# Fast Bayesian L1 Regularization for Sparse Logistic Regression

**Classification, Sparse logistic regression, Laplace prior, Bayesian learning**

## Abstract

In the paper we propose a sparse logistic regression method, in which the sparsity arises from the use of a Laplace prior, but where the usual regularization parameters are integrated out analytically. Such approach leads to the discontinuous criterion function. Two variants of training procedure which deals with discontinuities are presented. In the first approach we use coordinate descent method while in the second one we fuzzify the criterion by making it smooth and then use standard second-order optimization. Both methods lead to sparse decision rules which are comparable with the state-of-art SVM and RVM classifiers.

## 1. Introduction

In this paper, we present two methods for training Bayesian sparse logistic regression with Laplace prior been integrated out originally proposed in (Authors, 1900a) as a substantial improvement to the sparse logistic regression (SLogReg) approach of (Shevade & Keerthi, 2003). The SLogReg algorithm employs an $L1$-norm regularisation term (Tikhonov & Arsenin, 1977), corresponding to a Laplace prior over the model parameters (Williams, 1995), in order to identify a sparse sub-set of the most discriminatory features. Both the generalization ability of the classifier and the level of sparsity achieved are critically dependent on the value of a regularization parameter, which must be carefully tuned to optimize performance. This is normally achieved by a computationally intensive search for the minimizer of a cross-validation based estimate of generalization performance. Instead, a Bayesian approach could be adopted, in which the regularization parameter is integrated out analytically, using an uninformative Jeffery's prior, in the style of (Buntine & Weigend, 1991) (see also (Lehrach et al., 2006)). The resulting parameterless classification algorithm (BLogReg) is much easier to use and is comparable in performance with the original sparse logistic regression algorithms, but is two or three orders of magnitude faster, as there is no longer a need for a model selection stage to optimize the regularization parameter. We propose two possible ways of training the classifier. In the first case we minimize exact Bayesian criterion using coordinate descent method. In the second case we approximate criterion function with its continuous analogue and then use second-order optimization. It tends to be faster while showing approximately the same accuracy. However, in the problems with large amount of features it becomes dependant on the choice of starting point and the exact procedure is preferable.

The existing sparse logistic regression optimization problem is reviewed in Section 2. The modified Bayesian logistic regression is then introduced in Section 3. Section 4 presents two training procedures for Bayesian criterion function resulting in BLogReg and FuzzyBLogReg algorithms. Experimental results obtained on the well-studied problems from the UCI repository are presented in Section 5, demonstrating the competitiveness of the algorithm. Finally, the work is summarized and conclusions drawn in Section 6.

## 2. Sparse Logistic Regression

We are commonly faced with statistical pattern recognition problems, where we must learn some decision rule distinguishing between objects belonging to one of two classes, based on a set of $n$ training examples, $\mathcal{D} = (\mathcal{T}, \mathcal{X}) = \{(t_i, \vec{x}_i)\}_{i=1}^{n}, \vec{x}_i \in \mathbb{R}^d, t_i \in \{-1, +1\}$. Logistic regression is a classical approach to this problem, that attempts to estimate the *a-posteriori* probability of class membership based on a linear combination of the input features,

$$p(1|\vec{x}) = \frac{1}{1 + \exp\{-y(\vec{x})\}}$$

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

where

$$y(\vec{x}_i) = \sum_{j=1}^{d} w_j x_{ij} + w_0.$$

The parameters of the logistic regression model, $\vec{w} = (w_0, w_1, \ldots, w_d)$, can be found by maximizing the *likelihood* of the training examples, or equivalently by minimizing the negative log-likelihood. Assuming $\mathcal{D}$ represents an independent and identically distributed (i.i.d.) sample from a Bernoulli distribution, the negative log-likelihood is given by

$$L(\mathcal{T}|\mathcal{X}, \vec{w}) = \sum_{i=1}^{n} \log\left\{1 + \exp\left(-t_i y(\vec{x}_i)\right)\right\}.$$

The resulting model is however fully dense, in the sense that none of the model parameters $\vec{w}$ are in general exactly zero. Ideally we would prefer a model based on a small selection of the most informative features, with the remaining features being "pruned" from the model. A sparse model can be introduced by adding a regularization term to the negative log-likelihood (Williams, 1995), corresponding to a Laplace prior over $\vec{w}$, to give a modified training criterion,

$$F = L(\mathcal{T}|\mathcal{X}, \vec{w}) + \lambda R(\vec{w}) \quad \text{where} \quad R(\vec{w}) = \sum_{i=1}^{d} |w_i| \tag{1}$$

and $\lambda$ is a regularization parameter, controlling the bias-variance trade-off and simultaneously the sparsity of the resulting model. Note that the usual bias parameter $w_0$ is normally left unregularized. At a minima of $F$, the partial derivatives of $F$ with respect to the model parameters will be uniformly zero, giving

$$\left|\frac{\partial L(\mathcal{T}|\mathcal{X}, \vec{w})}{\partial w_i}\right| = \lambda \quad \text{if } |w_i| > 0$$
$$\left|\frac{\partial L(\mathcal{T}|\mathcal{X}, \vec{w})}{\partial w_i}\right| < \lambda \quad \text{if } |w_i| = 0.$$

This implies that if the sensitivity of the negative log-likelihood with respect to a model parameter, $w_i$, falls below $\lambda$, then the value of that parameter will be set exactly to zero and the corresponding input feature can be pruned from the model. The principal shortcomings of this approach lie in the training algorithm no longer involving an optimization problem with continuous derivatives and in the need for lengthy cross-validation trials to determine a good value for the regularization parameter $\lambda$. Below we suggest a possible solution for these two problems based on integrating out regularization parameter $\lambda$ and approximation of criterion with a smooth function.

## 3. Bayesian Regularization

In this section, we demonstrate how the regularization parameter may be eliminated, following the methods of (Buntine & Weigend, 1991) and (Williams, 1995) and then briefly describe the relationship of this approach with alternative Bayesian methods based on evidence framework.

### 3.1. Eliminating the Regularization Parameter $\lambda$

Minimization of (1) has a straight-forward Bayesian interpretation; the posterior distribution for the model parameters $\vec{w}$ can be written as

$$p(\vec{w}|\mathcal{D}, \lambda) \propto p(\mathcal{D}|\vec{w}) p(\vec{w}|\lambda).$$

$F$ is then, up to an additive constant, the negative logarithm of the posterior density. The prior over model parameters, $\vec{w}$, is then given by a separable Laplace distribution

$$p(\vec{w}|\lambda) = \left(\frac{\lambda}{2}\right)^N \exp\{-\lambda R(\vec{w})\} = \prod_{i=1}^{N} \frac{\lambda}{2} \exp\left\{-\lambda |w_i|\right\}, \tag{2}$$

where $N$ is the number of active (non-zero) model parameters. A good value for the regularization parameter $\lambda$ can be estimated, within a Bayesian framework, by maximizing the *evidence* (MacKay, 1992a; MacKay, 1992c; MacKay, 1992b) or alternatively it may be integrated out analytically (Buntine & Weigend, 1991; Williams, 1995). Here we take the latter approach, where the prior distribution over model parameters is given by marginalizing over $\lambda$,

$$p(\vec{w}) = \int p(\vec{w}|\lambda) p(\lambda) d\lambda.$$

As $\lambda$ is a scale parameter, an appropriate ignorance prior is given by the improper Jeffrey's prior, $p(\lambda) \propto 1/\lambda$, corresponding to a uniform prior over $\log \lambda$. Substituting equation (2) and noting that $\lambda$ is strictly positive,

$$p(\vec{w}) = \frac{1}{2^N} \int_0^\infty \lambda^{N-1} \exp\{-\lambda R(\vec{w})\} d\lambda.$$

Using the Gamma integral, $\int_0^\infty x^{\nu-1} e^{-\mu x} dx = \frac{\Gamma(\nu)}{\mu^\nu}$, we obtain

$$p(\vec{w}) = \frac{1}{2^N} \frac{\Gamma(N)}{R(\vec{w})^N} \quad \Longrightarrow \quad -\log p(\vec{w}) \propto N \log R(\vec{w}),$$

giving a revised optimization criterion for sparse logistic regression with Bayesian regularization[1],

$$Q = L(\mathcal{T}|\mathcal{X}, \vec{w}) + N \log R(\vec{w}), \tag{3}$$

[1] $N \log R(\vec{w})$ is assumed to be zero when $N = 0$.

in which the regularization parameter has been eliminated, for further details and theoretical justification, see (Williams, 1995). Similar approach for integrating out scale parameters of Gaussian distribution based on the use of Jeffrey's prior and exponential prior has been proposed by (Figueiredo, 2003), along with an Expectation-Maximization (EM) style training algorithm.

### 3.2. Relationship with the Evidence Framework

It has been observed that the "integrate-out" approach to dealing with the regularization parameter (Buntine & Weigend, 1991) is likely to lead to over-regularized models that under-fit the data, for neural network models with a traditional Gaussian *weight-decay* prior, and that *evidence* framework (MacKay, 1992a; MacKay, 1992c; MacKay, 1992b) is generally to be preferred (MacKay, 1994). However, it is relatively straight-forward to show that, in the case of the Laplace prior, the iterative update formula for the effective regularization parameter (4) is identical to the update formula for the regularization parameter under the evidence framework (Williams, 1995).

## 4. Minimizing the Bayesian Training Criterion

### 4.1. An Exact Optimization Procedure

For deriving optimization procedure for Bayesian training criterion (3) we first consider the case with predefined regularization coefficient (1). An efficient training algorithm proposed by (Shevade & Keerthi, 2003) seeks to minimize the cost function (1) by optimizing one parameter at a time via Newton's method. However, due to the discontinuity in the first derivative at the origin, care must be taken when the value of a model parameter passes through zero. This is achieved by bracketing the optimal value for a model parameter, $w_i$, by upper and lower limits ($H$ and $L$ respectively) such that the interval does not include 0, except perhaps at a boundary. These limits can be computed using the gradient of $F$ with respect to $w_i$ computed at its current value and at zero, from both above and below, as shown in Table 1.

A model parameter must be selected for optimization at each iteration, the parameter with the gradient of the greatest magnitude is a sensible choice. In order to improve the speed of convergence, we begin by optimizing only active parameters (those with non-zero values), and only consider inactive parameters if no active parameter can be found with a non-zero gradient.

*Table 1.* Special cases that must be considered in optimizing $F$ with respect to $w_i$ in order to avoid difficulties due to the discontinuity in the first derivative at the origin.

| **Case** | $w_i$ | $\frac{\partial F}{\partial w_i}\Big\vert_{w_i}$ | $\frac{\partial F}{\partial w_i}\Big\vert_{0^-}$ | $\frac{\partial F}{\partial w_i}\Big\vert_{0^+}$ | $L$ | $H$ |
|---|---|---|---|---|---|---|
| 1 | 0 | — | $< 0$ | $< 0$ | 0 | $+\infty$ |
| 2 | 0 | — | $> 0$ | $> 0$ | $-\infty$ | 0 |
| 3 | $< 0$ | $> 0$ | — | — | $-\infty$ | $w_i$ |
| 4 | $> 0$ | $< 0$ | — | — | $w_i$ | $+\infty$ |
| 5 | $< 0$ | $< 0$ | $> 0$ | — | $w_i$ | 0 |
| 6 | $> 0$ | $> 0$ | — | $< 0$ | 0 | $w_i$ |
| 7 | $< 0$ | $< 0$ | — | $> 0$ | 0 | $+\infty$ |
| 8 | $> 0$ | $> 0$ | $< 0$ | — | $-\infty$ | 0 |
| 9 | $< 0$ | $< 0$ | $\leq 0$ | $\geq 0$ | 0 | 0 |
| 10 | $> 0$ | $> 0$ | $\leq 0$ | $\geq 0$ | 0 | 0 |

Iterative optimization procedures do not generally reduce the gradient *exactly* to zero, and so in practice we only consider parameters $\vec{w}$ if they have a gradient exceeding a pre-defined tolerance parameter $\tau$. The algorithm terminates when no such parameter can be found. For a complete description of the training algorithm, see (Shevade & Keerthi, 2003).

In that paper it is demonstrated that the cost function for sparse logistic regression using a Laplace prior can be iteratively minimized in an efficient manner one parameter at a time. Note that the objective function is non-smooth, as the first derivatives exhibit discontinuities at $w_i = 0,\ \forall i \in \{1, 2, \ldots, N\}$, but is otherwise smooth. These properties of the objective function are clearly evident from the first and second derivatives,

$$\frac{\partial}{\partial w_i} \log R(\vec{w}) = \frac{w_i}{|w_i|} \frac{1}{R(\vec{w})}$$
$$\frac{\partial^2}{\partial w_i^2} \log R(\vec{w}) = -\frac{1}{R(\vec{w})^2}.$$

The training criterion incorporating a fully Bayesian regularization term (3) can be minimized via a simple modification of the existing training algorithm for sparse logistic regression. Differentiating the original and modified training criteria (1,3), we have that

$$\nabla F = \nabla L(\mathcal{T}|\mathcal{X}, \vec{w}) + \lambda \nabla R(\vec{w})$$
$$\nabla Q = \nabla L(\mathcal{T}|\mathcal{X}, \vec{w}) + \tilde{\lambda} \nabla R(\vec{w})$$

where

$$1/\tilde{\lambda} = \frac{1}{N} \sum_{i=1}^{N} |w_i|. \tag{4}$$

From a gradient descent perspective, minimizing $Q$ effectively becomes equivalent to minimizing $F$, assum-

ing that the regularization parameter, $\lambda$, is continuously updated according to (4) following every change in the vector of model parameters, $\vec{w}$ (Williams, 1995). This requires only a very minor modification of the code implementing the sparse logistic regression algorithm, whilst eliminating the only training parameter and hence the need for a model selection procedure in fitting the model.

### 4.2. Fast Approximate Optimization Procedure

Due to the discontinuities of Bayesian criterion (3) we cannot use advanced optimization strategies such as Newton method directly. However, it is possible to replace discontinuous function $N \log R(\vec{w})$ by its continuous approximation. Recall that $N$ is a number of non-zero weights, so each time we set a weight to zero we should reduce $N$ by one.

Now consider the following smooth estimate of $N$

$$\hat{N} = n - \sum_{i=1}^{n} \exp\left(-\frac{w_i^2}{2\sigma^2}\right),$$

where $\sigma > 0$ is some positive fuzzification coefficient. It is easy to see that $\hat{N}$ equals zero if all the weights are zeros and equals $N$ if there are $N$ relatively large weights while all the others are zeros.

It can be shown that minimum of Bayesian criterion (3) lies within the same hyperoctant $\mathcal{H}_{ML}$ where maximum likelihood point $\vec{w}_{ML}$ is located. Hence we may use constrained Newton optimization for searching minimum of the criterion within the hyperoctant $\mathcal{H}_{ML}$. Note that the function $\hat{N} \log R(\vec{w})$ is smooth function in $\mathcal{H}_{ML}$ except the point $\vec{w} = 0$. Consider now a domain

$$\mathcal{M} = \{\vec{w} \in \mathcal{H}_{ML}, |w_i| \geq \varepsilon > 0, \forall i = 1, \ldots, n\}.$$

Function $\hat{N} \log R(\vec{w})$ is smooth in $\mathcal{M}$. So we may use Newton method for optimizing approximate Bayesian criterion

$$\hat{Q} = L(\mathcal{T}|\mathcal{X}, \vec{w}) + \hat{N} \log R(\vec{w}). \tag{5}$$

If the constraint becomes active, i.e. $w_i = \varepsilon$, then such weight is set to zero. This condition establishes the relation between $\sigma$ and $\varepsilon$

$$\lim_{\varepsilon \to +0} \left(C_0 - \exp\left(-\frac{\varepsilon^2}{2\sigma^2}\right)\right) \log(C_1 + \varepsilon) = C_0 \log C_1, \tag{6}$$

$$\forall C_0 > 0, \quad C_1 > 0.$$

Figure 1 shows the behaviour with respect to single weight change of exact criterion

$$C_0 \log(C_1 + |w_i|)$$

and its approximate estimate

$$\left(C_0 - \exp\left(-\frac{w_i^2}{2\sigma^2}\right)\right) \log(C_1 + |w_i|)$$

with $C_0 = 2$, $C_1 = 0.03$, $\sigma = 0.3$. Note that approximate regularizer is continuous for all values of $w_i$ and is smooth for $w_i > 0$ (so as for $w_i < 0$).

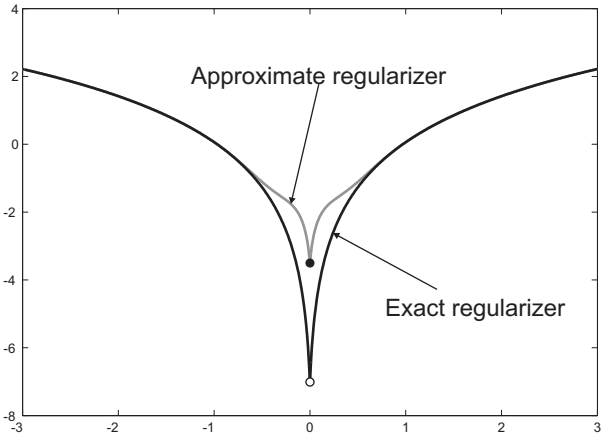


*Figure 1.* Behaviour of exact and approximate regularizers with respect to single weight change. Exact regularizer has a removable jump at point 0.

In case all weights are zeros regularizer $\hat{N} \log R(\vec{w})$ is supposed to be zero. To guarantee this for our approximation we should require the limit (6) to converge to zero when $C_0 = 1$ and $C_1 = 0$. This yields

$$\lim_{\varepsilon \to +0} \left(1 - \exp\left(-\frac{\varepsilon^2}{2\sigma^2}\right)\right) \log \varepsilon = \lim_{\varepsilon \to +0} \frac{\varepsilon^2}{2\sigma^2} \log \varepsilon = 0.$$

This is true when $\sigma = \varepsilon^k$ with $k < 1$. Setting $\varepsilon$ to relatively small value and $\sigma$ e.g. to $\sqrt{\varepsilon}$ gives valid approximation of Bayesian criterion (3) with its smooth analogue (5) throughout the whole $\mathcal{H}_{ML}$. As an optimization starting point $\vec{w}_{ML}$ can be taken.

## 5. Results

In this section, we evaluate the performance of two training procedures (BLogReg and FuzzyBLogReg) of the proposed logistic regression method, Relevance Vector Machine (Tipping, 2001) with basis functions $\phi_i(\vec{x}) = \langle \vec{x}_i, \vec{x} \rangle$, which is known to be alternative variant of applying Bayesian learning principles to the regularization of logistic regression, and the Support Vector Machine (Guyon et al., 1992) with linear kernel function and regularization coefficient $C = 1$. For all four methods error rates were measured. Be-

*Table 2.* Error rates together with standard deviations (in percents).

| Data set | BLogReg | FuzzyBLogReg | LinearSVM | LinearRVM |
|---|---|---|---|---|
| Bupa liver disorders | $33.28 \pm 2.60$ | $32.64 \pm 3.33$ | $31.59 \pm 0.58$ | $32.35 \pm 0.73$ |
| German credit numeric | $25.12 \pm 0.97$ | $24.60 \pm 0.69$ | $24.92 \pm 1.11$ | $24.76 \pm 0.22$ |
| Heart | $18.96 \pm 0.66$ | $18.81 \pm 0.99$ | $18.81 \pm 0.99$ | $20.07 \pm 1.49$ |
| Australian | $15.07 \pm 0.91$ | $15.19 \pm 0.84$ | $16.17 \pm 0.65$ | $15.57 \pm 0.22$ |
| Pima-indians-diabetes | $23.10 \pm 0.96$ | $23.10 \pm 0.96$ | $23.39 \pm 0.80$ | $23.67 \pm 0.50$ |
| Thyroid-sick.test | $4.28 \pm 0.35$ | $4.07 \pm 0.30$ | $3.97 \pm 0.57$ | $3.91 \pm 0.38$ |
| Wisconsin Diagnostic Breast Cancer | $2.99 \pm 0.48$ | $2.78 \pm 0.15$ | $3.30 \pm 0.45$ | $3.23 \pm 0.72$ |
| Wisconsin Prognostic Breast Cancer | $22.73 \pm 1.13$ | $27.98 \pm 1.81$ | $22.53 \pm 1.81$ | $22.63 \pm 0.90$ |
| **Rank** | **22.50** | **17.00** | **19.50** | **21.00** |
| Color legend | Place 1 | Place 2 | Place 3 | Place 4 |

*Table 3.* Training time together with standard deviations (in seconds).

| Data set | BLogReg | FuzzyBLogReg |
|---|---|---|
| Bupa liver disorders | $0.67 \pm 0.40$ | $0.26 \pm 0.11$ |
| German credit numeric | $1.44 \pm 0.16$ | $0.33 \pm 0.06$ |
| Heart | $0.68 \pm 0.05$ | $0.34 \pm 0.04$ |
| Australian | $2.41 \pm 0.31$ | $0.57 \pm 0.15$ |
| Pima-indians-diabetes | $0.33 \pm 0.05$ | $0.14 \pm 0.02$ |
| Thyroid-sick.test | $4.14 \pm 0.65$ | $3.13 \pm 1.55$ |
| Wisconsin Diagnostic Breast Cancer | $3.31 \pm 1.06$ | $3.29 \pm 0.79$ |
| Wisconsin Prognostic Breast Cancer | $3.78 \pm 0.81$ | $0.51 \pm 0.13$ |

sides, we compare BLogReg and FuzzyBLogReg using their training time and obtained sparsity. Benchmark datasets were taken from UCI repository (Newman et al., 1998). For each data set nominal features were transformed into a set of binary ones, unknown values were changed to mean values for each feature and then each sample was normalized in a way that each feature has zero mean and unit variance. 5x2-fold cross validation strategy was used for estimating error rates, training time and sparsity of classifiers for each data set. This strategy is known to be one of the most reliable ones for measuring classifiers characteristics (Dietterich, 1998). Tables 2, 3 and 4 report about experimental results. Rank was calculated in a usual way: for each data set classifiers get points corresponding to their place (from 1 to 4); then points are summed for all data sets.

These results allow to make the following conclusions. All four algorithms show comparable performance in terms of error rates. It means that BLogReg and FuzzyBLogReg converge to similar points $\vec{w}_{MP}$. As a result both algorithms show approximately the same sparsity. However, FuzzyBLogReg is appeared to be faster than BLogReg for all benchmark tasks. Meanwhile, it should be noted that in all benchmark tasks number of objects was significantly greater than number of features (i.e. number of freedom degrees).

The next experiment demonstrates the case when number of features is comparable with number of objects. Mushroom data set from UCI repository was taken as benchmark problem. There are 111 features in the data set after conversion to numeric features. Since there are 8124 objects in the data set different subsets of size 200 (Sample 1 to 5) were taken randomly preserving class prior probabilities. Tables 5, 6 and 7 report about the results. As before all algorithms show comparable performance in terms of error rates. However, fuzzy variant of BLogReg demonstrates significantly lower sparsity comparing to exact procedure BLogReg. This happens because Bayesian criterion $Q$ is no more unimodal function (as criterion $F$ was), nor its approximation $\hat{Q}$. When there

*Table 4.* Sparsity together with standard deviations

| Data set | #Features | BLogReg | FuzzyBLogReg |
|---|---|---|---|
| Bupa liver disorders | 6 | $5.30 \pm 0.27$ | $5.10 \pm 0.22$ |
| German credit numeric | 24 | $18.50 \pm 1.06$ | $17.20 \pm 0.45$ |
| Heart | 20 | $11.10 \pm 0.89$ | $8.90 \pm 0.89$ |
| Australian | 38 | $19.60 \pm 1.47$ | $17.30 \pm 1.48$ |
| Pima-indians-diabetes | 8 | $6.90 \pm 0.42$ | $6.70 \pm 0.27$ |
| Thyroid-sick.test | 31 | $11.60 \pm 0.42$ | $11.30 \pm 0.67$ |
| Wisconsin Diagnostic Breast Cancer | 30 | $9.80 \pm 0.27$ | $10.30 \pm 0.67$ |
| Wisconsin Prognostic Breast Cancer | 33 | $7.00 \pm 0.00$ | $11.30 \pm 5.14$ |

*Table 5.* Error rates together with standard deviations (in percents).

| Data set | BLogReg | FuzzyBLogReg | LinearRVM | LinearSVM |
|---|---|---|---|---|
| Agaricus-lepiota (mushroom) 1 | $1.48 \pm 0.30$ | $1.09 \pm 0.14$ | $2.62 \pm 0.51$ | $1.28 \pm 0.21$ |
| Agaricus-lepiota (mushroom) 2 | $1.73 \pm 0.00$ | $1.09 \pm 0.14$ | $2.81 \pm 0.62$ | $0.99 \pm 0.00$ |
| Agaricus-lepiota (mushroom) 3 | $1.53 \pm 0.21$ | $1.33 \pm 0.37$ | $3.46 \pm 0.68$ | $1.33 \pm 0.37$ |
| Agaricus-lepiota (mushroom) 4 | $2.12 \pm 0.62$ | $1.33 \pm 0.77$ | $4.94 \pm 2.93$ | $1.68 \pm 0.88$ |
| Agaricus-lepiota (mushroom) 5 | $1.88 \pm 1.01$ | $1.33 \pm 1.11$ | $4.59 \pm 0.81$ | $1.33 \pm 1.11$ |
| **Rank** | **15.00** | **7.00** | **20.00** | **8.00** |
| Color legend | Place 1 | Place 2 | Place 3 | Place 4 |

are few features and many objects in the training set both methods tend to converge to the same or close points. But with the increase of number of features, the number of local extrema increases and approximate method becomes dependant on the choice of starting point. In particular starting optimization from $\vec{w}_{ML}$ yields to accurate but non-sparse decision rules. For the cases when sparsity is of extreme importance BLogReg should be preferred.

## 6. Conclusions

In this paper we demonstrate that the regularization parameter arising in the sparse logistic regression algorithm (SLogReg) of (Shevade & Keerthi, 2003) can be eliminated, via Bayesian marginalization. (Authors, 1900a) and (Authors, 1900b) clearly demonstrate that the proposed algorithm for sparse logistic regression with Bayesian regularization (BLogReg) is competitive with the original SLogReg and RVM algorithms in terms of performance and sparsity. However, as the need for a cross-validation based model selection process is obviated, the improved algorithm is two to three orders of magnitude faster than its predecessor.

For optimization of new criterion $Q$ two procedures were proposed. The first one is based on (Shevade & Keerthi, 2003) coordinate descend method while the second one uses smooth approximation of criterion function $\hat{Q}$ with further Newton optimization algorithm. All algorithms demonstrate comparable performance (including Support Vector Machines and Relevance Vector Machines with inner product as kernel (basis) function). However, training speed of fuzzy variant is higher comparing to exact procedure. Meanwhile, in tasks when number of features is comparable with number of objects fuzzy variant shows significantly lower sparsity and in these situations for promoting sparsity exact BLogReg should be preferred.

## References

Authors, A. (1900a). Hidden for anonymity.

Authors, A. (1900b). Hidden for anonymity.

Buntine, W. L., & Weigend, A. S. (1991). Bayesian back-propagation. *Complex Systems*, *5*, 603–643.

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, *10*, 1895–1924.

*Table 6.* Training time together with standard deviations (in seconds).

| **Data set** | **BLogReg** | **FuzzyBLogReg** |
|---|---|---|
| Agaricus-lepiota (mushroom) 1 | $2.48 \pm 0.40$ | $0.15 \pm 0.05$ |
| Agaricus-lepiota (mushroom) 2 | $2.58 \pm 0.36$ | $0.13 \pm 0.01$ |
| Agaricus-lepiota (mushroom) 3 | $2.64 \pm 0.46$ | $0.13 \pm 0.01$ |
| Agaricus-lepiota (mushroom) 4 | $2.73 \pm 0.48$ | $0.14 \pm 0.02$ |
| Agaricus-lepiota (mushroom) 5 | $2.77 \pm 0.42$ | $0.12 \pm 0.01$ |

*Table 7.* Sparsity together with standard deviations

| **Data set** | **#Features** | **BLogReg** | **FuzzyBLogReg** |
|---|---|---|---|
| Agaricus-lepiota (mushroom) 1 | 111 | $17.60 \pm 0.42$ | $110.40 \pm 0.55$ |
| Agaricus-lepiota (mushroom) 2 | 111 | $17.90 \pm 0.22$ | $111.00 \pm 0.00$ |
| Agaricus-lepiota (mushroom) 3 | 111 | $16.20 \pm 0.76$ | $107.60 \pm 3.34$ |
| Agaricus-lepiota (mushroom) 4 | 111 | $17.60 \pm 1.29$ | $107.30 \pm 3.07$ |
| Agaricus-lepiota (mushroom) 5 | 111 | $16.70 \pm 1.25$ | $109.40 \pm 1.67$ |


Figueiredo, M. (2003). Adaptive sparseness for supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*, 1150–1159.

Guyon, I. M., Boser, B. E., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. *Fifth Annual Workshop on Computational Learning Theory*.

Lehrach, W. P., Husmeier, D., & Williams, C. K. I. (2006). A regularized discriminative model for the prediction of peptide-peptide interactions. *Bioinformatics*, *22*, 532–540.

MacKay, D. J. C. (1992a). Bayesian interpolation. *Neural Computation*, *4*, 415–447.

MacKay, D. J. C. (1992b). The evidence framework applied to classification networks. *Neural Computation*, *4*, 720–736.

MacKay, D. J. C. (1992c). A practical Bayesian framework for backprop networks. *Neural Computation*, *4*, 448–472.

MacKay, D. J. C. (1994). Hyperparameters : optimise or integrate out? In G. Heidbreder (Ed.), *Maximum entropy and bayesian methods*. Kluwer.

Newman, D. J., Hettich, S., Blake, C. L., & Merz, C. J. (1998). UCI repository of machine learning databases.

Shevade, S. K., & Keerthi, S. S. (2003). A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, *19*, 2246–2253.

Tikhonov, A. N., & Arsenin, V. Y. (1977). *Solutions of ill-posed problems*. New York: John Wiley.

Tipping, M. E. (2001). Sparse Bayesian learning and the Relevance Vector Machine. *Journal of Machine Learning Research*, *1*, 211–244.

Williams, P. M. (1995). Bayesian regularization and pruning using a Laplace prior. *Neural Computation*, *7*, 117–143.