Text Classification - Naïve Bayes Algorithm to Verify New Term Weighting Scheme

Hatem A. Nassrat

December 5, 2008

Abstract

Text classification is of great interest to various users today, especially due to its potential use in the world wide web (aka internet). However, current text classification algorithms tend to not handle the imbalanced nature of documents that have a natural distribution of class labels. Classification algorithm tend to favour classes trained with a large number of documents rather. Classes with a small number of document examples tend to be ignored. This study evaluates a scheme that aims to solve this issue through modifying the term weighting scheme used to build inverted indices, which would be compatible with any classification algorithm that utilizes inverted indices.

1 Introduction

Text classification is pointed out to be of booming interest, due to the availability of electronic documents today. In text classification, data imbalance often occurs. This problem occurs when some classes have more examples than others. Using the standard term weighting approaches when classifying such skewed data, the minor categories are often ignored. In some cases we need such rare classes, for example when classifying data for cancer detection, or earthquake prediction [6].

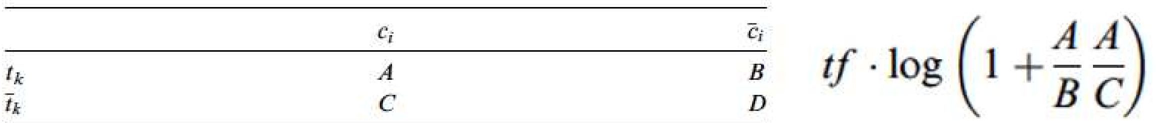
This study studies the text weighting scheme introduced by [6] and re-evaluates their method under new light. In [6] they have tested their scheme using Complement NB and Support Vector Machine classifiers, built using the Reuters 21578 and MVC datasets. Here we similarly use the Reuters 21578 dataset and implement the standard Naïve Bayesian algorithm.

The authors of [6] have left out all the implementation details of how they performed their evaluation, and classified their documents. In general that would not be a disadvantage due to these methods being around for a long period of time, however, in this case there was a single issue that required extra consideration.

Documents in the Reuters 21578 are tagged with multiple topic fields and this require a means of classifying each document with multiple classes. To solve this issue the system built for this study, used binary classifiers for each of the topical classes. These classifiers instructed whether the label should be attached to each of the documents or should not, thus producing a vector of topics for each document. For evaluation purposes this vector can be used to match up with the pre tagged documents, allowing for various evaluation measures.

2 Probability Based Term weighting

The term weighting scheme introduced in [6], utilizes a ratio between the number of document belonging to the class with the term being considered against the number of documents without that given term. Similarly it utilizes a ratio between the number of documents within the class having the term, against the number of documents outside the class having the term. Figure 1 shows the formula along with a diagram representing the meaning of each of the letters. The *tf* is the logarithm of the term frequency of a given term in the collection.



	c_i	$\bar{c}_i$
t_k	A	B
$\bar{t}_k$	C	D

$$tf \cdot \log\left(1 + \frac{A}{B}\frac{A}{C}\right)$$

Figure 1: Probalistic Weighting Scheme

3 Data Prep and System Design

Using the standard Reuters 21578 dataset, data structure has been built to house each document within this dataset. For your convenience we have briefly described this dataset bellow.

Reuters 21578

A dataset containing financial news documents from the early 90s. It has been manually labelled with tags, mapping documents to various classes. [2]

3.1 Reuters Document

The document data structure housed a parsed document. This data structure had two main attributes. The first being *words* and the second *topics*.

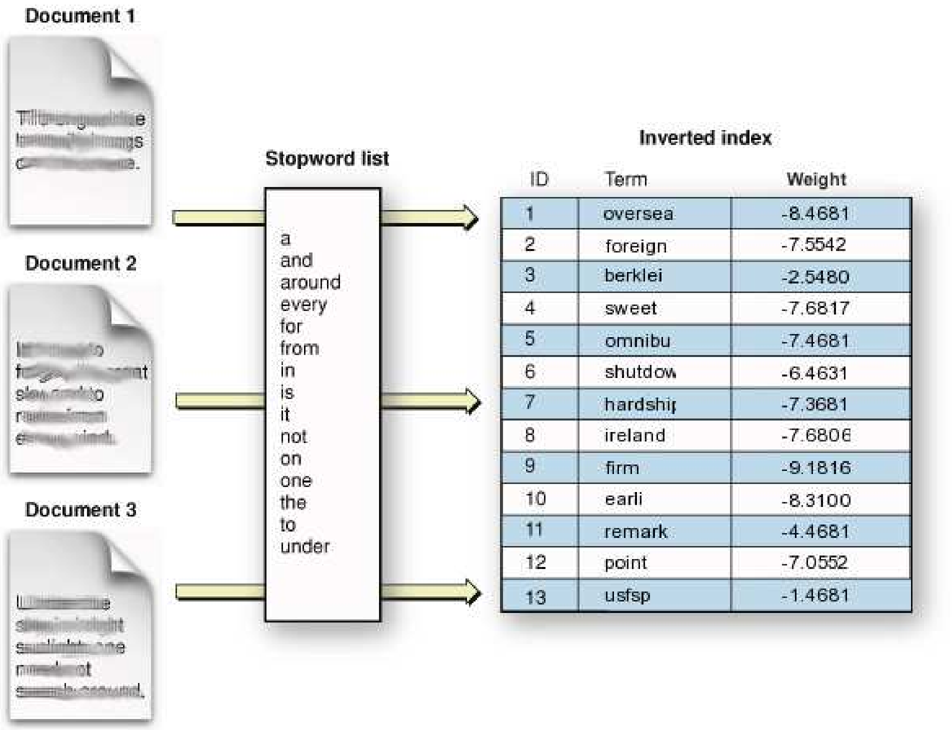
words

A hash map mapping words to their word frequencies within the document. Words appearing in the title of the document were given a higher weight than words appearing in its other sections.

topics

A list of parsed topics from the document. These topics were found under the *Topics* tag each enclosed within a *D* tag.

The Reuters 21578 contains documents which were not labelled with *Topic* tags. Such documents were pre filtered, along with documents that contained the attribute label "BY-



ID	Term	Weight
1	oversea	-8.4681
2	foreign	-7.5542
3	berklei	-2.5480
4	sweet	-7.6817
5	omnibu	-7.4681
6	shutdow	-6.4631
7	hardship	-7.3681
8	ireland	-7.6806
9	firm	-9.1816
10	earli	-8.3100
11	remark	-4.4681
12	point	-7.0552
13	usfsp	-1.4681

Figure 2: Snippet of one Inverted index weighted using Probalistic Weighting Scheme

PASS" and "NOT-USED", as they were not used by the dataset creators.

3.2 Inverted Index

To create the text classifier its inverted index must be built. Since we have multiple classifiers (one per topic), for each of the 120 topics an inverted index was built.

To build the index words in each of the documents have been filtered through a stop word list. Following this step a famous Porter Stemming algorithm was applied thus reducing the terms back to their common derived root. This inverted index, has been processed to utilize the term weighting formula directly. This is a very useful step when building the index as it reduces computation cost when using the index, as the biggest part of the computation has already been computed. This technique has been by many software like [1]. Figure 2 briefly displays this process with an instance of the dataset being represented.

A combined inverted index containing each of the indices for each of the topics is stored in a Berkeley Database file on the local file system.

3.3 Naïve Bayes Classifier

A standard Naïve Bayesian Classifier was built around each of the inverted index. For ease of implementation, the code represented a complete set of classifiers as one classifier object. This object exposes a method *classify* which accepts a document and returns the vector of topics that were positively classified by each of the sub classifiers.

Similarly methods for ease of evaluation of the classifier were added to the classifier. The method *accuracy* accepts a single document and returns the accuracy of the classified document. This is done by calculating the cosine similarity between the vector of topics that was classified against the vector of topics that was pre tagged. This function utilizes binary weights for each of the vectors. The second method *evaluate*, calculates the macro-averaged precision, the mar-averaged recall and the average *accuracy* of a given test set of documents.

4 Evaluation

Using the system design described in Section 3, we aimed to evaluate whether the new term weighting approach proposed in [6] is useful and performs as expected. To do so, it was decided to compare the Naïve Bayesian classifier produced using the new term weighting scheme against the standard TFIDF term weighting scheme. Moreover, we have decided to validate using 10 fold cross validation to verify the results.

To perform this task 20 classifiers had to be built in total, 10 for each system. Each classifier contains 120 sub classifiers one for each topic. The build process for this design using the entire Reuters dataset completes in under a day (less than 24 hours), along with the respective evaluation.

4.1 Results

As can be seen in figure 1, the results validate the findings in [6]. The proposed term weighting scheme performs with a much higher accuracy with such an imbalanced dataset such as the Reuters 21578. However, in terms of recall, the standard TFIDF algorithm has a much higher correctness rate. This indicates that the new term weighting scheme induces less false positives while the TFIDF system induces less false negatives (i.e. and returns more false positives, depending on your level of optimism).

Table 1: Accuracy, Precision and Recall

	Prob			**TFIDF**		
Cross-Fold	Accuracy	Prec	Recall	Accuracy	Prec	Recall
1	63.78%	64.00%	63.97%	09.49%	00.97%	96.31%
2	49.89%	50.48%	49.83%	09.17%	00.96%	91.93%
3	44.07%	44.85%	44.40%	08.39%	00.91%	82.36%
4	39.35%	39.99%	39.35%	07.81%	00.83%	77.28%
5	43.82%	44.76%	43.58%	08.11%	00.85%	80.77%
6	46.61%	47.06%	46.65%	08.21%	00.86%	81.74%
7	49.36%	49.35%	49.75%	08.27%	00.83%	84.72%
8	44.00%	44.42%	43.93%	08.02%	00.82%	80.84%
9	42.19%	42.72%	42.30%	07.85%	00.83%	78.13%
10	45.29%	45.91%	45.08%	08.20%	00.87%	82.01%
AVG	46.84%	47.35%	46.88%	08.35%	00.87%	83.61%

5 Conclusion

The new term weighting scheme proposed in [6] is very useful when utilizing imbalanced datasets. However, there are areas of improvement in terms of reducing the number of false negatives that result from the classifier built using this scheme. That being said, utilizing such a scheme when classifying documents is very useful compared to traditional weighting schemes as we require high precision more often than we do recall in most of our information retrieval needs.

References

[1] Ft3 documentation. http://ft3.sourceforge.net/v0.3/devguide.html.

[2] Machine learning for nlp. esslli 2007 course - reuters21578 xml collection. http://ronaldo.cs.tcd.ie/esslli07/data/reuters21578-xml/.

[3] Naive bayes classifier. http://en.wikipedia.org/wiki/Naive_Bayesian_classification.

[4] Precision and recall. http://en.wikipedia.org/wiki/Precision_and_recall.

[5] Search kit programming guide: Establishing a suitable source to search. http://developer.apple.com/documentation/UserExperience/Conceptual/SearchKitConcepts/searchKi

[6] Ying Liu, Han Tong Loh, and Aixin Sun. Imbalanced text classification: A term weighting approach. *Expert Systems with Applications*, 36(1):690 – 701, 2009.