THE DBACL TEXT CLASSIFIER: COST MINIMIZATION WITH BAYESOL

L.A. BREYER

Abstract. This note describes the details of cost minimization for the **dbacl** classifier, as implemented in the utility **bayesol**. Due to limited precision of floating point operations, such a minimization cannot be done in the straightforward way.

1. Introduction

dbacl is a digramic Bayesian classifier for text documents (see Breyer, 2002). The program reads a corpus of text documents, and computes a corresponding maximum entropy model. This model can be used to compute the probability of an unknown document, and thereby classify it among several previously learned corpora.

Normally, **dbacl** outputs a set of scores representing the average information present per document feature, for each model to be compared. This information is measured in bits, and is related to the posterior probability for the category C, given the document string s, as follows:

$$\mathbf{score}_2(s, C) = -\log_2 \mathbb{P}(C \mid s)/\chi_C(s), \tag{1}$$

where $\chi_C(s)$ is the complexity of s, ie the number of features which exist in s and are recognized by the category model C. Full details are given in (Breyer, 2002).

The score comparison is particularly attractive when all categories are based on the same feature sets, so that $\chi_C(s)$ is the same for all categories C. In that case, the *maximum a posteriori* (MAP) estimate reduces to finding the category C for which the $\mathbf{score}_2(s, C)$ is smallest. Geometrically, $\mathbf{score}_2(s, C)$ is like a distance to

Date: April, 2004.

1991 *Mathematics Subject Classification.* Primary 60J, Secondary 60F.

Key words and phrases. SPAM, Bayes theorem, email, Maximum entropy.

the category C, and then the MAP estimate identifies the closest category to the string s.

This distance-like property can be made precise via the interpretation of the $\mathbf{score}_e(s, C)$ as a limiting Kullback-Leibler divergence. Note that we've switched from base 2 logarithms to natural logarithms. Suppose that the sequence of features in s forms a stationary process with marginal distribution $\mathbb{P}$. We assume such a statement can be made precise, but we won't do so here. Then by the Shannon-McMillan-Breiman theorem, and in the notation of (Breyer, 2002), we expect

$$\lim_{\chi(s)\to\infty} \chi(s)^{-1} \log \mathbb{P}_C(s) = -D(\mathbb{P} \| \widetilde{R}_C).$$

Comparing this with (1) (replacing 2 with e) suggests the connection.

2. Minimizing Costs

An alternative to the MAP estimate consists in specifying a *cost* or *utility* matrix $\mathcal{L}[i, j]$. This is the approach taken by the companion utility **bayesol**, and is the focus of this note.

Note that minimizing costs or maximizing utility are mathematically equivalent problems, and we shall from now on assume that $\mathcal{L}[i, j]$ is the *cost of misclassifying the true category i as j*. By convention, $\mathcal{L}[i, i] = 0$ for all i, and for a classification problem with k distinct categories, the matrix $\mathcal{L}$ is obviously $k \times k$. In particular, it always has an overall minimum value, and without loss of generality, we shall therefore assume that $\mathcal{L}[i, j] \geq 0$.

The **bayesol** utility is able to handle a quite general form of cost matrix, which it uses to find the category whose overall cost is a minimum. More precisely, consider the possible decisions w, which we identify as functions on strings s, whose value represents the inferred category. Thus $w(s) = j$ means that the decision w yields category j for the string s. It is well known that the optimal Bayesian decision is the function w_* such that

$$w_* = \arg\min_{w} \sum_{i=1}^{k} \mathcal{L}[i, w(s)]\mathbb{P}(C_i \mid s).$$

By writing $w = \sum_i i \cdot 1_{\{w=i\}}$, it is not hard to obtain an expression for $w_*(s)$, namely

$$w_*(s) = \arg\min_a \sum_{i=1}^{k} \mathcal{L}[i,a]\mathbb{P}(C_i \mid s). \tag{2}$$

While this expression is easy to compute in principle, the realities of limited precision arithmetic render it mostly useless. In the next section, we describe how **bayesol** copes with this problem.

3. Approximately identifying the minimizer

While the expression (2) is simple enough to compute in principle, in actual fact it is next to impossible to use accurately, due to catastrophic loss of precision.

The problem arises due to the typical form of the conditional probabilities $\mathbb{P}(C_i \mid s)$ computed by **dbacl**. In normal cases, these are exponentially small, which is why the scores (1) are the preferred quantities to use for decisions.

Practical experience with **dbacl** suggests that the range of scores for typical email classification is between 5 and 100, with most values distributed betwen 10 and 20. Even so, since $\mathbb{P}(C \mid s) \sim \exp\big(\chi_C(s)\mathbf{score}(C,s)\big)$, the document complexity matters much more, since $\chi_C(s)$ has a typical range between 500 and several thousand features. It is therefore clear that $\mathbb{P}(C \mid s)$ cannot be reasonably represented by typical floating point variables.

The solution used in **bayesol** is to infer the minimizer entirely in the log scale. Let $L[i,j] = \log \mathcal{L}[i,j]$, then we can rewrite (2) in the equivalent form

$$w_*(s) = \arg\min_a \sum_{i=1}^{k} \exp\Big(-\chi_{C_i}(s)\mathbf{score}_e(s,C_i) + L[i,a]\Big). \tag{3}$$

Naturally, we generally cannot simply sum the terms in (3) directly, but luckily we don't need to. We can identify the sum with approximately the smallest value by comparing the exponents directly. Let

$$v[a,i] = -\chi_{C_i}(s)\mathbf{score}_e(s,C_i) + L[i,a], \tag{4}$$

and suppose that, for fixed a, we reorder these terms in decreasing order, obtaining a sorted vector $v_*[a] = \{a_1, \ldots, a_k\}$, whose first term a_1 is the largest exponent $\max_i v[a,i]$ above, and whose last term $a_k = \min_i v[a,i]$ is the smallest. The other elements satisfy $a_n \geq a_{n+1}$.

We now compare $v_*[a] = \{a_1, \ldots, a_n\}$ and $v_*[b] = \{b_1, \ldots, b_n\}$ termwise, from largest to smallest, stopping at the first difference. Thus we first compare a_1 and b_1, and if $a_1 = b_1$, then we compare a_2 and b_2 etc. We write $v_*[a] \rhd v_*[b]$ if at the first difference, $a_n > b_n$, and we assume that this implies that the corresponding sums are ordered the same way..

This procedure does not always yield exactly the smallest sum overall, but to obtain a counterexample, it is necessary that the terms of $v_*[a]$ and $v_*[b]$ be *very* close. In practice, such closeness is extremely rare, and can be ignored.

Lemma 1. *Consider two finite sequences $a = \{a_1, \ldots, a_k\}$ and $b = \{b_1, \ldots, b_k\}$ such that $a_n \geq a_{n+1}$ and $b_n \geq b_{n+1}$ for all $n = 1, \ldots, k-1$. Define $a \rhd_r b$ if and only if $a_j = b_j$ for $j < r$, and $a_r > b_r$. Also define, for $r = 1, \ldots, k-1$,*

$$S_r(a) = \sum_{n=r}^{k} e^{a_n}, \qquad S_r(b) = \sum_{n=r}^{k} e^{b_n}. \tag{5}$$

Then we have

$$a \rhd_r b \text{ implies } S_1(a) > S_1(b), \text{ provided } a_r - b_r > \log(k-r). \tag{6}$$

Proof. By writing

$$S_1(a) = \sum_{n=1}^{r-1} + S_r(a) = \sum_{n=1}^{r-1} + e^{a_r}\Big(1 + \theta_r(a)\Big),$$

and similarly for b, it follows that

$$\Big\{ a \rhd_r b \text{ and } S_1(a) > S_1(b) \Big\}, \text{ if and only if } a_r - b_r > \log\left(\frac{1+\theta_r(a)}{1+\theta_r(b)}\right).$$

But clearly, $0 \leq \theta_r(a), \theta_r(b) \leq k - r - 1$, because

$$\theta_r(a) = \sum_{n=r+1}^{k} e^{a_n - a_r},$$

and $a_r \geq a_n$ for all $n \geq r$. Consequently, we obtain the claimed bound. $\square$

Corollary 1. *Suppose that there are k categories, that for $i = 1, \ldots, k$, $\chi_{C_i}(s) = \chi(s)$ is independent of s, and suppose also that $L[i,j] = \ell[i,j] \cdot \chi(s)$. If for all i, j, k, l, either*

$$\Big| \mathbf{score}_e(s, C_j) - \mathbf{score}_e(s, C_i) + \ell[i,k] - \ell[j,l] \Big| > \log(k)/\chi(s),$$

or

$$\Big|\mathbf{score}_e(s,C_j) - \mathbf{score}_e(s,C_i) + \ell[i,k] - \ell[j,l]\Big| = 0,$$

then the **bayesol** *minimizer is the true minimizer.*

Proof. For the **bayesol** minimizer to be correct, we must verify that $a \rhd_r b$ implies $S_1(a) \geq S_1(b)$ for all relevant sequences. From (3), the typical terms a_n and b_n respectively have the form

$$a_n = \chi(s)\Big(-\mathbf{score}_e(s,C_i) + \ell[i,k]\Big), \qquad b_n = \chi(s)\Big(-\mathbf{score}_e(s,C_j) + \ell[j,l]\Big).$$

If $a_n = b_n$, then the comparison of $S_1(a)$ and $S_1(b)$ depends on other terms, either before or after the n-th terms, so we can assume $a_n \neq b_n$ without loss of generality. By relabeling a and b if necessary, let n be such that $a \rhd_n b$. By Corollary 1, to obtain $S_1(a) \geq S_1(b)$, it is sufficient that $|a_n - b_n| > \log k$. Rearranging this inequality, we obtain the stated conditions. □

Note that for typical documents, $\chi(s) > 500$, so that for example with 5 categories or less, the scores should be no closer than 0.003, unless they are identical.

4. Connection with MAP estimate

The MAP estimate is a special case of the general cost minimizing estimate. The MAP estimate can be obtained by setting

$$L[i,j] = \begin{cases} 0 & \text{if } i \neq j \\ -\infty & \text{if } i = j. \end{cases} \tag{7}$$

It is not hard to see that the cost minimizer agrees with the MAP estimate, since (3) becomes simply

$$\begin{aligned} w_*(s) &= \arg\min_a \sum_{i=1, i\neq a}^{k} \exp\Big(-\chi_{C_i}(s)\cdot \mathbf{score}_e(s,C_i)\Big), \\ &= \arg\min_a \Big\{ Z(s) - \exp\Big(-\chi_{C_a}(s)\cdot \mathbf{score}_e(s,C_a)\Big)\Big\}, \end{aligned} \tag{8}$$

where $Z(s) = \sum_{i=1}^{k} \exp\Big(-\chi_{C_i}(s)\mathbf{score}_e(s,C_i)\Big)$. Since $Z(s)$ is independent of a, it is easy to see that we also have

$$w_*(s) = \arg\max_a \Big\{\chi_{C_a}(s)\cdot \mathbf{score}_e(s,C_a)\Big\}. \tag{9}$$

However, note that in general, $Z(s)$ would depend on a if $L[i,j]$ is nonzero for $i \neq j$. Also, even in the MAP case, $Z(s)$ can only be computed very approximately, and (9) cannot be relied on for calculations.

5. Cost rankings

One big difference between the MAP estimation procedure and a general cost minimization is that the former allows a natural ranking of documents, from most probable to least probable. Such a ranking is trivial to deduce from the scores obtained from **dbacl**, via the formula (1): the lower the score (closer to zero), the more probable the category.

By contrast, the cost minimization problem induces several natural orderings of the categories. The simplest ordering is based on directly using the conditional costs which are minimized in (3). This is approximately reported via **bayesol**'s *-n* switch.

A practical difficulty with this scheme is that the non-optimal costs often exhibit by very little, or no difference, which makes the ordering impossible. Moreover, even in the best conditions, the very precision problems which plague the minimization also affect the rankings so obtained. The reported costs are only roughly accurate on the log scale.

The **bayesol** utility also offers a different scheme (via the *-N* switch) to obtain a category ordering based on the costs. In this scheme, the first category is based on the minimal cost. Next, this best category is removed from the list of available categories, and the second best category is chosen to be that category which minimizes the cost for the reduced category set. This is repeated until no further removals are possible.

$$
\begin{aligned}
c_1 &= \arg\min_a \sum_{i=1}^k \exp\Big(-\chi_{C_i}(s)\mathbf{score}_e(s, C_i) + L[i,a]\Big) \\
c_2 &= \arg\min_{a \neq c_1} \sum_{i=1, i\neq c_1}^k \exp\Big(-\chi_{C_i}(s)\mathbf{score}_e(s, C_i) + L[i,a]\Big) \\
&\vdots \\
c_k &= \{i : i \neq c_1, \ldots, c_{k-1}\}.
\end{aligned}
$$

This methodology is easy to compute recursively, and each subproblem has the same form as the original. In fact, no change in the code is needed if the $L[i,a]$ matrix is censored appropriately at each n-th stage, obtaining a matrix $L_n[i,j]$ by setting $L_0[i,j] = L[i,j]$ and

$$L_{n+1}[i,j] = \begin{cases} +\infty & \text{if } i = j = c_n \\ -\infty & \text{if } i = c_n \text{ or } j = c_n \\ L_n[i,j] & \text{otherwise.} \end{cases}$$

It is important to note that the ordering $\{c_1, \ldots, c_k\}$ coincides with the MAP ordering in the case when L is chosen as in Section 7.

This is easy to see inductively as follows (outline): at the first stage, c_1 coincides with the MAP estimate, by the argument in the earlier section. After censoring, only L_1 still has the form required for MAP, when restricted to the remaining categories. Thus c_2 coincides with the MAP estimate on the restricted set, ie c_2 is the second best MAP estimate, and so on.

However, a suprirising fact is that in the MAP case (7), the computed costs do *not* resemble the MAP costs. To see this, we assume that the Laplace approximation holds, ie that the sum of exponential terms is approximately equal to its leading term. This is quite realistic in practical examples, typically due to the magnitude of $\chi(s)$.

$$\begin{aligned}
\mathbf{cost}(c_1) &= \min_a \sum_{i=1}^k \exp\Big(-\chi_{C_i}(s)\mathbf{score}_e(s, C_i) + L[i,a]\Big) \\
&= \min_a \sum_{i \neq a} \exp\Big(-\chi_{C_i}(s)\mathbf{score}_e(s, C_i)\Big) \\
&\sim \min_a \max_{i \neq a} \Big\{\exp\Big(-\chi_{C_i}(s)\mathbf{score}_e(s, C_i)\Big)\Big\} \\
\mathbf{cost}(c_2) &= \min_{a \neq c_1} \sum_{i=1, i \neq c_1}^k \exp\Big(-\chi_{C_i}(s)\mathbf{score}_e(s, C_i) + L[i,a]\Big) \\
&\sim \min_{a \neq c_1} \max_{i \neq a, c_1} \Big\{\exp\Big(-\chi_{C_i}(s)\mathbf{score}_e(s, C_i)\Big)\Big\} \\
&\vdots \\
\mathbf{cost}(c_k) &= \text{undefined.}
\end{aligned}$$

If the $\chi_{C_i}(s)$ do not depend on i say, then it follows from these estimates that $\log \mathbf{cost}(c_1)$ is proportional to the *second best* score, that $\log \mathbf{cost}(c_2)$ is proportional to the *third best score* etc. This is qualitatively quite different from the MAP ordering of scores.

Clearly, this result is troubling. The orderings are the same, but the extra qualitative information about the shape of the posterior is lost. Fortunately, it is possible to recover the shape at least approximately. The key is to realize that the **cost** function can be added to a constant without harming the minimization.

For a fixed string s, consider the alternative cost function **cost'** defined by

$$\mathbf{cost'}(s) = \min_a \sum_{i=1}^k \exp\Big(-\chi_{C_i}(s)\mathbf{score}_e(s, C_i) + L[i, a]\Big) + \sum_{i=1}^k \mathbb{P}(C \mid s).$$

As the last term does not depend on a, it does not affect the minimization, yielding the same ordering as for the original **cost**.

By rearranging the equation in the case of L given by (7), we obtain easily

$$\begin{aligned}\mathbf{cost'}(c_1) &= \min_a \Big\{2 \sum_{i=1, i\neq a}^k \exp\Big(-\chi_{C_i}(s)\mathbf{score}_e(s, C_i)\Big) + \exp\Big(-\chi_{C_a}(s)\mathbf{score}_e(s, C_a)\Big)\Big\} \\ &= 2\sum_{i=1}^k \exp\Big(-\chi_{C_i}(s)\mathbf{score}_e(s, C_i)\Big) - \max_a \Big\{\exp\Big(-\chi_{C_a}(s)\mathbf{score}_e(s, C_a)\Big)\Big\} \\ &\sim \max_a \Big\{\exp\Big(-\chi_{C_a}(s)\mathbf{score}_e(s, C_a)\Big)\Big\},\end{aligned}$$

where we again assume that the Laplace approximation holds. A similar calculation shows that $\log \mathbf{cost'}(c_2)$ is proportional to the second best score, etc. Thus with this modified cost function, we obtain a natural generalization of the MAP ordering, which keeps the shape of the posterior intact, at least in the MAP case.

References

[1] Breyer, L.A. (2002) *The dbacl text classifier.*

E-mail address: `laird@lbreyer.com`