Raising the Baseline for High-Precision Text Classifiers

Aleksander Kołcz
Microsoft Live Labs
One Microsoft Way
Redmond, WA
USA
ark@microsoft.com

Wen-tau Yih
Microsoft Research
One Microsoft Way
Redmond, WA
USA
scottyih@microsoft.com

ABSTRACT

Many important application areas of text classifiers demand high precision and it is common to compare prospective solutions to the performance of Naive Bayes. This baseline is usually easy to improve upon, but in this work we demonstrate that appropriate document representation can make outperforming this classifier much more challenging. Most importantly, we provide a link between Naive Bayes and the logarithmic opinion pooling of the mixture-of-experts framework, which dictates a particular type of document length normalization. Motivated by document-specific feature selection we propose monotonic constraints on document term weighting, which is shown as an effective method of fine-tuning document representation. The discussion is supported by experiments using three large email corpora corresponding to the problem of spam detection, where high precision is of particular importance.

Keywords

high precision text classification, Naive Bayes, low false positive rates, email spam detection

Categories and Subject Descriptors

H.4.m [**Information Systems**]: Information Retrieval; D.2 [**Machine Learning**]: Text Categorization

1. INTRODUCTION

Practical classification problems in the text categorization (TC) domain often involve sharp constraints with respect to the precision, false-positive or false-negative rates. While the overall accuracy is important, a spam filtering solution, for example, may be unacceptable if it destroys or misdelivers legitimate email as spam even at a small rate. In a cost-sensitive classification framework, one can often see significant asymmetry in the way different misclassification mistakes are weighted. In this context, classification systems are beneficial only as long as the probability of certain types of errors is sufficiently low. Although a large amount of research has been devoted to improving the quality of text classification techniques, the focus has been primarily on the overall accuracy or global quality metrics such as the error rate, F-measure, precision-recall break-even or area under the ROC curve. Methods that improve the global metrics may also improve classification performance in the region of high specificity, but they are rarely investigated in this context.

In this work we focus on the aspects of document representation, and in particular on the impact of document sparsity, term weighting and length normalization in problems demanding high specificity. We concentrate on the important case of Naive Bayes [23], which is a highly scalable learner of great practical importance, and for which a number of recent improvements have been proposed, making it quite competitive with more complex techniques such as SVMs [29, 8].

In the context of the Naive Bayes (NB) classifier, it has been suggested that post-induction document-specific feature selection tends to outperform traditional document-independent approaches in applications with low tolerance for false positive errors [20]. We show that these benefits can be improved by making document-specific feature selection part of the induction process. We also extend the concept of local sparsity control to the area of term weighting. We provide evidence that the soft approach (i.e., downweighting of less relevant features rather than their complete elimination) to the reduction of active document terms can provide substantial improvements to classification performance and in our experiments it tends to outperform hard feature selection. When the soft and hard feature selection approaches are combined, feature elimination leads only to marginal improvements.

Document length normalization provides a mechanism for controlling the influence of any particular term on a document by document basis. Although it has been used widely with other text classifiers, its use with Naive Bayes is very recent [29] and not particularly well understood. In this work, we show that certain types of length normalization cast Naive Bayes into the mixture-of-experts framework, which provides a solid basis for this type of transformation and also explains its effectiveness for this classifier. Based on our analysis it becomes apparent that for Naive Bayes, $L1$ normalization may be more appropriate than the traditional $L2$ normalization, which is also supported by the experimental findings.

In our experiments with several email collections we demonstrate that with appropriate document representation, Naive

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
KDD'07, August 12–15, 2007, San Jose, California, USA.
Copyright 2007 ACM 978-1-59593-609-7/07/0008 ...$5.00.

Bayes can compete and even outperform state-of-the-art learners such as Logistic Regression and Support Vector Machines. This is particularly true for datasets with some degree of class noise, which is quite typical in practical applications of text mining. These improvement in performance of NB do not take away its attractiveness in terms of speed of learning and ease of implementation.

The paper is organized as follows: In Section 2 we discuss why NB does not perform well in the text classification tasks that require high precision. We then investigate three directions of improving NB in this setting – document-specific feature selection at induction time (Section 3), $L1$ document length normalization linked to logarithmic opinion pooling (Section 4) and combined supervised and unsupervised feature weighting (Section 5). The experimental setup is presented in Section 6. As shown in Section 7, all these techniques improve the NB performance and, in addition, can be combined and yield even better results as demonstrated by our experimental results. Related work is reviewed in Section 8 and the paper is concluded in Section 9.

2. ANALYSIS OF NAIVE BAYES FOR HIGH-PRECISION PREDICTIONS

Consider a binary classification task defined over domain X, where based upon a training set

$$\{d_i, y_i\} : x_i \in X, \quad y_i \in \left\{C, \overline{C}\right\}, \quad i = 1..T$$

the objective is to find a mapping $\left\{f : X \rightarrow \left\{C, \overline{C}\right\}\right\}$ such that its expected loss is sufficiently low. The definition of loss can be application specific and often is taken to be the error rate. In many problems, the misclassification costs are asymmetric and in some cases the cost of one type of error can be high enough to demand very low, or even near zero, probability of occurrence. In some web search applications, for example, it is required that the top-N results returned of a user query have very high precision even if this significantly restricts the number of potentially relevant responses that can make it to top-N. In spam detection, users have low tolerance for false positive errors and accept email filtering solutions as long as the chance of losing some important email communications is negligibly low. If misclassification cost values and accurate estimates of posterior probabilities are available, optimum decisions can be made by setting the decision threshold in the probability space to minimize the expected misclassification cost [11]. Due to the practical problems in obtaining these, it is often convenient to work with the Neyman-Pearson criterion by setting the limit on the maximum acceptable false-positive rate or alternatively on the minimum acceptable precision.

Typically, a classifier returns a score proportional to its "confidence". In the case of NB, the score is computed as

$$score\,(d) = const + \sum_i f_i \cdot \log \frac{P\,(t_i|C)}{P\left(t_i|\overline{C}\right)} \qquad (1)$$

where the constant term captures the effect of class priors (which can be ignored if classification threshold is chosen based on a validation set). For the multinomial variant of NB [24], typically used in text applications, the summation in Eq. (1) is carried out over the terms present in document d (as opposed to all possible terms) and the value of f_i corresponds to the frequency of occurrence of term t_i in d. The occurrences of terms in d are assumed to be independent given the class label and the class conditional probabilities $P\,(t_j|C)$ are estimated as

$$P\,(t_j|C) = \frac{1 + \sum_{x_i \in C} f_{ij}}{V + \sum_j \sum_{x_i \in C} f_{ij}} \qquad (2)$$

where $f_{ij} \geq 0$ is the number of occurrences of term t_j in document d_i and V is the vocabulary size. In Eq. (2) the Laplace technique is applied to smooth the probability estimates. The multinomial model was extended in [29] and [8], whereby the values f_{ij} no longer have to correspond to in-document frequency but to a function thereof. In particular, in [29] it was suggested to map f_{ij} to a real-valued $tf \times idf$ weight and additionally normalize these features on a per-document basis so that the $L2$ norm of each feature vector is one.

During the tuning process of a classifier, a threshold is chosen such that decisions with scores exceeding the threshold are classified as "positive". In this context, the classifier's inability to perform well at a low enough false-positive rate can be seen as evidence of its overconfidence, whereby erroneous decisions are made with apparent high confidence. While this behavior can be observed in many learners, it is particularly common for Naive Bayes (NB), due to its assumption of conditional independence of features given the class label. Although NB can have a reasonably low error rate [7], in some cases feature inter-correlations get compounded resulting in overconfident predictions. As shown in [1, 20], this is particularly true for long documents, which is a reason why feature selection can have a strong positive effect for this type of classifier.

Since feature inter-correlations are at the heart of the overconfidence problem for NB, a number of modifications to the learner have been proposed aiming to detect and counter such effects [12, 36]. Unfortunately, these adjustments tend to increase the complexity of model induction to a large extent, which eliminates the key advantage of NB in practical applications, i.e., its scalability and ease of implementation. In some instances, however, the proposed improvements have been shown to significantly improve NB's performance with negligible increase in classifier complexity. We focus on the ones proposed in [20] and [29], which advocate the use of document-specific feature selection as well as $tf \times idf$ term weighting coupled with document length normalization, respectively.

3. DOCUMENT-SPECIFIC FEATURE SELECTION

Supported by psychological evidence [22], document classification can be done fairly accurately by looking at only a small portion of the text [31]. Indeed, previous work on document-specific feature selection (DSFS) [20], which uses only a small set of "important" words in a document, has shown to improve NB's performance significantly [16], especially in settings with highly skewed misclassification costs. However, the method described in [20] was applied post-induction and thus was not able to take full advantage of the DSFS process. While being much simpler operationally (the selection of the optimum feature count can be applied without the potentially expensive retraining of the classifier), the technique might be suboptimal for some learners since they do not get a chance to induce a model over the reduced document representation, although as shown in [20][16] it works

quite well for Naive Bayes.

We propose to naturally extend the DSFS process so that it affects classifier induction. We hypothesize that by doing so, the method might not only be more suitable for discriminative learners such as SVMs, but also more effective for Naive Bayes itself. The original and modified document-specific feature selection process are compared below.

Document-Specific Feature Selection

Post-hoc	Full Induction
1. Train a classifier	1. Train a classifier
2. Rank feature weights	2. Rank feature weights
3. Use top-N features per doc in evaluation	3. Retain top-N features per training document
	4. Retrain with the new representation
	5. Use top-N features per doc in evaluation

DSFS relies on the choice of a single cut-off parameter for all documents, regardless of their length and content. While this can be seen to regularize Naive Bayes, it may be suboptimal for many documents, for example those containing more numerous strongly relevant terms than suggested by the cut-off threshold. One possible way to address this problem is to consider *soft* document-specific term weighting instead of *hard* feature selection, which is decided by the term frequency and a predefined cut-off threshold. This issue will be examined in more detail in Section 5.

4. LOGARITHMIC OPINION POOL AND NAIVE BAYES

While "pure" versions of the Naive Bayes classifier may perform poorly when faced with large volumes of high dimensional data, many improvements and modifications have been suggested, which make Naive Bayes competitive with state-of-the-art discriminative learners. In one significant improvement of NB, Rennie *et al.* [29] proposed to use $tf \times idf$ feature weighting as well as $L2$ document length normalization to improve the performance of the classifier in text applications. In this section, we advocate instead using an $L1$ norm, which can be linked to the logarithmic opinion pooling framework of combining experts judgements.

The document representation based on using an $L2$ norm over the $tf \times idf$ features has been widely used in Information Retrieval (hence we will denote a NB using it as NB-IR), as well as for text classifiers such as Rocchio or linear SVMs [10]. While it appears to help in improving Naive Bayes performance, the justification for its use is rather weak. Using $L2$ norm makes sense in IR where document similarity is often expressed in terms of the cosine of the angle between document vectors (which for vectors having unit $L2$ norm is equivalent to their dot product). For Naive Bayes, length normalization reduces the influence of long documents in NB parameter estimation, but it is not clear if $L2$ norm if best for this purpose. Here we present an argument putting its use in the mixture-of-experts context.

Let us assume that a document d contains N terms. In the two-class context, the odds of the document belonging to the class C as opposed to $\overline{C}$ are computed by multinomial Naive Bayes as

$$\frac{p\left(C|d\right)}{p\left(\overline{C}|d\right)} = \frac{p\left(C\right)}{1-p\left(C\right)}\frac{p\left(d|C\right)}{p\left(d|\overline{C}\right)} = \frac{p\left(C\right)}{1-p\left(C\right)}\prod_i\left(\frac{p\left(t_i|C\right)}{p\left(t_i|\overline{C}\right)}\right)^{f_i}$$

where f_i is the number of occurrences of term t_i in d, with $\sum_i f_i = N$. Under the $tf \times idf$ weighting and $L2$ length-normalization transform of [29], the formula is changed to:

$$\frac{p\left(C|d\right)}{p\left(\overline{C}|d\right)} = \frac{p\left(C\right)}{1-p\left(C\right)}\frac{p\left(d|C\right)}{p\left(d|\overline{C}\right)} = \frac{p\left(C\right)}{1-p\left(C\right)}\prod_i\left(\frac{p\left(t_i|C\right)}{p\left(t_i|\overline{C}\right)}\right)^{z_i} \quad (3)$$

where for each term t_i in a document, z_i is its normalized $tf \times idf$ weight factor, so that $z_i \geq 0$ and $\sum_i z_i^2 = 1$.

Let

$$o_i = \frac{p\left(C|t_i\right)}{p\left(\overline{C}|t_i\right)}$$

denote the odds of term t_i belonging to the target class rather than the anti-target. Through this reformulation,

$$\frac{p(t_i|C)}{p(t_i|\bar{C})} = \frac{p(C|t_i)p(t_i)/p(C)}{p(\bar{C}|t_i)p(t_i)/p(\bar{C})} = \frac{1-p(C)}{p(C)}\cdot o_i,$$

we can express Eq. (3) as

$$\frac{p\left(C|d\right)}{p\left(\overline{C}|d\right)} = \frac{p\left(C\right)}{1-p\left(C\right)}\left(\frac{1-p\left(C\right)}{p\left(C\right)}\right)^{\sum_i z_i}\prod_i o_i^{z_i}$$

and if the document-length normalization based on the $L1$ norm instead of $L2$, then we have $\sum_i z_i = 1$, which yields

$$\frac{p\left(C|d\right)}{p\left(\overline{C}|d\right)} = \prod_i o_i^{z_i}$$

Thus the posterior odds for a document are a weighted geometric mean of the term-based odds for terms contained in a document. This type of formula has been widely used to combine probability distributions in the mixture of experts framework and is known as logarithmic opinion pooling [17]. Under this interpretation, the terms found in a document are considered as possibly correlated "experts", whose opinions are pooled or aggregated. The term weight z_i corresponds to the relative reliability of expert i. If all experts are considered equally reliable, the posterior probability of the classifier is computed as the geometric mean of the term-wise posteriors. In the log space this is equivalent to taking the arithmetic average of log-odds weights rather than just their sum as usually done for Naive Bayes. Note that unlike NB, in mixtures of experts there is no assumption of the experts' mutual independence conditioned on the class label. Indeed, much research has been devoted to derive weight values z_i that are able to take expert inter-correlation into account . Since odds are a measure of classifier confidence, taking the mean of individual opinions has the advantage that it cannot exceed the maximum of component odds. In practical terms, a document that contains only ambiguous terms cannot result in a highly confident decision. This is in contrast to the regular Naive Bayes where the compounding effect of many weakly positive or negative features may give the appearance of very high overall confidence. In the following, we will denote the mixture-of-experts variant of NB as NB-MX.

5. SUPERVISED AND UNSUPERVISED TERM WEIGHTING

Different term weighting methods besides the in-document term frequency have been studied extensively in the Information Retrieval community, but have not been fully studied in the context of NB research. In this section, we introduce several term weighing mechanisms including the traditional unsupervised methods such as $tf \times idf$ and also supervised methods such as feature weights learned by models in the previous stage. We also investigate approaches that improve these term weightings. In particular, we study the effect of combining the supervised and unsupervised approaches, and also proposed a monotonic term weighting transformation using a parameterized softmax function.

5.1 Term weighting for text categorization

Term weighting has been widely used in Information Retrieval and Text Categorization (TC). Typically, each term appearing in a document receives a positive weight, which is supposed to emphasize attributes of likely importance and de-emphasize common and irrelevant ones. Some of the most widely used techniques are based upon multiplicative combining of the in-document frequency with the inverse document frequency of the term in the training collection. Note that such measure of term importance does not take into account the class information, which may be relevant to the categorization task. It was suggested that for TC problems it may be advantageous to use supervised term weighting schemes [6] that derive the weight from functions used in ranking features for selection, such as Information Gain (IG) or χ^2. Although the benefits were shown for some datasets, it appeared that overall such supervised weighting schemes were quite comparable to the ones based on the traditional $tf \times idf$ approach. In related research, [33] suggested a novel term weighting measure that was shown to improve more consistently over $tf \times idf$ and methods investigated in [33] for SVM and KNN classifiers.

Recently, it was observed that the ranking of features derived from absolute weight values of certain linear classifiers can be competitive or superior to traditional feature ranking function such as IG, especially when applied in the context of the same learning algorithm [26]. In particular, the feature ranking induced by a linear SVM has showed good performance in that regard. This suggests that the classifier itself can be effective at deriving feature ranking. This was found to be the case for NB, where the ranking of features according to the absolute weights assigned to them by the classifier

$$ALO\left(t_i\right) = \left|weight\left(t_i\right)\right| = \left|\log \frac{P\left(t_i|C\right)}{P\left(t_i|\overline{C}\right)}\right| \tag{4}$$

was shown to outperform IG-based ranking in DSFS [20]. Although the use of *ALO* type weights for feature ranking in NB is rather new, it has been known that the related asymmetric the odds-ratio (OR) criterion

$$OR\left(t_i, C\right) = \frac{P\left(t_i|C\right)}{1 - P\left(t_i|C\right)} \cdot \frac{1 - P\left(t_i|\overline{C}\right)}{P\left(t_i|\overline{C}\right)}$$

works in terms of selecting relevant features for NB in text categorization [27, 8]. In this work we will use the *ALO* measure (4) as the supervised component of a term weighting function for NB.

5.2 Combination of supervised and unsupervised term weights

Given that supervised and unsupervised term weighting approaches are based on different types of information, it is natural to ask whether instead of using one to the exclusion of the other, a better overall weighting might be achieved by combining them together. We chose *ALO* (Eq. (4)) as the supervised component of a term weighting function, while retaining *idf* as the unsupervised component, i.e.,

$$tw\left(t_i\right) = idf\left(t_i\right) \cdot ALO\left(t_i\right) \tag{5}$$

Other term weighting functions could be used in place of *ALO* and *idf* and, indeed, more than two measures might be incorporated into an aggregate term weighting function that combines several measures of reliability. We consider a multiplicative scheme that, given N weighting functions, computes the combined term weight for term i in document d as

$$tw\left(t_i\right) = \prod_{j=1..N} f_j\left(t_i, d\right) \tag{6}$$

where $f_j\left(x_i, d\right) > 0$ is the term weight assigned by the j-th function.

Computing term weights (Eq. (5)) requires a two-step induction process similar to the one presented in Section 3, i.e.,

1. A NB-MX Naive Bayes classifier is built using the original feature representation
2. The absolute weight values are incorporated into the term weighting function
3. A second NB-MX classifier is built using the modified document representation

Note that one can also consider performing DSFS and term weighting at the same time. Also, the process described above could continue beyond the first two models, with a compounding effect of importance weights produced by the consecutive classifiers. It seems likely, however, that the weights of the individual NB classifiers will be highly correlated, thus providing little rational for continuing with the procedure beyond the first two models.

5.3 Optimizing term weights under monotonicity constraints

If document length normalization is not performed, for linear classifiers the score function is symmetrical with respect to classifier weights and term weights, i.e.,

$$score = b + \sum_i tw_i \cdot w_i \tag{7}$$

Usually the term weights are fixed and the parameter vector, w, is optimized. However, one could also reverse their roles and, for a fixed w, optimize the term weights, especially when the in-document term frequency is not taken to be part of the weighting function. This indeed was suggested by several researchers in the context of Naive Bayes, where the parameter weights are inexpensive to compute. While typically term weights are assumed to be non-negative, this requirement was dropped in [18] and [13]. In [13], a linear SVM was applied to optimize the term weight vector, while in [18] the use of Logistic Regression was proposed. In essence, Naive Bayes is used here as a "term weighting" function for the more expensive algorithms, although since the

term weights can take negative values this is an "unorthodox" method of applying term weights. As such, these methods need to be evaluated from the standpoint of whether NB term weighting provides a better or worse performance for the target algorithm (e.g., linear SVM) when compared to the native document representation or an alternative form of term weighting. Also, because of the potential negativity of weights and lack of length normalization, the mixture of experts analogy can no longer be made.

Document length normalization introduces nonlinearity that breaks the symmetry between term weights and classifier weights in Eq. (7). It also makes term weights document-specific, while maintaining their relative relationship, i.e., the ratio of any two weights before and after normalization remains the same. Joint optimization of classifier weights and term weights is possible, but is generally difficult due to the size of the parameter space. Term weighting works with many more choices than feature selection, which itself is a hard problem. We note, however, that the set of potential choices can be meaningfully constrained by the initial choice of the term weighting function. The good performance of feature selection functions in supervised term weighting suggests that these functions are not only useful in determining feature ranking but also in their relative importance. It is also possible that the importance values are suboptimal for a given classification task. Notice that any two feature selection functions that rank terms in the same order may behave differently when considered as term weighting functions. In particular, they may have different *steepness* as a function of rank, with steeper functions highly emphasizing the strongest terms and being analogous to aggressive document-specific feature selection. In contrast, flatter functions will favor document classification with significant contribution from a larger set of a document's features, which is analogous to mild document-specific feature selection. For any two equivalent ranking functions, it is difficult to stipulate *a priori* which one is more suitable for term weighting in a particular learning method, which suggests that their quality needs to be assessed via classification performance of the resulting classifier.

Let the initial ranking of features be

$$rank\,(t_1) \leq ... \leq rank\,(t_N)$$

Assuming that ranking of features is maintained, one can thus formulate the search for optimum feature weighting, as finding a set of values

$$tw\,(t_1) \geq ... \geq tw\,(t_N) \geq 0$$

such that the performance of a given learning method built over such document representation is maximized. Even with such monotone constraints, optimizing for both classifier parameters and term weights may be difficult. We therefore consider a parameterized monotonic transformation of the original term weights

$$f(\alpha, x) : x_1 \geq x_2 \Rightarrow f(\alpha, x_1) \geq f(\alpha, x_2)$$

for which the best parameter settings can be easily determined using a validation set. For a fixed ranking function one can consider a parameterized monotonic transformation of x that preserves term ranking, but also allows one to control the steepness of the mapping via parameter $\alpha > 0$. In this work we use a parameterized version of the *softmax* function. Given a set of values $\{x_i : i = 1..N\}$, it transforms them as

$$softmax\,(x_i) = \frac{\exp\,(\alpha \cdot x_i)}{\sum_j \exp\,(\alpha \cdot x_j)} \quad (8)$$

This normalization is applied on a per-document basis. For large values of α, Eq. (8) approximates the case of classifying with just a single "most important" feature, while for low values of the parameter is equivalent to treating all term weights as equal. We feel that this captures quite well the intent of document-specific term weighting, although other forms of transforming the feature ranking function could be considered. Also in the context of NB applied to large amounts of data, where speed and scalability are key, it seems counterintuitive to employ an optimization process for term weighting that is more expensive than the process of inducing NB itself. In this sense, evaluating the impact of Eq. (8) over a small set of α is acceptable and similar in complexity to the search for optimum smoothing parameter for estimating individual probabilities.

Table 1: Dataset statistics

Dataset	trn set size	tst set size	feature count
Hotmail	765,000	150,000	2,644,921
TREC-2005	30,727	61,455	208,844
TREC-2006	12,607	25,214	133,495

6. EXPERIMENTS

To evaluate the impact of the modeling choices described in this work we chose three email datasets, corresponding to the spam filtering problem, where Naive Bayes tends to be used quite often [25]. While other TC applications also often require high levels of precision, this is particularly true in spam detection. In the previous work, researchers typically emphasize comparing the performance of different spam classification solutions in the region corresponding to low false-positive rates. The data we used corresponded to the 2005 and 2006 TREC Spam Filtering Track datasets [5, 3], as well as to a non-public Hotmail dataset, which has been used in the work of Yih *et al.* [35]. All three collections contained a time-sorted mix of spam and non-spam messages. In the experiments we used the initial portion of each dataset for training and the remaining portion for testing. As discussed among others in [4], although cross-validation is often used to estimate classification performance in the text domain, in spam filtering it is not appropriate as it tends to produce overoptimistic estimates. In addition to the fact that content distribution of email in general has strong time dependence, spam detection is an adversarial problem, and spam techniques are rapidly evolving to evade the current detection techniques. The details of the datasets are given in Table 1. For the case of the Hotmail data, we used the same experimental setup as described in [35]. For the two TREC datasets, we considered training with a smaller fraction of the dataset than used for testing to better reflect the fact that batch-trained filters tend to operate over prolonged periods of time without adaptation, which may cause their performance to degrade with respect to the online filtering approaches.

As baselines for Naive Bayes, we used the standard multinomial version of the classifier (labeled as NB), as well as a modification of Rennie *et al.* [29], which uses *idf* feature weighting and $L2$ length normalization of document feature

vectors (labeled here as the NB-IR). To provide comparisons for Naive Bayes performance we used Logistic Regression (LR) based on binary features and linear SVM based on *idf* weighted and $L2$ length normalized feature vectors, which reflects the typical document representation used in conjunction with these two classifiers [15, 10]. Both Logistic Regression and SVMs are considered as state-of-the-art learning algorithms in the text categorization domain, and have also been applied successfully to the spam filtering problem [21, 9, 35]. We used the default parameter settings of SVM_{light}[1] and for Logistic Regression we assumed Gaussian prior of variance 1 for all features.

Spam filters often extract a number of specialized features capturing the domain knowledge of filter creators. We did not include any of such features. Instead, the email messages were represented as bag of words contained in the message bodies and subject lines. The feature selection was limited to ignoring terms that occurred fewer than three times in the training collection. The in-document frequency of each feature was ignored so as to provide clearer comparison of the impact of feature-based weighting and also because for NB, ignoring term frequency can actually improve the classifier performance [32].

In performing document-specific feature selection (DSFS), we considered limiting document sparsity to top-N features, with N in {5, 10, 25, 50, 75, 100, 150, 200, 300, 500, 1000}. In the first set of experiments we assessed the difference of performing post-induction DSFS vs. making it part of the classifier learning. As illustrated in Figure 1, training Naive Bayes with the reduced representation does have a positive effect on performance. It is also apparent, however, that the optimum sparsity settings obtained via the post-induction method and the double induction method match closely. A useful heuristic might therefore consist of searching for the optimum setting using the faster post-induction method and performing re-induction of NB using the final sparsity setting to insure better accuracy. Note that feature selection has a substantial impact on NB performance and results in large improvements over the baseline multinomial Naive Bayes.

Feature selection can be combined with feature weighting and to assess the impact of such combination we considered the following weighting functions listed in Table 3. In the formulas listed, w_i corresponds to the *ALO* weight of Eq. (4), while idf_i denotes the *idf* weight according to

$$idf_i = const - \log(1 + f_i)$$

where f_i is the number of training documents containing term i. Although the term weighting functions can be applied with and without document length normalization, in our experiments we restricted ourselves to NB-MX, i.e., normalization according to the $L1$ measure. We will label the different variants of NB-MX used by the type of the term weighting function used, e.g., NB-MX using *abs_idf* term weighting will be referred to as `abs_idf`. The functions `abs` and `softmax_abs` represent purely supervised, `idf` and `softmax_idf` purely unsupervised weighting functions, while `abs_idf` and `softmax_abs_idf` correspond to combining the supervised and unsupervised notions of feature relevance. For the softmax weighting functions, we tested the steepness parameter α in {0.01, 0.05, 0.1, 1, 1.5} and reported

[1] http://www.cs.cornell.edu/People/tj/svm_light/

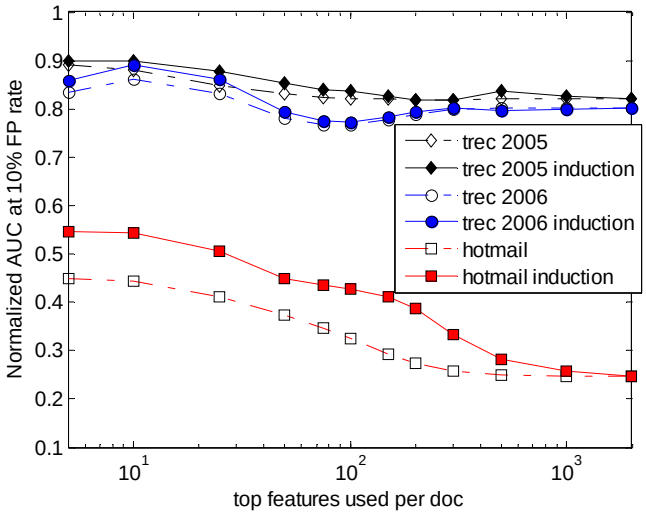

Figure 1: Effects of using post-induction DSFS vs having it impact the classifier during induction. Note that the optimum settings between the two methods are very consistent.

the results based on the best choice. The weighting function `idf` closely corresponds to NB-IR with the exception that it uses $L1$ rather than $L2$ document length normalization. We used $L1$ normalization throughout since it better fits the mixture-of-experts framework as described in Section 4. The *geo* case is the special case of combining term-wise log odds via a simple geometric mean, which reflects the impact of document length normalization under uniform term weighting.

Following [35], we decided to evaluate classifier performance by focusing on the ROC analysis in the area corresponding to the false positive rate (FP) range of [0, 0.14]. This includes the essential region of importance for the spam filtering application without focusing on any particular setting[2]. To provide a single figure of merit we modified the commonly used metric of area under the ROC (AUC), by normalizing the area in the section of the ROC plot

$$\{(FP, TP) : 0 \leq FP \leq 0.1,\ 0 \leq TP \leq 1\}$$

so that the maximum possible value is 1. This is achieved by dividing this area by the length of the FP region of interest, i.e., $FP \leq 0.1$, which we believe brackets the constraints of the spam filtering application. The figure of merit will be called $AUC_{0.1}$.

7. RESULTS

Figure 1 compares the effects (in terms of $AUC_{0.1}$) of document-specific feature selection for NB with and without having DSFS influence the induction process (see Section 3 for description of the two approaches). As expected, having

[2] While in personal filtering of email the acceptable FP range would be much lower, the dataset used in [35] is due to a large and diverse user population and is affected by class noise (i.e., user labeling mistakes), as well as judgement disagreements over messages sent to multiple recipients. This considerably extends the viable FP range.

Table 2: Normalized AUC corresponding to the FP range of [0,0.1], with the maximum possible value of 1. Apart from the benchmark algorithms of SVM and LR, for each NB variant performance is measured using all features and with DSFS. The best performer in each column is typed in bold, whereas the best among the NB variant is italicized.

	hotmail		trec-2005		trec-2006	
classifier	all features	DSFS	all features	DSFS	all features	DSFS
NB	0.2479	0.5468	0.8196	0.8994	0.8017	0.8889
NB-IR	0.5561	0.5709	0.9207	0.9252	0.9521	0.9539
LR	0.4877	NA	0.9461	NA	0.9384	NA
SVM	0.4830	NA	**0.9477**	NA	**0.9754**	NA
abs	0.5840	0.5859	0.9218	0.9218	0.9314	0.9315
softmax$_{abs}$	0.5894	0.5894	0.9262	0.9262	0.9377	0.9378
abs_idf	**0.5959**	*0.5959*	0.9449	0.9450	*0.9623*	0.9627
softmax$_{abs_idf}$	0.5945	0.5945	*0.9475*	*0.9476*	0.9617	*0.9644*
idf	0.5804	0.5826	0.9359	0.9365	0.9542	0.9552
softmax$_{idf}$	0.5739	0.5760	0.9159	0.9173	0.9603	0.9637
geo	0.5051	0.5672	0.8941	0.9042	0.8711	0.8927

DSFS impact the weights of the final classifier has a beneficial effect, particularly for the Hotmail and TREC-2005 datasets. It is also apparent, however, that both methods are consistent when it comes to identifying the optimum number of features. Thus the post-induction method (as being faster) should be an attractive heuristic in identifying the optimum DSFS setting, which can then be used to train the final NB classifier. Note that the best performance is reached for low document sparsity settings (5 and 10 features per document), which supports the arguments of NB performing best in for low sparsity representations [26]. In the remaining experiments, whenever DSFS is applied, it uses the full induction process.

Table 2 provides comprehensive performance comparison in AUC_{10} for the learning algorithms considered. For each dataset, the left column lists the performance using all features, where the right column provides the best results achieved via DSFS. Since feature selection was not performed for the benchmark discriminative learners, for SVM and LR the right column is left blank. For NB-MX methods using the softmax variant of term weighting, the results listed correspond to the best results achieved over the set of α parameters considered. The rows following the LR and SVM benchmarks correspond to the variants of NB-MX using $L1$ feature vector normalization and different types of term weighting (detailed in Table 3). The top two rows represent the NB baseline and NB-IR using *idf* based term weighting and $L2$-based normalization.

Among the different NB variants, NB-MX performs the best, particularly using feature weights incorporating both the *ALO* and *idf* relevance measures. The straightforward relevance integration of `abs_idf` works very well, but in some cases it can be improved with the parametric softmax. Even with just the *idf* weights, $L1$-based length normalization tends to improve over $L2$-based length normalization, which suggests that the mixture-of-experts way of averaging term-wise posterior distributions has an advantage over the IR-inspired approach.

NB-MX is very competitive with discriminative learners. While LR and SVM could also be optimized with feature weighting and selection, the comparison is still important since LR and SVM in the default configurations tend to perform very well. The softmax$_{abs_idf}$ variant outperforms LR on all three datasets and SVM on the Hotmail dataset. SVM still beats its competitors for TREC-2005 and TREC-2006 but we note that even then the differences between SVM and softmax$_{abs_idf}$ are rather small. It is interesting that both SVM and LR performed poorly compared to NB for the Hotmail dataset. We attribute this behavior to the fact that Hotmail data is known to have a significant amount of class noise (2-5%), which discriminative learners may find confusing and thus overfit [2]. The TREC datasets on the other hand were hand-labeled and cleaned so the amount of class noise is expected to be low. We intend to further investigate and confirm the impact class noise in future work.

Table 3: Term weighting functions we considered. Only the normalized versions were used in the experiments.

weighting function	not-normalized	L1 length normalized
abs	$\lvert w_i \rvert$	$\frac{\lvert w_i \rvert}{\sum_j \lvert w_j \rvert}$
softmax$_{abs}$	$\exp(\alpha \cdot \lvert w_i \rvert)$	$\frac{\exp(\alpha \cdot \lvert w_i \rvert)}{\sum_j \exp(\alpha \cdot \lvert w_j \rvert)}$
abs_idf	$idf_i \cdot \lvert w_i \rvert$	$\frac{\lvert idf_i \cdot w_i \rvert}{\sum_j \lvert idf_j \cdot w_j \rvert}$
softmax$_{abs_idf}$	$\exp(\alpha \cdot idf_i \cdot \lvert w_i \rvert)$	$\frac{\exp(\alpha \cdot idf_i \cdot \lvert w_i \rvert)}{\sum_j \exp(\alpha \cdot idf_j \cdot \lvert w_j \rvert)}$
idf	idf_i	$\frac{\lvert idf_i \rvert}{\sum_j \lvert idf_j \rvert}$
softmax$_{idf}$	$\exp(\alpha \cdot idf_i)$	$\frac{\exp(\alpha \cdot idf_i)}{\sum_j \exp(\alpha \cdot idf_j)}$
geo	NA	$\frac{1}{N}$

Table 2 indicates that the differences between different NB variants due to feature weighting and normalization are more prominent than the differences due to feature selection vs using all features. Indeed, in some cases NB-MX actually performed best using all features. DSFS made most impact for classifiers using uniform term weighting (NB and geo), with particularly dramatic improvements in the case NB. However, the geo variant of NB provides the most straightforward way of improving the classifier's performance.

While AUC_{10} provides a reasonable figure of merit, it is informative to examine the actual ROC curves in the region of interest. For comparison, we chose the configurations using all features without parameter optimization. NB-MX was therefore represented by the abs_idf variant. In ad-

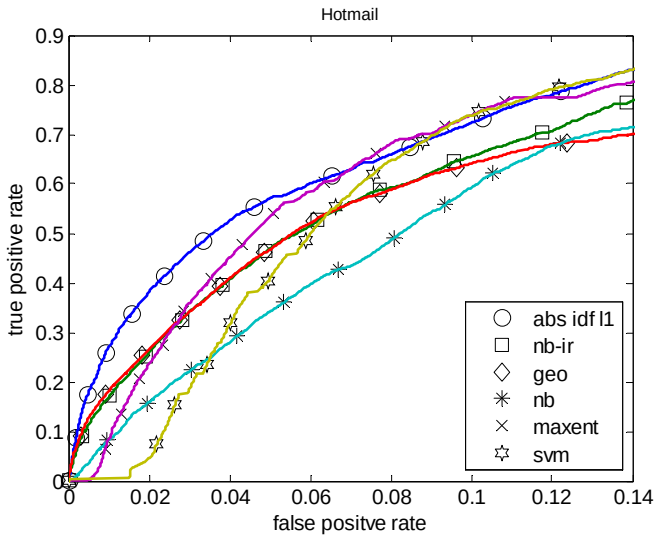

Figure 2: ROC comparison for the Hotmail dataset. NB-MX with *abs_idf* term weighting has a large advantage for low FP rates.

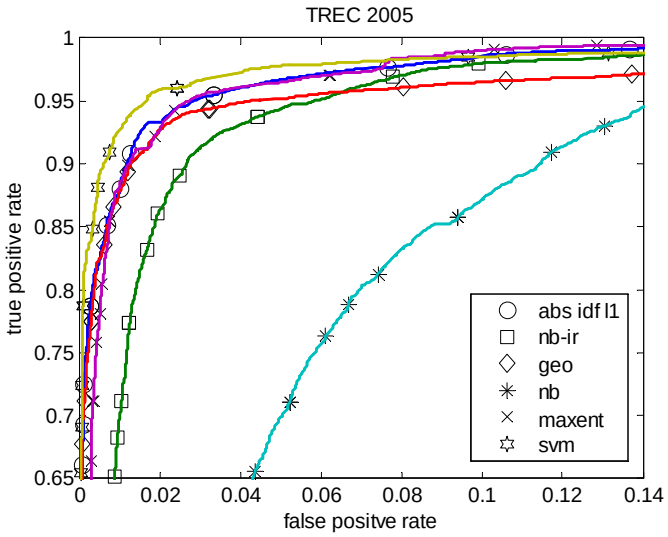

Figure 3: ROC comparison for the TREC-2005 dataset. NB and NB-IR lag in performance in the region of low FP rates.

dition, `NB`, `NB-IR`, `geo`, `LR`, and `SVM` are also included. The ROC curves for the individual datasets are shown in Figures 2–4. For TREC-2005 (Figure 3) and TREC-2006 (Figure 4), NB-MX is very close to LR, with SVM showing clear dominance over all others. NB-IR shows lower performance in the low-FP region but catches up at higher FP rates. For the Hotmail dataset (Figure 2), NB-MX shows clear dominance in the low-FP range, both over the discriminative learners and other NB variants. LR and SVM catch on eventually, but for this noisy dataset using a generative learner such as NB-MX is actually advantageous when high-precision spam detection is concerned.

8. RELATED WORK

Many researchers noted the relatively high improvement of performance of standard NB to feature selection, especially in high-dimensional problems such as text. Several methods were found to be effective, particularly odds-ratio, information gain, and feature ranking derived from linear SVMs [27][26]. More recently, it was suggested that in applications involving highly asymmetric misclassification costs, document-specific feature selection may be more beneficial than regular feature selection, which when too aggressive may leave some documents without useful features [20].

Logarithmic option pooling has been studied extensively as a method of aggregating probability distribution in the mixture of experts context [17]. Other approaches of combining probability distributions are discussed in [14]. Robinson [30] proposed a method of improving the calibration of Naive Bayes posterior probabilities in spam detection based on meta-analysis. His formula uses unweighted geometric mean of term-specific probabilities, which makes it related to applying $L1$ document length normalization over uniform term-weighting. The use of geometric mean with NB has also been advocated in [32].

In [8] several different normalization techniques were investigated with Naive Bayes. However, the normalization was applied to term frequency vectors and not to term importance weights. Also, the normalization was defined somewhat loosely, without requiring that a feature vector has unit norm according to a well defined metric. Nevertheless, it was found that normalization of in-document frequencies has a large positive effect on NB. In the same work, it was also found that term importance weighting based on risk ratio improves NB performance in TC, where risk ratio can be related to $\exp(ALO)$. An $L1$ like normalization of document feature vectors was also advocated in [19] as a way of curbing the influence of long document on parameter estimates.

In [18] and [13] the weights learned by Naive Bayes were used as term weights, and with this modified document representation a more complex learning algorithm was used to train the final model without document length normalization. In [28], a related scheme was proposed, but to keep the complexity of the overall model low, the feature vector for each document was split into a small number of natural components (e.g., the subject line and body of an email) over which a Naive Bayes model was trained. The scores provided by these models were used to generate a low dimensional document representation used to train a Logistic Regression model.

For applications of linear text classifiers demanding low false-positive rates, Yih *et al.* [35] proposed to chain several models, each trained on the subset of the data that could not be classified by the previous model with high enough confidence. In [35], this method was shown to be particularly effective for NB. In related work, Wu *et al.* [34] considered building a tree of Naive Bayes classifier, with the root of the tree building using all data, and all other nodes conditioned on the partition of the data (at default threshold) by the parent node. Alternative approaches to improving classifier performance at low FP rates involve training with costs or utilities. These however have not been found particularly

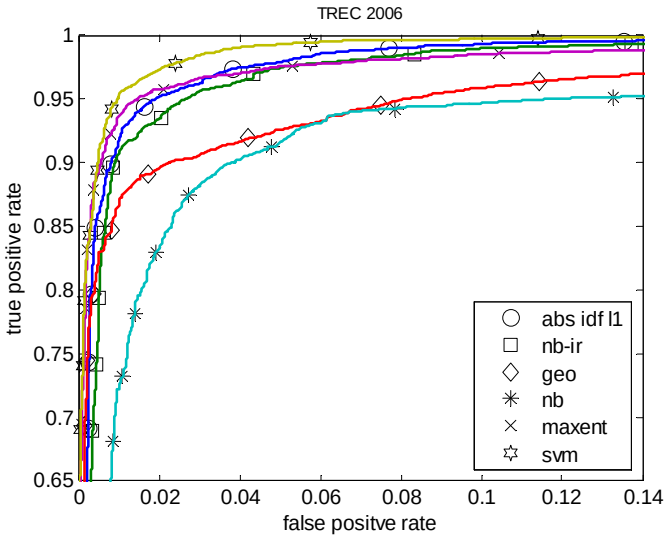

Figure 4: ROC comparison for the TREC-2006 dataset. NB-MX tracks closely the LR classifier. NB-IR performs comparably for this dataset.

effective for NB [35].

9. CONCLUSIONS

In this work we focused on text classification problems requiring relatively low false-positive rates. We demonstrated that with appropriate document representation, the Naive Bayes classifier provides a baseline for high-precision TC that is hard to beat, even for top performers such as linear SVMs or Logistic Regression. In particular, $L1$-based feature vector length normalization allows Naive Bayes to be interpreted in the mixture of experts framework, with the posterior output of the classifier corresponding to weighted geometric mean of the term-specific posteriors. This puts the use of document length normalization for Naive Bayes on a stronger footing and indeed this type of transformation appears to have a practical advantage over the traditional $L2$ approach. Our results show that document-length normalization plays a major role in improving Naive Bayes performance, where even the simplest form of using the geometric mean tends to outperform multinomial Naive Bayes by a wide margin. Both supervised and unsupervised term weighting functions improve these results further, while the combination of these two approaches yields the best overall results. We find this indicative of the value of combining different feature relevance measures in term weighting. While in this work we focused on *idf* and *ALO* derived term weights, many alternatives could be considered or combined for this purpose. Such combination-type term weights might benefit other text classifiers, such as SVMs as well.

Feature relevance measures are typically used for ranking and their values are not necessarily optimum as term weights. We proposed to optimize their values under the constraint that ranking order of term weights is preserved. We believe that this type of constraint is meaningful, as supported be experimental results, and should be instrumental in avoiding the potential overfitting involved in optimizing term weights and classifier weights at the same time. More thorough investigating of this topic will be a subject of future work.

Document-specific feature selection is quite effective for Naive Bayes and its variants. We showed that it can be further improved by making it part of the classifier induction (as in traditional feature selection), although the originally proposed post-induction approach provides a useful heuristic for cheaply identifying the optimum setting.

10. ACKNOWLEDGMENTS

The authors would like to thank Max Chickering, Dennis Decoste and Yang Song as well as the anonymous reviewers for their helpful comments and suggestions.

11. REFERENCES

[1] P. Bennett. Assessing the calibration of naive Bayes' posterior estimates. Technical Report CMU-CS-00-155, School of Computer Science, Carnegie Mellon University, 2000.

[2] L. Chen, J. Huang, and Z. Gong. An anti-noise text categorization method based on support vector machines. In *Proceedings of AWIC 2005*, pages 272–278, 20025.

[3] G. Cormack. The TREC 2006 spam filter evaluation track. *Virus Bulletin*, (1), 2007.

[4] G. Cormack and A. Bratko. Batch and online spam filter comparison. In *Proceedings of the 3rd Conference on Email and Anti-Spam (CEAS-2006)*, 2006.

[5] G. Cormack and T. Lynam. TREC 2005 spam track overview. In *Proceedings of TREC 2005 - the Fourteenth Text REtrieval Conference*, 2005.

[6] F. Debole and F. Sebastiani. Supervised term weighting for automated text categorization. In *Proceedings of the 2003 ACM symposium on Applied Computing*, pages 784–788, 2003.

[7] P. Domingos and M. Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130, 1997.

[8] P. Domingos and M. Pazzani. Some effective techniques for naive Bayes text classification. *IEEE Transactions on Knowledge and Data Engineering*, 18(11), 2006.

[9] H. Drucker, D. Wu, and V. Vapnik. Support Vector Machines for Spam Categorization. *IEEE Transactions on Neural Networks*, 10(5):1048–1054, 1999.

[10] L. Edda and J. Kindermann. Text categorization with support vector machines. how to represent texts in input space? *Machine Learning*, 46:423–444, 2002.

[11] C. Elkan. The foundations of cost-sensitive learning. In *IJCAI*, pages 973–978, 2001.

[12] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.

[13] T. Gartner and P. Flach. WBCsvm : Weighted Bayesian classification based on support vector machines. In *Proceedings of the Eighteenth International Conference on Machine Learning*, 2001.

[14] C. Genest and J. Zidek. Combining probability distributions: A critique and an annotated bibliography. *Statistical Science*, 1986.

[15] J. Goodman. Sequential conditional generalized iterative scaling. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 9–16, 2001.

[16] P. Graham. A plan for spam http://www.paulgraham.com/spam.html, 2002.

[17] G. Hinton. Products of experts. In *Proceedings of the Ninth International Conference on Artificial Neural Networks (ICANN99)*, pages 1–6, 1999.

[18] S. Hong, J. Hosking, and T. Natarajan. Ensemble modeling through multiplicative adjustment of class probability. In *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002)*, pages 621–624, 2002.

[19] A. Juan and H. Ney. Reversing and smoothing the multinomial naive Bayes text classifier. In *Proceedings of the 2nd Int. Workshop on Pattern Recognition in Information Systems (PRIS 2002)*, pages 200–212, 2002.

[20] A. Kołcz. Local sparsity control for naive Bayes with extreme misclassification costs. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2005.

[21] A. Kołcz and J. Alspector. SVM-based filtering of e-mail spam with content-specific misclassification costs. In *Proceedings of the Workshop on Text Mining (TextDM'2001)*, 2001.

[22] M. Lee and E. Corlett. Sequential sampling models of human text classification. *Cognitive Science*, 27(2):159–1193, 2003.

[23] D. D. Lewis. Naive (Bayes) at forty: the independence assumption in information retrieval. In *Proceedings of the 10th European Conference on Machine Learning*, pages 4–15, 1998.

[24] A. K. McCallum and K. Nigam. A comparison of event models for Naive Bayes text classification. In *AAAI-98 Workshop on Learning for Text Categorization*, 1998.

[25] V. Metsis, I. Androutsopoulos, and G. Paliouras. Spam filtering with Naive Bayes - which Naive Bayes? In *Proceedings of the 3rd Conference on Email and Anti-Spam (CEAS-2006)*, 2006.

[26] D. Mladenic, J. Brank, M. Grobelnik, and N. Milic-Frayling. Feature selection using linear classifier weights: Interaction with classification models. In *Proceedings of the 27th Annual International ACM SIGIR Conference (SIGIR2004)*, 2004.

[27] D. Mladenic and M. Grobelnik. Feature selection for unbalanced class distribution and Naive Bayes. In *Proceedings of ICML 1999*, 1999.

[28] R. Raina, Y. Shen, A. Ng, and A. McCallum. Classification with hybrid generative/discriminative models. In *Proceedings of NIPS 16*, 2004.

[29] J. Rennie, L. Shih, J. Teevan, and D. Karger. Tackling the poor assumptions of Naive Bayes text classifiers. In *Proceedings of the Twentieth International Conference on Machine Learning*, 2003.

[30] G. Robinson. A statistical approach to the spam problem. *Linux Journal*, (107), 2003.

[31] M. Sauban and B. Pfahringer. Text categorisation using document profiling. In *Proceedings of PKDD 2003*, pages 411–422, 2003.

[32] K. Schneider. Techniques for improving the performance of naive Bayes for text classification. In *Proceedings of CICLing 2005*, pages 682–693, 2005.

[33] P. Soucy and G. Mineau. Beyond TFIDF weighting for text categorization in the vector space model. In *Proceedings of the Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005)*, pages 1130–1135, 2005.

[34] H. Wu, T. Phang, B. Liu, and X. Li. A refinement approach to handling model misfit in text categorization. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002.

[35] W. Yih, J. Goodman, and G. Hulten. Learning at low false positive rates. In *Proceedings of the 3rd Conference on Email and Anti-Spam (CEAS-2006)*, 2006.

[36] Z. Zheng and K. T. G.I. Webb. Lazy Bayesian rules: A lazy semi-naive Bayesian learning technique competitive to boosting decision trees. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 493–502, 1999.