DBACL AT THE TREC 2005

L.A. BREYER

Abstract. The dbacl classifier is an open source command line tool which performs spam classification based on simple maximum entropy models; dbacl learns each class separately, and individual categories can be mixed and matched for classification.

Here we present the simulation results obtained for TREC 2005, with an empirical comparison of several feature extraction methods. We also try to gain insight into their different performance characteristics, with limited success.

1. Introduction

`dbacl` is an open source command line tool which performs text and email classification using separate maximum entropy type models for each document class. It can be used for spam filtering within a typical Unix `procmail` recipe. An (occasionally) updated technical account is bundled with the source code as part of the documentation.

The system follows the Bayesian approach of separating the initial model assumptions (likelihood and prior) from the model fitting step (posterior). The latter step depends on features, which can also be viewed as perturbations of the prior model suggested by the training data.

The models depend on features which can be somewhat general, such as overlapping n-grams, and prior distributions which model character transitions. The scores produced by `dbacl` have natural information theoretic interpretations. The combination of features and prior distributions completely characterize the models, in the sense that there are no other *necessary* parameters which must be selected further to adjust the classifications.

Of course, a threshold or some other transformation can be used to adjust the output, and this is precisely what was tested in the TREC spam track. We will show an example where two natural output transformations give different ROC outputs. This is somewhat puzzling as the underlying classifier is in both cases the same.

The `dbacl` program is a batch learning system, which is primarily designed to learn a large number of emails once, and efficiently classify many messages afterwards. Since the TREC evaluation system performs an online test where each message is first classified, then trained, this continual learning imposes a severe computational burden for complex models. This is the reason why the four classifiers tested here are all unigram feature based, with the main difference being the choice of features used. In terms of the priors used, only the third classifier

---

Date: February, 2006.

Key words and phrases. spam, Bayes theorem, email, maximum entropy, TREC.

attempts to estimate priors directly from the data, with fairly disappointing results and a serious online time penalty.

However, efficiency is a natural constraint in spam filtering, where the sheer volume of traffic means that slow solutions are essentially unworkable, regardless of their actual merit. Moroever, while models which incorporate more complex dependencies never fail to gather much interest, it isn't clear that they actually offer superior accuracy against simpler systems.

As a learning system, `dbacl` differs from many state of the art binary classification systems in that it computes class models separately and independently. In other words, *spam* is learned without any reference to *ham* and vice versa, unlike for example SVM systems (Drucker et al, 1999) which try to separate their datasets optimally (note that the successful *ijs PPM6* classifier presented elsewhere in these proceedings also learns classes separately).

Before describing the test results, we present `dbacl` (v.1.11 was tested) as a mathematical model where the role of the various assumptions and switches used in the simulations is explicitly visible.

2. Features, priors and spaminess

The two ingredients of a `dbacl` model are features and a (prior) reference measure. Let $\mathfrak{S}$ be the set of finite octet strings, i.e. strings on a finite alphabet of 256 distinct characters. The set $\mathfrak{S}$ is countably infinite, and we assume given on $\mathfrak{S}$ a probability measure $\mathbb{R}$, i.e. a set of positive numbers $\mathbb{R}(s)$, $s \in \mathfrak{S}$, with $\sum_s \mathbb{R}(s) = 1$.

The measure $\mathbb{R}$ is a prototypical classifier, where $0 \leq \mathbb{R}(s) \leq 1$ is the score assumed by string s. Without data to learn, this can be expected to be a poor classifier; `dbacl` can represent measures $\mathbb{R}$ with constant character transition probabilities p_{ij} (where $\sum_j p_{ij} = 1$, $0 \leq i, j \leq 255$) i.e. based on *character digrams*. In all cases,

$$\mathbb{R}(s) = p_{c_0} \prod_{t=0}^{n} p_{c_t c_{t+1}}, \quad s = c_0 c_2 \cdots c_n \in \mathfrak{S}.$$

The two types of measure tested were the *uniform* measure, where all possible character transitions are uniform ($p_{ij} = 1/256$), and a *Dirichlet* model (MacKay and Peto, 1995), where the transition probabilities are estimated from frequencies observed in the training set. The uniform model needs no computations to prepare, and is preferred for speed as well as (it turns out) stability.

To explain the features used, suppose we specify a family (which may be infinite) of elements $t \in \mathfrak{S}$ called *tokens*. Any string $s \in \mathfrak{S}$ may have zero or more occurrences of any token $t \in \mathfrak{S}$ as a substring. The function $f_t : \mathfrak{S} \to \{0, 1, \dots\}$ which counts the number of times that the token t occurs in s is called a *feature*. It is convenient to talk about features directly rather than tokens, since they appear in the formulas below and represent *sufficient statistics* for the models. Moreover, features (i.e. functions $f : \mathfrak{S} \to \{0, 1, \dots\}$) are more general than tokens and can be used to count transformed substrings such as lower cased tokens, etc. Nevertheless, `dbacl`'s features are all derived from tokens or token classes.

We write $\mathcal{F}$ for the set of features. When two strings s and s' have exactly the same feature counts $f(s) = f(s')$ for all $f \in \mathcal{F}$, we say that they are equivalent. This equivalence relation partitions the set $\mathfrak{S}$ into classes, and is important during learning, because the feature classes must be enumerated to obtain the model weights. In particular, this explains why dbacl features are token counters rather than completely general functions: the calculations would become intractable. The quantity $\chi(s) = \sum_f f(s)$ is finite for all s, and depends only on the equivalence class.

In the particular tested configurations, `dbacl` represents the estimated probability of a string s for a single category by the formula

$$\widehat{\mathbb{P}}(s) = Z^{-\chi(s)} \exp\left(\sum_{f\in\mathcal{F}} \lambda_f f(s)\right) \mathbb{R}(s), \quad s \in \mathfrak{S}. \tag{1}$$

The weights $(\lambda_f : f \in \mathcal{F})$ are real numbers which are nonzero only for features which occur in the training sets, and Z is an approximate normalizing constant. This is therefore a member of an exponential family based on $\mathcal{F}$. In finite dimensional settings, exponential families are well understood (Barndorff-Nielsen, 1978), however the classical theory deals primarily with a small number of nonzero weights.

In text classification, for features which count the occurrence of some token, the number of nonzero weights is very large, which breaks the classical statistical properties and makes the natural setting of (1) infinite dimensional (Portnoy, 1988). The appearance of a large number of weights is natural, because of the typically heavy tails of word distributions, and also the evolving nature of spam, which specifically attempts to bypass word filters. Since `dbacl` is primarily designed for batch learning, it doesn't perform expiry of features or other housekeeping tasks - these can be better handled by controlling the training sets.

Let $\mathfrak{T} = \{\tau_1, \ldots, \tau_n\}$ be a set of training documents, i.e. $\tau_i \in \mathfrak{S}$ for all $i = 1, \ldots, n$. It is convenient to write $\tau = \tau_1\tau_2\cdots\tau_n$ for the string concatenation of the training documents. In `dbacl`, the collection $\mathfrak{T}$ is treated as if it was the single training document $\tau \in \mathfrak{S}$. This is a natural simplification, in view of the (statistical) sufficiency of features in maximum entropy models, and it is also natural from a practical point of view, since email is often collected into archives where the distinction between individual messages can be hard to identify (ambiguous mbox formats, mailing list digests, forwarded messages often blur the line beteen message units).

To obtain the weights (λ_f), `dbacl` minimizes the Kullback-Leibler divergence $D(\mathbb{Q}\|\mathbb{R}) = \sum_s \mathbb{Q}(s)\log(\mathbb{Q}(s)/\mathbb{R}(s))$ from the prior $\mathbb{R}$ to the set of probabilities $\mathbb{Q}$ defined by

$$C(\mathfrak{T},\mathcal{F}) = \Big\{\mathbb{Q} : \sum_{s:\chi(s)=\chi(\tau)} f(s)\mathbb{Q}(s) = f(\tau) \text{ for all } f \in \mathcal{F} \text{ with } f(\tau) > 0\Big\},$$

$$\widehat{\mathbb{P}} = \arg\min\big\{D(\mathbb{Q}\|\mathbb{R}) : \mathbb{Q} \in C(\mathfrak{T},\mathcal{F})\big\}. \tag{2}$$

The result is the I-projection $\widehat{\mathbb{P}}$ of $\mathbb{R}$, which also has several alternative interpretations. The Gibbs Conditioning Principle (Cover and Thomas, 1991) identifies (1) as a limiting posterior distribution, given the training constraint $C(\mathfrak{T},\mathcal{F})$ on the empirical distribution. The exponential family form is actually implied by (2). The minimization is also equivalent to choosing the weights by Maximum Likelihood. The features can themselves be viewed as perturbations of the initial (reference) measure $\mathbb{R}$, and $\mathbb{R}$ can be viewed in (1) as a smoothing term which is used to fill the probability gaps for strings s where active features haven't been learned (i.e. when $f(s) > 0$ with $\lambda_f = 0$). The probability $\widehat{\mathbb{P}}$ itself is intended as an approximation of the law $\mathbb{P}$ on $\mathfrak{S}$ which gives rise to $\mathfrak{T}$, at least within the equivalence classes of defined by $\mathcal{F}$.

For the present TREC simulations, the *spaminess* score is defined as

$$X(s) = \chi(s)^{-1}\Big[\log \widehat{\mathbb{P}}_{①}(s) - \log \widehat{\mathbb{P}}_{②}(s)\Big]. \tag{3}$$

As required by the guidelines, $X(s)$ is large when a message is likely to be *spam*, and small when it is likely to be *ham*, provided of course that both probabilities $\widehat{\mathbb{P}}_{ⓘ}$ are good approximations to the true class probabilities governing the test samples.

3. TREC SIMULATIONS

The TREC simulations were performed against several gold standards. Here, a *gold standard* is a collection $(\mathfrak{S}, \mathbb{P}, T)$, where $\mathbb{P}$ is a fixed probability measure on $\mathfrak{S}$, and $T : \mathfrak{S} \to \{①, ②\}$ is a measurable binary function with the interpretation that $T(s) = ①$ if s is a spam message, $T(s) = ②$ otherwise.

For the actual TREC experiments, we also assume given a sequence of independent emails (strings) $S_1, \ldots, S_{m+n}$ distributed according to $\mathbb{P}$, where m is the number of S_i such that $T(S_i) = ①$, and n is the number such that $T(S_i) = ②$. For $k = 1, \ldots, n+m$, the training sets are

$$\mathfrak{T}^k_{ⓘ} = \mathfrak{T}^0_{ⓘ} \cup \Big\{S_j : 1 \le j \le k,\ T(S_j) = ⓘ\Big\}, \quad i = 1, 2, \tag{4}$$

where $\mathfrak{T}^0_{ⓘ}$ is an initial training set independent of all S_i. At each stage k, the `dbacl` utility first computes the probability models $\widehat{\mathbb{P}}_{①} = \widehat{\mathbb{P}}_{①,k}$ and $\widehat{\mathbb{P}}_{②} = \widehat{\mathbb{P}}_{②,k}$ explicitly as (1), based on the training collections $\mathfrak{T}^k_{①}$ (*spam*) and $\mathfrak{T}^k_{②}$ (*ham*) respectively. Then, the next document S_{k+1} is classified, and training is repeated.

All four submitted filters operate in exactly this way, but each represents a variation of the feature sets and reference measure. The feature definitions are obtained implicitly, by transforming then parsing the input in different ways.

The results of the simulation are shown in Table 1, where the misclassification rates of each type are listed. Note that the percentages are normalized relative to each type of input, not the full size of the dataset. For example, 0.28% for Filter I on `mrx` represents 113 of 40048 *spam* messages, but there are 49,090 total messages.

Approximate binomial confidence intervals for the Table 1 entries are given in Table 2. This is only an approximation and of limited value, since clearly the filter scores are highly dependent from the continual learning.

While the precise definitions of $\mathcal{F}$ and $\mathbb{R}$ can be obtained in each case from reading the supplied source code, here we highlight the major differences:

- In Filter I, the set $\mathcal{F}$ is generated from case sensitive tokens obtained from common email headers such as SMTP trace fields, Subject, From, To, as well as the mail body after MIME decoding and HTML tag removal. The rule for token matching is heuristic, i.e. the tokenizer tries to extract meaningful expressions. The reference probability $\mathbb{R}(s)$ is the law of a uniform random walk on individual characters of s.
- In Filter II, all tokens are lower cased, and the recipe for token matching is more regular than filter 1, consisting of sequences *aaadddppp* where *aaa*, *ddd*, *ppp* are sequences of purely alpha, digit and punctuation characters respectively (these are defined according to the locale in force during the simulation). The measure $\mathbb{R}$ is the same as in Filter I.

dataset	size	spam	uncaught spam % ($\widehat{T} = ②, T = ①$) I	II	III	IV	lost ham % ($\widehat{T} = ①, T = ②$) I	II	III	IV
mrx	49,086	81%	0.28	0.23	0.35	1.21	1.14	1.63	6.84	4.86
trec05p-1	92,189	57%	0.90	0.93	1.05	3.87	0.41	0.51	0.83	0.91
sa	6,034	31%	1.59	2.39	1.59	4.08	1.25	1.28	4.03	1.18
tm	170,201	11%	9.80	11.63	11.04	12.06	0.28	0.29	0.29	0.26
sb	7,006	11%	36.13	33.16	42.45	49.94	0.05	0.03	0.47	0.26

TABLE 1. Simulation results (misclassifications). Approximate binomial confidence intervals are omitted, but depend on the size and spam percentage.

- Filter III is like Filter II, but the reference measure $\mathbb{R}_{ⓘ}$ is obtained by estimating character transition frequencies in a conjugate Dirichlet model (MacKay and Peto, 1995). Note that two measures are estimated, one for each ⓘ.
- Filter IV is a simplification of Filter II, where all headers and internal HTML links are essentially ignored: only the mail body and Subject header generate features, but not e.g. the recipient.

All these filters can be viewed as variations of Filter II, where either the tokenizer is hand optimized (Filter I), the headers are ignored (Filter IV) or the prior distribution is replaced with a (hopefully) better estimate (Filter III).

As a general rule, the filters I and II performed better than either III or IV, and this is visible across all test corpora. It is not surprising that IV should perform the worst, as the header data is deliberately not used. It is well known in spam filtering that headers are a rich source of clues.

Interpreting the results for Filter III is more difficult. Here, the prior $\mathbb{R}_{ⓘ}$ is estimated in terms of observed character transition frequencies rather than chosen arbitrarily. Recall that in the Bayesian framework, the prior represents an initial

dataset	uncaught spam % I	II	III	IV
mrx	0.23-0.34	0.19-0.29	0.29-0.41	1.11-1.32
trec05p-1	0.82-0.98	0.85-1.01	0.97-1.14	3.70-4.03
sa	1.08-2.26	1.75-3.18	1.08-2.26	3.24-5.08
tm	9.39-10.23	11.18-12.09	10.60-11.49	11.61-12.53
sb	32.74-39.62	29.85-36.60	38.94-46.02	46.36-53.51
dataset	lost ham % I	II	III	IV
mrx	0.93-1.38	1.38-1.91	6.33-7.38	4.42-5.32
trec05p-1	0.35-0.48	0.44-0.59	0.74-0.92	0.82-1.01
sa	0.94-1.64	0.96-1.67	3.45-4.67	0.87-1.56
tm	0.25-0.31	0.26-0.31	0.26-0.31	0.24-0.29
sb	0.01-0.14	0.00-0.12	0.31-0.67	0.15-0.42

TABLE 2. Approximate 95% confidence intervals computed under the simplifying assumption of independent binomial trials.

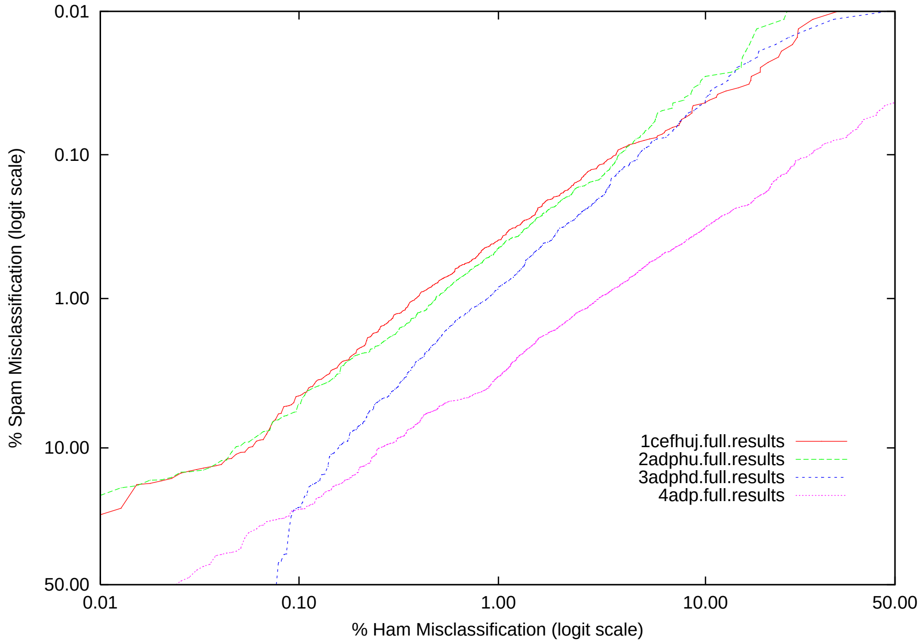

FIGURE 1. Empirically determined ROC curves on the `trec05p-1` corpus for each of the four submitted filters. Each point on the curve represents a classifier $\widehat{T}_t$ for some fixed threshold value t. A classifier located to the northwest of another is more accurate in the sense of (6).

belief, which is updated through the data into a posterior belief. Since $\mathbb{R}_{①}$ ought to be a better initial guess than the uniform measure used for the other filters, the observed performance is disappointing.

The other obvious feature of Table 1 is the poor spam catching performance on the corpora `tm` and `sm`. Neither of these test sets is public, which makes *post mortem* analysis more difficult.

One general issue which can affect performance are parsing problems, which tend to occur disproportionately in *spam*. This is because malformed messages can constitute clever attacks on filtering technology (until addressed). This incentive does not exist for legitimate email, which tends to be more regular and easier to parse. The table shows a much more consistent grouping of *ham* misclassifications over the corpora.

The case of `tm` is the most interesting due to the size of the corpus. Following some discussion, the tentative conclusion is that, for various legitimate reasons, a non-negligible part of the *spam* vocabulary may be present in the *ham* training corpus, confusing the classifier.

The case of `sb` is puzzling, but has not been investigated. It is likely that the low absolute number of spams has a serious effect on the accuracy of the approximation $\widehat{\mathbb{P}}_{①}$, but parsing issues may also play a role.

4. Threshold filters

The *spaminess* defined in (3) is the basis for the ROC diagnostic used in the spam track. For each $-\infty < t < \infty$, let

$$\widehat{T}_t(s) = \begin{cases} ① & \text{if } X(s) > t \\ ② & \text{otherwise.} \end{cases} \tag{5}$$

This defines a family of classifiers indexed by the threshold, where each $\widehat{T}_t$ attempts to approximate the gold standard's T function. Every point on the ROC curve corresponds to a single classifier $\widehat{T}_t$, for some value of t, and vice versa. The misclassification rates listed in Table 1 represent the coordinates of the filter $\widehat{T}_0$ on each dataset.

The combination of (5) and (3) gives the standard MAP (*maximum a posteriori*) classification rule, where t can be viewed as a choice of prior, and the document class with highest estimated probability wins.

Comparing classifiers on the ROC plot is based on the idea that we can partially order two classifiers by their misclassification rates. Thus if $\widehat{T}$ and $\widehat{T}'$ are two classifiers, then $\widehat{T}$ is better than $\widehat{T}'$ on the gold standard $(\mathfrak{S}, \mathbb{P}, T)$ if it makes fewer mistakes of both types, i.e.

$$\begin{aligned} \mathbb{P}\big(\widehat{T} = ① \,|\, T = ②\big) &\leq \mathbb{P}\big(\widehat{T}' = ① \,|\, T = ②\big), \\ \mathbb{P}\big(\widehat{T} = ② \,|\, T = ①\big) &\leq \mathbb{P}\big(\widehat{T}' = ② \,|\, T = ①\big). \end{aligned} \tag{6}$$

Not all classifier pairs can be compared this way, but when possible it is uncontroversial. On the ROC plot, $\widehat{T}$ is better than $\widehat{T}'$ in this way if the coordinates of $\widehat{T}$ are upwards and to the left of the coordinates of $\widehat{T}'$.

Figure 1 shows the ROC for all four submitted filters on the large public corpus. Clearly, every point on the Filter IV curve is dominated by some point on each of the Filter I and II curves. Thus for every choice of threshold t, there exists thresholds u_1, u_2 such that $\widehat{T}_{u_1}(\mathrm{I})$ and $\widehat{T}_{u_2}(\mathrm{II})$ are better than $\widehat{T}_t(\mathrm{IV})$. Filter III also dominates Filter IV for much of the threshold range, except where the ham misclassifications are below 0.1. Neither Filter I or Filter II dominates the other consistently; in fact if confidence intervals were shown, it would be difficult to distinguish the filters at any point of the curve. Other datasets exhibit roughly similar orderings.

The behaviour of Filter IV can be understood by looking more closely at the spaminess (3). For unigram models, let s be an input document, and define the empirical distribution of the features as $\ell(s)(f) = f(s)/\chi(s)$, where $\sum_f \ell(s)(f) = 1$. Thus $\ell(s)$ is a discrete probability on $\mathcal{F}$, which takes the role of the *type* of s (Csiszár, 1998). Using the standard information theoretic identity

$$-\log \widehat{\mathbb{P}}(s) = \chi(s)\Big[D(\ell(s)\|\widehat{P}) + H(\ell(s))\Big] \geq 0, \tag{7}$$

where $D(\nu\|\mu) = \sum_f \nu(f)\log\big(\nu(f)/\mu(f)\big)$, $H(\nu) = -\sum_f \nu(f)\log\nu(f)$, and $\widehat{P}(f) = \widehat{\mathbb{P}}(f(s) = 1 \,|\, \chi(s) = 1)$, we obtain the following alternative expression for (3), noting the reversal of $\widehat{P}_{②}$ and $\widehat{P}_{①}$:

$$X(s) = D(\ell(s)\|\widehat{P}_{②}) - D(\ell(s)\|\widehat{P}_{①}). \tag{8}$$

Now, the collection of features $\mathcal{F}(\mathrm{IV})$ is actually included in $\mathcal{F}(\mathrm{II})$. As explained in Section 3, the difference $\mathcal{F}(\mathrm{II})\backslash\mathcal{F}(\mathrm{IV})$ consists solely of features which are found

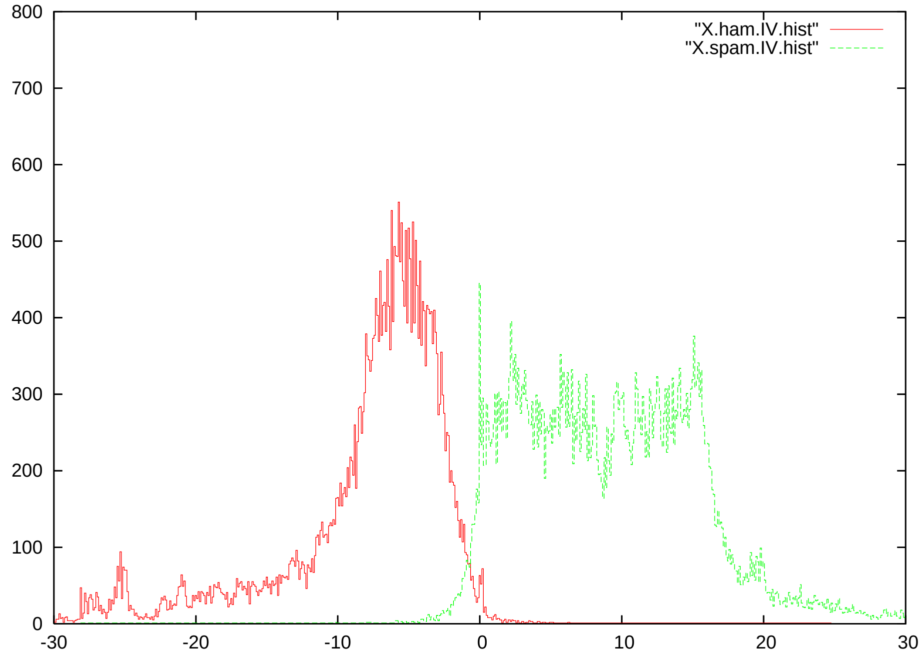

FIGURE 2. Distribution of $X_{\rm IV}$ in `trec05p-1` dataset. Scores on the left are for *ham* messages, on the right for *spam* messages.

in certain message headers. Using (1) and remembering that λ_f is zero unless $\tau(f) > 0$, we have

$$(9) \qquad \widehat{\mathbb{P}}_{\rm II}(s) = \left(Z_{\rm II}/Z_{\rm IV}\right)^{-\chi(s)} \exp\left(\sum_{f\in\mathcal{F}({\rm II})\setminus\mathcal{F}({\rm IV})} \lambda_f f(s)\right)\cdot \widehat{\mathbb{P}}_{\rm IV}(s), \quad s \in \mathfrak{S}.$$

Writing the above as $\widehat{\mathbb{P}}_{\rm II}(s) = \varphi(s)\widehat{\mathbb{P}}_{\rm IV}(s)$ for simplicity, we find using (7) and (8) that

$$(10) \qquad X_{\rm II}(s) = \chi(s)^{-1}\Big[\log\varphi_{①}(s) - \log\varphi_{②}(s)\Big] + X_{\rm IV}(s) = Y(s) + X_{\rm IV}(s).$$

We can interpret this as follows: the effect of adding unrelated features $f \in \mathcal{F}({\rm II})\setminus\mathcal{F}({\rm IV})$ into the models is to shift the spaminess score by a quantity $Y(s)$. This quantity can be viewed as another classifier Y, correlated with X_{IV}.

Figure 2 shows the distribution of $X_{\rm IV}(s)$ on each of the *ham* and *spam* messages as computed during the `trec05p-1` simulation. Since learning takes place continuously, these are not steady state distributions. The region of overlap between the curves indicates the quality of binary classification. For ideal classification, the two regions under the curves would be completely separated.

Figure 3 shows the effect of $Y(s)$. The area of overlap between the curves is smaller, and the curves are slightly better separated. For each $-\infty < t < \infty$, the quantity $\mathbb{P}(\widehat{T}_t = ① \,|\, T = ②) = \mathbb{P}(X > t \,|\, T = ②)$ is the area under the *ham* curve to the right of the line $x = t$. Since the observed effect of Y is to shift the right tail of the *ham* curve slightly to the left, the area under the curve has been slightly

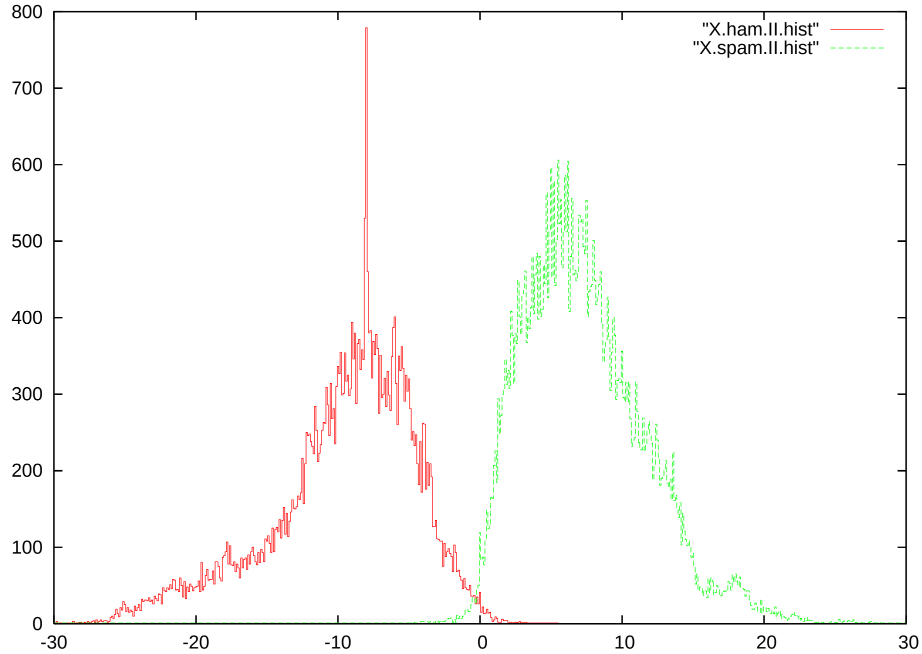

FIGURE 3. Distribution of X_{II} in `trec05p-1` dataset. Scores on the left are for *ham* messages, on the right for *spam* messages.

reduced for t near zero. This can be written approximately

$$\mathbb{P}(X_{\mathrm{II}} > t \,|\, T = ②) \leq \mathbb{P}(X_{\mathrm{IV}} > t \,|\, T = ②), \quad t \text{ near zero.}$$

A similar observation on the *spam* curve implies that, because the left tail is slightly shifted to the right,

$$\mathbb{P}(X_{\mathrm{II}} < t \,|\, T = ①) \leq \mathbb{P}(X_{\mathrm{IV}} < t \,|\, T = ①), \quad t \text{ near zero.}$$

Comparing with (6), we have a range of t close to zero such that the classifiers $\widehat{T}_t(\mathrm{IV})$ are dominated by $\widehat{T}_t(\mathrm{II})$. This domination is what we see in Figure 1.

The behaviour of Y is hard to predict on purely theoretical grounds, as it depends on how well the extra features predict the classes. There is no reason why the Y shift should be always beneficial.

We now briefly discuss the behaviour of Filter III. Using (1) and (3) as before, we have

$$X_{\mathrm{III}}(s) = \log\big(Z^{②}/Z^{①}\big) + \sum_{f \in \mathcal{F}(\mathrm{II})} \big(\lambda_f^{①} - \lambda_f^{②}\big) f(s) + \log\big(\mathbb{R}_{①}(s)/\mathbb{R}_{②}(s)\big). \tag{11}$$

The first two terms have the same form as in the other filters, but the last term never vanishes. This implies that even when s does not contain any previously learned features (so that the second term disappears), the classifier is still able to calculate a nontrivial score for s based on character frequencies. This is therefore a kind of smoothing that does not exist in Filter II.

As seen in Table 1, the overall performance of Filter III is disappointing compared with the simpler versions I and II. Figures 7 and 8 contrast the computed spaminess values over the `trec05p-1` dataset for Filters II and III respectively.

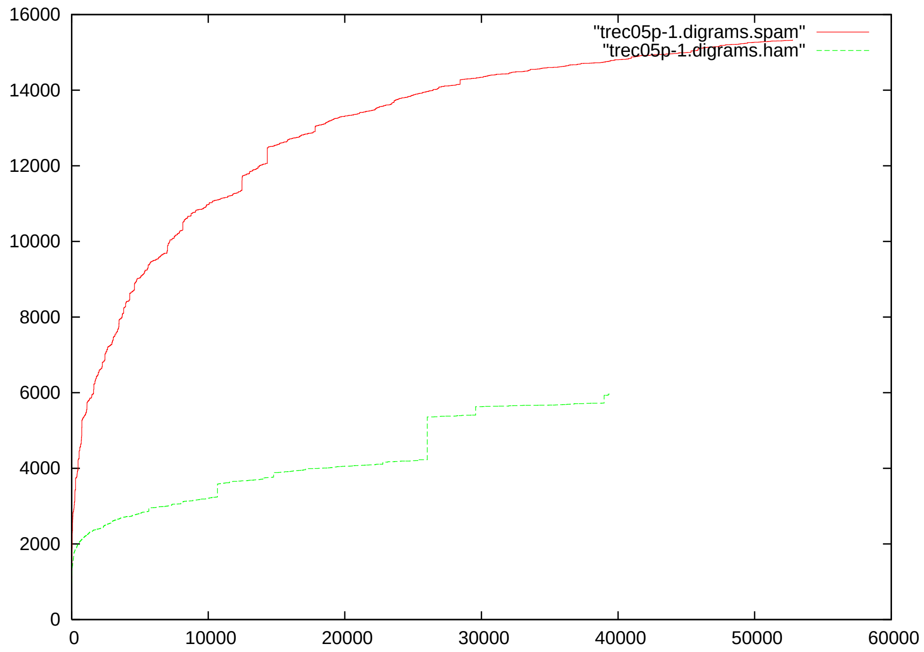

FIGURE 4. Total number of unique digrams parsed from the `trec05p-1` dataset by Filter III, as a function of the number of messages seen so far.

The most obvious difference is that the spaminess is much larger initially in the case of Filter III. Since the measures $\mathbb{R}_{①}$ are estimated separately, from character digrams occurring in separate sets of messages, we must expect that some digrams occur only in one set of messages and not the other.

In fact, Figure 4 shows that the *spam* collection (which has 52,790 messages) contains nearly three times the number of unique digrams that are seen by the filter in the *ham* collection (39,399 messages).

When a digram, *ab* say, occurs in some test document s, then according to (2) and (11), the ratio of its probability under each of $\mathbb{R}_{①}$ and $\mathbb{R}_{②}$ is evaluated. If *ab* is common in the *spam* collection but rare or nonexistent in the *ham* collection seen so far, then this ratio can be large. Only when *ab* is relatively common in both collections should we expect the ratio to be $O(1)$, while if *ab* occurs only in *ham* but not *spam*, then this ratio will tend to be vanishingly small.

Thus, the term $\log\big(\mathbb{R}_{①}(s)/\mathbb{R}_{②}(s)\big)$ is dominated by digrams which tend to occur in one class or the other, whereas the imbalance exhibited by Figure 4 strongly points to a bias in the direction of *spam*.

5. A CAUTIONARY REMARK

The definition (3) of X is in some sense natural, but it is not the only way of embedding `dbacl` into a family of classifiers $\widehat{T}_t$. For example, we could have taken simply

$$X'(s) = \Big[\log \widehat{\mathbb{P}}_{①}(s) - \log \widehat{\mathbb{P}}_{②}(s)\Big]. \tag{12}$$

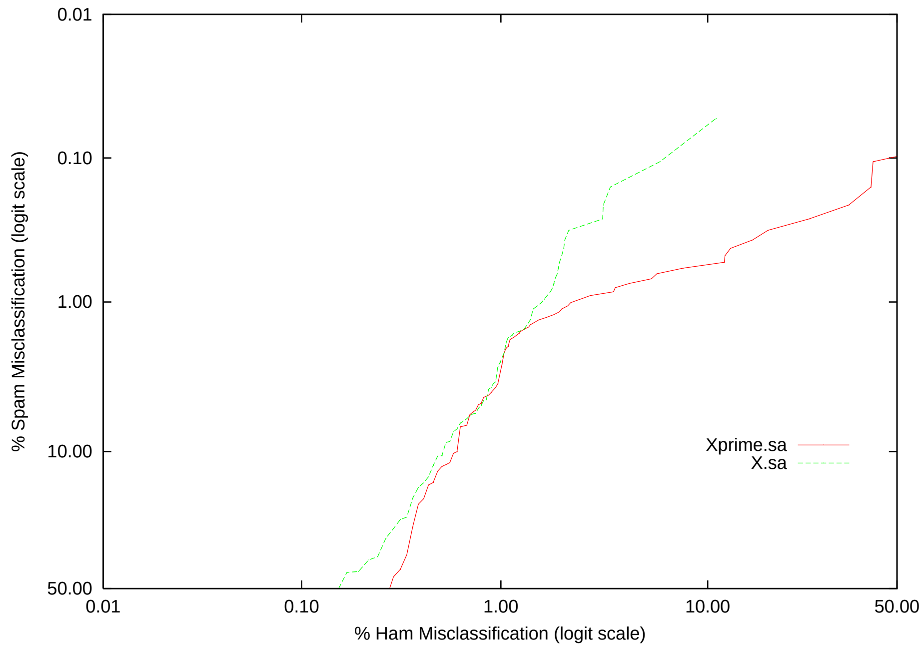

FIGURE 5. Two ROC curves for the same Filter I on the same `spamassassin_corpus` dataset. These curves correspond to choices X and X' defined by (3) and (12).

Clearly, we have $X'(s) > 0$ if and only if $X(s) > 0$, so the two alternatives for spamines give exactly the same results for the default threshold. However, the ROC curves for X and X' differ significantly, even though they always intersect at the point representing $\widehat{T}_0$ (see Figure 5). This suggests that the ROC picture should not be viewed as an *intrinsic* property of the classifier, but at most as a property of the chosen parametrization.

6. Variations of the spam rate

In this section, we report on an intriguing pattern in Table 1, namely that the misclassification rates seem to be monotonic functions of the *spam rate* (percentage of spam). More precisely, note that the datasets have been ordered by *spam* rate, and that the rate of uncaught *spam* decreases with increasing spam rate, while the rate of lost *ham* increases instead (only roughly, the `sa` dataset being a partial exception).

Unfortunately, an analysis of this using the machinery of classical statistics is not possible, because the *spaminess* X doesn't follow the classical theory for exponential families. The phenomenon is interesting however, so we expand on it here.

Table 3 shows the results of performing the simulation runs on the `trec05p-1` partial datasets, which were not listed in Table 1. Only two filters were tested, I and IV, due to time constraints. Again, the monotonicity as a function of the *spam* rate is striking.

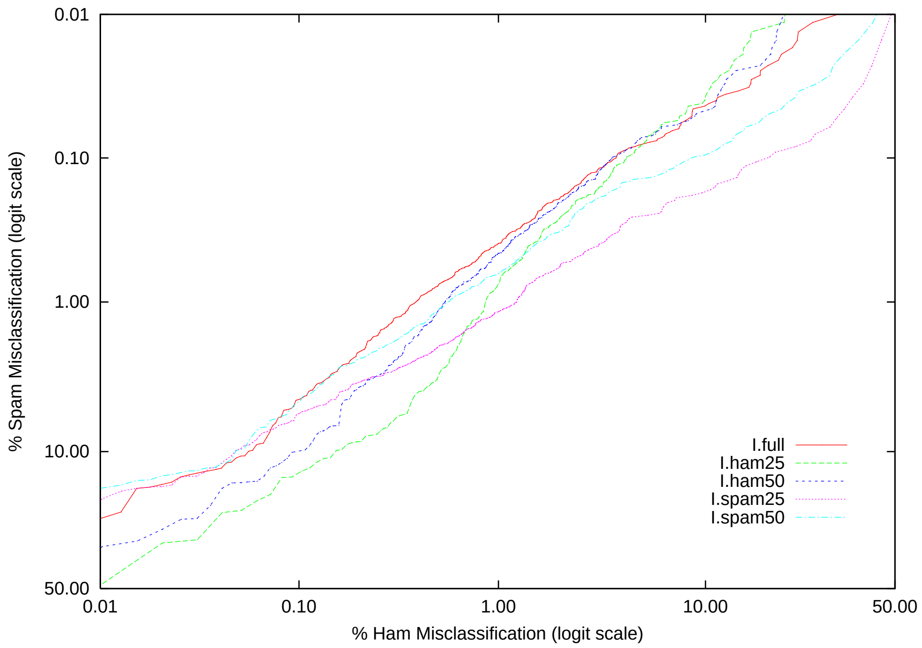

FIGURE 6. Effect of varying the spam rate for `trec05p-1` dataset on Filter I. Reducing the amount of training data shifts the curve in a single direction.

The ROC curves for Filter I are plotted simultaneously in Figure 6. The changing spam rate clearly induces a tilting of the basic curve depending on the class being reduced. A similar plot exists for Filter IV.

Consider what happens when the dataset is reduced from `trec05p-1/full` (experiment A) to `trec05p-1/ham25` (experiment B). In the full dataset, we have m samples of type ①, and n samples of type ②. In the reduced case, we still have m samples of type ①, but only $n/4$ samples of type ②. The estimated models $\widehat{\mathbb{P}}_{①}$ are the same in both cases. In Figure 6, the classifiers $\widehat{T}_t^A$ dominate $\widehat{T}_t^B$ in the sense of (6) over a good part of the threshold range.

Suppose we try to classify a document s. Using (8), we have

$$X^A(s) = D(\ell(s)\|\widehat{P}^A_{②}) - D(\ell(s)\|\widehat{P}_{①})$$

dataset	size	spam	uncaught spam % ($\widehat{T} = ②, T = ①$) I	IV	lost ham % ($\widehat{T} = ①, T = ②$) I	IV
`trec05p-1/ham25`	62,541	84%	0.31	1.46	1.68	2.96
`trec05p-1/ham50`	72,376	72%	0.58	2.57	0.85	1.44
`trec05p-1/full`	92,189	57%	0.90	3.87	0.41	0.91
`trec05p-1/spam50`	65,682	40%	1.95	6.94	0.28	0.71
`trec05p-1/spam25`	52,578	25%	4.33	11.69	0.16	0.58

TABLE 3. Simulation results (misclassifications).

$$X^B(s) = D(\ell(s)\|\widehat{P}^B_{②}) - D(\ell(s)\|\widehat{P}_{①}).$$

In classical statistics, these would be likelihood ratio statitics whose asymptotic χ^2-distribution (after normalization by the size of s) only depend on the number of parameters of the respective models. However, these asymptotics do not apply in our case, because the number of parameters (i.e. unique features) grows too quickly with the number of training samples (i.e. total feature counts), thereby preventing a Gaussian limit, see (Portnoy, 1988).

A partial but incomplete picture is suggested by information theory. Since experiment B uses a subset of the ② messages from experiment A, it follows that, whenever a document s is classified, the model $\widehat{\mathbb{P}}^A_{②}$ has been estimated from strictly more data than the model $\widehat{\mathbb{P}}^B_{②}$. Thus $\widehat{\mathbb{P}}^A_{②}$ should be a better predictor than $\widehat{\mathbb{P}}^B_{②}$ for typical documents s, so that $D(\ell(s)\|\widehat{P}^A_{②}) \le D(\ell(s)\|\widehat{P}^B_{②})$ with high probability, i.e. $X^A(s) \le X^B(s)$, conditionally on $T(s) = ②$. If this assumption holds, then we can deduce

$$\mathbb{P}\big(X^A > t \,|\, T = ②\big) \le \mathbb{P}\big(X^B > t \,|\, T = ②\big).$$

This is equivalent to the first half of (6), but we cannot hope to compare $D(\ell(s)\widehat{P}^A_{②})$ with $D(\ell(s)\widehat{P}^B_{②})$ when $T(s) = ①$, so the second half of (6) can't be argued this way.

7. Acknowledgements

I thank Gordon Cormack and his team for their hard work and flexibility in creating the simulations, and some interesting discussions. Thanks also to Tony Meyer for helpful comments related to the `tm` dataset.

References

[1] Barndorff-Nielsen, O. (1978) *Information and Exponential Families in Statistical Theory.* Wiley.

[2] Breyer, L.A. (2005) *The dbacl Text Classifier* `http://www.lbreyer.com/gpl`.

[3] Cover, T.M. and Thomas, J.A. (1991) *Elements of Information Theory.* Wiley.

[4] Csiszár, I. (1998) *The Method of Types.* IEEE Trans. Inf. Th. **44** (6) 2505–2523.

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

[6] MacKay, D. and Peto, L. (1995) *A Hierarchical Dirichlet Language Model.* Natural Language Engineering 1(3): 289–307.

[7] Portnoy, S. (1988) Asymptotic behaviour of likelihood methods for exponential families when the number of parameters tends to infinity. *Ann. Statistics* 16, 356–366.

Project homepage: `http://dbacl.sourceforge.net`
E-mail address: `laird@lbreyer.com`

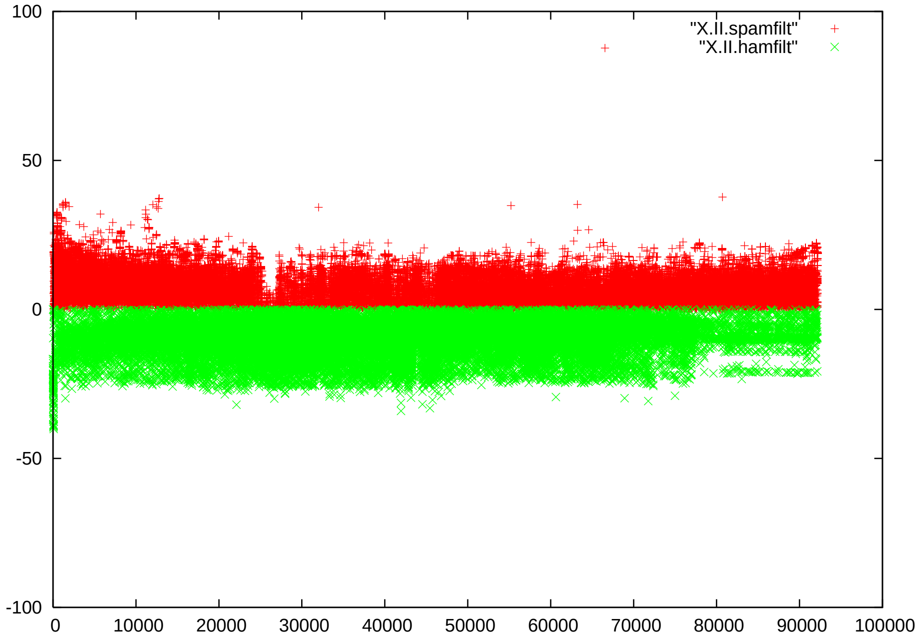

FIGURE 7. Spaminess scores (Filter II) for all the messages in `trec05p-1` dataset.

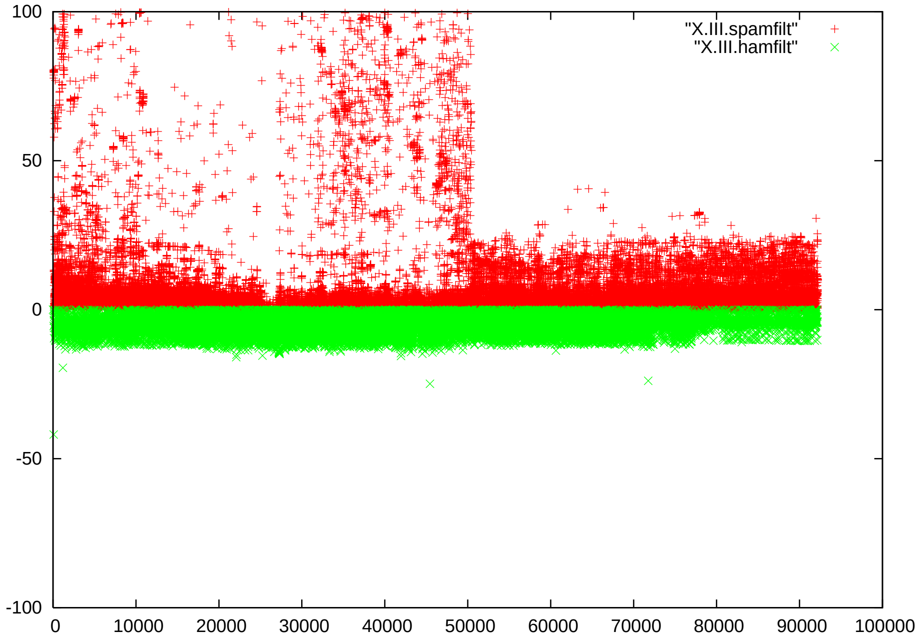

FIGURE 8. Spaminess scores (Filter III) for all the messages in `trec05p-1` dataset. Unlike Filter II, the scores are much larger initially.