# Sparse Bayesian Classifiers for Text Categorization - Batch Learning

**Alexander Genkin**
*agenkin@dimacs.rutgers.edu*
**David D. Lewis**
*ddlewis2@worldnet.att.net*
**David Madigan**
*madigan@stat.rutgers.edu*

December 1, 2002

## 1 Introduction

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, where a feature vector represents the documents. Standard statistical classification tools such as Naive Bayes, logistic regression, and decision trees are immediately relevant and have been used with some success.

Researchers in statistical text categorization face two particular challenges. First, the scale of many applications overwhelms many standard learning algorithms. Document feature vectors with the bag-of-words representation are often of dimension $10^5 - 10^6$. The number of documents can also be very large (although a paucity of labeled examples is more usual). Second, treating text categorization as a standard

classification problem ignores extra-document information such as category descriptions and drifting category definitions.

The former challenge motivated early interest in computationally less challenging approaches such as Naive Bayes and Rocchio. More recently, raw computing power coupled with algorithmic developments have enabled regularization-based methods such as support vector machines and ridge logistic regression to achieve substantially improved accuracy on standard test collections. The latter challenge has attracted less attention.

Bayesian classifiers provide a form of automated regularization and can incorporate external information via prior distributions and offer the possibility of addressing both challenges. In what follows we pursue the Bayesian approach.

We make a distinction between batch learning and online learning. Batch learning algorithms take as input a set of labeled documents and output a categorization algorithm. Online learning algorithms operate on one labeled document at a time, at each stage producing an updated categorization algorithm. This paper concerns batch learning; a companion report will discuss online learning.

# 2 Formal Framework

Inductive supervised learning infers a functional relation $y = f(\mathbf{x})$ from a set of training examples $D = \{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_n}, y_n)\}$. In what follows the inputs are vectors $[x_{i_1}, \ldots, x_{i_d}]^T$ in $\Re^d$, $y \in \{-1, 1\}$, and we refer to $f$ as a classifier. We assume that a vector of parameters, $\boldsymbol{\beta}$, defines $f$ and we write $f(\mathbf{x}; \beta)$. The learning procedures we consider output either a point estimate of $\boldsymbol{\beta}$ or a posterior distribution for $\beta$. Usually the input vectors will correspond to term frequencies but the framework is general.

Our objective is to produce a classifer that makes accurate predictions for future input vectors. Typically this requires the learning procedure to control complexity and avoid over-fitting the training data ("regularization"). The Bayesian approach to complexity places a prior distribution on $\boldsymbol{\beta}$, typically resulting in estimates that shrink towards zero. Using so-called Laplacian (i.e., double exponential) priors can result in posterior modes for some components of $\boldsymbol{\beta}$ that are exactly zero. Such

"sparse classifiers" yield excellent predictive performance on standard datasets and are closely related to support vector machines (SVM) (see Girosi, 1998, and Zhang and Oles, 2001), relevance vector machines (RVM) (Tipping, 2001), and to the lasso (Tibshirani, 1995).

We consider classifiers of the form:

$$p(y = 1|\mathbf{x}) = \psi(\boldsymbol{\beta}^T h(x)).$$

The logistic link function is a common choice for $\psi$, but we instead adopt the probit model, $\psi(z) = \Phi(z)$, where $\Phi$ denotes the standard Gaussian distribution function. The probit link function leads to simpler learning algorithms. See Chambers and Cox, 1967, for a comparison of the logistic and probit models. Later we may consider alternative forms for $\mathbf{h}$, but for now $\mathbf{h}(\mathbf{x}) = [1, x_1, \ldots, x_d]^T$ (i.e., the original input variables).

We adopt a hierarchical prior for $\boldsymbol{\beta}$, specifically, for $i = 1, \ldots, d$:

$$p(\beta_i|\tau_i) = N(0, \tau_i)$$

and

$$p(\tau_i) \propto 1/\tau_i.$$

The prior on the $\tau$'s is the Jeffreys' prior. Replacing it with an exponential prior

$$p(\tau_i|\gamma) = \frac{\gamma}{2} \exp(-\frac{\gamma}{2}\tau_i),$$

induces the following Laplacian prior on $\boldsymbol{\beta}$

$$p(\beta_i|\gamma) = \frac{\sqrt{\gamma}}{2} \exp(-\sqrt{\gamma}|\beta_i|).$$

Tipping (2001) uses a gamma prior in place of the exponential.

The maximum *a posteriori* (MAP) estimates under the Laplacian prior are the lasso estimates. Note that the Laplacian prior requires a choice for the hyperparameter $\gamma$, perhaps via cross-validation, whereas the Jeffreys prior is more convenient insofar as it has no tunable hyperparameters.

# 3 Learning Algorithms - EM

Here we describe an Expectation Maximization (EM) algorithm for finding the posterior mode. The algorithm makes use of the latent variable representation for probit regression introduced by Albert and Chib (1993) and we describe this first. Define $n$ independent latent variables $z_1, \ldots, z_n$ where $z_i$ has distribution $N(\boldsymbol{\beta}^T h(x), 1)$. Then define $y_i = 1$ if $z_i > 0$ and $y_i = -1$ if $z_i \leq 0$. It is straightforward to show that the $y_i$ are then independent Bernoulli variables with $p(y_i = 1) = \Phi(\boldsymbol{\beta}^T h(x_i)), i = 1, \ldots, n$, and we recover the standard probit model. Figure 1 shows the corresponding graphical Markov model using the BUGS plate notation.

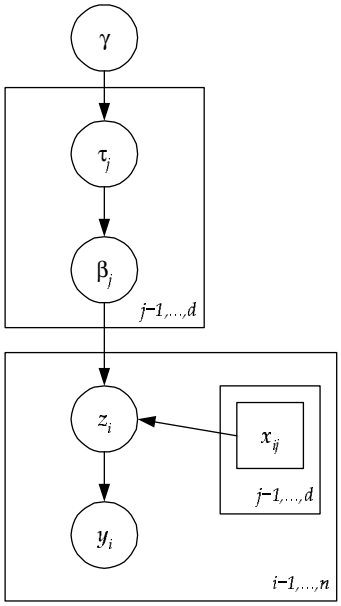


Figure 1: *Probit Model with Hierarchical Prior and Latent Variables.*

Observe that if the $z_i$ are known, and $\boldsymbol{\beta}$ has a multivariate normal prior, then the posterior distribution for $\beta$ is available in closed form using standard normal linear model results. The $z_i$ are latent, but given the data $y_i$, $z_i$ has truncated normal distribution. This setup facilitates the EM algorithm.

For both the Laplacian and FJ priors, the EM algorithm estimates $\boldsymbol{\beta}$ by treating both $\tau$ and $\mathbf{z}$ as missing. The complete data log-posterior is:

$$\log p(\beta|\mathbf{y}, \tau, \mathbf{z}) \quad \propto \quad \log p(\mathbf{z}|\beta) + \log p(\boldsymbol{\beta}|\tau)$$

$$\begin{aligned} &\propto -\|\mathbf{H}\beta - \mathbf{z}\|^2 - \boldsymbol{\beta}^T \Gamma(\tau)\beta \\ &\propto -\boldsymbol{\beta}^T H^T (H\boldsymbol{\beta} - 2\mathbf{z}) - \boldsymbol{\beta}^T \Gamma(\boldsymbol{\tau})\boldsymbol{\beta}, \end{aligned}$$

where $\mathbf{\Gamma}(\tau) = \text{diag}(\tau_i^{-1}, \ldots, \tau_d^{-1})$ and $\mathbf{H}$ is the design matrix with rows $\mathbf{h(x_1)^T}, \ldots, \mathbf{h(x_n)^T}$. Note that the complete data log-likelihood is linear in $\mathbf{z}$. The E-step requires computing $E[\tau_i^{-1}|\hat{\boldsymbol{\beta}}^{(t)}, \mathbf{y}, \gamma]$ and $E[z_i|\hat{\boldsymbol{\beta}}^{(t)}, \mathbf{y}, \gamma]$. For the former, under the Laplacian prior, we have that:

$$p(\tau_i|\hat{\boldsymbol{\beta}}^{(t)}, \mathbf{y}, \gamma) \propto p(\hat{\boldsymbol{\beta}}^{(t)}|\tau_i)p(\tau_i|\gamma).$$

Then

$$\begin{aligned} \omega_i \equiv E[\tau_i^{-1}|\hat{\boldsymbol{\beta}}^{(t)}, \mathbf{y}, \gamma] &= \frac{\int_0^\infty \frac{1}{\tau_i} p(\tau_i|\gamma)p(\hat{\beta}^{(\mathbf{t})}|\tau_i)d\tau_i}{\int_0^\infty p(\tau_i|\gamma)p(\hat{\beta}^{(\mathbf{t})}|\tau_i)d\tau_i} \\ &= \gamma|\hat{\beta}^{(t)}|^{-1}. \end{aligned}$$

For the latter, $z_i$ is Gaussian with mean $\boldsymbol{\beta}^T h(x_i)$, but left-truncated at zero if $y_i = 1$ and right-truncated at zero of $y_i = -1$. Thus:

$$v_i \equiv E[z_i|\hat{\boldsymbol{\beta}}^{(t)}, \mathbf{y}, \gamma] =$$

$$\begin{cases} \boldsymbol{\beta}^T h(x_i) + \frac{N(\beta^T h(x_i))}{1-\Phi(-\beta^T h(x_i))} & \text{if } y_i = 1 \\ \boldsymbol{\beta}^T h(x_i) + \frac{N(\beta^T h(x_i))}{\Phi(-\beta^T h(x_i))} & \text{if } y_i = -1. \end{cases}$$

So the E-step replaces $\Gamma$ by its conditional expectation $\mathbf{\Omega} = \text{diag}(\omega_1, \ldots, \omega_d)$ and replaces $\mathbf{z}$ by $\mathbf{v} = (v_1, \ldots, v_n)^T$.

The M-step carries out a maximization with respect to $\boldsymbol{\beta}$ leading to:

$$\hat{\beta}^{(t+1)} = (\mathbf{\Omega} + \mathbf{H^T H})^{-1} \mathbf{H^T v}.$$

The EM procedure for the Jeffreys prior requires a minor modification of the above algorithm. Specifically:

$$\mathbf{\Omega} = \text{diag}\left(|\hat{\beta}_1^{(t)}|^{-2}, \ldots, |\hat{\beta}_d^{(t)}|^{-2}\right)$$

# 4 Learning Algorithms - ECM

One practical difficulty with the preceding algorithm is that each step requires inversion of a $d \times d$ matrix, where $d$ is the number of features representing the documents. For $d$ bigger than a few hundred this will be problematic. The Expectation

Conditional Maximization (ECM) algorithm (Meng and Rubin, 1993) deals with the problem by maximimizing the *beta* vector in disjoint blocks.

# 5 Results

At this stage we have conducted preliminary experiments using the ModApte version of Reuters−21578 collection of news stories. [1]. These data have the advantage of having many published analyses. The collection contains 7769 documents in the training set and 3019 in the testing set. We consider a subset of this dataset that consists of documents that belong to any of the 10 most populous categories. It has 6775 documents for training and 2258 for testing. We remove stopwords and punctuation marks before we do the analyses.

Table 1 shows the performance of our current algorithm in comparison with results for Naive Bayes, logistic regression, and support vector machines (SVM) from Zhang and Oles (2001). The sparse Bayes results use a simple feature selection algorithm that selects 300 features per category. Note this potentially places us at disadvantage when compared to the other algorithms' results shown in the Table; we expect that the performance of our algorithm will improve with judicious feature selection and/or dimensionality reduction. Nonetheless, our algorithm is performing reasonably well with the exception of the "interest" catgeory. Performance results here use the so-called "F1-measure," the geometric mean of precision and recall (Yang, 1999). Precision and recall are standard measure used in the text categorization literature:

$$\text{precision} = \frac{\text{true positive}}{\text{true positive} + \text{false positive}} \times 100,$$

$$\text{recall} = \frac{\text{true positive}}{\text{true positive} + \text{false negative}} \times 100.$$

# 6 Workplan

The immediate tasks we will undertake are:

[1] http://www.daviddlewis.com/resources/testcollections/reuters21578/

| Topic | Naive Bayes | Log. Reg. | SVM | Sparse Bayes 300 |
|---|---|---|---|---|
| earn | 96.6 | 98.4 | 98.1 | 95.6 |
| acq | 91.7 | 95.2 | 95.3 | 89.0 |
| money-fx | 70.0 | 75.2 | 74.4 | 76.8 |
| grain | 76.7 | 88.4 | 89.6 | 90.0 |
| crude | 84.1 | 85.9 | 84.8 | 84.5 |
| trade | 52.3 | 72.9 | 73.4 | 68.9 |
| interest | 68.2 | 78.2 | 75.9 | 60.1 |
| ship | 76.4 | 81.9 | 82.4 | 80.0 |
| wheat | 58.1 | 88.2 | 88.9 | 89.7 |
| corn | 52.4 | 88.7 | 86.2 | 86.9 |

Table 1: F1 Performance results for the ModApte version of Reuters-21578. The Naive Bayes, logistic regression, and SVM results used 10,000 features.

- Refine and validate the current implementation via EM and establish benchmark performance on multiple datsets.
- Experiment with different feature selection algorithms.
- Implement, validate, and refine the ECM algorithm for handling larger feature vectors.
- Establish benchmark performance for ECM.

The relevant tasks for the first quarter of 2003 include:

- Conduct experiments with different types of prior specifications derived from category descriptions.
- Design and implement hierarchical sparse Bayesian classifiers.
- Investigate non-stationary models.

## References


Albert, J.H. and Chib, S. (1993). Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, **88**, 669–679.

Chambers, E.A. and Cox, D.R. (1967). Discrimination between alternative binary response models. *Biometrika*, **54**, 573–578.

Figueiredo, M.A.T. (2001). Adaptive sparseness using Jeffreys prior. *Neural Information Processing Systems*, Vancouver, December 2001.

Figueiredo, M.A.T. and Jain, A.K. (2001). Bayesian learning of sparse classifiers. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Hawaii, December 2001.

Girosi, F. (1998). An equivlance between sparse approximation and support vector machines. *Neural Computation*, **10**, 1445–1480.

Meng, X.L. and Rubin, D.R. (1993). Maximum likelihood estimation via the ECM algorithm: a general framework. *Biometrika*, **80**, 267–278.

Tibshirani, R. (1995). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, **58**, 267-288.

Tipping, M.E. (2001). Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, **1**, 211-244.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization and retrieval. *Information Retrieval Journal*, **1**, 69–90.

Zhang, T. and Oles, F. (2001). Text categorization based on regularized linear classifiers. *Information Retrieval*, **4**, 5–31.