Spam Filtering with Naive Bayes – Which Naive Bayes? [*]

Vangelis Metsis [†]
Institute of Informatics and Telecommunications,
N.C.S.R. “Demokritos”,
Athens, Greece

Ion Androutsopoulos
Department of Informatics,
Athens University of Economics and Business,
Athens, Greece

Georgios Paliouras
Institute of Informatics and Telecommunications,
N.C.S.R. “Demokritos”,
Athens, Greece

ABSTRACT

Naive Bayes is very popular in commercial and open-source anti-spam e-mail filters. There are, however, several forms of Naive Bayes, something the anti-spam literature does not always acknowledge. We discuss five different versions of Naive Bayes, and compare them on six new, non-encoded datasets, that contain ham messages of particular Enron users and fresh spam messages. The new datasets, which we make publicly available, are more realistic than previous comparable benchmarks, because they maintain the temporal order of the messages in the two categories, and they emulate the varying proportion of spam and ham messages that users receive over time. We adopt an experimental procedure that emulates the incremental training of personalized spam filters, and we plot ROC curves that allow us to compare the different versions of NB over the entire tradeoff between true positives and true negatives.

1. INTRODUCTION

Although several machine learning algorithms have been employed in anti-spam e-mail filtering, including algorithms that are considered top-performers in text classification, like Boosting and Support Vector Machines (see, for example, [4, 6, 10, 16]), Naive Bayes (NB) classifiers currently appear to be particularly popular in commercial and open-source spam filters. This is probably due to their simplicity, which makes them easy to implement, their linear computational complexity, and their accuracy, which in spam filtering is comparable to that of more elaborate learning algorithms [2]. There are, however, several forms of NB classifiers, and the anti-spam literature does not always acknowledge this.

In their seminal papers on learning-based spam filters, Sahami et al. [21] used a NB classifier with a *multi-variate Bernoulli* model (this is also the model we had used in [1]), a form of NB that relies on Boolean attributes, whereas Pantel and Lin [19] in effect adopted the *multinomial* form of NB, which normally takes into account term frequencies. McCallum and Nigam [17] have shown experimentally that the multinomial NB performs generally better than the multi-variate Bernoulli NB in text classification, a finding that Schneider [24] and Hovold [12] verified with spam filtering experiments on Ling-Spam and the PU corpora [1, 2, 23]. In further work on text classification, which included experiments on Ling-Spam, Schneider [25] found that the multinomial NB surprisingly performs even better when term frequencies are replaced by Boolean attributes.

The multi-variate Bernoulli NB can be modified to accommodate continuous attributes, leading to what we call the *multi-variate Gauss* NB, by assuming that the values of each attribute follow a normal distribution within each category [14]. Alternatively, the distribution of each attribute in each category can be taken to be the average of several normal distributions, one for every different value the attribute has in the training data of that category, leading to a NB version that John and Langley [14] call *Flexible Bayes* (FB). In previous work [2], we found that FB clearly outperforms the multi-variate Gauss NB on the PU corpora, when the attributes are term frequencies divided by document lengths, but we did not compare FB against the other NB versions.

In this paper we shed more light on the five versions of NB mentioned above, and we evaluate them experimentally on six new, non-encoded datasets, collectively called *Enron-Spam*, which we make publicly available.[1] Each dataset contains ham (non-spam) messages from a single user of the Enron corpus [15], to which we have added fresh spam messages with varying ham-spam ratios. Although a similar approach was adopted in the public benchmark of the TREC 2005 Spam Track, to be discussed below, we believe that our datasets are better suited to evaluations of personalized filters, i.e., filters that are trained on incoming messages of a particular user they are intended to protect, which is the type of filters the experiments of this paper consider. Unlike Ling-Spam and the PU corpora, in the new datasets we maintain the order in which the original messages of the two categories were received, and we emulate the varying proportion of ham and spam messages that users receive over time. This allows us to conduct more realistic experiments, and to take into account the incremental training of personal filters. Furthermore, rather than focussing on a handful of relative misclassification costs (cost of false positives vs. false negatives; $\lambda = 1, 9, 999$ in our previous work),

[*] This version of the paper contains some minor corrections in the description of Flexible Bayes, which were made after the conference.

[†] Work carried out mostly while at the Department of Informatics, Athens University of Economics and Business.

CEAS 2006 - Third Conference on Email and Anti-Spam, July 27-28, 2006, Mountain View, California USA

[1] The Enron-Spam datasets are available from `http://www.iit.demokritos.gr/skel/i-config/` and `http://www.aueb.gr/users/ion/publications.html` in both raw and pre-processed form. Ling-Spam and the PU corpora are also available from the same addresses.

we plot entire ROC curves, which allow us to compare the different versions of NB over the entire tradeoff between true positives and true negatives.

Note that several publicly available spam filters appear to be using techniques described as "Bayesian", but which are very different from any form of NB discussed in the academic literature and any other technique that would normally be called Bayesian therein.[2] Here we focus on NB versions published in the academic literature, leaving comparisons against other "Bayesian" techniques for future work.

Section 2 below presents the event models and assumptions of the NB versions we considered. Section 3 explains how the datasets of our experiments were assembled and the evaluation methodology we used; it also highlights some pitfalls that have to be avoided when constructing spam filtering benchmarks. Section 4 then presents and discusses our experimental results. Section 5 concludes and provides directions for further work.

2. NAIVE BAYES CLASSIFIERS

As a simplification, we focus on the textual content of the messages. Operational filters would also consider information such as the presence of suspicious headers or token obfuscation [11, 21], which can be added as additional attributes in the message representation discussed below. Alternatively, separate classifiers can be trained for textual and other attributes, and then form an ensemble [9, 22].

In our experiments, each message is ultimately represented as a vector $\langle x_1, \dots, x_m \rangle$, where $x_1, \dots, x_m$ are the values of attributes $X_1, \dots, X_m$, and each attribute provides information about a particular token of the message.[3] In the simplest case, all the attributes are Boolean: $X_i = 1$ if the message contains the token; otherwise, $X_i = 0$. Alternatively, their values may be term frequencies (TF), showing how many times the corresponding token occurs in the message.[4] Attributes with TF values carry more information than Boolean ones. Hence, one might expect NB versions that use TF attributes to perform better than those with Boolean attributes, an expectation that is not always confirmed, as already mentioned. A third alternative we employed, hereafter called *normalized* TF, is to divide term frequencies by the total number of token occurrences in the message, to take into account the message's length. The motivation is that knowing, for example, that "rich" occurs 3 times in a message may be a good indication that the message is spam if it is only two paragraphs long, but not if the message is much longer.

Following common text classification practice, we do not assign attributes to tokens that are too rare (we discard tokens that do not occur in at least 5 messages of the training data). We also rank the remaining attributes by information gain, and use only the m best, as in [1, 2, 21], and elsewhere. We experimented with $m = 500$, 1000, and 3000. Note that the information gain ranking treats the attributes as Boolean, which may not be entirely satisfactory when employing a NB version with non-Boolean attributes. Schneider [24] experimented with alternative versions of the information gain measure, intended to be more suitable to the TF-valued attributes of the multinomial NB. His results, however, indicate that the alternative versions do not lead to higher accuracy, although sometimes they allow the same level of accuracy to be reached with fewer attributes.

From Bayes' theorem, the probability that a message with vector $\vec{x} = \langle x_1, \dots, x_m \rangle$ belongs in category c is:

$$p(c \mid \vec{x}) = \frac{p(c) \cdot p(\vec{x} \mid c)}{p(\vec{x})}.$$

Since the denominator does not depend on the category, NB classifies each message in the category that maximizes $p(c) \cdot p(\vec{x} \mid c)$. In the case of spam filtering, this is equivalent to classifying a message as spam whenever:

$$\frac{p(c_s) \cdot p(\vec{x} \mid c_s)}{p(c_s) \cdot p(\vec{x} \mid c_s) + p(c_h) \cdot p(\vec{x} \mid c_h)} > T,$$

with $T = 0.5$, where c_h and c_s denote the ham and spam categories. By varying T, one can opt for more true negatives (correctly classified ham messages) at the expense of fewer true positives (correctly classified spam messages), or vice-versa. The a priori probabilities $p(c)$ are typically estimated by dividing the number of training messages of category c by the total number of training messages. The probabilities $p(\vec{x} \mid c)$ are estimated differently in each NB version.

2.1 Multi-variate Bernoulli NB

Let us denote by $F = \{t_1, \dots, t_m\}$ the set of tokens that correspond to the m attributes after attribute selection. The multi-variate Bernoulli NB treats each message d as a set of tokens, containing (only once) each t_i that occurs in d. Hence, d can be represented by a binary vector $\vec{x} = \langle x_1, \dots, x_m \rangle$, where each x_i shows whether or not t_i occurs in d. Furthermore, each message d of category c is seen as the result of m Bernoulli trials, where at each trial we decide whether or not t_i will occur in d. The probability of a positive outcome at trial i (t_i occurs in d) is $p(t_i \mid c)$. The multi-variate Bernoulli NB makes the additional assumption that the outcomes of the trials are independent given the category. This is a "naive" assumption, since word co-occurrences in a category are not independent. Similar assumptions are made in all NB versions, and although in most cases they are over-simplistic, they still lead to very good performance in many classification tasks; see, for example, [5] for a theoretical explanation. Then, $p(\vec{x} \mid c)$ can be computed as:

$$p(\vec{x} \mid c) = \prod_{i=1}^{m} p(t_i \mid c)^{x_i} \cdot (1 - p(t_i \mid c))^{(1-x_i)},$$

and the criterion for classifying a message as spam becomes:

$$\frac{p(c_s) \cdot \prod_{i=1}^{m} p(t_i \mid c_s)^{x_i} \cdot (1 - p(t_i \mid c_s))^{(1-x_i)}}{\sum_{c \in \{c_s, c_h\}} p(c) \cdot \prod_{i=1}^{m} p(t_i \mid c)^{x_i} \cdot (1 - p(t_i \mid c))^{(1-x_i)}} > T,$$

where each $p(t \mid c)$ is estimated using a Laplacean prior as:

$$p(t \mid c) = \frac{1 + M_{t,c}}{2 + M_c},$$

and $M_{t,c}$ is the number of training messages of category c that contain token t, while M_c is the total number of training messages of category c.

[2] These techniques derive mostly from P. Graham's "A plan for spam"; see `http://www.paulgraham.com/spam.html`.

[3] Attributes may also be mapped to character or token n-grams, but previous attempts to use n-grams in spam filtering led to contradictory or inconclusive results [2, 12, 19].

[4] We treat punctuation and other non-alphabetic characters as separate tokens. Many of these are highly informative as attributes, because they are more common in spam messages (especially obfuscated ones) than ham messages; see [2].

2.2 Multinomial NB, TF attributes

The multinomial NB with TF attributes treats each message d as a bag of tokens, containing each one of t_i as many times as it occurs in d. Hence, d can be represented by a vector $\vec{x} = \langle x_1, \ldots, x_m \rangle$, where each x_i is now the number of occurrences of t_i in d. Furthermore, each message d of category c is seen as the result of picking independently $|d|$ tokens from F with replacement, with probability $p(t_i \mid c)$ for each t_i.[5] Then, $p(\vec{x} \mid c)$ is the multinomial distribution:

$$p(\vec{x} \mid c) = p(|d|) \cdot |d|! \cdot \prod_{i=1}^{m} \frac{p(t_i \mid c)^{x_i}}{x_i!},$$

where we have followed the common assumption [17, 24, 25] that $|d|$ does not depend on the category c. This is an additional over-simplistic assumption, which is more questionable in spam filtering. For example, the probability of receiving a very long spam message appears to be smaller than that of receiving an equally long ham message.

The criterion for classifying a message as spam becomes:

$$\frac{p(c_s) \cdot \prod_{i=1}^{m} p(t_i \mid c_s)^{x_i}}{\sum_{c \in \{c_s, c_h\}} p(c) \cdot \prod_{i=1}^{m} p(t_i \mid c)^{x_i}} > T,$$

where each $p(t \mid c)$ is estimated using a Laplacean prior as:

$$p(t \mid c) = \frac{1 + N_{t,c}}{m + N_c},$$

and $N_{t,c}$ is now the number of occurrences of token t in the training messages of category c, while $N_c = \sum_{i=1}^{m} N_{t_i,c}$.

2.3 Multinomial NB, Boolean attributes

The multinomial NB with Boolean attributes is the same as with TF attributes, including the estimates of $p(t \mid c)$, except that the attributes are now Boolean. It differs from the multi-variate Bernoulli NB in that it does not take into account directly the absence ($x_i = 0$) of tokens from the message (there is no $(1 - p(t_i \mid c))^{(1-x_i)}$ factor), and it estimates the $p(t \mid c)$ with a different Laplacean prior.

It may seem strange that the multinomial NB might perform better with Boolean attributes, which provide less information than TF ones. As Schneider [25] points out, however, it has been proven [7] that the multinomial NB with TF attributes is equivalent to a NB version with attributes modelled as following Poisson distributions in each category, assuming that the document length is independent of the category. Hence, the multinomial NB may perform better with Boolean attributes, if TF attributes in reality do not follow Poisson distributions.

2.4 Multi-variate Gauss NB

The multi-variate Bernoulli NB can be modified for real-valued attributes, by assuming that each attribute follows a normal distribution $g(x_i; \mu_{i,c}, \sigma_{i,c})$ in each category c, where:

$$g(x_i; \mu_{i,c}, \sigma_{i,c}) = \frac{1}{\sigma_{i,c}\sqrt{2\pi}} e^{-\frac{(x_i - \mu_{i,c})^2}{2\sigma_{i,c}^2}},$$

and the mean ($\mu_{i,c}$) and typical deviation ($\sigma_{i,c}$) of each distribution are estimated from the training data. Then, assuming again that the values of the attributes are independent given the category, we get:

$$p(\vec{x} \mid c) = \prod_{i=1}^{m} g(x_i; \mu_{i,c}, \sigma_{i,c}),$$

and the criterion for classifying a message as spam becomes:

$$\frac{p(c_s) \cdot \prod_{i=1}^{m} g(x_i; \mu_{i,c_s}, \sigma_{i,c_s})}{\sum_{c \in \{c_s, c_h\}} p(c) \cdot \prod_{i=1}^{m} g(x_i; \mu_{i,c}, \sigma_{i,c})} > T.$$

This allows us to use normalized TF attributes, whose values are (non-negative) reals, unlike the TF attributes of the multinomial NB. Real-valued attributes, however, may not follow normal distributions. With our normalized TF attributes, there is also the problem that negative values are not used, which leads to a significant loss of probability mass in the (presumed) normal distributions of attributes whose variances are large and means are close to zero.

2.5 Flexible Bayes

Instead of using a single normal distribution for each attribute per category, FB models $p(x_i \mid c)$ as the average of $L_{i,c}$ normal distributions with different mean values, but the same typical deviation:

$$p(x_i \mid c) = \frac{1}{L_{i,c}} \cdot \sum_{l=1}^{L_{i,c}} g(x_i; \mu_{i,c,l}, \sigma_c),$$

where $L_{i,c}$ is the number of different values X_i has in the training data of category c. Each of these values is used as the mean $\mu_{i,c,l}$ of a normal distribution of that category. All the distributions of a category c are taken to have the same typical deviation, estimated as $\sigma_c = \frac{1}{\sqrt{M_c}}$, where M_c is again the number of training messages in c. Hence, the distributions of each category become narrower as more training messages of that category are accumulated; in the case of our normalized TF attributes, this also alleviates the problem of probability mass loss of the multi-variate Gauss NB. By averaging several normal distributions, FB can approximate the true distributions of real-valued attributes more closely than the multi-variate Gauss NB, when the assumption that the attributes follow normal distributions is violated.

The computational complexity of all five NB versions is $O(m \cdot N)$ during training, where N is the total number of training messages. At classification time, the computational complexity of the first four versions is $O(m)$, while the complexity of FB is $O(m \cdot N)$, because of the need to sum the L_i distributions. Consult [2] for further details.

3. DATASETS AND METHODOLOGY

There has been significant effort to generate public benchmark datasets for anti-spam filtering. One of the main concerns is how to protect the privacy of the users (senders and receivers) whose ham messages are included in the datasets.

The first approach is to use ham messages collected from freely accessible newsgroups, or mailing lists with public archives. Ling-Spam, the earliest of our benchmark datasets, follows this approach [23]. It consists of spam messages received at the time and ham messages retrieved from the archives of the Linguist list, a moderated and, hence, spam-free list about linguistics. Ling-Spam has the disadvantage that its ham messages are more topic-specific than the

[5] In effect, this is a unigram language model. Additional variants of the multinomial NB can be formed by using n-gram language models instead [20]. See also [13] for other improvements that can be made to the multinomial NB.

messages most users receive. Hence, it can lead to over-optimistic estimates of the performance of learning-based spam filters. The SpamAssassin corpus is similar, in that its ham messages are publicly available; they were collected from public fora, or they were donated by users with the understanding they may be made public.[6] Since they were received by different users, however, SpamAssassin's ham messages are less topic-specific than those a *single* user would receive. Hence, the resulting dataset is inappropriate for experimentation with personalized spam filters.

An alternative solution to the privacy problem is to distribute information about each message (e.g., the frequencies of particular words in each message), rather than the messages themselves. The Spambase collection follows this approach. It consists of vectors, each representing a single message (spam or ham), with each vector containing the values of pre-selected attributes, mostly word frequencies. The same approach was adopted in a corpus developed for a recently announced ECML-PKDD 2006 challenge.[7] Datasets that adopt this approach, however, are much more restrictive than Ling-Spam and the SpamAssassin corpus, because their messages are not available in raw form, and, hence, it is impossible to experiment with attributes other than those chosen by their creators.

A third approach is to release benchmarks each consisting of messages received by a particular user, after replacing each token by a unique number in all the messages. The mapping between tokens and numbers is not released, making it extremely difficult to recover the original messages, other than perhaps common words and phrases therein. This bypasses privacy problems, while producing messages whose token sequences are very close, from a statistical point of view, to the original ones. We have used this encoding scheme in the PU corpora [1, 2, 23]. However, the loss of the original tokens still imposes restrictions; for example, it is impossible to experiment with different tokenizers.

Following the Enron investigation, the personal files of approximately 150 Enron employees were made publicly available.[8] The files included a large number of personal e-mail messages, which have been used to create e-mail classification benchmarks [3, 15], including a public benchmark corpus for the TREC 2005 Spam Track.[9] During the construction of the latter benchmark, several spam filters were employed to weed spam out of the Enron message collection. The collection was then augmented with spam messages collected in 2005, leading to a benchmark with 43,000 ham and approximately 50,000 spam messages. The 2005 Spam Track experiments did not separate the resulting corpus into personal mailboxes, although such a division might have been possible via the 'To:' field. Hence, the experiments corresponded to the scenario where a single filter is trained on a collection of messages received by many different users, as opposed to using personalized filters.

As we were more interested in personalized spam filters, we focussed on six Enron employees who had large mailboxes. More specifically, we used the mailboxes of employees `farmer-d`, `kaminski-v`, `kitchen-l`, `williams-w3`, `beck-s`, and `lokay-m`, in the cleaned-up form provided by Bekkerman [3], which includes only ham messages.[10] We also used spam messages obtained from four different sources: (1) the SpamAssassin corpus, (2) the Honeypot project,[11] (3) the spam collection of Bruce Guenter (BG),[12] and spam collected by the third author of this paper (GP).

The first three spam sources above collect spam via traps (e.g., e-mail addresses published on the Web in a way that makes it clear to humans, but not to crawlers, that they should not be used), resulting in multiple copies of the same messages. We applied a heuristic to the spam collection we obtained from each one of the first three spam sources, to identify and remove multiple copies; the heuristic is based on the number of common text lines in each pair of spam messages. After removing duplicates, we merged the spam collections obtained from sources 1 and 2, because the messages from source 1 were too few to be used on their own and did not include recent spam, whereas the messages from source 2 were fresher, but they covered a much shorter period of time. The resulting collection (dubbed SH; SpamAssassin spam plus Honeypot) contains messages sent between May 2001 and July 2005. From the third spam source (BG) we kept messages sent between August 2004 and July 2005, a period ending close to the time our datasets were constructed. Finally, the fourth spam source is the only one that does not rely on traps. It contains all the spam messages received by GP between December 2003 and September 2005; duplicates were not removed in this case, as they are part of a normal stream of incoming spam.

The six ham message collections (six Enron users) were each paired with one of the three spam collections (SH, BG, GP). Since the vast majority of spam messages are not personalized, we believe that mixing ham messages received by one user with spam messages received by others leads to reasonable benchmarks, provided that additional steps are taken, as discussed below. The same approach can be used in future to replace the spam messages of our datasets with fresher ones. We also varied the ham-spam ratios, by randomly subsampling the spam or ham messages, where necessary. In three of the resulting benchmark datasets, we used a ham-spam ratio of approximately 3:1, while in the other three we inverted the ratio to 1:3. The total number of messages in each dataset is between five and six thousand. The six datasets emulate different situations faced by real users, allowing us to obtain a more complete picture of the performance of learning-based filters. Table 1 summarizes the characteristics of the six datasets. Hereafter, we refer to the first, second, . . . , sixth dataset of Table 1 as Enron1, Enron2, . . . , Enron6, respectively.

In addition to what was mentioned above, the six datasets were subjected to the following pre-processing steps. First, we removed messages sent by the owner of the mailbox (we checked if the address of the owner appeared in the 'To:', 'Cc:', or 'Bcc:' fields), since we believe e-mail users are increasingly adopting better ways to keep copies of outgoing messages. Second, as a simplification, we removed all HTML tags and the headers of the messages, keeping only their

[6] The SpamAssassin corpus and Spambase are available from `http://www.spamassassin.org/publiccorpus/` and `http://www.ics.uci.edu/∼mlearn/MLRepository.html`.

[7] See `http://www.ecmlpkdd2006.org/challenge.html`.

[8] See `http://fercic.aspensys.com/members/manager.asp`.

[9] Consult `http://plg.uwaterloo.ca/ gvcormac/spam/` for further details. We do not discuss the other three corpora of the 2005 Spam Track, as they are not publicly available.

[10] The mailboxes can be downloaded from `http://www.cs.umass.edu/∼ronb/datasets/enron_flat.tar.gz`.

[11] Consult `http://www.projecthoneypot.org/`.

[12] See `http://untroubled.org/spam/`.

Table 1: Composition of the six benchmark datasets.

ham + spam	ham:spam	ham, spam periods
`farmer-d` + GP	3672:1500	[12/99, 1/02], [12/03, 9/05]
`kaminski-v` + SH	4361:1496	[12/99, 5/01], [5/01, 7/05]
`kitchen-l` + BG	4012:1500	[2/01, 2/02], [8/04, 7/05]
`williams-w3` + GP	1500:4500	[4/01, 2/02], 12/03, 9/05]
`beck-s` + SH	1500:3675	[1/00, 5/01], [5/01, 7/05]
`lokay-m` + BG	1500:4500	[6/00, 3/02], [8/04, 7/05]

subjects and bodies. In operational filters, HTML tags and headers can provide additional useful attributes, as mentioned above; hence, our datasets lead to conservative estimates of the performance of operational filters. Third, we removed spam messages written in non-Latin character sets, because the ham messages of our datasets are all written in Latin characters, and, therefore, non-Latin spam messages would be too easy to identify; i.e., we opted again for harder datasets, that lead to conservative performance estimates.

One of the main goals of our evaluation was to emulate the situation that a new user of a personalized learning-based anti-spam filter faces: the user starts with a small amount of training messages, and retrains the filter as new messages arrive. As noted in [8], this incremental retraining and evaluation differs significantly from the cross-validation experiments that are commonly used to measure the performance of learning algorithms, and which have been adopted in many previous spam filtering experiments, including our own [2]. There are several reasons for this, including the varying size of the training set, the increasingly more sophisticated tricks used by spam senders over time, the varying proportion of spam to ham messages in different time periods, which makes the estimation of priors difficult, and the topic shift of spam messages over time. Hence, an incremental retraining and evaluation procedure that also takes into account the characteristics of spam that vary over time is essential when comparing different learning algorithms in spam filtering. In order to realize this incremental procedure with the use of our six datasets, we needed to order the messages of each dataset in a way that preserves the original order of arrival of the messages in each category; i.e., each spam message must be preceded by all spam messages that arrived earlier, and the same applies to ham messages. The varying ham-ratio ratio over time also had to be emulated. (The reader is reminded that the spam and ham messages of each dataset are from different time periods. Hence, one cannot simply use the dates of the messages.) This was achieved by using the following algorithm in each dataset:

1. Let S and H be the sets of spam and ham messages of the dataset, respectively.

2. Order the messages of H by time of arrival.

3. Insert $|S|$ spam slots between the ordered messages of H by $|S|$ independent random draws from $\{1, \ldots, |H|\}$ with replacement. If the outcome of a draw is i, a new spam slot is inserted after the i-th ham message. A ham message may thus be followed by several slots.

4. Fill the spam slots with the messages of S, by iteratively filling the earliest empty spam slot with the oldest message of S that has not been placed to a slot.

The actual dates of the messages are then discarded, and we assume that the messages (ham and spam) of each dataset arrive in the order produced by the algorithm above. Figure 1 shows the resulting fluctuation of the ham-spam ratio over batches of 100 adjacent messages each. The first batch contains the "oldest" 100 messages, the second one the 100 messages that "arrived" immediately after those of the first batch, etc. The ham-spam ratio in the entire dataset is 2.45.

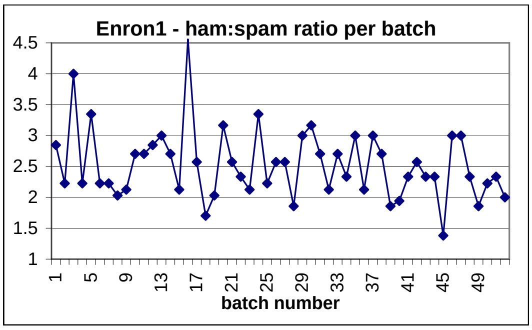

Figure 1: Fluctuation of the ham-spam ratio.

In each ordered dataset, the incremental retraining and evaluation procedure was implemented as follows:

1. Split the sequence of messages into batches $b_1, \ldots, b_l$ of k adjacent messages each, preserving the order of arrival. Batch b_l may have less than k messages.

2. For $i = 1$ to $l - 1$, train the filter (including attribute selection) on the messages of batches $1, \ldots, i$, and test it on the messages of b_{i+1}.

Note that at the end of the evaluation, each message of the dataset (excluding b_1) will have been classified exactly once. The number of true positives (TP) is the number of spam messages that have been classified as spam, and similarly for false positives (FP, ham misclassified as spam), true negatives (TN, correctly classified ham), and false negatives (FN, spam misclassified as ham). We set $k = 100$, which emulates the situation where the filter is retrained every 100 new messages.[13] We assume that the user marks as false negatives spam messages that pass the filter, and inspects periodically for false positives a "spam" folder, where messages identified by the filter as spam end up.

In our evaluation, we used spam recall ($\frac{TP}{TP+FN}$) and ham recall ($\frac{TN}{TN+FP}$). Spam recall is the proportion of spam messages that the filter managed to identify correctly (how much spam it blocked), whereas ham recall is the proportion of ham messages that passed the filter. Spam recall is the complement of spam misclassification rate, and ham recall the complement of ham misclassification rate, the two measures that were used in the TREC 2005 Spam Track. In order to evaluate the different NB versions across the entire tradeoff between true positives and true negatives, we present the evaluation results by means of ROC curves, plotting *sensitivity* (spam recall) against 1− *specificity* (the complement of ham recall, or ham misclassification rate). This is the

[13] An NB-based filter can easily be retrained on-line, immediately after receiving each new message. We chose $k = 100$ to make it easier to add in future work additional experiments with other learning algorithms, such as SVMs, which are computationally more expensive to train.

NB version	Enr1	Enr2	Enr3	Enr4	Enr5	Enr6
FB	7.87	3.46	1.43	1.31	0.11	0.34
MV Gauss	5.56	4.75	1.97	12.7	3.36	5.27
MN TF	0.88	0.95	0.20	0.50	0.75	0.18
MV Bernoulli	2.10	0.95	1.09	0.45	1.14	0.88
MN Boolean	2.31	1.97	2.04	0.43	0.39	0.20

Table 2: Maximum difference (×100) in spam recall across 500, 1000, 3000 attributes for $T = 0.5$.

NB version	Enr1	Enr2	Enr3	Enr4	Enr5	Enr6
FB	0.61	0.23	1.72	0.54	0.48	0.34
MV Gauss	1.17	0.75	5.94	1.77	5.91	4.88
MN TF	2.17	1.38	1.02	0.61	1.70	1.22
MV Bernoulli	1.47	0.63	6.37	2.04	2.11	1.22
MN Boolean	0.53	0.68	0.10	0.48	1.36	2.17

Table 3: Maximum difference (×100) in ham recall across 500, 1000, 3000 attributes for $T = 0.5$.

normal definition of ROC analysis, when treating spam as the positive and ham as the negative class.

The ROC curves capture the overall performance of the different NB versions in each dataset, but fail to provide a picture of the progress made by each NB version during the incremental procedure. For this reason, we additionally examine the learning curves of the five methods in terms of the two measures for $T = 0.5$, i.e., we plot spam and ham recall as the training set increases during the incremental retraining and evaluation procedure.

4. EXPERIMENTAL RESULTS

4.1 Size of attribute set

We first examined the impact of the number of attributes on the effectiveness of the five NB versions.[14] As mentioned above, we experimented with 500, 1000, and 3000 attributes. The full results of these experiments (not reported here) indicate that overall the best results are achieved with 3000 attributes, as one might have expected. The differences in effectiveness across different numbers of attributes, however, are rather insignificant. As an example, Tables 2 and 3 show the maximum differences in spam and ham recall, respectively, across the three sizes of the attribute set, for each NB version and dataset, with $T = 0.5$; note that the differences are in percentage points. The tables show that the differences are very small in all five NB versions for this threshold value, and we obtained very similar results for all thresholds. Consequently, in operational filters the differences in effectiveness may not justify the increased computational cost that larger attribute sets require, even though the increase in computational cost is linear in the number of attributes.

4.2 Comparison of NB versions

Figure 2 shows the ROC curves of the five NB versions in each one of the six datasets.[15] All the curves are for 3000 attributes, and the error bars correspond to 0.95 confidence intervals; we show error bars only at some points to avoid cluttering the diagrams. Since the tolerance of most users on misclassifying ham messages is very limited, we have restricted the horizontal axis ($1-\text{specificity} = 1-\text{ham recall}$) of all diagrams to $[0, 0.2]$, i.e., a maximum of 20% of misclassified ham, in order to improve the readability of the diagrams. On the vertical axis (sensitivity, spam recall) we show the full range, which allows us to examine what proportion of spam messages the five NB versions manage to block when requesting a very low ham misclassification rate (when $1-\text{specificity}$ approaches 0). The optimal performance point in an ROC diagram is the top-left corner, while the area under each curve (AUC) is often seen as a summary of the performance of the corresponding method. We do not, however, believe that standard AUC is a good measure for spam filters, because it is dominated by non-high specificity (ham recall) regions, which are of no interest in practice. Perhaps one should compute the area for $1 - \text{specificity} \in [0, 0.2]$ or $[0, 0.1]$. Even then, however, it is debatable how the area should be computed when ROC curves do not span the entire $[0, 0.2]$ or $[0, 0.1]$ range of the horizontal axis (see below).

A first conclusion that can be drawn from the results of Figure 2 is that some datasets, such as Enron4, are “easier” than others, such as Enron1. There does not seem to be a clear justification for these differences, in terms of the ham-spam ratio or the spam source used in each dataset.

Despite its theoretical association to term frequencies, in all six datasets the multinomial NB seems to be doing better when Boolean attributes are used, which agrees with Schneider’s observations [25]. The difference, however, is in most cases very small and not always statistically significant; it is clearer in the first dataset and, to a lesser extent, in the last one. Furthermore, the multinomial NB with Boolean attributes seems to be the best performer in 4 out of 6 datasets, although again by a small and not always statistically significant margin, and it is clearly outperformed only by FB in the other 2 datasets. This is particularly interesting, since many NB-based spam filters appear to adopt the multinomial NB with TF attributes or the multi-variate Bernoulli NB (which uses Boolean attributes); the latter seems to be the worst among the NB versions we evaluated. Among the NB versions that we tested with normalized TF attributes (FB and the multi-variate Gauss NB), overall FB is clearly the best. However, FB does not always outperform the other NB version that uses non-Boolean attributes, namely the multinomial NB with TF attributes.

The FB classifier shows signs of impressive superiority in Enron1 and Enron2; and its performance is almost undistinguishable from that of the top performers in Enron5 and Enron6. However, it does not perform equally well, compared to the top performers, in the other two datasets (Enron3, Enron4), which strangely include what appears to be the easiest dataset (Enron4). One problem we noticed with FB is that its estimates for $p(c \mid \vec{x})$ are very close to 0 or 1; hence, varying the threshold T has no effect on the classification of many messages. This did not allow us to obtain higher ham recall (lower $1-\text{specificity}$) by trading off spam recall (sensitivity) as well as in the other NB versions, which is why the FB ROC curves are shorter in some of the diagrams. (The same comment applies to the multi-variate Gauss NB.) Having said that, we were able to reach a ham recall level of 99.9% or higher with FB in most of the datasets.

Overall, the multinomial NB with Boolean attributes and FB obtained the best results in our experiments, but the dif-

[14] We used a modified version of FILTRON [18] for our experiments, with WEKA’s implementations of the five NB versions; see `http://www.cs.waikato.ac.nz/∼ml/weka/`.

[15] Please view the figures in color, consulting the on-line version of this paper if necessary; see `http://www.ceas.cc/`.

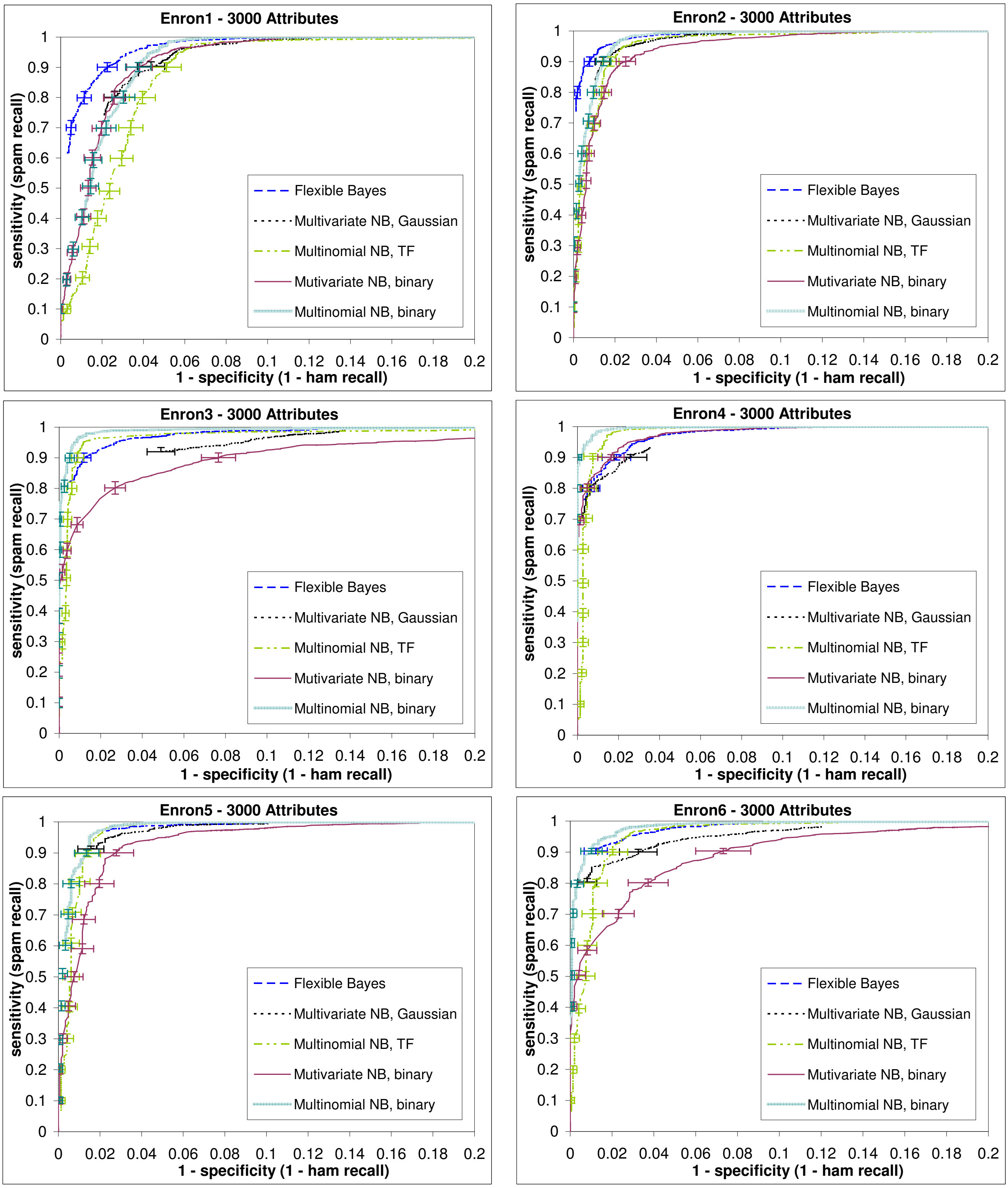

Figure 2: ROC curves of the five NB versions with 3000 attributes.

NB version	Enr1	Enr2	Enr3	Enr4	Enr5	Enr6	Avg.
FB	90.50	93.63	96.94	95.78	99.56	99.55	95.99
MV Gauss	93.08	95.80	97.55	80.14	95.42	91.95	92.32
MN TF	95.66	96.81	95.04	97.79	99.42	98.08	97.13
MV Bern.	97.08	91.05	97.42	97.70	97.95	97.92	96.52
MN Bool.	96.00	96.68	96.94	97.79	99.69	98.10	**97.53**

Table 4: Spam recall (%) for 3000 attributes, $T = 0.5$.

NB version	Enr1	Enr2	Enr3	Enr4	Enr5	Enr6	Avg.
FB	97.64	98.83	95.36	96.61	90.76	89.97	94.86
MV Gauss	94.83	96.97	88.81	99.39	97.28	95.87	95.53
MN TF	94.00	96.78	98.83	98.30	95.65	95.12	96.45
MV Bern.	93.19	97.22	75.41	95.86	90.08	82.52	89.05
MN Bool.	95.25	97.83	98.88	99.05	95.65	96.88	**97.26**

Table 5: Ham recall (%) for 3000 attributes, $T = 0.5$.

ferences from the other NB versions were often very small. Taking into account its smoother trade-off between ham and spam recall, and its better computational complexity at run time, we tend to prefer the multinomial NB with Boolean attributes over FB, but further experiments are necessary to establish its superiority with confidence. For completeness, Tables 4 and 5 list the spam and ham recall, respectively, of the NB versions on the 6 datasets for $T = 0.5$, although comparing at a fixed threshold T is not particularly informative; for example, two methods may obtain the same results at different thresholds. On average, the multinomial NB with Boolean attributes again has the best results, both in spam and ham recall.

4.3 Learning curves

Figure 3 shows the learning curves (spam and ham recall as more training messages are accumulated over time) of the multinomial NB with Boolean attributes on the six datasets for $T = 0.5$. It is interesting to observe that the curves do not increase monotonically, unlike most text classification experiments, presumably because of the unpredictable fluctuation of the ham-spam ratio, the changing topics of spam, and the adversarial nature of anti-spam filtering. In the "easiest" dataset (Enron4) the classifier reaches almost perfect performance, especially in terms of ham recall, after a few hundreds of messages, and quickly returns to near-perfect performance whenever a deviation occurs. As more training messages are accumulated, the deviations from the perfect performance almost disappear. In contrast, in more difficult datasets (e.g., Enron1) the fluctuation of ham and spam recall is continuous. The classifier seems to adapt quickly to changes, though, avoiding prolonged plateaus of low performance. Spam recall is particularly high and stable in Enron5, but this comes at the expense of frequent large fluctuations of ham recall; hence, the high spam recall may be the effect of a tradeoff between spam and ham recall.

5. CONCLUSIONS AND FURTHER WORK

We discussed and evaluated experimentally in a spam filtering context five different versions of the Naive Bayes (NB) classifier. Our investigation included two versions of NB that have not been used widely in the spam filtering literature, namely Flexible Bayes (FB) and the multinomial NB with Boolean attributes. We emulated the situation faced by a new user of a personalized learning-based spam filter, adopting an incremental retraining and evaluation procedure. The six datasets that we used, and which we make publicly available, were created by mixing freely available ham and spam messages in different proportions. The mixing procedure emulates the unpredictable fluctuation over time of the ham-spam ratio in real mailboxes.

Our evaluation included plotting ROC curves, which allowed us to compare the different NB versions across the entire tradeoff between true positives and true negatives. The most interesting result of our evaluation was the very good performance of the two NB versions that have been used less in spam filtering, i.e., FB and the multinomial NB with Boolean attributes; these two versions collectively obtained the best results in our experiments. Taking also into account its lower computational complexity at run time and its smoother trade-off between ham and spam recall, we tend to prefer the multinomial NB with Boolean attributes over FB, but further experiments are needed to be confident. The best results in terms of effectiveness were generally achieved with the largest attribute set (3000 attributes), as one might have expected, but the gain was rather insignificant, compared to smaller and computationally cheaper attribute sets.

We are currently collecting more data, in a setting that will allow us to evaluate the five NB versions and other learning algorithms on several real mailboxes with the incremental retraining and evaluation method. The obvious caveat of these additional real-user experiments is that it will not be possible to provide publicly the resulting datasets in a non-encoded form. Therefore, we plan to release them using the encoding scheme of the PU datasets.

6. REFERENCES

[1] I. Androutsopoulos, J. Koutsias, K. Chandrinos, and C. Spyropoulos. An experimental comparison of Naive Bayesian and keyword-based anti-spam filtering with encrypted personal e-mail messages. In *23rd ACM SIGIR Conference*, pages 160–167, Athens, Greece, 2000.

[2] I. Androutsopoulos, G. Paliouras, and E. Michelakis. Learning to filter unsolicited commercial e-mail. technical report 2004/2, NCSR "Demokritos", 2004.

[3] R. Beckermann, A. McCallum, and G. Huang. Automatic categorization of email into folders: benchmark experiments on Enron and SRI corpora. Technical report IR-418, University of Massachusetts Amherst, 2004.

[4] X. Carreras and L. Marquez. Boosting trees for anti-spam email filtering. In *4th International Conference on Recent Advances in Natural Language Processing*, pages 58–64, Tzigov Chark, Bulgaria, 2001.

[5] P. Domingos and M. Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29(2–3):103130, 1997.

[6] H. D. Drucker, D. Wu, and V. Vapnik. Support Vector Machines for spam categorization. *IEEE Transactions On Neural Networks*, 10(5):1048–1054, 1999.

[7] S. Eyheramendy, D. Lewis, and D. Madigan. On the Naive Bayes model for text categorization. In *9th International Workshop on Artificial Intelligence and Statistics*, pages 332–339, Key West, Florida, 2003.

[8] T. Fawcett. In "vivo" spam filtering: a challenge

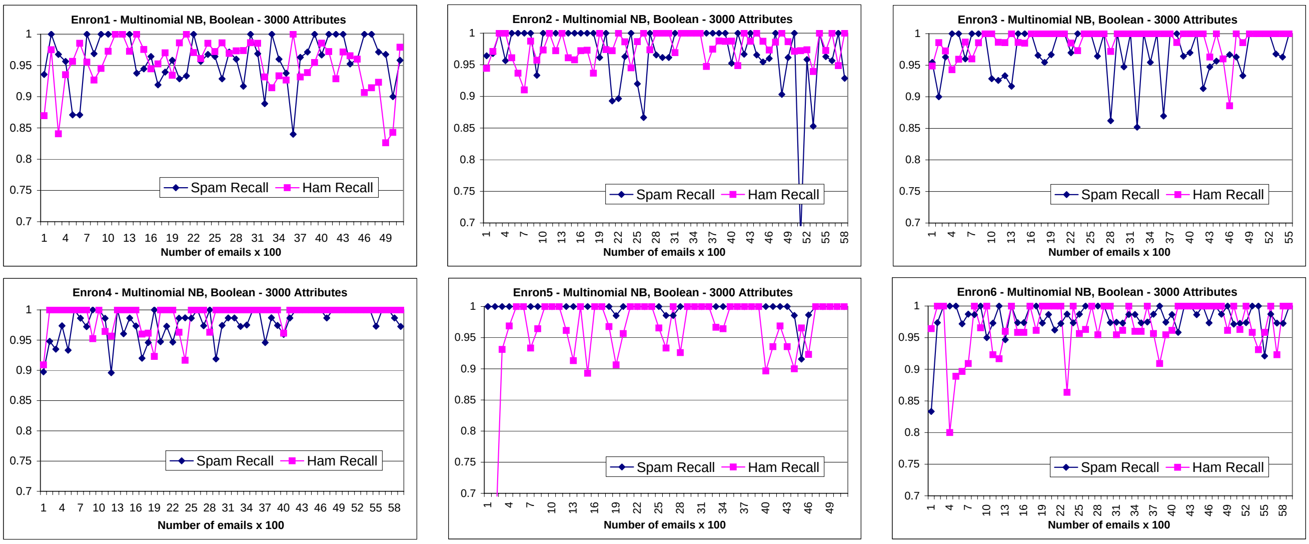

Figure 3: Learning curves for the multinomial NB with Boolean attributes and $T = 0.5$.

problem for KDD. *SIGKDD Explorations*, 5(2):140–148, 2003.

[9] S. Hershkop and S. Stolfo. Combining email models for false positive reduction. In *11th ACM SIGKDD Conference*, pages 98–107, Chicago, Illinois, 2005.

[10] J. G. Hidalgo. Evaluating cost-sensitive unsolicited bulk email categorization. In *17th ACM Symposium on Applied Computing*, pages 615–620, 2002.

[11] J. G. Hidalgo and M. M. Lopez. Combining text and heuristics for cost-sensitive spam filtering. In *4th Computational Natural Language Learning Workshop*, pages 99–102, Lisbon, Portugal, 2000.

[12] J. Hovold. Naive Bayes spam filtering using word-position-based attributes. In *2nd Conference on Email and Anti-Spam*, Stanford, CA, 2005.

[13] J. T. J.D.M. Rennie, L. Shih and D. Karger. Tackling the poor assumptions of Naive Bayes classifiers. In *20th International Conference on Machine Learning*, pages 616–623, Washington, DC, 2003.

[14] G. John and P. Langley. Estimating continuous distributions in Bayesian classifiers. In *11th Conference on Uncertainty in Artificial Intelligence*, pages 338–345, Montreal, Quebec, 1995.

[15] B. Klimt and Y. Yang. The Enron corpus: a new dataset for email classification research. In *15th European Conference on Machine Learning and the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 217–226, Pisa, Italy, 2004.

[16] A. Kolcz and J. Alspector. SVM-based filtering of e-mail spam with content-specific misclassification costs. In *Workshop on Text Mining, IEEE International Conference on Data Mining*, San Jose, California, 2001.

[17] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *AAAI'98 Workshop on Learning for Text Categorization*, pages 41–48, Madison, Wisconsin, 1998.

[18] E. Michelakis, I. Androutsopoulos, G. Paliouras, G. Sakkis, and P. Stamatopoulos. Filtron: a learning-based anti-spam filter. In *1st Conference on Email and Anti-Spam*, Mountain View, CA, 2004.

[19] P. Pantel and D. Lin. SpamCop: a spam classification and organization program. In *Learning for Text Categorization – Papers from the AAAI Workshop*, pages 95–98, Madison, Wisconsin, 1998.

[20] F. Peng, D. Schuurmans, and S. Wang. Augmenting naive bayes classifiers with statistical language models. *Information Retrieval*, 7:317–345, 2004.

[21] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A Bayesian approach to filtering junk e-mail. In *Learning for Text Categorization – Papers from the AAAI Workshop*, pages 55–62, Madison, Wisconsin, 1998.

[22] G. Sakkis, I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. Spyropoulos, and P. Stamatopoulos. Stacking classifiers for anti-spam filtering of e-mail. In *Conference on Empirical Methods in Natural Language Processing*, pages 44–50, Carnegie Mellon University, Pittsburgh, PA, 2001.

[23] G. Sakkis, I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. Spyropoulos, and P. Stamatopoulos. A memory-based approach to anti-spam filtering for mailing lists. *Information Retrieval*, 6(1):49–73, 2003.

[24] K.-M. Schneider. A comparison of event models for Naive Bayes anti-spam e-mail filtering. In *10th Conference of the European Chapter of the ACL*, pages 307–314, Budapest, Hungary, 2003.

[25] K.-M. Schneider. On word frequency information and negative evidence in Naive Bayes text classification. In *4th International Conference on Advances in Natural Language Processing*, pages 474–485, Alicante, Spain, 2004.