International Journal of Engineering, Science and Mathematics

Vol.6Issue 6, October2017,
ISSN: 2320-0294 Impact Factor: 6.765
Journal Homepage: http://www.ijesm.co.in, Email: ijesmj@gmail.com Double-
Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gage as well as in Cabell's Directories of Publishing Opportunities, U.S.A

Preliminary Investigations on the spam filtering using statistical classification techniques

Dr.A.V.Rajeswari M.Sc, M.Phil, Ph.D

Abstract (12pt)

A study on the thereotical and application of the statistical filtering techniques for the spam classification problems has been conducted and its results are presented. The research methodology applied in this study starts with building the dataset, correcting errors in the datasetand discusses the techniques to compute the probability of tokens in the dataset and the statistical application of the token values. The analysis shown is used to depict that statistical filtering is better than heuristic-based filtering because the former approach gives specific information about making a decision. All the four popular techniques in use today namely- Bayesian Combination by Paul Graham, Bayesian Combination by Brian Burton, Robinson's Geometric Mean Test and Fisher-Robinson's Inverse Chi- Square and their merits and demerits have been discussed in detail.

Keywords:

Spam Classfication;
Statistical Filtering;
Bayes Classifier;
Machine Learning;

Author correspondence:

Dr.A.V.Rajeswari M.Sc,M.Phil,Ph.D,
Professor, Department of Physics,
J.M.J College for Women, Tenali, Andhra Pradesh, India-522202

1. Introduction

In this publication, the mathematical world around the foundational concepts of spam filtering and the statistical combination used to determine whether an e-mail is spam or not are investigated and presented. Some statistical combinations use the Bayes Theorem and others use a different combination of the data collected. Server- side email filters, such as DSPAM, SpamAssassin, SpamBayes, Bogofilter and ASSP, make use of Bayesian spam filtering techniques, and the functionality is sometimes embedded within mail server software itself.

Statistical filtering is better compared to heuristic-based filtering[1]. Statistical filtering involves measuring probability. One of the great benefits of statistical language classification is that you're actually measuring something, as opposed to guessing scores. Heuristic-based filters, on the other hand, assign a score to each feature, which is really just an arbitrary number without mathematical significance. The user doesn't know what it means, but worse still, neither does the developer of the filter. It's very difficult for a systems administrator to interpret the meaning of a score, since it doesn't relate to any real measurement. Statistical classifiers, on the other hand, measure something very specific—mathematical probability. The idea that "there is an X percent chance that this message is spam" is a lot easier to comprehend than "this message scored a 3.52." It also gives filter authors and systems administrators a look into the thought process of the filter, so they can better ensure its correct operation.

2. Research Method

The methodology investigated employes the following four stepsfour steps in spam filtering:

1. Building thedataset
2. Correcting errors in thedataset
3. Use techniques to compute the probability of tokens in thedataset.
4. Statistical Combination of the token values to determine whether an e-mail is spam ornot.

Each of the four steps are discussed in detailbelow.

2.1 Building a Historical Dataset

The historical dataset includes the memories of previous messages and meaningful words, phrases, and combinations to watch for in future emails. In order to reach a decision about a particular message, the historical dataset must first be built.

The characteristics used in statistical filtering are generated based on the content of the message; therefore, no matter how much spam changes over time, it will always have some unique content we can use to identify it. As new words begin to appear in spam, they're automatically added to the filter's memory. Building and maintaining the dataset, is done either by feeding the filter a collection of saved messages (a corpus) or from scratch.

2.1.1.Corpus Feeding

Corpus feeding involves building a historical dataset by analyzing a collection *(corpus)* of saved mail. We start off with a small collection of both spam and legitimate mail that has been presorted and classified (by a human). The filter will process each message and build a table containing the occurrence of every token (how many times it has appeared in both spam and legitimate mail). When we're finished, we end up with a very large collection of tokens and their corresponding counters, representing their appearance in the corpus.

The following example shows a small slice of preliminary data from such an exercise.

Feature	**Appearances/Spam**	**Appearances/Ham**
fun	19	9
girlfriend	4	0
mariners	0	7
tell	8	30
the	96	48
vehicle	11	3
viagra	20	1

As you can see, the word "the" has appeared in 96 messages that were considered spam and in 48 legitimate messages. In a perfectly balanced email corpus, this would suggest that the word is more likely to be present in spam, but most corpora are unbalanced. In order to determine the true disposition of each token, we'll also need to know how much spam and nonspam is in thecorpus.

Two additional counters are created in the dataset for this purpose. For simplicity's sake, our sample corpus will comprise twice as much spam as nonspam (which is usually about right in the real world):

Total Spam	224
Total Legitimate Mail	112

Anywhere from several hundred to a few thousand messages are processed in this fashion, building one large table at the end of the process. Once complete, the table is stored as our dataset, on disk, so that it can be accessed every time an email arrives.

2.1.2. Starting from Scratch

It's not entirely necessary to build the dataset from a historical corpus of mail. In fact, many spam filters prefer to start from scratch and build the dataset as email is received. The philosophy is that building from

an empty corpus prevents the filter from getting hung up on stale data, which can hurt long-term accuracy. Starting from scratch means that the filter knows nothing until a message is trained. Once a message is trained, either the tokens are added to the dataset or their counters are incremented if they already exist. For example, if the word "calculus" is already present in the dataset, either its innocent counter or its spam counter will be incremented whenever a message is trained containing that word. If the word is not present in an email, it's left alone in the dataset.

Depending on the training mode used, the filter will add information to the dataset at different times. The training mode can also dictate which specific tokens are incremented. Since we also keep track of message totals, the total number of messages is also updated whenever a message is trained, so that we know the size of the dataset at any point in time. Building from an empty dataset generally takes time and messages. Even with many messages over a short period of time, certain areas of a user's email behavior may not be well represented, and infrequent emails such as monthly newsletters may end up in the bit bucket unintentionally. However, accuracy rises over time using this approach, eventually reaching a plateau once enough training has takenplace.

2.2.Correcting Errors

As the historical dataset is built, training errors are likely to occur while the filter learns. When this happens, it's important to reverse whatever information has been previously recorded about themessage.

If the message has been previously learned, it must first be unlearned. This is done by decrementing the counters under which the message was originally classified. It can then be learned by incrementing the correct counters. This is referred to as *retraining* the message. For example, if the spam filter originally thought a message containing the word "free" was legitimate, the data may initially look like the following.

Token	**Nonspam Hits**	**Spam Hits**
free	10	32
Total Spam:	65	
Total Nonspam:	20	

Once the message is submitted for retraining, the data will be reversed.

Token	**Nonspam Hits**	**Spam Hits**
free	9	33
Total Spam:	66	
Total Nonspam:	19	

When relearning a message, it's important to relearn all of the original tokens used in the original (erroneous) classification. This includes the original full headers of the message, message body, and all content. Regardless of how the original data is retained, the retraining process will usually decrement one counter and increment the other. If the message is a spam that was classified erroneously, the innocent counters of all trained tokens pertaining to the message are decremented and the spam counters are incremented. This is essentially how the filter learns. As the mistakes are corrected, the dataset reflects these new values, which then directly affect the disposition of eachtoken.

2.2.1. The Tokenizer and Calculating Token Values

When a message is processed, the tokenizer breaks it down into very small components (tokens) and assigns each token a value. A *token value* is a numeric representation of the token's disposition (good or evil), based on its appearance in previous emails. For example, if a word has appeared primarily in spam, it would be assigned a value reflecting a very strong probability of being spam (such as 99percent).[2]

Consider the following test corpus with exactly twice the amount of spam as legitimate mail.

Total Spam:	224
Total Legitimate Mail:	112

If we look at the number of occurrences of the word "the" in the corpus, we see that its appearance in spam is proportionately identical to its appearance in legitimate mail. That is, it appears just as often in spams as it does in legitimate messages. In our test corpus, the word "the" has appeared in 96 spams and 48 nonspams.

Feature	Appearances/Spam	Appearances/Ham
the	96	48

Common sense tells us that this word should be assigned a neutral value of 50 percent, since it appears in spam and nonspam equally. Now all we need is a mathematical formula that will give us the same reasonable results.

There are two popular mathematical formulas to choose from: Graham's and Gary Robinson's. Some filters let you choose which of these two techniques to use which we will discuss next. Selecting the correct option may be an exercise in trial and error, and may depend on the aggressiveness of your anti-spam policy. The first technique (Graham's) has been shown to be more aggressive in preventing spam, while Robinson's is more aggressive in preventing falsepositives.

2.2.2.Graham's Technique for Computing Word Values

Graham provides a very simple formula for determining the probability that a given token is an identifier of spam. His approach takes into account the fact that many users will have an unbalanced collection of spam and legitimate email and computes a probability that factors in the total number of messages in each corpus.

In the following formula, *SH* and *IH* represent the total number of appearances in spam and innocent email for the token being computed. *TS* and *TI* represent the total number of spam and innocent messages in the user's corpus.

$$P = \frac{(SH)/(TS)}{((SH)/(TS)) + ((IH)/(TI))}$$

Using the following formula for "the" token

P= [96/224]/[(96/224)+(48/112)]

This calculation leaves us with *P*, a number between 0.0 and 1.0, where 0.5 is neutral (and the result of calculating "the" in our example). Tokens with a probability higher than a neutral 0.5 are considered guilty (or "spammy"), while tokens with a lower probability are considered hammy (or innocent). Tokens with a stronger disposition will be computed with a value closer to 0.0 or 1.0.

2.2.3.Robinson's Technique for Combining Word Values

Robinson improved on Graham's technique by factoring in a way to calibrate the value of a token probability based on the amount of historical data available for each token. Robinson believes that tokens with very few historical occurrences should have a weaker confidence than ones with many data points, and suggested a new way of calculating individual token probabilities.

Robinson adds onto Graham's existing calculation by applying a confidence factor. The resulting formula weakens the value of a token if it has very few data points. His function uses three variables:

N	The total number of times a token has appeared in the dataset—in either spam or legitimate mail
X	The assumed value of a token when *N*=0. A good starting value is 0.5000
S	A constant that is tuned for performance, with a reasonable default value of 1

"In cases where there is very little information to work with," Robinson states, "we have some data, but we don't really have enough to be confident. In reality we should do something when *N* is any low number—0 being the most extreme case."

We first calculate the probability of a word using Graham's techniques. Call this *p*(*w*) for word *w*, then calculate as follows.

$$F(W) = \frac{SX + N(P(W))}{S + N}$$

Robinson's approach yields results similar to Graham's, except that it is calibrated based only on the information available for a specific token. By taking into account cases in which very little data has been collected for a particular token, Robinson's technique yields a slight improvement in accuracy for many types of hard-to-classify messages. It is particularly useful in improving the identification of legitimate mail, thereby reducing falsepositives.

2.2.4.Single-Corpus Tokens

Some tokens are such strong indicators that they will appear in only one particular corpus of mail (spam or nonspam). These are referred to as *single-corpus tokens*. One caveat when assigning values to tokens is that no token should ever have a probability of 100 percent or 0 percent, because these are absolute probabilities. For example, because email evolves, there is no guarantee that messages containing the word "offer" have a 100 percent chance of beingspam.

To solve this problem, a static value is assigned to single-corpus tokens. If a token appears in only one corpus, we assign it a hard-coded value. For example, most filters will assign tokens that appear only in our spam corpus a probability of 0.9900, and they will assign a probability of 0.0100 to tokens appearing only in legitimate mail. If Robinson's technique is being used, this value is assigned to the $p(w)$ value used in his formula.

When we apply these calculations to an example test corpus we end up with a list of probabilities for each token

Feature	**Spam**	**Ham**	**Probability**
fun	19	9	0.5135
girlfriend	4	0	0.9900
mariners	0	7	0.0100
tell	8	30	0.1176
the	96	48	0.5000
vehicle	11	3	0.6470
viagra	20	1	0.9090

2.2.5A Biased Filter

Graham notes in his research that by doubling the occurrence of tokens in legitimate messages we can help to prevent false positives. Since spam filters typically err on the side of caution, Graham chooses to classify a message as spam only if there is an overwhelming amount of supporting data. The following is a modified version of Graham's approach that multiplies the number of innocent occurrences for a token by 2.[3]

$$P = \frac{(SH)/(TS)}{((SH)/(TS)) + ((IH)/(TI))}$$

Many spam filters allow bias to be turned on or off. In many scenarios, applying bias has resulted in up to 50 percent fewer false positives, with only slight drops in the identification of spam.

2.2.6. Hapaxes

Another thing to consider when assigning values to tokens is how we handle new tokens that we've never seen before, or for which we haven't collected enough data. These tokens are referred to as *hapaxes*.

Filters based on Graham's original outline implement a threshold of occurrence—a minimum number of appearances in historical email before the filter will assign a calculated probability to the token. If the token doesn't meet the threshold, it will be assigned a relatively neutral value, such as 0.4000 or 0.5000, known as the *hapaxial value*, which is applied to the Graham-derived probability of the token. Therefore,

if we were to use a threshold of five occurrences, a token would need to appear in either three legitimate emails (for a total of six occurrences), five spam, or a combination between the two in order to be assigned a real tokenvalue.

2.2.7.Final Product

Once we apply these concepts, including a bias toward innocent email and an occurrence threshold of five, the result is a table of data that is slightly different from our original example. Here are the probabilities for our tokens after applying all of our design rules.

Feature	Spam	Ham	Probability
fun	19	9	0.3454
girlfriend	4	0	0.4000
mariners	0	7	0.0100
tell	8	30	0.0625
the	96	48	0.3333
vehicle	11	3	0.4782
viagra	20	1	0.8333

Although it may not be obvious, our final table identifies not only which characteristics are present in email, but also which are the most useful to us. If we look at the values in the table, we can see which tokens have the strongest disposition of spam (closest to 1.0) and nonspam (closest to0.0).

3. Results and Analysis

Let's look at an example to see how the computer uses information to determine whether it is a spam or legitimate e-mail. Here's an example:

```
From: "Julie Ellison" <gcgbswamlgqy@sbcglobal.net>
Reply-To: "Julie Ellison"
<gcgbswamlgqy@sbcglobal.net> Subject: Don't Pay For
Name Brand Drugs

Date: Sun, 11 Apr 2004 10:21:05 +0600

Content-Type: text/plain

CANADIAN GENERICS NOW HAS VALIUM!

Know where to find discounted Prescriptions? Buy your
      personal prescription drugs on the internet and $ave!
Allergies, Weight Loss, Muscle and Pain Relief Men and Womens Health,
heartburn, migraines, Impotence Get meds from Canada here:
http://$scribbleheterozygous.zzstrore2.com/gp/default.asp?id=gm03Order
Some HERE
```

Our decision matrix is populated with the 15 most interesting tokens in this message.

Spam	Innocent	Probability	Token
35	0	0.999900	NOW
27	0	0.999900	drugs
32	0	0.999900	meds
37	0	0.999900	Loss
76	0	0.999900	Date*Apr
50	0	0.999900	here
25	0	0.999900	Relief
28	0	0.999900	Weight
71	1	0.938004	Url*http

129	2	0.932180	Content-Type*text
109	3	0.885617	and
13	19	0.127251	personal
28	1	0.856461	HERE
27	1	0.851932	Url*gp
23	1	0.830545	Pain

To the human eye, it is fairly easy to tell whether the message is spam or not simply by glancing at the token values, but computers require logic. In many cases there may be a lot of variation in the probabilities of the tokens.

Given all of these different token probabilities, we need a way to combine them. We use statistical combination to solve a probability-based decision matrix. To implement it, we look at each token in our decision matrix as an independent test. Statistical combination provides a way to combine the results of many tests to create a single result. Some filters are based on only one combination technique, while others support a wide range of options that can be enabled or disabled. There is no one solution but choosing a particular combination depends on the environment.

I'll discuss about the four popular approaches in use today.

3.1 Bayesian Combination (Paul Graham)[4]

Bayesian combination uses Bayes' Theorem to combine statistics. We use Bayesian logic to combine individual probabilities to produce a single outcome.

$$\frac{AB}{AB + (1-A)(1-B)}$$

Graham's approach uses a decision matrix with a window size of 15 tokens (the 15 most interesting tokens). Bayes' Theorem allows multiple probabilities to be combined in thisfashion:

$$\frac{(0.93)(0.67)}{(0.93)(0.67) + (1-0.93)(1-0.67)}$$

The product of these 15 probabilities is divided by itself and its inverse. For example,

$$\frac{ABC\ldots N}{ABC\ldots N + (1-A)(1-B)(1-C)\ldots(1-N)}$$

This gives us the result 0.9642, or roughly 96 percent. When applied to a set of 15 probabilities, Bayes' Theorem expands to support any number of factors. Although 15 tokens are most commonly used in a calculation, some implementations use more. Some even use every single token in the message.

This combination usually results in a very extreme final value—either very close to 0.0 or very close to 1.0—representing a probability (between 0 percent and 100 percent). Graham's implementation assumes that any result that is greater than or equal to 0.90 is an indicator of spam and anything less is an indicator of legitimate email. Since the results are rather extreme, it is very rare to find many results lurking in the middle. This means that there is usually very little gray area for uncertainty. This approach is very effective at filtering out most types of spam. Because the results are generally very extreme, there is very little uncertainty.

3.2.Bayesian Combination (Brian Burton)

The author of SpamProbe, Brian Burton, analyzed Graham's approach and made several enhancements. He extended the window size of the decision matrix to 27 elements and allowed for a single token to populate two slots in the matrix if it appears at least twice in a message. Burton discovered a basic problem in Graham's approach that caused filters to frequently require a "tie-breaker" when there was too much strong data to fit in the decision matrix. Depending on how the filter processed the data, some messages hinged on mere randomness—the first side to fill up that eighth slot would win.

Burton also found that it was possible to have too little information when processing smaller messages. The decision matrix would be starved, populated with unimportant information. By allowing tokens appearing multiple times within an email to take up two slots in the matrix, Burton essentially gave a heavier weight to suchtokens.

This resulted in higher levels of accuracy as compared to Graham's standard approach. In our example, the decision matrix to support an alternative Bayesian algorithm would look like the following:

Spam	Innocent	Probability	Token
50	00	0.999900	Url*e
50	00	0.999900	Url*e
50	00	0.999900	order
50	00	0.999900	offer
36	00	0.999900	ONE
50	00	0.999900	Click
52	00	0.999900	here
34	00	0.999900	Be
4	41	0.019655	Precedence*bulk
3	30	0.020136	case
4	27	0.029545	respond
7	41	0.033896	Original
7	41	0.033896	Original
9	41	0.043163	following
8	27	0.057395	Is
8	27	0.057395	Is
9	30	0.058070	sample
73	01	0.937506	Url*http
73	01	0.937506	Url*http
72	01	0.937503	in
12	30	0.075576	sample
4	9	0.083275	permanently
53	1	0.915487	Your
49	1	0.909215	Click
43	1	0.897841	ARE
11	18	0.111035	ago
117	3	0.888531	and

Because tokens appearing more than once in the message take up two slots in the decision matrix, tokens such as "Url*http" now have a heavier "weight" in the final calculation than those that appear only once.Some spam filters implement both versions of the Bayesian combination algorithms to avoid any chance of missing a message due to one algorithm's weakness. Graham's implementation has been proven to work with a majority of common-size messages, while Burton's implementation is stronger with

messageshaving too much or too little data. Combined, these two algorithms generally result in a higher level of accuracy than any one algorithm alone, with no noticeable increase in falsepositives.

3.3.Robinson's Geometric Mean Test

One of the complaints about most implementations of the Bayesian approach is that the result is generally very extreme—hovering right around either 0 percent or 100 percent. This leaves very little room to determine just how guilty the filter considered a message to be.[6]

Robinson's geometric mean test measures both the "spamminess" and "hamminess" of the data in the decision matrix and also provides more granular results ranging between 0 percent and 100 percent. Generally, a result of around 55 percent or higher using Robinson's algorithm is an indicator of spam. The algorithm also has another nice feature in that it is able to express a gray "uncertain" area. It returns a wide range of values (unlike Bayesian combination, which usually returns 0.0 or 1.0), allowing the filter author to consider some ranges as a "maybe"result.

Robinson's approach dramatically changes the way probabilities are combined. His algorithm provides three outputs:

P	The level of spamminess in a message
Q	The level of nonspamminess in a message
S	A combined indicator

$$P = 1 - ((1 - P1)(1 - P2)\ldots(1 - PN))^{(1/N)}$$

$$Q = 1 - ((P1)(P2)\ldots(PN))^{(1/N)}$$

$$S = \frac{1 + \left(\frac{P - Q}{P + Q}\right)}{2}$$

Robinson's geometric mean test has been made obsolete by Fisher- Robinson's inverse chi-square algorithm, discussed next. Although the algorithm is not as accurate as some other techniques, it is excellent for calculating something that Graham's Bayesian approach cannot: confidence.

DSPAM uses Robinson's modified algorithm for determining a confidence factor. The granularity provided by Robinson's algorithm allows the filter to evaluate itself and determine whether or not a decision was solid. If the result shows that the confidence of the decision wasn't very high, the filter can then perform additional calculations or other tasks to try to improve its guess.

3.4. Fisher-Robinson's Inverse Chi-Square

Robinson released the *chi-square* approach in 2003, using the research of another well-known mathematician—Sir Ronald Fisher. Fisher-Robinson's chi-square got its name from the use of Fisher's chi- square distribution of combining individual probabilities. The chi-square algorithm provides the added benefit of being very sensitive to uncertainty. It produces granular results similar to Robinson's geometric mean test, in which the result of a calculation may fall within a wide midrange of values to indicate a level of uncertainty.[5]

The chi-square algorithm's decision matrix is different from that of Bayesian combination in that it includes all tokens within a specific range of probability (usually 0.0 through 0.1 and 0.9 through 1.0) and doesn't require sorting. What's even more different about the chi-square approach is the statistical combination used. Robinson's statistical combination uses the formula below. In this formula, the following variables areused:

N	The total number of tokens used in the decision matrix
F1F2...Fn	The product of the probabilities of all included tokens
H	C-1(-2 LN (F1F2...FN), 2N)
S	C-1(-2 LN ((1.0 – F1)(1.0 – F2)...(1.0 – FN)), 2N)

The result gives us two indicators, one of ham and one of spam. These can be combined to give us a Graham-like indicator having a probability between 0.0 and 1.0.

I = (1 + H –S)/2

In the formula, C^{-1} () represents the inverse chi-square function, which is applied to derive a statistical probability. Chi-square is a very popular statistic because it is relatively easy to calculate and interpret. The pdf of the inverse chi-square function is

$$f(x;\nu) = \frac{2^{-\nu/2}}{\Gamma(\nu/2)} x^{-\nu/2-1} e^{-1/(2x)}$$

4. Conclusion

Spam has increased a lot over the years and efficient techniques have to be developed to combat spam. In this aspect, statistical filtering is better than heuristic-based filtering because the former approach gives specific information about making a decision. The important step in spam filtering is building the dataset of words in e-mails and calculating the probability of each of the words being either spam or innocent. There are several techniques to calculate the probability of the word being spam or not. The technique to be used depends on how aggressive the spam policy is. Once the dataset is built the computer uses this dataset to determine whether the incoming e-mail is spam or not by implementing statistical combination. There are four popular techniques in use today namely- Bayesian Combination by Paul Graham, Bayesian Combination by Brian Burton, Robinson's Geometric Mean Test and Fisher-Robinson's Inverse Chi- Square. Each technique has its advantages and the use of a technique depends on the situation. The first two Bayesian techniques are very effective but leave very little room to determine just how guilty the filter considered a message to be. The last two statistical combinations give an estimate of the confidence factor so that if the result is not quite confident the calculation can be improved to generate results with increased confidence. On the whole, a combination of the above mentioned techniques can be used to effectively filter outspam.

References

[1] Patrick Pantel and Dekang Lin. Spamcop: A spam classication & organization program. In Learning for TextCategorization: Papers from the 1998 Workshop , Madison, Wisconsin, 1998. AAAI Technical Report WS-98-05.
URL http://www.cs.ualberta.ca/ ppantel/Download/Papers/aaai98.pdf .

[2] C. Orasan and R. Krishnamurthy. A corpus-based investigation of junk emails. In Language Resources and Evaluation Conference (LREC- 2002) , Las Palmas, Spain, 2002.

[3] Androutsopoulos, G. Paliouras, V. Karkaletsis, G. Sakkis, C.D. Spyropoulos, and P. Stamatopoulos. Learning to filter spam e-mail: A comparison of a naive bayesian and a memory-based approach. In H. Zaragoza, P. Gallinari, and M. Rajman, editors, Proceedings of the Workshop on Machine Learning and Textual Information Access, 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2000) , pages 1{13, Lyon, France, 2000b. URL http://arXiv.org/abs/cs/0009009

[4] P. Graham. Hackers and Painters, Big Ideas from the Computer Age, chapter 8, pages 121–130.O'Reilly, 2004.

[5] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods In A. Smola, P. Bartlett, B. Schlkopf, and D. Schuurmans, editors, Advances in Large Margin Classiers , pages 61{74. MIT Press, 1999

[6] Androutsopoulos, J. Koutsias, K.V. Chandrinos, G. Paliouras, and C.D. Spyropoulos. An evaluation of naïve bayesian anti-spam ltering. In G. Potamias, V. Moustakis, and M.n van Someren, editors, Proceedings of theWorkshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning(ECML 2000) , pages 9{17, Barcelona, Spain, 2000a. URL http://arXiv.org/abs/cs.CL/0006013 .