CS535D Project: Bayesian Logistic Regression through Auxiliary Variables

Mark Schmidt

Abstract

This project deals with the estimation of Logistic Regression parameters. We first review the binary logistic regression model and the multinomial extension, including standard MAP parameter estimation with a Gaussian prior. We then turn to the case of Bayesian Logistic Regression under this same prior. We review the cannonical approach of performing Bayesian Probit Regression through auxiliary variables, and extensions of this technique to Bayesian Logistic Regression and Bayesian Multinomial Regression. We then turn to the task of feature selection, outlining a trans-dimensional MCMC approach to variable selection in Bayesian Logistic Regression. Finally, we turn to the case of estimating MAP parameters and performing Bayesian Logistic Regression under L1 penalties and other sparsity promoting priors.

1 Introduction

In this project, we examined the highly popular Logistic Regression model. This model has traditionally been appealing due to its performance in classification, the potential to use its outputs as probabilitic estimates since they are in the range $[0, 1]$, and the interpretation of the coefficients in terms of the 'log-odds' ratio [1]. It is especially popular in biostatistical applications where binary classification tasks occur frequently [1]. In this first part of the report, we review this model, its multi-class generalization, and standard methods of performing maximum likelihood (ML) or maximum *a posteriori* (MAP) parameter estimation under a zero-mean Gaussian prior for the regression coefficients. We then turn to the case of obtaining Bayesian posterior density estimates of the regression coefficients. In particular, we examine the recently proposed extensions of the Bayesian Probit Regression auxiliary variable model to the Logistic Regression and Multinomial Regression scenarios. Finally, we turn to the challenging task

of incorporating feature selection into these models, focusing on trans-dimensional sampling methods, and MAP and/or Bayesian estimation under priors that encourage sparsity.

1.1 Binary Logistic Regression Model

We use X to denote the n by p design matrix, containing p features measured for n instances. We use y to denote the length n class label vector, where the values take on either $+1$ or -1, corresponding to the class label for the n^{th} instance. Finally, we will use w to represent the length p vector of parameters of the model. Primarily for ease of presentation, we will not address the 'bias' term w_0 in this document, but all techniques herein are easily modified to include a bias term. Under the standard (binary) Logistic Regression model, we express the probability that an instance i belongs to the class $+1$ as:

$$\pi(y_i = +1|x_i, w) = \frac{1}{1 + exp(-w^T x_i)} \tag{1}$$

For binary reponses, we can compute the probability of the 'negative' class using the sum rule of probability: $\pi((y_i = -1|x_i, w) = 1 - \pi(y_i = +1|x_i, w)$. We typically assume independent Gaussian priors with means of 0 and variance of v on the coefficients of the model:

$$w_i \sim \mathbf{N}(0, v) \tag{2}$$

To perform MAP parameter estimation, we take the log of the likelihood (1) over all examples, times the prior (2) over all parameters (ignoring the normalizing constant) to give the following objective function:

$$f = -\sum_{i=1}^{n} log(1 + exp(-y_i w^T x_i)) - \frac{1}{2v^2} w^T w \tag{3}$$

From this expression, we see that the Maximum Likelihood estimate is obtained by setting v to ∞. Differentiating the above with respect to w to we obtain the following expressions for the gradient and Hessian (using σ to denote the sigmoid function $\sigma(x) = 1/(1 + exp(-x))$:

$$g = -\sum_{i=1}^{n}(1 - \sigma(y_i w^T x_i))y_i x_i - \frac{w}{v^2} \tag{4}$$

$$H = -\sum_{i=1}^{n}\sigma(w^T x_i)(1 - \sigma(w^T x_i))x_i x_i^T - \frac{1}{v^2}I_p \tag{5}$$

We note that the Hessian is negative-definite and subsequently that the original function is (log)concave, indicating that any local maximizer of this objective will be a global maximizer. A simple method to maximize this objective is to repeat Newton iterations starting from an initial value of w until the norm of the gradient is sufficiently small (noting that that the gradient will be 0 at a maximizer). This results in a simple fixed-point iterative update as follows:

$$w = w + \alpha H_m^{-1} g \tag{6}$$

Where H_m is a modification of the Hessian to be sufficiently negative-definite, or a negative-definite approximation to the (inverse) Hessian (see [2]). The step size α can be set to 1, but convergence may be hastened by using line search methods satisfying sufficient descent conditions (see [3] or [4]). We have implemented an approach of this type making use of Matlab's 'fminunc' function in the directory 'LOGREG', and an example calling this code is included as 'example_LOGREG.m'. Other methods for optimizing this objective are discussed and compared in [4].

1.2 Multinomial Logistic Regression Model

The binary Logistic Regression model has a natural extension to the case where the number of classes, K, is greater than 2. This is done using the softmax generalization [1]:

$$\pi(y_{i,k}|x_i, w) = \frac{exp(w_k^T x_i)}{\sum_{j=1}^{K} exp(w_j^T x_i)} \tag{7}$$

In this case, we have a matrix of target labels y that is n by K, and $y(i,j)$ is set to $+1$ if instance i has class j, and 0 otherwise. The weights are now expanded to a p by K matrix, and we now have

an individual weight vector corresponding to each class. Note that writing the class probabilities in this way makes it clear that we are employing an exponential family distribution. By observing that the normalizing denominator enforces that the probilities summed over the classes must be equal to 1, we can set the parameter vector for one of the classes to be a vector of zeros. Using this, we can see that in this case the softmax likelihood will be identical to the binary logistic regression case when we have two classes. Note also that the coefficients used in a softmax function retain their interpretability in terms of changes to the log-odds, but that these changes are now relative to the class whose parameters are set to zero [1].

Again assuming an independent Gaussian prior on the elements of w, we can write the multi-class penalized log-likelihood for use in MAP estimation as follows:

$$f = -\sum_{i=1}^{n}[y_i w^T x_i - log(\sum_{j=1}^{K} exp(w_k^T x_i))] - \frac{1}{2v^2}\sum_{j=1}^{K} w_j^T w_j \tag{8}$$

Above we have introduced y_i as an indicator to select the appropriate column of w for the instace i. We see that the log-likelihood term has the familiar (numerator - denominator) form, subsequently we expect the gradient and Hessian to contain moments of the distribution. If we use SM(i,k) to denote the softmax probability of instance i for class k, and $\delta_{i==j}$ as the kronecker delta function for i and j, we express the gradient for the parameters of class k and the Hessian for the parameters of classes k and j as follows:

$$g_k = -\sum_{i=1}^{n}[x_i(y_i - SM(i,k))] - \frac{1}{v^2}w_k \tag{9}$$

$$H_{kj} = -\sum_{i=1}^{n} x_i x_i^T [SM(i,k)(\delta_{i=j} - SM(i,j))] - \frac{\delta_{j=k}}{v^2} \tag{10}$$

The Hessian remiains negative definite, but now has $(pK)^2$ elements instead of (pK), making computing and/or inverting the Hessian much more expensive. It is noteworthy that in the softmax case we can (and will) have a higher degree of correlation between variables than we did for the binary case since we

have additional correlation between the classes. We have implemented MAP estimation for multinomial regression making use of Matlab's 'fminunc' function (and hence using updates based on the gradient and an inverse Hessian approximation as discussed for the binary case) in the directory 'MLOGREG', and an example calling this code is included as 'example_MLOGREG.m'. Note that this code is not vectorized, so could be made much more efficient.

2 Bayesian Auxiliary Variable Methods

Above we have described in detail the logistic and multinomial regression models, and overviewed some straightforward methods to perform MAP parameter estimation in such models under a Gaussian prior. However, we would much rather be doing Bayesian parameter estimation in these models, in order to obtain posterior distributions of the model parameters. We now turn to Bayesian methods of estimating posterior distributions in logistic regression models. In particular, we will focus on the Gibbs sampling method employing auxiliary variables and joint updates to the regression coefficients and auxiliary variables proposed in [5].

2.1 Binary Probit Regression

As discussed in class, we can derive conjugate priors for the logistic regression likelihood function, but they are not terribly intuitive. Fortunately, we can transform the model into an equivalent formulation that includes auxiliary variables, and admits standard conjugate priors to the likelihood function. This method is an extension of the well-known auxiliary variable method for Binary Probit Regression of [6]. Before discussing the logistic regression, we will first review the simpler Bayesian Binary Probit Regression model, as presented in [5].

Using Φ to denote the Gaussian cumulative distribution function, Binary Probit Regression uses the following likelihood:

$$\pi(y_i = 1|x_i) = \Phi(x_i^T w) \tag{11}$$

Since there is no conjugate prior to the Gaussian cumulative distribution function, we introduce a set of n auxiliary variables z_i with:

$$z_i = x_i^T w + \epsilon_i \tag{12}$$

Here, $\epsilon_i \sim \mathbf{N}(0, 1)$, and y_i takes the value of 1 iff z_i is positive. Introducing the auxiliary variables gives an equivalent model, but this model is more amenable to sampling since w is removed from the likelihood. In the particular case of a Gaussian prior on w, it admits a straightforward Gibbs sampling strategy where z_i is sampled from independent (univariate) truncated Gaussian distributions, and w can can be sampled from a multvariate Gaussian distribution. Specifically, a straightforward Gibbs sampling strategy with $\pi(w) \sim \mathbf{N}(b, v)$ can be implemented using the following [5]:

$$z_i|w \propto \mathbf{N}(x_i^T w, 1) I(z_i > 0) y_i = 1 \tag{13}$$

$$z_i|w \propto \mathbf{N}(x_i^T w, 1) I(z_i \leq 0) y_i \neq 0 \tag{14}$$

$$w|z, y \sim \mathbf{N}(B, V) \tag{15}$$

$$B = V(v^{-1} b + X^T z) \tag{16}$$

$$V = (v^{-1} + X^T X)^{-1} \tag{17}$$

$$\tag{18}$$

As before, we will assume that the mean of the prior on the regression coefficients b is zero. Following from this, we note above that B is the normal equations in Least Squares estimation, and that V is the corresponding inverse Hessian (corresponding to the inverse convariance or precision matrix). The Gibbs sampler produced by the above strategy is trivial to implement (given currently available linear algebra software), since it only requires sampling from a multivariate Gaussian, and truncated univariate Gaussians. Unfortunately, as discussed in class, this straightforward Gibbs sampling approach is innefficient since the element of w are highly correlated with the elements of z.

To combat the correlation inherent in w and z in the above model, [5] proposed a method to update w and z jointly by using the product rule to decompose the joint probability of the model as follows:

$$\pi(w, z|y) = \pi(z|y)\pi(w|z) \tag{19}$$

The proposed method samples each z_i from a Gaussian distribution with means and variances derived from a leave-one-out marginal predictive density (see (5) in [5]), updating the conditional means after each update to z_i, then sampling w from its conditional normal distribution after all of the z_i have been sampled. Although results are presented showing that this joint updating strategy offers an advantage, we will not review it in detail since there exists a simpler method to facilitate joint updating in logistic regression. Nevertheless, we have implemented this block updating scheme (based on the pseudocode present in the paper) in the directory 'PROBREGSAMP'. For our implementations, we used a simple rejection sampling approach for sampling from truncated distributions, where each rejected sample restricts the sampling density envelope. An example showing how to run this code is at 'example_PROBREGSAMP'.

2.2 Binary Logistic Regression

Beginning from the binary Bayesian Probit Regression model above, [5] propose to perform binary Bayesian Logistic Regression by replacing the independent Gaussian prior on ϵ with independent logistic distributions. Unfortunately, this significantly complicates the simple sampling strategies above. To facilitate straightforward sampling of this model [5] introduce an addtional set of auxiliary variables $\lambda_{1:n}$ and modify the noise function to be a scale mixture of normals with a marginal logistic distribution as follows (where KS denotes the Kolmogorov-Smirnov distribution):

$$\epsilon_i \sim \mathbf{N}(0, \lambda_i) \tag{20}$$

$$\lambda_i = (2\psi_i)^2 \tag{21}$$

$$\psi_i \sim KS \tag{22}$$

If we temporarily view λ as constant, we see that this is identical to the Probit model above, except that each value of z_i has an individual term λ_i for its noise variance. Subsequently, we know how to sample from this model for fixed λ. It simply involves using Weighted Least Squares instead of Least Squares (and associated inverse Hessian), and sampling from individual truncated normals that have different variances. Subsequently we can implement a Gibbs sampler if we are able to sample from the KS distribution. Fortunately, [5] outline an rejection sampling method to simulate from the KS distribution using the Generalized Inverse Gaussian as the sampling density. Thus, we can implement a straightforward Gibbs sampler for binary Bayesian Logistic Regression using this rejection sampling approach in addition to the following:

$$z_i|w, \lambda \propto \mathbf{N}(x_i^T w, \lambda_i) I(z_i > 0) if y_i = 1 \tag{23}$$

$$z_i|w, \lambda \propto \mathbf{N}(x_i^T w, \lambda_i) I(z_i \leq 0) if y_i \neq 0 \tag{24}$$

$$w|z, y, \lambda \sim \mathbf{N}(B, V) \tag{25}$$

$$B = V(v^{-1} b + X^T W z) \tag{26}$$

$$V = (v^{-1} + X^T W X)^{-1} \tag{27}$$

$$W = diag(\lambda^{-1}) \tag{28}$$

$$\tag{29}$$

Two approaches are presented in [5] to perform block sampling of the parameters. The first uses the same strategy as in the Probit model, where w and z are updated jointly in the same way using the above modifications to the conditionals, followed by an update to the addtional auxiliary variables λ. We implemented this strategy (based on the pseudocode from the paper) in the directory 'LOGREGSAMP', 'example_LOGRESAMP' is an example script calling this function.

The second (and the author's preferred) strategy for block sampling presented in [5] updates z and λ jointly, followed by an update to w. Sampling w and λ remains identical in this approach, but sampling z_i becomes easier. In this approach, $z_i|w, \lambda$ follows a truncated logistic distribution with mean $x_i^T w$ and a scale of 1. Not only does this obviate the need for computing marginal predictive densities, the inverse of the cumulative distribution function of the logistic distribution has a closed form and is subsequently trivial to sample from (although we again used a simple adaptive rejection sampling technique in our implementation). We implemented this strategy (based on the pseudocode from the paper) in the directory 'LOGREGSAMP', 'example_LOGRESAMP2' is an example script calling this function.

2.3 Multinomial Logistic Regression

Unlike the Probit Regression case, the binary Logistic Regression sampling techniques above have a trivial extension to the multi-class scenario. In addition to having a y and w variable for each class as we saw in Section 1, we now have a z and λ vector for each class. The Gibbs sampler presented in [5] simply loops over the classes, performing the binary logistic regression sampling technique for the current classes keeping all other classes fixed. We implemented this strategy (based on the pseudocode from the paper) in the directory 'MLOGREGSAMP', 'example_MLOGRESAMP' is an example script calling this function. Unfortunately, we found that this does not make an especially effective sampling strategy, and that the technique stays in areas of the distribution that were far from the MAP estimate, and did not produce accurate classification results. We hypothesize that this is due to several factors. The first factor is simply the larger number of parameters in this model. Another factor is that, as discussed previously, there can inherently be a much higher degree of correlation in the softmax case than in the binary scenario. Finally, we note that the sampling strategy of looping over the classes, and running the binary sampler is not especially clever about dealing with these correlations, since it requires seperate sampling of the z values for each class in addition to the w values for each class, and a joint update would likely improve the performance.

Before moving on to feature selection, I would like to outline some extensions of the above models

that I would have liked to explored, if I had more time. One idea with significant potential for improving the sampling strategies is to integrate out parameters. Given the high degree of correlation between variables (especially in the multi-class case), this would likely improve the sampling strategies significantly. Another area of exploration is to not the view covariance or hyper-parameters as fixed, and explore posterior estimates with priors on these distributions. This is especially relevant from the point of view of model generalization, since the covariance and hyper-parameters can significantly affect the classification performance of the model.

3 Feature Selection

A major appeal of Logistic Regression, besides its intuitive multi-class generalization, is the interpretation of its coefficients. As discussed in [1], researchers often explore different combinations of the features in order to produce a parsimonious regression model that still provides effective prediction performance. In this section, we discuss automated approaches to this feature selection problem. We first present an extension to the above models that incorporates feature selection through trans-dimensional sampling. We then turn our focus to priors that encourage sparsity in the final model.

3.1 Trans-Dimensional Sampling

Focusing on the binary logistic regression scenario, one method to incorporate feature selection into the procedure is to add yet another set of auxiliary variables, $\gamma_{1:p}$. Specifically, if the binary variable γ_i is set to 1 then the corresponding variable is included in the model, and if γ_i is set to 0 then the corresponding variable is excluded from the model (ie. set to 0). [5] proposes this model, and suggest using the model presented earlier for binary logistic regression with these auxiliary variables, with joint updates to $\{z, \lambda\}$ and to $\{\gamma, w\}$. They propose that $\gamma|z$ can be sampled using (Reversible-Jump) Metropolis-Hastings steps. Specifically, sampling z, λ, and w remains the same (but using only the active covariate set), and we accept a trans-dimensional step from γ to $\gamma_\star$ (under a symmetric proposal) using the following acceptance probability:

$$\alpha = min\{1, \frac{|V_{\gamma_\star}|^{1/2}|v_\gamma|^{1/2}exp(0.5B_{\gamma_\star}^T V_{\gamma_\star}^{-1}B_{\gamma_\star})}{|V_\gamma|^{1/2}|v_{\gamma_\star}|^{1/2}exp(0.5B_\gamma^T V_\gamma^{-1}B_\gamma)}\} \tag{30}$$

[5] uses a simple proposal distribution, they flip the value of a randomly chosen element of γ. We implemented sampling from the above model in the case of binary logistic regression (with feature selection) in the directory 'LOGREGSAMP_FS', an example running this routine is 'example_LOGREGSAMP_FS'.

3.2 Priors Encouraging Sparsity

Although the above strategy to incorporate feature selection into the model is a simple extension of the logistic regression model, it has major drawbacks. Specifically, updating single components causes very slow exploration of the space of 2^p variables. As discussed in class, we could jointly update correlated components to significantly improve the results. An alternate strategy, especially relevant when p is very large, is to use priors that encourage sparsity.

3.3 MAP Estimation of the Logistic LASSO

The LASSO prior advocated in [7] (but utlilized earlier under the name 'Basis Function Pursuit' [8]) is currently a popular strategy for enforcing sparsity in the weights of the regression coefficients. From the point of view of optimization, the LASSO prior consists of using a scaled value of the L1-norm of the weights as the penalty/regularization function, instead of the squared L2-norm discussed earlier. Specifically, our objective function becomes:

$$f = -\sum_{i=1}^{n} log(1 + exp(-y_i w^T x_i)) - \frac{1}{v}||w||_1 \tag{31}$$

Although the above objective is still concave, a major disadvantage of this objective function is that it is non-differentiable at points where any w_i is zero. Hence, we need to use slightly less generic optimization approaches for finding the MAP estimates. Furthermore, we cannot use efficient methods such as the one presented in [9] for Least Squares estimation under an L1 penalty in order to optimize the logistic regression likelihood function. The most widely used method for optimizing the logistic

regression likelihood subject to an L1 penalty are Iteratively Reweighted Least Squares (IRLS) schemes (employ a Least Squares LASSO solver as subroutine) [7], and coordinate descent methods such as [10]. However, both of these schemes are relatively innefficient, since coordinate descent methods have slow convergence and the IRLS methods require the solution to a large number of individual LASSO problems, themselves not-completely trivial optimization problem. In this project, we adopted the efficient 'Grafting' approach presented in [11], this approach is briefly outlined below.

We define the derivative of the objective function as follows:

$$g = -\sum_{i=1}^{n}(1 - \sigma(y_i w^T x_i))y_i x_i - \frac{sign(w)}{v} \tag{32}$$

Since the sign of 0 is ill-defined, the Grafting approach uses the following convention (recall that the global optimum will have a gradient value of 0, where it is defined):

$$sign(w_i) = 1 if \sum_{i=1}^{n}(1 - \sigma(y_i w^T x_i))y_i x_i > 1/v \tag{33}$$

$$sign(w_i) = -1 if \sum_{i=1}^{n}(1 - \sigma(y_i w^T x_i))y_i x_i < 1/v \tag{34}$$

$$sign(w_i) = 0 if \sum_{i=1}^{n}(1 - \sigma(y_i w^T x_i))y_i x_i = 1/v \tag{35}$$

Although this is not the gradient of the function, it is correct if w_i is not 0, and chooses the correct sign for values of w_i that are equal to 0. However, this definition may cause problems when multiple variables have a value of 0. Subsequently, the Grafting procedure begins with all variables set to 0, and introduces one zero-valued variable at a time into the model (the one with largest gradient magnitude). The 'free set' of variables is optimized using a standard conjugate gradient or Quasi-Newton solver, and variables are removed that return to 0 (they may later be re-introduced). This continues until $|\sum_{i=1}^{n}(1 - \sigma(y_i w^T x_i))y_i x_i| \leq 1/v$. At this point, no variables can be introduced that will decrease the log-likelihood enough to compensate for the increase in the penalty term. If the final model is sparse, this method will be much more efficient than IRLS or coordinate descent methods.

We implemented the optimization of the logistic regression likelihood subject to an L1 penalty with the Grafting procedure in the folder 'LOGREGL1', and an example calling this code is 'example_LOGREGL1'. The derivation of the multinomial case is almost identical (substiting in the appropriate objective and derivatives), and we implemented this in the folder 'MLOGREGL1'. An example calling this code is 'example_MLOGREGL1'. In both cases, we make use of Matlab's 'fminunc' unconstrained optimization function in order to solve the sub-problems. Note that although the binary code is efficient, the multinomial code is fairly innefficient since it is not vectorized and we do not make of indexing to reduce the model size passed to the optimization routine.

3.4 Bayesian Estimation of the Logistic LASSO

[7] showed that the LASSO penalty is equivalent to the use of independent double exponential (Laplace) priors:

$$\pi(w_i) = \frac{1}{2v} exp(-|w_i|/v) \tag{36}$$

As discussed in [12], this prior can be represented as a scale mixture of Gaussian distributions:

$$\pi_(w_i) = \int_0^\infty \mathbf{N}(w_i|0,\psi)\frac{1}{2v^2} exp(-\psi/2v^2) d\psi \tag{37}$$

[13] present a Gibbs sampling algorithms for sampling from this model (integrating out ψ) in the Least Squares scenario. Given more time, I would have explored combining this prior with the auxiliary variable approach to logistic regression discussed above, since it is a relatively straightforward extension. For further discussions of sparsity encouraging priors, we refer to [13] which discusses 'spike and slab' priors, and [12] which extensively discusses other (sparse) priors that take the form of scale mixture of Gaussian.

4 Experiment

Included with the code are 3 of the data sets used in [5], including functions to load these data sets in the appropriate format ('pima.m', 'aus.m', and 'heart.m'). Since all the data sets in [5] were binary, the classic Iris data set is included as a multi-class data set (it can be loaded with 'iris.m'). However, in this section we provide experimental results on publicly available MNIST digit recognition data. This data set makes a slightly more interesting scenario than the cases above since the number of features is subtantially larger (256, compared to 4-15), and both the features and coefficients have a natural visual interpretation. Since the multinomial optimization code is relatively slow and the multinomial sampling strategy is relatively inneffective, we concentrated on the binary task of recognizing digits of the number 2 compared to digits of the number 3. The data set subsequently consisted of 189 examples of the 256 features representing the pixel intensities of 16 by 16 images of either the digit 2 or the digit 3. The code to re-run the experiments is located in the 'experiments' directory.

4.1 MAP Estimation

We first compared the generalization performance of MAP estimation with L2 and L1 penalties. The data set was divided randonly into 100 training examples and 89 testing examples. The MAP estimates with $(1/v) = 1$ for both methods, and with $(1/v) = 0.00001$ for the L1 penalty and $(1/v^2) = 0.00001$ for the L2 penalty were computed, and their performance on the test set was measured. The L2 penalized parameters formed a dense set of coefficients that completely separated the training data (this is unsurprising, given that the number of features is larger than the number of examples). The L1 penalized parameters also completely separated the training data, but only 8 of the coefficients had non-zero weights with $(1/v) = 1$, and only 24 of the coefficients had non-zero weights with $(1/v) = 0.00001$. On the test set, both L2 penalized sets of parameters resulted in 4 errors. The 8 coefficients resulting from the first set of L1 penalized parameters resulted in 8 test errors, while the second set of L1 penalized parameter's 24 coefficients resulted in 4 test errors. Subsequently, only 24 out of the 256 pixels were actually required to achieve thes test performance of a full dense model in this scenario.

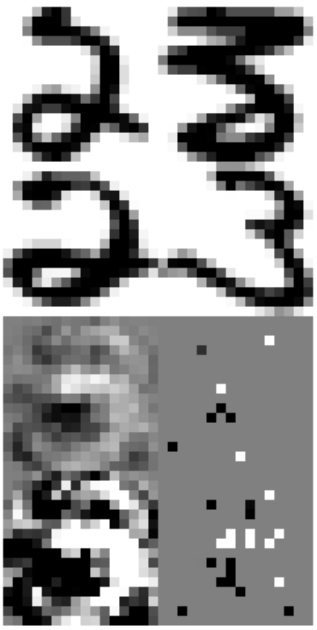

Figure 1. Top two rows: Examples of the number 2 and the number 3. Third row: L2 penalized parameters (left) and L1 penalized parameters (right) with $(1/v) = 1$. Bottom row: L2 penalized parameters (left) with $(1/v^2) = 0.00001$ and L1 penalized parameters (right) with $(1/v) = 0.00001$. In the log-odds maps, bright indicates that it increases the log-odds of the class being assigned the label '3' when the corresponding pixel is dark, dark indicates that it decreases the log-odds of the class being assigned the label '3' when the corresponding pixel is dark.

As a visual illustration of this experiment, Figure 1 shows two examples of the input data. Figure 1 also shows a representation of the coefficients estimated by the four models represented as 16 by 16 images. From these images, we see that the L2 penalized coefficients have an obvious visual represntation, as we can see an area that resembles a "3" that increases the log-odds of the class label being 3. The visualization of the L1 penalized coefficients does not give a terribly intuitive result, as it simply appears as a set of selected pixels that significantly increase or decrease the log-odds. However, it is interesting that the 8/24 selected pixels are not clustered (although not random either), and sample different areas of the image.

4.2 Bayesian Estimation

We now examine sampling the posterior distributions of the parameters under the logistic regression model. We used the 2 different blocked Gibbs sampling strategies presented earlier to generate 10000

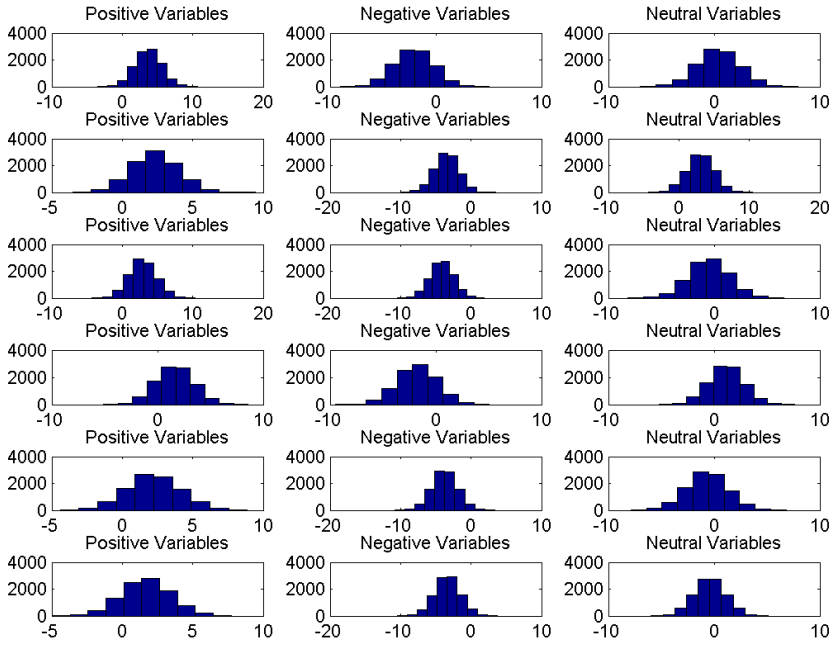

Figure 2. Estimated posterior distributions for the 18 of the parameters based on 10000 samples using the block Gibbs sampling strategy with joint updates to $\{w, z\}$.

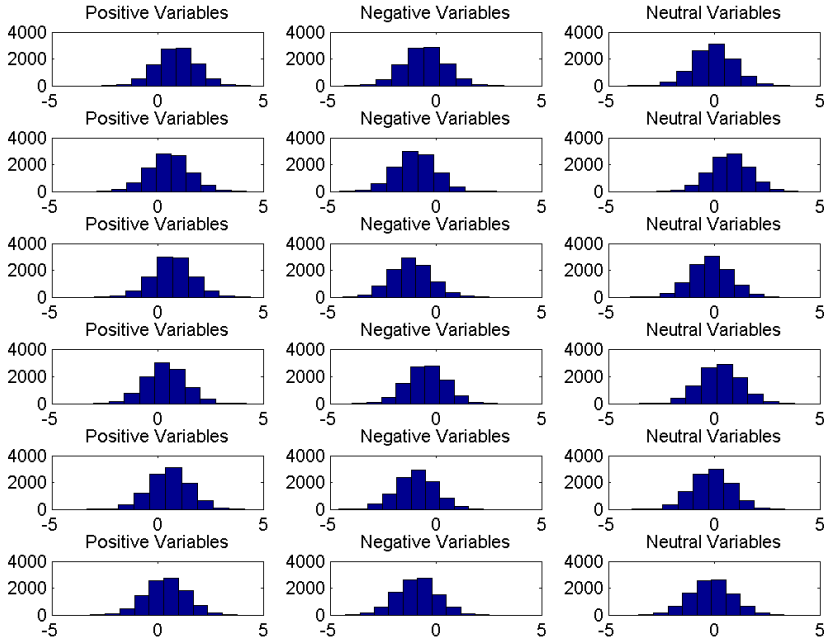

Figure 3. Estimated posterior distributions for the 18 of the parameters based on 10000 samples using the block Gibbs sampling strategy with joint updates to $\{\lambda, z\}$.

samples from the posterior distribution of the parameters using the full data set. We also compute the L1 penalized MAP parameters for the full data set with $1/v = 1$. This resulted in a model with 13 non-zero coefficients (although one of the coefficients was very small and we subsequently treat it as 0). Figures 2 and 3 plot histograms of the samples generated for 18 of the 256 variables. From left to right, we plotted the posterior estimates for the 6 positive coefficients estimated in the L1 MAP model, the 6 negative coefficients estimated in the L1 MAP model, and 6 randomly chosen coefficients among those set to 0 in the L1 MAP model. The results in Figure 2 were generated by jointly updated $\{w, z\}$ and then sampling λ, while the results in Figure 3 were generated by jointly updated $\{z, \lambda\}$ and then sampling w. As can be seen, the results of the different sampling strategies are very similar.

Examining the estimated posterior distributions of the coefficients that were selected by the L1 penalized methods, we see that they have distributions that are clearly centered away from zero, and in all cases in the appropriate direction. Examining the variables that were not selected by the L1 penalized method, we see that some have distributions centered at zero, indicating that they likely should be removed from the model (if we assume that the features are independent). However, some of the features removed by the L1 penalized method have distributions that are centered away from zero. In these cases, it seems that the L1 penalized method removed potentially relevant variables from the model.

4.3 Feature Selection

We repeated the above experiment with the trans-dimensional sampler to assess the relevance of the different features. We again generated 10000 samples, and Figure 4 plots the estimated distributions for the same 18 variables (a weight is recorded as being 0 when it is not included in the model). The acceptance ratio for this process was 0.79, an very high value. This seems to indicate that most individual variables are not especially relevant. Figure 4 confirms this intuition, as most variables end up spending time both in and out of the model. It is interesting to note that some of the coefficients selected by the L1 penalized model are clearly relevant, since these coefficients tended not to be removed from the model. However, we must recall that we are taking 1 step moves through a space of 2^{256}, and as such we are seeing an estimate of an infitesimally small fraction of the full posterior distribution. As such there may

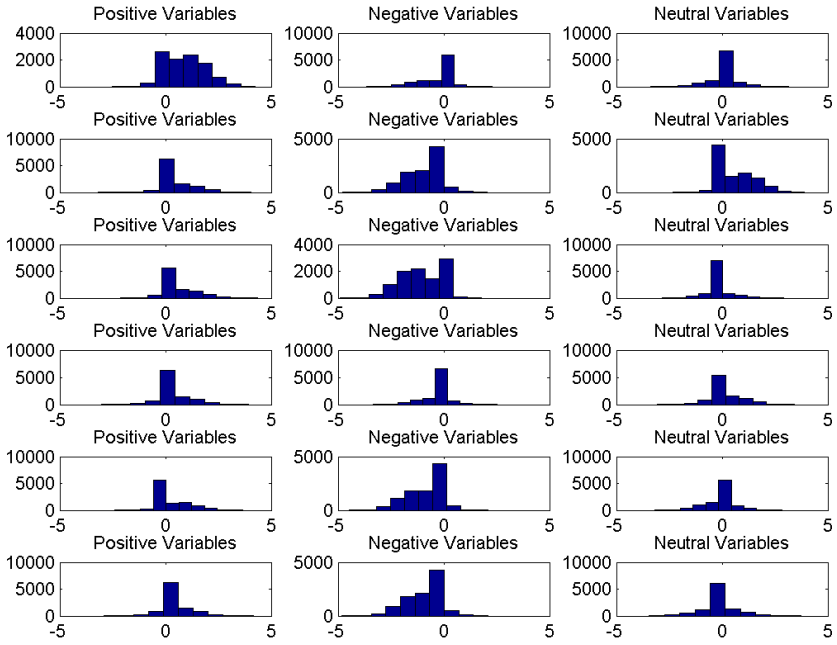

Figure 4. Estimated posterior distributions for the 18 of the parameters based on 10000 samples using the trans-dimensional block Gibbs sampling strategy.

be areas of the parameter space where even the highly relevant variables are removed. In addition, recall that since an individual variable is selected randomly to add or remove from the model at each iteration, that variables will tend to stay in or out of the model for approximately 256 iterations at a time if the trans-dimensional always accepted.

References

[1] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer, August 2001.

[2] S.J. Wright J. Nocedal. *Numerical Optimization*. Springer, New York, NY, USA, 1999.

[3] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[4] T.P. Minka. Algorithms for maximum-likelihood logistic regression. Technical report, Carnegie Mellon University, 2001.

[5] K. Held C.C. Holmes. Bayesian auxiliary variable models for binary and polychotomous regression. *Bayesian Analysis*, 1(1):145–168, 2006.

[6] J. H. Albert and S. Chib. Bayesian analysis of binary and polychotomous response data. 88(422), June 1993.

[7] R. Tibshirani. Regression shrinkage and selection via the lasso. Technical report, University of Toronto, 1994.

[8] Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1999.

[9] M. R. Osborne, Brett Presnell, and B. A. Turlach. A new approach to variable selection in least squares problems. *IMA Journal of Numerical Analysis*, 20(3):389–403, 2000.

[10] A. Genkin, D. Lewis, and D. Madigan. Large-scale bayesian logistic regression for text categorization. Submitted.

[11] Simon Perkins, Kevin Lacker, and James Theiler. Grafting: Fast, incremental feature selection by gradient descent in function space. *Journal of Machine Learning Research*, 3:1333–1356, 2003.

[12] J.E. Griffin and P.J. Brown. Alternative prior distributions for variable selection with very many more variables than observations. Technical report, Dept. of Statistic, University of Warwick, 2005.

[13] G. Casella T. Park. The bayesian lasso, 2005.