

# Skinning the Cat: Comparing Alternative Text Mining Algorithms for Categorization

Russell Albright, James A. Cox, and Kevin Daly, SAS Institute, Cary, NC

## ABSTRACT

Applying category labels to textual documents can be useful for 1) Search Indexing, 2) Document Filtering, and 3) Summarization. Many different algorithms have been proposed for applying category labels to text documents. We compare and contrast different approaches to text mining using Enterprise Miner for Text.

## INTRODUCTION

The automatic classification of documents into categories is an increasingly important task. As document collections continue to grow at remarkable rates, the task of classifying the documents by hand can become unmanageable. However, without the organization provided by a classification system, the collection as a whole is nearly impossible to comprehend and specific documents are difficult to locate. Examples of document collections that are often organized into categories include web pages, patents, news articles, email, research papers, and various knowledge bases.

Text categorization techniques have traditionally determined a subset of terms that are most diagnostic of particular categories and then tried to predict the categories using the weighted frequencies of each of those terms in each document. We will refer to this technique as the truncation approach (since only a subset of terms are used). This approach is subject to several deficiencies:

1. It does not take into account terms that are highly correlated with each other, such as synonyms. As a result, it is very important to employ a useful stemming algorithm, as well.
2. Documents are rated close to each other only according to co-occurrence of terms. Documents may be semantically similar to each other while having very few of the truncated terms in common. Most of these terms only occur in a small percentage of the documents.
3. The terms used need to be recomputed for each category of interest.

These problems present themselves also for text retrieval; as a result it has become de rigeur to use a reduced-dimensionality *vector-space* model when retrieving documents using search terms. In the vector-space model, vectors in a multi-dimensional space can represent both documents and terms. To determine which documents match retrieval terms, *Latent Semantic Analysis* [3] is used to find the nearest documents in that space to the search terms.

We believe that the use of a reduced-dimensionality normalized vector-space model to represent documents in multi-dimensional space can be useful for both classification and categorization of text documents, particularly in the context of categorization approaches that are based on Euclidean distances between documents, such as discriminant analysis, neural networks, and memory-based reasoning.

In this paper, we compare the singular value decomposition (SVD) technique for projecting documents into a  $k$ -dimensional subspace to that of truncation, using Enterprise Miner for Text. Different techniques for weighting the terms for both approaches are compared, and both neural networks and memory-based

Cat	Document
fin	1 <u>route</u> the <u>cash</u> and the <u>check</u> to the <u>bank</u>
fin	2 <u>borrow</u> with <u>credit</u> <u>borrow</u> based on <u>credit</u>
fin	3 I can <u>borrow</u> <u>cash</u> from the <u>bank</u>
riv	4 <u>river</u> <u>boat</u> <u>floats</u> up the <u>river</u>
riv	5 <u>boat</u> is by the <u>dock</u> near the <u>bank</u>
riv	6 the <u>river</u> <u>boat</u> is on the <u>south</u> <u>bank</u>
riv	7 the <u>boat</u> <u>floats</u> by the <u>dock</u> near the <u>river</u> <u>bank</u>
par	8 <u>check</u> the <u>parade</u> <u>route</u> to see the <u>floats</u>
par	9 the <u>parade</u> the <u>parade</u> the <u>parade</u>

		Documents										
		1	2	3	4	5	6	7	8	9		
T e r m s	route	1	1								1	
	cash	2	1	1								
	check	3	1								1	
	bank	4	1		1	1		1	1			
	borrow	5		2	1							
	credit	6		2								
	river	7				2		1	1			
	boat	8				1	1	1	1			
	floats	9				1			1	1		
	dock	10					1		1			
	south	11						1				
	parade	12									1	3

Example 1

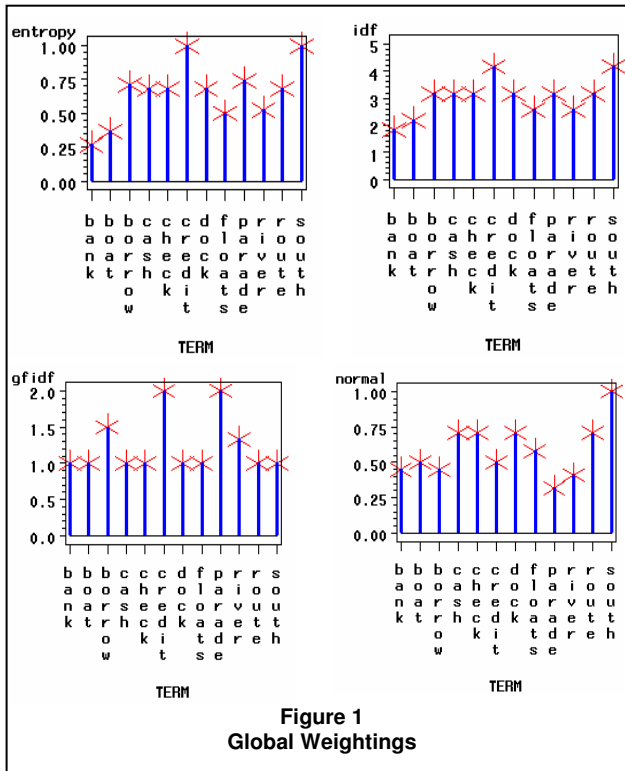
reasoning are used to predict the categories based on the result of whichever reduction technique is used.

The collection that we examine is the Reuters-21578 version of the Reuter's benchmark. This collection has become a standard for comparing alternative information retrieval and text mining techniques. Dumais *et al.* [4] provide a summary of several categorization approaches that have been applied to this document collection.

## BACKGROUND

The Enterprise Miner for Text uses a vector space model [6] for representing the collection of documents. In this approach, documents are represented as vectors of length  $n$ , where  $n$  is the number of unique terms that are indexed in the collection. The vector for each document is typically very sparse because few of the terms in the collection as a whole are contained in any one given document. The entries in the vector are the frequency that each term occurs in that document. If  $m$  is the number of documents in the collection, we now have an  $n$  by  $m$  matrix  $A$  that represents the document collection. Typically, the matrix is oriented with the rows representing terms and the columns representing documents.

Example 1 displays a sample document collection containing 9 documents. Twelve terms are being indexed. The remaining terms would be removed by a stop list. Each document belongs to one of the categories financial (fin), river (riv) or parade (par).



The matrix at the bottom of the example is a term-document frequency matrix. We leave the cells with a zero entry empty for readability. (Enterprise Miner for Text actually uses a compressed form of this sparse matrix.)

### WEIGHTINGS

Generally the entries in the term-document frequency matrix  $A$  are adjusted by a weighting factor [6]. These weightings can be critically important for developing a good categorization model.

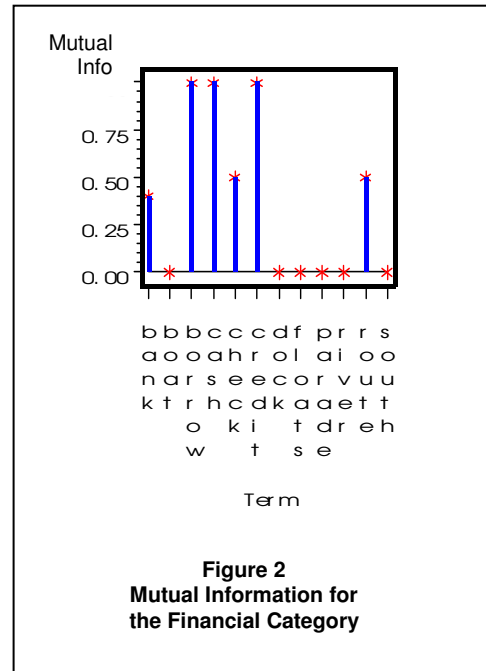
Without taking into consideration category-specific information, there are two types of weightings that can be applied to the frequency matrix; local weights (or cell weights) and global weights (or term weights). Local weights are created by applying a function to the entry in the cell of the term-document frequency matrix. Global weights are functions of the rows of the term-document frequency matrix. As a result, local weights only deal with the frequency of a given term within a given document, while global weights are functions of how the term is spread out across the collection.

It is generally beneficial to assign the words that occur frequently, but in relatively few documents, a high weight. The documents that contain those terms will be easy to set apart from the rest of the collection. On the other hand, terms that occur in every document should receive a low weight because of their inability to discriminate between documents.

### Local Weightings

Enterprise Miner for Text offers two variations of local weights. The binary local weight sets every entry in the frequency matrix to a 1 or a 0. In this case, the number of times the term occurred is not considered important. Only information about whether the term did or did not appear in the document is retained.

A less drastic approach is the log weighting. For this local weight, each entry is operated on by the log function. Large frequencies are dampened but they still contribute more to the model than terms that only occurred once.



### Global Weightings

Global weights are functions of how the word is distributed throughout the collection as a whole. These weightings are used to help determine which terms have the most discriminating power. Generally, the best terms are those that occur frequently, but only in a few documents.

Besides the option of not applying a global weight, Enterprise Miner for Text offers four weighting schemes:

1. Entropy – This setting actually calculates one minus the scaled entropy so that the highest weight goes to terms that occur infrequently in the document collection as a whole, but frequently in a few documents.
2. Inverse Document Frequency (IDF) – Dividing by the document frequency is another approach that emphasizes terms that occur in few documents.
3. Global Frequency Times Inverse Document Frequency – Magnifies the inverse document frequency by multiplying by the global frequency.
4. Normal – Scales the frequency. Entries are proportional to the entry in the term-document frequency matrix.

In Figure 1, we apply the four global weights to the document collection given in Example 1. The plots reveal the weighting for each of the indexed words. The term "bank" which is contained in many of the documents has a low weight in each of the cases. On the other hand, most of the schemes assign a high weight to "parade" which occurs three times but in a single document.

### Mutual Information

It is also possible to implement weighting schemes that make use of the target variable. Yang *et al.* [7] investigate the use of several weighting approaches including information gain,  $\chi^2$  and mutual information. We performed several preliminary investigations that showed that mutual information and  $\chi^2$  performed well with the SVD approach. For this paper, we only considered the mutual information weighting scheme.

Figure 2 gives the mutual information weightings (scaled to be

		Terms						
		c a s h	c h e c k	b o r r o w	c r e d i t	d o c k	s o u t h	p a r a d e
D o c u m e n t s	1	1	1					
	2			2	2			
	3	1		1				
	4							
	5					1		
	6						1	
	7					1		
	8		1					1
	9							3

Table 1

between 0 and 1) for the terms in the financial category of Example 1. Terms that only appear in the financial category have a weight of 1, terms that do not appear in the financial category have a weight of 0, and terms that appear in both categories have a weight between 0 and 1. Note how different these weightings are than in the four graphs in Figure 1.

**DIMENSION REDUCTION AND TRUNCATION**

Since the number of words in the collection can easily reach hundreds of thousands, a method for reducing the number of variables in the weighted term-frequency matrix must be introduced. The most common approach for text mining is truncation.

Truncation involves discarding words in the term-document frequency matrix that have a small weight. Although Example 1 has very few dimensions, we demonstrate the approach using the entropy weighting (see the first bar chart in Figure 1). Based on the chart, we may decide to index only the terms "borrow", "cash", "check", "credit", "dock", "parade", and "south" because these were the k=7 terms with the highest entropy weighting.

As a result, we reduce the dimension of our problem from 12 to 7 by using the contents of Table 1 below rather than the representation contained in Example 1. Note also that we have transposed the results so that observations are documents and variables are terms.

The use of the representation in Table 1, although it is more condensed than that given in Example 1, still makes it difficult to compare documents. Notice that if we use the co-occurrence of items from the matrix above as a measure of similarity, then Documents 1 and 8 are more similar than Documents 1 and 2 (compare the columns with one another). This is true in both Table 1 and Table 2. This is because Documents 1 and 8 share the word "check", while Documents 1 and 2 have no words in common. In actuality, however, Documents 1 and 8 are not related at all, but Documents 1 and 2 are very similar.

**DIMENSION REDUCTION AND THE SVD**

As an alternative to the truncation method, Enterprise Miner for Text can also reduce the dimension by using the singular value decomposition (SVD) of the weighted term-document frequency matrix. The mathematics of this are detailed in the appendix. As a result, documents are represented as vectors in the best-fit k-dimensional subspace. The similarity of two documents can be assessed by the dot products of the two vectors. In addition the dimensions in the subspace are orthogonal to each other.

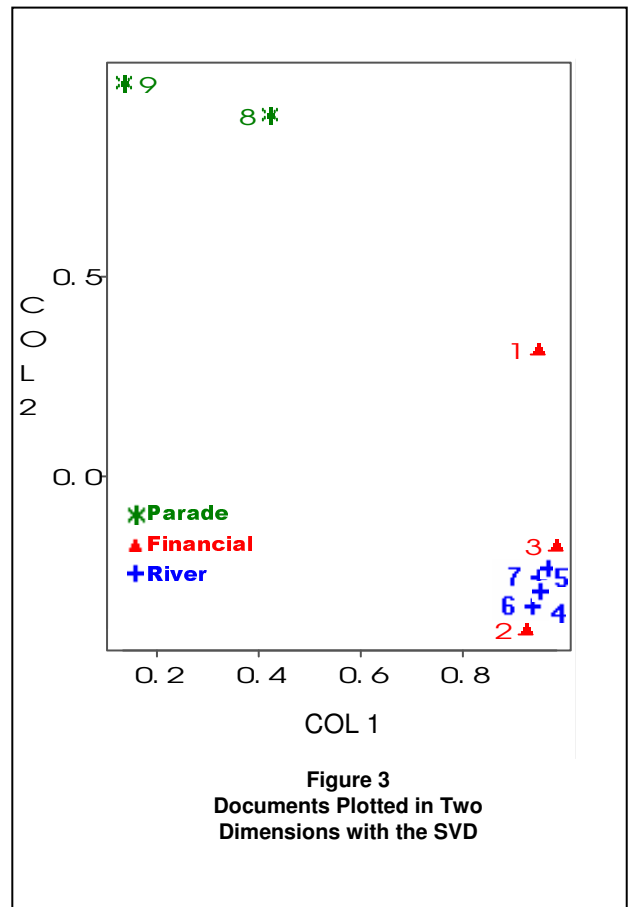


Figure 3 Documents Plotted in Two Dimensions with the SVD

Unfortunately, most clustering and predictive modeling algorithms work by segmenting Euclidean distance, so Enterprise Miner for Text also normalizes the document vectors to have a length of one. This essentially placing each one on the unit hypersphere, so that Euclidean distances between points will directly correspond to the dot products of their vectors.

As an example of this approach, we apply the SVD dimension reduction technique to Example 1. We choose to reduce to 2 dimensions and we plot the results in Figure 3. Note that this two-dimensional projection correctly places Document 1 closer to Document 2 than it is to Document 8, even though the word overlap is less. This is due to the ability of the SVD to take into account semantic similarity rather than simple word similarity. Note also the circular arrangement of the points --- due to the normalization, the points in two dimensions are arranged in a half-circle.

In this small example, setting k to be two is sufficient to incorporate much of the similarity information. In larger, real-world examples, many more dimensions are required, anywhere from several to several hundred, depending on the domain. It should be small enough that most of the noise is incorporated in the non-included dimensions, while including most of the signal in the reduced dimensions.

Category	train	val	test
earn	2305	572	1087
acq	1242	408	719
money_fx	345	193	179
grain	303	129	149
crude	324	65	189
trade	274	95	117
interest	229	118	131
wheat	144	67	71
ship	136	61	89
corn	140	41	56

Table 2

Category	Best with Local-Global	Local-Global Used	Mutual Information Results
earn	98.5	log-ent	97.5
acq	94.0	log-idf	87.0
money-fx	82.1	log-idf	68.3
grain	83.2	log-ent	88.9
crude	86.8	log-idf	78.2
trade	81.1	log-ent	66.6
interest	77.6	log-ent	58.7
ship	83.5	bin-ent	60.2
wheat	77.8	log-idf	79.1
corn	67.3	log-ent	80.1
microavg.	89.7		83.7

Table 3

## PREDICTIVE MODELING TOOLS

### MEMORY-BASED REASONING

In memory-based reasoning, a predicted value for a dependent variable is determined based on retrieving the  $k$  nearest neighbors to this example and having them vote on the value. This is potentially useful for categorization when there is no rule that defines what the target value should be.

Memory-based reasoning works particularly well when the terms have been compressed using the SVD, since the Euclidean distance is a natural measure for determining the nearest neighbors.

### NEURAL NETWORKS

We also investigated predictive performance with a nonlinear neural network containing two hidden layers. Nonlinear neural networks are capable of modeling higher-order term interaction.

An advantage of neural networks is the ability to predict multiple binary targets simultaneously by a single model. Unfortunately, when the term weighting is dependent on the category (as in mutual information) a separate network must be trained for each category.

## THE REUTER'S COLLECTION

The Reuter's data set is a collection of over 20,000 Reuter's business newswire posts. There are several versions of the Reuter's collection. We are working with the Reuter's-21578 version which is available from <http://www.research.att.com/~lewis/reuters21578.html>.

Various testing-training splits have been used for the collection. For our study, we used the Modapte split. This split places 9603 stories into the training data and 3299 stories for testing. Each article in the split has been assigned to one or more of a total of 118 categories. Three of the categories have no training data associated with them and many of the categories are underrepresented in the training data. For this reason we present our results for the top ten most often occurring categories.

The Modapte split separates the collection chronologically for the test-training split. The oldest documents are placed in the training set and the most recent documents placed in the testing set. The split does not contain a validation set. We created a validation set by partitioning the Modapte training data into two data sets

chronologically. The first 75% of the Modapte training documents were used for our training set and the remaining 25% were used for validation.

The top ten categories are listed in Table 2, along with the number of documents available for testing, validation and training.

## RESULTS

All the results given in this section were derived after first removing nondiscriminating terms such as articles and prepositions with a stop list. We also did not consider any terms that occurred in fewer than 2 of the documents in the training data.

### CHOICE OF WEIGHTS

For the choice of local and global weights, there are 15 different combinations. We used the SVD and MBR while varying  $k$  in order to determine which weighting combination worked best.

Finally, we compared the mutual information weighting criterion with the various combinations of local and global weighting schemes.

In order to investigate which weightings provide the best predictive modeling, we classified the documents after doing a singular value decomposition using values of  $k$  in increments of 10 from  $k=10$  to  $k=200$ . For each value of  $k$  and every category, we applied each combination of local and global weights. We then considered the average of precision and recall (see the appendix) in order to determine which weightings and dimensions worked the best. For this experiment, our predictive model was built with the memory-based reasoning node.

Table 3 summarizes our findings by comparing the best local-global weighting scheme for each category with the mutual information result. The results varied among the categories and among the various values for  $k$ , but the log-entropy, and log-idf weighting combinations consistently performed well. The binary-entropy and binary-idf also performed fairly well. Other combinations of weightings did not perform as well. The *microavg* category at the bottom was determined by calculating a weighted average based on the number of documents that were contained in each of the ten categories.

### SVD AND THE NUMBER OF DIMENSIONS

Our investigation of the weighting combinations also revealed to

	SVD MBR	SVD Neural	Trunc MBR	Trunc Neural
earn	<b>97.5</b>	97.3	96.3	96.7
acq	87.0	<b>90.5</b>	74.9	83.5
money-fx	68.3	<b>70.0</b>	50.7	55.1
grain	88.9	87.7	87.6	<b>90.6</b>
crude	78.2	<b>84.4</b>	73.0	78.1
trade	<b>66.6</b>	65.5	62.7	60.8
interest	58.7	<b>66.2</b>	51.7	42.7
ship	60.2	<b>82.3</b>	60.8	59.6
wheat	79.1	83.1	<b>85.6</b>	77.0
corn	80.1	79.5	<b>87.1</b>	80.4
microavg.	83.7	<b>86.2</b>	77.9	79.8

Table 4

us a reasonable range for the value of the number of dimensions  $k$ . Depending on the category and the weighting combination, the optimal values of  $k$  varied from 20 to as much as 200. Within this range of values, there were often several local maximum values, but as long as  $k$  remained in the range from 40-200, the average of precision and recall were relatively insensitive to changes in  $k$ .

In order to get a better understanding of the significance of  $k$ , we chose to examine two categories more closely, earn and corn. Of the ten categories, the earn category gave us our highest accuracy and the corn category gave the poorest results. In Figure 4, we plot the results of using MBR to predict the corn category with values of  $k$  ranging from 20 to 200. The plot contains a pair of results, one considering the log-entropy weighting and the other using mutual information. For this category, neither the value for  $k$  nor the weighting made a significant difference in the average of precision and recall. Our average of precision and recall for the earn category remained nearly constant for values of  $k$  from 20 to 200.

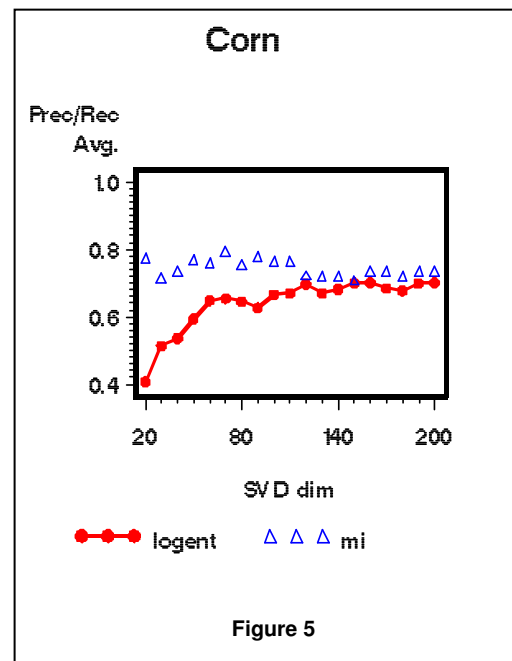
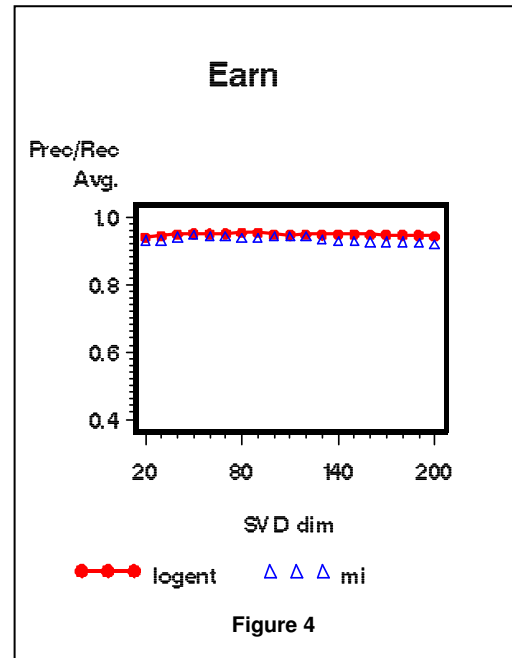
In Figure 5 we plot our results for the corn category on the same scale as that of the earn category. In this case, much more variation is evident in the precision and recall average, especially for the log-entropy weighting. With that weighting, the average increased until  $k$  reached 100 and then leveled off. The improvement in the average for small values of  $k$  was significant.

### TRUNCATION RESULTS

While holding the number of dimensions fixed to be 80, we considered how the truncation approach compared with the SVD technique.

Truncation is highly sensitive to which  $k$  terms are chosen. We initially considered a local-global weighting scheme along with mutual information in order to evaluate the truncation approach. But, because the local-global weighting scheme is independent of the categories, it is often impossible to get a collection of 80 terms that serve as strong predictors for each and every category. As a result, when we ran experiments using a local-global weighting scheme and truncation, our precision-recall averages were below the baseline and are not recorded in Table 4.

Since terms with a high mutual information weighting do not necessarily occur very many times in the collection as a whole, we found it necessary to first multiply the mutual information weight by the log of the frequency of the term. The highest 80 terms according to this product were kept. This guaranteed that we would keep at least a few terms from every document.



Our results for the truncation approach using mutual information came in lower than that of the SVD for seven of the ten categories and about 50% worse overall (cf. the micro-averaged case). They are included in Table 4. Note that before, the number of dimensions needed for an SVD to reach an asymptote with a given category ranged from 20 to about 100. In order to get the same predictive power with truncation, it may be that many more words need to be kept.

It is interesting to note that the three categories in which truncation performed better were grain, corn, and wheat, which are strongly correlated with each other, and likely contain similar vocabulary. None of the other categories appear to exhibit such

a correlation. It is certainly likely that an SVD would tend to put documents for any of those categories close together in the multi-dimensional space.

We might expect that such an effect could be ameliorated by using the mutual information weighting instead of a local-global weighting. Interestingly, we can see from Table 3 that the only times the precision-recall average for mutual information exceeded that from the local-global weighting were for the same three categories.

## NEURAL RESULTS

Table 4 also includes results that compare the neural network approach to that of MBR. On average, the neural network slightly outperformed MBR for both the SVD and the Truncation reductions. The differences, however, appear to be category dependent.

## CONCLUSION

While some general rules of thumb are available for text mining parameters, strict guidelines are not. This statement is substantiated by much of what we have reported here. What might be effective for one collection or category may not be effective for others.

Our study has validated the usefulness of the SVD in certain situations. Two important observations can be made. The first is that it does particularly well with small dimensions, under 200 or so. Because of the "curse of dimensionality", this can be important when it comes to training a predictive model. The alternative, truncation, may need many more dimensions in order to produce the same predictive power.

Secondly, the technique does not seem to require a category-dependent weighting scheme in order to generate reasonable categorization averages, as Figure 5 and Table 3 revealed. Relative to local-global weighting, the SVD approach seems to reach an asymptote with fewer dimensions when using the mutual information weighting, which makes sense, but the asymptote itself is about the same with both weighting approaches. One possible explanation for this second point is the tendency of mutual information weights to be biased toward terms that occur infrequently.

Although we did not explore the possibility in this paper, a category-specific weighting scheme cannot be used when clustering documents. The strengths of the SVD may particularly stand out in this situation, since truncation with a small number of terms is difficult to apply in that situation.

Our investigations revealed a significant difference between the precision and recall averages of the various categories. For instance, while we could get a nearly 98% average with the earn category, for other categories, like interest, our best was much lower.

This difficulty was also evident in other research [4]. Since many of the documents are assigned to more than one category, weighting the discriminating words is particularly challenging. One approach that we have not yet fully explored is to first make a decision about whether a given document belongs within a certain hierarchy. Once this is determined, a decision could be made as to which particular category the document belongs.

## APPENDIX

### WEIGHTINGS

#### Local Weightings

Again let  $A$  be the term-frequency matrix with entries  $a_i$ . If no weights are applied then  $a_{ij}=f_{ij}$  holds, where  $f_{ij}$  is simply the frequency of the term  $i$  in document  $j$ . The term importance weightings are then applied to  $f_{ij}$  and are of either the local or global variety.

We considered two common local weighting schemes, binary and log (all logs are taken base 2). The binary weighting has the usual definition

$$a_{ij} = \text{bin}(f_{ij}) = \begin{cases} 1, & f_{ij} > 0 \\ 0, & f_{ij} = 0 \end{cases},$$

and the log weighting is given by,

$$a_{ij} = \log(f_{ij} + 1).$$

#### Global Weightings

Recall that  $n$  is the number of terms in the matrix  $A$ . Let

$$p_{ij} = \frac{f_{ij}}{\sum_j f_{ij}}$$

be the probability that term  $i$  is found in document  $j$  and let

$$d_i = \sum_j \text{bin}(f_{ij})$$

be the number of documents containing term  $i$ . Then, the following global weights can be computed:

1. Entropy (the equation represents one minus the scaled entropy)

$$g_i = 1 + \sum_j \frac{p_{ij} \log(p_{ij})}{\log(n)}$$

2. Inverse Document Frequency (IDF)

$$g_i = \log\left(\frac{n}{d_i}\right) + 1$$

3. Global Frequency times Inverse Document Frequency (GFIDF)

$$g_i = \frac{\sum_j f_{ij}}{d_i}$$

		Category c	
		1	0
Term $x_i$	1	A	B
	0	C	D

Table 5

		Actual	
		1	0
Predicted	1	A	B
	0	C	D

Table 6

#### 4. NORMAL

$$g_i = \frac{1}{\sum_j f_{ij}^2}$$

A global weight  $g_i$  provides an individual weight for term  $i$ . The global weight is applied to the matrix  $A$  by calculating  $a_{ij}g_i$  for all  $i$ .

#### MUTUAL INFORMATION

Let  $x_i$  represent the binary random variable for whether term  $t_i$  occurs and let  $c$  be the binary random variable representing whether a particular category occurs. Consider the two-way contingency table for  $x_i$  and  $c$  given in Table 5.

$A$  represents the number of times  $x_i$  and  $c$  co-occur,  $B$  is the number of times that  $x_i$  occurs without  $c$ ,  $C$  is the number of times  $c$  occurs without  $x_i$ , and  $D$  represents the number of times that both  $x_i$  and  $c$  do not occur. As before,  $m$  is the number of documents in the collection so that  $n = A+B+C+D$ . Define  $P(x_i)$  to be

$$P(x_i = 1) = \frac{A+B}{m} \text{ and } P(x_i = 0) = \frac{C+D}{m};$$

$P(c)$  to be

$$P(c = 1) = \frac{A+C}{m} \text{ and } P(c = 0) = \frac{B+D}{m};$$

and  $P(x_i, c)$  to be

$$P(x_i = 1, c = 1) = \frac{A}{m},$$

$$P(x_i = 1, c = 0) = \frac{B}{m},$$

$$P(x_i = 0, c = 1) = \frac{C}{m},$$

and.

$$P(x_i = 0, c = 0) = \frac{D}{m}$$

The mutual information  $MI(t_i, c)$  between a term  $t_i$  and a category  $c$  is a variation of the entropy calculation given above. It is defined to be

$$MI(x_i, c) = \sum_{x_i, c} p(x_i, c) \log\left(\frac{p(x_i, c)}{p(x_i)P(c)}\right)$$

Mutual information gives us an idea of the strength of dependence between  $x_i$  and  $c$ . If  $t_i$  and  $c$  have a large mutual information, the term will be useful in distinguishing when the category  $c$  occurs.

#### SINGULAR VALUE DECOMPOSITION

The SVD is a form of an orthogonal matrix factorization [5]. Without loss of generality, let  $m$  be greater than or equal to  $n$ . A  $m$  by  $n$  matrix  $A$ , can be decomposed into three matrices

$$A = U\Sigma V^t$$

where

$$U^t U = V^t V = I$$

and

$$\Sigma = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_n).$$

The columns of  $U$  and  $V$  are referred to as the left and right singular vectors, respectively, and the singular values of  $A$  are defined by the diagonal entries of  $\Sigma$ . If the rank of  $A$  is  $r$  and  $r < n$  then  $\sigma_{r+1}, \sigma_{r+2}, \dots, \sigma_n = 0$ .

For our purposes, the key benefit from the SVD is that

$$A_k = \sum u_i \cdot \sigma_i \cdot v_i^t,$$

$k < n$ , provides the least squares best fit to  $A$ . For details see Golub and van Loan [5]. The process of acquiring  $A_k$  is known as the forming the truncated SVD. The higher the value of  $k$ , the better the approximation to  $A$ .

Deerwester *et al.* [3] have utilized the SVD in relation to the term-document frequency matrix and developed a method for projecting documents into the reduced dimension space that is defined by  $A_k$ . As a result, documents can be represented in  $k$  dimensions rather than  $n$  dimension and the semantic content of the text collection is contained in the column space of  $A$ . The approach is often used in information retrieval and is referred to as *latent semantic indexing*. The papers by Berry *et al* [1,2] provide good references.

#### PRECISION AND RECALL

Precision and recall were originally used to measure the ability of search engines to return documents that are relevant to a query and to avoid returning documents that are not relevant to a query. The two measures have also become standard in determining the effectiveness of a binary text classifier. In this context, a "relevant" document is one that actually belongs to the category. A classifier has high precision if it assigns a low percentage of "non-relevant" documents to the category. On the other hand, recall indicates how well the classifier was able to find "relevant" documents and assign them to the category.

The recall and precision can be calculated from the two-way contingency table found in Table 6.

So if  $A$  is the number of documents predicted to be in the category that actually belong to the category,  $A+C$  is the number of documents that actually belong to the category, and  $A+B$  is the number of documents predicted to be in the category, then

$$\text{Precision} = A/(A+B) \text{ and} \\ \text{Recall} = A/(A+C).$$

Obtaining both high precision and high recall are generally mutually conflicting goals. If one wants a classifier to obtain a high precision then only assign documents to the category that are definitely in the category. Of course, this would be done at the expense of missing some documents that might also belong to the category and, hence, lowering the recall. The average of precision and recall is often used to combine the two measures into a single result.

## REFERENCES

1. M. W. Berry, S. Dumais, G. O'Brien. *Using linear algebra for intelligent information retrieval*, SIAM Review, 37, pp. 573-595, 1995.
2. M. W. Berry, Z. Drmac, E. R. Jessup, *Matrices, vector spaces, and information retrieval*, SIAM Review, 41(1999), pp. 335-362.
3. S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman, *Indexing by latent semantic analysis*, Journal of the American Society for Information Science, 41, pp. 391-407, 1990.
4. S. Dumais, J Platt, D. Heckerman and M. Sahami, *Inductive Learning Algorithms and Representations for Text Categorization, Proceedings of the 1998 ACM 7<sup>th</sup> international conference on information and knowledge management*, 148-155, 1998.
5. G. Golub and C. F. van Loan, *Matrix Computations*. The Johns Hopkins University Press, Baltimore, third edition, 1996.
6. G. Salton and M. McGill, *Introduction to Modern Information Retrieval*, McGraw-Hill, New York, 1983.
7. Y. Yang and J. Pedersen, *A comparative study on feature selection in text categorization. In Machine Learning: Proceedings of the Fourteenth International Conference (ICML 97)*, 412-420, 1997.

## ACKNOWLEDGEMENTS

We would like to extend our thanks to Ross Bettinger, who read our draft and provided valuable feedback.

## CONTACT INFORMATION

Your comments and questions are valued and encouraged.

Contact the author at:

James A. Cox, Ph.D.  
SAS Institute Inc.  
SAS Campus Dr  
Cary NC 27513  
Work Phone: 919-531-7963  
Fax: 919-677-444  
Email: james.cox@sas.com  
Web: [www.sas.com](http://www.sas.com)