Statistics and The War on Spam

David Madigan

Rutgers University

Abstract

Text categorization algorithms assign texts to predefined categories. The study of such algorithms has a rich history dating back at least forty years. In the last decade or so, the statistical approach has dominated the literature. The essential idea is to infer a text categorization algorithm from a set of labeled documents, i.e., documents with known category assignments. The algorithm, once learned, automatically assigns labels to new documents. Motivated by a successful application to "spam" or "unsolicited bulk e-mail" filtering, this chapter will present the "Naive Bayes" approach to statistical text categorization.

1 Introduction

"Are you an Inventor? Have a Great Idea? We can help"

"Now available - cash maker"

"Find out if your mortgage rate is too high, NOW. Free Search"

"Instantaneously Attract Women"

"Confirming your FREE PDA"

These are the subject lines from unsolicited e-mail messages I received in a one hour period. Once a mild annoyance, unsolicited bulk e-mail–also known as spam–comprises close to half of all message traffic on the Internet. Spam is a big problem for several reasons: Spam imposes a significant burden on Internet Service Providers; the resulting costs ultimately trickle down to individual subscribers. Spam supports pyramid schemes, dubious health products and remedies, and other fraudulent activities. Spam exposes minors to innappropriate material. Spam results in a loss of

productivity; the cumulative costs add up quickly when all e-mail users spend a few minutes a day dealing with and disposing of spam.

Defenders of spam draw comparisons with print catalogs. Catalog companies use regular mail to send catalogs to potential new customers. Why is it OK to send unsolicited catalogs, and not OK to send unsolicited e-mails? One key difference is cost - spammers can send millions of messages for little or no cost. Paul Graham, an anti-spam activist, proposed the following thought experiment[1]:

> "Suppose instead of getting a couple print catalogs a day, you got a hundred. Suppose that they were handed to you in person by couriers who arrived at random times all through the day. And finally suppose they were all just ugly photocopied sheets advertising pornography, diet drugs, and mortgages."

Given the central role of e-mail in our professional lives, the analogy with spam is not so farfetched.

The problem of spam is attracting increasing media attention and a sort of War on Spam has begun. In what follows I describe the critical role Statistics plays in this War.

2 Spam Filters

Systemic progress against spam may require complex worldwide legislation. Individual users, however, can protect themselves with so-called "spam filters." A spam filter is a computer program that scans incoming e-mail and sends likely spam to a special spam folder. Spam filters can make two kinds of mistakes. A filter might fail to detect that a particular message is spam and incorrectly let it through to the inbox (a false negative). Equally a filter might incorrectly route a perfectly good message to the spam folder (a false positive). An effective spam filter will have a low false negative rate, and perhaps more criticially, a low false positive rate.

[1] http://www.paulgraham.com

Early spam filters used hand-crafted rules to identify spam. Here are the antecedents for some typical spam rules:

- <Subject> contains "FREE" in CAPS
- <Body> contains "University Diploma"
- <Body> contain an entire line in CAPS
- <From:> starts with numbers

This approach can produce effective spam filters. In fact, merely looking for the word "FREE" for instance, catches about 60% of the e-mails in my collection of spam, with less than 1% false positives. However, the hand-crafting approach suffers from two significant drawbacks. First, creating rules is a tedious, expensive, and error-prone process. Second, humans are unlikely to spot more obscure indicators of spam such as the presence of particular HTML tags or the use of certain font colors.

In the last couple of years, the statistical approach to constructing spam filters has emerged as a method-of-choice and provides the core technology for several leading commercial spam filters. The statistical approach starts with a collection of e-mail messages that have been hand-labeled as spam or not-spam; creating these "training data" is the hard part. Next, a statistical algorithm scans the collection and identifies discriminating features along with associated weights. Finally, the trained algorithm scans new e-mail messages as they arrive and automatically labels each one as spam or not-spam.

Spam filter builders use many different statistical algorithms including decision trees, nearest-neighbor methods, and support vector machines. In what follows, we describe "Naive Bayes," a straightforward and popular approach, that has achieved excellent results on some standard test collections.

3 Representing E-mail for Statistical Algorithms

All statistical algorithms for spam filtering begin with a vector representation of individual e-mail messages. Researchers have studied many different representations

but most applications use the so-called "bag-of-words" representation. Bag-of-words represents each e-mail message by a "term vector." The length of the term vector is the number of distinct words in all the e-mail messages in the training data. The entry for a particular word in the term vector for a particular e-mail message is usually the number of occurences of the word in the e-mail message. For example, Table 1 presents toy training data comprising four e-mail messages. These data contain ten distinct words: the, quick, brown, fox, rabbit, ran, and, run, at, and rest. Table 2 shows the corresponding set of term vectors.

#	**Message**	**Spam**
1	the quick brown fox	no
2	the quick rabbit ran and ran	yes
3	rabbit run run run	no
4	rabbit at rest	yes

Table 1: Training data comprising four labeled e-mail messages.

#	**and**	**at**	**brown**	**fox**	**quick**	**rabbit**	**ran**	**rest**	**run**	**the**
1	0	0	1	1	1	0	0	0	0	1
2	1	0	0	0	1	1	2	0	0	1
3	0	0	0	0	0	1	0	0	3	0
4	0	1	0	0	0	1	0	1	0	0

Table 2: Term vectors corresponding to the training data.

If the training data comprise thousands of e-mail messages, the number of distinct words often exceeds 10,000. Two simple strategies to reduce the size of the term vector somewhat are to remove "stop words" (words like and, of, the, etc.) and to reduce words to their root form, a process known as stemming (so, for example, "ran" and "run" reduce to "run"). Table 3 shows the reduced term vectors along with the spam label.

The bag-of-words representation has some limitations. In particular, this representation contains no information about the order in which the words appear in the e-mail messages!

	X_1	X_2	X_3	X_4	X_5	X_6	Y
#	**brown**	**fox**	**quick**	**rabbit**	**rest**	**run**	**Spam**
1	1	1	1	0	0	0	0
2	0	0	1	1	0	2	1
3	0	0	0	1	0	3	0
4	0	0	0	1	1	0	1

Table 3: Term vectors after stemming and stopword removal with the Spam label, coded as 0=no, 1=yes.

4 Naive Bayes for Spam

Let $X = (X_1, \ldots, X_d)$ denote the term vector for a random e-mail message, where d is the number of distinct words in the training data, after stemming and stopword removal. Let Y denote the corresponding spam label. The Naive Bayes model seeks to build a model for:

$$Pr(Y = 1|X_1 = x_1, \ldots, X_d = x_d).$$

From Bayes theorem, we have:

$$Pr(Y = 1|X_1 = x_1, \ldots, X_d = x_d) = \frac{Pr(Y = 1) \times Pr(X_1 = x_1, \ldots, X_d = x_d|Y = 1)}{Pr(X_1 = x_1, \ldots, X_d = x_d)}, \tag{1}$$

or, on the log odds scale,

$$\log \frac{Pr(Y = 1|X_1 = x_1, \ldots, X_d = x_d)}{Pr(Y = 0|X_1 = x_1, \ldots, X_d = x_d)} = \log \frac{Pr(Y = 1)}{Pr(Y = 0)} + \log \frac{Pr(X_1 = x_1, \ldots, X_d = x_d|Y = 1)}{Pr(X_1 = x_1, \ldots, X_d = x_d|Y = 0)}. \tag{2}$$

The log odds scale avoids the normalizing constant in the denominator of the right hand side of (1).

The first term on the right hand side of (2) involves the prior probability that a random message is spam. The second term on the right hand side of (2) involves two conditional probabilities, namely the conditional probability of a term vector given that the term vector's message is spam, and the conditional probability of a term vector given that the term vector's message is not spam. This is problematic insofar as it involves the joint probability distribution of d random variables, $X_1, \ldots, X_d$,

where d is potentially large. The key assumption of the Naive Bayes model is that these random variables are conditionally independent given Y. That is:

$$Pr(X_1 = x_1, \ldots, X_d = x_d | Y = 1) = \prod_{i=1}^{d} Pr(X_i = x_i | Y = 1)$$

and

$$Pr(X_1 = x_1, \ldots, X_d = x_d | Y = 0) = \prod_{i=1}^{d} Pr(X_i = x_i | Y = 0)$$

leading to:

$$\log \frac{Pr(Y = 1 | X_1 = x_1, \ldots, X_d = x_d)}{Pr(Y = 0 | X_1 = x_1, \ldots, X_d = x_d)} = \log \frac{Pr(Y = 1)}{Pr(Y = 0)} + \sum_{i=1}^{d} \log \frac{Pr(X_i = x_i | Y = 1)}{Pr(X_i = x_i | Y = 0)}. \tag{3}$$

This independence assumption is unlikely to reflect reality. For instance, in my collection of of spam messages, the word "mortgage" is strongly associated with the word "interest," a violation of the conditional independence assumption.

Nonetheless, the independence assumption provides a drastic reduction in the number of distinct probabilities that we need to estimate from the training data and yet often performs well in practice.

4.1 Binary Naive Bayes

For now, let us simplify the representation to binary term vectors. That is, let $X_i = 1$ if word i is present one or more times in a message and $X_i = 0$ otherwise, $i = 1, \ldots, d$. Without the Naive Bayes assumption, the model needs estimates for 2×2^d parameters for the term vectors (one probability for every possible term vector for both spam and not-spam). With the Naive Bayes assumption, the model needs $2 \times d$ estimates. The extreme nature of the core independence assumption earns the model its "Naive" name. Some earlier literature refers to the model as "Idiot's Bayes."

There remains the issue of how to estimate the model's probabilities from the training data. Specifically, sticking with the binary representation, the model needs estimates for $\log \frac{Pr(Y=1)}{Pr(Y=0)}$ and for $\log \frac{Pr(X_i=1|Y=1)}{Pr(X_i=1|Y=0)}$, $i = 1, \ldots, d$ (the literature refers to these latter terms as "weights of evidence"). The obvious approach is to estimate each probability

by its observed fraction in the training data and plug these estimated probabilities into the expressions for the log odds. For the example in Table 3 this leads to, for instance,

$$\log \frac{\widehat{Pr}(Y=1)}{\widehat{Pr}(Y=0)} = \log \frac{2/4}{2/4} = 0$$

and

$$\log \frac{\widehat{Pr}(X_3=1|Y=1)}{\widehat{Pr}(X_3=1|Y=0)} = \log \frac{2/2}{1/2} = \log 2$$

where $\widehat{Pr}$ denotes an estimated probability. These correspond to the maximum likelihood estimates of the probabilities.

This approach does not provide unbiased estimates for the weights of evidence, and also runs into practical difficulties when it estimates some probabilities as zero. For example, in the example, $\widehat{Pr}(X_5 = 1|Y = 0) = 0$, leading to an undefined estimate for the corresponding weight of evidence. The standard solution to this problem is to "smooth" the estimates by adding a small positive constant to each numerator and denominator of each probability estimate. The Appendix describes one particular rationale for choosing the value of the small positive constant, focusing on reducing bias (and suggests setting the constant to 0.5).

Table 4 shows the resulting estimated weights of evidence for the example.

	X_1	X_2	X_3	X_4	X_5	X_6
	brown	**fox**	**quick**	**rabbit**	**rest**	**run**
Term Present	-1.10	-1.10	0	0.51	1.10	0
Term Absent	0.51	0.51	0	-1.10	-0.51	0

Table 4: Estimated Weights of evidence for the example.

Now suppose a new e-mail message arrives:

> The quick rabbit rests

Does the Naive Bayes model predict that this message is spam? After stemming and stopword removal, Table 5 shows the term vector for this message along with the associated weights of evidence.

	X_1	X_2	X_3	X_4	X_5	X_6
	brown	**fox**	**quick**	**rabbit**	**rest**	**run**
Term Vector	0	0	1	1	1	0
Weight of Evidence	0.51	0.51	0	0.51	1.10	0

Table 5: Making predictions for a new e-mail message.

According to the model, the log-odds that this message is spam is the prior log odds (zero in this case) plus the sum of the relevant weights of evidence: $0.51 + 0.51 + 0 + 0.51 + 1.10 + 0 = 2.63$. This corresponds to a predicted probability of 0.93. The spam filter builder chooses a threshold and then compares this predicted probability to the threshold. If the probability exceeds the threshold, the filter routes the message to the spam folder. Otherwise the filter lets the message through to the inbox. The choice of the threshold must reflect the builder's relative concerns about the two kinds of errors mentioned earlier. A low threshold may result in a higher false positive rate - more legitimate messages ending up in the spam folder. A high threshold may result in a higher false negative rate - more spam messages ending up in the inbox. Most commercial spam filters allow the user to select the threshold.

4.2 Multinomial Naive Bayes

The binary representation ignores the number of occurences of each term in a message, yet these term frequencies may provide useful useful clues for identifying spam. Incorporation of term frequencies requires a model for $Pr(X_i = x_i), i = 1, \ldots, d, x_i \in \{0, 1, 2, \ldots\}$. One can apply standard models for non-negative integers such as Poisson and Negative Binomial in this context but the literature reports mixed success (see, for example, Eyheramendy *et al.*, 2003, and the classic text by Mosteller and Wallace, 1984). A number of researchers have reported success with the so-called multinomial Naive Bayes model. This model assumes that message length is independent of spam status. We refer the interested reader to Eyheramendy *et al.* (2003), Lewis (1998), and McCallum and Nigam (1998) for details.

4.3 Naive Bayes Effectiveness

The Naive Bayes model is simple to implement and scales well to large-scale training data. The literature on spam filtering reports a number of experiments where Naive Bayes delivers competitive false positive and false negative rates.

Sahami *et al.* (1998) conducted experiments with a corpus of 1,789 e-mail messages, almost 90% of which were spam. They randomly split this corpus into 1,538 training messages and 251 testing message. A Naive Bayes classifier, as described above, achieved a false negative rate of 12% and a false positive rate of 3%. Extending the term vector representation to include hand-crafted phrases and some domain-specific features reduced the false negative rate to 4% and the false positive rate to 0%.

Androutsopoulos *et al.* (2000) report experiments using the "Ling-spam" corpus of 2,893 e-mail messages, 481 of which are spam. They reported a false positive rate of 1% computed via 10-fold cross-validation. Their Naive Bayes algorithm outperformed a set of rules built into a widely used commerical e-mail client.

Carreras and Marquez (2001) on yet another corpus (PU1) report a Naive Bayes false positive rate of 5%. Neither Androutsopoulos *et al.* nor Carreras and Marquez report false negative rates.

Newer statistical algorithms such as regularized logistic regression, support vector machines, and boosted decision trees can achieve higher effectiveness. Carreras and Marquez (2001), for example, reported experiments in which boosted decision trees out-performed Naive Bayes by a few percentage points on the Ling-spam corpus.

5 Discussion: Beyond Spam Filtering

Spam filtering has provided the primary focus for this chapter and we have highlighted the central role of Statistics. In fact spam filtering is one particular example of the broader topic of "text categorization." Text categorization algorithms assign texts to predefined categories. The study of such algorithms has a rich history dating back at least forty years. In the last decade or so, the statistical approach has dominated the literature. The essential idea, as with statistical spam filtering, is to infer a text

categorization algorithm from a set of labeled documents, i.e., documents with known category assignments, where a feature vector represents the documents. Sebastiani (2002) provides an overview.

Applications of statistical text categorization include:

- Building web directories like the dmoz open directory and Yahoo!
- Routing incoming customer service e-mail
- Identifying "interesting" new stories for intelligence analysts
- Classifying financial news stories as "significant" or "insignificant"

While Naive Bayes is rarely the top-performing algorithm in any of these applications, it invariably provides reasonable effectiveness with minimal computational effort.

6 Appendix: Smoothing Estimated Weights of Evidence to Reduce Bias

We consider a binomial sampling model for a single term. Let N_1 denote the number of spam e-mail messages in the training data and N_2 the number of not-spam messages. Let R_1 denote the number of spam messages containing the term and R_2 the number of not-spam messages containing the term (see Table 6). The binomial sampling model assumes that N_i is fixed, and that R_i is binomially distributed with parameters N_i and θ_i, $i = 1, 2$. Let $W(S:T) = \log \frac{\theta_1}{\theta_2}$ denote the target weight of evidence and $\widehat{W}(S:T)$ the corresponding estimated weight of evidence. As in Spiegelhalter and Knill-Jones (1984), we want to choose $a, b, c, d \in \Re^+$ such that:

$$E(\widehat{W}(S:T)) = E(\log \frac{R_1 + a}{N_1 + b} / \frac{R_2 + c}{N_2 + d})$$

is as close as possible to $W(S:T)$. That is, we want to minimize the bias of $\widehat{W}(S:T)$.

Following Cox (1970, p.33) let:

$$R_i = N_i\theta_i + U_i\sqrt{N_i}, i = 1, 2,$$

	Spam	**Not Spam**
Term Present	R_1	R_2
Term Absent	$N_1 - R_1$	$N_2 - R_2$

Table 6: Frequency counts required for estimating a single weight of evidence.

and thus:

$$\begin{aligned} E(U_i) &= 0 \\ E(U_i^2) &= \theta_i(1-\theta_i) \\ E(U_i^3) &= N_i^{-1/2}\theta_i(1-\theta_i)(1-2\theta_i) \\ E(U_i^4) &= 3\theta_i^2(1-\theta_i)^2 + N_i^{-1}\theta_i(1-\theta_i)(1-6\theta_i(1-\theta_i)). \end{aligned}$$

We consider:

$$\begin{aligned} & \widehat{W}(S:T) - W(S:T) \\ = \; & \log\frac{R_1+a}{N_1+b} - \log\frac{R_2+c}{N_2+d} - \log\frac{\theta_1}{\theta_2} \\ = \; & \log\frac{N_1\theta_1 + U_1\sqrt{N}_1 + a}{N_1+b} - \log\frac{N_2\theta_2 + U_2\sqrt{N}_2 + c}{N_2+d} - \log\frac{\theta_1}{\theta_2}. \\ = \; & \log\left(1 + \frac{U_1}{\theta_1\sqrt{N_1}} + \frac{a}{N_1\theta_1} + \frac{d}{N_2} + \frac{dU_1}{\theta_1 N_2\sqrt{N_1}} + \frac{ad}{\theta_1 N_1 N_2}\right) \\ & -\log\left(1 + \frac{U_2}{\theta_2\sqrt{N_2}} + \frac{c}{N_2\theta_2} + \frac{b}{N_1} + \frac{bU_2}{\theta_2 N_1\sqrt{N_2}} + \frac{bc}{\theta_2 N_1 N_2}\right) \end{aligned}$$

Expanding this expression, taking expectations, and ignoring terms of order N^{-2} and lower, where $N = \min(N_1, N_2)$ we have:

$$\begin{aligned} & E\left(\widehat{W}(S:T) - W(S:T)\right) \\ = \; & \frac{a-1/2}{\theta_1 N_1} - \frac{c-1/2}{\theta_2 N_2} - \frac{b-1/2}{N_1} + \frac{d-1/2}{N_2} + O(N^{-2}). \end{aligned}$$

The absolute value of this expression is clearly minimized when $a = b = c = d = 1/2$. With these values, the bias is given by:

$$-\frac{1-\theta_1^2}{24\theta_1^2 N_1^2} + \frac{1-\theta_2^2}{24\theta_2^2 N_2^2} + O(N^{-3}).$$

It is straightforward but tedious to show that setting $a = b = c = d = 1/2$ also minimizes the bias for the Poisson and multinomial sampling schemes. Curiously, for a binomial sampling that conditions on the row totals of Table 6 instead of the column totals, simple expressions for a, b, c, and d do not seem to exist.

References

Androutsopoulos, G. Paliouras, V. Karkaletsis, G. Sakkis, C. Spyropoulos, and P. Stamatopoulos (2000). Learning to filter spam e-mail: A comparison of a naive bayesian and a memory-based approach. In: *Workshop on Machine Learning and Textual Information Access, 4th European Conference on Principles and Practice of Knowledge Discovery in Databases*.

Carreras, X. and Marquez, L. (2001). Boosting trees for anti-spam email filtering. In: *Proceedings of RANLP-01, Jth International Conference on Recent Advances in Natural Language Processing*. http://citeseer.nj.nec.com/article/carreras01boosting.html

Cox, D.R. (1970). *The Analysis of Binary Data*. Chapman and Hall, London.

Eyheramendy, S., Lewis, D.D., and Madigan, D. (2003). On the naive bayes model for text classification. In: *Proceedings of The Ninth International Workshop on Artificial Intelligence and Statistics*, C.M. Bishop and B.J. Frey (Editors), 332-339.

Lewis, D.D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. In: *ECML'98, The Tenth European Conference on Machine Learning*, 4–15.

McCallum, A. and Nigam, K. (1998). A comparison of event models for naive Bayes text classification. In: *Proceedings of the AAAI-98 Workshop on Learning for Text Categorization*. AAAI Press.

Mosteller, F. and Wallace, D.L. (1984). *Applied Bayesian and Classical Inference (Second Edition)*. Springer-Verlag, New York.

Sahami, M., Dumais, S., Heckerman, D., and Horvitz, E. (1998). A Bayesian Approach to Filtering Junk E-Mail. In: Learning for Text Categorization: Papers from

the 1998 Workshop. AAAI Technical Report WS-98-05.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, **34**, 1–47.

Spiegelhalter, D.J. and Knill-Jones, R.P. (1984). Statistical and knowledge based approaches to clinical decision support systems, with an application to gastroenterology (with discussion). *Journal of the Royal Statistical Society (Series A)*, **147**, 35-77.