

A Simple Feature Selection Method for Text Classification

Pascal Soucy and Guy W. Mineau

Dept. of Computer Science

Université Laval, Québec, Canada, G1K 7P4

{Pascal.Soucy, Guy.Mineau}@ift.ulaval.ca

Abstract

In text classification most techniques use *bag-of-words* to represent documents. The main problem is to identify what words are best suited to classify the documents in such a way as to discriminate between them. Feature selection techniques are then needed to identify these words. The feature selection method presented in this paper is rather simple and computationally efficient. It combines a well known feature selection criterion, the information gain, and a new algorithm that selects and adds a feature to a bag-of-words if it does not occur too often with the features already in a small set composed of the best features selected so far for their high information gain. In brief, it tries to avoid considering features whose discrimination capability is sufficiently covered by already selected features, reducing in size the set of the features used to characterize the document set. This paper presents this feature selection method and its results, and how we have predetermined some of its parameters through experimentation.

1 Introduction

In automatic text classification most techniques use *bag-of-words* to represent documents. The main problem is then to identify what words are best suited to classify the documents in predefined classes. Feature selection techniques are then needed to identify these words.

Many feature selection techniques use some metric to determine the relevancy of a term with regard to a classification criterion. One such technique uses the information gain metric assessed over the set of all words encountered in all texts, and then chooses the best k words according to that metric [Lewis and Ringuette, 1994]. Because of computational efficiency issues, it may not be possible to use the learning algorithm itself to determine which k is best suited to learn the descriptions of the classes of documents so that the accuracy rate of the algorithm is maximized. In that case, there is little guidance to determine what value k should be. Some naive techniques just predetermine a k using the learning algorithms over a small set of data sets, and

then uses that value of k directly on other data sets even though these data sets may be quite different from those used to learn the value of k . To our opinion, this k becomes rather arbitrary unless data sets can be characterized in such a way to be easily associated with previously tested data sets.

The feature selection method presented in this paper is rather simple and computationally efficient. Efficiency is a major issue in text classification of large corpora (like found on the WWW) since many thousand features are usually involved. Our method is a filter approach [Hall and Smith, 1999], as the learning algorithm is not involved in the selection process of features and is self-adaptable to the nature of the corpus. It combines a feature selection method known to be among the best for text classification [Yang and Pedersen, 1997], that is, thresholding on information gain, and a new algorithm that selects and add features to a bag-of-words according to their average cooccurrence in the training set with the features already in a small set composed of the best features selected so far for their high information gain, called the pool. This paper presents this feature selection method and how we have predetermined some of its parameters, called thresholds, through experimentation. It also shows how these thresholds can be used as such in other domains and still render a good performance level in terms of the relevancy of the feature set extracted for subsequent classification purposes.

2 Experimental Testbeds

2.1 Learning Algorithms

In order to assess the potential of our feature selection method, we used it with two different learning algorithms that are widely used in text classification and retrieval: k -nearest neighbors (KNN) algorithms (with weighted and non weighted features according to different weighting functions) and Rainbow, a naive Bayes classification algorithm for text classification [McCallum, 1996; Fürnkranz *et al.*, 1998]. However, because of lack of space, this paper only presents the results obtained with the non weighted features KNN algorithm and Rainbow, since all weighted KNN algorithms performed more poorly than their non-

Task name	Dataset	Classes	Size ¹	Train/Test Balance
WebKBCourse	WebKB ²	Course/noncourse	400	80/320
ReutersCornWeat	Reuters-21578 ³	Corn/wheat	400	40/360
Ling-Spam	Ling-Spam ⁴	Spam/non-spam	120	40/80
WWWPrisoner	WWW ⁵	Relevant/irrelevant	60	40/20
WWWBeethoven	WWW ⁶	Relevant/irrelevant	60	40/20
UsenetNews1	Usenet ⁷	Soc.history/ Soc.religion.christians	67	40/27

Table 1 : The data sets used in our experiments. Each data set is a binary classification task between two classes. Size is the total number of documents in the data set. Train/Test Balance is the number of documents in the training and testing sets respectively. Note that the training sets have been voluntarily kept small for we believe that for most applications, few examples are available or if larger sets are available, manual labeling of the documents becomes prohibitive.

weighted counter-part. Appendice A briefly describes the weighting functions tested to try and improve the non-weighted version of our KNN algorithm, without success.

2.2 Data Sets

For our experiments, we used data sets previously studied in other work, and our own data sets. The data sets used in our experiments are described in table 1. These data sets may be categorized in two types: those including documents written with a concise vocabulary and those more freely written. The former set includes data sets such as Reuters (brief news) and WebKB (Web pages mainly retrieved from computer science university departments) while the latter set includes noisy data sets such as Ling-Spam (consisting of emails belonging to a mailing-list, including junk emails), Prisoner (the TV show) and Beethoven (the composer) (Web pages retrieved with Altavista using a keyword search), and News (unmoderated Usenet newsgroups on history and christianism). The difficulty associated with a classification task depends highly on the type of vocabulary used [Scott and Matwin, 1999]. Words were stemmed using the Porter algorithm for our KNN algorithms, and using the –use-stemming option with Rainbow.

¹ The size may be different from the original data set to ensure computationally manageable experiments. In that case, subsets of the original data sets have been selected randomly. For WWW pages research, the size is small due to the fact that there are very few relevant documents.

² Described in [8] and is available at www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/.

³ Well known data sets available at <http://www.research.att.com/~lewis/reuters21578.html>.

⁴ The Ling-Spam anti-spam filtering corpus, available at: http://www.iit.demokritos.gr/~ionandr/lingspam_public.tar.gz.

⁵ WWW pages manually classified according to their relevance to a subject, the TV show *The Prisoner*.

⁶ WWW pages manually classified according to their relevance to a subject, Beethoven biographies.

⁷ Usenet articles collected from soc.history and soc.religion.christians newsgroups. Only the first message of a thread of discussion has been retrieved. See [Scott and Matwin, 1999] for explanations about this choice.

3 Feature selection

Information Gain (IG) in text classification measures the reduction in entropy (heterogeneity of a set) gained using a term presence or absence to determine class membership. It is often used in text classification in the bag-of-words approach [Joachims, 1998; Nigam *et al.*, 1999; Sahami *et al.*, 1998]. This measure is the same used by ID3, a decision tree learning algorithm, to select the best attribute at each step of the tree construction [Quinlan, 1986].

3.1 The k best features using the Information Gain

Under this approach, a text classification task is considered using the first k words that rank highest on that metric. That number k , however, depends highly on the kind of corpus and task involved. Under a Bayesian classification approach, many noticed that a naive Bayesian classifier works best with many features, but better if not every available feature is used. From various experiments it was noticed that the k which seemed to be an approximation of the average number of features leading to reasonable results with that algorithm varies from 2000 features [Craven *et al.*, 1998] to 500 features [Sahami *et al.*, 1998] depending on the nature of the corpus.

These findings show a great variation in the size of the required vocabulary. If we consider that Bayesian classifiers suffer when irrelevant features are kept as selected features, it is difficult to justify why a predefined number of features would allow to optimize the accuracy rate of the classifier. Actually, in such data sets as Reuters, there are some words that are so characteristic of some class that fewer features need to be selected than with a data set involving newsgroups or email messages. It seems obvious that the number of required features highly depends on the nature of the corpus and of the learning algorithm. Therefore, how could we characterize the nature of the corpus in terms of the available features so that guidelines on how to select a proper k are easily inferred? This paper proposes one such way to determine k according to the nature of the terms composing the vocabulary with regard to their discrimination power relative to one another.

3.2 Using Information Gain with a threshold

A simple feature selection method could look at the IG of each feature (taken individually) with regard to the classification task. For instance, there are methods that select any feature with an IG greater than some threshold [Yang and Pedersen, 1997]. This is fully relevant to Bayesian classifiers. It is easy to see that features with low IG are useful to characterize only a handful of documents. Therefore, their probability distribution over the whole set of documents can not be estimated correctly. Bayesian classifiers rely on these probability estimates.

Similarly, infrequently used dimensions (attributes) in a KNN approach introduce noise more than anything else. As a matter of fact, KNN seems to produce better results when fewer features are involved, using the IG to keep the best features [Joachims, 1998]. KNN bias depends mostly on the kind of similarity function it uses, but it always uses the vector model, a weakness of this model being the fact that features are considered to be independent of each other. The function that seemed most successful (see Appendix A, a non weighted cosine similarity function), assigns to each feature the same weight. Given that, we should only consider features that are promising in terms of class characterization, as the presence of many irrelevant features diminish the influence of very good (relevant) features. Choosing a fixed number of features may lead to consider irrelevant ones (as with the Reuters data sets) if k is too high, or remove some good features (as with junk email classification) if k is too low.

However, even if we obtained good results on average using the KNN and Bayes classifiers, by looking at each data set individually we noticed that it would be possible to restrict the set of selected features even more. For instance, with the Reuters data set, we saw that in most binary classification tasks (for example, classifying documents according to their relevance to “corn” or “wheat”), only a few words (often under 10) were needed to obtain the best accuracy score. We observed that high IG features could represent classification alternatives (in which case one is useless), or could be complementary. Therefore, we could assess the cooccurrence of highly ranked features and eliminate those that represent alternatives of already selected high IG features.

We conducted extensive experiments where we had the number of features ranging from 1 to 5000 and the IG threshold ranging from 0 to 1. From all these experiments, we found that the best threshold to classify the data sets that we had was 0.1 in average. Therefore, we propose to first identify the terms whose IG is higher or equal to 0.1, and then, select only those features that are not covered by the strong features already in the set (in terms of their discrimination capability), as explained below. In what follows, we prove that this further refinement of the term set improves both the accuracy of the classifier and the interpretability of

the characterization so produced since the number of selected features is sensibly reduced.

3.3 Our Feature Selection Algorithm

The algorithm of Figure 1 proposes a refinement of the set of selected features (already offering an IG higher than some threshold) by removing those features that cooccur with very high IG features. Cooccurrence is a kind of measure of dependency between features, which is a weakness of most statistical models used today to characterize a set of features. Because of computational issues, dependencies between terms are usually ignored. We propose to identify a small subset of terms that we know are relevant for the classification task at hand (according to their individual IG), and to compute dependencies between any new term and these preselected terms. The cooccurrence metric that we consider

A Simple Feature Selection Algorithm: $\mu\text{Cooccurrence}$

```
Let  $V$  = set containing every word found in the corpus,  
sorted by Information Gain (in decreasing order)  
Let  $Pool = \{\}$  (the pool)  
Let  $D$  = set of documents  
Let  $igain\_thres = 0.1$   
Let  $cooc\_thres = 0.45$   
Let  $nbkeep = 5$ 
```

```
 $Pool \leftarrow nbkeep$  first features from  $V$ 
```

```
FOR each word  $w$  IN  $V - Pool$   
  IF  $IG(w,D) < igain\_thres$  THEN  
     $V = V - \{w\}$   
  ELSIF  $\mu\text{Cooc}(w,Pool,D) > cooc\_thres$   
    THEN  $V = V - \{w\}$   
  ENDIF  
ENDFOR
```

```
 $\mu\text{Cooc}(w, Pool,D)$  :  
  Let  $Sum = 0$   
  FOR each  $p_i$  in  $Pool$   
     $Sum \leftarrow Sum + (P(w | p_i) \text{ in } D)$   
  RETURN  $(Sum / nbkeep)$ 
```

Figure 1 : A Simple Feature Selection Algorithm

in the algorithm of Figure 1, written $\mu\text{Cooc}(w,Pool,D)$, is based on a partial cooccurrence probability of w with each word in $Pool$, considering the whole training set. It is computed as the mean value of $P(w|p_i)$ for all p_i in $Pool$, where $P(w|p_i)$ is the conditional probability to find w in any document if it already contains p_i .

Let us take an example and assume that the word “corn” has an IG value of 0.95, meaning that it almost perfectly divides the training set. Then let us suppose that $P(\text{starch}|\text{corn}) = 1$ and $IG(\text{starch}) < IG(\text{corn})$. Since “starch” always occurs when “corn” does and since its IG is lower

than that of “corn”, “starch” is unlikely to bring new information (in terms of discrimination power) to the feature set, and could therefore be removed from it.

Needless to say that the reality is not that perfect. Removing a feature only when it is *totally* covered by some other feature (in terms of its discrimination capability) would remove only a few features, if any, which goes against our aggressive selection objective. Some approximation of coverage between terms is thus required to effectively reduce the vocabulary to those features that are meaningful in terms of their overall discrimination power. Our approach is to consider a mean of partial cooccurrence between any new term and the terms in P , the set of the *nbkeep* highest ranked features according to their IG, called the pool. Of course, *nbkeep* and *cooc_thres* (see the algorithm of Figure 1) have been empirically determined using our own data sets: Prisoner, Beethoven and News, and tests have been conducted over previously studied data sets: *Reuters*, *Ling-Spam* and *WebKB*, in order to prevent over fitting the parameters on these data sets. Due to a lack of space, we won’t show here the results of this parameterization, but we concluded from these tests that the values used in the algorithm of Figure 1 hold for all our data sets, and seem to hold for other data sets that we are currently experiencing with.

4 Results

Table 2 shows the results obtained with the different feature selection methods that we tested over each corpus, under our two learning algorithms: a non weighted features KNN algorithm (first three columns) and Rainbow, a well-used Bayesian classifier (last five columns). For each method, we show k , the number of features selected and used thereafter by the learning algorithm, and the accuracy score (%) obtained subsequently.

The first column shows the k features selected when we set a threshold of 0.1 on the Information Gain score (IG) of each feature. Again, this threshold was assessed empirically over all possible values of IG ranging from 0 to 1. From the resulting set, our μ Cooccurrence refinement algorithm was applied to produce a smaller set of features, and the results are shown in the second column. One can see that the number of features is almost always reduced, and if so, significantly and without impact on the accuracy rate, meaning that we could eliminate useless features in terms of the prediction capability of the learning algorithms used. This helps produce a characterization of the learned concepts with a smaller vocabulary, and therefore facilitates the interpretation of the results and reduces the computation cost to classify new documents.

Then we wondered if this minimal number of features could have been reached through a fine tuning of the IG score that we used. We found that an IG of 0.13 would produce the same number of selected features (taken over all corpora), but that the accuracy rate would then go down, as shown in the third column. Obviously, the selected features

in this case are not the same as those of the previous experiment. Furthermore, we see that the small fluctuation in the IG rate (0.03) has great impact on the individual corpora in terms of the number of selected features. Considering our detailed search for a good IG rate, it is not surprising to see such a fluctuation and its negative impact on the accuracy rate of the classifiers. This tends to show that 0.1 is a better threshold than 0.13, as validated in a previous experiment, and that our feature selection method is therefore useful to reduce that set by removing useless (or not so useful) features. It also suggests, at least on this type of corpora, that our method is sound and useful to determine a relevant k which, without such a method, would be impossible to determine.

For the Bayesian classifier, we observe the same phenomenon. The fourth column gives the number of selected features for $IG = 0.1$ and the accuracy rate obtained after training Rainbow with these features. For comparison purposes, we also trained Rainbow with 2000 features (fifth column) as is suggested in [Craven *et al.*, 1998]. By comparing the fourth and fifth columns we can see that the 2000 feature sets performs better than what we obtain with $IG = 0.1$ except in cases where there are a few words with extreme discrimination capabilities like with the Reuters and Beethoven corpora. Despite this small loss in accuracy, the feature set with $IG = 0.1$ is nevertheless significantly smaller than 2000, which would undoubtedly help the interpretation of the learned concepts.

In the next experiment (sixth column), we applied our μ Cooccurrence refinement algorithm to reduce even further the size of the feature set. Again we obtain a reduced set without any impact on the accuracy rate, facilitating even more the subsequent interpretation of the learned concepts.

However, by setting $IG = 0.13$, we obtain the same number of features (taken over all feature sets) and a slightly smaller accuracy rate (see the seventh column).

Finally, as expected, an unbounded number of features has a negative impact on the accuracy rate of a Bayesian classifier, as shown in the last column. Though not illustrated in Table 2, this phenomenon also holds for KNN classifiers when the number of attributes is extremely high and the number of relevant attributes is much lower than the number of attributes, which is the case in text classification. Therefore, there is a need to determine a proper k for training the classifier with the relevant attributes.

5 Complexity Issues

Let N be the number of documents, let V be the size of the vocabulary, and let P be the size of the pool. By choosing to compute cooccurrence only between the elements of the vocabulary and the pool, we avoid having to compute it for all $O(V^2)$ possibilities (when pairs of features are considered), producing a $O(VPN)$ complexity factor rather than $O(V^2N)$ for assessing the cooccurrence of terms over all N documents. Since computing the IG for all features in V is

Data Set	IG = 0.1 KNN		μ Cooc KNN		IG = 0.13 KNN		IG = 0.1 Bayes		k=2000 Bayes	μ Cooc Bayes		IG = 0.13 Bayes		All features Bayes	
	k	%	k	%	k	%	k	%	%	k	%	k	%	k	%
WebKBCourse	37	97.5	35	96.9	24	95.3	37	95	95.6	35	95	24	94.1	12150	94.2
ReutersCornWheat	8	98.3	8	98.3	6	98.3	8	97.4	89.2	8	97.4	6	97.4	3393	81.1
Ling-Spam	86	95	77	95	47	91.3	86	86.3	90	77	86.3	47	87.5	4484	86.3
WWWPrisoner	53	85	9	90	50	90	53	70	80	9	70	50	70	5076	70
WWWBeethoven	121	85	8	85	106	80	121	90	85	8	90	106	85.7	3327	90
UsenetNews1	184	85.2	130	85.2	35	85.2	184	92.6	96.3	130	92.7	35	92.6	8522	92.6
Micro-average	95.6		95.5		93.7		92.7		93.1	92.7		92.2		90.3	
Number of features	489		267		268		489		12000	267		268		36952	

Table 2 : Number of features (k) and accuracy score (%) for each data set under different feature selection and learning methods. The total number of features is the summation of the k column; the micro-average calculates the average of correctly classified instances over all data sets and is therefore not an average of the individual accuracy scores.

$O(VN)$, and since ordering them by their IG is $O(V \log V)$, for large corpora (where $\log_2 V \ll N$), the dominating complexity factor of our algorithm is the assessment of the cooccurrence of terms. Consequently, our method works under a complexity factor of $O(VPN)$. Provided that we can determine P independently of V and N , as was done in our experiments, the resulting complexity figure for our method then becomes $O(VN)$, which is the same complexity figure for initially assessing the IG of each feature in the vocabulary. Consequently, our μ Cooccurrence feature selection algorithm has the same asymptotic complexity as a feature selection method that solely computes the IG of each feature in its feature set, with a hidden constant that has now doubled however. Nevertheless, the method is computationally efficient.

6 Conclusion and Future Work

We have presented in this paper a very simple feature selection method for text classification that refines a set of features previously selected according to an IG criterion. This refinement is based on the cooccurrence of new features with a predetermined subset of highly ranked features, called the pool, hoping to avoid to include features whose discrimination capability with regard to the task is (partially) covered by the features of the pool. We seek to improve the classification speed of the learner without reducing accuracy and reduce the set of attributes that will be used later on to support the interpretation of the learned concepts. This paper shows that we reached this objective for the corpora that we studied. The method that we propose in this paper certainly proved to be promising for certain types of text classification tasks, as demonstrated by our experiments. At first glance, we are tempted to conclude that μ Cooccurrence has the capability to remove the features that brings no further information to those already selected. Of course further experiments will be conducted to assess the validity of that claim under various conditions.

Additional investigation will also be conducted in order to understand why μ Cooccurrence seems to have more impact on (non weighted feature) KNN algorithms than on Bayesian classifiers. Though we feel that the probabilities that ponder the involvement of attributes provides some explanation, we still can not understand why weighted features KNN approaches failed to surpass the non weighted KNN algorithm that we used, despite extensive experimentation using a wide range of distance functions. [Domingos and Pazzani, 1997] demonstrated why naive Bayes learners perform well in spite of the (so-called) word independence assumption. Therefore, we will focus our investigation towards the study of word dependence and its impact on the KNN model.

Further effort is also needed to extend the μ Cooccurrence algorithm so that it can handle multiple concept learning. Our experiments have been conducted only over binary classification tasks.

Another field of investigation will be to use multistrategic approaches to feature selection. For instance, we could assess the information gain of *sequences* of features. We could base our choice of feature sequences on the first few levels of decision trees built from our initial feature set, and increment the vocabulary accordingly.

Finally, though we set the IG and cooccurrence thresholds at 0.1 and 0.45 through extensive experimentation, more study on the impact of these parameters on the classification task with regard to different types of corpora is required.

Appendix A

The learning bias of the KNN model is mainly related to its distance (similarity) function and to its neighbor weighting function to compute that distance. Distance weighted KNN adjusts the importance of each neighbor by weighting their contribution to the classification of a new instance. In this section, we consider the weights associated with the features themselves rather than with individual instances.

This section presents the weight and distance functions that have been studied in our search to determine which KNN algorithm would be used in the assessment of our feature selection method. Surprisingly, the simplest of these functions worked better than the others.

Let us define an instance x as

$$\langle a_1(x), a_2(x), a_3(x), \dots, a_n(x) \rangle$$

where $a_j(x)$ is the weight of the j th attribute (feature) on x .

Using non-weighted features

If we consider that the weight of each feature is the same, then we have:

$$\begin{aligned} a_j(x) &= 1 && \text{If the word is in document } x \\ a_j(x) &= 0 && \text{otherwise} \end{aligned}$$

Using weighted features

The goal reached using weighted features is to stretch the neighborhood space so that irrelevant features are given less importance. There are several ways to do this feature weighting. Here are a few examples that were tested:

- the weight of a feature for a document is its IG if the word is present in the document, 0 otherwise
- the weight of a feature is its frequency in the document
- the weight of a feature is its TFIDF (*Term frequency/Inverse Document Frequency*)

We also tried many variants of these methods, for example, squaring the IG so that features with high IG would get yet higher weights. Under these various weight assignment policies, we experimented with two distance functions:

Euclidian distance function

$$d(x1, x2) = \sqrt{\sum_{j=1}^n (a_j(x1) - a_j(x2))^2}$$

Cosine similarity function [Salton89]

$$\text{CosSim}(x1, x2) = \frac{\sum_{j=1}^n a_j(x1) \cdot a_j(x2)}{\sqrt{\sum_{j=1}^n a_j(x1)^2 \cdot \sum_{j=1}^n a_j(x2)^2}}$$

References

[Craven *et al.*, 1998] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to extract symbolic knowledge from the World Wide Web. *Technical report*. Departement of Computer Science. Carnegie Mellon Univ.. 1998.

[Domingos and Pazzani, 1997] P. Domingos and M. Pazzani. Beyond independence: Conditions of the optimality of

the simple bayesian classifier. *Machine Learning*. (29) pp 103-130. 1997.

[Fürnkranz *et al.*, 1998] J. Fürnkranz, T. Mitchell and E. Riloff. A Case Study in Using Linguistic Phrases for Text Categorization on the WWW. *Working Notes of the 1998 AAAI/ICML Workshop on Learning for Text Categorization*.

[Hall and Smith, 1999] M. A. Hall and L. A. Smith. Feature selection for machine learning: comparing a correlation-based filter approach to the wrapper. *Proceedings of the Florida Artificial Intelligence Symposium (FLAIRS-99)*. 1999.

[Joachims, 1998] T. Joachims. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. *Proceedings of the European Conference on Machine Learning*. Springer. 1998.

[Lewis and Ringuette, 1994] D. D. Lewis and M. Ringuette. A comparison of two learning algorithms for text categorization. *Third Annual Symposium on Document Analysis and Information Retrieval*. pp 81-93. Las Vegas. 1994.

[McCallum, 1996] A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. <http://www.cs.cmu.edu/~mccallum/bow>. 1996.

[Nigam *et al.*, 1999] K. Nigam, J. Lafferty and A. McCallum. Using Maximum Entropy for Text Classification. *IJCAI-99 Workshop on Machine Learning for Information Filtering*. 1999

[Quinlan, 1986] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1) pp 81-106. 1986.

[Sahami *et al.*, 1998] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A Bayesian approach to filtering junk e-mail., *AAAI/ICML Workshop on Learning for Text Categorization*. Wisconsin. 1998.

[Salton, 1989] G. Salton. *Automatic text processing: the transformation, analysis, and retrieval of Information by Computer*. Reading, MA: Addison-Wesley. 1989.

[Scott and Matwin, 1999] S. Scott and S. Matwin. Feature engineering for text classification. *Proceedings of ICML-99, 16th International Conference on Machine Learning*. pp. 379-388. Morgan Kaufmann Publishers. 1999.

[Yang and Pedersen, 1997] Y. Yang and J. P. Pedersen. A Comparative Study on Feature Selection in Text Categorization. *Proceedings of the Fourteenth International Conference on Machine Learning (ICML'97)*. 1997.