Knowledge-Based Systems 67 (2014) 105–116

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

ForesTexter: An efficient random forest algorithm for imbalanced text categorization

Qingyao Wu [a,*], Yunming Ye [a], Haijun Zhang [a], Michael K. Ng [b], Shen-Shyang Ho [c]

[a] Shenzhen Key Laboratory of Internet Information Collaboration, Shenzhen Graduate School, Harbin Institute of Technology, Rm.202, C# Building, HIT Campus at Xili University Town, Shenzhen 518055, PR China
[b] Department of Mathematics, Hong Kong Baptist University, Fong Shu Chuen Building FSC 1212, Kowloon Tong, Hong Kong
[c] School of Computer Engineering, Nanyang Technological University, C2I, Blk N4, #B1a-02, Nanyang Avenue, Singapore 639798, Singapore

ARTICLE INFO

Article history:
Received 20 July 2013
Received in revised form 4 April 2014
Accepted 8 June 2014
Available online 19 June 2014

Keywords:
Text categorization
Imbalanced classification
Random forests
SVM
Stratified sampling

ABSTRACT

In this paper, we propose a new random forest (RF) based ensemble method, ForesTexter, to solve the imbalanced text categorization problems. RF has shown great success in many real-world applications. However, the problem of learning from text data with class imbalance is a relatively new challenge that needs to be addressed. A RF algorithm tends to use a simple random sampling of features in building their decision trees. As a result, it selects many subspaces that contain few, if any, informative features for the minority class. Furthermore, the Gini measure for data splitting is considered to be skew sensitive and bias towards the majority class. Due to the inherent complex characteristics of imbalanced text datasets, learning RF from such data requires new approaches to overcome challenges related to feature subspace selection and cut-point choice while performing node splitting. To this end, we propose a new tree induction method that selects splits, both feature subspace selection and splitting criterion, for RF on imbalanced text data. The key idea is to stratify features into two groups and to generate effective term weighting for the features. One group contains positive features for the minority class and the other one contains the negative features for the majority class. Then, for feature subspace selection, we effectively select features from each group based on the term weights. The advantage of our approach is that each subspace contains adequate informative features for both minority and majority classes. One difference between our proposed tree induction method and the classical RF method is that our method uses Support Vector Machines (SVM) classifier to split the training data into smaller and more balance subsets at each tree node, and then successively retrains the SVM classifiers on the data partitions to refine the model while moving down the tree. In this way, we force the classifiers to learn from refined feature subspaces and data subsets to fit the imbalanced data better. Hence, the tree model becomes more robust for text categorization task with imbalanced dataset. Experimental results on various benchmark imbalanced text datasets (Reuters-21578, Ohsumed, and imbalanced 20 newsgroup) consistently demonstrate the effectiveness of our proposed ForesTexter method. The performance of our proposed approach is competitive against the standard random forest and different variants of SVM algorithms.

1. Introduction

Text categorization or classification is the task to automatically classify text documents into predefined categories [1]. Text categorization has many applications such as spam filtering [2], news organization [3], and user profiling analysis [4]. Given a set of labeled training documents, the system uses this training set to build a classifier. The inferred model can then be used to classify new text documents. A variety of machine learning classifiers have been applied to this task, including conventional classification methods such as *k*-Nearest Neighbors (*k*NN) [5] and Support Vector Machines (SVM) [6], and ensemble methods such as boosting-based system (boostexter) [7] and various combinations of Naive Bayes (NB) [8].

Ensemble classification is an active area of research in machine learning and data mining. An ensemble of classifiers is a set of base classifiers whose individual decisions are combined in some way to classify new examples. There are a great number of researches on ensemble classifier design. One of the most popular approaches for

* Corresponding author. Tel./fax: +86 2603 3008.
E-mail addresses: wuqingyao.china@gmail.com (Q. Wu), yym@hitsz.edu.cn (Y. Ye), hjzhang@hitsz.edu.cn (H. Zhang), mng@math.hkbu.edu.hk (M.K. Ng), ssho@ntu.edu.sg (S.-S. Ho).

http://dx.doi.org/10.1016/j.knosys.2014.06.004
0950-7051/

constructing classifier ensembles is the random forest method [9]. A random forest (RF) is an ensemble classifier based on a set of decision tree models trained on randomly selected data subsets and feature subsets. Each decision tree is built using a bootstrap sample of the data (data subset selection). Node splitting is based on the best feature among a set of randomly chosen features (feature subspace selection). Many studies, both theoretical and empirical, demonstrate the advantages of the RF method [10–13]. In fact, it has been found to be accurate and computationally feasible across various data domains [14–17].

Although existing classifier learning and ensemble learning techniques have shown great success in many text categorization applications, the problem of learning from imbalanced text data is a relatively new challenge that has attracted growing attention from both academia and industry [18–21]. This increased interest is reflected in the recent installment of several major conferences, workshops, and special issues [22–24]. The class imbalance problem occurs when the class of interest (positive or minority class) is relatively rare in the dataset as compared to the negative or majority class. The fundamental issue with this problem is the risk of imbalanced data to significantly compromise the performance of most conventional learning algorithms. These algorithms assume or expect balanced class distributions or equal misclassification costs. Therefore, when these algorithms are presented with complex imbalanced data sets, they fail to properly model the characteristics of the data and affect the classification accuracies across the classes of the data. In real-world domains, the ratio of the minority class to the majority classes can by drastic, say 1 to 100, 1 to 1000 or even 1 to 10,000 [20]. For such cases, it is found that the conventional classifiers tend to ignore the minority instances and simply classify all instances as the majority class. As a result, the majority class has close to 100% accuracy while the minority class has 0–10% accuracy [20].

A recent comparative study [25] shows that the direct application of ensemble learning algorithms do not solve the class imbalanced problem. Due to the inherent complex characteristics of imbalanced datasets, learning from such data requires new strategies and technologies to build good base classifiers that achieve a balanced degree of predictive accuracy for both the minority and majority classes on the imbalanced text dataset.

In this paper, we propose the ForesTexter algorithm for learning a RF ensemble for imbalanced text categorization. In particular, we propose a novel stratified sampling method for feature subspace selection and construct a splitting classifier for data partition at each node of the tree models in the RF ensemble. Compared to the simple random sampling method, the stratified sampling method ensures the features selected in the subspace are more balance and informative for both the minority and the majority classes. The conventional tree node splitting criteria, such as information gain and Gini measure, are considered to be skew sensitive to class imbalance and lead to complex and deeply nested tree models [26] which are inefficient and ineffective for applications with imbalanced dataset. In our proposed approach, a supervised learning technique is used to learn the splits at the nodes of the tree. In particular, the Support Vector Machine (SVM) classifier is used in this study for reasons we would explained in Section 3.1. The SVM successively partitions the training data into smaller subsets with more balanced class distributions till the data cannot be further partitioned. In this way, we force the model to learn from the more effective feature subspaces and data subsets. The model becomes more robust and fits the class imbalance data better. Experimental results on benchmark datasets (Reuters-21578, Ohsumed, and 20 newsgroup) show that our proposed ForesTexter algorithm is effective for the imbalanced text categorization problem. The performance of the ForesTexter algorithm is competitive against conventional RF and different variants of SVM.

The rest of the paper is organized as follows. In Section 2, we briefly review the related work. In Section 3, we describe our proposed ForesTexter algorithm. In Sections 4 and 5, we describe the experimental setup and present the results on three imbalanced text datasets. Finally, we discuss the experimental results and conclude with possible future work in Sections 6 and 7.

2. Related work

2.1. The difficulty with imbalanced data classification

A dataset is considered to be imbalanced if the number of instances in one class greatly outnumbers the number of instances in the other class. In this paper, we focus on the classification task on two-class imbalanced dataset. The class with the smaller number of instances is usually the class of interest (positive or minority class) from the viewpoint of the learning task, and the class with the larger number of instances is the negative or majority class. For example, in a disease diagnostic problem where the disease cases (minority class) are usually quit rare as compared with normal ones (majority class).

Most standard learning algorithms are based on the underlying assumptions of balanced data distributions among classes or equal misclassification costs. When the data fits the model well, the classification performance can be very high. However, these algorithms cannot handle the imbalanced datasets. As a result, they provide severely imbalanced degree of prediction accuracies for the different classes. In fact, it has been found that the data imbalance issue does not hinder the performance of learning algorithms by itself. There are other challenges that follow from the general data imbalance issues such as feature space heterogeneity, class overlapping, and small disjuncts (i.e., disjoint data sets from the same class) [20]. These difficulties together with imbalanced data further hamper the effectiveness of training a conventional classifier model for imbalanced data.

1. *Feature space heterogeneity:* Generally imbalanced datasets have uneven distributions of features for minority and majority classes, where minority class features are very limited. In [27], it is reported that the values of features relevant to the minority class are not comparable with those of the features to majority class. Since most widely used classification algorithms, such as the naive Bayes and decision tree, rely on measures computed over the entire feature set to learn classifier, the resulting classifier models are inevitably affected by the complex nature of the feature distributions.
2. *Class overlapping:* Serious class overlapping often leads to poor classification performance [28]. Consider the data distribution shown in Fig. 1, where the "+" and "−" represent the positive and negative instances of the minority and majority classes,

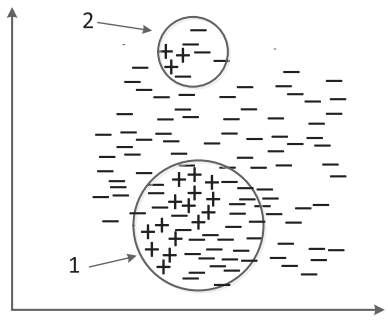

Fig. 1. An example of imbalanced dataset with class overlapping and disjoint sets of data from the same class (so-called small disjuncts).

respectively. Positive and negative instances are within the circular regions 1 and 2. As a result, it is likely that the positive instances will be misclassified as negative class. This issue is often observed in imbalanced data classification tasks where limited minority class instances are dispersed in different region of the feature spaces.

3. *Small disjuncts:* The problem of small disjuncts has been shown to significantly depreciate classification performance [29] when the minority class has multiple subconcepts. Fig. 1 shows a within-class imbalanced minority class with two subconcepts (regions 1 and 2). In this case, classifiers have to learn both the subconcepts of the minority class. Unlabeled instances in region 2 are misclassified more often than those from the region 1 because region 2 has a relative small number of instances.

In order to deal with imbalanced data, many techniques have been developed. Typically, the techniques to handle the class imbalance problem can be grouped into two classes: data level methods and algorithmic level methods [30,20,25]. For data level methods, the solutions involve a preprocessing step to re-balance the data distribution by under-sampling the majority class [31] or by over-sampling the minority one [32,33] in order to decrease the effect of the skewed class distribution in the dataset. For algorithmic level methods, the solutions extend and customize existing machine learning algorithms for class imbalance problem by shifting the bias of a classifier to favor the minority class, such as cost-sensitive learning [34,35] and recognition-based learning [36,37].

2.2. Random forest

The random forest (RF) is an ensemble classifier consisting of a collection of tree-structured base classifiers. The concept of building multiple decision trees was first introduced by Williams [38]. Ho [39] then developed the idea of using a subspace of features for each tree construction during the ensemble process. Dietterich [40] proposed a random splits strategy for tree construction. Subsequently, Breiman [9] formulated the RF algorithm, which introduces further randomness into the ensemble process. This process is accomplished by using bootstrap sample of the data subsets and random sample of feature subspaces to build multiple decision trees. The RF algorithm associated with a set of training documents D and N_f features can be described as follows:

1. Use bagging to generate K subsets $\{D_1, D_2, \ldots, D_K\}$ by randomly sampling D with replacement.
2. For each data set D_k, one builds a decision tree model. During decision tree construction, randomly sample a subspace of *mtry* dimension ($mtry \ll N_f$) from the available features present in the training documents at each node. Compute all possible splits based on the *mtry* features. The data partitions from the best split (e.g., the largest Gini measure) are used to generate child nodes. Repeat this process until a stopping criterion is met: e.g., the local learning documents are homogeneous with respect to the class or the number of documents is less than n_{min}.
3. Combine the K unpruned trees $h_1(X_1), h_2(X_2), \ldots, h_K(X_K)$ into a random forest ensemble, and use the majority votes among the trees for classification decision.

As stated in the introduction, the RF ensemble method is known to increase the accuracy of single decision tree classifiers by returning a classification decision based on the decisions from all the decision trees. However, directly applying the RF model to imbalanced datasets does not solve the class imbalance problem in the decision tree classifiers by themselves. The challenging issue is that the random feature subspace selection and the corresponding Gini splitting criterion involved in splitting a tree node are considered to be highly skew sensitive to class imbalance.

- Random feature subspace selection: Text data are usually high dimensional. A large number of features (negative features) are often not informative to the minority class of interest. Only a few positive features are relevant to the minority class [41]. RF algorithm tends to use a simple random sampling of features and to select many subspaces that contain few information positive features in building their decision trees. In Fig. 2a, one observes that the chance that an informative positive feature is randomly included into a feature subspace decreases as the number of the total features increases. In Fig. 2b, one observes that the performance of RF first increases and then decreases with increasing number of total features. One plausible reason for these observations is that the lack of informative features in the selected subspaces affects the RF performance.
- Gini splitting criterion: One way of representing text data utilizes term frequencies and document frequencies. As it turns out, the values of positive features are not comparable with the negative features due to the imbalanced document frequencies for majority and minority classes. Decision tree models, such as CART (Classification and Regression Tree) and C4.5 (successor of the ID3 (Iterative Dichotomiser 3) decision tree algorithm), using either information gain or Gini measure splitting criterion, require discrete inputs (using static discretization for continuous features). Moreover, it requires the feature space to be divided into axis-parallel rectangles. When there is class imbalance in the dataset, the Gini measure has a strong bias towards features of majority class. The resulting tree topology and its corresponding space partitions are far more complex than class balanced datasets. One observes that a balanced and well-separated labeled data has simple decision tree boundaries in Fig. 3a. While on the dataset with a more skewed distribution (as shown in Fig. 3b), decision tree has a relative complex decision tree boundary. This type of boundaries (axis-parallel to the axes) usually leads to highly nested decision trees. On the other hand, the dashed lines boundaries of SVM (oblique orientations to the axes) are more simple and more effective.

2.3. Support Vector Machine

Support Vector Machine (SVM) finds the optimal margin hyperplane as the decision boundary that best separates a binary-class dataset based on the structural risk minimization principle [42]. Given a training dataset $\{(x_i, y_i)\}_{i=1}^{M}$ where $x_i \in R^N$ and $y_i \in \{-1, 1\}$, one minimizes the following function:

$$g(w, \xi) = \frac{1}{2}\|w\|^2 + C\sum_i \xi_i \tag{1}$$

subject to:

$$y_i((w \cdot \phi(x_i)) + b) \geqslant 1 - \xi_i$$

where C is the regularization coefficient and ξ_i, $i = 1, \cdots, M$, are slack variables to measure the degree of misclassification of x_i. Reference [43] gives an elementary introduction to the SVM method.

SVM has been shown to be an accurate classifier model for text categorization under balance distribution. Joachims [6,44,45] provides both theoretical and empirical evidences that SVM is very suitable for text categorization. He compared SVM with other classification methods and showed that SVM is competitive against the other tested classifiers in the experiments. However, when the class distribution is too imbalanced, the SVM performance may bias to the majority class. In Eq. (1), minimizing the first term

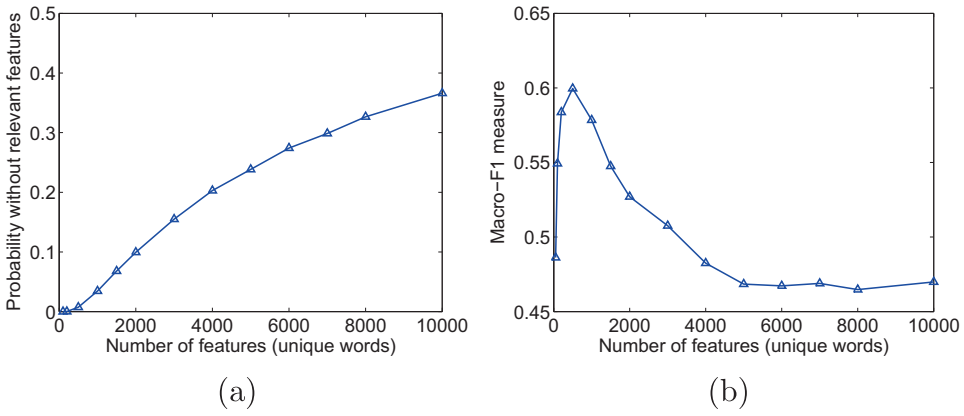

Fig. 2. (a) The probability of subspaces without relevant features against the total number of features N_f. (b) The performance of RF on Reuters-21578 against the total number of features N_f. In the example, we set the number of relevant features $R = 100$, the subspace size $mtry = \sqrt{N_f}$. The probability of selecting a subspace of $mtry$ features without relevant features is around $P \approx (1 - \frac{R}{N_f})^{mtry}$. The performance of RF decreases as the probability of subspaces without relevant features increases.

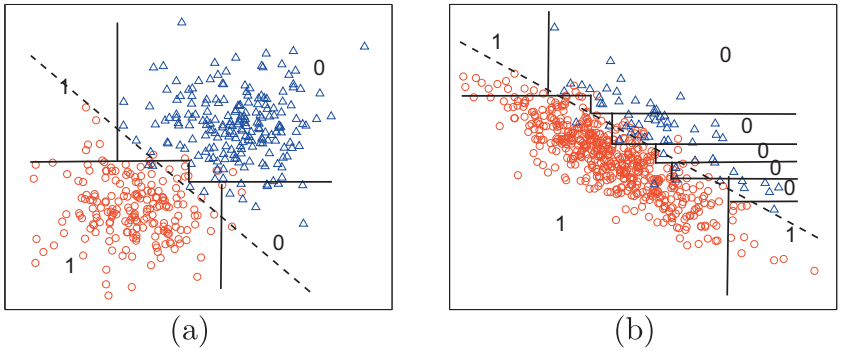

Fig. 3. An example using a synthetic dataset to show the decision boundaries for a two-class classification problem (classes 0 and 1). The solid lines and dashed lines are boundaries of decision tree and SVM methods respectively. (a) The decision boundaries on a relatively well-separated balanced dataset. (b) The decision boundaries on the dataset with a more skewed distribution.

$\|w\|^2/2$ is equivalent to maximizing the margin, while minimizing the second term $\sum_i \xi_i$ amounts to minimizing the associated error. The parameter C is to establish a balance between maximizing the margin and minimizing the error. This approach works well in the class balance datasets, but usually fails in situations with a high degree of class imbalance. The SVM classifier tends to simply classify all the instances as majority class because it results in largest margin and only suffers a small amount of cumulative error on the instances of minority class. This results in ineffectiveness of SVM in imbalance text categorization. There have been many works in the community that apply general sampling techniques to the SVM framework to handle imbalanced datasets [46–49].

3. ForesTexter

In this section, we first provide an overview of our proposed learning method for imbalanced data. Then, we discuss the positive and negative features used for feature subspace selection in RF. After that, we introduce a stratified sampling method. Finally, we describe in detail the new random forest algorithm, the ForesTexter, for imbalanced text categorization.

3.1. Hybrid method for learning from imbalanced data

We propose a novel and yet simple method to deal with the problem of model misfit in class imbalance learning. Our proposed approach is motivated by the tree model in RF and the SVM model for text categorization. It is a hybrid classification approach that combines the SVM and tree structured recursive partition techniques. A SVM offers two main advantages for text classification. The main advantage of using the SVM classifier, as previously pointed out, is that the SVM decision boundary is usually more simple and straightforward compared with those of the boundary of decision tree. Also, it has been shown that SVM can scale up to considerable dimensionality. It is effective for data splitting in the presence of very large number of features (or feature subspace). For high dimensional text classification problems, it is a relatively robust and flexible model compared to other methods, such as neural networks [50,6]. Although the SVM model does not fit the imbalance data well, tree model can be used together with SVM to improve the performance of the SVM classifier for imbalanced text categorization.

The resulting SVM-based tree model is a new tree induction algorithm and is very effective for imbalanced text categorization. A SVM-based tree model builds a hierarchal tree for classification according to the classical top-down procedure. To construct the tree, one follows a successive splitting procedure. It begins with the training of a SVM classifier on the entire dataset at the root node. Then, the induced SVM classifier divides the training data into subsets (or sub-regions) for prediction. However, model misfit generally leads to poor prediction performance. In order to improve the performance, we retrain SVM sub-classifiers using the training instances in the sub-regions. The retraining procedure consists of two steps: *feature subspace selection* and *splitting choice*. The *feature subspace selection* employs a stratified sampling

strategy in which the features for majority and minority are selected in a balanced way. The *splitting choice* trains a SVM classifier based on the selected feature subspace and the data subset at the node for further data partitioning. Instances are classified to the child node that maximizes the estimated relevance. The training dataset is recursively partitioned into smaller subsets while moving down the tree. This process continues until the remaining instances at the node cannot be further split by the classifiers. In this way, we force the classifier to learn from feature subspaces and data sub-regions that are more balance with respect to the data class distribution, making the entire classification model more robust for handling the feature space heterogeneity, class overlapping, and small disjuncts problems in imbalanced data.

In Fig. 4, one observes that a SVM-based tree partitions the data into more (class) homogeneous subsets. After building a classifier, one obtains decision boundaries shown with dashed circles. There is a decrease in class imbalance. Then, we build two sub-classifiers to further partition the instances. The solid circles are the new decision boundaries. We perform this process recursively to refine the classification boundaries and improve its performance for the imbalanced data. The resulting decision boundaries of the SVM-based tree approach are more effective compared to conventional decision tree method for imbalanced text categorization. To boost the predicting performance further, we extend the SVM-based tree model using an ensemble approach utilizing the random forest framework. In general, the RF ensemble framework is considered more robust than a single tree model. One advantage of the RF framework is that all the tree models can be trained in parallel. Meanwhile, RF blends elements of random subspaces and bagging. Due to the subspace sampling technique, RF generates trees very rapidly. It has also been found that RF is able to offer an increase in accuracy over bagging and boosting decision trees in learning from high dimensional data [51].

3.2. Positive and negative features

In a text categorization task, the documents are represented by a bag-of-words model. In other words, the document is represented by a vector, of dimension the size of the vocabulary, containing word frequency counts. Vocabularies of hundreds of thousands of words are common, hence the document vectors are usually of high dimensions. Feature selection has been applied to the text categorization task to improve classification effectiveness and computational efficiency. A number of feature selection metrics have been explored in text categorization, such as information gain, mutual information, and chi-square (χ^2). Empirical studies have shown that these metrics are useful for improving the performance of k-nearest neighbors (kNN), linear least squares fit (LLSF) and naive Bayes (NB) [5]. Extensive surveys and comparative studies of feature selection and term weighting methods for text categorization are found in [5,52–54].

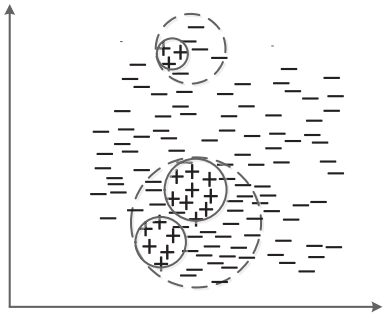

Fig. 4. An example of classification predictions based on the hybrid method of random forest and SVM.

Our approach models the feature subspace selection in RF as a feature selection and term weighting problem. We can make use of the idea in positive and negative features and stratified sampling to solve the subspace selection problem.

Let $\mathbf{T}$ be a set of N_f features (or terms) $\{t_1, t_2, \ldots, t_{N_f}\}$, and c be the class of interest. We first compute the following four dependency tuples (A_1, A_2, A_3, A_4) to examine the relationship between features and classes (see Table 1). A_1 denotes the number of times a feature t_i and a class c co-occur. A_4 denotes the number of times that neither t_i nor c appears in a document. These two elements represent the positive membership of t_i in c. On the other hand, A_2 and A_3 denote their negative membership (non-membership) that t_i occurs without the occurrence of category c, and category c occurs without the occurrence of feature t_i, respectively. Then, we perform a Boolean test to stratify the feature set $\mathbf{T}$ into two groups, one group containing positive features and the other one containing negative features, as follows:

$$\begin{cases} sign(t_i, c) = \text{pos}, & \text{if } A_1A_4 - A_2A_3 \geqslant 0, \\ sign(t_i, c) = \text{neg}, & \text{otherwise}. \end{cases}$$

where t_i is a feature term taken from $\mathbf{T}$, and $sign(t_i, c)$ is a function of (A_1, A_2, A_3, A_4) to determine whether there is any correlation between feature t_i and class c. Hence, it establishes whether t_i is a positive (pos) or a negative (neg) feature.

Based on the $sign(t_i, c)$ function, we then stratify the feature set $\mathbf{T}$ into two groups: $\mathbf{T}_p$ and $\mathbf{T}_n$, as follows:

$$\mathbf{T}_p = \{t_i \in \mathbf{T} | sign(t_i, c) \text{ is positive}\}$$

and

$$\mathbf{T}_n = \{t_i \in \mathbf{T} | sign(t_i, c) \text{ is negative}\}.$$

where $\mathbf{T} = \mathbf{T}_p \cup \mathbf{T}_n$ and $\mathbf{T}_p \cap \mathbf{T}_n = \emptyset$. Without loss of generality, we define the set of positive features as $\mathbf{T}_p = \{t_1, t_2, \ldots, t_{N_p}\}$ and the negative feature set as $\mathbf{T}_n = \{t_{N_p+1}, \ldots, t_{N_p+N_n}\}$ to be used in Section 3.3. N_p and N_n are the number of positive and negative features, respectively. The total number of features N_f is $N_p + N_n$.

Zheng et al. [27] considered the correlations between positive and negative features. They evaluated the performance of combining the positive and negative features for text categorization. In this study, we further investigate the distributions of these two types of features. Specifically, we use χ^2 statistics to compute their term weight. The χ^2 statistic of a term t_i with respect to c is defined as follows [55]:

$$\chi^2(t_i, c) = \frac{N \times (A_1A_4 - A_2A_3)^2}{(A_1 + A_3)(A_2 + A_4)(A_1 + A_2)(A_3 + A_4)} \tag{2}$$

where N is the total number of documents.

The term weight quantifies how much term t_i contributes to the discriminative semantics of the class of interest c. Features with less discriminative power have small weights, while features with large discriminative capability get large weights. Some features may have very small or even zero weights. To prevent a large fraction of features from getting essentially zero weight, we set the lowest weight to a small positive value $1/N_f$, where N_f is the number of features.

Table 1
Relationship of feature t_i and category c.

	Presence of t_i	Absence of t_i
Labeled as c	A_1	A_3
Not labeled as c, i.e., $\bar{c}$	A_2	A_4

3.3. Stratified sampling method

The stratified sampling method for feature subspace selection is accomplished by the following procedure:

(i) Partition the set of features in $\mathbf{T}$ into positive features $\mathbf{T}_p$ and negative features $\mathbf{T}_n$, and consider a non-negative function φ (in this study, we use χ^2 statistics as the φ function) to measure the term weight of both positive and negative features;
(ii) Normalize the term weight for these two types of features separately. For positive features $t_i \in \mathbf{T}_p$, their corresponding normalized φ_i is computed as $\theta_i = \varphi_i / \sum_{k=1}^{N_p} \varphi_k$; for negative features $t_i \in \mathbf{T}_n$, we have $\theta_i = \varphi_i / \sum_{k=N_p+1}^{N_f} \varphi_k$. The normalized weighting value θ_i is in the range of 0 and 1.
(iii) Select $mtry_p = mtry \times \frac{N_p}{N_f}$ features from $\mathbf{T}_p$ and $mtry_n = mtry - mtry_p$ features from $\mathbf{T}_n$ according to their normalized weight values, are then merge them to form a subspace with *mtry* features for tree construction. *mtry* is specified to contain at least one from each group.

For imbalanced text categorization, the weights of positive and negative features are not necessarily comparable with each other due to the class imbalance of majority class and minority class. In general, features for the majority class have higher chance to obtain a larger weight. Consider the term weight values in Table 2. The maximum χ^2 term weights of positive and negative features for different classes of the Reuters-21578, Ohsumed, and imbalanced 20 newsgroup datasets are listed, where "+" and "−" represent the positive and negative features, respectively. One observes that the weights of positive and negative features associated with different categories can be very different. Therefore, the positive and negative features are normalized and selected separately in our approach. In this way, we guarantee that the subspace at any node contains both positive and negative features. The selected features are determined by their weights. This means that, in principle, the features selected in the subspace are more informative for imbalanced text categorization than those selected by a simple random sampling method.

3.4. The ForesTexter Algorithm

Our proposed ForesTexter algorithm involves training an ensemble of SVM-based trees using the RF framework for imbalanced text categorization. Compared to single SVM-based tree classification model, the RF-based ensemble approach is more robust for classification. Specifically, the ForesTexter algorithm constructs an ensemble of our proposed SVM-based trees from training documents D using a set of N_f features $\mathbf{T} = \{t_1, t_2, \ldots, t_{N_f}\}$ is summarized as follows:

1. Divide the feature set $\mathbf{T}$ into positive features and negative features $\mathbf{T}_p$ and $\mathbf{T}_n$, and compute their normalized term weight θ_i.
2. Use bagging method to generate K data subsets $\{D_1, D_2, \ldots, D_K\}$ from D.
3. Build SVM-based tree model $h_i(D_i)$ for each data set D_i in a top-down manner. At each node, use stratified sampling method to select a subspace of *mtry* features (or dimension) from $\mathbf{T}_p$ and $\mathbf{T}_n$ according to their term weight θ_i. Train a SVM classifier at each node with the feature subspace and D_i at the node. This process continues until a stopping criterion is met: all documents belong to the same class or the documents cannot be further split by the induced classifier.
4. Combine the K unpruned trees $h_1(D_1), h_2(D_2), \ldots, h_K(D_K)$ into an ensemble classifier, and use the majority votes to make classification prediction.

There are two parameters for the ForesTexter algorithm: the subspace dimension *mtry* and the number of trees K. In general, it is necessary to construct more trees for more complex learning problems. The value of *mtry* controls the strength and randomness of the generated trees. By setting $mtry = N_f$, in essence we are using a decision tree model. With $mtry = 1$, we produce completely random trees. ForesTexter algorithm constructs an ensemble of hierarchical tree models using SVM classifier at each node. In practice we find that the SVM parameters at each node do not require fine-tuning. In the experiments, we simply use default SVM parameters for tree construction, and we find this to work well.

Table 2
Maximum term weight values (based on χ^2 statistics) of positive and negative features with respect to different categories on the Reuters-21578, Ohsumed and imbalanced 20 newsgroup datasets.

Reuters-21578			Ohsumed			Imbalanced 20 newsgroup		
Id	max χ^2		Id	max χ^2		Id	max χ^2	
	+	−		+	−		+	−
1	3357.0	592.7	1	271.0	82.5	1	327.3	92.8
2	1401.3	906.7	2	3720.0	216.0	2	338.2	68.2
4	2337.2	122.1	4	865.0	108.0	3	426.9	16.9
6	1285.6	78.1	6	1010.0	66.4	4	128.9	20.9
8	4674.9	47.7	8	2850.0	41.0	5	253.0	10.9
10	3063.1	34.0	9	2670.0	44.6	6	223.8	31.1
12	2252.2	29.2	10	2260.0	73.4	7	367.3	94.6
14	5103.3	21.5	11	3190.0	52.8	8	499.6	6.6
16	3016.3	20.0	12	814.0	55.8	9	609.1	5.8
18	2770.6	22.1	13	3760.0	36.2	10	360.7	19.5
20	3786.1	27.2	14	1630.0	42.8	11	582.3	5.2
22	634.2	18.7	15	1670.0	26.0	12	686.1	10.3
24	3191.7	9.9	16	841.0	39.7	13	145.0	4.1
26	3266.1	6.6	17	2400.0	35.5	14	404.2	3.5
28	1127.2	11.2	18	1370.0	16.7	15	181.7	2.8
30	2224.7	6.8	19	3770.0	33.7	16	312.2	12.3
32	3053.8	6.3	20	1590.0	25.8	17	486.7	2.6
34	2091.6	5.0	21	1180.0	25.5	18	584.8	2.8
38	2944.5	3.7	22	991.0	159.0	19	503.4	2.6
40	3925.3	2.9	23	3590.0	14.1	20	586.0	1.5

The ForesTexter algorithm perform classification for an unseen instance x_i using a top-down manner to navigate the instance x_i through the trees to their leaf nodes. Each tree produces a vector of relevant scores for classification by accumulating the probability outputs of the SVM classifiers in the decision path x_i traversed. Specifically, let $\mathcal{N}$ be the set of nodes in the decision path, $\text{Prob}_n(x_i, c)$ be the probability output of x_i to class c given by the SVM classifier in node n. We compute the relevant scores for classification as follows: $\text{Prob}(x_i, c) = \sum_{n \in \mathcal{N}} \text{Prob}_n(x_i, c)$. This procedure is iterated over all trees in the ensemble. The prediction over all the trees are combined as the ensemble prediction.

Our proposed ForesTexter approach is different from the Random Forest (RF) algorithm proposed in [9] and the Stratified Random Forest (SRF) algorithm proposed in [56]. The RF algorithm [9] uses a simple random sampling of feature subspace and Gini gain splitting strategy in building their decision trees. The SRF algorithm [56] employs the stratified sampling method to select strong and weak features. The linear discriminative analysis (LDA) model is employed in SRF as the splitting function in the nodes of the trees. Our stratified sampling method, on the other hand, is based on positive and negative features and their term weights. Moreover, we use a SVM classifier at each node for tree construction.

3.5. Computational complexity

Given a training data set containing N documents and N_f feature terms, the computational cost (i.e., the time required to solve a problem) for obtaining the positive and negative feature stratification and χ^2 term weighting is $O(N)$ for one feature. In case of N_f features, the cost is $O(NN_f)$. The computational cost is linear and depending on the number of documents and feature terms. Note that this step is performed before learning the ensemble model. The complexity of learning our proposed tree ensemble model depends on the number of nodes in the trees that are required to construct SVM classifiers. To simplify the analysis, we assume that the tree is a binary tree with tree height h. The total number of nodes in the tree is up to $O(2^h)$ (each internal node has two branches). At each node, the cost of training a SVM classifier is $f(m) \approx O(m^2)$ [57], where m is the number of training documents available at the node and $m \ll N$. Then, the overall complexity for building a tree model with height h is $O(m^2 2^h)$. In fact, a precise complexity analysis is hard to achieve because the tree height depends on many factors. If the tree structure is extremely imbalanced, the tree height h will be about $O(N)$. In this case, the complexity is $O(N^2 2^N)$. In general, for a balanced tree, h is about $O(\sqrt{N})$. This leads to a computational complexity of $O(N^2 2^{\sqrt{N}})$ for one tree, and $O(KN^2 2^{\sqrt{N}})$ for K trees.

4. Experimental setup

To evaluate the performance of the proposed FORESTEXTER algorithm, we conduct experiments on three commonly used corpora for text categorization: Reuters-21578,[1] Ohsumed,[2] and 20 newsgroup.[3] A summarized description of these three text collections is given in Table 3.

The Reuters-21578 corpus consists of 21,578 news articles taken from Reuters newswire. We use the "ModApte" split and discard those documents with multiple category labels and those categories less than 10 documents. This leaves us with 8203 documents in 40 categories. There are 5891 training documents and 2312 testing documents. After preprocessing (stemming and stop-words filtering), this corpus contains 18,933 distinct features. The distribution of the number of documents across different categories in this data set is highly imbalanced (see Fig. 5).

Table 3
A collections used in our empirical study.

Data set	Documents		Terms	Classes
	Training	Test		
Reuters-21578	5891	2312	18,933	40
Ohsumed	6286	7643	14,503	23
Imbalanced 20NG	11,314	7532	26,214	20

The second text collection is the Ohsumed corpus compiled by William Hersh. It is a set of 348,566 medline documents consisting of titles and/or abstracts from 270 medical journals over 1987–1991. Following [6], we use the subset consisting of the first 10,000 documents from the medical abstract of the year 1991 for training and the second 10,000 documents for testing. After stemming and removing stop-words, the training corpus of Ohsumed contains 14,503 distinct features which occur in at least three documents. We consider the classification task for 23 MeSH "diseases" categories where the class distribution of these categories is also highly imbalanced.

20 Newsgroup (20NG) is a data collection with almost equal data distribution across all classes. It consists of 19,997 news documents partitioned (nearly) evenly across 20 different newsgroups. There are three versions of the dataset (original version, bydate version, and the cross-post version). We use the "bydata" version (the data was split into training (60%) and test (40%) sets, and the duplicates and headers were removed) as in [58,59]. Note that the "bydate" version is easier for cross-experiment comparison. After preprocessing, the training set contains 26,214 distinct features. However, in this paper, we focus on class imbalanced problem. In order to generate an imbalanced 20NG data collection, we eliminate instances in different classes to generate an imbalanced class distribution. After that, the number of documents in different categories of the new 20NG ranges from 10 to 500. This new dataset class distribution is highly imbalanced. It is quite different from the original 20NG dataset (see Fig. 5).

The three collections are quite different in document length. Reuters-21578 and Ohsumed consist of relative short documents, the length of which are around 800 bytes on average, while the 20NG collection consists of relatively long documents having about 1800 bytes on average. The term frequency (TF) representation for these datasets in Matlab format is available at http://www.zju-

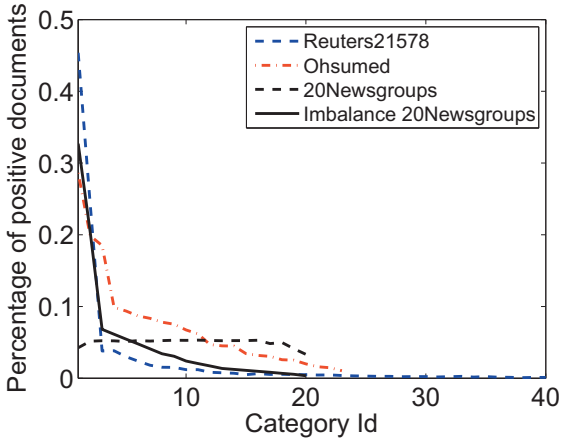

Fig. 5. The percentage of documents associated with different classes in the training set.

[1] http://www.research.att.com/lewis.
[2] ftp://medir.ohsu.edu/pub/ohsumed.
[3] http://qwone.com/jason/20Newsgroups/.

cadcg.cn/dengcai/Data/TextData.html. For each dataset, we compute the term frequency-inverse document frequency (TF-IDF) representation as follows: $(1+\log tf)\times\log N/df$, where N is the total number of documents, tf and df are the term frequency and the document frequency, respectively.

One of the simple methods to overcome the imbalanced issue in a classification task is the sampling methods. The motivation of using sampling methods in imbalanced learning is to modify the imbalanced dataset to achieve a balanced dataset. Random undersampling method selects a subset of the original data-set with balance class distribution by randomly eliminating instances from the majority class. On the other hand, random oversampling method creates a superset of the original data-set by randomly replicating minority class instances. The main advantage of these techniques is that they are independent of the underlying classifier. In our experiments, we combine random undersampling and random oversampling techniques with SVM classifier as the performance baselines.

We use the linear SVM classifier implemented in LibSVM.[4] The value of the regularization parameter C is selected from $\{2^{-5},2^{-3},\ldots,2^{15}\}$. The optimal regularization parameter is chosen using a 3-fold cross-validation on the training set. For our ForesTexter algorithm, we use χ^2 as the term weight function when computing a feature stratification, and employ a linear SVM with default parameter value $C=1.0$ as splitting criterion while splitting a tree node. For the number of trees K and the subspace dimension $mtry$, we follow the setting in [56] and use $mtry=N_f/10$ and $K=100$ in standard RF and ForesTexter, where N_f is the total number of features in the dataset.

5. Experimental results

In this section, we present experimental results to compare the performance of the proposed ForesTexter algorithm with other learning algorithms: standard RF and three variants of SVM algorithms (standard SVM, undersampling SVM, and oversampling SVM) on the Reuters-21578, Ohsumed and the imbalanced 20NG data collections. We show that performance of the proposed algorithm is competitive against these algorithms.

5.1. Evaluation measures

The classifiers may have a high overall accuracy with the majority class having close to 100% accuracy and the minority class having close to 0–10% because overall accuracy is biased towards the majority class. Hence, accuracy measure is not a proper evaluation metric for class imbalance problem. We use the AUC, the area under the receiver operating characteristics (ROC) curve [60], for evaluation. AUC has been proven to be a reliable performance measure for imbalanced and cost-sensitive problems [61].

Suppose we have two different outcomes for a binary class evaluation $B(tp\ rate, fp\ rate)$ with respect to the class of interest c, where tp (true positive) rate is the rate of documents correctly assigned to the class c, i.e.,

$$tp\ rate=\frac{\text{Positive instances correctly classified}}{\text{Total positive instances}},$$

and fp (false positive) rate is the rate of documents that belong to $\bar{c}$ but are assigned to c, i.e.,

$$fp\ rate=\frac{\text{Negative instances incorrectly classified}}{\text{Total negative instances}}.$$

[4] Available at http://www.csie.ntu.edu.tw/cjlin/libSVM/.

ROC graph is a two-dimensional graph in which fp rate is plotted on the X axis and tp rate is plotted on the Y axis. In classification evaluation, the classifier model produces a continuous output (i.e., an estimate of an instance's class membership probability) to which different thresholds are applied to predict class membership. If the classifier output is above the threshold, the classifier predicts the instance as class c, else $\bar{c}$. In this way, each threshold value produces a different prediction result to compute the results of $B(tp\ rate, fp\ rate)$. Each of the thresholds corresponds to a different point in ROC space. Conceptually, we can imagine varying a threshold from $-\infty$ to $+\infty$ and tracing a curve through ROC space, and calculate the area under the ROC curve to evaluate the strength of a classifier across various thresholds.

5.2. Positive and negative features

We investigate the skewness of the positive and negative features associated with different categories. The term weight values of positive and negative features in the Reuters-21578 dataset are shown in Fig. 6. We observe that the distributions of the positive and negative features with respect to different categories are quite different from each other. The result implies that these two feature types should be separately considered in the feature subspace selection while building the tree models. We sort the categories in descending order according to the number of instances associated with the classes. Then, we show the percentage of positive and negative instances in the classes. The results are given in Fig. 7. We observe that the percentage of positive (negative) features decreases (increases) as the number of learning class index increases.

In summary, the numbers and distributions of positive and negative features in imbalanced text datasets are highly skewed. A simple random sampling method for feature subspace selection in RF, treating all features equally, is not effective for imbalanced datasets.

5.3. Comparison results on imbalanced text collections

Next, we present the results for the binary classification task on the Reuters, Ohsumed and imbalanced 20NG collections. We convert the multi-class text classification problem into a set of binary class problems. We compare the performance of different algorithms on the binary classification problems. The performance is measured using AUC averaging over different binary classification problems. The results are given in Table 4, where the number in brackets behind the performance value represents the rank of the method on the corresponding data set. Each row in the table shows the results of different algorithms on a dataset. For each dataset, the compared algorithms are ranked in a decreasing order based on their performance. The number in boldface represents the best average AUG of the compared methods.

We can see from Table 4 that ForesTexter performs best on all three datasets followed by the different variants of SVM methods. The standard RF method performs the worst among the five compared algorithms on the Reuters-21578 and imbalanced 20NG. This is consistent with our earlier assertions that RF using the simple random sampling and the Gini measure is not reliable for for imbalanced text categorization. In comparison, undersampling SVM is the second best method on the Reuters-21578, whilst oversampling SVM is the second best method, slightly outperforming standard SVM, on the Ohsumed and imbalanced 20NG datasets. From Table 4, it is evident that the undersampling and oversampling version for SVM do not consistently outperform the standard SVM. SVM has the same performance as the oversampling SVM on Reuters-21578 and imbalanced 20NG. Recent studies [62,63] have shown that classifiers induced from certain imbalanced datasets

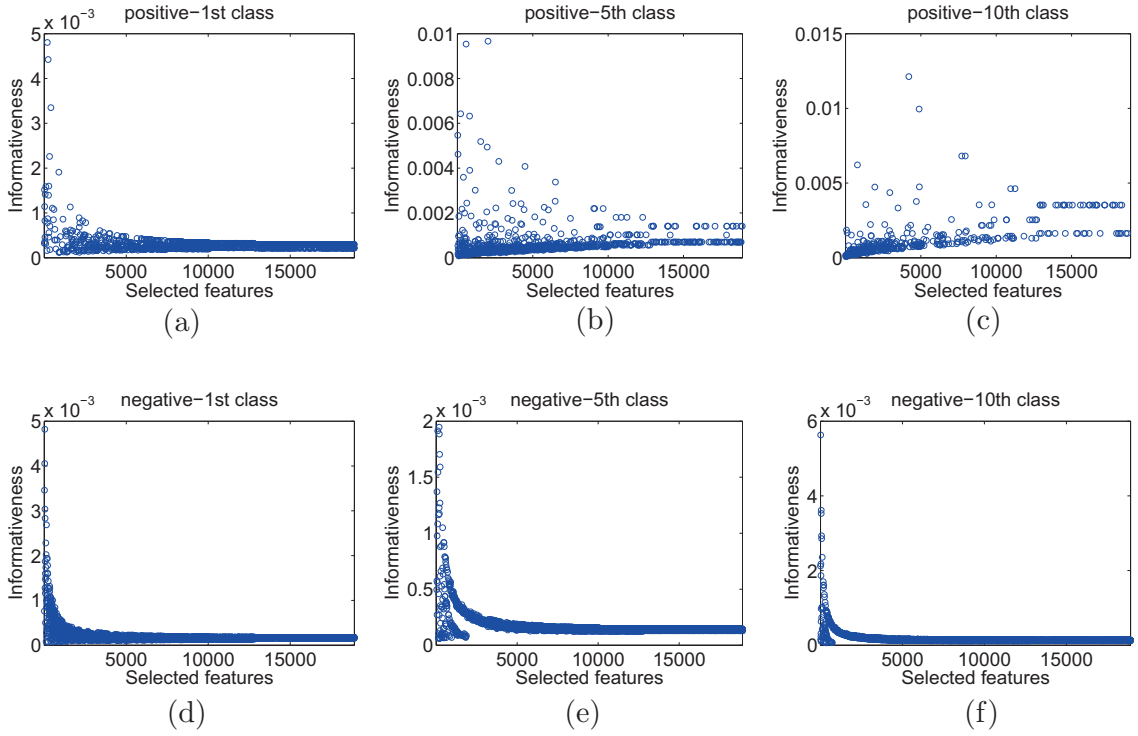

Fig. 6. The term weight values of positive and negative features associated with different categories in Reuters-21578 collection, where χ^2 statistic is used as term weight measure; (a–c) are the distributions of positive feature, and (d–f) are the distributions of negative feature in the 1st, 5th, 10th classes.

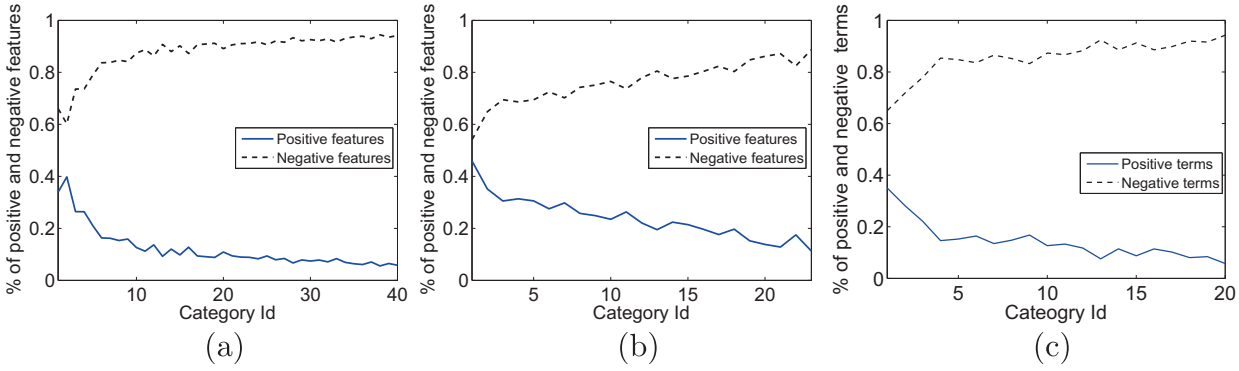

Fig. 7. The percentage of positive and negative features in the training set: (a) Reuters-21578; (b) Ohsumed; (c) imbalanced 20NG.

Table 4
Average AUC results of different learning algorithms on the Reuters-21578, Ohsumed and imbalanced 20NG datasets.

Dataset	VM	Under-SVM	Over-SVM	RF	ForesTexter
Reuters-21578	0.9755(3)	0.9915(2)	0.9753(4)	0.9593(5)	**0.9920**(1)
Ohsumed	0.9210(4)	0.9061(5)	0.9379(2)	0.9338(3)	**0.9465**(1)
Imbalanced 20NG	0.9375(2)	0.9216(4)	0.9375(2)	0.9085(5)	**0.9438**(1)

are comparable to classifiers induced from the same dataset balanced by sampling techniques. The results from our experiments are in concordance with these studies.

The above experimental results demonstrate the effectiveness of our proposed ForesTexter algorithm for learning imbalanced text datasets converted into multiple binary classification problems. We note that learning the minority classes with only a small number of associated instances is the most interesting case for our algorithm because it handles the cases where the data distribution of classes is highly imbalanced. In order to validate this point, we divide the classes of the datasets into three bins (or groups) according to the number of positive instances associated with the classes. The first bin (Bin1) contains large-size classes with a lot of instance included. The second bin (Bin2) consists of medium-size classes with a reasonable numbers of instances. The third bin (Bin3) consists of small-size classes such that the number of instances are insufficient for training a good model. For example, the classes in the first bin of the Reuters-21578 data collection (Bin1) are the largest three size classes containing most of the instances, i.e., $\{c_1, c_2, c_3\}$. The number of positive instances in these

Table 5
Average AUC results of different learning algorithms with respect to different sizes of categories on Reuters-21578.

#Classes	#Docs	SVM	Under-SVM	Over-SVM	RF	ForesTexter
Bin1 (c1–c3)	⩾320	0.9961(2)	0.9946(4)	0.9961(2)	0.9925(5)	**0.9968**(1)
Bin2 (c4–c10)	90–320	0.9952(2)	0.9922(4)	0.9942(3)	0.9899(5)	**0.9963**(1)
Bin3 (c11–c40)	10–90	0.9698(3)	0.9901(2)	0.9698(3)	0.9500(5)	**0.9908**(1)

Table 6
Average AUC results of different learning algorithms with respect to different sizes of categories on Ohsumed.

#Classes	#Docs	SVM	Under-SVM	Over-SVM	RF	ForesTexter
Bin1 (c1–c3)	⩾1000	0.8929(2)	0.8671(5)	0.8784(4)	0.8808(3)	**0.8961**(1)
Bin2 (c4–c10)	101–1000	0.9356(4)	0.9262(5)	0.9465(2)	0.9423(3)	**0.9533**(1)
Bin3 (c11–c23)	10–100	0.9197(4)	0.9043(5)	0.9470(2)	0.9415(3)	**0.9545**(1)

Table 7
Average AUC results of different learning algorithms with respect to different sizes of categories on imbalanced 20NG.

#Classes	#Docs	SVM	Under-SVM	Over-SVM	RF	ForesTexter
Bin1 (c1–c3)	⩾200	**0.9641**(1)	0.9593(4)	**0.9641**(1)	0.9378(5)	0.9638(3)
Bin2 (c4–c10)	70–200	0.9602(2)	0.9489(4)	0.9602(2)	0.9200(5)	**0.9606**(1)
Bin3 (c11–c20)	10–70	0.9136(2)	0.8911(5)	0.9136(2)	0.8916(4)	**0.9278**(1)

Table 8
Number of positive and negative documents and features with respect to different class bins on Reuters-21578.

	Bin1 (c1–c2)		Bin2 (c3–c9)		Bin3 (c10–c14)		Bin4 (c15–c40)	
	Pos.	Neg.	Pos.	Neg.	Pos.	Neg.	Pos.	Neg.
Number of features	6299	12615	3197	15717	1492	17422	739	18175
Proportion of features	0.333	0.667	0.169	0.831	0.079	0.921	0.039	0.961
Number of documents	2054	3837	150	5741	55	5836	18	5873
Proportion of documents	0.349	0.651	0.025	0.975	0.009	0.991	0.003	0.997

classes are larger than 320. On the other hand, the size of classes of the second bin (Bin2) is relatively small as compared to the classes in Bin1. The number of instances of these classes is in the range of 90 to 320. Finally, the third bin (Bin3) contains classes much smaller than those in Bin1 and Bin2. The number of instances in these classes is ranging from 10 to 50.

We compare the average AUC of different algorithms over the categories within each bin. The results are shown in Tables 5–7 for Reuters-21578, Ohsumed, and imbalanced 20NG, respectively. We rank the performance of the algorithms in a decreasing order. The performance ranking of the methods are given in brackets. The numbers in boldface indicate the best average AUG obtained by the compared algorithms. We observe that the performance of ForesTexter is competitive against the other tested algorithms across all three datasets. Moreover, one observes the results in the last rows of Tables 5–7 show that our proposed method outperforms the other tested methods in classes with small number of instances. The results illustrate our method works well on minority classes with only a small number of instances.

6. Discussion

The fundamental issue of the imbalanced classes in learning is how to improve the performance of learning algorithms on minority classes [20]. Most of the standard algorithms assume that the class distributions are balanced, and they fail to properly account for the unequal characteristics of the data samples and features of the minority classes.

Similar to the last experimental design in the previous section, we divide the classes of the Reuters-21578 into different bins in which the Bin1 is for the largest two classes (c1–c2), Bin2 for class 3 to 9 (c3–c9), Bin3 for class 10 to 14 (c10–c14), and Bin4 for the smallest classes (c15–c40). Table 8 shows the number of features and instances associated with the classes in different bins. One observes that the numbers of positive and negative features as well as data samples are highly skewed in imbalanced text data, especially for the minority classes in Bin4. In such cases, RF using simple random sample strategy to select subspaces has a high chance of missing the informative features for the minority classes. On the other hand, the proposed ForesTexter method uses stratified sampling method and ensemble of SVM-based trees to force the classifier to learn from the refined feature subspaces and data subspaces. We note that standard RF cannot provides a balanced degree of predictive accuracy over different bins: Bin1 has a relatively good accuracy and Bin4 has a relatively poor accuracy.

However, our proposed ForesTexter algorithm is able to provide a more balance degree of predictive accuracy for both minority and majority classes against RF. As shown in Table 4, the overall improvement of ForesTexter over RF is about 3.27% (ForesTexter AUC: 0.9920 versus RF AUC: 0.9593). While the improvements of ForesTexter over RF on Bin1, Bin2 and Bin3 are 0.43% (ForesTexter AUC: 0.9968 versus RF AUC: 0.9925), 0.64% (ForesTexter AUC: 0.9963 versus RF AUC: 0.9899) and 4.10% (ForesTexter AUC: 0.9908 versus RF AUC: 0.9500), respectively (see Table 5). The improvements of ForesTexter over RF are more significant on minority classes with limited number of instances. Our experimental results indicate that it is advantageous to use the proposed tree induction methods, including both feature subspace selection and data splitting choice, to generate RF for imbalanced text categorization.

7. Conclusion

In this paper, we describe a new RF algorithm, ForesTexter, for imbalanced text categorization. It consists of both stratified sampling feature subspaces and learning SVM classifier while splitting a tree node. The idea is to (i) ensure the feature subspace contains informative features for majority and minority classes using a stratified sampling method, and (ii) make the splitting at the nodes stronger and fit the imbalanced data better. The proposed approach identified two groups of features (positive and negative features) and their term weights. The tree building algorithm then stratified sample features from each group in building a tree. In learning a tree model, we retrain SVM sub-classifiers using the training data subsets and feature subspaces at the nodes to successively refine the tree model. We have conducted extensive experiments on three benchmark imbalanced text datasets (Reuters-21578, Ohsumed, and imbalanced 20NG) to demonstrate the effectiveness of this new algorithm. Experimental results have shown that the classification performance of our proposed method is better than those standard random forest and different variants of SVM algorithms, in particular on the minority classes with extremely small number of instances. Our future work includes investigating more effective techniques to solve the feature weighting problem for imbalanced text data. Furthermore, we will explore applications of our approach to other tasks such as multi-label text categorization.

Acknowledgements

Y. Ye's research supported in part by National Commonweal Technology R&D Program of AQSIQ China under Grant No. 201310087, Shenzhen Strategic Emerging Industries Program under Grants No. JCYJ20130329142551746. H. Zhang's research supported in part by the National Nature Science Foundation of China under Gran No. 61300209, the Shenzhen Foundation Research Fund under Grant No. JCY20120613115205826 and the Shenzhen Technology Innovation Program under Grant No. CXZZ20130319100919673. S.S. Ho's research supported in part by AcRF Grant RG-41/12 and Singapore NTU Start-Up Grant.

References

[1] F. Sebastiani, Machine learning in automated text categorization, ACM Comput. Surveys (CSUR) 34 (1) (2002) 1–47.
[2] G.V. Cormack, M.D. Smucker, C.L. Clarke, Efficient and effective spam filtering and re-ranking for large web datasets, Inform. Retriev. 14 (5) (2011) 441–465.
[3] L. Kallipolitis, V. Karpis, I. Karali, Semantic search in the world news domain using automatically extracted metadata files, Knowledge-Based Syst. 27 (2012) 38–50.
[4] K. Ikeda, G. Hattori, C. Ono, H. Asoh, T. Higashino, Twitter user profiling based on text and community mining for market analysis, Knowledge-Based Syst. 51 (2013) 35–47.
[5] Y. Yang, J. Pedersen, A comparative study on feature selection in text categorization, in: Proceedings of ICML'97 the IEEE International Conference on Machine Learning, 1997, pp. 412–420.
[6] T. Joachims, Text categorization with support vector machines: learning with many relevant features, in: The European Conference on Machine Learning (ECML), 1998, pp. 137–142.
[7] R. Schapire, Y. Singer, Boostexter: a boosting-based system for text categorization, Machine Learn. 39 (2) (2000) 135–168.
[8] Y.H. Li, A.K. Jain, Classification of text documents, Comput. J. 41 (8) (1998) 537–546.
[9] L. Breiman, Random forests, Machine Learn. 45 (1) (2001) 5–32.
[10] A. Liaw, M. Wiener, Classification and regression by random forest, R News 2 (3) (2002) 18–22.
[11] K.J. Archer, R.V. Kimes, Empirical characterization of random forest variable importance measures, Comput. Stat. Data Anal. 52 (4) (2008) 2249–2260.
[12] G. Biau, L. Devroye, G. Lugosi, Consistency of random forests and other averaging classifiers, J. Machine Learn. Res. 9 (2008) 2015–2033.
[13] G. Biau, Analysis of a random forests model, J. Machine Learn. Res. 98888 (1) (2012) 1063–1095.
[14] Q. Wu, Y. Ye, Y. Liu, M.K. Ng, Snp selection and classification of genome-wide snp data using stratified sampling random forests, IEEE Trans. Nanobiosci. 11 (3) (2012) 216–227.
[15] B. Goldstein, A. Hubbard, A. Cutler, L. Barcellos, An application of random forests to a genome-wide association dataset: methodological considerations & new findings, BMC Genet. 11 (2010) 49–61.
[16] F. Moosmann, E. Nowak, F. Jurie, Randomized clustering forests for image classification, IEEE Trans. Pattern Anal. Machine Intell. (2008) 1632–1646.
[17] V. Lepetit, P. Lagger, P. Fua, Randomized trees for real-time keypoint recognition, Proc. IEEE CVPR Conf. Comput. Vis. Pattern Recognit. (2005) 775–781.
[18] N. Tomašev, D. Mladenić, Class imbalance and the curse of minority hubs, Knowledge-Based Syst. 53 (2013) 157–172.
[19] V. García, R.A. Mollineda, J.S. Sánchez, A bias correction function for classification performance assessment in two-class imbalanced problems, Knowledge-Based Syst. 59 (2014) 66–74.
[20] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowledge Data Eng. 21 (9) (2009) 1263–1284.
[21] N.V. Chawla, Data mining for imbalanced datasets: an overview, in: Data Mining and Knowledge Discovery Handbook, 2010, pp. 875–886.
[22] N. Japkowicz, Learning from imbalanced data sets, in: Proceedings the AAAI'00 Workshop on Learning from Imbalanced Data Sets, AAAI Tech Report WS-00-05, AAAI.
[23] N.J.N.V. Chawla, A. Kolcz, Workshop learning from imbalanced data sets II, in: Proceedings of the ICML'03 Workshop on Learning from Imbalanced Data Sets.
[24] N.J.N.V. Chawla, A. Kolcz, Editorial: special issue on learning from imbalanced data sets, ACM SIGKDD Explorat. Newslett. 6 (1) (2004) 1–6.
[25] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches, IEEE Trans. Syst., Man, Cybernet., Part C: Appl. Rev. 42 (4) (2012) 463–484.
[26] D. Cieslak, N. Chawla, Learning decision trees for unbalanced data, in: Proceedings of the Machine Learning and Knowledge Discovery in Databases, 2008, pp. 241–256.
[27] Z. Zheng, X. Wu, R. Srihari, Feature selection for text categorization on imbalanced data, Proc. SIGKDD'04 ACM Int. Conf. Knowledge Discovery 6 (1) (2004) 80–89.
[28] S. Tan, X. Cheng, M.M. Ghanem, B. Wang, H. Xu, A novel refinement approach for text categorization, in: Proceedings of the 14th ACM International Conference on Information and Knowledge Management, 2005, pp. 469–476.
[29] T. Jo, N. Japkowicz, Class imbalances versus small disjuncts, ACM SIGKDD Explorat. Newslett. 6 (1) (2004) 40–49.
[30] Y. Sun, A. Wong, M. Kamel, Classification of imbalanced data: a review, Int. J. Pattern Recognit. Artif. Intell. 23 (4) (2009) 687–719.
[31] M. Kubat, S. Matwin, et al., Addressing the curse of imbalanced training sets: one-sided selection, in: Proceedings of the 14th International Conference on Machine Learning, vol. 97, 1997, pp. 179–186.
[32] M. Kubat, R. Holte, S. Matwin, Learning when negative examples abound, in: Proceedings of the 9th European Conference on Machine Learning, 1997, pp. 146–153.
[33] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, Smote: synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[34] Z.-H. Zhou, Cost-sensitive learning, in: Modeling Decision for Artificial Intelligence, 2011, pp. 17–18.
[35] C. Ferri, P. Flach, J. Hernández-Orallo, Learning decision trees using the area under the roc curve, in: Proceedings of the 19th International Conference on Machine Learning, vol. 2, 2002, pp. 139–146.
[36] X.-Y. Liu, Z.-H. Zhou, The influence of class imbalance on cost-sensitive learning: an empirical study, in: Proceedings of the 6th International Conference on Data Mining, 2006, pp. 970–974.
[37] K. Huang, H. Yang, I. King, M.R. Lyu, Learning classifiers from imbalanced data based on biased minimax probability machine, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, 2004, pp. 558–563.
[38] G.J. Williams, Combining decision trees: initial results from the mil algorithm, in: J.S. Gero, R.B. Stanton (Eds.), Artificial Intelligence Developments and Applications, 1988, pp. 273–289.
[39] T. Ho, The random subspace method for constructing decision forests, IEEE Trans. Pattern Anal. Machine Intell. 20 (8) (1998) 832–844.
[40] T. Dietterich, An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization, Machine Learn. 40 (2) (2000) 139–157.
[41] E. Gabrilovich, S. Markovitch, Text categorization with many redundant features: using aggressive feature selection to make SVMs competitive with C4.5, in: Proceedings of ICML'04 the IEEE International Conference on Machine Learning, pp. 321–328.
[42] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer-Verlag New York, Inc., New York, NY, USA, 1995.
[43] C.J. Burges, A tutorial on support vector machines for pattern recognition, Data Min. Knowledge Discovery 2 (2) (1998) 121–167.
[44] T. Joachims, Transductive inference for text classification using support vector machines, in: Proceedings of the 16th International Conference on Machine Learning, vol. 99, 1999, pp. 200–209.
[45] T. Joachims, Learning to Classify Text Using Support Vector Machines: Methods, Theory and Algorithms, Kluwer Academic Publishers, 2002.

[46] R. Akbani, S. Kwek, N. Japkowicz, Applying support vector machines to imbalanced datasets, in: Proceedings of the 15th European Conference on Machine Learning (ECML), Springer, 2004, pp. 39–50.
[47] F. Vilariño, P. Spyridonos, J. Vitrià, P. Radeva, Experiments with svm and stratified sampling with an imbalanced problem: detection of intestinal contractions, in: Pattern Recognition and Image Analysis, 2005, pp. 783–791.
[48] Y. Tang, Y.-Q. Zhang, N.V. Chawla, S. Krasser, Svms modeling for highly imbalanced classification, IEEE Trans. Syst., Man, Cybernet., Part B: Cybernet. 39 (1) (2009) 281–288.
[49] B.X. Wang, N. Japkowicz, Boosting support vector machines for imbalanced data sets, Knowledge Inform. Syst. 25 (1) (2010) 1–20.
[50] Y. Yang, X. Liu, A re-examination of text categorization methods, in: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1999, pp. 42–49.
[51] R. Caruana, N. Karampatziakis, A. Yessenalina, An empirical evaluation of supervised learning in high dimensions, in: Proceedings of ICML'08 the IEEE International Conference on Machine Learning, 2008, pp. 96–103.
[52] G. Forman, An extensive empirical study of feature selection metrics for text classification, J. Machine Learn. Res. 3 (2003) 1289–1305.
[53] F. Debole, F. Sebastiani, Supervised term weighting for automated text categorization, in: Text Mining and Its Applications, 2004, pp. 81–97.
[54] M. Lan, C.L. Tan, J. Su, Y. Lu, Supervised and traditional term weighting methods for automatic text categorization, IEEE Trans. Pattern Anal. Machine Intell. 31 (4) (2009) 721–735.
[55] Y. Yang, J. Pedersen, A comparative study on feature selection in text categorization, in: Proceedings of ICML'97 the IEEE International Conference on Machine Learning, 1997, pp. 412–420.
[56] Y. Ye, Q. Wu, Z.J. Huang, M.K. Ng, X. Li, Stratified sampling for feature subspace selection in random forests for high dimensional data, Pattern Recognit. 46 (3) (2012) 769–787.
[57] I.W. Tsang, J.T. Kwok, P.-M. Cheung, Core vector machines: fast svm training on very large data sets, J. Machine Learn. Res. 6 (1) (2006) 363.
[58] M. Lan, C.L. Tan, H.-B. Low, Proposing a new term weighting scheme for text categorization, in: Association for the Advancement of Artificial Intelligence (AAAI), vol. 6, 2006, pp. 763–768.
[59] A. Genkin, D.D. Lewis, D. Madigan, Large-scale Bayesian logistic regression for text categorization, Technometrics 49 (3) (2007) 291–304.
[60] J. Huang, C.X. Ling, Using auc and accuracy in evaluating learning algorithms, IEEE Trans. Knowledge Data Eng. 17 (3) (2005) 299–310.
[61] T. Fawcett, Roc graphs: notes and practical considerations for researchers, Machine Learn. 31 (2004) 1–38.
[62] G.E. Batista, R.C. Prati, M.C. Monard, A study of the behavior of several methods for balancing machine learning training data, ACM Sigkdd Explorat. Newslett. 6 (1) (2004) 20–29.
[63] N. Japkowicz, S. Stephen, The class imbalance problem: a systematic study, Intell. Data Anal. 6 (5) (2002) 429–449.