note that the proposed method is a continuous two-sided feature ranking method. The scores generated by this method may range from 0 to 0.5. The best feature as shown in Fig. 5 is the feature which completely separates instances of different classes (with the *DiscriminantAbility* equal to one) and has only two numbers of changes. This feature is given the highest score, equal to 0.5 which yields it the lowest rank. In contrast, the worst feature as shown in Fig. 6 is the feature for which instances of different classes all have

Table 4
Characteristics of text data sets used in experiments.

Name	Samples	Samples per each class (imbalance ratio)	Features	Description
NIPS_1	391	301, 90 (3.3:1)	9344	NIPS conference papers data set [57].
NIPS_2	396	301, 95 (3.2:1)	9344	
NIPS_3	445	301, 144 (2.1:1)	9344	
NIPS_4	445	301, 144 (2.1:1)	9344	
NIPS_5	441	301, 140 (2.2:1)	9344	
NIPS_6	453	301, 152 (2:1)	9344	
NIPS_7	453	301, 152 (2:1)	9344	
NIPS_8	452	301, 151 (2:1)	9344	
NIPS_9	376	301, 75 (4:1)	9344	
NIPS_10	356	301, 50 (6:1)	9344	
NIPS_11	338	301, 37 (8:1)	9344	
NIPS_12	331	301, 30 (10:1)	9344	
NIPS_13	321	301, 20 (15:1)	9344	

Table 5
Characteristics of two well-known UCI data sets used in experiments.

Name	Samples	Samples per each class	Features	Description
IONOSPHERE	351	225, 126	34	Ionosphere data set from UCI machine learning repository [54].
SONAR	208	111, 97	60	Sonar data set from UCI machine learning repository [54].

the same value (with the value of *DiscriminantAbility* equal to zero). This feature is given the lowest score, equal to 0, while yielding the highest rank. If a feature is highly reliable to predict whether an instance belongs to a class or not, it will have a score close to 0.5. Thus, this method gives a chance to both positive and negative predictor features to be selected for classification.

4. Experimental framework

The procedure of obtaining results is described in this section. First, we discuss the data sets, learning methods and performance measures used in our experiments. Moreover, we analyze the obtained results and simultaneously five important descriptive performance measures are introduced and measured.

4.1. Data sets

Most of the researches on feature selection methods as an imbalanced learning method have focused on text classification [3–5,14]. In addition, there are many other applications which would be advantageous to investigate using feature selection methods. However, to show the effectiveness of DBFS to tackle the class imbalance problem, we have chosen different data sets from variant well-known domains of microarray, mass spectrometry and text mining used in previous studies for fair comparisons. Also, we apply our method to two data sets from the UCI machine learning repository [54]. These data sets are grouped into three different sets: biological, text mining and UCI data sets. The biological set consists of nine microarray and mass spectrometry data sets. For text data sets, we discarded rare features (words) that were presented in less than 10 instances (documents) which left us with 9344 features (words). Biological and text mining data sets all have small sample sizes and high dimensional imbalanced data sets. All of these data sets are publicly available on the corresponding author's website. Tables 3–5 show a summary of the characteristics of the data sets used in this paper to assess the performance of the proposed method.

4.2. Learning methods

In order to assess the performance of different feature ranking methods, the performance of every applying classifier trained on the features selected by each ranking method on a particular data set is compared to the classifier's performance when trained with all features of that data set as the baseline performance. Since there are various classifiers which are common in the literature with different biases, we need to evaluate the feature ranking methods on different classifiers with different biases to truly compare the methods. The classifiers employed in this research are the same as those used in the previous studies [1,3,5,14] i.e. linear SVM (LSVM), nearest neighbor (1-NN) and Naïve Bayes (NB). NB is a simple probabilistic classifier based on Bayes' theorem with the assumption that all features are class-conditionally independent [9]. 1-NN is a lazy and instance-based learning algorithm which classifies each test instance based on its closest training instance [9]. In contrast, LSVM computes a maximum separating hyper plane with linear kernel for classification task [9]. Linear SVM is a strong and stable algorithm and these qualities make it moderately resistant to feature selection. Conversely, feature selection has a stronger influence on the 1-NN and NB classifiers which are weaker algorithms in general and their performances can vary greatly with a small change in their training sets.

It is worth mentioning that since the class imbalance problem is commonly accompanied by the issue of high dimensionality of the data set [14,24] applying feature selection approaches is a necessary step [14]. Ingenious sampling and algorithmic approaches may not be enough to combat the small sample size and high dimensional class imbalance problem. According to Putten and Someren [39], Forman [3] and Wasikowski and Chen [14] and others [1,3–5] in high dimensional imbalanced data sets, feature selection accompanied by standard classifiers may alone combat the class imbalance problem. Taking these findings into account and in order to avoid distraction from the main text and to make the paper well focused, we aim at showing that feature ranking methods (especially the proposed one) alone are able to improve the poor performance of standard classifiers on this type of data sets. So, we considered three of the most well-known standard classifiers with different biases in our experiments i.e. Naïve Bayes, Nearest Neighbor and Linear SVM (similar to previous studies in our field of research) and do not consider algorithmic approaches (the second category of imbalanced learning methods which are designed to develop a classifier that is intrinsically insensitive to class skewness of the data set) in our experiments. Although investigating the effects of accompanying the proposed feature ranking method (DBFS) with those classifiers that are mainly designed to focus on imbalanced data sets such as the ones proposed in [29–35,59], could be mentioned in future research work.

Since all data sets are composed of a moderate number of instances, evaluations for each feature ranking method are done using 4-fold stratified cross validation repeated ten times with different sets of folds for each data set.

4.3. Evaluation statistics

On extremely imbalanced data sets, algorithms have difficulties in classifying instances from the minority class because they simply classify instances as the majority class achieving a high accuracy. So, in these data sets, accuracy is not a good performance measure. There are a number of other statistics such as AUC (Area Under receiver operating characteristic Curve), F-measures, precision and recall. These statistics are commonly used to evaluate learning methods focusing on the importance of the minority class. For more information about these measures, interested reader can refer to [1,3,14]. So, we compare feature ranking methods according to the so popular F1-measure which equally weighs precision and recall. Also, as we aim to compare across all possible thresholds, we quantify the strength of different methods with a non-parametric measure as well. The ROC (Receiver Operating Characteristic curve) will allow us to find the strength of a classifier at each possible threshold. To quantify this curve with a single statistic, we evaluated the classifiers using AUC. Based on above mentioned performance measures, the evaluation is carried on when respectively 0.1%, 0.25%, 0.5%, 1%, 2.5%, 5% and 10% of features are selected by the feature ranking methods and are fed to NB, 1-NN and LSVM classifiers for training phase.

Table 6
Wilcoxon signed-ranks ($\alpha = 0.1$) pair wise comparison of the AUC (white cells) and F1 (gray cells) performance measures of NB, 1-NN, LSVM classifiers averaged over all data sets when 0.1% of original features are selected by rival feature ranking methods.

	CHI			IG			S2N			PCC			FAST			DBFS		
Classifier	NB	1-NN	LSV M	NB	1-NN	LSV M	NB	1-NN	LSV M	NB	1-NN	LSV M	NB	1-NN	LSV M	NB	1-NN	LSV M
CHI				0.001 0 □	0.001 7 ■	0.0003 ■	0.658 0 ○	0.497 9 □	0.5699 □	0.414 0 □	0.123 0 ○	0.1311 ○	0.026 2 □	0.858 3 □	0.1396 □	0.002 2 □	0.003 7 ■	0.0002 ■
IG	0.025 1 ○	0.000 5 ●	0.0010 ●				0.009 6 ○	0.008 7 ●	0.002 2 ●	0.012 9 ○	0.000 2 ●	0.0001 ●	0.000 8 ○	0.005 5 ●	0.0619 ●	0.637 8 □	0.807 6 ○	0.0363 ■
S2N	0.426 4 □	0.569 9 ○	0.2914 □	0.049 5 □	0.005 0 ■	0.0024 ■				0.499 7 □	0.184 2 ○	0.2360 ○	0.030 9 □	0.661 2 □	0.1997 □	0.009 0 ■	0.009 0 ■	0.0004 ■
PCC	0.372 0 □	0.177 9 □	0.0459 ■	0.148 5 □	0.000 1 ■	0.0002 ■	0.048 6 ○	0.243 2 □	0.4455 □				0.020 3 ○	0.076 8 ■	0.0090 ■	0.001 5 ■	0.000 2 ■	0.0001 ■
FAST	0.306 5 ●	0.592 2 ○	0.1011 □	0.018 6 ○	0.008 1 ■	0.0019 ■	0.935 3 ●	0.987 0 □	0.3896 □	0.262 7 ●	0.177 9 ○	0.9353 ○				0.000 1 ■	0.001 9 ■	0.0010 ■
DBFS	0.108 0 ○	0.001 5 ●	0.1011 ○	0.987 0 ○	0.858 3 ○	0.1886 □	0.057 5 ●	0.008 1 ●	0.0335 ●	0.094 5 ●	0.000 1 ●	0.0008 ●	0.026 2 □	0.009 9 ●	0.0055 ●			

Table 7
Wilcoxon signed-ranks ($\alpha = 0.1$) pair wise comparison of the AUC (white cells) and F1 (gray cells) performance measures of NB, 1-NN, LSVM classifiers averaged over all data sets when 0.25% of original features are selected by rival feature ranking methods.

	CHI			IG			S2N			PCC			FAST			DBFS		
Classifier	NB	1-NN	LSV M	NB	1-NN	LSV M	NB	1-NN	LSV M	NB	1-NN	LSV M	NB	1-NN	LSV M	NB	1-NN	LSV M
CHI				0.000 3 ■	0.000 1 ■	0.0001 ■	0.708 9 ○	0.180 8 ○	0.1997 ○	0.018 6 ■	0.807 6 □	0.3720 □	0.008 1 ●	0.548 1 ○	0.3221 □	0.000 1 ■	0.006 7 ■	0.0001 ■
IG	0.0012 8 ●	0.000 1 ●	0.0010 ●				0.005 5 ●	0.000 2 ●	0.00 01 ●	0.935 3 □	0.024 0 ●	0.0015 ●	0.000 8 ●	0.033 5 ●	0.0575 ●	0.002 4 ■	0.066 6 ■	0.0002 ■
S2N	0.2234 3 □	0.188 6 □	0.1779 □	0.000 5 ■	0.000 1 ■	0.0017 ■				0.007 4 ■	0.020 3 ■	0.0130 ■	0.018 6 ●	0.614 8 □	0.0824 ■	0.000 1 ■	0.000 1 ■	0.0001 ■
PCC	0.0208 1 ●	0.389 6 ○	0.7826 ○	0.566 3 □	0.001 1 ■	0.0495 ■	0.000 3 ●	0.009 9 ●	0.0067 ●				0.001 5 ●	0.445 5 ○	0.7826 □	0.005 0 ■	0.005 5 ■	0.00 02 ■
FAST	0.2913 6 □	0.684 9 □	0.1311 □	0.004 5 ■	0.009 8 ■	0.0108 ■	0.520 2 □	0.465 1 ○	0.8583 □	0.030 9 ■	0.372 0 □	0.1485 □				0.000 1 ■	0.001 9 ■	0.0008 ■
DBFS	0.0240 5 ●	0.000 5 ●	0.0005 ●	0.614 8 □	0.445 5 ○	0.1011 ○	0.013 0 ●	0.000 2 ●	0.0003 ●	0.372 0 ○	0.003 3 ●	0.0186 ●	0.001 4 ●	0.003 3 ●	0.0009 ●			

Table 8
Wilcoxon signed-ranks ($\alpha = 0.1$) pair wise comparison of the AUC (white cells) and F1 (gray cells) performance measures of NB, 1-NN, LSVM classifiers averaged over all data sets when 0.5% of original features are selected by rival feature ranking methods.

	CHI			IG			S2N			PCC			FAST			DBFS		
Classifier	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM
CHI				0.0057 ■	0.0015 ■	0.0003 ■	0.8076 □	0.0099 ●	0.0090 ●	0.0019 ■	0.8076 ○	0.7578 □	0.0022 ●	0.1485 ○	0.9870 ○	0.0001 ■	0.0005 ■	0.0001 ■
IG	0.0033 ●	0.0009 ●	0.0009 ●				0.0716 ●	0.0003 ●	0.0001 ●	0.2234 □	0.0203 ●	0.0011 ●	0.0002 ●	0.0033 ●	0.0221 ●	0.0004 ■	0.0108 ■	0.0003 ■
S2N	0.3065 □	0.0619 ■	0.8076 □	0.0055 ■	0.0001 ■	0.0055 ■				0.0001 ■	0.0055 ■	0.0012 ■	0.0119 ●	0.9870 □	0.0716 □	0.0001 ■	0.0001 ■	0.0001 ■
PCC	0.0335 ●	0.4651 ○	0.1886 ○	0.5699 □	0.0099 ■	0.0309 ■	0.0027 ●	0.0030 ●	0.2113 ○				0.0011 ●	0.1886 ○	0.7089 ○	0.0033 ■	0.0007 ■	0.0001 ■
FAST	0.2627 □	0.3548 □	0.1080 □	0.0071 ■	0.0024 ■	0.0045 ■	0.9095 □	0.7089 □	0.1997 □	0.0575 ■	0.0768 ■	0.0945 ■				0.0001 ■	0.0004 ■	0.0005 ■
DBFS	0.0156 ●	0.0004 ●	0.0001 ●	0.7332 ○	0.3896 ○	0.0017 ●	0.0074 ●	0.0001 ●	0.0001 ●	0.4264 ○	0.0017 ●	0.0003 ●	0.0067 ●	0.0007 ●	0.0003 ●			

Table 9
Wilcoxon signed-ranks ($\alpha = 0.1$) pair wise comparison of the AUC (white cells) and F1 (gray cells) performance measures of NB, 1-NN, LSVM classifiers averaged over all data sets when 1% of original features are selected by rival feature ranking methods.

	CHI			IG			S2N			PCC			FAST			DBFS		
Classifier	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM
CHI				0.0005 ■	0.0001 ■	0.0001 ■	0.7826 ○	0.0001 ●	0.0050 ●	0.0094 ■	0.5922 ○	0.2360 □	0.0015 ●	0.9870 ○	0.9870 □	0.0002 ■	0.0005 ■	0.0001 ■
IG	0.0392 ●	0.0004 ●	0.0004 ●				0.0221 ●	0.0000 ●	0.0001 ●	0.2914 □	0.0004 ●	0.0004 ●	0.0003 ●	0.0119 ●	0.0716 ●	0.0005 ■	0.0262 ■	0.0008 ■
S2N	0.9612 □	0.0045 ■	0.2234 □	0.1677 □	0.0000 ■	0.0003 ■				0.0001 ■	0.0003 ■	0.0008 ■	0.1311 ○	0.0033 ■	0.0186 ■	0.0001 ■	0.0000 ■	0.0000 ■
PCC	0.0050 ●	0.0883 ●	0.2491 ○	0.1579 ○	0.0309 ■	0.0221 ■	0.0004 ●	0.0001 ●	0.0064 ●				0.0014 ●	0.7826 □	0.9612 □	0.0067 ■	0.0003 ■	0.0000 ■
FAST	0.0534 ■	0.4852 □	0.8076 ○	0.0041 ■	0.0130 ■	0.0363 ■	0.4077 □	0.0534 ●	0.2360 ○	0.0074 ■	0.1230 □	0.4651 □				0.0002 ■	0.0015 ■	0.0010 ■
DBFS	0.0003 ●	0.0002 ●	0.0002 ●	0.0262 ●	0.1154 ○	0.0203 ●	0.0006 ●	0.0001 ●	0.0000 ●	0.1779 ○	0.0005 ●	0.0001 ●	0.0006 ●	0.0009 ●	0.0015 ●			

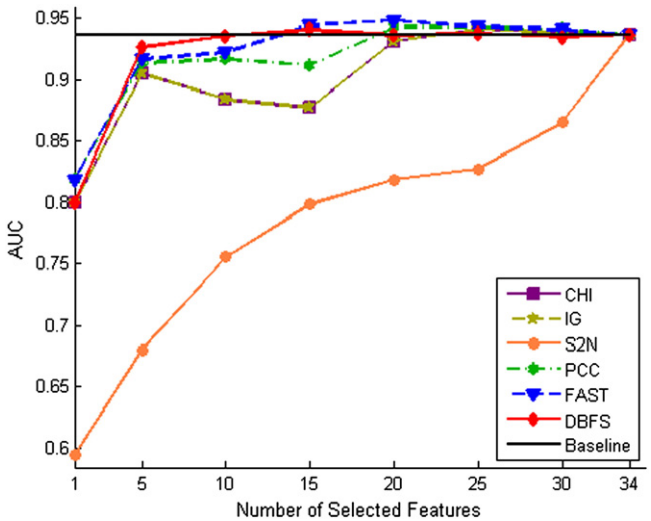

Fig. 7. The performance of the NB classifier across the IONOSPHERE data set in terms of the AUC evaluation statistic.

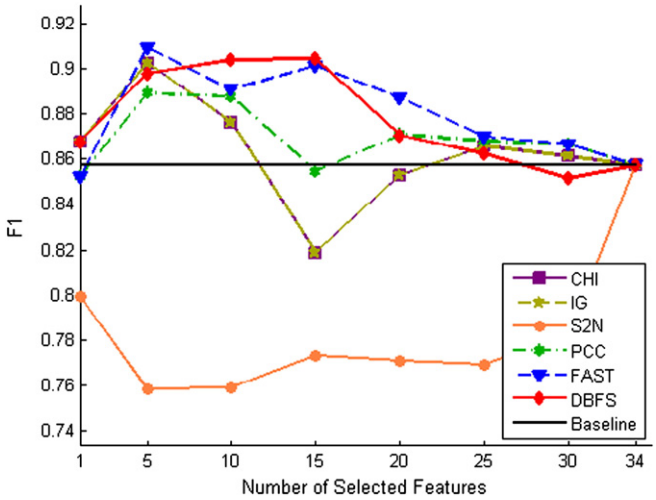

Fig. 8. The performance of the NB classifier across the IONOSPHERE data set in terms of the F1 evaluation statistic.

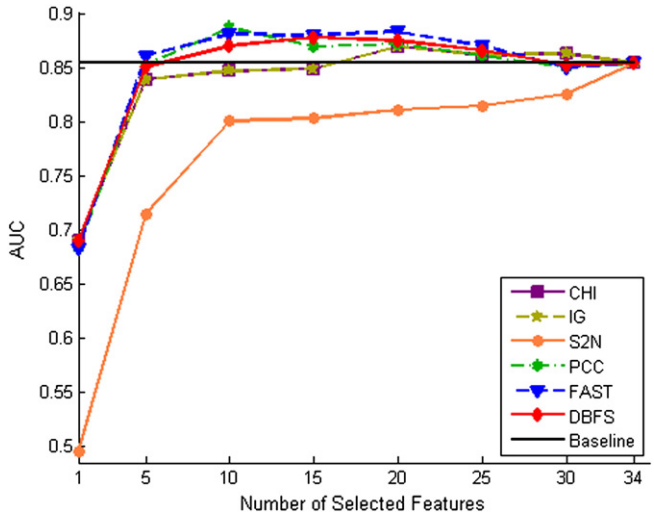

Fig. 9. The performance of the 1-NN classifier across the IONOSPHERE data set in terms of the AUC evaluation statistic.

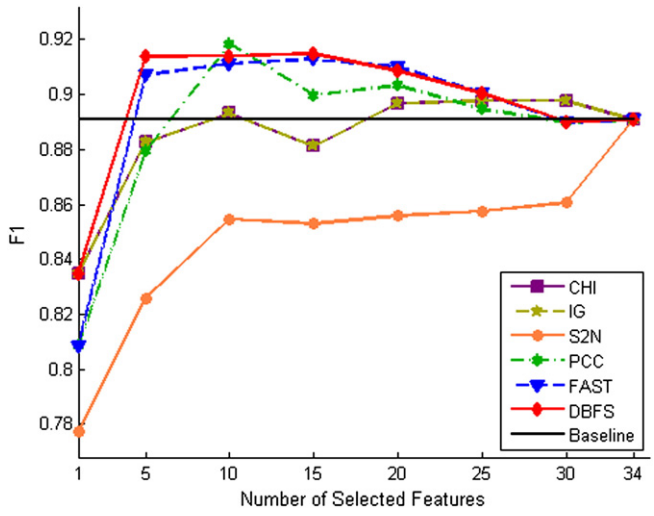

Fig. 10. The performance of the 1-NN classifier across the IONOSPHERE data set in terms of the F1 evaluation statistic.

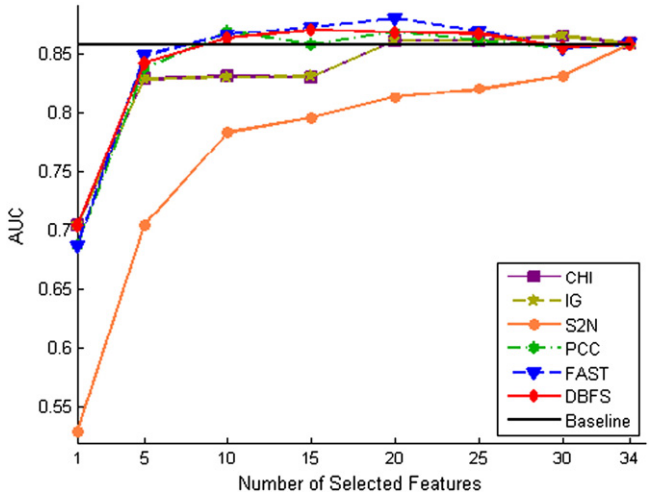

Fig. 11. The performance of the LSVM classifier across the IONOSPHERE data set in terms of the AUC evaluation statistic.

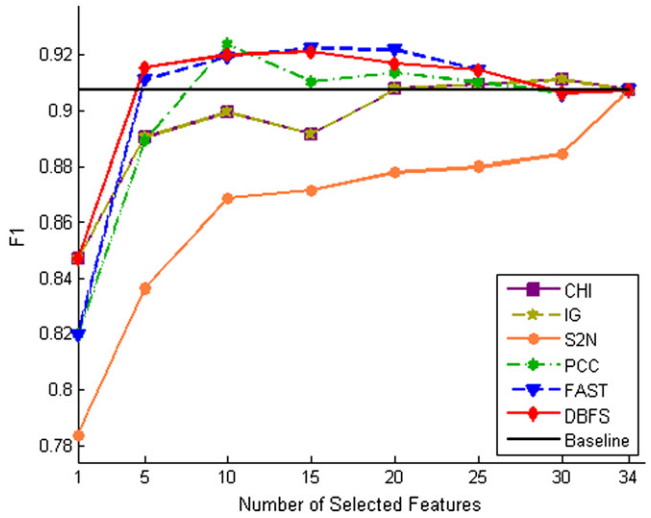

Fig. 12. The performance of the LSVM classifier across the IONOSPHERE data set in terms of the F1 evaluation statistic.

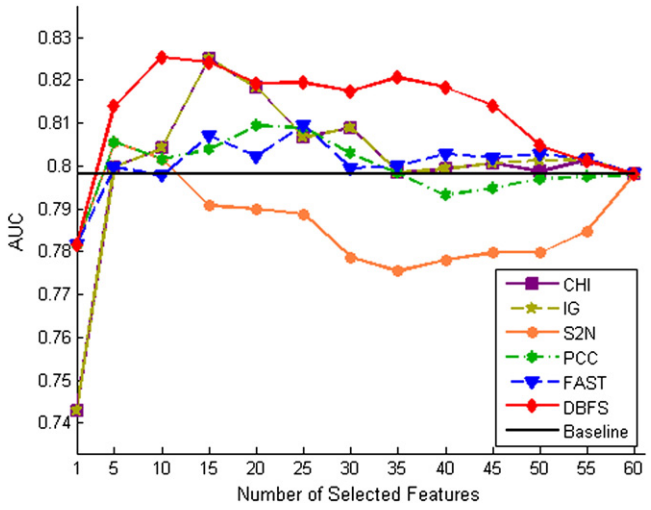

Fig. 13. The performance of the NB classifier across the SONAR data set in terms of the AUC evaluation statistic.

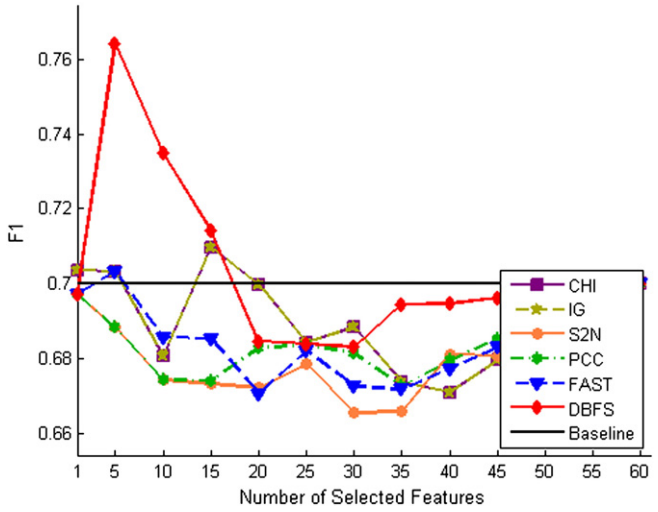

Fig. 14. The performance of the NB classifier across the SONAR data set in terms of the F1 evaluation statistic.

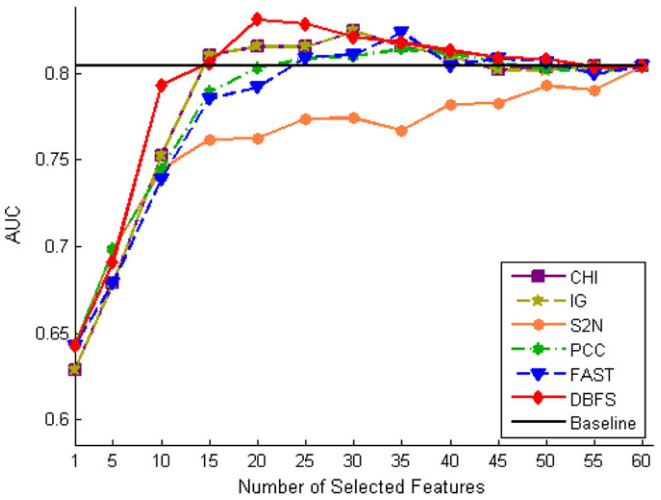

Fig. 15. The performance of the 1-NN classifier across the SONAR data set in terms of the AUC evaluation statistic.

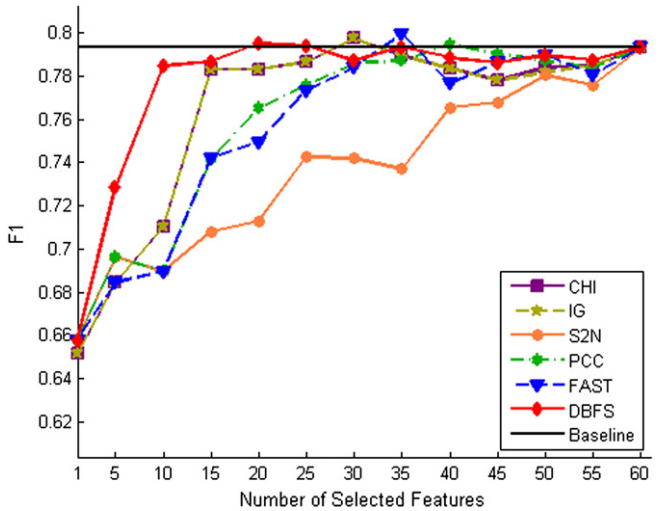

Fig. 16. The performance of the 1-NN classifier across the SONAR data set in terms of the F1 evaluation statistic.

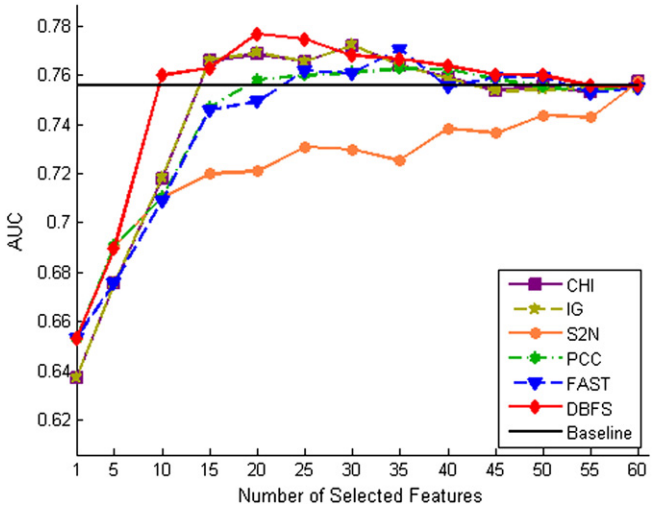

Fig. 17. The performance of the LSVM classifier across the SONAR data set in terms of the AUC evaluation statistic.

4.4. Experimental results

We compare the performance of DBFS with five well-known state of the art feature ranking methods i.e. CHI [1,3,14], IG [3,14,42], PCC [1,3,14,42], S2N [14,58] and FAST [1,14]. Also, we are going to answer the following questions as some descriptive comparison measures.

The first question we will answer is that by averaging the achieved performance over all data sets, which feature ranking methods perform the best? In order to answer this question, we average the performance of different feature ranking methods over biological and text data sets. Results are not shown here due to the page limit. The results show the superiority of the proposed feature ranking method.

To show that the performance of DBFS is statistically significant in comparison with other rival feature selection methods, the p-value results of a non-parametric Wilcoxon signed-ranks test [60,61] at a significant level ($\alpha = 0.1$) are illustrated in Tables 6–9 when 0.1%, 0.25%, 0.5% and 1% of features are selected for each classifier. The upper triangular cells (white cells) in Tables 6–9 indicate the p-values gained according to AUC evaluation measure and the lower triangular ones (gray cells) show the p-values gained according to F1 evaluation measure. The ●/■ symbols respectively denote that the performance of the feature ranking method in a row significantly outperforms/degrades the performance of the one in a column. The ○/□ symbols indicate that outperformance/degradation in the average results are not statistically significant.

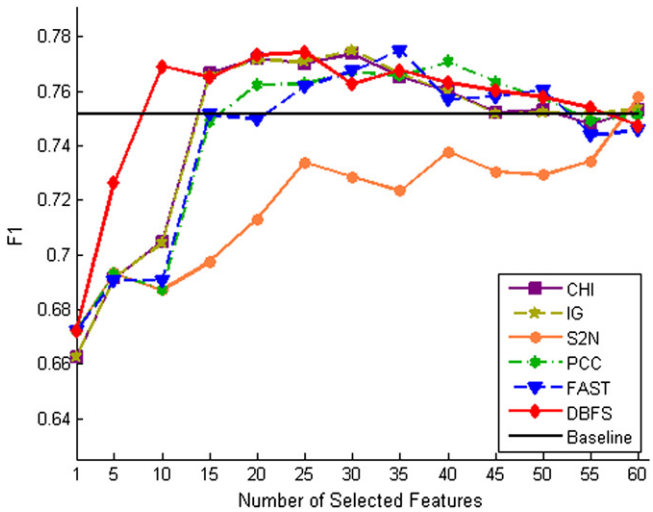

Fig. 18. The performance of the LSVM classifier across the SONAR data set in terms of the F1 evaluation statistic.

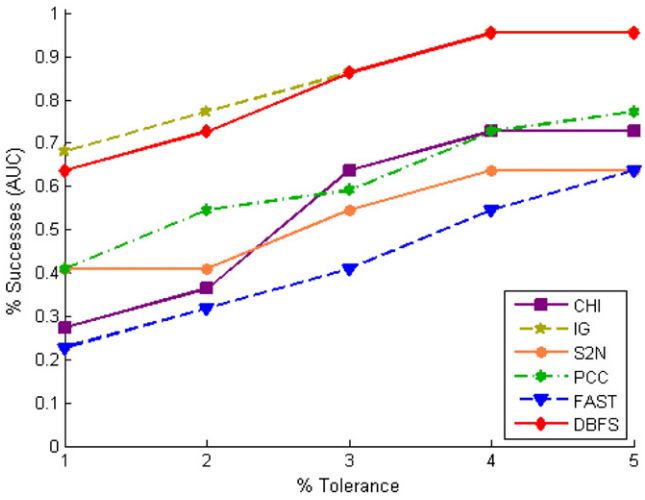

Fig. 19. The percentage of data sets for which case feature ranking methods perform within the tolerance of the best performing method. The curves show AUC of NB classifiers when applied on 10 features selected by corresponding feature ranking methods.

According to the averaged results over all biological and text data sets and statistical analyses, i.e. Tables 6–9, DBFS feature ranking method performs the best using AUC and F1 evaluation statistics across all percentages of selected features statistically with a significant level ($\alpha = 0.1$). The second best method is the IG method for 1-NN and LSVM classifiers and the PCC method for the NB classifier. Note that all feature ranking methods outperform the baseline method which acknowledges that feature selection is a key solution for learning algorithms on small sample size and high dimensional imbalanced data sets. Considering results from different classifiers, a clear trend is that the gains in performance of NB and 1-NN classifiers over the baseline are larger than those for the LSVM. This is mainly due to the fact that LSVM is more resistant to feature selection than NB and 1-NN classifiers.

The summit performance of feature selection approaches in terms of evaluation measures used in this study is roughly the points where the percentage of selected features is 0.5 or 1. Since machine learning and data mining communities share a common objective of attempting to find a model with high performance and low complexity, it seems that selection of 0.5 to 1% of features in high dimensional imbalanced data sets is the right course of action to achieve these objectives.

Also, it is notable that selection of more than 1% of features results in a significant reduction in performance; however, we expect that this reduction must happen after selecting a large percentage of features (about half of the original number of features) due to the appearance of noisy and irrelevant features. To further investigate this issue, we evaluated the performance of

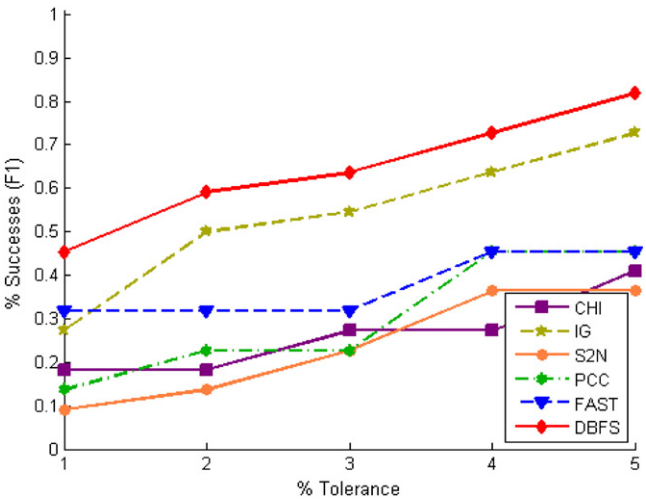

Fig. 20. The percentage of data sets for which case feature ranking methods perform within the tolerance of the best performing method. The curves show F1 of NB classifiers when applied on 10 features selected by corresponding feature ranking methods.

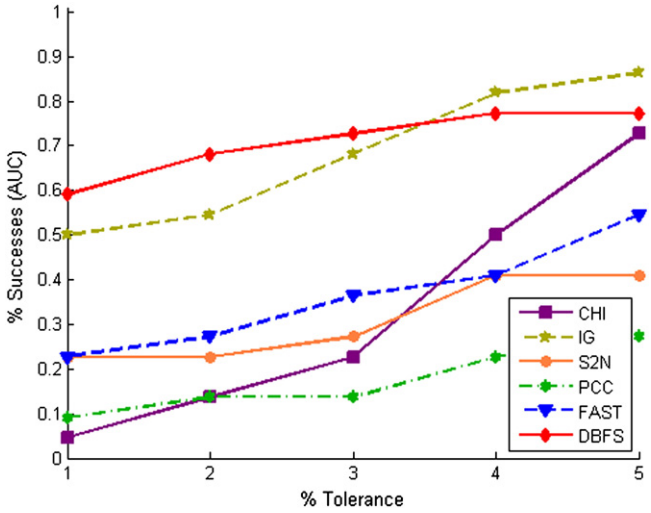

Fig. 21. The percentage of data sets for which case feature ranking methods perform within the tolerance of the best performing method. The curves show AUC of 1-NN classifiers when applied on 0.1% of features selected by corresponding feature ranking methods.

different feature ranking methods on two well-known UCI data sets (IONOSPHERE and SONAR). Figs. 7–18 show the results of NB, 1-NN and LSVM classifiers using different numbers of selected features according to AUC and F1 evaluation statistics. As the legends show, dashed lines with pentagram markers indicate the IG method and solid lines with a square, circle and diamond, respectively indicate CHI, S2N and DBFS methods. Moreover, the dash-dotted line with an asterisk and the dashed line with a downward-pointing triangle indicate PCC and FAST feature ranking methods, respectively. Also, the solid black line shows the baseline performance where all the features were used for classification.

According to Figs. 7–18, despite minor fluctuations, the performance of all classifiers increases as the number of selected features increases until a salient number of features is selected. Afterwards, a decrease in performance is perceived due to the fact that the greater the number of selected features, the higher the chance of being prone to irrelevant and noisy features.

Nonetheless, this statement does not stand for biological and text mining data sets. To explore the reason, features of CNS1, CNS2 and NIPS_1 data sets were extensively studied. The observations show that the exclusion of these data sets from this statement is due to the presence of a high number of irrelevant features. Thus, it is vital to select the appropriate number of selected features. With too few features, the performance of the model may not be desirable, whereas, a large number of selected

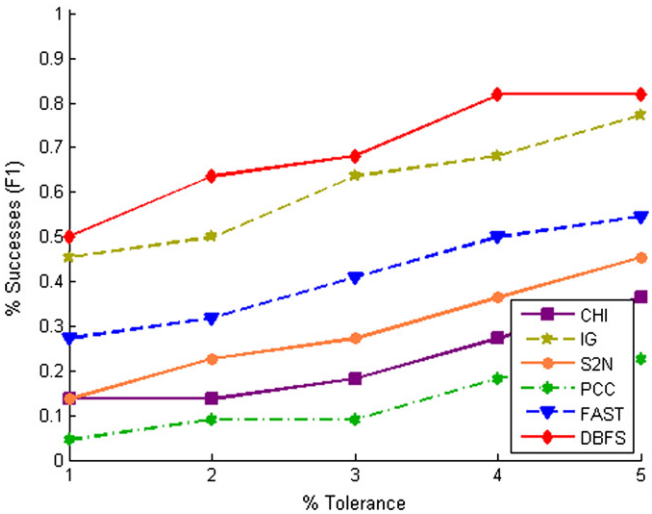

Fig. 22. The percentage of data sets for which case feature ranking methods perform within the tolerance of the best performing method. The curves show F1 of 1-NN classifiers when applied on 0.1% of features selected by corresponding feature ranking methods.

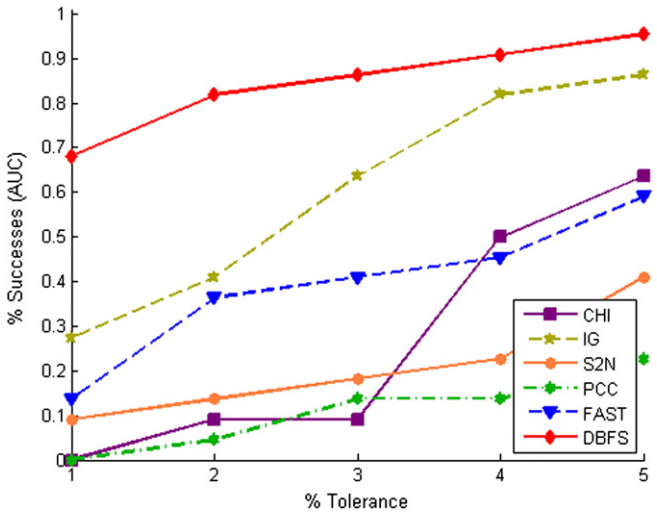

Fig. 23. The percentage of data sets for which case feature ranking methods perform within the tolerance of the best performing method. The curves show AUC of LSVM classifiers when applied on 0.1% of features selected by corresponding feature ranking methods.

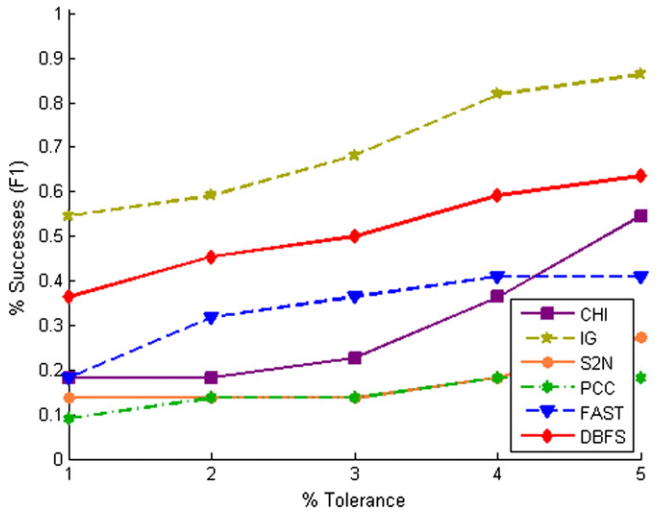

Fig. 24. The percentage of data sets for which case feature ranking methods perform within the tolerance of the best performing method. The curves show F1 of LSVM classifiers when applied on 0.1% of features selected by corresponding feature ranking methods.

features leads to overfitting of data and hence incurs misclassification costs. Another trend that can be seen in the experimental results is that selecting about 0.5% of features appears to be the point where every feature ranking methods peak across each evaluation statistics. With more than 0.5% of features being selected, the performance degrades significantly. Since the goal of data mining researches is to find the simple yet accurate hypothesis; on high dimensional imbalanced data sets it seems that by selecting about 0.5% of features, this goal is achievable. Thus, we recommend building the primary classifier with 0.5% of features and empirically comparing the results with classifiers trained on different numbers of selected features to find the best number of features.

The second question we try to answer is: which feature ranking methods are more likely to perform the best for a single data set?

Most data mining researchers would prefer a learning method which gives the best results for their problem of interest and where they would not care about the average results anymore. Thus, we used the framework introduced in [3] to better answer this question. In the first step of this framework, for each data set, we take the best performance scores achieved by all feature ranking methods. Then, for each feature ranking method, we find the ratio of data sets for which the desired method performs within a certain tolerance level of the best score achieved for each data set. For instance, if the best score is 0.9, then

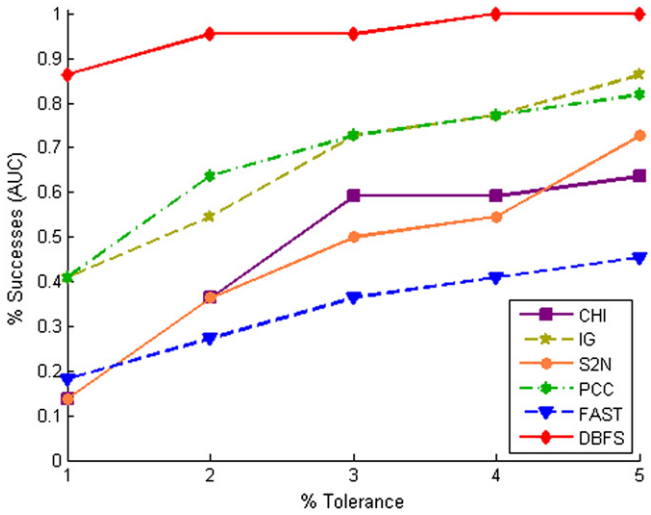

Fig. 25. The percentage of data sets for which case feature ranking methods perform within the tolerance of the best performing method. The curves show AUC of NB classifiers when applied on 0.5% of features selected by corresponding feature ranking methods.

for an admissible tolerance level of 1%, the performance equal to 0.892 would be acceptable and the performance equal to 0.885 would be unacceptable. Feature ranking methods with the highest ratios are those that are closest to the optimal. We performed this test for both the smallest percentage of selected features (0.1% of features selected) and the optimal percentage of selected features (0.5% of features selected) with all three classifiers across both evaluation statistics. Figs. 19–24 show the results for 0.1% of selected features and Figs. 25–30 show the results for the optimal percentage (0.5%) of selected features.

As is illustrated, by lower levels of tolerance, especially for the 1% tolerance level, the performance of DBFS is marginally better and for the AUC evaluation statistic, the difference is more promising. Also, as the percentage of selected features increases, the margin would increase as well. At the highest levels of allowed tolerances, the differences between feature ranking methods decrease. Thus, the feature ranking method with nearly the optimal performance is DBFS across all classifiers and evaluation statistics. For LSVM and 1-NN classifiers, IG and FAST methods are the second best ranking methods and for the NB classifier, IG and PCC methods are the second best. Since most researchers would prefer the smallest possible values for tolerances, the 1% tolerance level is the most remarkable. Therefore, DBFS is undoubtedly the method of choice.

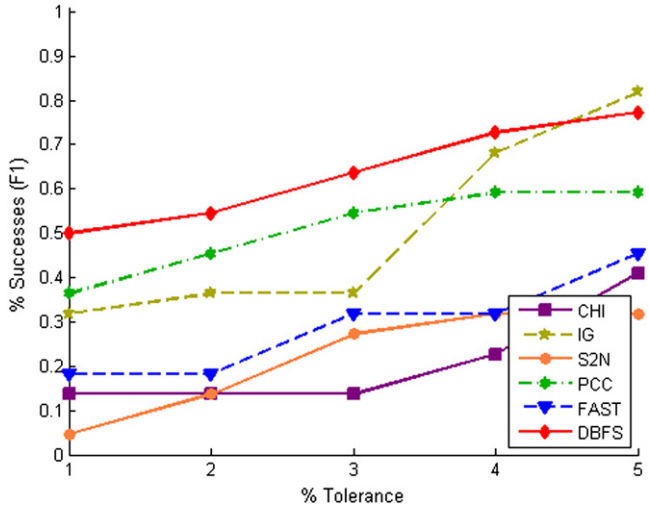

Fig. 26. The percentage of data sets for which case feature ranking methods perform within the tolerance of the best performing method. The curves show F1 of NB classifiers when applied on 0.5% of features selected by corresponding feature ranking methods.

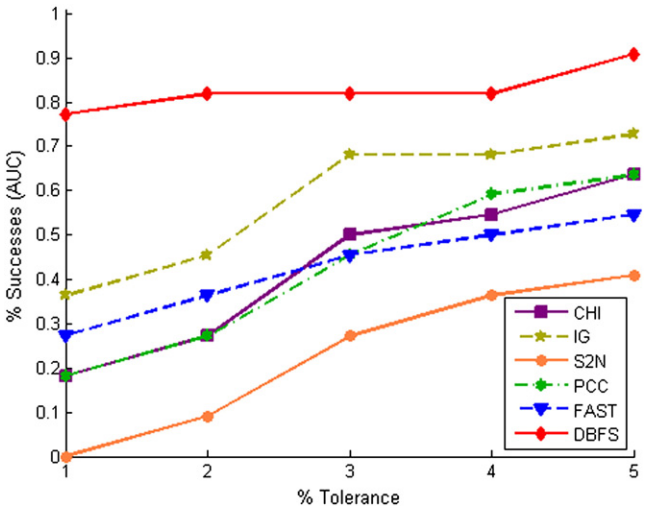

Fig. 27. The percentage of data sets for which case feature ranking methods perform within the tolerance of the best performing method. The curves show AUC of 1-NN classifiers when applied on 0.5% of features selected by corresponding feature ranking methods.

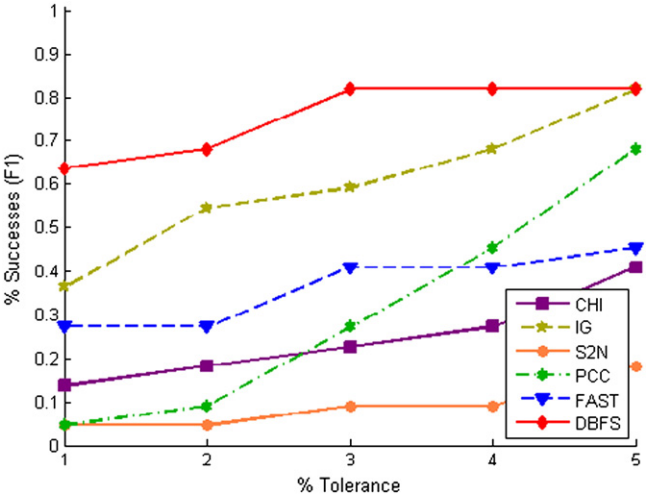

Fig. 28. The percentage of data sets for which case feature ranking methods perform within the tolerance of the best performing method. The curves show F1 of 1-NN classifiers when applied on 0.5% of features selected by corresponding feature ranking methods.

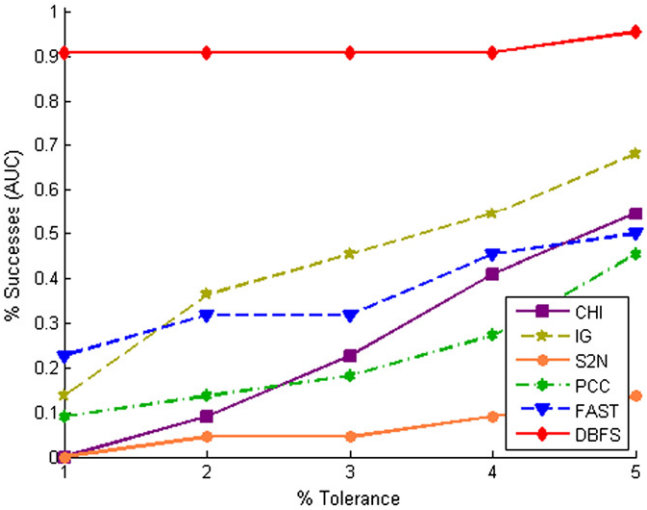

Fig. 29. The percentage of data sets for which case feature ranking methods perform within the tolerance of the best performing method. The curves show AUC of LSVM classifiers when applied on 0.5% of features selected by corresponding feature ranking methods.

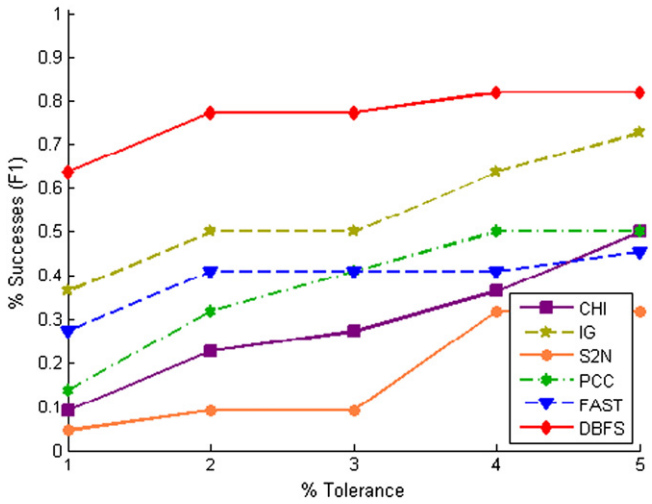

Fig. 30. The percentage of data sets for which case feature ranking methods perform within the tolerance of the best performing method. The curves show F1 of LSVM classifiers when applied on 0.5% of features selected by corresponding feature ranking methods.

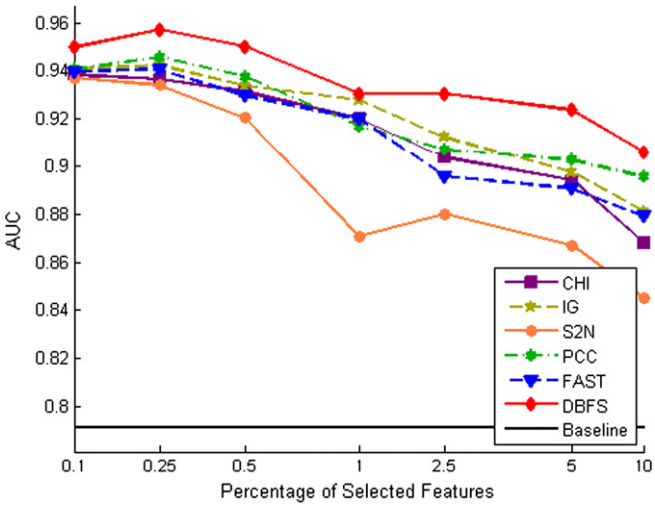

Fig. 31. The performance of the NB classifier across all biological data sets in terms of the AUC evaluation statistic. Each point represents the value of AUC averaged over all biological data sets.

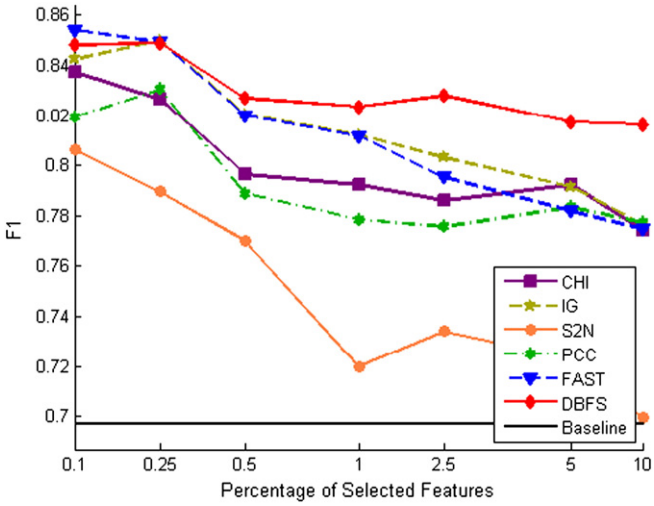

Fig. 32. The performance of the NB classifier across all biological data sets in terms of the F1 evaluation statistic. Each point represents the value of F1 averaged over all biological data sets.

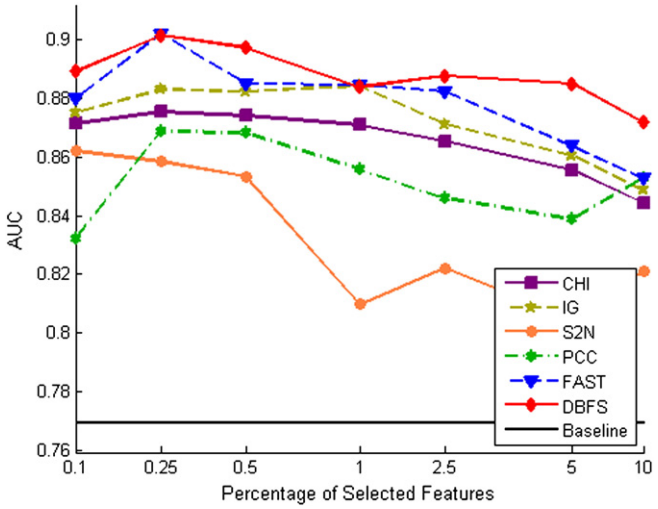

Fig. 33. The performance of the 1-NN classifier across all biological data sets in terms of the AUC evaluation statistic. Each point represents the value of AUC averaged over all biological data sets.

The third question which we aim to answer is: which feature ranking methods are the best choices for different domains?

As was stated before, imbalanced data sets are pervasive in different real world applications. Based on the inherent characteristics of data sets in each domain, different machine learning methods are appropriate. So, we divided data sets into biological and text mining domains and analyzed each feature ranking method from the recent perspective. Figs. 31–42 show the average AUC and F1 performances of different feature ranking methods across classifiers for biological data sets. Similarly, Figs. 37–42 show the same results for the text mining problems.

To statistically compare the performances of rival feature ranking methods in each domain (biology and text), the p-values resulting from a non-parametric Wilcoxon signed-ranks test [60,61] at a significant level ($\alpha = 0.1$) are illustrated in Tables 10–13 and Tables 14–17 when 0.1%, 0.25%, 0.5% and 1% of features selected for each classifier in each domain, separately. The upper triangular cells (white cells) in these tables indicate the p-values gained according to AUC evaluation measure and the lower triangular ones (gray cells) show the p-values gained according to F1 evaluation measure. The ●/■ symbols respectively denote that the performance of the feature ranking method in a row significantly outperforms/degrades the performance of the one in a column. The ○/□ symbols indicate that outperformance/degradation in the average results are not statistically significant.

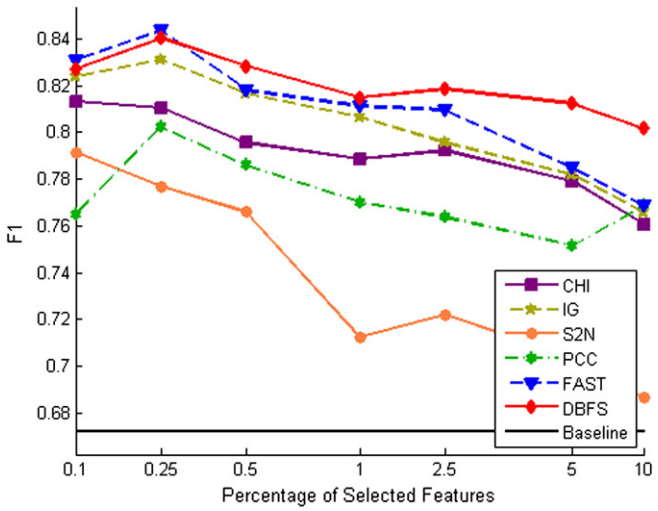

Fig. 34. The performance of the 1-NN classifier across all biological data sets in terms of the F1 evaluation statistic. Each point represents the value of F1 averaged over all biological data sets.

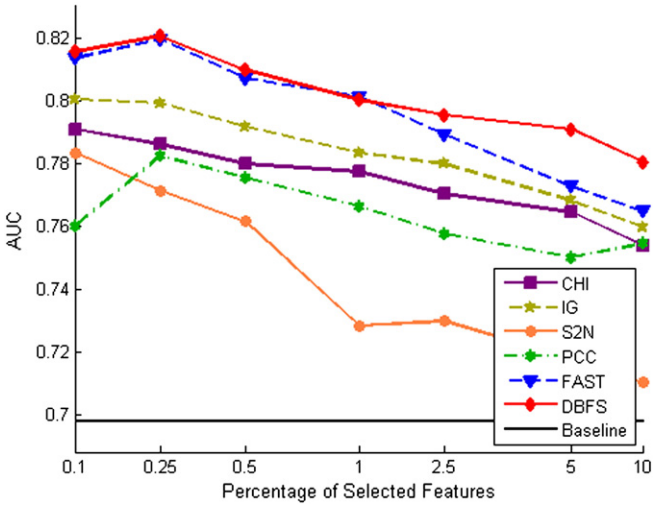

Fig. 35. The performance of the LSVM classifier across all biological data sets in terms of the AUC evaluation statistic. Each point represents the value of AUC averaged over all biological data sets.

For the biological data sets, the AUC performance of the DBFS method using the NB classifier is statistically better than other feature ranking methods at a significance level of $\alpha = 0.1$. From Figs. 31–36, one may find that FAST performs a little better than DBFS when 0.1% of the features are selected but it must be noted that this improvement is not statistically significant for $\alpha = 0.1$. By increasing the percentage of the selected features, DBFS usually outperforms other feature ranking methods; however for the biological domain, there is very little difference between FAST and DBFS methods. Thus, we believe that for the biological domain, both FAST and DBFS methods would be strong choices.

For the text mining domain, with only 0.1% of features being selected, DBFS and IG methods perform the best. By selecting a greater percentage of features, using the NB classifier, the DBFS method is the best ranked feature ranking method followed by the PCC method. But using 1-NN and LSVM classifiers, the second ranked feature ranking method behind DBFS, would be IG. Therefore, according to Figs. 37–42 and Tables 14–17, it is obvious that the best feature ranking method to be applied on text mining problems is again the DBFS method.

As another question, it is interesting to know which feature ranking methods perform the best regardless of the classifier used.

It is clear that some feature ranking methods perform better when accompanied by a specific classifier. For example, since the RELIEF feature ranking method [46] is designed based on the nearest neighbor philosophy, it offers the 1-NN algorithm more improvement than simple correlation coefficients [1]. In many situations, it is preferred to find a feature ranking method which performs the best on different classifiers with different biases in comparison with a feature ranking

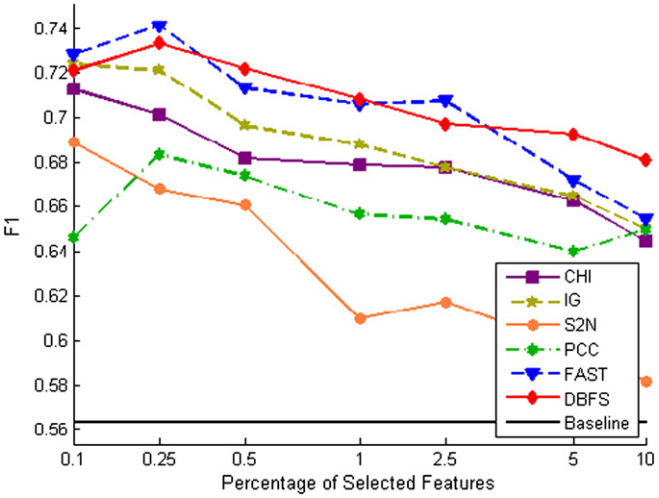

Fig. 36. The performance of the LSVM classifier across all biological data sets in terms of the F1 evaluation statistic. Each point represents the value of F1 averaged over all biological data sets.

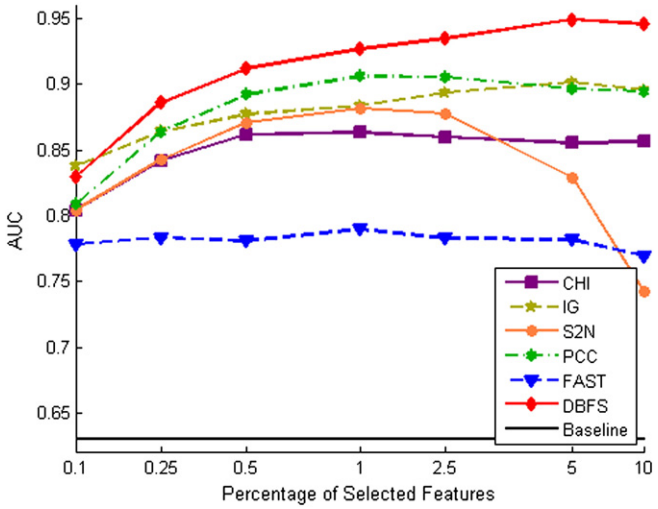

Fig. 37. The performance of the NB classifier across all text data sets in terms of the AUC evaluation statistic. Each point represents the value of AUC averaged over all text data sets.

method that performs well on just a specific classifier. Thus, to investigate the ability of DBFS regardless of the classifier used, we averaged the AUC and F1 performances of all classifiers over all data sets. The results for AUC and F1 evaluation measures are depicted in Figs. 43 and 44 respectively. Those feature ranking methods that have the highest performance, regardless of the classifier used, have a higher chance to select as the best features. Considering the results, DBFS has the highest performance across AUC and F1 evaluation statistics over all data sets regardless of the classifier used. As it is shown, selecting 0.1% of features, DBFS and IG methods tie for the best F1 performance. By selecting a greater percentage of features, DBFS performs statistically much better than all other feature ranking methods across AUC and F1 measures with a significance level of 0.05.

One might be interested to see how skewness of class distribution (degree of class imbalance) would affect the performance of each feature ranking method. Figs. 45–52 show the AUC and F1 evaluation statistics averaged over all classifiers versus various class ratios for the NIPS data set (NIPS_7, NIPS_9, NIPS_10, NIPS_11, NIPS_12, NIPS_13, respectively) when 0.1%, 0.25%, 0.5% and 1% of features are selected. Not surprisingly, by increasing the class imbalance ratio, F1 and AUC evaluation measures decrease.

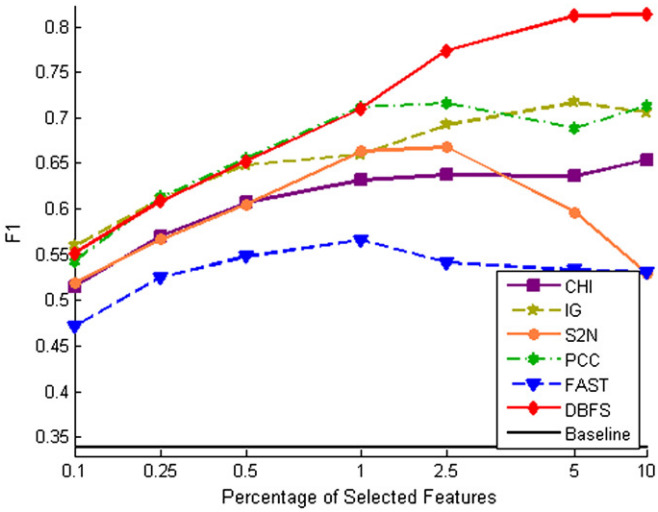

Fig. 38. The performance of the NB classifier across all text data sets in terms of the F1 evaluation statistic. Each point represents the value of F1 averaged over all text data sets.

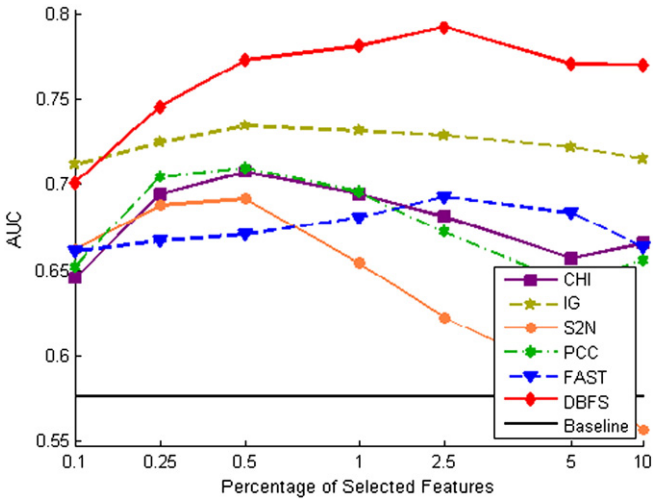

Fig. 39. The performance of the 1-NN classifier across all text data sets in terms of the AUC evaluation statistic. Each point represents the value of AUC averaged over all text data sets.

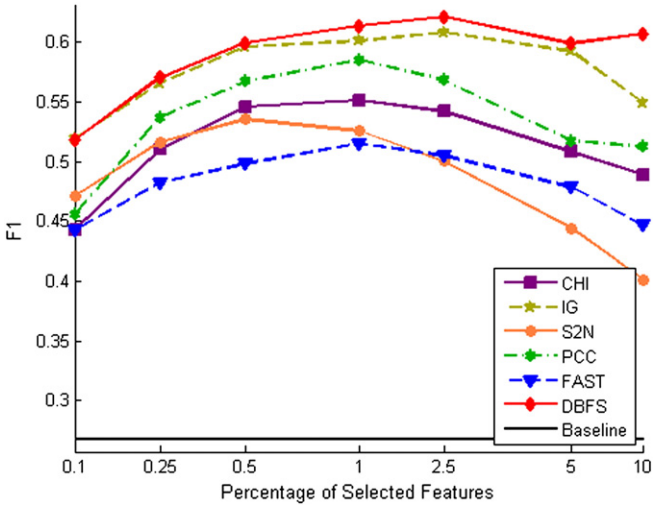

Fig. 40. The performance of the 1-NN classifier across all text data sets in terms of the F1 evaluation statistic. Each point represents the value of F1 averaged over all text data sets.

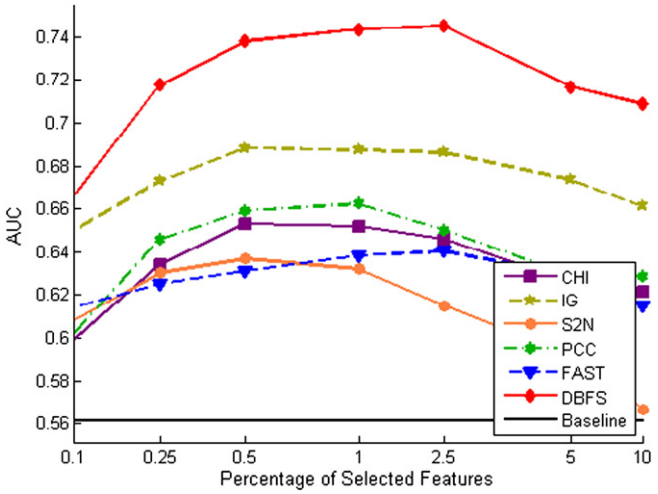

Fig. 41. The performance of the LSVM classifier across all text data sets in terms of the AUC evaluation statistic. Each point represents the value of AUC averaged over all text data sets.

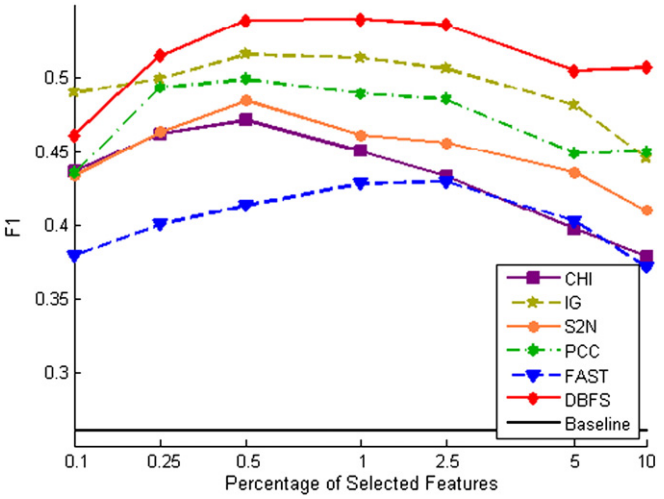

Fig. 42. The performance of the LSVM classifier across all text data sets in terms of the F1 evaluation statistic. Each point represents the value of F1 averaged over all text data sets.

Table 10
Wilcoxon signed-ranks ($\alpha = 0.1$) pair wise comparison of the AUC (white cells) and F1 (gray cells) performance measures of NB, 1-NN, LSVM classifiers averaged over all biological data sets when 0.1% of original features are selected by rival feature ranking methods.

	CHI			IG			S2N			PCC			FAST			DBFS		
Classifier	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM
CHI				0.5781 □	0.6406 □	0.2031 □	1.0000 ○	0.5469 ○	0.4258 ○	0.6406 □	0.0195 ●	0.0273 ●	0.6406 □	0.5703 □	0.0742 ■	0.1289 □	0.0977 ■	0.0547 ■
IG	0.6875 ○	0.4609 ○	0.4961 ○				1.0000 ○	0.4609 ○	0.1641 ○	0.9453 ○	0.0273 ●	0.0195 ●	0.8438 ○	0.9102 □	0.3008 □	0.2031 □	0.3594 □	0.3008 □
S2N	0.1289 □	0.1289 □	0.2500 □	0.1641 □	0.0742 ■	0.0742 ■				0.8438 □	0.1289 ○	0.1641 ○	0.7422 □	0.1641 □	0.0977 ■	0.0977 ■	0.0977 ■	0.0742 ■
PCC	0.3008 □	0.0117 ■	0.0078 ■	0.2031 □	0.0039 ■	0.0039 ■	0.7344 ○	0.1641 □	0.1641 □				0.7422 ○	0.0273 ■	0.0078 ■	0.0977 ■	0.0078 ■	0.0039 ■
FAST	0.0977 ●	0.2031 ○	0.3008 ○	0.2031 ○	0.7344 ○	0.9102 ○	0.0195 ●	0.0391 ●	0.1289 ○	0.0195 ●	0.0039 ●	0.0039 ●				0.0391 ■	0.4258 □	1.0000 □
DBFS	0.5703 ○	0.3594 ○	0.3594 ○	0.9102 ○	0.8203 ○	0.9102 □	0.0273 ●	0.1289 ○	0.0977 ●	0.0742 ●	0.0117 ●	0.0039 ●	0.2500 □	0.6523 □	0.4961 □			

Table 11
Wilcoxon signed-ranks ($\alpha = 0.1$) pair wise comparison of the AUC (white cells) and F1 (gray cells) performance measures of NB, 1-NN, LSVM classifiers averaged over all biological data sets when 0.25% of original features are selected by rival feature ranking methods.

	CHI			IG			S2N			PCC			FAST			DBFS		
Classifier	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM
CHI				0.0313 ■	0.0313 ■	0.0313 ■	0.3750 ○	0.0781 ●	0.1641 ○	0.8203 □	0.6523 ○	0.5703 ○	0.2031 □	0.0391 ■	0.0078 ■	0.0273 ■	0.1641 □	0.0273 ■
IG	0.0625 ●	0.0156 ●	0.0156 ●				0.097 7 ●	0.0078 ●	0.0156 ●	0.2500 □	0.1641 ○	0.0977 ●	1.0000 ○	0.0742 ■	0.0391 ■	0.0977 ■	0.2500 □	0.0273 ■
S2N	0.0195 ■	0.0078 ■	0.0273 ■	0.0039 ■	0.0039 ■	0.0039 ■				0.2500 □	0.3008 □	0.2500 □	0.3008 □	0.0039 ■	0.0039 ■	0.0078 ■	0.0078 ■	0.0117 ■
PCC	0.7422 ○	0.4258 □	0.0742 ■	0.3125 □	0.0547 ■	0.0273 ■	0.0977 ●	0.1289	0.2500 ○				0.8203 ○	0.0547 ■	0.0195 ■	0.1641 □	0.0742 ■	0.0547 ■
FAST	0.4258 ○	0.0547 ■	0.0039 ●	1.0000 □	0.1641 ○	0.0195 ●	0.0078 ●	0.0039 ●	0.0039 ●	0.0977 ●	0.0117 ●	0.0039 ●				0.0195 ■	0.7344 ○	0.6523 □
DBFS	0.0742 ●	0.0977 ●	0.0742 ●	1.0000 □	0.5703 ○	0.3008 ○	0.0117 ●	0.0039 ●	0.0078 ●	0.2500 ○	0.0977 ●	0.0742 ●	0.8203 □	0.9102 □	0.8203 □			

Table 12
Wilcoxon signed-ranks ($\alpha = 0.1$) pair wise comparison of the AUC (white cells) and F1 (gray cells) performance measures of NB, 1-NN, LSVM classifiers averaged over all biological data sets when 0.5% of original features are selected by rival feature ranking methods.

	CHI			IG			S2N			PCC			FAST			DBFS		
Classifier	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM
CHI				0.6406 □	0.1641 □	0.1289 □	0.0195 ●	0.0117 ●	0.0391 ●	0.6523 □	0.4258 ○	0.6523 ○	0.6523 ○	0.2031 □	0.0195 ■	0.0273 ■	0.0117 ■	0.0117 ■
IG	0.1289 ○	0.1289 ○	0.0977 ●				0.0547 ●	0.0039 ●	0.0039 ●	0.4961 □	0.1641 ○	0.0547 ●	0.2500 ○	0.8203 □	0.0977 ■	0.1094 □	0.0742 ■	0.0547 ■
S2N	0.1289 □	0.0547 ■	0.1289 □	0.0195 ■	0.0039 ■	0.0117 ■				0.0117 ■	0.0977 ■	0.0977 ■	0.1641 □	0.0078 ■	0.0039 ■	0.0078 ■	0.0039 ■	0.0039 ■
PCC	0.7344 □	0.4258 □	0.5703 □	0.3594 □	0.1641 □	0.0977 ■	0.3594 ○	0.1641 ○	0.4258 ○				1.0000 ○	0.1289 □	0.0078 ■	0.1953 □	0.0078 ■	0.0039 ■
FAST	0.0547 ●	0.0977 ●	0.0547 ●	0.7422 □	1.0000 ○	0.2500 ○	0.0039 ●	0.0039 ●	0.0039 ●	0.0547 ●	0.0117 ●	0.0078 ●				0.1094 □	0.5469 □	0.7422 □
DBFS	0.2031 ○	0.0273 ●	0.0078 ●	0.6523 ○	0.3594 ○	0.0391 ●	0.0273 ●	0.0039 ●	0.0039 ●	0.3594 ○	0.0391 ●	0.0078 ●	1.0000 ○	0.4258 ○	0.4961 ○			

Table 13
Wilcoxon signed-ranks ($\alpha = 0.1$) pair wise comparison of the AUC (white cells) and F1 (gray cells) performance measures of NB, 1-NN, LSVM classifiers averaged over all biological data sets when 1% of original features are selected by rival feature ranking methods.

	CHI			IG			S2N			PCC			FAST			DBFS		
Classifier	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM
CHI				0.1641 □	0.0391 ■	0.0391 □	0.0039 ●	0.0117 ●	0.0039 ●	0.5938 ○	0.1289 ○	0.2031 ○	0.8203 ○	0.2500 □	0.0977 ■	0.0742 ■	0.0977 ■	0.0742 ■
IG	0.0977 ●	0.0742 ●	0.5703 ○				0.0039 ●	0.0039 ●	0.0039 ●	0.3008 ○	0.0195 ●	0.0273 ■	0.4258 ○	0.8203 □	0.1289 □	0.1641 □	0.5703 ○	0.0977 ■
S2N	0.0273 ■	0.0078 ■	0.0195 ■	0.0195 ■	0.0039 ■	0.0078 ■				0.0273 ■	0.0273 ■	0.0039 ■	0.0039 ■	0.0039 ■	0.0039 ■	0.0273 ■	0.0039 ■	0.0039 ■
PCC	1.0000 □	0.1641 □	0.0742 ■	0.3594 □	0.0273 ■	0.0391 ■	0.0742 ●	0.0117 ●	0.0391 ●				0.6523 □	0.0391 ■	0.0195 ■	0.3008 □	0.0117 ■	0.0039 ■
FAST	0.2500 ○	0.0977 ●	0.2031 ○	0.8203 □	0.4258 ○	0.3008 ○	0.0039 ●	0.0039 ●	0.0039 ●	0.2500 ○	0.0195 ●	0.0117 ●				0.1484 □	0.5703 ○	0.7422 ○
DBFS	0.0273 ●	0.0391 ●	0.0742 ●	0.4258 ○	0.3594 ○	0.0977 ●	0.0039 ●	0.0039 ●	0.0039 ●	0.0195 ●	0.0039 ●	0.0078 ●	0.4961 ○	0.5703 ○	0.8203 □			

Table 14
Wilcoxon signed-ranks ($\alpha = 0.1$) pair wise comparison of the AUC (white cells) and F1 (gray cells) performance measures of NB, 1-NN, LSVM classifiers averaged over all text data sets when 0.1% of original features are selected by rival feature ranking methods.

	CHI			IG			S2N			PCC			FAST			DBFS		
Classifier	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM
CHI				0.0002 □	0.0002 ■	0.0002 □	0.7869 □	0.7354 □	0.8926 □	0.5417 □	0.9460 □	1.0000 □	0.0171 ●	0.7354 □	0.7869 □	0.0081 ■	0.0215 ■	0.0002 ■
IG	0.0266 ●	0.0002 ●	0.0002 ●				0.0024 ●	0.0081 ●	0.0034 ●	0.0034 ●	0.0007 ●	0.0005 ●	0.0002 ●	0.0005 ●	0.0061 ●	0.2439 ○	0.5879 ○	0.0681 ■
S2N	0.7869 ○	0.7354 ○	0.7869 ■	0.2439 □	0.0215 ■	0.0215 □				0.4922 □	0.9219 ○	0.9460 ○	0.0266 ●	0.6848 ○	0.8926 □	0.0398 ■	0.0479 ■	0.0007 ■
PCC	0.0327 ●	0.7869 ○	0.8394 □	0.4143 □	0.0012 ■	0.0061 ■	0.0840 ●	0.8457 □	0.6848 ○				0.0171 ●	0.7869 □	0.4973 □	0.0061 ■	0.0034 ■	0.0002 ■
FAST	0.0215 ■	0.9460 ○	0.0134 ■	0.0017 ■	0.0007 ■	0.0002 ■	0.1272 □	0.1099 □	0.0171 ■	0.0081 ■	0.3396 ○	0.0061 ■				0.0002 ■	0.0012 ■	0.0002 ■
DBFS	0.2439 ○	0.0012 ●	0.1909 ○	0.9460 □	0.9460 □	0.0681 ■	0.4548 ○	0.0398 ●	0.1272 ○	0.4143 ○	0.0005 ●	0.0681 ■	0.0024 ●	0.0012 ●	0.0002 ■			

Table 15
Wilcoxon signed-ranks ($\alpha = 0.1$) pair wise comparison of the AUC (white cells) and F1 (gray cells) performance measures of NB, 1-NN, LSVM classifiers averaged over all text data sets when 0.25% of original features are selected by rival feature ranking methods.

	CHI			IG			S2N			PCC			FAST			DBFS		
Classifier	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM
CHI				0.0012 ■	0.0002 ■	0.0002 ■	0.9460 □	0.8926 ○	0.6848 ○	0.0215 ■	0.4143 □	0.1099 ○	0.0005 ●	0.0266 ●	0.2734 ○	0.0002 ■	0.0266 ■	0.0002 ■
IG	0.0061 ●	0.0007 ●	0.0105 ●				0.0215 ●	0.0046 ●	0.0002 ●	0.6848 ○	0.0803 ●	0.0046 ●	0.0002 ●	0.0002 ●	0.0002 ●	0.0105 ■	0.2439 ■	0.0005 ■
S2N	1.0000 □	0.8926 ○	1.0000 ○	0.0134 ■	0.0007 ■	0.0574 ■				0.0215 ■	0.0327 ■	0.0171 ■	0.0017 ●	0.0803 ●	0.4548 ○	0.0012 ■	0.0005 ■	0.0002 ■
PCC	0.0105 ●	0.1099 ○	0.2163 ○	1.0000 ○	0.0081 ■	0.4973 □	0.0002 ●	0.0398 ●	0.0134 ●				0.0002 ●	0.0171 ●	0.0681 ●	0.0215 ■	0.0327 ■	0.0002 ■
FAST	0.0574 ■	0.0215 ■	0.0005 ■	0.0007 ■	0.0005 ■	0.0002 ■	0.2163 □	0.0681 ■	0.0081 ■	0.0005 ■	0.0017 ■	0.0012 ■				0.0002 ■	0.0002 ■	0.0002 ■
DBFS	0.1272 ○	0.0017 ●	0.0002 ●	0.6355 □	0.7354 ○	0.2734 ○	0.1677 ○	0.0046 ●	0.0105 ●	0.7869 □	0.0134 ●	0.1465 ○	0.0002 ●	0.0005 ●	0.0002 ●			

Table 16
Wilcoxon signed-ranks ($\alpha = 0.1$) pair wise comparison of the AUC (white cells) and F1 (gray cells) performance measures of NB, 1-NN, LSVM classifiers averaged over all text data sets when 0.5% of original features are selected by rival feature ranking methods.

	CHI			IG			S2N			PCC			FAST			DBFS		
Classifier	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM
CHI				0.0017 ■	0.0024 ■	0.0005 ■	0.2734 □	0.1677 ○	0.1272 ○	0.0002 ■	0.7869 □	0.3757 □	0.0002 ●	0.0105 ■	0.0398 ●	0.0002 ■	0.0046 ■	0.0005 ■
IG	0.0171 ●	0.0017 ●	0.0024 ●				0.4548 ○	0.0061 ●	0.0005 ●	0.0398 ■	0.0478 ●	0.0061 ●	0.0002 ●	0.0005 ●	0.0002 ●	0.0002 ■	0.0479 ■	0.0007 ■
S2N	0.9460 □	0.3757 □	0.4143 ○	0.0803 ■	0.0024 ■	0.0803 ■				0.0005 ■	0.0327 ■	0.0024 ■	0.0002 ●	0.1099 ○	0.4973 ●	0.0002 ■	0.0012 ■	0.0002 ■
PCC	0.0007 ●	0.1099 ○	0.0803 ●	0.9460 ○	0.0803 ■	0.1465 □	0.0017 ●	0.0046 ●	0.3757 ○				0.0002 ●	0.0081 ●	0.0081 ●	0.0081 ■	0.0105 ■	0.0002 ■
FAST	0.0134 ■	0.0171 ■	0.0002 ■	0.0012 ■	0.0002 ■	0.0002 ■	0.0803 ■	0.1099 □	0.0012 ■	0.0012 ■	0.0007 ■	0.0007 ■				0.0002 ■	0.0002 ■	0.0002 ■
DBFS	0.0398 ●	0.0034 ●	0.0017 ●	0.9460 ○	0.8394 ○	0.0266 ●	0.1272 ○	0.0005 ●	0.0024 ●	1.0000 □	0.0266 ●	0.0081 ●	0.0012 ●	0.0005 ●	0.0002 ●			

Table 17
Wilcoxon signed-ranks ($\alpha = 0.1$) pair wise comparison of the AUC (white cells) and F1 (gray cells) performance measures of NB, 1-NN, LSVM classifiers averaged over all text data sets when 1% of original features are selected by rival feature ranking methods.

	CHI			IG			S2N			PCC			FAST			DBFS		
Classifier	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM	NB	1-NN	LSVM
CHI				0.0005 ■	0.0002 ■	0.0002 ■	0.0171 ■	0.0012 ●	0.1099 ○	0.0012 ■	0.5417 □	0.0266 ■	0.0002 ●	0.2734 ○	0.1099 ○	0.0002 ■	0.0007 ■	0.0002 ■
IG	0.1465 ○	0.0012 ●	0.0002 ●				0.8926 ○	0.0002 ●	0.0005 ●	0.0398 ■	0.0061 ●	0.0020 ●	0.0002 ●	0.0046 ●	0.0012 ●	0.0002 ■	0.0266 ■	0.0017 ■
S2N	0.0479 ●	0.1677 □	0.3396 □	0.4973 ○	0.0002 ■	0.0081 ■				0.0002 ■	0.0012 ■	0.0215 ■	0.0002 ●	0.2439 □	0.8926 □	0.0002 ■	0.0002 ■	0.0002 ■
PCC	0.0002 ●	0.0017 ●	0.0061 ●	0.0134 ●	0.3396 □	0.2163 □	0.0002 ●	0.0007 ●	0.0923 ●				0.0002 ●	0.2734 ○	0.0803 ●	0.0061 ■	0.0034 ■	0.0002 ■
FAST	0.0002 ■	0.0398 ■	0.1099 □	0.0012 ■	0.0017 ■	0.0012 ■	0.0002 ■	0.8926 □	0.0574 ■	0.0002 ■	0.0012 ■	0.0171 ■				0.0002 ■	0.0002 ■	0.0002 ■
DBFS	0.0012 ●	0.0005 ●	0.0002 ●	0.0266 ●	0.2734 ○	0.0942 ●	0.0398 ●	0.0002 ●	0.0002 ●	0.8926 □	0.0327 ●	0.0007 ●	0.0002 ●	0.0002 ●	0.0002 ●			

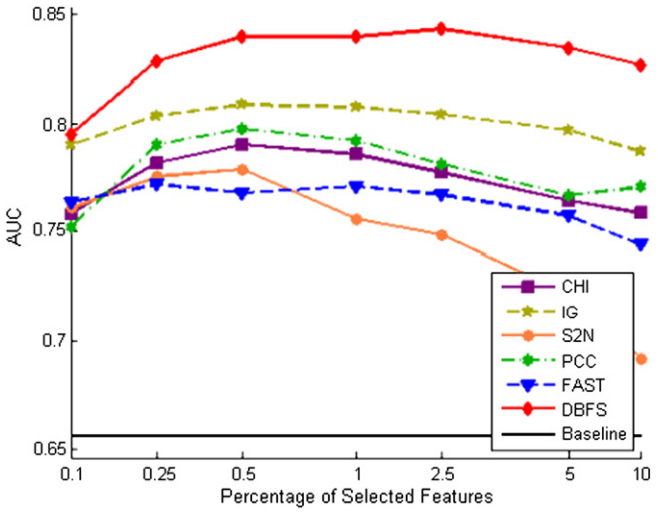

Fig. 43. The averaged performance of all classifiers across all data sets in terms of the AUC evaluation statistic. Each point represents the value of AUC averaged over all data sets regardless of the classifier used.

For the smallest percentage of selected features (0.1%) with moderate imbalance ratios (up to a 1:8 class ratio), DBFS and IG feature ranking methods perform comparably well according to AUC evaluation statistic; however, for larger imbalance ratios, FAST performs comparably well. For this percentage of selected features (0.1%) with small imbalance ratios (up to 1:6 class ratio), DBFS and IG perform comparably well according to the F1 evaluation statistic. However, by increasing the imbalance ratio, IG, PCC and S2N feature ranking methods significantly outperform the DBFS feature ranking method. Selecting a moderate percentage of features (i.e. 0.5% of features selected), DBFS is significantly the best feature ranking method especially for higher imbalance ratios according to AUC evaluation statistic. According to the F1 evaluation measure, with a moderate percentage of features (i.e. 0.5%) and moderate imbalance ratios (up to 1:8 class ratio), DBFS performs better than other rival feature ranking methods but with higher imbalance ratios, the performance of DBFS, IG, S2N and PCC feature selection methods is comparable.

Based on these empirical evaluations and statistical analyses, it can be inferred that DBFS is the method of choice when 0.5% or more percentage of features are selected. This improvement is more tangible according to AUC evaluation statistic especially with higher imbalance ratios. Also, it should be mentioned that DBFS works quite good comparing to other rival feature ranking methods for both balanced and imbalanced small sample sizes and high dimensional data. However, its superiority is more significant when dealing with imbalanced small sample size and high dimensional data sets.

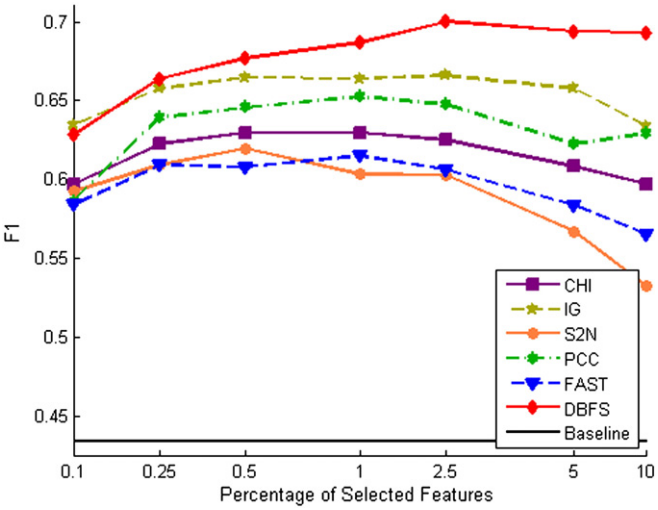

Fig. 44. The averaged performance of all classifiers across all data sets in terms of the F1 evaluation statistic. Each point represents the value of F1 averaged over all data sets regardless of the classifier used.

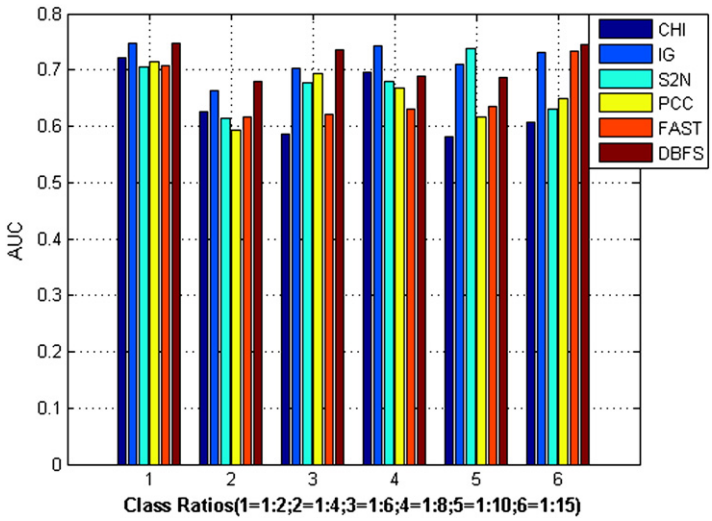

Fig. 45. Averaged AUC over all classifiers across different class ratios when 0.1% of features are selected.

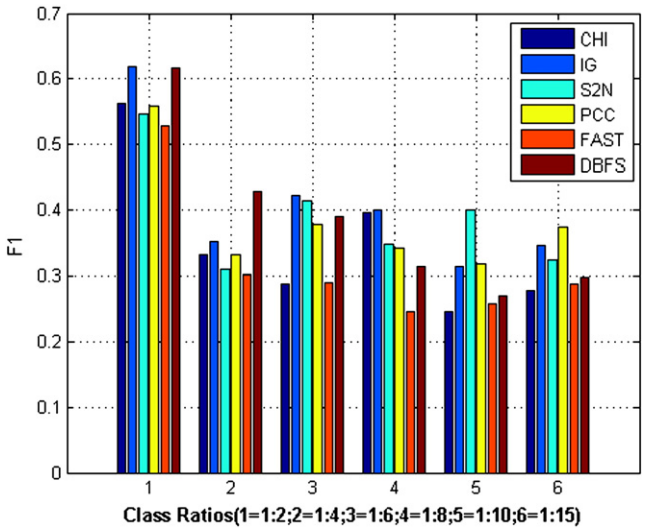

Fig. 46. Averaged F1 over all classifiers across different class ratios when 0.1% of features are selected.

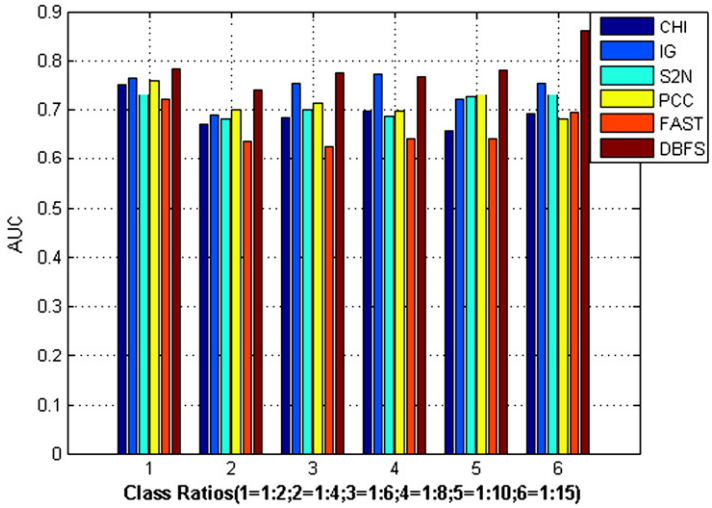

Fig. 47. Averaged AUC over all classifiers across different class ratios when 0.25% of features are selected.

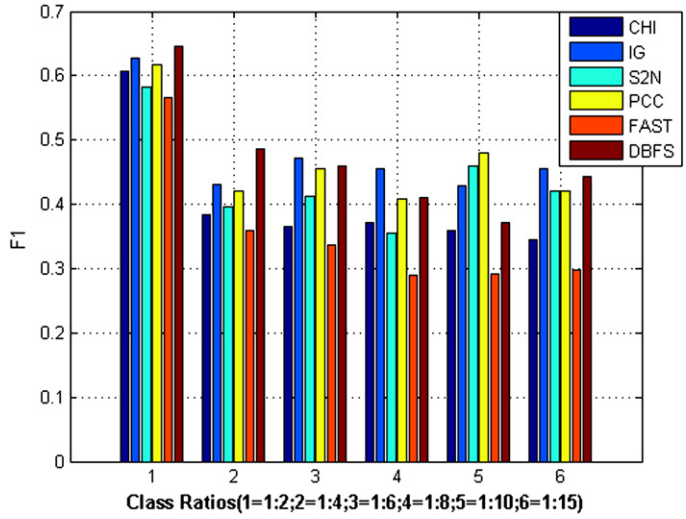

Fig. 48. Averaged F1 over all classifiers across different class ratios when 0.25% of features are selected.

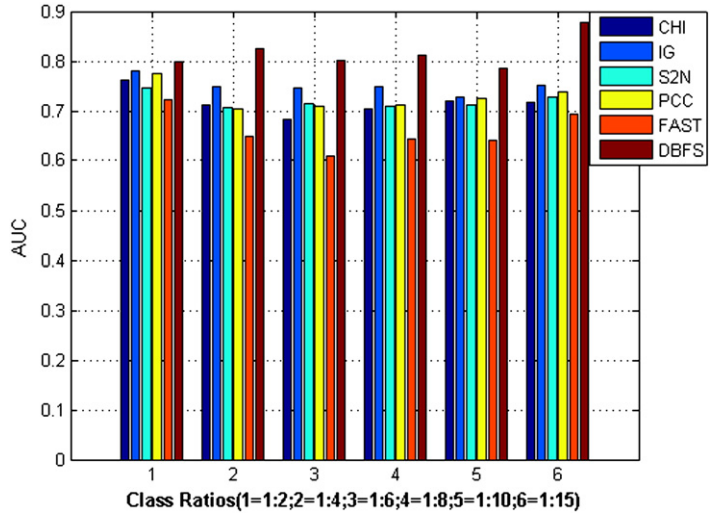

Fig. 49. Averaged AUC over all classifiers across different class ratios when 0.5% of features are selected.

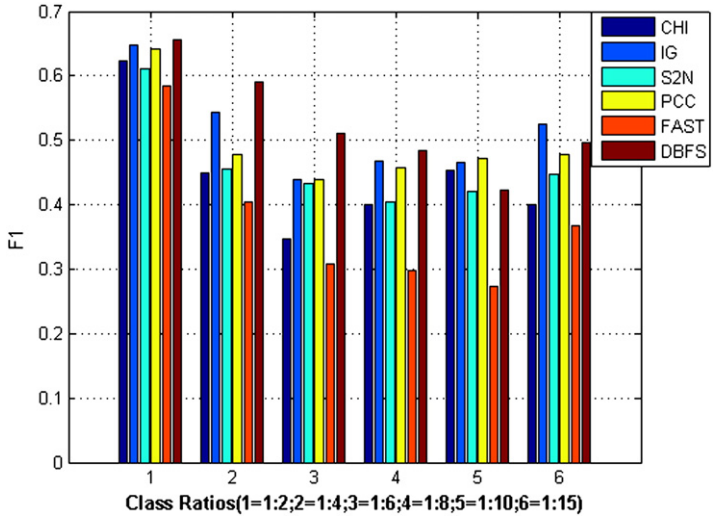

Fig. 50. Averaged F1 over all classifiers across different class ratios when 0.5% of features are selected.

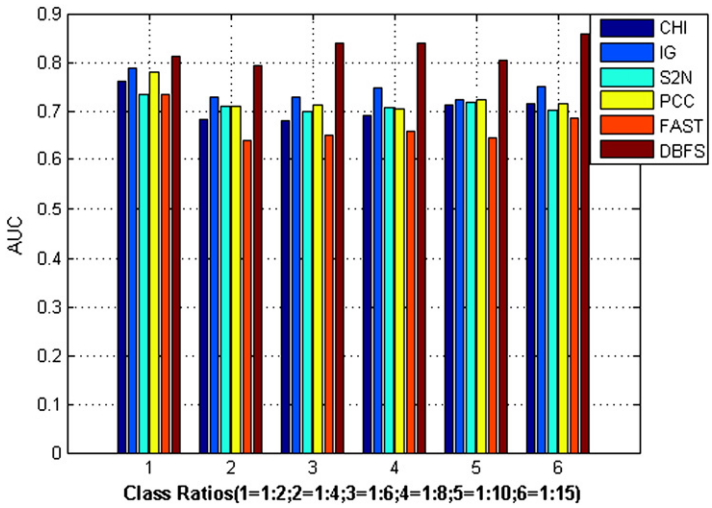

Fig. 51. Averaged AUC over all classifiers across different class ratios when 1% of features are selected.

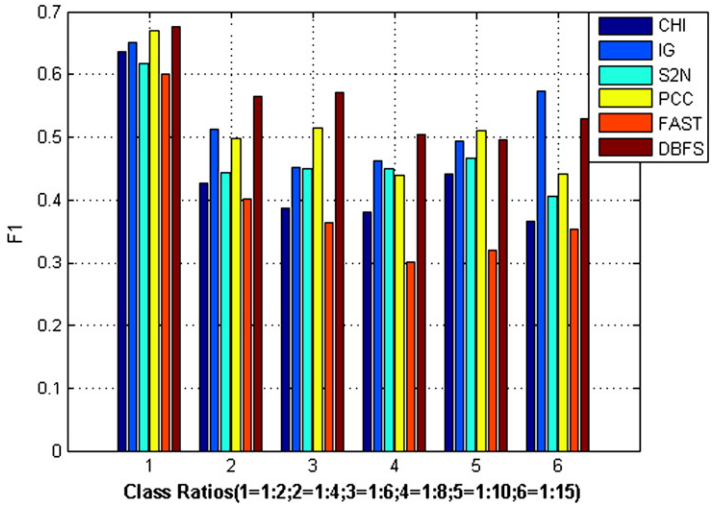

Fig. 52. Averaged F1 over all classifiers across different class ratios when 1% of features are selected.

5. Computational complexity analysis

As the final point, we aim to compare the order of computational complexity of rival feature ranking methods.

According to the formulas illustrated in Table 1, the order of calculating the rank of a desirable feature for IG, CHI, S2N and FAST feature ranking methods is $O(n)$ where n is the number of instances in a data set. In addition, if f shows a number of features in a data set, the order of total computational complexity for these feature ranking methods will be $O(n*f)$.

In order to compute the computational complexity of the DBFS method, it is notable that estimating the PDF for a fixed number (i.e. K) of equally spaced points in the instance space, gives us a good estimation for the underlying distribution of those instances. Taking this point into account, to compute the rank of a specific feature in the DBFS method, K equally spaced points in the instance space of each class are selected. If cl shows the number of classes in a data set, the PDF of each feature in each class will be estimated in $K*cl$ number of selected points. The computational complexity for estimating the PDF value for a specific instance is $O(n)$. So, the final computational complexity of estimating PDF for a specific feature in each class is $O(n*cl)$. The computational complexity of computing *DiscriminantAbility* and *numChanges* values is of the order $O(cl^2)$ (these values are computed by considering the estimated PDF values of the $K*cl$ selected points). Therefore, the final order of computational complexity of the proposed feature ranking method (DBFS) is $O(f*n*cl^2)$ where cl is the number of classes in a data set which is often very small and for most of the imbalanced data sets such as the ones studied in this paper, equals to 2. Thus, in terms of computational complexity, DBFS is nearly comparable to other rival feature ranking methods.

6. Conclusion and future work

In this study, we present a feature selection scheme to tackle the small sample size and high dimensional problem in imbalanced data sets. Dealing with imbalanced data sets, combination of imbalanced data and the small sample size presents a new challenge to the community [88,66]. Some of the different approaches used to tackle the class imbalance problem could make the problem with learning on a small data set even worse [14]. Since, the class imbalance problem is commonly accompanied by the issue of high dimensionality of the data set [14,24], applying feature selection approaches is a necessary step [14]. The motivation to introduce the presented scheme, DBFS, is based on the observation that ingenious sampling and algorithmic approaches may not be enough to combat the high dimensional class imbalance problem and in these cases, feature selection may alone combat the class imbalance problem [1,3–5,14,39]. The DBFS method is a novel feature ranking approach developed based on the probability density estimation of each feature in each class. The idea behind this approach is that the distribution of features over classes brings a significant benefit to feature selection algorithms to explore the merit of a feature.

To study the advantages of the DBFS method, it is compared to five well-known state of the art feature selection approaches across different well-known data sets from microarray, mass spectrometry and text mining domains over three of the most common classifiers with different biases i.e. Naïve Bayes (NB), Nearest Neighbor (1-NN) and linear SVM (LSVM) classifiers. To assess the performance of rival feature ranking methods, AUC and F1 evaluation statistics are used in this paper. Experiments are designed in such a way that investigates the ability of different feature ranking methods from five different perspectives. In the first perspective, the averaged results over all biological and text data sets show that the DBFS feature ranking method performs the best using AUC and F1 evaluation statistics across various percentages of selected features. Since, most data mining researches favor to find the feature ranking method that performs more closely to the optimal for a specific data set regardless of its average performance, the second perspective finds those feature ranking methods that are more likely to perform the best for a single data set. Experimental results illustrate that the closest feature ranking method to the optimal is DBFS across all classifiers and evaluation statistics. Furthermore, due to the inherent characteristics of data sets in each domain, different feature ranking methods are appropriate in each domain. Thus, investigations from the third perspective show that for the biological domain, DBFS and FAST perform the same when 0.1% of features are selected. With a greater percentage of selected features, DBFS is a better choice of action. Also, for the text mining domain, with only 0.1% of features being selected, DBFS and IG are the best performing feature ranking methods. By selecting a greater percentage of features, the DBFS method is the best feature ranking method across the NB classifier, followed by the PCC method. But using 1-NN and LSVM classifiers, IG would be the second ranked feature ranking method behind DBFS. Also, regardless of the classifier used, DBFS has the highest performance across AUC and F1 evaluation statistics over all data sets.

Moreover, we investigated how the performance of feature selection methods evolves when the imbalance ratio increases. The results indicate that across various class imbalance ratios, DBFS feature ranking method outperforms other rival feature selection methods especially when more than 0.5% of features are selected for the classification task. This improvement is more tangible according to AUC evaluation statistic especially with higher imbalance ratios.

As the final analysis, we concluded that the proposed feature ranking method performs much better than almost all rival feature ranking methods. Moreover, using a moderate percentage of selected features (0.5% or 1% of the original features) which are often desirable; our method achieves the highest performance compared to other rival methods which is statistically significant. Also, the order of computational complexity of DBFS is nearly comparable with other rival feature ranking methods.

As future work, we intend to investigate the performance of DBFS for handling multi-class problems, i.e. problems with more than two classes. Our preliminary results on these problems are promising. Also, it may be beneficial to consider the relations between features in a linear wrapper-based feature selection method, to achieve a higher performance and generality over classifiers. Furthermore, it is interesting to study how the proposed feature selection approach performs when combined with algorithmic approaches (classifiers focus on imbalanced data sets such as the ones proposed in [29–35,59]).

References

[1] X. Chen, M. Wasikowski, FAST: a roc-based feature selection metric for small samples and imbalanced data classification problems, in: Proceeding of the 14th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining, 2008, pp. 124–132.
[2] I. Guyon, J. Weston, S. Barnhill, V. Vapnik, Gene selection for cancer classification using support vector machines, Machine Learning 46 (1–3) (2002) 389–422.
[3] G. Forman, An extensive empirical study of feature selection metrics for text classification, Journal of Machine Learning Research 3 (2003) 1289–1305.
[4] D. Mladenic, M. Grobelnik, Feature selection for unbalanced class distribution and Naive Bayes, in: Proceedings of the 16th International Conference on Machine Learning, 1999, pp. 258–267.
[5] Z. Zheng, X. Wu, R. Srihari, Feature selection for text categorization on imbalanced data, SIGKDD Explorations 6 (2004) 80–89.
[6] D. Casasent, X. Chen, Feature reduction and morphological processing for hyperspectral image data, Applied Optics 43 (2) (2004) 1–10.
[7] T. Fawcett, F. Provost, Adaptive fraud detection, Data Mining and Knowledge Discovery 1 (1996) 291–316.
[8] P.K. Chan, S.J. Stolfo, Toward scalable learning with non-uniform class and cost distributions: a case study in credit card fraud detection, in: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining, 2001, pp. 164–168.
[9] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, in: Second edition, Wiley, 1997.
[10] T. Mitchell, Machine Learning, McGraw Hill, 1997.
[11] S. Visa, A. Ralescu, Issues in mining imbalanced data sets — a review paper, in: Proceedings of the 16th Midwest Artificial Intelligence and Cognitive Science Conference, 2005, pp. 67–73.
[12] N. Japkowicz, Learning from imbalanced data sets: a comparison of various strategies, Proceedings of Learning from Imbalanced Data (2000) 10–15.
[13] G. Weiss, Mining with rarity: a unifying framework, SIGKDD Explorations 6 (1) (2004) 7–19.
[14] M. Wasikowski, X. Chen, Combating the small sample class imbalance problem using feature selection, IEEE Transactions on Knowledge and Data Engineering (2009).
[15] N. Japkowicz, Proceedings of the AAAI'2000 Workshop on Learning from Imbalanced Data Sets. AAAI Tech Report WS-00-05, in: 2000.
[16] Proceedings of the ICML'2003 Workshop on Learning from Imbalanced Data Sets, in: N. Chawla, N. Japkowicz, A. Kolcz (Eds.), 2003.

[17] M. Kubat, S. Matwin, Learning when negative examples abound, in: Proceedings of the 9th European Conference on Machine Learning ECML, 97, 1997, pp. 146–153.
[18] N. Chawla, K. Bowyer, L. Hall, P. Kegelmeyer, SMOTE: Synthetic Minority Over-sampling Technique, Journal of Artificial Intelligence Research 16 (2002) 321–357.
[19] H. Han, W.-Y. Wang, B.-H. Mao, Borderline-SMOTE: a new oversampling method in imbalanced data sets learning, Advances in Intelligent Computing (2005) 878–887.
[20] T. Jo, N. Japkowicz, Class imbalances versus small disjuncts, SIGKDD Explorations vol. 6 (1) (2004) 40–49.
[21] M. Kubat, S. Matwin, Addressing the curse of imbalanced data set: one sided sampling, in: Proceedings of the 14th International Conference on Machine Learning, 1997, pp. 179–186.
[22] X. Chen, B. Gerlach, D. Casasent, Pruning support vectors for imbalanced data classification, Proceedings of International Joint Conference on Neural Networks (2005) 1883–1888.
[23] R. Barandela, R.M. Valdovinos, J.S. S'anchez, F.J. Ferri, The imbalanced training sample problem: under or over sampling, in: Joint IAPR International Workshops on Structural, Syntactic, and Statistical Pattern Recognition (SSPR/SPR04), 2004, pp. 806–814.
[24] N. Chawla, N. Japkowicz, A. Kotcz, Editorial: special issue on learning from imbalanced data sets, SIGKDD Explorations 6 (1) (2004) 1–6.
[25] A. Estabrooks, T. Jo, N. Japkowicz, A multiple resampling method for learning from imbalanced data sets, Computational Intelligence 20 (1) (2004) 18–36.
[26] A. An, N. Cercone, X. Huang, A case study for learning from imbalanced data sets, in: Proceedings of the 14th Biennial Conference of the Canadian Society on Computational Studies of intelligence: Advances in Artificial intelligence, vol. 2056, 2001, pp. 1–15.
[27] A. Raskutti, A. Kowalczyk, Extreme rebalancing for SVMs: a SVM study, SIGKDD Explorations 6 (1) (2004) 60–69.
[28] C. Elkan, K. Noto, Learning classifiers from only positive and unlabeled data, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2008, pp. 213–220.
[29] N. Chawla, A. Lazarevic, L. Hall, K. Bowyer, SMOTEBoost: improving prediction of the minority class in boosting, Principles of Knowledge Discovery in Databases LNAI 2838 (2003) 107–119.
[30] C. Chen, A. Liaw, L. Breiman, Using random forest to learn imbalanced data, University of California, Berkeley, Tech. Rep., 2004.
[31] R. Schapire, The strength of weak learnability, Machine Learning 5 (1990) 197–227.
[32] P. Domingos, MetaCost: a general method for making classifiers cost-sensitive, in: Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1999, pp. 155–164.
[33] C. Elkan, The foundations of cost-sensitive learning, In Proceedings of the 17th International Joint Conference on, Artificial Intelligence (2001) 973–978.
[34] K. Huang, H. Yang, I. King, M. Lyu, Learning classifiers from imbalanced data based on biased minimax probability machine, in: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2(27), 2004, pp. 558–563.
[35] K. Ting, The problem of small disjuncts: its remedy on decision trees, in: Proceedings of the 10th Canadian Conference on Artificial Intelligence, 1994, pp. 91–97.
[36] Filo, D., Yang, J., (1997), Yahoo! Inc.
[37] C. Elkan, Magical thinking in data mining: Lessons from CoIL challenge 2000, in: Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2001, pp. 426–431.
[38] H. Xiong, X. Chen, Kernel-based distance metric learning for microarray data classification, BMC Bioinformatics 7 (1) (2006).
[39] P. Van der Putten, M. van Someren, A bias-variance analysis of a real world learning problem: the coil challenge 2000, Machine Learning 57 (1–2) (2004) 177–195.
[40] H. Liu, J. Sun, L. Liu, H. Zhang, Feature selection with dynamic mutual information, Pattern Recognition 42 (2009) 1330–1339.
[41] G.H. John, R. Kohavi, K. Pfleger, Irrelevant feature and the subset selection problem, in: Proceedings Of the 11th International Conference on Machine Learning ICML, 94, 1994, pp. 121–129.
[42] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, Journal of Machine Learning Research, special Issue on variable and Feature Selection 3 (2003) 1157–1182.
[43] L. Yu, H. Liu, Efficient feature selection via analysis of relevance and redundancy, Journal of Machine Learning Research 5 (2004) 1205–1224.
[44] L. Yu, H. Liu, Feature selection for high-dimensional data: a fast correlationbased filter approach, in: Proceedings of the International Conference on Machine Learning, 2003.
[45] M.A. Hall, Correlation-based feature subset selection for machine learning, Ph.D. Dissertation, Department of Computer Science, University of Waikato, Hamilton, New Zealand (1999).
[46] K. Kira, L. Rendell, The feature selection problem: traditional methods and new algorithm, in: Proceedings of the 9th International Conference on Machine Learning, 1992, pp. 249–256.
[47] K. Fukunaga, Introduction to Statistical Pattern Recognition, in: 2nd ed., Academic Press, 1990.
[48] C.M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
[49] A.R. Webb, Statistical Pattern Recognition, Second edition Wiely, 2002.
[50] S. Pomeroy, P. Tamayo, M. Gaasenbeek, L. Sturla, M. Angelo, M. McLaughlin, J. Kim, L. Goumnerova, P. Black, C. Lau, J. Allen, D. Zagzag, J. Olson, T. Curran, C. Wetmore, J. Biegel, T. Poggio, S. Mukherjee, R. Rifkin, A. Califano, G. Stolovitzky, D. Louis, J. Mesirov, E. Lander, T. Golub, Prediction of central nervous system embryonal tumour outcome based on gene expression, Nature (2002) 436–442.
[51] 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. Bloomþeld, E.S. Lander, Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, Science 286 (1999) 531–537.
[52] M.A. Shipp, K.N. Ross, P. Tamayo, A.P. Weng, J.L. Kutok, R.C. Aguiar, M. Gaasenbeek, M. Angelo, M. Reich, G.S. Pinkus, T.S. Ray, M.A. Koval, K.W. Last, A. Norton, T.A. Lister, J. Mesirov, D.S. Neuberg, E.S. Lander, J.C. Aster, T.R. Golub, Diffuse large b-cell lymphoma outcome prediction by gene expression profiling and supervised machine learning, Nature Medicine 8 (2002) 68–74.
[53] E.F. Petricoin, A.M. Ardekani, B.A. Hitt, P.J. Levine, V.A. Fusaro, S.M. Steinberg, G.B. Mills, C. Simone, D.A. Fishman, E.C. Kohn, L.A. Liotta, Use of proteomic patterns in serum to identify ovarian cancer, The Lancet 359 (2002) 572–577.
[54] A. Asuncion, D.J. Newman, UCI Machine Learning Repository, University of California, Department of Information and Computer Science, Irvine, CA, 2007. http://www.ics.uci.edu/~mlearn/MLRepository.html.
[55] E.F. Petricoin III, D.K. Ornstein, C.P. Paweletz, A. Ardekani, P.S. Hackett, B.A. Hitt, A. Velassco, C. Trucco, L. Wiegand, K. Wood, C.B. Simone, P.J. Levine, W.M. Linehan, M.R. Emmert-Buck, S.M. Steinberg, E.C. Kohn, L.A. Liotta, Serum proteomic patterns for detection of prostate cancer, Journal of the National Cancer Institute 94 (20) (2002) 1576–1578.
[56] http://cilab.ujn.edu.cn/dataset/lung_cancer.rar.
[57] www.cs.nyu.edu/~roweis/data.html.
[58] 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, E.S. Lander, Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, Science 286 (1999) 531–537.
[59] H.N. Qu, G.Z. Li, W.S. Xu, An asymmetric classifier based on partial least squares, Pattern Recognition 43 (10) (2010) 3448–3457.
[60] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.
[61] F. Wilcoxon, Individual comparisons by ranking methods, Biometrics 1 (1945) 80–83.
[62] H. He, Y. Bai, E.A. Garcia, S. Li, ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning, IEEE Joint Conference on Neural Networks (2008) 1322–1328.
[63] D. Mease, A.J. Wyner, A. Buja, Boosted classification trees and class probability/quartile estimation, Journal of Machine Learning Research 8 (2007) 409–439.
[64] B.X. Wang, N. Japkowicz, Boosting support vector machines for imbalanced data sets, Knowledge and Information Systems 25 (1) (2010) 1–20.
[65] Y.M. Sun, M.S. Kamel, A.K.C. Wong, Cost-sensitive boosting for classification of unbalanced data, Pattern Recognition 40 (12) (2007) 3358–3378.
[66] H. He, E. Garcia, Learning from imbalanced data, IEEE Transactions on Data and Knowledge Engineering 21 (9) (2009) 1263–1284.
[67] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 99 (2011) 1–22.
[68] G. Batista, R. Prati, M. Monard, Balancing strategies and class overlapping, in: International Symposium on Intelligent Data Analysis, Springer, 2005, pp. 24–35.
[69] V. Garcia, J.S. Sanchez, R.A. Mollineda, An empirical study of the behaviour of classifiers on imbalanced and overlapped data sets, in: Proceedings of the Congress on Pattern Recognition 12th Iberoamerican Conference on Progress in Pattern Recognition, Image Analysis and Applications (CIARP'07), Springer-Verlag, 2007, pp. 397–406.

[70] K. Napierala, J. Stefanowski, S. Wilk, Learning from imbalanced data in presence of noisy and borderline examples, in: Proceedings of the 7th international conference on Rough sets and current trends in computing (RSCTC'10), Springer-Verlag, 2010, pp. 158–167.
[71] S. Hu, Y. Liang, L. Ma, Y. He, MSMOTE: improving classification performance when training data is imbalanced, in: Proceeding 2nd International Workshop Computer Science Engineering, vol. 2, 2009, pp. 13–17.
[72] J. Laurikkala, Improving identification of difficult small classes by balancing class distribution, in: Proceeding of Conference AI in Medicine in Europe: Artificial Intelligence Medicine, 2001, pp. 63–66.
[73] G.E.A.P.A. Batista, R.C. Prati, M.C. Monard, A study of the behavior of several methods for balancing machine learning training data, ACM SIGKDD Explorations Newsletter 6 (1) (2004) 20–29.
[74] J. Stefanowski, S. Wilk, Selective pre-processing of imbalanced data for improving classification performance, in: Data Warehousing and Knowledge Discovery (Lecture Notes in Computer Science Series 5182), 2008, pp. 283–292.
[75] L.M. Manevitz, M. Yousef, One-class SVMs for document classification, Journal of Machine Learning Research 2 (2001) 139–154.
[76] H.J. Lee, S. Cho, The novelty detection approach for difference degrees of class imbalance, Lecture Notes in Computer Science 4233 (2006) 21–30.
[77] L. Manevitz, M. Yousef, One-class document classification via neural networks, Neurocomputing 70 (2007) 1466–1481.
[78] N. Japkowicz, Supervised versus unsupervised binary-learning by feedforward neural networks, Machine Learning 42 (2001) 97–122.
[79] L. Breiman, Bagging predictors, Machine Learning 24 (1996) 123–140.
[80] C. Seiffert, T. Khoshgoftaar, J. Van Hulse, A. Napolitano, Rusboost: a hybrid approach to alleviating class imbalance, IEEE Transaction on Systems, Man, and Cybernetics, Part A: Systems and Humans 40 (1) (2010) 185–197.
[81] S. Wang, X. Yao, Diversity analysis on imbalanced data sets by using ensemble models, in: IEEE Symposium on Computational Intelligence and Data Mining, 2009, pp. 324–331.
[82] J. Błaszczyński, M. Deckert, J. Stefanowski, S. Wilk, Integrating selective pre-processing of imbalanced data with Ivotes ensemble, in: Rough Sets and Current Trends in Computing (Lecture Notes in Computer Science Series 6086), Springer-Verlag, 2010, pp. 148–157.
[83] W. Fan, S.J. Stolfo, J. Zhang, P.K. Chan, Adacost: misclassification cost-sensitive boosting, in: 6th International Conference of Machine Learning, 1999, pp. 97–105.
[84] K.M. Ting, A comparative study of cost-sensitive boosting algorithms, in: Proceeding of 17th International Conference of Machine Learning, 2000, pp. 983–990.
[85] M. Joshi, V. Kumar, R. Agarwal, Evaluating boosting algorithms to classify rare classes: comparison and improvements, in: Proceeding IEEE International Conference on Data Mining, 2001, pp. 257–264.
[86] S.J. Raudys, A.K. Jain, Small sample size effects in statistical pattern recognition: recommendations for practitioners, IEEE Transaction on Pattern Analysis and Machine Intelligence 13 (3) (1991) 252–264.
[87] W.H. Yang, D.Q. Dai, H. Yan, Feature extraction uncorrelated discriminant analysis for high-dimensional data, IEEE Transaction on Knowledge and Data Engineering 20 (5) (2008) 601–614.
[88] R. Caruana, Learning from imbalanced data: rank metrics and extra tasks, in: Proceeding of American Association for Artificial Intelligence (AAAI) Conference, AAAI Technical Report WS-00-05, 2000, pp. 51–57.
[89] H. Jing, B. Wang, Y. Yang, Y. Xu, A general framework of feature selection for text categorization, in: Proceedings of MLDM 2009, 5632, Springer LNAI, 2009, pp. 647–662.
[90] J.V. Hulse, T. Khoshgoftaar, Knowledge discovery from imbalanced and noisy data, Data & Knowledge Engineering 68 (12) (2009) 1513–1542.
[91] T.M. Khoshgoftaar, K. Gao, J.V. Hulse, Feature selection for highly imbalanced software measurement data, recent trends in information reuse and integration, in: Tansel Ozyer, et al., (Eds.), Springer-Verlag/Wien, 2012, pp. 167–189.

Mina Alibeigi received her B.Sc. and M.Sc. degrees in computer engineering from Shiraz University in 2008 and 2010 respectively. She is currently a PhD student in Artificial Intelligence and Robotics at Tehran University. Her research interests include imbalanced data sets, dimension reduction, clustering, bio-inspired learning algorithms, cognitive science, robotics and imitation learning.

Sattar Hashemi received the PhD degree in computer engineering from Iran University of Science and Technology in conjunction with Monash University, Australia, in 2008. Following academic appointments at Shiraz University, he is currently the head of computer department at electrical and computer engineering school, Shiraz University, Shiraz, Iran. He is recognized for contributions in the fields of machine learning and data mining. He has published many refereed papers and book chapters on data stream classification, social networks, database intrusion detection, and computer security.

Ali Hamzeh received his Ph.D. degree in artificial intelligence from Iran University of Science and Technology, 2007. He is currently an assistant professor of artificial intelligence in the Electrical and Computer Engineering School, Shiraz University, Shiraz, Iran. His research interests include recommender systems, social networks, evolutionary algorithms, game theory and computer security. He has published many refereed papers and book chapters on these fields.