Sparse Logistic Regression for Text Categorization

Alexander Genkin
DIMACS, Rutgers University
96 Frelinghuysen Road
Piscataway, NJ 08854-8018
agenkin
@dimacs.rutgers.edu

David D. Lewis
David D. Lewis Consulting
858 W. Armitage Ave., #296
Chicago, IL 60614
sigir05pap
@daviddlewis.com

David Madigan
DIMACS and Department of Statistics
Rutgers University
96 Frelinghuysen Road
Piscataway, NJ 08854-8018
dmadigan@rutgers.edu

ABSTRACT

This paper studies regularized logistic regression and its application to text categorization. In particular we examine a Bayesian approach, lasso logistic regression, that simultaneously selects variables and provides regularization. We present an efficient training algorithm for this approach, and show that the resulting classifiers are both compact and have state-of-the-art effectiveness on a range of text categorization tasks.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing

General Terms

Algorithms, Experimentation

Keywords

text categorization, supervised learning, logistic regression, regularization, sparsity

1. INTRODUCTION

Feature selection, the deliberate discarding of some available predictor features before running a learning algorithm, is widely applied in machine learning tasks, including text classification. Feature selection may be used for a range of reasons [9], including computational efficiency in training, computational efficiency at classification time, avoiding overfitting and improving effectiveness of the learned classifier, and making the classifier and classification decisions more comprehensible to people.

There are, however, several downsides to feature selection as a preprocessing step. First, even when an individual feature selection mechanism has a clear statistical foundation, this is rarely true for the combined process of feature selection followed by a particular supervised learning algorithm. This makes it difficult or impossible to predict how effectiveness will change as training data increases, to choose the number of features in a principled way, to understand the reliability of predictions, or in general to make use of theoretical insights into the behavior of learning algorithms (admittedly weak as those often are). Second, most feature selection methods used in preprocessing consider each feature in isolation and thus may choose redundant or otherwise ineffective combinations of features. Those methods that do consider combinations tend to be expensive computationally, involving large numbers of runs of the core supervised learning algorithms. Finally, since the interactions between feature selection and learning algorithm are poorly understood, there is little to guide one in combining these feature selection methods with, for instance, domain knowledge, even when the learning algorithm itself can use such knowledge.

Recent advances in efficient, regularized learning algorithms, such as SVMs [15], boosting [31], and penalized logistic regression [39] have greatly reduced the need to use feature selection for efficient training or to avoid overfitting. However, feature selection is still often desired, particularly in applied contexts, to improve human interpretability and to reduce memory and computational requirements at prediction time [16, 26]. In what follows, we show that by using a particular regularized learning approach, these final two virtues of feature selection can be achieved in a single learning algorithm with clear statistical foundation.

Recent papers have demonstrated that regularized logistic regression provides outstanding predictive performance [39, 23] across a range of text classification tasks and corpora. The regularization these papers describe takes the form of a penalty on the L_2 norm of the regression parameter. This is equivalent to ridge regression [12] for the logistic model [30, 19]. In this work we turn to L_1 norm regularization, which simultaneously selects variables and provides regularization, leading to sparse models. L_1 penalization of logistic regression corresponds to the *lasso* algorithm [34] for linear regression. We present experimental results on 268 text categorization tasks on three corpora showing that lasso logistic regression systematically outperforms ridge logistic regression. We present an optimization algorithm for efficient fitting of lasso logistic regression models with 10's of thousands of predictors. Software implementing this algorithm has been made publicly available at

http://www.stat.rutgers.edu/∼madigan/BBR/.

We begin in Section 2 with the definition of regularized logistic regression, both ridge and lasso versions, and also give a Bayesian interpretation of those procedures. Section 3 describes the algorithm we use to fit our model to training data. Section 4 describes the data sets and methods we use in our experiments, and Sections 5 and 6 the experimental results. Specifically, Section 5 demonstrates that lasso logistic regression effectiveness higher than that of ridge logistic regression and very similar to that of SVMs. In the Section 6 we study the ability of lasso logistic regression to produce sparse models; we find those models yield higher effectiveness when they have the same sparsity as models produced by ridge logistic regression with traditional feature selection as a preprocessing step.

2. LOGISTIC REGRESSION WITH REGULARIZATION

We want to learn a text classifier, $y = f(\boldsymbol{x})$, from a set of training examples $D = \{(\boldsymbol{x_1}, y_1), \ldots, (\boldsymbol{x_i}, y_i), \ldots, (\boldsymbol{x_n}, y_n)\}$. For text categorization, the vectors $\boldsymbol{x_i} = [x_{i1}, ..., x_{ij}, ..., x_{id}]^T$ comprise transformed word frequencies from documents (Section 4.1). The values $y_i \in \{-1, +1\}$ are class labels encoding membership $(+1)$ or nonmembership (-1) of the vector in the category.[1]

Logistic regression is a conditional probability model of the form

$$p(y_i = +1|\boldsymbol{\beta}, \boldsymbol{x_i}) = \frac{1}{1 + \exp(-\boldsymbol{\beta}^T \boldsymbol{x_i})}. \quad (1)$$

For a text categorization problem, $p(y = +1|\boldsymbol{x_i})$ corresponds to the probability that the ith document belongs to the category. The decision of whether to assign the category can be based on comparing the probability estimate with a threshold or, more generally, based on maximizing the expected effectiveness [4, 20].

Maximum likelihood estimation of the parameters $\boldsymbol{\beta}$ is equivalent to minimizing the negated log-likelihood:

$$l(\boldsymbol{\beta}) = -\sum_{i=1}^{n} \ln(1 + \exp(-\boldsymbol{\beta}^T \boldsymbol{x_i} y_i), \quad (2)$$

Correspondingly, finding the ridge logistic regression parameters is done by minimizing:

$$l_{ridge}(\boldsymbol{\beta}) = l(\boldsymbol{\beta}) + \lambda \sum_{j=1}^{d} \beta_j^2, \quad (3)$$

whereas lasso logistic regression requires minimization of:

$$l_{lasso}(\boldsymbol{\beta}) = l(\boldsymbol{\beta}) + \lambda \sum_{j=1}^{d} |\beta_j|, \quad (4)$$

where λ is a hyperparameter controlling degree of regularization.

One may view these same techniques from a Bayesian point of view by using a prior distribution for $\boldsymbol{\beta}$. The simplest approach is to impose a univariate Gaussian prior with mean zero and variance $\tau > 0$ on each parameter β_j:

$$p(\beta_j|\tau) = N(0, \tau) = \frac{1}{\sqrt{2\pi\tau}} \exp(\frac{-\beta_j^2}{2\tau}). \quad (5)$$

[1]We encode class labels as $-1/+1$ rather than $0/1$ to simplify presentation of our fitting algorithm.

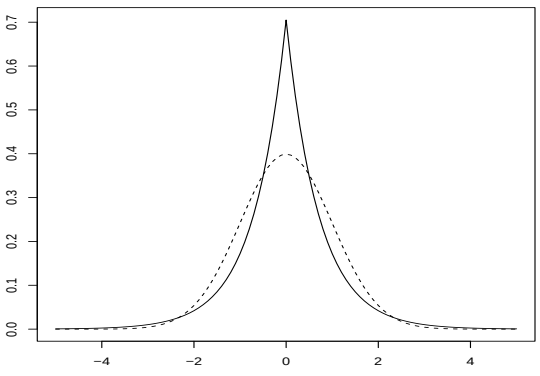

Figure 1: The density of the Laplace and Gaussian (dashed line) distributions with the same mean and variance.

The mean of zero encodes a prior belief that β_j will be near zero. The variance τ is a positive constant we must specify. A small value of τ represents a strong prior belief that β_j are close to zero. A large value of τ represents a weaker such belief. We assume *a priori* that the components of $\boldsymbol{\beta}$ are independent and hence the overall prior for $\boldsymbol{\beta}$ is the product of the priors for each of its component β_j's. Finding the maximum *a posteriori* (MAP) estimate of $\boldsymbol{\beta}$ with this prior is equivalent to ridge logistic regression (3) with $\lambda = 1/(2\tau)$.

Ridge logistic regression has been widely used in text categorization, see for example [39, 23, 38]. But a Gaussian prior, while favoring values of β_j near zero, does not favor β_j's being exactly equal to zero. Absent unusual patterns in the data, the MAP estimates of all or almost all β_j's will be non-zero. To obtain sparse text classifiers with a Gaussian prior, previous authors have used various feature selection mechanisms - see, for example, [39] and [38].

Ad hoc feature selection is not necessary, however, if we simply choose the right prior. Suppose we use the double exponential (Laplace) prior distribution:

$$p(\beta_j|\lambda) = \frac{\lambda}{2} \exp(-\lambda|\beta_j|). \quad (6)$$

As before, the prior for $\boldsymbol{\beta}$ is the product of the priors for its components. MAP estimation in this case is exactly lasso logistic regression (4). For typical data sets and choices of λ, most parameters in the MAP estimate for $\boldsymbol{\beta}$ can be zero.

Figure 1 shows the density of the Laplace distribution, with the suggestive cusp at zero. Tibshirani [34] was the first to suggest Laplace priors in the regression context. He pointed out that the MAP estimates using the Laplace prior are the same as the estimates produced by the lasso algorithm. Since then the use of constraints or penalties based on the absolute values of coefficients has been used to achieve sparseness in a variety of data fitting tasks (see, for example, [5, 6, 8, 35, 33]).

Figures 2 and 3 show the effect of hyperparameter settings on the logistic regression parameters on a particular data set with eight predictor variables. Figure 2 shows the effect of a Gaussian prior distribution on each parameter. When τ is small, each β_j has a small prior variance, and

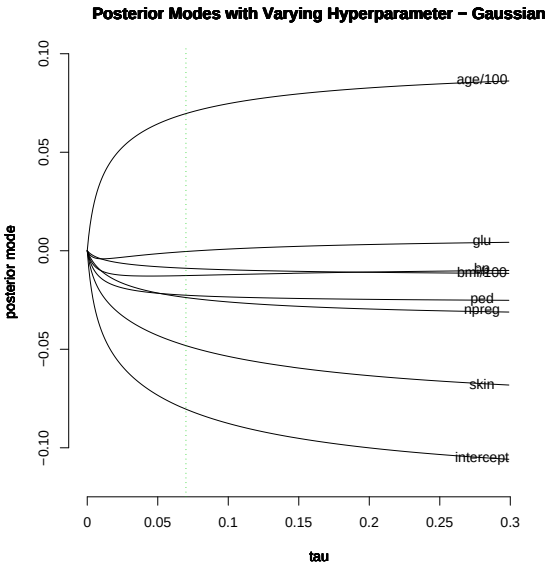

Figure 2: MAP estimates (y-axis) of the eight parameters of a logistic regression as the hyperparameter τ of a Gaussian prior on those parameters varies (x-axis). For example, the vertical dotted line shows the eight estimates corresponding to a particular choice of τ.

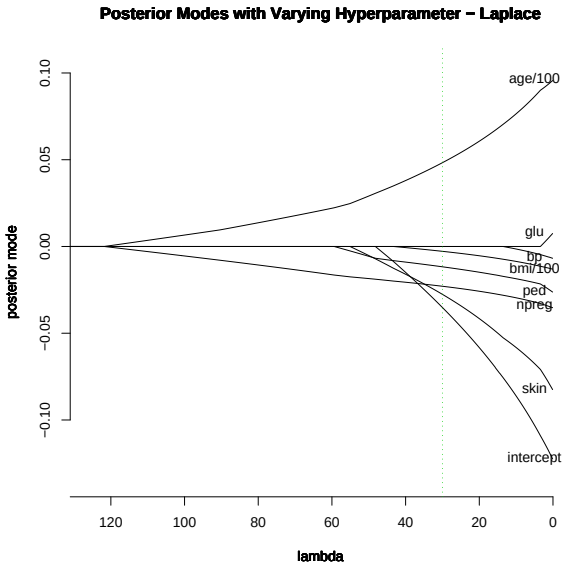

Figure 3: MAP estimates of logistic regression parameters on the same data set as Figure 2, shown for varying values of a Laplace prior hyperparameter λ. For example, the vertical dotted line shows the eight estimates corresponding to a particular choice of λ, a choice that selects six of the eight predictor variables.

the resulting MAP estimates are small, approaching zero as $\tau \to 0$. When τ is large, the MAP estimates are similar to the maximum likelihood estimates. The vertical dashed line corresponds to $tau = 0.01$. Figure 3 shows the equivalent picture for the Laplace prior. As with the Gaussian prior, the MAP estimates range from all zeroes to the maximum likelihood estimates. However, unlike the Gaussian case, particular choices for the hyperparameter lead to MAP estimates where some components of $\boldsymbol{\beta}$ are zero while others are not. The vertical dashed line corresponds to prior distributions with a variance of 0.01. Note that with this particular choice for the hyperparameter, the posterior modes of two of the parameters are zero. Hastie *et al.* [10] show similar plots for linear regression.

3. ALGORITHM

For maximum likelihood logistic regression the most common optimization approach in statistical software is some variant of the multidimensional Newton-Raphson method [3]. Newton algorithms have the advantage of converging in very few iterations. For high-dimensional problems such as text categorization, however, Newton algorithms have the serious disadvantage of requiring $O(d^2)$ memory, where d is the number of model parameters. A variety of alternate optimization approaches have therefore been explored for maximum likelihood logistic regression, and for MAP logistic (or probit) regression with a Gaussian prior. Some of these algorithms, such as limited memory BFGS [27], conjugate gradient [27], and hybrids of conjugate gradient with other methods [18], compute the gradient of the objective function at each step. This requires only $O(d)$ memory (in addition to the data itself). Efron *et al.* [5] describe a new class of "least angle" algorithms for lasso linear regression and related models. Madigan and Ridgeway [25] discuss possible extensions to the logistic model.

Other methods solve a series of partial optimization problems. Some of these methods use the subproblems to maintain an evolving approximation to the gradient of the full objective, which still requires $O(d)$ memory. Others use each subproblem only to make progress on the overall objective, using only constant memory beyond that for the parameter vector. The smaller problems may be based on processing one example at a time, as in dual Gauss-Seidel [39], stochastic gradient descent [1], exponentiated gradient descent [17], and Bayesian online algorithms [2, 28]. Or the one dimensional subproblems may be based on processing one parameter at a time, as in iterative scaling [13], and cyclic coordinate descent [39, 33]. Some of these algorithms have already shown promise on text categorization or other language processing tasks.

Of these, we base our work on the cyclic coordinate descent algorithm [39] due to its speed and simplicity of implementation. This algorithm requires an objective function that is convex and smooth, namely having a continuous first order derivative and a piece-wise continuous second order derivative. This holds for (3) but not for (4), which is still convex, but does not have a finite derivative at 0. So modifications are needed to adapt the algorithm for the lasso logistic regression. We first present the Algorithm *CLG* from [39], clarifying some details omitted in their presentation, and making clear our implementation and tuning choices where they presented several alternatives. We then present our modification of CLG to support lasso logistic regression.

3.1 Ridge case

As the name "cyclic coordinate descent" suggests, each step of the algorithm approximately minimizes the objective along one coordinate before going on to the next, in a fashion that guarantees the overall optimum is approached. The way it's done guarantees the convergence and speed of the algorithm. The idea is to construct an upper bound on the second derivative of the objective on an interval around the current value; since the objective is convex, this will give rise to the quadratic upper bound on the objective itself on that interval. Minimizing the quadratic bound on the interval gives one step of the algorithm.

Let $f(\beta_j)$ be the objective expressed as the function of only one component β_j with all the rest being fixed, and $Q(\beta_j, \Delta_j)$ be an upper bound on the second derivative of $f(\cdot)$ in the Δ_j-vicinity of β_j, $\Delta_j > 0$:

$$Q(\beta_j, \Delta_j) \geq f''(\beta_j + \delta) \text{ for all } \delta \in [-\Delta_j, \Delta_j].$$

The minimum of the quadratic upper bound on f will then be located at $\beta_j + \Delta v_j$ where

$$\Delta v_j = -f'(\beta_j)/Q(\beta_j, \Delta_j)$$

This is a tentative step, we now need to check if Δv_j happens to fall inside the interval $[-\Delta_j, \Delta_j]$; if not, the step is reduced to the interval boundary.

For the objective in (3) the upper bound takes the form:

$$Q(\beta_j, \Delta_j) = \sum_i x_{ij}^2 F(\boldsymbol{\beta}^T \boldsymbol{x_i} y_i, \Delta_j x_{ij}) + 2\lambda,$$

where the function $F(r, \delta)$ is the upper bound on the second derivative of the logistic c.d.f. in the interval $[r - |\delta|, r + |\delta|]$. We use the tight upper bound:

$$F(r, \delta) = \begin{cases} 0.25, & \text{if } |r| \leq |\delta| \\ 1/(2 + \exp(|r| - |\delta|) + \exp(|\delta| - |r|)), & \text{otherwise.} \end{cases}$$

Now the tentative step computation takes the form:

$$\Delta v_j = \frac{\sum_i \frac{x_{ij} y_i}{1+\exp(\boldsymbol{\beta}^T \boldsymbol{x_i} y_i)} - 2\beta_j \lambda}{\sum_i F(\boldsymbol{\beta}^T \boldsymbol{x_i} y_i, \Delta_j x_{ij}) x_{ij}^2 + 2\lambda}, \quad (7)$$

The algorithm is presented in Figure 4. The size of the approximating interval Δ_j is updated as suggested by [39]. The convergence criterion in our implementation is $\sum_i |\Delta r_i| / (1 + \sum_i |r_i|) \leq 5 \cdot 10^{-4}$. See [39] for some other choices of stopping criteria and update rules for Δ_j.

3.2 Lasso case

The objective in the lasso case (4) is convex and satisfies the above formulated smoothness requirements everywhere except at 0. So the algorithm will work correctly as long as we operate within positive numbers only or within negative numbers only. Within these limits the tentative step computation at step 4 in case of Laplace prior takes the form:

$$\Delta v_j = \frac{\sum_i \frac{x_{ij} y_i}{1+\exp(\boldsymbol{\beta}^T \boldsymbol{x_i} y_i)} - \lambda s}{\sum_i x_{ij}^2 F(\boldsymbol{\beta}^T \boldsymbol{x_i} y_i, \Delta_j x_{ij})}. \quad (8)$$

where $s = \beta_j / |\beta_j|$ and $\beta_j \neq 0$. At each tentative step we have to check if this step is trying to cross over 0, and in that case reduce it to 0. In case when the current value of β_j is 0 we have to try both directions and see if either of them gives an improvement to the objective (could not be both due to the convexity). If neither works then β_j stays at 0 for the iteration. This change affects only Step 4 of the Algorithm *CLG*. In Figure 5 we present the modified Step 4 to be plugged into Algorithm *CLG*. The resulting algorithm is highly efficient. For instance, training time for a single category on the ModApte data set (9,603 training examples, 18,979 model parameters, Section 4.1) averages 10.7 seconds with a Gaussian prior and 17.5 seconds with a Laplace prior. Shevade and Keerthi have recently presented a similar algorithm for logistic regression with an L_1 penalty [33].

```
(1)   initialize β_j ← 0, Δ_j ← 1 for j = 1, ..., d
(2)   for k = 1, 2, ... until convergence
(3)       for j = 1, ..., d
(4)           compute tentative step Δv_j by formula (7)
(5)           Δβ_j ← min(max(Δv_j, −Δ_j), Δ_j)
                (reduce the step to the interval)
(6)           β_j ← β_j + Δβ_j
(7)           Δ_j ← max(2|Δβ_j|, Δ_j/2)
(8)       end
(9)   end
```

Figure 4: Algorithm *CLG* for fitting ridge logistic regression.

```
...
(4.1)     if β_j = 0
(4.2)         s ← 1 (try positive direction)
(4.3)         compute Δv_j by formula (8)
(4.4)         if Δv_j ≤ 0 (positive direction failed)
(4.5)             s ← −1 (try negative direction)
(4.6)             compute Δv_j by formula (8)
(4.7)             if Δv_j ≥ 0 (negative direction failed)
(4.8)                 Δv_j ← 0
(4.9)             endif
(4.10)        endif
(4.11)    else
(4.12)        s ← β_j/|β_j|
(4.13)        compute Δv_j by formula (8)
(4.14)        if s(β_j + Δv_j) < 0 (cross over zero)
(4.15)            Δv_j ← −β_j
(4.16)        endif
(4.17)    endif
...
```

Figure 5: Algorithm for fitting lasso logistic regression: replacement for Step 4 in Algorithm *CLG*.

4. EXPERIMENTAL METHODS

To study whether lasso logistic regression provides an efficient and effective text categorization approach, we tested it on several large data sets. In this section we discuss our experimental methods: how texts were represented as numeric vectors, what collections of documents and category labels were used, and how effectiveness was measured (and

optimized). We also discuss two state-of-the-art text categorization approaches with which we compared lasso logistic regression: support vector machines, and ridge logistic regression combined with feature selection.

4.1 Datasets

Our experiments used three standard text categorization test collections. The first collection was the ModApte subset of the Reuters−21578 collection of news stories [22].[2] The ModApte subset contains 9,603 Reuters news articles in the training set, and 3,299 in the test set. Documents in the ModApte data set belong to 0 or more of a set of 90 "Topic" categories corresponding to news areas of economic interest. (We used the 90 Topic categories that have at least one positive training example and one positive test example on the ModApte subset.) There were 21,989 unique terms in the ModApte data set, 18,978 of which occur in the training set and thus potentially have nonzero parameters in a classifier.

The second data set was RCV1-v2[3], a test categorization test collection of 804,414 newswire stories based on data released by Reuters, Ltd.[4] We used the LYRL2004 training/test split ([22]) of RCV1-v2, which has 23,149 training documents and 781,265 test documents. However, for efficiency reasons we used a fixed, random, roughly 10% subset (77,993 documents) of the test documents as our test set in all experiments.

We used 101 RCV1-v2 "Topic" categories that have at least one positive training example. (The Topic categories in RCV1-v2 are different from those in Reuters-21578, and cover a broader range of news types.) For our text representation we used stemmed token files distributed with RCV1-v2. A total of 47,152 unique terms were present in the training set, and 288,062 unique terms in the union of the training set and the 77,993 document test set.

The third data set consists of Medline records from the years 1987 to 1991, formatted for the SMART retrieval system, and distributed as part of the OHSUMED text retrieval test collection [11]. [5] Of the 348,566 OHSUMED records, we used the 233,445 records where the title, abstract, and MeSH (Medical Subject Headings) category fields were all nonempty. We used the 83,944 such documents from the years 1987 and 1988 as our training set, and the 149,501 such documents from the years 1989 to 1991 as our test set.

We did not use the queries and relevance judgments distributed with the collection. Instead, we based binary classification tasks on predicting the presence or absence of MeSH controlled vocabulary terms in the records [24]. We used the same 77 categories as [21]. These were drawn from a set of 119 MeSH Heart Disease categories extracted by Yiming Yang from the April 1994 (5th Ed.) UMLS CD-ROM, distributed by the National Library of Medicine in the United States. All text from the $.T$ (title) and $.W$ (abstract) fields was used in computing the TF weights. There were 73,269 distinct terms in the training set, and 122,076 in the union of the training and test sets.

Text processing for all three collections was done using Lemur toolkit[6], which performed a simple tokenization into words (using its TrecParser module), discarded words from the SMART stopword list of 572 words[7], and applied the Lemur variant of the Porter stemmer (Porter, 1980, 2003) to remove word endings. All stems from text in the `<TITLE>` and `<BODY>` SGML elements were combined to produce raw TF weights.

For training and classification purposes, we represent each document as a vector of term weights using a form of *log TF* $\times$ *IDF* (term frequency times inverse document frequency) weighting [22]. All IDF weights were computed on the training set. Then cosine normalization was applied [29].

In addition to one parameter for each term, our vectors include a constant term, 1.0. (The constant term is not taken into account during cosine normalization, and is not changed by it.) The presence of a constant term is usual in statistical practice, but often omitted in text categorization studies. The model parameter for the constant term (i.e. the intercept) is included in the regularization penalty.

4.2 Benchmark Algorithms

We compared the effectiveness of lasso logistic regression to two state-of-the-art approaches to text categorization: logistic regression with feature selection and support vector machines.

To produce sparse text classifiers when using ridge logistic regression, it is common to attempt to discard low quality features before model fitting. We tested three traditional feature selection methods in combination with ridge logistic regression, as well as trying no feature selection whatsoever.

Each feature selection approach is based on computing some quality measure for each feature, ranking features by that measure, and then using only the top-ranked features when learning a classifier. A different set of features was chosen for each category. We tested each method at choosing feature sets with 5, 50, or 500 features, with the same number of features used for all categories. The intercept term was always used, so the total number of model parameters was 6, 51, and 501, respectively.

The first feature quality measure was the chi-square test for independence between two variables. As a feature selection measure, it chooses the features that are least independent from the class label, and is widely used in text categorization [37, 32].

Our second measure, bi-normal separation (BNS), was the best measure on several criteria in a recent comparison of feature selection methods for text categorization [7]. Forman defines it as:

$$B(j) = |\Phi^{-1}(\frac{a}{a+b}) - \Phi^{-1}(\frac{c}{c+d})| \qquad (9)$$

where Φ is the standard normal cumulative distribution function. Forman provides a justification of BNS in term of ROC (receiver operating characteristic) analysis.

Most feature selection measures studied in text classification research (including the two discussed above) take into account only the binary presence or absence of terms in documents. In contrast, the most effective text representations are non-binary ones such as *TF* $\times$ *IDF* weighting. Our third measure, the Pearson product-moment correlation, makes

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

[3] *http://www.ai.mit.edu/projects/jmlr/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm*

[4] *http://about.reuters.com/researchandstandards/corpus/*

[5] *ftp://medir.ohsu.edu/pub/ohsumed/*

[6] *http://www-2.cs.cmu.edu/~lemur/*

[7] Available at *ftp://ftp.cs.cornell.edu/pub/smart/english.stop* or as part of the RCV1-v2 data set.

	sum of positive ranks	sum of negative ranks	two-sided p-value (Wilcoxon)
lasso - ridge	1667	163	.000
lasso - SVM	969.5	921.5	.863
ridge - **SVM**	204.5	1811.5	.000

Table 1: Comparison of lasso and ridge logistic regression and SVM on ModApte corpus using Wilcoxon paired signed-ranks test. For all topic categories, the differences between $F1$ values are calculated and ranked from smallest to largest by absolute value. Them sum of positive ranks (first method has greater $F1$) and sum of negative ranks (second method has greater $F1$) are compared. Algorithms significantly better at the $p = 0.050$ level are indicated in bold.

use of the values of within document weights in choosing features. It has been used for feature selection in a variety of machine learning tasks [9].

The support vector machine algorithm for learning linear classifiers has consistently been one of the most effective approaches in text categorization studies [22]. It also produces models with a form of instance sparseness that may or may not translate into sparseness of linear model coefficients. We trained a single SVM classifier for each category using Version 5.00 of *SVM_Light* [14, 15].[8] All *SVM_Light* parameters, except the regularization parameter C (see below), were set at their default values.

Regularization parameter values for all modeling algorithms: SVM (C parameter for *SVM_Light*), ridge, and lasso logistic regression (λ in (3) and (4)) – were chosen through cross-validation. After parameter fitting, the threshold for each classifier was tuned to minimize the number of training set errors on that category. This is a common approach in tuning SVMs for the unbalanced training sets seen in text categorization. We adopt it for comparability, though it means we do not take advantage of logistic regression's ability to optimize the expected value of the actual effectiveness measure being used.

5. EFFECTIVENESS OF MODELS

Our first set of experiments compared the effectiveness of lasso logistic regression with ridge logistic regression and SVMs. For each category and for each algorithm we computed the $F1$ effectiveness measure ([36, 22]). Tables 1, 2, and 3 show the number of categories on which algorithm A had greater, less, or the same value of $F1$ as algorithm B.

As a significance test, we looked at the category-wise difference in $F1$ between pairs of algorithms for each data set and applied the 2-tailed Wilcoxon paired signed-ranks test. By this measure, lasso logistic regression and SVM are significantly better than ridge logistic regression on all data sets; SVM is better than lasso logistic regression on two of three data sets, but in all three cases the difference is statistically insignificant.

To get a sense of overall effectiveness, we computed macroaveraged $F1$ for each algorithm/collection pair, see Table 4. The conclusion that we draw from this table is basically the same: lasso logistic regression and SVM produce results close to each other and significantly outperform ridge logistic regression.

[8] *http://svmlight.joachims.org/*

	sum of positive ranks	sum of negative ranks	two-sided p-value (Wilcoxon)
lasso - ridge	3925	926	.000
lasso - SVM	2112	2739	.267
ridge - **SVM**	1031	3625	.000

Table 2: Comparing $F1$ on RCV1-v2 corpus. Details as in Table 1.

	sum of positive ranks	sum of negative ranks	two-sided p-value (Wilcoxon)
lasso - ridge	2359	416	.000
lasso - SVM	1462	1239	.540
ridge - **SVM**	599	2102	.000

Table 3: Comparing $F1$ on OHSUMED corpus. Details as in Table 1.

6. SPARSITY OF MODELS

The number of features the lasso model selected (starting with the full feature set) varied from category to category, but was always a small proportion of the full feature set. The number of selected features ranged from 0 to 511 for ModApte, from 3 to 1737 for RCV1-v2, and from 0 to 965 for OHSUMED. Figure 6 provides more details.

Consider now a situation that requires both good effectiveness and sparsity at the same time. Figure 6 shows that the number of features in the lasso model varies widely and is hard to predict. In order to allow a direct comparison we decided to use L_1 penalty for not only the regularization, but also as a device to reduce the number of features in the model to the requested limit. This resulted in the algorithm we called *Squeezer*.

Squeezer first builds the lasso model with cross-validated hyperparameter selection. If the number of features in the model turns out to be within the requested limit, the algorithm stops. (Trying to relax the hyperparameter in order to increase the number of features up to the limit would result in overfitted model.)

If the number of features in the model exceeds the requested limit, squeezer gradually increases the hyperparameter value until the number of features falls within the limit. Finally, binary search is performed in the interval between

	ModApte	RCV1-v2	OHSUMED
lasso	52.03	56.54	51.30
ridge	39.71	51.40	42.99
SVM	52.09	57.26	49.35

Table 4: Macroaveraged $F1$ measure for lasso logistic regression, ridge logistic regression, and support vector machines on three text categorization test collections.

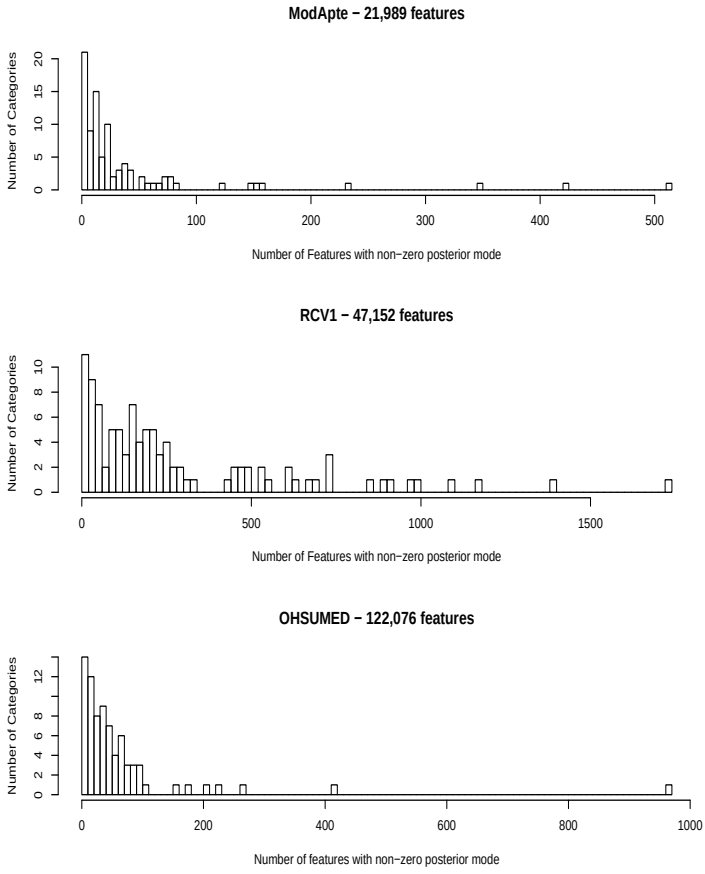

Figure 6: Number of features that the lasso selected for each of the three datasets (ModApte, RCV1-v2, and OHSUMED).

the two most recent hyperparameter values until the smallest hyperparameter value is found that results in the model with the number of features within the limit.

Now we can compare directly the effectiveness of different models with limited number of features. Table 5 compares squeezer against ridge logistic regression with three feature pre-selection methods. Squeezer demonstrates consistently higher effectiveness than the three feature pre-selection methods for target feature set sizes of 500 and 50, and is roughly equivalent for feature set sizes of 5.

7. CONCLUSION AND FUTURE WORK

We demonstrated in our experiments with three text collections that lasso logistic regression yields consistently higher effectiveness than ridge and is very close to SVM. At the same time it offers the substantial advantage of sparsity of the fitted model. Using our *Squeezer* algorithm, we compared lasso logistic regression models against ridge used with traditional feature pre-selection procedures to obtain a requested level of sparsity, and we found lasso models to yield higher effectiveness in that case also.

Our current work-in-progress focuses on taking advantage of the Bayesian interpretation of regularized algorithms. One direction is to combine prior domain knowledge with learning from training data. The simplest case would be to give rare words priors with a higher variance, reflecting that they have higher content than more common words. This would be a more principled alternative to the usual term weighting heuristics (Section 4.1). The other direction is to automate hyperparameter selection using so-called marginal probability approach ([30]) and avoid tedious cross-validation.

	Number of features	500	50	5
ModApte	squeezer	51.91	51.98	47.71
	ridge + BNS	42.09	50.03	48.87
	ridge + chi-square	38.63	47.29	46.65
	ridge + correlation	38.38	47.41	46.85
RCV1-v2	squeezer	56.15	51.83	28.68
	ridge + BNS	50.89	43.97	30.60
	ridge + chi-square	50.57	45.60	32.31
	ridge + correlation	50.13	47.37	35.63
OHSUMED	squeezer	51.35	50.56	43.03
	ridge + BNS	43.46	44.27	42.40
	ridge + chi-square	42.38	45.18	36.55
	ridge + correlation	41.73	43.65	38.53

Table 5: Macroaveraged $F1$ measure for different algorithms resulting in the fixed number of features in the model.

Acknowledgements

We thank Reuters, Ltd., Bill Hersh, and the National Library of Medicine for making available data sets used in this research. The KD-D group supported this work through National Science Foundation grant "Monitoring Message Streams" EIA-0087022 to Rutgers University. The NSF also partially supported Madigan and Lewis's work through ITR grant DMS-0113236. We thank Andrei Anghelescu, Susana Eyeramendy, Paul Kantor, Vladimir Menkov, and Fred Roberts for suggestions, comments, or help with data sets.

8. REFERENCES

[1] C. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.

[2] H. T. Chai, K. M. A. Ng and H. L. Chieu. Bayesian online classifiers for text classification and filtering. In *Proceedings of SIGIR 2002: The Twenty-Fifth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 97–104, 2002.

[3] J. E. Dennis Jr. and R. B. Schnabel. A view of unconstrained optimization. In G. L. Nemhauser, A. H. G. R. Kan, and M. J. Todd, editors, *Optimization*. Elsevier, Amsterdam, 1989.

[4] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley-Interscience, New York, 1973.

[5] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Ann. Statist.*, 32(2):407499, 2004.

[6] M. A. T. Figueiredo. Adaptive sparseness for supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1150–1159, 2003.

[7] G. Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of*

Machine Learning Research, 3(3):1289–1305, March 2003.

[8] F. Girosi. An equivlance between sparse approximation and support vector machines. *Neural Computation*, 10:1445–1480, 1998.

[9] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3(3):1157–1182, March 2003.

[10] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data mining, Inference and Prediction*. Springer, New York, 2001.

[11] W. Hersh, C. Buckley, T. J. Leone, and D. Hickman. Ohsumed: an interactive retrieval evaluation and new large test collection for research. In *SIGIR '94: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 192–201, 1994.

[12] A. Hoerl and R. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1970.

[13] R. Jin, R. Yan, J. Zhang, and A. Hauptmann. A faster iterative scaling algorithm for conditional exponential model. In *International Conference on Machine Learning (ICML 2003)*, 2003.

[14] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Machine Learning: ECML'98, 10th European Conference on Machine Learning*, pages 137–142, 1998.

[15] T. Joachims. *Learning to Classify Text using Support Vector Machines*. Kluwer, 2002.

[16] J. Kittler. Feature selection and extraction. In T. Y. Young and K.-S. Fu, editors, *Handbook of Pattern Recognition and Image Processing*, pages 59–83. Academic Press, Orlando, 1986.

[17] J. Kivinen and M. K. Warmuth. Relative loss bounds for multidimensional regression problems. *Machine Learning*, 45(3):301–329, 2001.

[18] P. Komarek and A. Moore. Fast robust logistic regression for large sparse datasets with binary outputs. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.

[19] S. le Cessie and J. van Houwelingen. Ridge estimators in logistic regression. *Applied Statistics*, 41:191–201, 1997.

[20] D. D. Lewis. Evaluating and optimizing autonomous text classification systems. In *SIGIR '95: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 246–254, 1995.

[21] D. D. Lewis, R. Schapire, J. Callan, and R. Papka. Training algorithms for linear text classifiers. In *Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval*, pages 298–306. ACM Press, 1996.

[22] D. D. Lewis, Y. Yang, T. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, pages 361–397, 2004.

[23] F. Li and Y. Yang. A loss function analysis for classification methods in text categorization. In *The Twentith International Conference on Machine Learning (ICML'03)*, pages 472–479, 2003.

[24] H. J. Lowe and G. O. Barnett. Understanding and using the medical subject headings (mesh) vocabulary to perform literature searches. *Journal of the American Medical Association*, 271(14):1103–1108, 1994.

[25] D. Madigan and G. Ridgeway. Discussion of least angle regression. *Annals of Statistics*, to appear:*http://www.stat.rutgers.edu/∼madigan/PAPERS/lars3.pdf*, 2004.

[26] T. Mitchell and J. Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83:1023–1032, 1988.

[27] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 1999.

[28] G. Ridgeway and D. Madigan. A sequential monte carlo method for bayesian analysis of massive datasets. *Journal of Knowledge Discovery and Data Mining*, to appear:*http://www.stat.rutgers.edu/∼madigan/PAPERS/ridgeway2.pdf*, 2004.

[29] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[30] T. Santner and D. Duffy. *The Statistical Analysis of Discrete Data*. Springer-Verlag, 1989.

[31] R. E. Schapire and Y. Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[32] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34:1–47, 2002.

[33] S. K. Shevade and S. S. Keerthi. A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, 19:2246–2253, 2003.

[34] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58:267–288, 1996.

[35] M. Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, June 2001.

[36] C. J. van Rijsbergen. *Information Retrieval, 2nd Ed.* Butterworths, 1979.

[37] Y. Yang and J. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of ICML-97, 14th International Conference on Machine Learning ICML97*, pages 412—420, 1997.

[38] J. Zhang and Y. Yang. Robustness of regularized linear classification methods in text categorization. In *Proceedings of SIGIR 2003: The Twenty-Sixth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 190–197, 2003.

[39] T. Zhang and F. Oles. Text categorization based on regularized linear classifiers. *Information Retrieval*, 4(1):5–31, April 2001.