Arabian Journal for Science and Engineering (2020) 45:10471–10491
https://doi.org/10.1007/s13369-020-04763-5

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

A Novel Inherent Distinguishing Feature Selector for Highly Skewed Text Document Classification

Muhammad Sajid Ali[1] · **Kashif Javed**[1]

Received: 25 November 2019 / Accepted: 1 July 2020 / Published online: 22 July 2020

Abstract
A major problem text classification faces is the high dimensional feature space of the text data. Feature selection (FS) algorithms are used for eliminating the irrelevant and redundant terms, thus increasing accuracy and speed of a text classifier. For text classification, FS algorithms have to be designed keeping the highly imbalanced classes of the text data in view. To this end, more recently ensemble algorithms (e.g., improved global feature selection scheme (IGFSS) and variable global feature selection scheme (VGFSS)) were proposed. These algorithms, which combine local and global FS metrics, have shown promising results with VGFSS having better capability of addressing the class imbalance issue. However, both these schemes are highly dependent on the underlying local and global FS metrics. Existing FS metrics get confused while selecting relevant terms of a data with highly imbalanced classes. In this paper, we propose a new FS metric named inherent distinguished feature selector (IDFS), which selects terms having greater relevance to classes and is highly effective for imbalanced data sets. We compare performance of IDFS against five well-known FS metrics as a stand-alone FS algorithm and as a part of the IGFSS and VGFSS frameworks on five benchmark data sets using two classifiers, namely support vector machines and random forests. Our results show that IDFS in both scenarios selects smaller subsets, and achieves higher micro and macro F_1 values, thus outperforming the existing FS metrics.

Keywords Text document classification · Feature selection · Feature ranking metrics

1 Introduction

Advancement in Internet technologies revolutionizes the need for automatic textual data classification [1]. Nowadays, web is a main source of ever increasing unstructured textual data. Approximately 70–80% information of an organization is stored in an unstructured text format [2]. Arranging text documents into predefined different categories[1] is called text document classification [3]. Categorization of documents into known categories helps to find documents related to user queries. Text classification can be performed automatically and efficiently with the help of machine learning algorithms. Text classification is a prevailing application of natural language processing having a wide range of applications, e.g., game industry [4], stock price prediction [5], news agencies and escalating businesses via social media [6].

Broadly, there are three stages for implementing text classification with machine learning: (i) feature extraction or representation (ii) feature selection and (iii) classification [7]. In the feature extraction or data representation stage, each raw text is converted into a vector of numerical values. The most commonly used model for text representation is Bag of Words (BoW) [8]. Assuming a document to be a sequence of independent words, it does not take the structure and order of text into account. To make the BoW representation more compact and effective, various pre-processing steps such as stemming, stop words removal and pruning are applied [9]. In stemming, conversion of words[2] to their root form is performed, e.g., connection becomes connect, and taught is converted into teach. Stop words such as "the," "an," "a," "and," etc., are too frequently occurring words in

[1] In text classification, category is the same as the class.

✉ Muhammad Sajid Ali
photon526@gmail.com

Kashif Javed
kashif.javed@uet.edu.pk

[1] Department of Electrical Engineering, University of Engineering and Technology, Lahore, Pakistan

[2] In the paper, "features," "words" and "terms" are used interchangeably.

documents that do not provide any information about the category of a text and, thus should be removed with the help of a list of stop words [10]. Normally, a corpus of documents consists of a number of rarely occurring terms and a few very frequently occurring terms [11]. Such terms are irrelevant in discriminating among classes and are, thus removed via pruning [10]. In pruning, terms whose occurrence is below a certain lower threshold or above a certain upper threshold value are removed. Finally, a text document is represented as a vector $\mathbf{d} = \{tw_1, tw_2, tw_3, \ldots, tw_n\}$, where $W = \{w_1, w_2, w_3, \ldots, w_n\}$ is a vocabulary of n words and tw_i is the weight of the ith term [12].

In the quest for improving the performance of text classification, researchers have proposed a number of term weighting schemes. For example, terms can be represented by Boolean values indicating whether the term is present or absent. This is more suitable for shorter documents [13]. A term can also be represented by considering the number of times it appears in a document (called term count) or term count that is normalized by the length of a document (called term frequency) [14]. Among the schemes proposed in the literature, the most popular weighting scheme is term frequency-inverse document frequency (tf.idf) [15,16]. However, researchers have highlighted issues of tf.idf and have suggested alternatives of it. For example, [17] proposed a relevance frequency (rf) to capture the distribution of a term in different categories and showed that better performance for text classification can be obtained with tf.rf as compared to tf.idf. In another work [18], inverse gravity moment (igm) was proposed to capture the class distinguishing power of a term. It considers a term concentrated in one class to be more important than a term having relatively uniform distribution in a number of classes or all classes. Text classification based on tf.igm weighting scheme was found to outperform tf.idf but is computationally slightly more expensive. Similarly, a modified version of term frequency and inverse document frequency was proposed by [19]. The new scheme takes the amount of missing terms into account while calculating the weight of existing terms and has shown to improve text classification results.

Even for a moderate sized text data set, the $\mathbf{d}$ vector or vocabulary of words can contain tens of thousands of unique words [20]. Not all these words have the same discriminating power: terms can be relevant, irrelevant and even redundant for discriminating among the categories of the text documents [21]. Also, high dimensionality of the data degrades the classification accuracy and increases the computational complexity of a classifier [22,23]. Moreover, the processed text data are highly sparse as most of the words are absent and contain no information [24]. Therefore, the data used for text classification require dimensionality reduction or feature selection (FS) [25]. In this stage of text classification, filter[3] methods are preferred because they are simple and computationally efficient as compared to other feature selection methods [7].

Filter methods can be divided into two groups referred as global and local, which depends on whether the method assigns a unique score or multiple class-based scores to a feature [7]. For a local feature selection method, we define a globalization policy that converts the multiple local scores into a unique global score, while a global method uses the scores directly for ranking the features. Features are then sorted in descending order and the top N features are selected in the final set, where N can be determined empirically [26]. Some well-known examples of the global feature selection methods include information gain (IG) [27], Gini index (GI) [28] and distinguishing feature selector (DFS) [1]. Another criterion with which we can categorize filter methods is whether it uses a one-sided or two-sided feature ranking metric [29]. One-sided metrics select those features that are most indicative of membership of a given category (i.e., positive features), while the two-sided metrics implicitly combine the features most indicative of membership (i.e., positive features) and nonmembership (i.e., negative features) [30]. For example, GI and IG are two-sided FS metrics, while odds ratio and correlation coefficient are examples of one-sided metrics [29].

The final set when populated by a global filter with the addition of top N terms, there can be a number of classes whose terms may not be present in it, thus degrading the text classification performance [7]. To solve this problem, an improved global feature selection scheme (IGFSS) was proposed, which is an ensemble of local and global filter metrics [7]. The classification performance of global methods has shown to be improved by constructing a final set that contains almost equal number of terms from all the classes. In [31], the authors observed that IGFSS does not work well for imbalanced data sets having a large number of classes of varying sizes and addressed this limitation by suggesting a variable global feature selection scheme (VGFSS). This new approach shows better performance as compared to IGFSS and the global filters by selecting a variable number of features from each class depending on the distribution of terms in the classes. Although this approach shows promising results, its performance varies with the selection of different metrics, thus providing a room for improvement.

The major contributions of our research work in this paper are summarized:

- We discuss and highlight limitations of existing feature ranking metrics with the help of an example data set.
- We propose a new feature ranking metric named inherent distinguished feature selector (IDFS), which is effective

[3] We use filters, FS metrics or feature ranking metrics interchangeably.

in choosing relevant terms from highly imbalanced data sets.
- We show the effectiveness of IDFS against five well-known feature ranking metrics as a stand-alone FS algorithm and as a part of the IGFSS and VGFSS frameworks.
- We carry out experiments on five benchmark data sets using two classifiers, namely support vector machines [32] and random forests [33].

The remainder of this paper is organized in five sections. Section 2 provides a survey of the existing works related to feature selection. In Sect. 3, we illustrate problems with existing feature ranking metrics and also explain the working of our newly proposed IDFS feature ranking metric. Section 4 provides details of our experiments, while Sect. 5 presents the results and discusses them. Finally, we draw conclusions of the work done in this paper in Sect. 6.

2 Related Works

We divide this section into two subsections. In the first subsection, we provide an overview of the works of feature selection (FS) algorithms for text classification, while the second subsection focuses on the well-known filters.

2.1 Overview of Feature Selection Methods

The aim of feature selection is to further improve the performance of classifiers (e.g., text classifiers) while reducing their computational complexity [34]. This can be achieved by searching the feature space for an optimal feature subset whose relevance for the class variable is maximum and redundancy among the features is minimum [35]. In other words, we want to eliminate the irrelevant and redundant features. Feature selection algorithms proposed in the literature can be categorized based on multiple criteria [36]. The most important one is whether an FS algorithm seeks the help of the classifier while searching for the optimal subset [37]. Broadly, there can be three categories: filter, wrapper and embedded methods [38]. Unlike wrapper and embedded methods, filters search for the optimal feature subset without the guidance of the classifier [39]. The filters use a metric to evaluate the usefulness of features and output a ranked list of features according to their relative importance. Wrapper and embedded systems are more complex and computationally more expensive than filter methods due to which the latter methods are preferred for high-dimensional data sets such as text data [40].

Text classification researchers have mostly been focusing on designing filters because they are computationally the least expensive. We discuss some well-known examples in Sect. 2.2. In addition to the proposal of filters, we can find that new feature selection frameworks are being designed to further improve performance of text classification task.

Feature ranking methods ignore the redundancies among the terms. The output is a subset that consists of top ranked features highly relevant for the class but can possibly be redundant [41]. The final subset can miss two or more such features that are individually less relevant but together can better discriminate among the classes [39]. Therefore, designing algorithms for feature selection for high-dimensional data that take term dependencies into account without becoming computationally intractable is a major challenge. Toward this end, researchers have proposed two-stage algorithms. In the first stage, we select a subset of highly relevant terms while the second stage gets rid of redundant terms from the reduced space of features. For example, [42] found that combining IG in the first stage and principal component analysis or a genetic algorithm in the second stage can improve the text classification performance. Similarly, [25] proposed to capture the redundancy among terms selected in the first stage with the help of a Markov blanket in the second stage. The performance of text classification was found to improve significantly.

More recently, FS algorithm designers have focused on the class imbalance problem of the text data, which further aggravates in the one-vs-rest setting. To make feature selection more effective in the class imbalance scenario, [7] proposed an ensemble approach named IGFSS incorporating global and local filters for selecting equal number of features from each class. But [31] found that IGFSS approach does not work well on imbalanced data sets having large number of classes and proposed a new scheme named VGFSS. Unlike IGFSS, VGFSS selects a variable length of features depending on the class size and works well for imbalanced data sets. The downside is that its performance heavily relies on the underlying FS metrics. In case the metrics fail to select useful features, VGFSS also performs poorly.

2.2 Feature Ranking Metrics

Mostly, feature ranking metrics are based on document frequency. For two-class text classification, we present the basic notations in Table 1 and the confusion matrix in Table 2, which will allow us to better understand the feature ranking metrics. The confusion matrix defines true positive (tp), true negative (tn), false positive (fp) and false negative (fn), and their calculations are given in Table 1.

We further need to define term importance. There can be four different types of terms:

(a) Rare or Unique terms: terms present in only one class with higher weight of true positive rate (tpr) or $p(t_i|C_j)$ values and absent in other classes, where $tpr = p(t_i|C_j) =$

Table 1 The basic notations adapted from [31]

Notations	Values	Description
tp	count(t_i, C_j)	Document count of a term t_i in positive class C_j
fp	count$(t_i, \overline{C_j})$	Count how many times term t_i occurs in other classes except in C_j
fn	count$(\overline{t_i}, C_j)$	Count how many times respective term doesn't Occur into current class C_j
tn	count$(\overline{t_i}, \overline{C_j})$	Count how many times term t_i not occurs in Other classes except in C_j
M	$tp + fp + fn + tn$	Number of documents in corpus
$p(t_i)$	$(tp + fp)/M$	Probability of term t_i independent of class
$p(\overline{t_i})$	$(fn + tn)/M$	Probability of negative of term t_i independent of class
$p(C_j)$	$(tp + fn)/M$	Marginal probability of class C_j independent of term
$p(\overline{C_j})$	$(fp + tn)/M$	Marginal probability of negative of class C_j independent of term
$p(t_i, C_j)$	tp/M	Joint probability of term t_i and class C_j
$p(t_i, \overline{C_j})$	fp/M	Joint probability of term t_i and negative of class C_j
$p(\overline{t_i}, C_j)$	fn/M	Joint probability of negative of term t_i and class C_j
$p(\overline{t_i}, \overline{C_j})$	tn/M	Joint probability of negative of term t_i and negative of class C_j
$p(t_i \mid C_j)$	$tp/(tp + fn)$	Probability of term t_i when class C_j present
$p(t_i \mid \overline{C_j})$	$fp/(fp + tn)$	Probability of term t_i when negative of a class C_j present
$p(\overline{t_i} \mid C_j)$	$fn/(tp + fn)$	Probability of negative of term t_i when class C_j present
$p(\overline{t_i} \mid \overline{C_j})$	$tn/(fp + tn)$	Probability of negative of term t_i when negative of class C_j present
$p(C_j \mid t_i)$	$tp/(tp + fp)$	Probability C_j when term t_i is present
$p(C_j \mid \overline{t_i})$	$fn/(fn + tn)$	Probability C_j when negative of term t_i is present
$p(\overline{C_j} \mid t_i)$	$fp/(tp + fp)$	Probability of negative of C_j when term t_i is present
$p(\overline{C_j} \mid \overline{t_i})$	$tn/(fn + tn)$	Probability of negative of C_j when negative of term t_i is present

Table 2 The confusion matrix

Data class	Class classified as	
	Positive	Negative
Positive	$a = truepositive(tp)$	$c = falsenegative(fn)$
Negative	$b = falsepositive(fp)$	$d = truenegative(tn)$

$\frac{tp}{tp+fn}$ and false positive rate, $fpr = p(t_i|\overline{C_j}) = \frac{fp}{fp+tn}$ [43–45].

(b) Common terms: terms present either in most of the classes or in all classes.
(c) Negative terms: terms present in all classes but absent in a specific class is negative term for that class.
(d) Sparse terms: rare terms with very low tpr weight.

Accuracy (ACC) is a metric that simply takes the difference between tp and fp [13] and is given in Eq. (1). This metric assigns a higher score to frequent terms in the positive class as compared to the frequent terms in the negative class, thus penalizing latter terms that are highly relevant for the negative class.

$$\text{ACC} = tp - fp \tag{1}$$

Balanced accuracy measure (ACC2) [30] is an improved version of ACC given in Eq. (2). Although ACC2 addresses shortcomings of the ACC metric but can fail in differentiating important terms from redundant terms.

$$\text{ACC2} = |tpr - fpr| \tag{2}$$

Mutual information (MI) selects features based on their mutual dependence and chooses a higher number of terms of a larger class [46]. Its major weakness is that it is based on marginal probabilities and largely depends on tp values, ignoring class wise importance of terms [27]. The MI local metric is given in Eq. (3). If C_T denotes the total number of classes, then its global version is given in Eq. (4) [47].

$$\begin{aligned} \text{MI}\,(t_i, C_j) &= \log_2\left(\frac{p(t_i, C_j)}{p(t_i) \times p(C_j)}\right) \\ &= \log_2\left(\frac{tp \times M}{(tp + fp) \times (tp + fn)}\right) \end{aligned} \tag{3}$$

$$\text{MI}_{\max}\,(t_i) = \max_{j=1}^{C_T}\{MI(t_i, C_j)\} \tag{4}$$

Information gain (IG) is a metric, which gives more weight to negative terms as compared to common terms. Strong com-

mon and negative terms are weighted more than rare terms. IG performs poorly for a data set having large number of redundant features [48]. Information gain is given in Eq. (5).

$$\mathrm{IG}(t_i, C_j)_C = -P(C_j) \times \log_2 P(C_j) \quad \text{(a)}$$

$$\mathrm{IG}(t_i, C_j)_{p(C_j|t_i)} = P(t_i) \times P(C_j|t_i) \times \log_2 P(C_j|t_i) \quad \text{(b)}$$

$$\mathrm{IG}(t_i, C_j)_{p(C_j|\overline{t_i})} = P(\overline{t_i}) \times P(C_j|\overline{t_i}) \times \log_2 P(C_j|\overline{t_i}) \quad \text{(c)}$$

$$\mathrm{IG}(t_i, C_j) = (a) + (b) + (c) \quad (5)$$

Distinguishing feature selector (DFS) selects most prominent and strong rare features [1] but can neglect negative features. The mathematical expression of DFS is given in Eq. (6).

$$\mathrm{DFS}(t_i) = \sum_{j=1}^{C_T} \frac{p(C_j|t_i)}{p(t_i|\overline{C_j}) + p(\overline{t_i}|C_j) + 1} \quad (6)$$

Bi-normal separation (BNS) is the two threshold criterion between *tpr* and *fpr*, calculating normal inverse cumulative distribution function of these values a.k.a *z*-score [49] and is given in Eq. (7). It gives more importance to rare, weak rare and sparse terms. Also, negative terms are assigned more weight as compared to common terms. The BNS metric locally gives more importance to strong rare features. In case strong rare terms are not present, then more importance is given to sparse terms due to which the BNS metric's performance deteriorates.

$$\begin{aligned}\mathrm{BNS}(t_i) &= \sum_{j=1}^{C_T} \mid F^{-1}(tpr) - F^{-1}(fpr) \mid \\ &= \sum_{j=1}^{C_T} \mid F^{-1}(p(t_i|C_j)) - F^{-1}(p(t_i|\overline{C_j})) \mid\end{aligned} \quad (7)$$

Normalized difference measure (NDM) is an improvement over ACC2 proposed by [50]. It differentiates two terms having the same difference of tpr and fpr, by considering the minimum of tpr and fpr in the denominator. It is given by Eq. (8). NDM assigns correct labels to negative, and rare but for common terms it is biased toward larger classes. However, in large and highly skewed data, highly sparse terms are present in one or both the classes. NDM assigns high scores to sparse terms.

$$\begin{aligned}\mathrm{NDM} &= \frac{\mid tpr - fpr \mid}{\min(tpr, fpr)} = \frac{\mid (p(t_i|C_j)) - (p(t_i|\overline{C_j})) \mid}{\min(tpr, fpr)} \\ &= \frac{\mid recall - (p(t_i|\overline{C_j})) \mid}{\min(tpr, fpr)}\end{aligned} \quad (8)$$

Odds ratio (OR) is another well-known metric and is given in Eq. (9) [51]. Due to the $\log_2$ function, it is a one-sided local metric assigning scores to features in both positive and negative ranges. It gives more importance to negative and rare terms than common features.

$$\mathrm{OR}(t_i) = \sum_{j=1}^{C_T} \log_2 \frac{p(t_i|C_j)(1 - p(t_i|\overline{C_j}))}{(1 - p(t_i|C_j))p(t_i|\overline{C_j})} \quad (9)$$

2.3 The Text Categorization Pipeline

The general steps involved in automatic text categorization are shown in Fig. 1, which has been adopted from [52]. First, the text documents are pre-processed. In the second step, weight of the terms is calculated. The third step applies an FS algorithm. In the fourth and final step, quality of terms selected by the FS algorithm is tested using a classifier.

In Fig. 1, LFSM refers to the local feature selection metric and GFSM means a global feature selection metric. The pipeline remains the same even for global feature selection schemes, including ensemble-based IGFSS [7] and VGFSS [31] approaches that employ both global and local FS metrics.

2.4 The VGFSS Algorithm

Now, we describe the VGFSS approach for selecting terms.

(a) After applying the pre-processing step on the data set split the word-set into D_{train} and D_{test} sets.
(b) Apply the FS {IG, DFS, BNS, NDM, OR} metrics on vocabulary of D_{train} set to obtain the feature set and arrange the feature set in descending order of their global scores. Global score is obtained by combining local scores of the features.

$$global\_score(t_i) = GFSS\ (t_i, C_j) \quad (10)$$

(c) Assign the class label of terms based on the max local policy of GFSS metrics.

$$\mathrm{label}(t_i) = C_n = \max(\mathrm{GFSS}(t_i, C_j)) \quad (11)$$

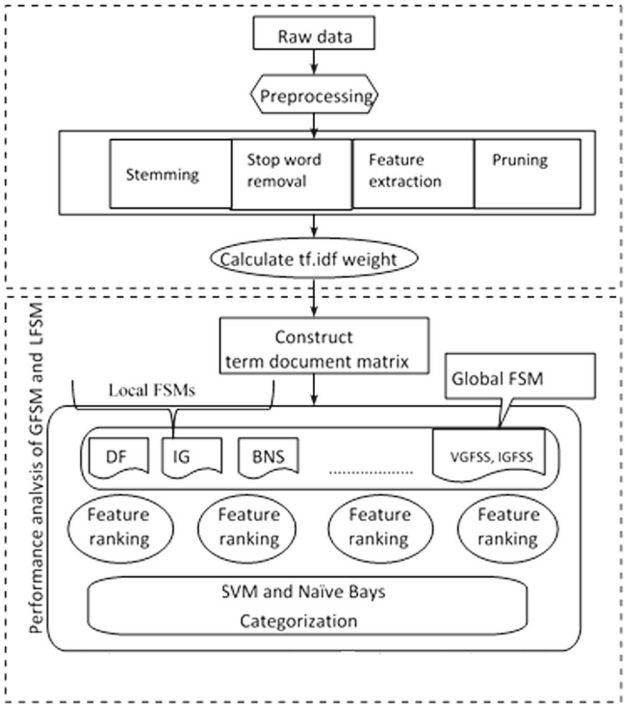

Fig. 1 The general pipeline steps for text classification

(d) Count label of each class and total terms (tt) in the vocabulary

$$\text{Count}(C_j) = \sum_{j=1}^{C_T} \text{count\_if}(\text{label}(t_i)) \tag{12}$$

$$tt = \sum_{j=1}^{C_T} \text{Count}(C_j) \tag{13}$$

(e) Let the length of the final feature set is N, then the variable split criterion for each class is:

$$var\_split(C_j) = \frac{Count(C_j) \times N}{tt} \tag{14}$$

(f) Selecting features from each class according to the var_split criterion and total count of the selected features is equal to or less than N number of features.

$$\begin{aligned} &\text{final\_feature\_set(FSS)} = N \\ &\text{if (sizeof(FSS)} > N) then \\ &N_{\text{extra}} = \text{sizeof(FSS)} - N \end{aligned} \tag{15}$$

(g) Arrange the features in descending order of their scores and eliminate N_{extra} features from the bottom.

Unlike VGFSS, IGFSS employs the OR metric to assign labels in which case we count negative and positive features of each class and calculate the positive feature ratio *(pfr)* and negative feature ratio *(nfr)* and select equal number of features from each class.

The VGFSS algorithm, which selects positive and negative terms from the classes based on their size, has shown promising results to address the text classification class imbalance problem. However, performance of VGFSS is highly dependent on the metric it employs. So, one question that arises is which metric in this VGFSS approach will produce the best results? Different metrics rank the features differently [41]. Hence, different metrics can represent each class with different number of features. If the metric models a class with less number of features, then alternatively the least number of features will be selected by both global FS metric and VGFSS approach for that class. For example, [31] use a synthetic data set that consists of three classes with different sizes {C1 = 2, C2 = 2, C3 = 4 } to investigate the VGFSS approach. When the IG metric was used as part of VGFSS, it represented the three classes with 9, 4 and 3 features, respectively, thus selecting a larger number of features from the smaller class C1 and selecting a smaller number of features from the larger class C3. This is in contradiction of the concept introduced by [31] that the largest class should get the largest number of features in the final set. This behavior varies from metric to metric and motivates us to propose a new metric.

3 Our Proposed Feature Ranking Metric: Inherent Distinguished Feature Selector (IDFS)

In this section, firstly we present a new metric that has a better capability of selecting the most relevant features in imbalanced data sets. Secondly, we illustrate the shortcomings of the existing feature ranking metrics with the help of a synthetic data set and show how our metric addresses those problems.

Researchers while proposing feature ranking metrics have described different important criteria for the selection of most useful terms of text data. For example, according to [1], a term present in some of the categories is important and should be assigned a higher score, i.e., a negative term should be assigned a higher score compared to common terms. Keeping these criteria in mind, we propose a new metric named inherent distinguished feature selector (IDFS) that inherently looks for features which are useful in any sense like rare, negative and common terms. IDFS assigns a higher score to negative terms as compared to common terms. Its mathematical expression is given in Eq. 16. The numerator of IDFS is a product of the precision, $P(C_j|t_i)$ and the ACC2 metric, where $P(C_j|t_i)$ is the probability of a class when a term is present. It also dictates global class worth independent of

classes, thus giving more weight to strong rare terms. In the denominator, probability of absence of a term when a class is present, and the presence of a term when other classes are present ($P(\overline{t_i}|C_j) + P(t_i|\overline{C_j})$) is used. This assigns a lower score to the irrelevant terms present in most of the classes. The ACC2 metric is included in the equation so as to select only those features near either fpr or tpr axis according to criterion [13]. An ϵ, which is of a small value such as 0.0005 is added in the denominator to further discriminate rare terms from the rest of the term categories.

$$\mathrm{IDFS}(t_i, C_j) = \frac{P(C_j|t_i) \times \mid P(t_i|C_j) - P(t_i|\overline{C_j}) \mid}{P(\overline{t_i}|C_j) + P(t_i|\overline{C_j}) + \epsilon} \tag{16}$$

To illustrate the shortcomings of existing well-known metrics and working of our IDFS metric, we refer to an example synthetic data set shown in Table 3. It consists of 10 documents and 12 unique terms. Among the terms, two ("lion," "usb") are rare terms, three ("mouse," "cat," "water") are common terms, six ("dog," "fridge," "laptop," "cow," "phone," "apple") are negative terms, and one ("bulb") is a sparse term. The three classes are named "house," "wild," and "technology" or "tech," and the size of each of these classes is {C1 = 2, C2 = 3, C3 = 5}, respectively. Table 4 shows the vector space model for this example. For simplicity, we have considered terms with single frequency in the synthetic data set. Furthermore, a feature is assigned that class label having greater tpr value.

Next, we show how the IDFS expression given in Eq. 16 is used to calculate the scores for the 12 features.

$$\begin{aligned}
\mathrm{IDFS(mouse)} &= \frac{\frac{2}{9}\left(\frac{2}{2}-\frac{7}{8}\right)}{\frac{0}{2}+\frac{7}{8}+\epsilon} + \frac{\frac{3}{9}\left(\frac{3}{3}-\frac{6}{7}\right)}{\frac{0}{3}+\frac{6}{7}+\epsilon} + \frac{\frac{4}{9}\left(\frac{4}{5}-\frac{5}{5}\right)}{\frac{1}{5}+\frac{5}{5}+\epsilon} \\
&= \sum \begin{Bmatrix} 0.032 & 0.056 & 0.074 \\ \text{house} & \text{wild} & \text{tech} \end{Bmatrix} = 0.162 \\
\mathrm{IDFS(cat)} &= \frac{\frac{2}{4}\left(\frac{2}{2}-\frac{2}{8}\right)}{\frac{0}{2}+\frac{2}{8}+\epsilon} + \frac{\frac{3}{4}\left(\frac{3}{3}-\frac{1}{7}\right)}{\frac{0}{3}+\frac{1}{7}+\epsilon} + \frac{\frac{0}{4}\left(\frac{0}{5}-\frac{4}{5}\right)}{\frac{5}{5}+\frac{4}{5}+\epsilon} \\
&= \sum \begin{Bmatrix} 0.036 & 4.484 & 0.0 \\ \text{house} & \text{wild} & \text{tech} \end{Bmatrix} = 4.515 \\
\mathrm{IDFS(dog)} &= \frac{\frac{1}{4}\left(\frac{1}{2}-\frac{3}{8}\right)}{\frac{1}{2}+\frac{3}{8}+\epsilon} + \frac{\frac{3}{4}\left(\frac{3}{3}-\frac{1}{7}\right)}{\frac{0}{3}+\frac{1}{7}+\epsilon} + \frac{\frac{0}{4}\left(\frac{0}{5}-\frac{4}{5}\right)}{\frac{5}{5}+\frac{4}{5}+\epsilon} \\
&= \sum \begin{Bmatrix} 0.036 & 4.484 & 0.0 \\ \text{house} & \text{wild} & \text{tech} \end{Bmatrix} = 4.515 \\
\mathrm{IDFS(fridge)} &= \frac{\frac{1}{4}\left(\frac{1}{2}-\frac{3}{8}\right)}{\frac{1}{2}+\frac{3}{8}+\epsilon} + \frac{\frac{0}{4}\left(\frac{0}{3}-\frac{4}{7}\right)}{\frac{3}{3}+\frac{4}{7}+\epsilon} + \frac{\frac{3}{4}\left(\frac{3}{5}-\frac{1}{5}\right)}{\frac{2}{5}+\frac{1}{5}+\epsilon} \\
&= \sum \begin{Bmatrix} 0.036 & 0.0 & 0.5 \\ \text{house} & \text{wild} & \text{tech} \end{Bmatrix} = 0.536 \\
\mathrm{IDFS(laptop)} &= \frac{\frac{1}{5}\left(\frac{1}{2}-\frac{4}{8}\right)}{\frac{1}{2}+\frac{4}{8}+\epsilon} + \frac{\frac{0}{5}\left(\frac{0}{3}-\frac{5}{7}\right)}{\frac{3}{3}+\frac{5}{7}+\epsilon} + \frac{\frac{4}{5}\left(\frac{4}{5}-\frac{1}{5}\right)}{\frac{1}{5}+\frac{1}{5}+\epsilon} \\
&= \sum \begin{Bmatrix} 0.0 & 0.0 & 1.99 \\ \text{house} & \text{wild} & \text{tech} \end{Bmatrix} = 1.99 \\
\mathrm{IDFS(water)} &= \frac{\frac{1}{4}\left(\frac{1}{2}-\frac{3}{8}\right)}{\frac{1}{2}+\frac{3}{8}+\epsilon} + \frac{\frac{2}{4}\left(\frac{2}{3}-\frac{2}{7}\right)}{\frac{1}{3}+\frac{2}{7}+\epsilon} + \frac{\frac{1}{4}\left(\frac{1}{5}-\frac{3}{5}\right)}{\frac{4}{5}+\frac{3}{5}+\epsilon} \\
&= \sum \begin{Bmatrix} 0.036 & 0.307 & 0.071 \\ \text{house} & \text{wild} & \text{tech} \end{Bmatrix} = 0.415 \\
\mathrm{IDFS(cow)} &= \frac{\frac{1}{2}\left(\frac{1}{2}-\frac{1}{8}\right)}{\frac{1}{2}+\frac{1}{8}+\epsilon} + \frac{\frac{1}{2}\left(\frac{1}{3}-\frac{1}{7}\right)}{\frac{2}{3}+\frac{1}{7}+\epsilon} + \frac{\frac{0}{2}\left(\frac{0}{5}-\frac{2}{5}\right)}{\frac{5}{5}+\frac{2}{5}+\epsilon} \\
&= \sum \begin{Bmatrix} 0.3 & 0.117 & 0.0 \\ \text{house} & \text{wild} & \text{tech} \end{Bmatrix} = 0.417 \\
\mathrm{IDFS(phone)} &= \frac{\frac{0}{3}\left(\frac{0}{2}-\frac{3}{8}\right)}{\frac{2}{2}+\frac{3}{8}+\epsilon} + \frac{\frac{1}{3}\left(\frac{1}{3}-\frac{2}{7}\right)}{\frac{2}{3}+\frac{2}{7}+\epsilon} + \frac{\frac{2}{5}\left(\frac{2}{5}-\frac{1}{5}\right)}{\frac{3}{5}+\frac{1}{5}+\epsilon} \\
&= \sum \begin{Bmatrix} 0.0 & 0.017 & 0.167 \\ \text{house} & \text{wild} & \text{tech} \end{Bmatrix} = 0.183 \\
\mathrm{IDFS(apple)} &= \frac{\frac{0}{7}\left(\frac{0}{2}-\frac{7}{8}\right)}{\frac{2}{2}+\frac{7}{8}+\epsilon} + \frac{\frac{2}{7}\left(\frac{2}{3}-\frac{5}{7}\right)}{\frac{1}{3}+\frac{5}{7}+\epsilon} + \frac{\frac{5}{4}\left(\frac{5}{5}-\frac{2}{5}\right)}{\frac{0}{5}+\frac{2}{5}+\epsilon} \\
&= \sum \begin{Bmatrix} 0.0 & 0.013 & 1.07 \\ \text{house} & \text{wild} & \text{tech} \end{Bmatrix} = 1.083 \\
\mathrm{IDFS(lion)} &= \frac{\frac{0}{3}\left(\frac{0}{2}-\frac{3}{8}\right)}{\frac{2}{2}+\frac{3}{8}+\epsilon} + \frac{\frac{3}{3}\left(\frac{3}{3}-\frac{0}{7}\right)}{\frac{0}{3}+\frac{0}{7}+\epsilon} + \frac{\frac{0}{3}\left(\frac{0}{5}-\frac{3}{5}\right)}{\frac{5}{5}+\frac{3}{5}+\epsilon} \\
&= \sum \begin{Bmatrix} 0.0 & 2000 & 0.0 \\ \text{house} & \text{wild} & \text{tech} \end{Bmatrix} = 2000 \\
\mathrm{IDFS(usb)} &= \frac{\frac{0}{4}\left(\frac{0}{2}-\frac{4}{8}\right)}{\frac{2}{2}+\frac{4}{8}+\epsilon} + \frac{\frac{0}{4}\left(\frac{0}{3}-\frac{4}{7}\right)}{\frac{3}{3}+\frac{4}{7}+\epsilon} + \frac{\frac{4}{4}\left(\frac{4}{5}-\frac{0}{5}\right)}{\frac{1}{5}+\frac{0}{5}+\epsilon} \\
&= \sum \begin{Bmatrix} 0.0 & 0.0 & 3.99 \\ \text{house} & \text{wild} & \text{tech} \end{Bmatrix} = 3.99 \\
\mathrm{IDFS(bulb)} &= \frac{\frac{0}{2}\left(\frac{0}{2}-\frac{2}{8}\right)}{\frac{2}{2}+\frac{2}{8}+\epsilon} + \frac{\frac{0}{2}\left(\frac{0}{3}-\frac{2}{7}\right)}{\frac{3}{3}+\frac{2}{7}+\epsilon} + \frac{\frac{2}{2}\left(\frac{2}{5}-\frac{0}{5}\right)}{\frac{3}{5}+\frac{0}{5}+\epsilon} \\
&= \sum \begin{Bmatrix} 0.0 & 0.0 & 0.67 \\ \text{house} & \text{wild} & \text{tech} \end{Bmatrix} = 0.67
\end{aligned}$$

Now, we compare IDFS against the five well-known feature ranking metrics on this synthetic data set. Table 5 shows the ranking generated by each of the six metrics along with the label assignment for the 12 features.

First, we discuss the ranks assigned to the terms by feature ranking metrics. This will allow us to understand how good

Table 3 A synthetic data set

Documents	Contents						Class-label
d1	Mouse	Cat	Dog	Fridge			House
d2	Mouse	Cat	Laptop	Water	Cow		House
d3	Mouse	Dog	Water	Phone	Lion		Wild
d4	Mouse	Cat	Dog	Apple	Lion	Water	Wild
d5	Mouse	Dog	Cow	Apple	Lion		Wild
d6	Laptop	Fridge	USB	Apple			Technology
d7	Mouse	Laptop	Cat	Apple	Phone	Water	Technology
d8	Mouse	Laptop	USB	Apple	Fridge		Technology
d9	Mouse	Fridge	USB	Apple	Phone	Bulb	Technology
d10	Mouse	Laptop	USB	Apple		Bulb	Technology

Table 4 The vector space representation of the synthetic data set

Class-label	Mouse	Cat	Dog	Fridge	Laptop	Water	Cow	Phone	Apple	Lion	USB	Bulb
House	1	1	1	1	0	0	0	0	0	0	0	0
House	1	1	0	0	1	1	1	0	0	0	0	0
Wild	1	0	1	0	0	1	0	1	0	1	0	0
Wild	1	1	1	0	0	1	0	0	1	1	0	0
Wild	1	0	1	0	0	0	1	0	1	1	0	0
Technology	0	0	0	1	1	0	0	0	1	0	1	0
Technology	1	1	0	0	1	1	0	1	1	0	0	0
Technology	1	0	0	1	1	0	0	0	1	0	1	0
Technology	1	0	0	1	0	0	0	1	1	0	1	1
Technology	1	0	0	0	1	0	0	0	1	0	1	1

a metric will work as a stand-alone FS algorithm. The "lion" is a rare term (i.e., present in only one class) belonging to the "wild" class. We can see that IDFS, DFS, IG and NDM metrics rank it at the top but is assigned a lower rank by the OR and BNS metrics. The term "dog" is present in all documents of the class "wild" with only one occurrence in the minor class "house," while "usb" is a rare term of the major class "tech" with 80% presence in that class. We can observe that "dog" is assigned a second rank by IDFS and a third position by DFS, while DFS assigns second place to "usb." DFS assigns it a greater weight because it is biased toward the major class. On the other hand, IDFS also employs the tpr and fpr values to differentiate the terms, due to which the minor classes also enjoy their share during feature selection in highly skewed data sets. The IG metric assigns a second rank to "dog," but the NDM, OR and BNS metrics assign a relatively lower score to it. The third important term is "usb" that is placed at the first position by OR, at second position by NDM, BNS and DFS while at third position by IDFS and IG metrics. Although the placement of "usb" at second position is the best for a balanced data set but for a skewed data set, the term "dog" should be placed higher so that rare terms of the minor class can be distinguishable. Next is "cat," which is a common term with 33% presence in the class "wild" and 20% presence in the major class "tech" while 100% presence in the minor class "house." IDFS is the only metric that ranks it at the fourth position, while the other FS metrics assign it a lower rank. If a data set is highly unbalanced and rare terms of minor classes are not ranked properly, then such features may be left out in the training procedure. Thus, the incoming test documents would not be classified properly, causing more false negative predictions than false positive ones, which is highly undesirable [53]. Therefore, "cat" should be placed higher than term "laptop" because it is a rare term of the minor class. All metrics except DFS assign a lower score to "laptop" because DFS favors terms with higher probabilities in the major class. The term "apple" is a rare term for the major class with 100% distribution, a rare term for the second class with 67% distribution but a negative term for the minor class "house." It should be assigned a lower score compared to "laptop" because it is strongly related to the two big classes. However, only IDFS and DFS rank it correctly, while the other metrics are ranking "apple" higher than "laptop." The term "bulb" has 2/5 distribution in the major class and is a weak rare term of the major class. It should be ranked lower than "cat" because it is a common term and a rare term of the

Table 5 The ranking of terms of the synthetic data set generated by different metrics

IDFS			DFS			IG			NDM			OR			BNS		
Terms	Rank	Label	Terms	Rank	Label	Terms	Rank	Label	Terms	Rank	Label	Terms	Rank	Label	Terms	Rank	Label
Lion	2000	Wild	Lion	1	Wild	Lion	0.88	Wild	Lion	3947	Wild	USB	17.872	Tech	Mouse	12.93	House
Dog	4.515	Wild	USB	0.833	Tech	Dog	0.769	Wild	USB	3739	Tech	Lion	17.856	Wild	USB	10.892	Tech
USB	3.99	Tech	Dog	0.789	Wild	USB	0.609	Tech	Bulb	1869	Tech	Bulb	14.946	Tech	Bulb	8.378	Tech
Cat	1.59	House	Laptop	0.671	Tech	Apple	0.605	House	Apple	1750.57	House	Dog	13.485	Wild	Apple	7.611	House
Laptop	1.199	Tech	Apple	0.65	Tech	Laptop	0.439	Wild	Dog	1605.326	Tech	Apple	12.702	House	Dog	6.675	Tech
Apple	1.083	Tech	Bulb	0.625	Tech	Cat	0.334	House	Laptop	1430	Wild	Mouse	12.082	Tech	Lion	6.516	Tech
Bulb	0.666	Tech	Cat	0.623	House	Fridge	0.284	Wild	Fridge	1143.333	Wild	Laptop	9.878	Wild	Laptop	5.539	Wild
Fridge	0.536	Tech	Fridge	0.602	Tech	Cow	0.246	Tech	Cow	803.329	Tech	Cow	9.471	Tech	Fridge	4.883	Wild
Cow	0.417	House	Cow	0.584	House	Bulb	0.235	Wild	Phone	750.164	House	Cat	9.258	House	Cow	4.822	Tech
Water	0.415	Wild	Water	0.546	Wild	Water	0.133	Tech	Cat	5.288	House	Fridge	9.012	Wild	Cat	3.964	House
Phone	0.183	Tech	Phone	0.542	Tech	Phone	0.119	House	Water	3.665	Tech	Phone	6.914	House	Phone	3.693	House
Mouse	0.162	Tech	Mouse	0.499	Tech	Mouse	0.108	Wild	Mouse	0.56	Tech	Water	5.36	Tech	Water	2.411	Tech

minor class (100% distributed in the class) and a sparse term for the two big classes. If we were to select the top 5 features based on either metric except IDFS for classifying the test documents, then "cat" a rare term of the minor class would have been left out, thus making it difficult for the classifier to predict the appropriate category of the new instances. The remaining five terms ("fridge," "cow," "water," "phone" and "mouse") are not useful for discriminating the classes and should be positioned in the order as ranked by IDFS and DFS. However, this is not the case. For example, OR and BNS assign "mouse," which is an irrelevant term a higher rank.

For this data set, the difference between IDFS and DFS seems to be minor. The term "cat," which is strongly related to the minor class "house," has obtained more importance by IDFS as compared to "laptop," which is strongly related to the major class "tech." Actually, DFS gives more weight to the features of the major class as compared to the rare features of the minor classes. The IDFS metric does not neglect rare features from the minor classes. Due to this reason, IDFS is expected to perform better during feature selection of imbalanced data sets as compared to DFS and other metrics. The time complexity of the IDFS metric is $O(n)$ and is the same as that of DFS.

The shortcomings of existing FS metrics can be summarized as below.

- The DFS metric ranks features of the major class higher than those of the minor class. Therefore, features of the minor class can be expected to be positioned at the bottom of the ranking in highly skewed data sets such Reuters52.
- The NDM denominator is $\min(tpr, fpr)$, but when both tpr and fpr approach zero the irrelevant features are assigned higher scores. The minor class rare features having minor distribution in the major classes can obtain a low score. This can be seen from Table 5, the "cat" feature which is a rare feature of the minor class "house," but having a minor distribution in other classes is assigned a lower score.
- For the OR metric, the presence of the conditional probability $tpr = p(t_i|C_j)$ in the numerator and its inverse in the denominator ranks the terms incorrectly. When a term is present in all classes such as the term "mouse," which is highly undesirable. Its tpr inverse value, which is negligible makes the denominator very small, thus assigning an overall high score to a common term.
- As BNS is a difference between inverse distributions of tpr and fpr, a term having a higher tpr in all classes gets a higher weight and rank when its fpr value is also comparable. This is highly undesirable. For example, due to highest values of tpr and fpr of "mouse" (a common term present in all classes) in Table 5, BNS assigns the highest score to it.
- IG ranks negative terms higher as compared to the common terms. For example, "laptop" is a negative term of the "wild" class and is placed higher than the common term "cat."

Now, we look at how well the features are assigned class labels, which are crucial for the working of VGFSS and IGFSS frameworks as mentioned toward the end of Sect. 2.

The label assignment performed by different metrics is computed with the same mechanism as used by IDFS, which has been shown above. We can observe that the label assignments of the features by IDFS and DFS metrics are better as compared to the rest of the metrics and are in accordance to our expectations as depicted in Table 4. The IG metric assigns an incorrect label to "bulb" and "mouse," i.e., "wild" class. Similarly, "cow" is assigned to the "tech" class, which is also wrong. In case of NDM, we can observe that it associates the "apple" term to the incorrect label "house," which is correctly assigned to the "tech" class by IDFS and DFS metrics. The IG, BNS and OR metrics each incorrectly labels "apple" to the "house" class. The term "laptop," which is a negative term for class "wild," should be assigned a label of the major class. But unlike IDFS and DFS, all metrics assign a wrong label to it because these FS metrics mostly assign labels on the basis of the negative classes. We can find that the IG, BNS and OR metrics could not label the correct category to most of the terms, while NDM assigns the correct labels to some of the features. IDFS and DFS have shown the best performance among all the metrics.

Hence, from our above discussion we can conclude that in comparison with the well-known FS metrics IDFS is not only better at ranking the terms but also assigns correct class labels to the them.

4 Empirical Evaluation

This section describes the experimental settings used for the empirical evaluation of the newly proposed IDFS metric.

4.1 Description of the Data Sets

In our experiments, five data sets have been used that are widely used by researchers. Table 6 shows the summary of the data sets. For each data set, we show total number of documents, number of terms, number of classes, sizes of the largest and smallest classes and the class names. All the data sets except for the Classic4 data set have been obtained from the well-known webpage of Ana Cardoso Cachopo[4] and their details are given in [54]. The Classic4 data set was

[4] http://ana.cachopo.org/datasets-for-single-label-text-categorization.

Table 6 Summary of data sets

Data set	Total docs	Total terms	Total classes	Largest class (instances)	Smallest class (instances)	Classes
20NG	7905	7902	20	Rec.sport.hockey (630)	Talk.religion.misc (377)	Alt.atheism, comp.graphics, comp.os.ms-windows.misc, comp.sys.ibm.pc.hardware, comp.sys.mac.hardware, comp.windows.x, misc.forsale, rec.autos, rec.motorcycles, rec.sport.baseball, rec.sport.hockey, sci.crypt, sci.electronics, sci.med, sci.space, soc.religion.christian, talk.politics.guns, talk.politics.mideast, talk.politics.misc, talk.religion.misc
WebKB	2810	7290	4	Student (1104)	Project (336)	Student, faculty, project, course
Reuters8	5485	13280	8	Earn (2840)	Grain (41)	Earn, acq, trade, ship, grain, crude, interest, money-fx
Classic4	5676	8698	4	Cacm (2553)	Med (810)	Cacm, cisi, cran, med
Reuters52	6532	5892	52	Earn (2840)	Platinum (1)	Trade, grain, ship, gold, acq, tin, ipi, earn, jobs, zinc sugar, cpi, nickel, money-fx, heat, interest, cocoa coffee, crude, cotton, livestock, money-supply, copper alum, rubber, nat-gas, reserves, bop, gnp, lead, dlr, potato iron-steel, orange, retail, strategic-metal, fuel, meal-feed wpi, tea, lei, gas, pet-chem, lumber, veg-oil, carcass, instal-debt income, platinum, jet, housing, cpu

obtained from the webpage of Volkan Tunali.[5] These data sets have different characteristics. For example, the WebKB data set has four categories and its classes are almost balanced, while R52 consists of 52 classes and its classes are highly imbalanced.

4.2 Data Sets Pre-processing

We stemmed all the data sets used in our experiments and removed the stop words with the help of the Weka APIs [55]. The multi-class text classification data sets were converted into two-class classification tasks using the one-vs-rest strategy, as it is most widely used by text classification researchers [13].

4.3 Feature Ranking Metrics

We compare the performance of five well known feature ranking metrics, namely IG, DFS, BNS, NDM and OR against our newly proposed IDFS metric. These metrics were implemented in the JAVA programming language.

4.4 Classification Algorithms

The quality of terms selected by the feature ranking metrics has been evaluated with the help of two well-known classification algorithms, namely support vector machines (SVM) [32] and random forests (RF) [33]. The implementation of these classifiers in Weka has been used. Weka uses the JAVA programming environment [55]. For RF, we have selected the depth of the tree to be 100, and the number of features, $n_{\text{try}} = \sqrt{n} * 2$, where n is the total number features in the training set.

4.5 Evaluation Measures

The macro averaged F_1 and micro averaged F_1 measures are the most commonly used evaluation measures to estimate the performance of a text classifier [13].

The F_1 measure is defined to be the harmonic mean of precision (p) and recall (r). We can estimate precision by considering the number of correct results out of the results marked correct by the classifier (i.e., $p = \frac{tp}{tp+fp}$), while recall is the number of correct results out of actual number of correct results (i.e., $r = \frac{tp}{tp+fn}$) [10]. Higher the F_1 value, better is the performance of the classifier. The best value of F_1 is 1, and its worst value is zero.

The micro F_1 is given in Eq. (17), while macro F_1 is given in Eq. (18). The micro F_1 does not take class distribution into account, and it tends to be biased toward large classes, i.e., favoring features of the larger classes. On the other hand, the macro F_1 considers class distributions and locally precision and recall values of individual classes, i.e., if a feature is rare and belongs to the minor class, then that feature might be getting a higher rank [52]. We use C_T to denote the total number of classes.

$$Micro\ Averaged\ F_1 = \frac{2 \times precision^{\mu} \times recall^{\mu}}{precision^{\mu} + recall^{\mu}} \tag{17}$$

where $precision^{\mu}$ and $recall^{\mu}$ are estimated as,

$$precision^{\mu} = \frac{\sum_{i=1}^{C_T} tp_i}{\sum_{i=1}^{C_T} (tp_i + fp_i)}$$

$$recall^{\mu} = \frac{\sum_{i=1}^{C_T} tp_i}{\sum_{i=1}^{C_T} (tp_i + fn_i)}$$

$$Macro\ Averaged\ F_1 = \frac{\sum_{k=1}^{C_T} \frac{2 \times p_k \times r_k}{p_k + r_k}}{C_T} \tag{18}$$

where p_k is the precision and r_k is the recall for the k^{th} class.

4.6 Evaluation Procedure

Each data set is randomly split into two sets, namely the training and test sets. The former contains 70% of the documents, while the latter has the remaining 30% documents. The data sets were pre-processed, and then, we applied feature selection algorithms on the training data set and generated a ranked list of terms in the decreasing order of importance. From this ranked list, we construct 4 nested subsets by progressively adding terms of decreasing importance to find an optimal smaller subset of terms. The size of these subsets is 100, 400, 700 and 1500 terms. The quality of each subset is evaluated by training two classifiers (SVM and RF) and measuring the macro F_1 and micro F_1 values on the test set.

To see how good IDFS is as a stand-alone global FS algorithm, we compare it against IG, DFS, BNS, NDM and OR metrics by repeating the above procedure for each of these feature ranking metrics. For each metric, we sum the local class-based scores of the metric to obtain a final global score for a term. In this scenario, IDFS is compared against a metric for $5_{datasets} \times 4_{subsets} \times 2_{classifiers} = 40$ subsets.

To see which metric performs the best with the IGFSS and VGFSS ensemble frameworks, we also evaluate the working of IDFS, IG, DFS, BNS, NDM and OR metrics as a part of them. In this scenario, IDFS is evaluated and compared against a metric for $2_{approaches} \times 5_{datasets} \times 4_{subsets} \times 2_{classifiers} = 80$ subsets.

[5] http://www.dataminingresearch.com/index.php/2010/09/classic3-classic4-datasets/.

5 Results and Discussion

This section presents the macro and micro F_1 values obtained by the top ranked terms of the six feature ranking (BNS, IDFS, DFS, IG, NDM and OR) metrics in two scenarios: when each metric is used as a stand-alone FS algorithm and when each metric is used as a part of the IGFSS and VGFSS ensemble frameworks. In the tables to follow, for a given subset we mark the maximum F_1 value of a metric(s) bold and underline that bold F_1 of a metric(s) whose value is the highest for a given classifier instance.

5.1 IDFS as a Stand-Alone Global Feature Selection Algorithm

Let us look at the performance of the IDFS metric when it works as a stand-alone FS algorithm. Table 7 shows the macro and micro F_1 values of IDFS and the other five metrics on the five data sets with SVM and RF classifiers.

First, we look at results obtained with the SVM classifier. We can observe that the highest micro and macro F_1 values are 0.770 and 0.798 on the 20NG data set, respectively. These values are attained by the subsets consisting of top 1500 ranked terms of IDFS. On the WebKB data set, the top 700 ranked terms selected by IDFS have resulted in the highest micro F_1 (0.907) and macro F_1 (0.890) values. In the case of the unbalanced Reuters8 data set, the subsets with top 1500 and top 400 ranked terms of IDFS have produced the highest micro F_1 and macro F_1 values, which are 0.980 and 0.941, respectively. Similarly, we can see that on the highly skewed Reuters52 data set, the highest values of micro and macro F_1 values are 0.943 and 0.613, respectively, and that have been achieved by subsets of top 700 ranked terms. Finally, it can be found that a subset with top 1500 ranked terms of IDFS have resulted in the highest micro and macro F_1 values, which are 0.985 and 0.975 on the Classic4 data set, respectively.

In the next half of Table 7, we present the performance shown by the RF classifier on the five data sets after evaluating the terms selected by BNS, IDFS, DFS, IG, NDM and OR metrics. On the 20NG data set, we can see 0.803 and 0.798 are the highest micro and macro F_1 values, respectively, which are achieved by IDFS. The highest classification accuracies on the WebKB data set are 0.901 and 0.889 and are attained by subsets of IDFS. For the Reuters8 data set, RF classifier exhibits the best performance with micro and macro F_1 (0.974 and 0.919) with IDFS subsets. Similarly, subsets of IDFS have obtained the highest micro and macro F_1 values 0.890 and 0.532, respectively, on Reuters52. Finally, the performance (0.975 and 0.969) of IDFS is shown to be the best on Classic4.

From the above results, we can observe that in balanced data sets with few categories the OR and DFS metrics have a comparable performance, while BNS performs better than NDM. Overall, we can conclude that our newly proposed IDFS as a stand-alone FS algorithm has a better capability of selecting the useful terms in comparison with BNS, DFS, IG, NDM and OR metrics on all the data sets with different characteristics.

5.2 IDFS as a Part of the IGFSS and VGFSS Frameworks

Next, we look at how effective IDFS is in comparison with BNS, DFS, IG, NDM and OR metrics when each of these metrics becomes a part of the two well-known ensemble approaches, namely IGFSS and VGFSS. The former was found not to work well for imbalanced data sets having a large number of classes of varying sizes.

First, we compare the performance of the six metrics as part of the IGFSS framework. The first half of the Table 8 shows results using SVM classifier, while the second half consists of results with RF classifier. On all the data sets with balanced and imbalanced classes, we can observe that both the classifiers show the highest micro and macro F_1 values when IDFS is used as a part of the IGFSS framework.

Now, the performance of VGFSS with the six metrics on the five data sets is shown in Table 9. When we analyze the results we find that the highest micro and macro F_1 values in almost all the cases are obtained when VGFSS employs IDFS. There is one instance of the RF classifier when VGFSS and NDM have shown the highest macro F_1 value.

Hence, we can conclude that performance of IGFSS and VGFSS ensemble approaches is improved when the newly proposed IDFS metric is used to select the terms.

5.3 Discussion of the Results

The text classification performance greatly depends on the FS metric being used to select the most useful terms. The presence of strong rare and strong negative terms in the final subset increases the performance of a text classifier. On the other hand, if weak rare and sparse terms are used to train a classifier, then performance of the classifier degrades on the test set. Deterioration in the performances of the SVM and RF classifiers was observed when sparse terms were added in the final feature set.

The **BNS** metric was found to give more importance to rare, weak rare and sparse terms. Also, negative terms are assigned higher score as compared to common terms. For strong rare terms, it is biased toward the largest class. Furthermore, when used with the ensemble approaches BNS assigns a wrong number of features to each class due to which skewed classes get very few or almost zero features. This causes the useful candidate features for that class to be pushed downward in the ranked list and are unable to get selected in the final feature set. The BNS metric performs better for bal-

Table 7 The micro and macro F_1 values of six FS metrics as stand-alone FS algorithms on five data sets

Dataset	Subset	(IG)	SVM micro F_1			(NDM)	(OR)	(IG)	SVM macro F_1			(NDM)	(OR)
			(DFS)	(BNS)	(IDFS)				(DFS)	(BNS)	(IDFS)		
20NG	**100**	0.552	0.552	0.493	**0.573**	0.483	0.507	0.582	0.590	0.546	**0.608**	0.534	0.540
	400	0.675	0.667	0.610	**0.689**	0.606	0.664	0.704	0.703	0.654	**0.711**	0.647	0.694
	700	0.708	0.698	0.645	**0.716**	0.639	0.699	0.736	0.738	0.690	**0.748**	0.684	0.727
	1500	0.751	0.744	0.724	**0.770**	0.702	0.748	0.775	0.775	0.754	**0.798**	0.738	0.772
Webkb	**100**	0.842	0.832	0.803	**0.874**	0.663	0.818	0.833	0.820	0.768	**0.836**	0.630	0.801
	400	0.857	0.862	0.849	**0.881**	0.758	0.863	0.838	0.845	0.830	**0.862**	0.739	0.845
	700	0.878	0.880	0.865	**0.907**	0.774	0.874	0.864	0.867	0.845	**0.890**	0.759	0.859
	1500	0.874	0.883	0.871	**0.896**	0.820	0.886	0.861	0.871	0.856	**0.880**	0.799	0.875
Reuters8	**100**	0.934	0.934	0.923	**0.943**	0.894	0.936	0.857	0.863	0.828	**0.912**	0.763	0.867
	400	0.963	0.966	0.949	**0.969**	0.924	0.959	0.912	0.920	0.890	**0.941**	0.816	0.906
	700	0.965	0.971	0.954	**0.974**	0.929	0.967	0.911	0.927	0.897	**0.937**	0.831	0.913
	1500	0.963	0.967	0.960	**0.980**	0.931	0.965	0.911	0.909	0.903	**0.925**	0.828	0.916
Reuters52	**100**	0.831	0.831	0.823	**0.884**	0.815	0.821	0.313	0.518	0.301	**0.551**	0.254	0.264
	400	0.902	0.906	0.887	**0.936**	0.877	0.869	0.490	0.574	0.466	**0.609**	0.435	0.408
	700	0.914	0.916	0.903	**0.943**	0.911	0.888	0.554	0.596	0.503	**0.613**	0.567	0.454
	1500	0.919	0.920	0.918	**0.939**	0.923	0.918	0.582	0.585	0.551	**0.605**	0.600	0.551
Classic4	**100**	0.920	0.913	0.901	**0.927**	0.870	0.906	**0.917**	0.909	0.901	0.912	0.869	0.904
	400	0.956	0.953	0.942	**0.972**	0.932	0.956	0.957	0.955	0.944	**0.963**	0.935	0.956
	700	0.958	0.959	0.956	**0.978**	0.937	0.963	0.957	0.959	0.957	**0.969**	0.939	0.962
	1500	0.966	0.964	0.961	**0.985**	0.957	0.965	0.966	0.964	0.961	**0.975**	0.958	0.965
			Random Forest micro F_1						**Random Forest macro** F_1				
20NG	**100**	0.615	0.604	0.521	**0.628**	0.521	0.588	0.612	0.623	0.569	**0.634**	0.564	0.603
	400	0.738	0.720	0.673	**0.749**	0.684	0.736	0.733	0.716	0.677	**0.744**	0.688	0.732
	700	0.752	0.751	0.720	**0.782**	0.718	0.756	0.748	0.746	0.722	**0.776**	0.722	0.750
	1500	0.760	0.779	0.792	**0.803**	0.777	0.790	0.757	0.774	0.788	**0.798**	0.776	0.786
Webkb	**100**	0.863	0.877	0.790	**0.895**	0.668	0.846	0.848	0.862	0.748	**0.875**	0.636	0.820
	400	0.870	0.875	0.864	**0.901**	0.769	0.871	0.858	0.863	0.845	**0.889**	0.746	0.856
	700	0.867	0.867	0.880	**0.893**	0.783	0.863	0.850	0.852	0.869	**0.879**	0.757	0.850
	1500	0.851	0.855	0.871	**0.878**	0.830	0.856	0.829	0.839	**0.859**	0.859	0.809	0.840
Reuters8	**100**	0.945	0.920	0.937	**0.972**	0.902	0.949	0.889	0.878	0.864	0.884	0.752	**0.894**
	400	0.953	0.930	0.956	**0.974**	0.938	0.953	0.894	0.878	0.899	**0.919**	0.852	0.880
	700	0.947	0.934	0.952	**0.967**	0.936	0.947	0.862	0.885	0.882	**0.902**	0.844	0.865
	1500	0.925	0.940	0.934	**0.949**	0.945	0.937	0.785	**0.887**	0.805	0.825	0.862	0.814
Reuters52	**100**	**0.851**	0.824	0.833	0.841	0.824	0.838	0.376	0.507	0.338	**0.532**	0.258	0.337
	400	0.856	0.872	0.876	**0.890**	0.856	0.846	0.419	0.472	0.460	**0.488**	0.405	0.381
	700	0.852	0.859	0.859	**0.878**	0.850	0.842	0.418	0.423	0.426	**0.455**	0.383	0.370
	1500	0.834	0.834	0.832	**0.855**	0.833	0.834	0.370	0.385	0.386	**0.434**	0.398	0.385
Classic4	**100**	0.917	0.918	0.896	**0.950**	0.879	0.918	0.913	0.916	0.896	**0.941**	0.880	0.916
	400	0.930	0.934	0.949	**0.972**	0.934	0.941	0.931	0.933	0.950	**0.967**	0.936	0.941
	700	0.937	0.937	0.949	**0.975**	0.943	0.944	0.936	0.937	0.949	**0.969**	0.945	0.943
	1500	0.932	0.934	0.946	**0.966**	0.960	0.932	0.931	0.934	0.947	**0.962**	0.960	0.933

Table 8 The micro and macro F_1 values of six FS metrics as part of the IGFSS framework on five data sets

Dataset	Subset	IGFSS (IG)	SVM micro F_1			IGFSS (NDM)	IGFSS (OR)	IGFSS (IG)	SVM macro F_1			IGFSS (NDM)	IGFSS (OR)
			IGFSS (DFS)	IGFSS (BNS)	IGFSS (IDFS)				IGFSS (DFS)	IGFSS (BNS)	IGFSS (IDFS)		
20NG	**100**	0.547	**0.550**	0.457	0.541	0.447	0.503	0.571	**0.579**	0.502	0.561	0.487	0.536
	400	0.673	0.677	0.617	**0.690**	0.610	0.672	0.697	0.703	0.645	**0.718**	0.641	0.696
	700	0.711	0.719	0.678	**0.726**	0.666	0.704	0.730	0.737	0.703	**0.748**	0.693	0.727
	1500	0.758	0.756	0.728	**0.775**	0.721	0.758	0.779	0.773	0.750	**0.790**	0.743	0.773
Webkb	**100**	0.834	0.841	0.814	**0.864**	0.686	0.840	0.823	0.830	0.792	**0.847**	0.670	0.818
	400	0.873	0.875	0.855	**0.899**	0.755	0.874	0.856	0.859	0.834	**0.871**	0.730	0.857
	700	0.880	0.885	0.870	**0.905**	0.761	0.883	0.867	**0.873**	0.853	**0.873**	0.733	0.871
	1500	0.875	0.882	0.875	**0.903**	0.819	0.872	0.864	0.869	0.860	**0.871**	0.791	0.858
Reuters8	**100**	0.933	0.935	0.907	**0.964**	0.880	0.926	0.873	0.860	0.815	**0.911**	0.777	0.863
	400	0.959	0.963	0.946	**0.985**	0.926	0.956	0.901	0.910	0.891	**0.932**	0.838	0.906
	700	0.958	0.964	0.954	**0.988**	0.926	0.959	0.902	0.912	0.905	**0.939**	0.824	0.909
	1500	0.966	0.967	0.960	**0.989**	0.948	0.963	0.916	0.914	0.911	**0.944**	0.879	0.917
Reuters52	**100**	0.822	0.858	0.789	**0.875**	0.801	0.802	0.309	**0.568**	0.222	0.549	0.250	0.263
	400	0.889	0.903	0.870	**0.928**	0.877	0.865	0.497	0.588	0.439	**0.619**	0.455	0.415
	700	0.905	0.917	0.891	**0.937**	0.906	0.887	0.534	0.609	0.482	**0.642**	0.538	0.467
	1500	0.913	0.920	0.908	**0.939**	0.911	0.903	0.545	0.602	0.525	**0.621**	0.578	0.506
Classic4	**100**	**0.909**	0.905	0.879	0.905	0.850	0.894	**0.908**	0.902	0.874	0.903	0.851	0.891
	400	0.951	0.950	0.940	**0.963**	0.932	0.955	0.950	0.950	0.941	**0.962**	0.932	0.954
	700	0.965	0.961	0.953	**0.974**	0.946	0.968	0.965	0.961	0.954	**0.974**	0.948	0.967
	1500	0.963	0.962	0.965	**0.975**	0.962	0.967	0.962	0.962	0.966	**0.975**	0.963	0.967

Table 8 continued

Dataset	Subset	IGFSS (IG)	SVM micro F_1				IGFSS (NDM)	IGFSS (OR)	IGFSS (IG)	SVM macro F_1				IGFSS (NDM)	IGFSS (OR)
			IGFSS (DFS)	IGFSS (BNS)	IGFSS (IDFS)					IGFSS (DFS)	IGFSS (BNS)	IGFSS (IDFS)			
		IGFSS(IG)	Random Forest micro F_1					IGFSS (OR)	IGFSS (IG)	Random Forest macro F_1					IGFSS (OR)
			IGFSS (DFS)	IGFSS (BNS)	IGFSS (IDFS)	IGFSS (NDM)				IGFSS (DFS)	IGFSS (BNS)	IGFSS (IDFS)	IGFSS (NDM)		
20NG	**100**	0.601	0.601	0.503	**0.617**	0.485	0.544	0.597	0.598	0.521	**0.610**	0.504	0.551		
	400	0.708	0.711	0.665	**0.738**	0.658	0.728	0.702	0.707	0.663	**0.733**	0.657	0.722		
	700	0.739	0.744	0.705	**0.775**	0.707	0.748	0.735	0.738	0.702	**0.769**	0.704	0.742		
	1500	0.769	0.773	0.749	**0.798**	0.764	0.776	0.765	0.767	0.745	**0.792**	0.760	0.771		
Webkb	**100**	0.876	0.875	0.832	**0.912**	0.683	0.860	0.862	0.862	0.809	**0.898**	0.664	0.843		
	400	0.871	0.875	0.870	**0.905**	0.757	0.864	0.855	0.862	0.853	**0.888**	0.733	0.850		
	700	0.863	0.869	0.871	**0.893**	0.788	0.874	0.845	0.856	0.859	**0.877**	0.765	0.865		
	1500	0.860	0.867	0.868	**0.886**	0.836	0.848	0.849	0.852	0.852	**0.867**	0.820	0.836		
Reuters8	**100**	0.944	0.923	0.920	**0.973**	0.890	0.937	0.894	0.877	0.844	**0.909**	0.746	0.885		
	400	0.953	0.931	0.954	**0.974**	0.941	0.948	0.888	0.883	0.900	**0.927**	0.873	0.875		
	700	0.945	0.937	0.956	**0.969**	0.939	0.947	0.877	0.888	0.886	**0.893**	0.849	0.880		
	1500	0.936	0.942	0.942	**0.961**	0.945	0.932	0.815	**0.891**	0.839	0.868	0.855	0.812		
Reuters52	**100**	**0.835**	0.834	0.811	0.826	0.817	0.824	0.341	0.445	0.306	**0.462**	0.305	0.343		
	400	0.864	0.875	0.864	**0.892**	0.862	0.861	0.450	0.475	0.443	**0.489**	0.417	0.431		
	700	0.861	0.879	0.870	**0.893**	0.857	0.860	0.429	0.475	0.445	**0.492**	0.403	0.416		
	1500	0.850	0.857	0.850	**0.877**	0.847	0.847	0.413	0.418	0.406	**0.451**	0.423	0.406		
Classic4	**100**	0.915	0.918	0.885	**0.941**	0.852	0.910	0.912	0.913	0.880	**0.929**	0.848	0.906		
	400	0.942	0.945	0.944	**0.972**	0.931	0.945	0.941	0.945	0.944	**0.967**	0.929	0.944		
	700	0.944	0.944	0.946	**0.974**	0.942	0.949	0.944	0.943	0.945	**0.969**	0.942	0.949		
	1500	0.935	0.945	0.946	**0.975**	0.951	0.938	0.935	0.945	0.946	**0.970**	0.952	0.937		

Table 9 The micro and macro F_1 values of six FS metrics as part of the VGFSS framework on five data sets

Dataset	Subset	SVM micro F_1						SVM macro F_1					
		VGFSS (IG)	VGFSS (DFS)	VGFSS (BNS)	VGFSS (IDFS)	VGFSS (NDM)	VGFSS (OR)	VGFSS (IG)	VGFSS (DFS)	VGFSS (BNS)	VGFSS (IDFS)	VGFSS (NDM)	VGFSS (OR)
20NG	**100**	0.576	**0.597**	0.501	0.577	0.504	0.549	0.599	**0.620**	0.551	0.619	0.547	0.573
	400	0.660	0.688	0.633	**0.700**	0.619	0.674	0.688	0.713	0.670	**0.721**	0.664	0.699
	700	0.706	0.717	0.675	**0.731**	0.657	0.718	0.733	0.741	0.710	**0.761**	0.694	0.740
	1500	0.755	0.761	0.720	**0.769**	0.720	0.766	0.776	0.783	0.748	**0.806**	0.747	0.784
Webkb	**100**	0.846	0.860	0.782	**0.883**	0.659	0.841	0.819	0.853	0.745	**0.857**	0.647	0.827
	400	0.855	0.870	0.857	**0.897**	0.768	0.865	0.837	0.859	0.838	**0.868**	0.744	0.851
	700	0.865	0.880	0.869	**0.910**	0.777	0.876	0.852	0.869	0.856	**0.877**	0.753	0.862
	1500	0.878	0.876	0.872	**0.899**	0.804	0.872	0.864	0.863	0.858	**0.865**	0.776	0.861
Reuters8	**100**	0.921	0.949	0.900	**0.975**	0.894	0.937	0.805	0.913	0.773	**0.921**	0.764	0.860
	400	0.945	0.963	0.943	**0.978**	0.915	0.958	0.853	0.903	0.876	**0.933**	0.797	0.899
	700	0.947	0.968	0.950	**0.988**	0.935	0.966	0.866	0.916	0.886	**0.937**	0.830	0.916
	1500	0.954	0.964	0.959	**0.987**	0.937	0.967	0.897	0.910	0.903	**0.933**	0.843	0.923
Reuters52	**100**	0.822	0.866	0.808	**0.907**	0.798	0.810	0.294	0.443	0.309	**0.535**	0.280	0.255
	400	0.895	0.910	0.874	**0.940**	0.894	0.886	0.503	0.557	0.405	**0.607**	0.554	0.456
	700	0.904	0.918	0.901	**0.943**	0.903	0.907	0.507	0.576	0.509	**0.591**	0.568	0.497
	1500	0.910	0.920	0.910	**0.942**	0.912	0.914	0.529	0.572	0.513	**0.600**	0.559	0.527
Classic4	**100**	0.906	0.911	0.887	**0.913**	0.883	0.894	0.901	0.910	0.888	**0.914**	0.882	0.893
	400	0.951	0.956	0.945	**0.966**	0.942	0.953	0.951	0.956	0.947	**0.971**	0.944	0.953
	700	0.957	0.962	0.962	**0.975**	0.944	0.964	0.956	0.962	0.963	**0.980**	0.946	0.963
	1500	0.959	0.963	0.963	**0.973**	0.956	0.967	0.960	0.963	0.964	**0.977**	0.958	0.967

Table 9 continued

Dataset	Subset	VGFSS (IG)	SVM micro F_1			VGFSS (NDM)	VGFSS (OR)	VGFSS (IG)	SVM macro F_1			VGFSS (NDM)	VGFSS (OR)
			VGFSS (DFS)	VGFSS (BNS)	VGFSS (IDFS)				VGFSS (DFS)	VGFSS (BNS)	VGFSS (IDFS)		
			Random Forest micro F_1						Random Forest macro F_1				
20NG	**100**	0.629	0.637	0.548	**0.660**	0.551	0.607	0.625	**0.653**	0.590	0.652	0.583	0.614
	400	0.721	0.733	0.702	**0.772**	0.690	0.729	0.715	0.727	0.709	**0.756**	0.698	0.726
	700	0.748	0.769	0.746	**0.785**	0.731	0.763	0.742	0.764	0.745	**0.769**	0.734	0.758
	1500	0.775	0.787	0.776	**0.806**	0.789	0.785	0.770	0.782	0.770	**0.790**	0.786	0.781
Webkb	**100**	0.839	0.875	0.810	**0.897**	0.652	0.854	0.818	0.862	0.770	**0.892**	0.620	0.835
	400	0.864	0.883	0.862	**0.888**	0.762	0.883	0.851	0.871	0.848	**0.879**	0.737	0.870
	700	0.851	0.863	0.875	**0.898**	0.796	0.855	0.833	0.845	0.865	**0.894**	0.772	0.840
	1500	0.844	0.845	**0.870**	0.867	0.829	0.855	0.825	0.824	0.855	**0.858**	0.810	0.841
Reuters8	**100**	0.921	0.928	0.910	**0.972**	0.894	0.917	0.813	0.880	0.794	**0.924**	0.727	0.800
	400	0.933	0.925	0.953	**0.974**	0.936	0.954	0.805	0.877	0.895	**0.911**	0.854	0.881
	700	0.931	0.936	0.954	**0.967**	0.939	0.948	0.798	0.886	**0.890**	0.885	0.858	0.867
	1500	0.931	0.943	0.947	**0.957**	0.925	0.938	0.798	**0.894**	0.859	0.845	0.795	0.829
Reuters52	**100**	0.824	0.864	0.816	**0.886**	0.810	0.826	0.360	0.460	0.355	**0.477**	0.282	0.342
	400	0.879	0.861	0.866	**0.884**	0.862	0.870	0.455	0.436	0.439	**0.471**	0.439	0.451
	700	0.871	0.854	0.873	**0.880**	0.851	0.863	0.437	0.406	**0.471**	0.459	0.412	0.438
	1500	0.852	0.835	0.851	**0.856**	0.842	0.838	0.417	0.388	0.425	**0.436**	0.394	0.402
Classic4	**100**	0.906	0.920	0.883	**0.958**	0.888	0.908	0.903	0.916	0.883	**0.929**	0.885	0.907
	400	0.929	0.942	0.934	**0.962**	0.934	0.947	0.928	0.941	0.934	**0.947**	0.936	0.947
	700	0.940	0.939	0.942	**0.970**	0.938	0.938	0.940	0.939	0.943	**0.956**	0.940	0.936
	1500	0.935	0.937	0.943	**0.967**	0.958	0.939	0.936	0.937	0.943	0.952	**0.959**	0.939

anced data sets such as the Classic4 and 20NG data sets, but for unbalanced data sets, its performance is poor. BNS is found to be more compatible with the RF classifier.

We find that the **IG** metric assigns wrong labels when used with VGFSS algorithm. It performs assignment of labels based on negative terms. IG assigns higher score to the strong negative and strong common terms as compared to the rare terms. Also, IG assigns labels based on size of classes and also ranks the terms in this perspective, i.e., negative terms of major classes are ranked higher. So, a negative term from the major class can be ranked higher than a negative term from a small class. Due to selection of features based on negative term criterion, IG's performance degrades in R52 data set and worsens for the VGFSS and IGFSS frameworks. For all other data sets, its performance is comparable with DFS metric.

The **NDM** metric was observed to assign correct labels to negative, rare and sparse terms, but for common terms, it is biased toward larger classes. The main weakness of NDM is that it can assign higher ranks to highly irrelevant sparse terms in large and highly skewed data. Even some strong features having strong recall value are pushed down in the ranked list and the performance degrades. Also, it is not suitable with the VGFSS ensemble approach.

The **OR** metric prioritizes terms belonging to the larger class during VGFSS process. It gives more importance to negative and rarer terms than common features. The IGFSS algorithm does not do well with the OR metric as the former needs negative features to perform well, but few negative features are provided by OR metric.

The **DFS** metric neglects most of the negative features. The VGFSS + DFS and IGFSS + DFS shows comparable performance for the SVM classifier. Although features selected by DFS metric are good in order, but too many strong features assigned to different classes are pushed down in the ranking during the selection process because of which these features are refrained from getting selected in the final subset and thus DFS's performance decreases. But as features from each class are selected by the proportion of class sizes during VGFSS, thus making VGFSS + DFS better.

The newly proposed **IDFS** metric is designed so that it uplifts the score of only those features, which are rare and having class distribution in fewer classes while absent in other classes as in the case of negative features. From our results, we saw that IDFS and DFS are more suitable with VGFSS and IGFSS ensemble methods as compared to the other metrics. But, both SVM and RF classifiers attain the highest accuracy values when the IDFS metric was used with VGFSS technique on the five data sets. Also, IDFS shows the best results for the highly skewed data sets such as Reuters52. The performance of IDFS was found to exceed that of the other FS metrics in skewed data sets with great margins as it does not ignore rare features of smaller classes. For the balanced data sets with high sparseness such as the 20NG data set, the performance of IDFS is comparable to that of the DFS, IG and OR metrics and in few cases less than them.

6 Conclusions and Future Work

To address the class imbalance problem of text classification, more recently ensemble approaches (e.g., improved global feature selection scheme (IGFSS) and variable global feature selection scheme (VGFSS)) were proposed, but their performance is heavily dependent on the underlying FS metrics. In this paper, we found that existing well-known metrics run into problems while selecting useful terms discriminating the skewed classes, and thus may not take full advantage of these ensemble frameworks. For example, we observed that VGFSS with existing FS metrics leaves out many relevant features from the small classes for highly skewed data sets such as Reuters52. To solve such problems, we proposed a new FS metric named novel inherent distinguished feature selector (IDFS), which is designed to select terms highly useful in discriminating the skewed classes especially from smaller classes. We carry out experiments on five data sets with two classifiers and investigate the effectiveness of IDFS against five well-known FS metrics as a stand-alone FS algorithm and as a part of IDFSS and VGFSS frameworks. The higher micro and macro F_1 values of subsets of top ranked terms of IDFS as a stand-alone FS algorithm and with the ensemble approaches show its superiority over existing FS metrics.

As a part of our future work, we are interested in solving another problem faced by the VGFSS scheme. As VGFSS selects a larger number of features from the larger class, a lot of sparse terms from the larger class can become the part of the final subset, thus degrading the performance of a text classifier. Another interesting area that we intend to explore is how to design effective feature selection algorithms for hierarchical learning tasks [56], where instances are classified at various levels of granularity.

References

1. Uysal, A.K.; Gunal, S.: A novel probabilistic feature selection method for text classification. Knowl. Based Syst. **36**, 226–235 (2012)
2. Grimes, S.: Unstructured data and the 80 percent rule. http://breakthroughanalysis.com/2008/08/01/unstructured-data-and-the-80-percent-rule/. Accessed 13 Oct 2019 (2019)
3. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. (CSUR) **34**(1), 1–47 (2002)
4. Marin, A.; Holenstein, R.; Sarikaya, R.; Ostendorf, M.: Learning phrase patterns for text classification using a knowledge graph and unlabeled data. In: 15th Annual Conference of the International Speech Communication Association (2014)

5. Li, X.; Xie, H.; Chen, L.; Wang, J.; Deng, X.: News impact on stock price return via sentiment analysis. Knowl. Based Syst. **69**(1), 14–23 (2014)
6. Rao, Y.; Xie, H.; Li, J.; Jin, F.; Wang, F.L.; Li, Q.: Social emotion classification of short text via topic-level maximum entropy model. Inf. Manag. **53**(8), 978–986 (2016)
7. Uysal, A.K.: An improved global feature selection scheme for text classification. Expert Syst. Appl. **43**, 82–92 (2016)
8. Mironczuk, M.; Protasiewicz, J.: A recent overview of the state-of-the-art elements of text classification. Expert Syst. Appl. **106**, 36–54 (2018)
9. Joachims, T.: Learning To Classify Text using Support Vector Machines. Kluwer Academic Publishers, Berlin (2002)
10. Aggarwal, C.C.; Zhai, C.: A survey of text classification algorithms. In: Mining Text Data, pp. 163–222. Springer (2012)
11. Grimmer, J.; Stewart, B.M.: Text as data: the promise and pitfalls of automatic content analysis methods for political texts. Pol. Anal. **21**, 267–297 (2013)
12. Ko, Y.: A study of term weighting schemes using class information for text classification. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1029–1030. Citeseer (2012)
13. Forman, G.: An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res. **3**(Mar), 1289–1305 (2003)
14. Lan, M.; Tan, C.L.; Low, H.B.; Sung, S.Y.: A comprehensive comparative study on term weighting schemes for text categorization with support vector machines. In: Special Interest Tracks and Posters of the 14th International conference on World Wide Web, pp. 1032–1033 (2005)
15. Zhang, W.; Yoshida, T.; Tang, X.: A comparative study of TF*IDF, LSI and multi-words for text classification. Expert Syst. Appl. **38**, 2758–2765 (2011)
16. Manning, C.D.; Raghavan, P.; Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
17. Lan, M.; Tan, C.L.; Su, J.; Lu, Y.: Supervised and traditional term weighting methods for automatic text categorization. IEEE Trans. Pattern Anal. Mach. Intell. **31**(4), 721–735 (2008)
18. Chen, K.; Zhang, Z.; Long, J.; Zhang, H.: Turning from tf-idf to tf-igm for term weighting in text classification. Expert Syst. Appl. **66**, 245–260 (2016)
19. Sabbaha, T.; Selamat, A.; Selamat, M.H.; Al-Anzi, F.S.; Viedmae, E.H.; Krejcar, O.; Fujita, H.: Modified frequency-based term weighting schemes for text classification. Appl. Soft Comput. **58**, 193–206 (2017)
20. Mengle, S.S.; Goharian, N.: Ambiguity measure feature-selection algorithm. J. Am. Soc. Inf. Sci. Technol. **60**(5), 1037–1050 (2009)
21. Maruf, S.; Javed, K.; Babri, H.A.: Improving text classification performance with random forests-based feature selection. Arab. J. Sci. Eng. **41**, 951–964 (2016)
22. Saeed, M.; Javed, K.; Babri, H.A.: Machine learning using bernoulli mixture models: clustering, rule extraction and dimensionality reduction. Neurocomputing **119**(7), 366–374 (2013)
23. Aceto, G.; Ciuonzo, D.; Montieri, A.; Pescapé, A.: Multi-classification approaches for classifying mobile app traffic. J. Netw. Comput. Appl. **103**, 131–145 (2018)
24. Harish, B.; Revanasiddappa, M.: A comprehensive survey on various feature selection methods to categorize text documents. Int. J. Comput. Appl. **164**(8), 1–7 (2017)
25. Javed, K.; Maruf, S.; Babri, H.A.: A two-stage markov blanket based feature selection algorithm for text classification. Neurocomputing **157**, 91–104 (2015)
26. Javed, K.; Babri, H.; Saeed, M.: Feature selection based on class-dependent densities for high-dimensional binary data. IEEE Trans. Knowl. Data Eng. **24**(3), 465–477 (2012)
27. Yang, Y.; Pedersen, J.O.: A comparative study on feature selection in text categorization. ICML **97**, 412–420 (1997)
28. Shang, W.; Huang, H.; Zhu, H.; Lin, Y.: A novel feature selection algorithm for text categorization. Expert Syst. Appl. **33**, 1–5 (2007)
29. Ogura, H.; Amano, H.; Kondo, M.: Comparison of metrics for feature selection in imbalanced text classification. Expert Syst. Appl. **38**(5), 4978–4989 (2011)
30. TaşCı, Ş.; Güngör, T.: Comparison of text feature selection policies and using an adaptive framework. Expert Syst. Appl. **40**(12), 4871–4886 (2013)
31. Agnihotri, D.; Verma, K.; Tripathi, P.: Variable global feature selection scheme for automatic classification of text documents. Expert Syst. Appl. **81**, 268–281 (2017)
32. Cortes, C.; Vapnik, V.: Support vector networks. Mach. Learn. **20**(3), 273–297 (1995)
33. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
34. Montieri, A.; Ciuonzo, D.; Aceto, G.; Pescapé, A.: Anonymity services tor, i2p, jondonym: classifying in the dark (web). IEEE Trans. Depend. Sec. Comput. **17**(3), 662–675 (2020)
35. Javed, K.; Babri, H.A.; Saeed, M.: Impact of a metric of association between two variables on performance of filters for binary data. Neurocomputing **143**, 248–260 (2014a)
36. Bolon-Canedo, V.; Sanchez-Marono, N.; Alonso-Betanzos, A.: Feature Selection for High-Dimensional Data. Springer, Basel (2015)
37. Labani, M.; Moradi, P.; Ahmadizar, F.; Jalili, M.: A novel multivariate filter method for feature selection in text classification problems. Eng. Appl. Artif. Intell. **70**, 25–37 (2018)
38. Guyon, I.; Gunn, S.; Nikravesh, M.; Zadeh, L.: Feature Extraction: Foundations and Applications. Springer, Berlin (2006)
39. Guyon, I.; Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. **3**(Mar), 1157–1182 (2003)
40. Rehman, A.; Javed, K.; Babri, H.A.; Asim, N.: Selection of the most relevant terms based on a max-min ratio metric for text classification. Expert Syst. Appl. **114**, 78–96 (2018)
41. Javed, K.; Saeed, M.; Babri, H.A.: The correctness problem: evaluating the ordering of binary features in rankings. Knowl. Inf. Syst. **39**(3), 543–563 (2014b)
42. Uğuz, H.: A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm. Knowl. Based Syst. **24**(7), 1024–1032 (2011)
43. Srividhya, V.; Anitha, R.: Evaluating preprocessing techniques in text categorization. Int. J. Comput. Sci. Appl. **47**(11), 49–51 (2010)
44. Flach, P.: Machine Learning: The Art and Science of Algorithms that Make Sense of Data. Cambridge University Press, Cambridge (2012)
45. Forman, G.: Bns feature scaling: an improved representation over TF-IDF for SVM text classification. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, pp. 263–270. ACM (2008)
46. Liu, H.; Sun, J.; Liu, L.; Zhang, H.: Feature selection with dynamic mutual information. Pattern Recognit. **42**(7), 1330–1339 (2009)
47. Wang, D.; Zhang, H.; Liu, R.; Lv, W.; Wang, D.: t-test feature selection approach based on term frequency for text categorization. Pattern Recognit. Lett. **45**, 1–10 (2014)
48. Lee, C.; Lee, G.G.: Information gain and divergence-based feature selection for machine learning-based text categorization. Inf. Process. Manag. **42**(1), 155–165 (2006)
49. Pinheiro, R.H.; Cavalcanti, G.D.; Correa, R.F.; Ren, T.I.: A global-ranking local feature selection method for text categorization. Expert Syst. Appl. **39**(17), 12851–12857 (2012)
50. Rehman, A.; Javed, K.; Babri, H.A.: Feature selection based on a normalized difference measure for text classification. Inf. Process. Manag. **53**(2), 473–489 (2017)

51. Chen, J.; Huang, H.; Tian, S.; Qu, Y.: Feature selection for text classification with naïve bayes. Expert Syst. Appl. **36**(3), 5432–5435 (2009)
52. Wang, F.; Li, Ch; Wang Js, XuJ; Li, L.: A two-stage feature selection method for text categorization by using category correlation degree and latent semantic indexing. J. Shanghai Jiaotong Univ. (Sci.) **20**(1), 44–50 (2015)
53. Mladenic, D.; Grobelnik, M.: Feature selection for unbalanced class distribution and naive bayes. In: Proceedings of the 16nth International Conference on Machine Learning, pp. 258–267 (1999)
54. Cachopo, AMdJC; et al.: Improving Methods for Single-Label Text Categorization. Instituto Superior Técnico, Portugal (2007)
55. Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; Witten, I.H.: The WEKA data mining software: an update. ACM SIGKDD Explor. Newsl. **11**(1), 10–18 (2009)
56. Montieri, A.; Ciuonzo, D.; Bovenzi, G.; Persico, V.; Pescapé, A.: A dive into the dark web: Hierarchical traffic classification of anonymity tools. In: IEEE Transactions on Network Science and Engineering, pp. 1–1 (2019)