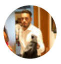
A Comparison of Term Weighting Schemes for Text Classification and Sentiment Analysis with a Supervised Variant of tf.idf

Conference Paper *in* Communications in Computer and Information Science · February 2016

DOI: 10.1007/978-3-319-30162-4_4


Giacomo Domeniconi
University of Bologna
36 PUBLICATIONS **1,293** CITATIONS
SEE PROFILE

Gianluca Moro
University of Bologna
121 PUBLICATIONS **1,562** CITATIONS
SEE PROFILE

Roberto Pasolini
University of Bologna
16 PUBLICATIONS **307** CITATIONS
SEE PROFILE

Claudio Sartori
University of Bologna
131 PUBLICATIONS **1,253** CITATIONS
SEE PROFILE

A Comparison of Term Weighting Schemes for Text Classification and Sentiment Analysis with a Supervised Variant of tf.idf

Giacomo Domeniconi, Gianluca Moro, Roberto Pasolini, and Claudio Sartori

DISI, Università degli Studi di Bologna, Italy
{giacomo.domeniconi, gianluca.moro, roberto.pasolini, claudio.sartori}@unibo.it

Abstract. In text analysis tasks like text classification and sentiment analysis, the careful choice of term weighting schemes can have an important impact on the effectiveness. Classic unsupervised schemes are based solely on the distribution of terms across documents, while newer supervised ones leverage the knowledge of membership of training documents to categories; these latter ones are often specifically tailored for either topic or sentiment classification. We propose here a supervised variant of the well-known tf.idf scheme, where the idf factor is computed without considering documents within the category under analysis, so that terms frequently appearing only within it are not penalized. The importance of these terms is further boosted in a second variant inspired by relevance frequency. We performed extensive experiments to compare these novel schemes to known ones, observing top performances in text categorization by topic and satisfactory results in sentiment classification.

Keywords: Term Weighting, Supervised Term Weighting, Text Classification, Sentiment Analysis

1 Introduction

The classification of text documents written in natural language in a number of predefined *categories* (or *classes*) is a task performed in many practical applications. In the canonical *text classification* or *categorization* task, for which many tailored solutions exist, the categories correspond to discussion topics (such as *sports*, *movies*, *travels* etc.) and the goal is to label each document with the topic it deals with, possibly more than one. In the typical machine learning-based approach, one or more classifiers are trained from pre-labeled documents and used to automatically label subsequent documents [24]. Other tasks exist with a different premise and using different techniques with respect to understanding topics discussed in text, but are structurally ascribable as text classification tasks. An example is the identification of languages used in documents, where categories represent languages rather than topics.

In text analysis, a research branch currently of high interest is *sentiment analysis*, where the opinion of people must be understood from what they write.

We specifically consider the *sentiment classification* task, where documents must be classified according to the expressed attitude towards a specific subject or object, which may be a politician, a restaurant, a book etc. In a very common case documents are reviews, such as those written by users of e-commerce sites, which must be classified as either *positive* or *negative*, or even *neutral* if such case is considered. This can then be seen as another example of text classification task, where categories represent opposite polarities of judgement.

In any text analysis task, the first hurdle is the unstructured nature of text data, which requires to employ a more suitable representation of documents. The most widely used approach for this is the Vector Space Model (VSM), where each document is represented as a vector in a multi-dimensional space. Features of these vectors usually correspond to specific words or *terms*, indicating their presence within each document without considering their position or their function: such vectors are also known as *bags of words*. After extracting single terms from documents and selecting a suitable set of them to be used as features, a vector for each document must be extracted with suitable values or *weights* indicating the importance of each term in the document.

These weights are consistently assigned according to a chosen *term weighting scheme*. Although few of them are well-known and largely employed, many different schemes and variants thereof have been proposed throughout the literature, each generally carrying some kind of improvement over previous ones. Given a text analysis task where VSM is employed, the choice of the term weighting scheme can have a significant impact on its outcomes.

For example, in [18] different weighting schemes are thoroughly tested on text classification by topic with SVM classifiers and substantial gaps can be observed between accuracy levels obtained with different schemes. Similarly, more recent studies [6, 22] analyzed the impact of term weighting on sentiment classification, for which some schemes have been specifically devised. Other tasks where term weighting has proven to have some influence over final results are cross-domain classification [11, 12], novelty mining [27] and discovery of gene-function associations [9]. From scientific research, the use of term weighting schemes has moved to practical applications, with its employment in projects of major IT enterprises such as Yahoo! [3] and IBM [23].

A weighting scheme is often composed by a *local* and a *global* factor: while the former estimates the relevance of each term within each single document regardless of other ones, the latter indicates the importance of each term across the whole collection of documents rather than in a specific one. Once weights are computed, resulting vectors are often normalized in order to avoid giving more bias to longer documents.

Many studies on term weighting focus on the global factor, for which two major approaches can be distinguished. *Unsupervised* weighting schemes are only based on the distribution of terms across documents and can be used in virtually any text analysis task. *Supervised* schemes are instead devised for classification tasks, as also leverage the information about membership of documents to categories. Older research investigated unsupervised schemes, including the well-known and widely used *tf.idf* scheme having its origins in information retrieval. More recent research is instead mostly focused on supervised schemes, especially used in text categorization and sentiment analysis.

Following a study of existing weighting schemes, both supervised and not, we propose here a variant of the classic *tf.idf* scheme, specifically by providing a supervised version of the global *idf* factor, normally based on the assumption that terms more frequent throughout a collection have minor discriminating power and are thus less important. In a text classification setting this does not necessary hold: a term appearing frequently throughout a specific category and rarely in other ones is strongly discriminative for that category, but its standard *idf* is low compared to less frequent terms. Following this intuition, the *idf* factor in our variant is computed for each category ignoring documents labeled with it, so that the importance of a term is not penalized by appearances inside the category itself. Of this scheme we also propose a further variant where it is combined with *relevance frequency*, another supervised global weighting factor.

This paper reprises work from [13], reporting the comparison between our proposed schemes and other ones in text classification by topic and also adding an overview and an evaluation of schemes aimed to sentiment analysis.

The paper is organized as follows. Section 2 gives an overview of existing term weighting methods, with focus on global factors. In Section 3 we introduce and motivate our two variants of *idf*. Section 4 presents the general setup of our experimental evaluation of different term weighting schemes, whose results are reported and discussed in Section 5. Finally, Section 6 sums up the work with conclusive remarks.

2 Term Weighting in Text Classification

The problem of text categorization has been extensively investigated in the past years, considering the ever-increasing applications of this discipline, such as news or e-mail filtering and organization, indexing and semantic search of documents, sentiment analysis and opinion mining, prediction of genetic diseases etc. [24]

In the machine learning approach, a knowledge model to classify documents within a set $\mathcal{C} = \{c_1, c_2, \ldots, c_{|\mathcal{C}|}\}$ of categories is built upon a training set $\mathcal{D}_T = \{d_1, d_2, \ldots, d_{|\mathcal{D}_T|}\}$ of documents with a known labeling $L : \mathcal{D}_T \times \mathcal{C} \to \{0, 1\}$ ($L(d, c) = 1$ if and only if document d is labeled with c). In order to leverage standard machine learning algorithms, documents are generally pre-processed to be represented in a Vector Space Model (VSM).

In the VSM, the content of a document d_j is represented as a vector $\mathbf{w}^j = \{w_1^j, w_2^j, \ldots, w_n^j\}$ in a n-dimensional vector space $\mathbb{R}^n$, where w_i^j is a weight that indicates the importance of a term t_i in d_j. Terms $t_1, t_2, \ldots, t_n$ constitute a set of features, shared across all documents. In other words, each weight w_i^j indicates how much the term t_i contributes to the semantic content of d_j.

Weights for each term-document couple are assigned according to a predefined term weighting scheme, which must meaningfully estimate the importance of each term within each document.

Three are the considerations discussed in the years regarding the correct assignment of weights in text categorization [4]:

1. the multiple occurrence of a term in a document appears to be related to the content of the document itself (*term frequency* factor);
2. terms uncommon throughout a collection better discriminate the content of the documents (*collection frequency* factor);
3. long documents are not more important than the short ones, normalization is used to equalize the length of documents.

Following these points, most weighting schemes are the product of a *local* (*term frequency*) factor L computed for each term-document couple and a *global* (*collection frequency*) factor G computed for each term on the whole collection. *Cosine normalization* is then usually applied to each document vector.

$$\mathbf{w}^j_{\text{normalized}} = \frac{1}{\sqrt{\sum_{i=1}^{n}(w_i^j)^2}} \cdot \mathbf{w}^j \tag{1}$$

There are a number of ways to calculate the local term frequency factor. The simplest one is binary weighting, which only considers the presence (1) or absence (0) of a term in a document, ignoring its frequency. Another obvious possibility is to consider the number of occurrences of the term in the document, which is often the intended meaning of "term frequency" (*tf*). Among other variants, the *logarithmic tf*, computed as $\log(1+tf)$, is now practically the standard local factor used in literature [4]. In this work, we have chosen logarithmic term frequency ($\log(1 + tf)$) as the local factor for all experiments.

As mentioned earlier, the global collection frequency factor can be *supervised* or *unsupervised*, depending whether it leverages or not the knowledge of membership of documents to categories. In the following, we summarize some of the most used and recent methods proposed in the literature of both types.

2.1 Unsupervised Term Weighting Methods

Generally, unsupervised term weighting schemes, not considering category labels of documents, derive from IR research. The most widely used unsupervised method is *tf.idf*, which perfectly embodies the three assumptions previously seen. The basic idea is that terms appearing in many documents are not good for discrimination, and therefore they will weight less than terms occurring in few documents. Over the years, researchers have proposed several variations in the way they calculate and combine the three components: *tf*, *idf* and normalization.

Indicating with $N = |\mathcal{D}_T|$ the total number of training documents and with $df(t_i)$ the count of training documents where term t_i appears at least once, the standard *tf.idf* formulation is

$$tf.idf(t_i, d_j) = tf(t_i, d_j) \cdot idf(t_i) = tf(t_i, d_j) \cdot \log\left(\frac{N}{df(t_i)}\right) , \tag{2}$$

where any one of the aforementioned local weighting factors can be used in place of *tf*. The *idf* factor multiplies the *tf* by a value that is greater when the term is rare in the collection $\mathcal{D}_T$ of training documents. The weights obtained by the formula above are then normalized according to the third assumption by means of cosine normalization (Eq. 1).

The standard idf factor given above can be replaced with other ones: a classic alternative is the *probabilistic idf*, defined as

$$pidf(t_i) = \log\left(\frac{N - df(t_i)}{df(t_i)}\right) . \tag{3}$$

In [26] is proposed a variant of the *idf* called *Weighted Inverse Document Frequency* (*widf*), given by dividing the $tf(t_i, d_j)$ by the sum of all the frequencies of t_i in all the documents of the collection:

$$widf(t_i) = \frac{1}{\sum_{d_x \in \mathcal{D}_T} tf(t_i, d_x)} \tag{4}$$

[5] propose a combination of *idf* and *widf*, called *Modified Inverse Document Frequency* (*midf*) that is defined as follows:

$$midf(t_i) = \frac{df(t_i)}{\sum_{d_x \in \mathcal{D}_T} tf(t_i, d_x)} \tag{5}$$

Of course the simplest choice, sometimes used, is to not use a global factor at all, setting it to 1 for all terms and only considering term frequency.

2.2 Supervised Term Weighting Methods

Since text classification is a supervised learning task, where the knowledge of category labels of training documents is necessary, many term weighting methods use this information to supervise the assignment of weights to each term.

A basic example of supervised global factor is *inverse category frequency*:

$$icf(t_i) = \log\left(\frac{|\mathcal{C}|}{cf(t_i)}\right) \tag{6}$$

where $df(t_i)$ denotes the count of categories where t_i appears in at least one relevant document. The idea of the *icf* factor is similar to that of *idf*, but using the categories instead of the documents: the fewer are the categories in which a term occurs, the greater is the discriminating power of the term.

Within text categorization, especially in the multi-label case where each document can be labeled with an arbitrary number of categories, it is common to train one binary classifier for each one of the possible categories. For each category c_k, the corresponding model must separate its *positive examples*, i.e. documents actually labeled with c_k, from all other documents, the *negative examples*. In this case, it is allowed to compute for each term t_i a distinct collection frequency factor for each category c_k, used to represent documents in the VSM only for verifying relevance to that specific category.

To summarize the various methods of supervised term weighting, we show in Table 1 the fundamental elements mostly used in the following formulas to compute the global importance of a term t_i for a category c_k.

- A denotes the number of documents belonging to category c_k where the term t_i occurs at least once;
- B denotes the number of documents belonging to c_k where t_i does not occur;
- dually C denotes the number of documents not belonging to category c_k where the term t_i occurs at least once;
- finally D denotes the number of documents not belonging to c_k where t_i does not occur.

These four elements sum to N, the total number of training documents.

Table 1. Fundamental elements of supervised term weighting.

	c_k	$\overline{c_k}$
t_i	A	C
$\overline{t_i}$	B	D

The standard *idf* factor can be expressed in this notation as

$$idf = \log\left(\frac{N}{A+C}\right) . \tag{7}$$

As suggested in [4] and [7], an intuitive approach to supervised term weighting is to employ common techniques for feature selection, such as χ^2, *information gain*, *odds ratio* and so on. In [7] the χ^2 factor is used to weigh terms, replacing the *idf* factor, and the results show that the *tf.*χ^2 scheme is more effective than *tf.idf* using a SVM classifier. Similarly [4] apply feature selection schemes multiplied by the *tf* factor, by calling them "supervised term weighting". In this work they use the same scheme for feature selection and term weighting, in contrast to [7] where different measures are used. The results of the two however are in contradiction: [4] shows that the *tf.idf* always outperforms χ^2, and in general the supervised methods not give substantial improvements compared to unsupervised *tf.idf.* The widely-used collection frequency factors χ^2, information gain (*ig*), odds ratio (*or*) and mutual information (*mi*) are described as follows:

$$\chi^2 = N \cdot \frac{(A \cdot D - B \cdot C)^2}{(A+C)\cdot(B+D)\cdot(A+B)\cdot(C+D)} , \tag{8}$$

$$ig = -\frac{A+B}{N} \cdot \log\frac{A+B}{N} + \frac{A}{N} \cdot \log\frac{A}{A+C} + \frac{B}{N} \cdot \log\frac{B}{B+D} , \tag{9}$$

$$or = \log\left(\frac{A \cdot D}{B \cdot C}\right) , \tag{10}$$

$$mi = \log\left(\frac{A \cdot N}{(A+B)\cdot(A+C)}\right) . \tag{11}$$

Any supervised feature selection scheme can be used for the term weighting. For example, the *gss* extension of the χ^2 proposed by [15] eliminates N at numerator and the emphasis to rare features and categories at the denominator.

$$gss = \frac{A \cdot D - B \cdot C}{N^2} \tag{12}$$

Relevance frequency [17] considers the terms distribution in the positive and negative examples, stating that, in multi-label text categorization, the higher the concentration of high-frequency terms in the positive examples than in the negative ones, the greater the contribution to categorization.

$$rf = \log\left(2 + \frac{A}{\max(1, C)}\right) \tag{13}$$

The *icf-based* scheme [28] combines this idea with *icf*:

$$\textit{icf-based} = \log\left(2 + \frac{A}{\max(1, C)} \cdot \frac{|\mathcal{C}|}{cf(t_i)}\right) \tag{14}$$

Further term weighting schemes are based on additional information. [25] proposes a scheme that leverages availability of past retrieval results, consisting of queries that contain a particular term, retrieved documents, and their relevance judgments. Another different approach to supervised term weighting [20] does not use the statistical information of terms in documents like methods mentioned above, but exploits instead the semantics of categories and terms.

2.3 Term Weighting Methods for Sentiment Analysis

While the sentiment classification is structurally equivalent to canonical text categorization with review polarities in place of topics, many techniques have been specifically devised for this problem, mainly in order to find and value terms expressing positive or negative sentiment. Some research on this task is focused the term weighting issue, with both studies of existing solutions and proposals of new tailored schemes, mostly supervised.

A first example of weighting scheme conceived for sentiment analysis is *delta tf.idf*, which is obtained by computing the standard idf factor separately on positive- and negative-labeled documents and taking the difference between them [21]. While this scheme is supervised as it leverages knowledge of labeling of training documents, it does not assign distinct weights to the two categories. Denoting with N^+ and df^+ total and per-term document counts restricted to positive documents and with N^- and df^- the same for negative ones, delta idf is defined as

$$\Delta idf(t_i) = \log\left(\frac{N^+}{df^+(t_i)}\right) - \log\left(\frac{N^-}{df^-(t_i)}\right) = \log\left(\frac{N^+ \cdot df^-(t_i)}{N^- \cdot df^+(t_i)}\right) . \tag{15}$$

A subsequent study [22] focused on term weighting in sentiment analysis reprised both the delta tf.idf idea and the classic BM25 scheme from information retrieval, proposing some variations of the *delta idf* factor based on probabilistic *idf* and BM25, an example of such variants is *smoothed delta idf*:

$$\mathit{\Delta idf}_{sm}(t_i) = \log\left(\frac{N^+ \cdot \mathit{df}^-(t_i) + 0.5}{N^- \cdot \mathit{df}^+(t_i) + 0.5}\right) . \tag{16}$$

A more recent study [6] takes into consideration other supervised weighting schemes: here the unique global factor of each term, called *importance of a term for expressing sentiment* (ITS), is the maximum between the positive and negative classes of a per-class weighting scheme, such as those proposed in [4, 7]. Among other schemes cited in the study is *Weighted Log Likelihood Ratio*, whose per-class value can be computed as

$$\mathit{wllr}(t_i, c_k) = \frac{A}{A+B} \cdot \log\left(\frac{A(C+D)}{C(A+B)}\right) . \tag{17}$$

Many of the same schemes are reviewed in [14] and tested with multiple learning algorithms. Here are also proposed more trivial schemes which are shown to perform well, such as the maximum *class density*, defined as

$$\mathit{cd}_{MAX}(t_i) = \max_{c_k \in \mathcal{C}} \mathit{cd}(t_i, c_k) \ , \qquad \text{where } \mathit{cd}(t_i, c_k) = \frac{A}{A+C} \ . \tag{18}$$

In [29] is addressed the so-called *over-weighting* problem, where the weighting scheme erroneously attributes excessive importance to rare (*singular*) terms: a regularization method appliable to existing schemes is proposed as a solution.

3 A Supervised Variant of Inverse Document Frequency

Here we introduce a supervised variant of *tf.idf.* The basic idea of our proposal is to avoid decreasing the weight of terms contained in documents belonging to the same category, so that words that appear in several documents of the same category are not disadvantaged, as instead happens in the standard formulation of *idf.* We refer to this variant with the name *idfec* (Inverse Document Frequency Excluding Category). Therefore, the proposed category frequency factor scheme is formulated as

$$\mathit{idfec}\ (t_i, c_k) = \log\left(\frac{|\mathcal{D}_T \setminus c_k| + 1}{\max\left(1, |d \in \mathcal{D}_T \setminus c_k : t_i \in d|\right)}\right) , \tag{19}$$

where "$\mathcal{D}_T \setminus c_k$" denotes training documents not labeled with c_k. Using the previous notation, the formula becomes:

$$\mathit{tf.idfec}\ (t_i, d_j) = \mathit{tf}(t_i, d_j) \cdot \log\left(\frac{C+D}{\max(1, C)}\right) . \tag{20}$$

Note that with this variant of *idf* we can have particular cases. If the i-th word is only contained in j-th document, or only in documents belonging to c_k, the denominator becomes 0. To prevent division by 0, the denominator is replaced by 1 in this particular case.

The *tf.idfec* scheme is expected to improve classification effectiveness over *tf.idf* because it discriminates where each term appears. For any category c_k, the importance of a term appearing in many documents outside of it is penalized as in *tf.idf.* On the other side, the importance is not reduced by appearances in the positive examples of c_k, so that any term appearing mostly within the category itself retains an high global weight.

This scheme is similar to *tf.rf* [17], as both penalize weights of a term t_i according to the number of negative examples where the t_i appears. The difference is in the numerator of the fraction, which values positive examples with the term in *rf* and negative ones without it in *idfec.*

To illustrate these properties we use a numerical example. Considering the notation shown in Table 1, suppose we have a corpus of 100 training documents divided as shown in Table 2, for two terms t_1 and t_2 and a category c_k.

Table 2. Example of document distribution for two terms.

	c_k	$\overline{c_k}$		c_k	$\overline{c_k}$
t_1	27	5	t_2	10	25
$\overline{t_1}$	3	65	$\overline{t_2}$	20	45

We can easily note how the term t_1 is very representative, and then discriminant, for the category c_k since it is very frequent within it $(A/(A+B)) = 27/30)$ and not in the rest of the documents $(C/(C+D) = 5/70)$. Similarly we can see that t_2 does not seem to be a particularly discriminating term for c_k. In the standard formulation, the *idf* is

$$\mathit{idf}(t_1) = \log(100/(27+5)) = \log(3.125)$$

and for our best competitor *rf* is

$$rf(t_1, c_k) = \log(2 + 27/5) = \log(7.4) ,$$

while with the *idfec* we obtain

$$\mathit{idfec}\ (t_1, c_k) = \log((65+5)/5) = \log(14) .$$

For t_2 we have instead:

$$\mathit{idf}(t_2) = \log(2.857) , \quad rf(t_2, c_k) = \log(2.4) , \quad \mathit{idfec}(t_2, c_k) = \log(2.8) .$$

We can see that our supervised version of *idf* can separate the weights of the two terms according to the frequency of terms in documents belonging to

c_k or not. In fact, while with the standard *idf* the weights of t_1 and t_2 are very similar, with *idfec* t_1 has a weight much greater than t_2 since t_1 is more frequent and discriminative for the category c_k. This kind of behavior is also exhibited by *rf*, but our method yields an even higher weight for the relevant term t_1.

In its base version, *tf.idfec* takes into account only the negative examples (C and D in Table 1). Instead it could be helpful, especially for the classification task, also to take into account how many documents belonging to c_k contain the term, i.e. how much the term occurs within the category more than in the rest of the collection. Considering this, in a way similar to [28], we propose to mix our idea with that of the *rf* in a new version of our weighting scheme, called *tf.idfec-based* (*tf.idfec-b.* for short) and expressed by the following formula:

$$\mathit{tf.idfec\text{-}b.}(t_i, d_j) = tf(t_i, d_j) \cdot \log\left(2 + \frac{A + C + D}{\max(1, C)}\right) \tag{21}$$

Using the example in the Table 2, the new term weighting scheme becomes for t_1 and t_2 respectively:

$$\mathit{idfec\text{-}b.}(t_1, c_k) = \log(21.4)\ , \qquad \mathit{idfec\text{-}b.}(t_2, c_k) = \log(5.2)\ .$$

With this term weighting scheme, the difference in weight between a very common term (t_2) and a very discriminative one (t_1) is even more pronounced.

4 Experimental Setup

We performed extensive experimental evaluation to compare the effectiveness of the proposed term weighting approach with other schemes. In the following, we describe in detail the organization of these experiments.

4.1 Benchmark Datasets

We used commonly employed text collections as benchmarks for text categorization by topic and sentiment classification in our experiments.

The **Reuters-21578** corpus[1] consists in 21,578 articles collected from Reuters. According to the ModApté split, 9,100 news stories are used: 6,532 for the training set and 2,568 for the test set. One intrinsic problem of the Reuters corpus is the skewed category distribution. In the top 52 categories, the two most common categories (*earn* and *acq*) contain, respectively, 43% and 25% of the documents in the dataset, while the average document frequency of all categories is less than 2%. In literature, this dataset is used considering a various number of categories: we considered two views of this corpus, *Reuters-10* and *Reuters-52* where only the 10 and the 52 most frequent categories are considered, respectively.

The **20 Newsgroups** corpus[2] is a collection of 18,828 Usenet posts partitioned across 20 discussion groups. Some newsgroups are very closely related

[1] `http://www.daviddlewis.com/resources/testcollections/reuters21578/`.
[2] `http://people.csail.mit.edu/jrennie/20Newsgroups/`.

to each other (e.g. comp.sys.ibm.pc. hardware / comp.sys.mac.hardware), while others are highly unrelated (e.g misc.forsale / soc.religion.christian). Likely to [17] we randomly selected 60% of documents as training instances and the remaining 40% make up the test set. Contrarily to Reuters, documents of 20 Newsgroups are distributed rather uniformly across categories.

Movie Review Data[3] contains 2,000 user reviews about movies evenly distributed between positive and negative. We took the v2.0 distribution, using the first 8 of the 10 provided folds for training and the remaining 2 to test.

The **Multi-Domain Sentiment Dataset**[4] [1] contains user reviews from Amazon.com of products of different categories (*domains*): for this it is mostly used to test transfer learning methods [10]. We considered separately the domains *books*, *dvd*, *electronics* and *kitchen*, using for each the standard 2,000 reviews evenly divided between positive and negative polarity. We used the first 800 reviews for each polarity as training set and the last 200 as test set.

4.2 Documents Processing and Learning Workflow

For each dataset, all documents were pre-processed by removing punctuation, numbers and stopwords from a predefined list, then by applying the common Porter stemmer to remaining words. This does not hold for the MDSD datasets, for which we used an already pre-processed distribution where n-grams are considered as terms in addition to single words.

We performed feature selection to keep only a useful subset of terms. Specifically, we extracted for each category the p terms appearing in most of its documents, where for p the following values were tested: 25, 50, 75, 150, 300, 600, 900, 1200, 1800, 2400, 4800, 9600, 14400, 19200 (the last two only for sentiment classification). This feature selection method may be considered counterproductive since we selected the most common terms, but it is actually correct considering the use of the VSM as the terms result to be as less scattered as possible. The task of term weighting is therefore crucial to increase the categorization effectiveness, giving a weight to each term according to the category to which the documents belong.

Since we tested both supervised and unsupervised term weighting methods, we used two different procedures. For unsupervised methods we processed the training set in order to calculate the collection frequency factor for each term, which was then multiplied by the logarithmic term frequency factor (referred to as *tf* in the following) for each term in training and test set. Finally, cosine normalization (Eq. 1) was applied to normalize the term weights.

For supervised methods we used the multi-label categorization approach, where a binary classifier is created for each category. That is, for each category c_k, training documents labeled with it are tagged as positive examples, while the remaining one constitute negative examples. We computed statistical information related to c_k (as described in Table 1) for each term of training documents. Logarithmic *tf* and cosine normalization are applied as above.

[3] `http://www.cs.cornell.edu/people/pabo/movie-review-data/`
[4] `https://www.cs.jhu.edu/~mdredze/datasets/sentiment/`

To train classifiers, we chose to use support vector machines (SVM), which are usually the best learning approach in text categorization [17, 24]: we tested both the linear kernel and the radial basis function (RBF) kernel. Furthermore, to test the effectiveness of classification by varying the term weighting scheme on another algorithm, we used the *Random Forest* learner [2], chosen for both its effectiveness and its speed.

4.3 Performance Evaluation

When dealing with text classification datasets, we measured the effectiveness in terms of precision (π) and recall (ρ) [19]. As a measure of effectiveness that combines π and ρ we used the well-known F_1 measure, defined as:

$$F_1 = \frac{2 \cdot \pi \cdot \rho}{\pi + \rho} \; . \tag{22}$$

For multi-label problems, the F_1 is estimated by its *micro-average* across categories, extracted from the component-wise sum of their confusion matrices [24].

For two-classes sentiment classification problems, we simply computed accuracy as the ratio of correctly classified test documents with respect to the total, which is equivalent to the micro-averaged F_1 measure on the two classes.

To evaluate the difference of performances between term weighting methods, we employed the McNemar's significance test [8, 16, 17], used to compare the distribution of counts expected under the null hypotesis to the observed counts.

Let's consider two classifiers f_a and f_b trained from the same documents but with two different term weighting methods and evaluated using the same test set: some test instances are correctly classify by both, while other ones are misclassified by one or both of them. We denote with n_{01} the number of test instances misclassified by f_a but not by f_b and vice versa with n_{10} misclassified only by f_b. The null hypothesis for the McNemar's significance test states that on the same test instances, two classifiers f_a and f_b will have the same prediction errors, which means that $n_{01} = n_{10} = 0$. So the χ statistic is defined as:

$$\chi = \frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}} \tag{23}$$

χ is approximately distributed as a χ^2 distribution with 1 degree of freedom, where the significance levels 0.01 and 0.001 correspond respectively to the thresholds $\chi_0 = 6.64$ and $\chi_1 = 10.83$. If the null hypothesis is correct, than the probability that χ is greater than 6.64 is less than 0.01, and similarly 0.001 for a value greater than 10.83. Otherwise we may reject the null hypothesis and assume that f_a and f_b have different performances, so the two weighting schemes have a different impact when used on the particular training set.

5 Experimental Results

We tested the effectiveness of classification varying the term weighting scheme on the datasets cited above: for each one we tested the classification varying the number of features p selected for each category. Due to space constraints, we show a selection of significant numeric results, describing similarities between them and other results not shown in detail.

5.1 Text Classification by Topic

We first performed tests on topic classification datasets, which are the primary target of our weighting schemes.

Table 3 shows the performance of 11 different term weighting methods: *tf.idfec*, *tf.idfec-based*, *tf.rf*, *tf.icf-based*, *tf.idf*, *tf.*χ^2, *tf.gss*, *tf.ig*, *midf*, *tf.oddsR* and *tf*. The best micro-averaged F_1 measure across different values of the p parameter for every dataset and learning algorithm are reported. In general, our second proposed scheme *tf.idfec-based* achieved top results in all datasets and with each classifier.

On *Reuters-52 tf.idfec-based* outperforms every other scheme with all classifiers: using a SVM with linear kernel, compared with the standard *tf.idf* we have a 0.8% improvement; the gap with standard supervised schemes is even higher. *tf.idfec-based* also slightly outperforms *tf.rf* and *tf.icf-based.* On *Reuters-10* the results for our proposed methods are very close to other schemes. Considering only 10 categories, supervised term weighting methods appear less relevant to classification effectiveness: this can be deduced from the difference between standard *tf.idf* and our supervised versions on *Reuters-52*, which contains the same documents of *Reuters-10* but labeled with more categories. However, our schemes outperform standard supervised term weighting by more than 10%. In *20 Newsgroups tf.idfec-based* obtains top performances in parity with other weighting schemes. Using linear SVM, the best result of *tf.idfec-based* is higher by 1.8% compared to that obtained from *tf.idf* and by 23.9% compared with a standard supervised method like *tf.ig.*

Observing all the results, we can see that our first proposed scheme *tf.idfec* obtains results always in line but slightly lower than the *tf.idfec-based* variant. This evidently means that considering only the information about the negative categories of the terms is not enough to achieve an optimal accuracy. Conversely, adding information about the ratio between A and C (from notation of Table 1), it is obtained an optimal mixture that leads to better classification results, using either SVM or RandomForest classifiers.

We employ the McNemar's significance test to verify the statistical difference between performances of the term weighting schemes. We report these results in Table 4, where each column is related to a dataset and a classifier and the weighting schemes are shown in decreasing order of effectiveness. Schemes not separated with lines do not give significantly different results, a single line denotes that the schemes above perform better than the schemes below with a significance level between 0.01 and 0.001 ($A > B$), while a double line denotes a significance level better than 0.001 ($A >> B$). Results on the *Reuters-10* corpus with Random Forest are missing because there are no significant statistical differences between them. The table shows that our proposed *tf.idfec-based* scheme always provides top effectiveness and that with SVM classifiers, either linear

Table 3. Micro-averaged F_1 (in thousandths) best results obtained with different global weighting factors ("b." is short for "based") on topic classification with different learning algorithms. The best result for each dataset and algorithm is marked in bold.

global wt. $\rightarrow$	*idfec*	*idfec-b.*	*rf*	*icf-b.*	*idf*	χ^2	*gss*	*ig*	*midf*	*oddsR*	none
Reuters-10											
SVM(LIN)	.929	**.933**	**.933**	.930	.930	.847	.863	.852	.920	.918	.932
SVM(RBF)	.932	**.937**	**.937**	.935	.933	.879	.895	.882	.923	.926	.934
RandomForest	**.904**	.902	.903	.899	.903	.901	.901	.903	.898	.903	.902
Reuters-52											
SVM(LIN)	.920	**.925**	.922	.916	.917	.828	.848	.822	.882	.890	.912
SVM(RBF)	.925	**.927**	.926	.924	.922	.848	.873	.848	.886	.895	.915
RandomForest	.855	**.868**	.867	.853	.858	.863	.861	.864	.858	.863	.866
20 Newsgroups											
SVM(LIN)	.754	**.759**	**.759**	.747	.741	.567	.512	.520	.606	.666	.709
SVM(RBF)	.712	**.713**	.712	**.713**	.702	.587	.555	.56	.609	.664	.677
RandomForest	.536	**.570**	.563	.537	.543	.568	**.570**	.565	.536	.562	.566

or RBF kernel, some term weighting methods are more efficient than others. The best methods in general seem to be the latest supervised methods, such as *tf.idfec-based*, *tf.rf*, *tf.icf-based* and *tf.idfec*. Instead, using RandomForest, the classic supervised methods seem to work better, with results comparable or slightly below with respect to *tf.idfec-based*.

Let's now observe how classification effectiveness varies by changing the number of features considered to create the dictionary of the VSM. We show results on Reuters-10 and 20 Newsgroups using a linear SVM classifier in Figure 1; results on Reuters-52 mostly follow the same trends discussed below for Reuters-10. To keep the plots readable, we show the results on a selection of schemes with best performances: *tf.idfec*, *tf.idfec-based*, *tf.rf*, *tf.icf-based*, *tf.idf*.

The plot for Reuters-10 shows that, when using *tf.idfec-based* we obtain the best results by using few features per category, considering the variations of p described in Section 4.2. We note that the best results on *Reuters-10* are obtained with 150 features per category and on *Reuters-52* (not shown) with 75 features, corresponding respectively to overall dictionary sizes of 498 and 970. However, we can see as the effectiveness of the classification using *tf.idfec-based* deteriorates by increasing the number of features considered, therefore introducing terms less frequent and discriminative. Analysing the behavior of the schemes from which *idfec-based* takes its cue, i.e. standard *tf.idf* and *tf.rf*, we note that this performance degradation is probably due to the *idf* factor of the weight, as even the *tf.idf* has the same type of trend results, while *tf.rf* seems to remain stable at values comparable to the best results also by increasing the dictionary size.

The right plot of Figure 1 shows that to obtain the best results with the *20 Newsgroups* corpus is necessary a greater number of features. Using *tf.idfec-based* we obtain the best result with a dictionary of about 10000 features; after that

Table 4. McNemar's significance test results. Each column is related to a dataset and a supervised classifier ('L' stands for linear kernel, 'R' for RBF), the global weighting factors are shown in decreasing order of effectiveness, separating groups of schemes by significance of differences in their perfomances. Results for RandomForest on *Reuters-10* are omitted as no significant difference is observed between them.

Reuters-10		*Reuters-52*			*20 Newsgroups*		
SVM(L)	SVM(R)	SVM(L)	SVM(R)	RndFor.	SVM(L)	SVM(R)	RndFor.
idfec-b.	***idfec-b.***	***idfec-b.***	***idfec-b.***	***idfec-b.***	***idfec-b.***	***idfec-b.***	***idfec-b.***
rf	*rf*	*rf*	*rf*	*rf*	*rf*	*icf-b.*	*gss*
1	*icf-b.*	***idfec***	***idfec***	1	***idfec***	***idfec***	χ^2
icf-b.	1	*idf*	*icf-b.*	*ig*	*icf-based*	*rf*	1
idf	*idf*	*icf-b.*	*idf*	*oddsR*	*idf*	*idf*	*ig*
idfec	***idfec***	1	1	χ^2	1	1	*rf*
midf	*oddsR*	*oddsR*	*oddsR*	*gss*	*oddsR*	*oddsR*	*oddsR*
oddsR	*midf*	*midf*	*midf*	*idf*	*midf*	*midf*	*idf*
gss	*gss*	*gss*	*gss*	*midf*	χ^2	χ^2	*icf-b.*
ig	*ig*	*ig*	χ^2	***idfec***	*ig*	*ig*	***idfec***
χ^2	χ^2	χ^2	*ig*	*icf-b.*	*gss*	*gss*	*midf*

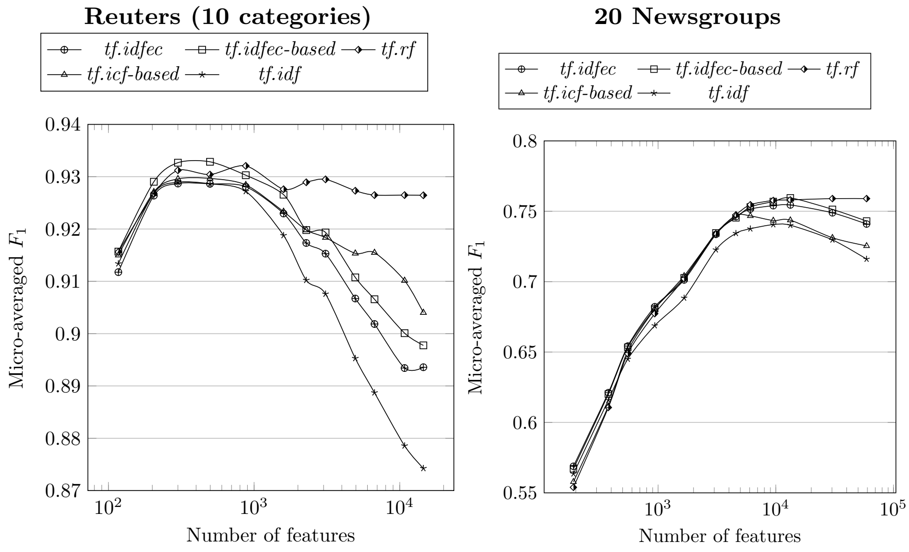


Fig. 1. Results obtained on topic classification datasets varying the number of top p features per category using a SVM classifier with linear kernel. The X axis (in logarithmic scale) indicates the resulting total number of features.

Table 5. Top accuracies (in thousandths) obtained with different global weighting factors ("b." is short for "based") on sentiment classification datasets with different learning algorithms. The best result for each dataset and algorithm is marked in bold.

global wt. $\rightarrow$	*idfec*	*idfec-b.*	*rf*	*idf*	*ig*	*midf*	none	Δidf	Δidf_{sm}	*wllr*	cd_M
Movie Review Data											
SVM(LIN)	.842	**.845**	.842	.842	.837	.83	.837	.797	.827	.79	**.845**
SVM(RBF)	.802	.797	.797	.795	.787	.795	.787	**.807**	**.807**	.802	.792
RandomForest	.81	.815	.817	**.835**	.82	.817	.82	.807	**.835**	.812	.805
MDSD *books*											
SVM(LIN)	.802	.802	**.81**	.802	.805	.802	.805	.77	.797	.767	.8
SVM(RBF)	.742	.76	.755	.745	.745	.747	.745	**.765**	**.765**	.727	.75
RandomForest	.815	.812	**.827**	.807	.807	.817	.817	.755	.78	.745	.8
MDSD *dvd*											
SVM(LIN)	.815	.817	.83	.81	.815	.812	.815	.792	.807	.802	**.825**
SVM(RBF)	**.75**	.737	.735	.725	.715	.712	.715	.722	.722	.71	.717
RandomForest	**.812**	.807	.82	**.812**	.807	.807	.805	.77	.785	.77	.817
MDSD *electronics*											
SVM(LIN)	.842	.835	**.847**	.84	.845	.837	.845	.822	.842	.81	.842
SVM(RBF)	.772	.767	.78	.765	.762	.772	.762	**.787**	**.787**	.755	.765
RandomForest	**.847**	.825	.842	.82	.83	.825	.832	.817	.832	.832	.835
MDSD *kitchen*											
SVM(LIN)	.887	.887	**.89**	.882	.887	.885	.887	.837	.865	.857	.885
SVM(RBF)	.82	.817	.815	.81	.81	.81	.81	**.822**	**.822**	.81	.815
RandomForest	.857	.862	.867	.857	.86	.862	**.882**	.84	.847	.825	.872

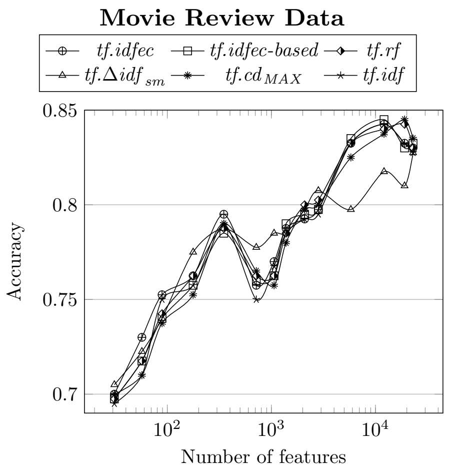
the efficiency shows a slight decrease, but more moderate than that shown in the Reuters dataset.

5.2 Sentiment Classification

We conceived our supervised weighting schemes for text classification by topic, achieving optimal results in it. However, we tested them also on the sentiment classification, checking whether they still guarantees good accuracy levels.

In addition to those considered above, we included in the new tests some weighting schemes especially tailored for sentiment classification: *delta idf* and variants thereof, *wllr* and cd_{MAX}. As only two classes are considered (positive and negative reviews), we used higher values for the number p of features per class, up to 19,200, as the total number of resulting features is generally lower for equal values of p. Always given the two-class setting, we evaluate here the accuracy of a classifier simply as the ratio of correctly classified test documents, as discussed above.

In Table 5, likely to above, we report the best accuracy obtained testing different values of p on the tested schemes: to give space to sentiment classification-specific schemes we removed some of the previous ones whose results were not



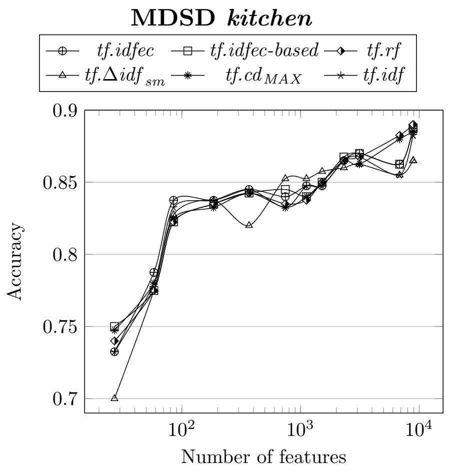


Fig. 2. Results obtained on sentiment classification datasets varying the number of top p features per category using a SVM classifier with linear kernel. The X axis (in logarithmic scale) indicates the resulting total number of features.

among the best. In these tests, our schemes outperformed other ones only in some cases, with *tf.rf* and especially new schemes yielding strong performances.

Should be noted however that the differences between performances in these tests are smaller than those observed in topic classification. While some schemes generally perform better than others, the gaps are not significant in most cases. McNemar's significance test confirms this observation: also due to the relatively low number of available test documents, the performance difference between almost every possible couple of weighting schemes is statistically not significant.

The plots in Figure 2 show how the accuracy on the Movie Review data and the MDSD *kitchen* domain using a SVM classifier with linear kernel varies for the tested values of p on a selection of the best performing weighting schemes: *tf.idf*, *tf.rf*, $tf.\Delta idf_{sm}$, $tf.cd_{MAX}$, *tf.idfec* and *tf.idfec-based*. The trends for other tested domains of MDSD are similar enough to those for *kitchen*, with small differences mostly on the overall average accuracy level.

With respect to classification by topic, we see again that the difference of performance between different weighting schemes is more restrained; this holds also for most of the schemes not shown in the plots. The number of features per class has instead an important impact on accuracy, with an especially fast growth for lower values of p in MDSD: as discussed above, this is due to the presence of only two classes, yielding an overall low number of features.

Comparing the accuracy trends as p varies for the different schemes, as above, there is no clear indication of which one is better for different numbers of considered features. However, we still notice a slight positive trend for *tf.idfec* and *tf.idfec-based* with narrower selections of features and occasionally in other situations. Comparing the two schemes, contrarily to topic classification where the *tf.idfec-based* variant always outperforms the *tf.idfec* base, the latter here sometimes yields a slightly better accuracy.

For what regards other schemes, *tf.rf* and $tf.cd_{MAX}$ exhibit some performance peaks, especially with higher numbers of features; $tf.\Delta idf_{sm}$ has instead a trend which slightly differentiates it from other ones, with some significant differences of accuracy, either positive or negative.

6 Conclusions

Starting from the general text classification problem, considering both categorization by topic and distinction between positive and negative sentiment, we briefly reviewed existing methods for both unsupervised and supervised and proposed a novel solution as a modification of the classic *tf.idf* scheme. We devised a base *tf.idfec* scheme where *idf* is computed without considering documents belonging to the modeled category: this prevents giving a low weight to terms largely present in it. The *tf.idfec-based* variant mixes our base scheme with the relevance frequency from *tf.rf*, thus also effectively boosting weights of terms which appear frequently in the category under analysis.

We performed extensive experimental studies on benchmark datasets for text classification by topic and by sentiment: the *Reuters* corpus with either 10 or 52 categories and *20 Newsgroups* for the former, the *Movie Review Data* and the *Multi-Domain Sentiment Dataset* for the other. We compared our two weighting schemes against other known ones using both SVM and Random Forest learning algorithms and different levels of feature selection.

The results show that the *tf.idfec-based* method combining *idfec* and *rf* generally gets top results for topic classification. Through statistical significance tests, we showed that the proposed scheme always achieves top effectiveness and is never worse than other methods. The results show a close competition between our *tf.idfec-based* and *tf.rf*: the best results obtained with the different datasets and algorithms, varying the amount of feature selection, are very similar, but with some differences. *tf-rf* seems more stable when the number of features is high, while our *tf.idfec-based* gives excellent results with few features and shows some decay (less than 4%) when the number of features increases.

On the other side, tests on sentiment classification suggest that gaps between different weighting schemes, including some of them specifically devised for this task, are not statistically significant, although also due in our case to the low number of test documents. Here our schemes are not superior to other ones, as they have not been developed for this task, but generally appear to be nearly as good as those yielding better accuracy estimates.

In the future, we plan to test further variants of our scheme, possibly inspired by existing ones and trying to obtain significantly good results also in sentiment classification, for which we aim to perform tests on larger datasets. Also, we are considering to expand the study of existing term weighting schemes for text categorization and sentiment analysis to a more complete survey, complete of experimental comparison of their performances on several datasets.

References

1. Blitzer, J., Dredze, M., Pereira, F.: Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In: Association of Computational Linguistics. vol. 7, pp. 440–447 (2007)
2. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
3. Carmel, D., Mejer, A., Pinter, Y., Szpektor, I.: Improving term weighting for community question answering search using syntactic analysis. In: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management. pp. 351–360. CIKM '14, ACM, New York, NY, USA (2014), http://doi.acm.org/10.1145/2661829.2661901
4. Debole, F., Sebastiani, F.: Supervised term weighting for automated text categorization. In: In Proceedings of SAC-03, 18th ACM Symposium on Applied Computing. pp. 784–788. ACM Press (2003)
5. Deisy, C., Gowri, M., Baskar, S., Kalaiarasi, S., Ramraj, N.: A novel term weighting scheme midf for text categorization. Journal of Engineering Science and Technology 5(1), 94–107 (2010)
6. Deng, Z.H., Luo, K.H., Yu, H.L.: A study of supervised term weighting scheme for sentiment analysis. Expert Systems with Applications 41(7), 3506–3513 (2014)
7. Deng, Z.H., Tang, S.W., Yang, D.Q., Li, M.Z.L.Y., Xie, K.Q.: A comparative study on feature weight in text categorization. In: Advanced Web Technologies and Applications, pp. 588–597. Springer (2004)
8. Dietterich, T.G.: Approximate statistical tests for comparing supervised classification learning algorithms. Neural Comput. 10(7), 1895–1923 (Oct 1998), http://dx.doi.org/10.1162/089976698300017197
9. Domeniconi, G., Masseroli, M., Moro, G., Pinoli, P.: Random perturbations of term weighted gene ontology annotations for discovering gene unknown functionalities. In: Fred, A., Dietz, J.L.G., Aveiro, D., Liu, K., Filipe, J. (eds.) Knowledge Discovery, Knowledge Engineering and Knowledge Management, Communications in Computer and Information Science, vol. 553, pp. 181–197. Springer International Publishing (2015), http://dx.doi.org/10.1007/978-3-319-25840-9_12
10. Domeniconi, G., Moro, G., Pagliarani, A., Pasolini, R.: Markov chain based method for in-domain and cross-domain sentiment classification. In: International Conference on Knowledge Discovery and Information Retrieval (KDIR 2015) (2015)
11. Domeniconi, G., Moro, G., Pasolini, R., Sartori, C.: Cross-domain text classification through iterative refining of target categories representations. In: Proceedings of the 6th International Conference on Knowledge Discovery and Information Retrieval (2014)
12. Domeniconi, G., Moro, G., Pasolini, R., Sartori, C.: Iterative refining of category profiles for nearest centroid cross-domain text classification. In: Fred, A., Dietz, J.L.G., Aveiro, D., Liu, K., Filipe, J. (eds.) Knowledge Discovery, Knowledge Engineering and Knowledge Management, Communications in Computer and Information Science, vol. 553, pp. 50–67. Springer International Publishing (2015), http://dx.doi.org/10.1007/978-3-319-25840-9_4
13. Domeniconi, G., Moro, G., Pasolini, R., Sartori, C.: A study on term weighting for text categorization: A novel supervised variant of tf.idf. In: 4th International Conference on Data Management Technologies and Applications (DATA 2015) (2015)
14. Fattah, M.A.: New term weighting schemes with combination of multiple classifiers for sentiment analysis. Neurocomputing (2015)

15. Galavotti, L., Sebastiani, F., Simi, M.: Experiments on the use of feature selection and negative evidence in automated text categorization. In: Research and Advanced Technology for Digital Libraries, pp. 59–68. Springer (2000)
16. Lan, M., Sung, S.Y., Low, H.B., Tan, C.L.: A comparative study on term weighting schemes for text categorization. In: Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on. vol. 1, pp. 546–551. IEEE (2005)
17. Lan, M., Tan, C.L., Su, J., Lu, Y.: Supervised and traditional term weighting methods for automatic text categorization. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(4), 721–735 (2009)
18. Leopold, E., Kindermann, J.: Text categorization with support vector machines. how to represent texts in input space? Mach. Learn. 46(1-3), 423–444 (Mar 2002), http://dx.doi.org/10.1023/A:1012491419635
19. Lewis, D.D.: Evaluating and optimizing autonomous text classification systems. In: Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 246–254. SIGIR '95, ACM, New York, NY, USA (1995), http://doi.acm.org/10.1145/215206.215366
20. Luo, Q., Chen, E., Xiong, H.: A semantic term weighting scheme for text categorization. Expert Syst. Appl. 38(10), 12708–12716 (Sep 2011), http://dx.doi.org/10.1016/j.eswa.2011.04.058
21. Martineau, J.C., Finin, T.: Delta tfidf: An improved feature space for sentiment analysis. In: Third International AAAI Conference on Weblogs and Social Media (2009)
22. Paltoglou, G., Thelwall, M.: A study of information retrieval weighting schemes for sentiment analysis. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. pp. 1386–1395. ACL '10, Association for Computational Linguistics, Stroudsburg, PA, USA (2010), http://dl.acm.org/citation.cfm?id=1858681.1858822
23. Papineni, K.: Why inverse document frequency? In: Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies. pp. 1–8. Association for Computational Linguistics (2001)
24. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. 34(1), 1–47 (Mar 2002)
25. Song, S.K., Myaeng, S.H.: A novel term weighting scheme based on discrimination power obtained from past retrieval results. Inf. Process. Manage. 48(5), 919–930 (Sep 2012), http://dx.doi.org/10.1016/j.ipm.2012.03.004
26. Tokunaga, T., Makoto, I.: Text categorization based on weighted inverse document frequency. In: Special Interest Groups and Information Process Society of Japan (SIG-IPSJ. Citeseer (1994)
27. Tsai, F.S., Kwee, A.T.: Experiments in term weighting for novelty mining. Expert Systems with Applications 38(11), 14094–14101 (2011)
28. Wang, D., Zhang, H.: Inverse-category-frequency based supervised term weighting schemes for text categorization. Journal of Information Science and Engineering 29(2), 209–225 (2013), http://www.scopus.com/inward/record.url?eid=2-s2.0-84876262286&partnerID=40&md5=00897fb85a83bfd704cb2428254e1950
29. Wu, H., Gu, X.: Reducing over-weighting in supervised term weighting for sentiment analysis. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics (2014)