Surrogate Maximization/Minimization Algorithms for AdaBoost and the Logistic Regression Model

Zhihua Zhang ZHZHANG@CS.UST.HK
James T. Kwok JAMESK@CS.UST.HK
Dit-Yan Yeung DYYEUNG@CS.UST.HK
Department of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong

Abstract

Surrogate maximization (or minimization) (SM) algorithms are a family of algorithms that can be regarded as a generalization of expectation-maximization (EM) algorithms. There are three major approaches to the construction of surrogate functions, all relying on the convexity of some function. In this paper, we solve the boosting problem by proposing SM algorithms for the corresponding optimization problem. Specifically, for AdaBoost, we derive an SM algorithm that can be shown to be identical to the algorithm proposed by Collins et al. (2002) based on Bregman distance. More importantly, for LogitBoost (or logistic boosting), we use several methods to construct different surrogate functions which result in different SM algorithms. By combining multiple methods, we are able to derive an SM algorithm that is also the same as an algorithm derived by Collins et al. (2002). Our approach based on SM algorithms is much simpler and convergence results follow naturally.

1. Introduction

Boosting is among the most successful recent developments in the machine learning community. Essentially, boosting can be formulated as an optimization problem. In general, there exist three different types of boosting algorithms according to the loss functions used. AdaBoost (for "adaptive boosting"; also called ExpBoost) is based on the exponential loss function, LogitBoost is based on the log loss function (or negative log likelihood function), and L_2Boost is based on the squared loss function. For both AdaBoost and LogitBoost, since there exists an explicit equivalence relationship between them and logistic exponential models, maximum likelihood estimation methods are natural choices for solving the optimization problem. For L_2Boost, on the other hand, some least squares fitting methods, such as the functional gradient descent algorithm (Friedman et al., 2000; Friedman, 2001; Bühlmann & Yu, 2003), can be used.

In this paper, we focus on AdaBoost and LogitBoost. Our work has been motivated by some recent works (Kivinen & Warmuth, 1999; Lafferty, 1999; Collins et al., 2002) which are based on Bregman distance optimization methods. Simply put, the Bregman distance between two vectors is defined via a convex function on a convex set that contains these two vectors. Della Pietra et al. (1997) applied Bregman distance optimization to log-linear models, and Della Pietra et al. (1997) and Collins et al. (2002) discussed its relationship with generalized iterative scaling (Darroch & Ratcliff, 1972) for log-linear models. Like generalized iterative scaling, the core spirit of Bregman distance optimization is from convex analysis (Rockafellar, 1970). It is an interesting novelty to unify AdaBoost and logistic regression within the framework of Bregman distance. However, this approach requires considerable mathematical skills to construct a Bregman function that matches the problem in question. Furthermore, in order to use Bregman distance optimization, it is necessary to reformulate the unconstrained optimization problem for boosting as an equivalent constrained optimization problem subject to linear constraints. This makes the problem much more technically involved. Della Pietra et al. (2001) also recognized these difficulties and sought to use the Legendre transformation technique (Rockafel-

Appearing in *Proceedings of the 21st International Conference on Machine Learning*, Banff, Canada, 2004.

lar, 1970). The main difference between (Della Pietra et al., 2001) and (Collins et al., 2002) is that the former works with the argument at which a convex conjugate takes on its value, while the latter works with the value of the functional itself. This makes it more natural to formulate a duality theorem.

Since log likelihood functions are closely related to convex (or concave) functions, convexity plays a central role in both statistical inference and computational statistics. In computational statistics, a successful example is the well-known expectation-maximization (EM) algorithm (Dempster et al., 1977). Becker et al. (1997) and Lange et al. (2000) demonstrated that the EM algorithm can be derived from either Jensen's inequality or the concavity property of the log function. Along this line, a family of EM-like algorithms without missing data (Becker et al., 1997) have been devised to handle cases involving no missing data. Lange et al. (2000) unified this family of algorithms within the framework of the so-called *optimization transfer* algorithms, in which all algorithms rely on an optimizing function to serve as a surrogate function for the objective function of the optimization problem. By invoking convexity arguments, a general principle providing guidelines on constructing surrogate functions, as well as some specific examples for special cases, have been discussed in (Lange et al., 2000). Depending upon the context, this often relies on three important tools, namely, *Jensen's inequality*, *first-order Taylor approximation*, and the *low quadratic bound principle*.

Optimization transfer algorithms are very efficient because they can make an intractable or complex optimization problem simpler. For example, optimization transfer decouples the correlation among parameters so that we can estimate the parameters in parallel. It can also locally linearize a convex function near some value to make the problem at hand more tractable. It can avoid the computational problem of inverting large matrices as required by Newton's method. Moreover, optimization transfer enjoys the same local convergence properties as standard EM.

Some other names have been used for optimization transfer methods. In the context of multidimensional scaling (MDS), optimization transfer is referred to as *iterative majorization*, while in convex optimization, it is usually called the *auxiliary function* method. To contrast it with the standard EM algorithm for missing data problems, Meng (2000) suggested to refer to it as the *SM algorithm*. Here, S stands for the surrogate step while M stands for the maximization (or minimization, depending on the optimization problem at hand) step. In this paper, we prefer the name 'SM algorithm' as it reflects more accurately the spirit of this family of algorithms.

Motivated by the idea of using surrogate functions, we apply the SM algorithm to the optimization problem corresponding to boosting. Based on Jensen's inequality and first-order Taylor approximation, we devise SM algorithms for both AdaBoost and LogitBoost. The devised SM algorithms are exactly equivalent to the parallel Bregman distance algorithms of Collins et al. (2002), and thus our method can be seen as providing a new derivation for their algorithms. Compared with (Della Pietra et al., 2001) and (Collins et al., 2002), the mathematical skills required for our approach are much simpler because we only need to utilize Jensen's inequality or first-order Taylor approximation over a convex function. More importantly, our approach naturally guarantees convergence of the iterative algorithms for both AdaBoost and LogitBoost.

In this paper, we pay more attention to LogitBoost in order to show the potential of SM algorithms in machine learning. Although *every road leads to Rome*, a good traveling manual is still necessary. Our goal is to demonstrate how to use three popular methods for constructing the surrogate function. Based on Jensen's inequality, we decouple the correlation among the estimated parameters and decompose the original high-dimensional optimization problem into a set of one-dimensional sub-problems, allowing us to handle each sub-problem separately. Although we cannot obtain a one-step closed-form iterative procedure, we present a gradient SM algorithm by borrowing ideas from the gradient EM algorithm (Lange, 1995). Moreover, we show that the iterative procedure of Collins et al. (2002) can be regarded as a generalized SM algorithm analogous to the generalized EM algorithm (Dempster et al., 1977). Based on first-order Taylor approximation, we express the original objective function as the difference of two convex functions (i.e., a convex function plus a concave function), leading to a quadratic surrogate function. Based on the low quadratic bound principle (Böhning & Lindsay, 1988), we devise a quadratic SM algorithm. The essence of quadratic SM algorithms is to approximate the Hessian matrix in Newton's method with a simple positive definite matrix. In particular, quadratic SM uses a constant matrix. This implies that we need to compute the inverse of the Hessian matrix just once. As a result, this can avoid the need for inverting large matrices in each iterative step. Finally, based on the combination of Jensen's inequality and first-order Taylor approximation, we present the fourth method for constructing surrogate functions. More surprisingly, using this method, we can devise SM algorithms which are

equivalent to the parallel Bregman distance algorithms of Collins et al. (2002).

2. AdaBoost and Logistic Regression

Let $\mathcal{T} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ be a finite set of training examples, where each instance $\mathbf{x}_i$ from a domain or instance space $\mathcal{X}$ corresponds to a label $y_i \in \{-1, +1\}$. We also assume that we are given a set of real-valued functions, $h_1, \ldots, h_m$, on $\mathcal{X}$. In the boosting literature, these h_i's are usually called *weak* or *base hypotheses*.

Boosting aims at labeling the $\mathbf{x}_i$'s using a linear combination of these weak hypotheses. In other words, we want to find a parameter vector $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_m)^T \in \mathbb{R}^m$ such that $f_\lambda(\mathbf{x}_i) = \sum_{j=1}^m \lambda_j h_j(\mathbf{x}_i)$ is a good approximation of the underlying function for class labels. Instead of using f_λ directly as a classification rule, we usually postulate that the y_i's are determined by a probabilistic model associated with $f_\lambda(\mathbf{x}_i)$. For example, Friedman et al. (2000) suggested that the posterior probability of y is given by a logistic function of $f_\lambda(\mathbf{x})$, as

$$\hat{P}(y|\mathbf{x}) = \frac{1}{1 + e^{-y \sum_{j=1}^m \lambda_j h_j(\mathbf{x})}}. \tag{1}$$

Analogously, AdaBoost is based on the exponential loss function

$$L_e(\boldsymbol{\lambda}) = \sum_{i=1}^n e^{-y_i \sum_{j=1}^m \lambda_j h_j(\mathbf{x}_i)}, \tag{2}$$

while LogitBoost is based on the following loss function

$$L_l(\boldsymbol{\lambda}) = \sum_{i=1}^n \ln\left(1 + e^{-y_i \sum_{j=1}^m \lambda_j h_j(\mathbf{x}_i)}\right). \tag{3}$$

Clearly, LogitBoost is equivalent to the problem of maximizing the log likelihood function of the logistic regression model.

3. Generic Structure of SM Algorithm

In many applications, we have to consider the problem of maximizing an arbitrary function $L(\boldsymbol{\theta})$ w.r.t. some parameter $\boldsymbol{\theta}$. Given an estimate $\boldsymbol{\theta}(t)$ at the tth iteration, the SM algorithm (Lange et al., 2000; Meng, 2000) consists of the following two steps:

1. **Surrogate Step (S-Step):** Substitute a surrogate function $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}(t))$ for $L(\boldsymbol{\theta})$, such that

$$L(\boldsymbol{\theta}) \geq Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}(t)) \tag{4}$$

   for all $\boldsymbol{\theta}$, with equality holding at $\boldsymbol{\theta} = \boldsymbol{\theta}(t)$.

2. **Maximization Step (M-Step):** Obtain the next parameter estimate $\boldsymbol{\theta}(t+1)$ by maximizing the surrogate function $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}(t))$ w.r.t. $\boldsymbol{\theta}$, i.e.,

$$\boldsymbol{\theta}(t+1) = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}(t)). \tag{5}$$

The SM algorithm aims at transforming intractable optimization problems to tractable ones. For example, this can be achieved by decoupling the correlation among parameters so that each parameter can be estimated separately in a one-step closed-form iterative manner. Another advantage of using the SM algorithm is that we can avoid inverting large matrices as required by Newton's method. Moreover, as

$$L(\boldsymbol{\theta}(t+1)) \geq Q(\boldsymbol{\theta}(t+1)|\boldsymbol{\theta}(t)) \geq Q(\boldsymbol{\theta}(t)|\boldsymbol{\theta}(t)) = L(\boldsymbol{\theta}(t)),$$

the SM algorithm also enjoys the same local convergence properties as the standard EM algorithm.

Despite these nice properties, it is not always possible to obtain a closed-form solution for $\boldsymbol{\theta}(t+1)$ in the M-step. In the same spirit as the generalized EM algorithm (Dempster et al., 1977), we can use a generalized SM algorithm. That is, instead of maximizing $Q(\boldsymbol{\theta}|\boldsymbol{\theta}(t))$, we only attempt to find a $\boldsymbol{\theta}(t+1)$ such that $Q(\boldsymbol{\theta}(t+1)|\boldsymbol{\theta}(t)) \geq Q(\boldsymbol{\theta}(t)|\boldsymbol{\theta}(t))$. Alternatively, in the same spirit as the gradient EM algorithm (Lange, 1995), we may also devise a gradient SM algorithm, as:

$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) - (\nabla^2 Q(\boldsymbol{\theta}(t)|\boldsymbol{\theta}(t)))^{-1} \nabla L(\boldsymbol{\theta}(t)).$$

Note that the above-mentioned SM algorithm can be applied equally well to the minimization of $L(\boldsymbol{\theta})$, by simply reversing the inequality sign in (4) and changing the 'max' to 'min' in (5). Therefore, in the sequel, 'M' stands for either maximization or minimization depending on the optimization problem.

Clearly, construction of the surrogate function is key to the SM algorithm. On the one hand, the closer is the surrogate function to $L(\boldsymbol{\theta})$, the more efficient is the SM algorithm. On the other hand, a good surrogate function should preferably have a closed-form solution in the M-step. Lange et al. (2000) described some general principles as well as three methods in particular for the design of surrogate functions, in which function convexity plays a central role.

Suppose $f : \mathcal{S} \to (-\infty, +\infty]$ is convex on a closed convex set $\mathcal{S} \subseteq \mathbb{R}^q$. The first method stems from Jensen's inequality

$$f(\sum_{i=1}^k \alpha_i \mathbf{u}_i) \leq \sum_{i=1}^k \alpha_i f(\mathbf{u}_i),$$

where $\alpha_1 \geq 0, \ldots, \alpha_k \geq 0$ and $\sum_{i=1}^{k} \alpha_i = 1$. This inequality can be used to decouple the correlation among the $\mathbf{u}_i$'s. The second method makes use of the following property. When $f(\cdot)$ is also differentiable on its domain $\mathcal{S}$, it can be linearized by first-order Taylor approximation, as

$$f(\mathbf{u}) \geq f(\mathbf{v}) + \nabla f(\mathbf{v})^T(\mathbf{u}-\mathbf{v}), \text{ for } \mathbf{u}, \mathbf{v} \in \mathcal{S}.$$

Since most continuous functions can be expressed as the difference of two convex functions, we can often use this trick in constructing the surrogate function. For example, if for any $f(\mathbf{u}) = g(\mathbf{u}) - h(\mathbf{u})$ where both $g(\mathbf{u})$ and $h(\mathbf{u})$ are convex, we can write $f(\mathbf{u}) \leq g(\mathbf{u}) - h(\mathbf{v}) - \nabla h(\mathbf{v})^T(\mathbf{u}-\mathbf{v})$. The third method uses the low quadratic bound principle (Böhning & Lindsay, 1988). Suppose there exists a $\mathbf{u}$-independent positive semi-definite matrix $\mathbf{B}$ such that $\mathbf{B} - \nabla^2 f(\mathbf{u})$ is positive semi-definite. Then, it can be shown that

$$f(\mathbf{u}) \leq f(\mathbf{v}) + \nabla f(\mathbf{v})^T(\mathbf{u}-\mathbf{v}) + \frac{1}{2}(\mathbf{u}-\mathbf{v})^T\mathbf{B}(\mathbf{u}-\mathbf{v}).$$

This is often used to define a quadratic surrogate function that can avoid the inversion of the Hessian matrix in Newton's method.

4. SM Algorithm for AdaBoost

In this section, we present an SM algorithm for AdaBoost. Let us define

$$g_{ij} = -y_i h_j(\mathbf{x}_i) \tag{6}$$

and $\mathbf{g}_i = (g_{i1}, \ldots, g_{im})^T$. As in (Collins et al., 2002), we assume that $\sum_{j=1}^{m} |g_{ij}| \leq 1$. Moreover, without loss of generality, we assume throughout this paper that $g_{ij} \neq 0$ for all i, j. If there exists some $g_{ij} = 0$, we can simply remove the corresponding term and still obtain the same results below.

Let us denote the tth iterate of λ_j by $\lambda_j(t)$. From (2), we have

$$\begin{aligned} L_e(\boldsymbol{\lambda}) &= \sum_{i=1}^{n} e^{\sum_{j=1}^{m} |g_{ij}| \frac{g_{ij}}{|g_{ij}|}(\lambda_j - \lambda_j(t)) + \boldsymbol{\lambda}(t)^T \mathbf{g}_i} \\ &= \sum_{i=1}^{n} e^{\sum_{j=1}^{m} |g_{ij}| \frac{g_{ij}}{|g_{ij}|}(\lambda_j - \lambda_j(t)) + (1-\alpha_i)0} \times e^{\boldsymbol{\lambda}(t)^T \mathbf{g}_i}, \end{aligned}$$

where

$$\alpha_i = \sum_{j=1}^{m} |g_{ij}|. \tag{7}$$

Since $\exp(\cdot)$ is convex, it can be shown that

$$\begin{aligned} L_e &\leq \sum_{i=1}^{n} e^{\boldsymbol{\lambda}(t)^T \mathbf{g}_i} \Big\{ 1 - \alpha_i + \sum_{j=1}^{m} |g_{ij}| e^{\frac{g_{ij}}{|g_{ij}|}(\lambda_j - \lambda_j(t))} \Big\} \\ &\equiv Q_e(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t)). \end{aligned}$$

Clearly, $Q_e(\boldsymbol{\lambda}(t)|\boldsymbol{\lambda}(t)) = L_e(\boldsymbol{\lambda}(t))$, and thus the RHS can be used as a surrogate function of $L_e(\lambda)$. Note also that $Q_e(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ has decoupled the relationship among the λ_j's. To minimize $Q_e(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ w.r.t. λ_j's, we set

$$\frac{\partial Q_e(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))}{\partial \lambda_j} = \sum_{i=1}^{n} g_{ij} e^{\boldsymbol{\lambda}(t)^T \mathbf{g}_i} e^{\frac{g_{ij}}{|g_{ij}|}(\lambda_j - \lambda_j(t))}$$

to zero, and obtain

$$\sum_{i \in S_j^+} |g_{ij}| e^{\boldsymbol{\lambda}(t)^T \mathbf{g}_i} e^{\lambda_j - \lambda_j(t)} = \sum_{i \in S_j^-} |g_{ij}| e^{\boldsymbol{\lambda}(t)^T \mathbf{g}_i} e^{\lambda_j(t) - \lambda_j},$$

where $S_j^+ = \{i : g_{ij} > 0\}$ and $S_j^- = \{i : g_{ij} < 0\}$. We take log on both sides and simplify to obtain the following update equation for λ_j, as:

$$\lambda_j(t+1) = \lambda_j(t) + \frac{1}{2} \ln \left(\frac{\sum_{i \in S_j^-} |g_{ij}| e^{\boldsymbol{\lambda}(t)^T \mathbf{g}_i}}{\sum_{i \in S_j^+} |g_{ij}| e^{\boldsymbol{\lambda}(t)^T \mathbf{g}_i}} \right).$$

As $L_e(\boldsymbol{\lambda}(t+1)) \leq Q_e(\boldsymbol{\lambda}(t+1)|\boldsymbol{\lambda}(t)) \leq Q_e(\boldsymbol{\lambda}(t)|\boldsymbol{\lambda}(t)) = L_e(\boldsymbol{\lambda}(t))$, local convergence is guaranteed.

Note that this iterative procedure is equivalent to that of the parallel-update optimization algorithm of (Collins et al., 2002). However, while ours is built upon the SM algorithm and relies only on the convexity of the exponential function, the one in (Collins et al., 2002) requires the construction of a Bregman distance which is much more mathematically involved. Moreover, convergence of our algorithm follows directly from the SM algorithm. In fact, the Bregman distance optimization algorithm of (Collins et al., 2002) can also work with the first-order Taylor expansion of a convex function. However, the argument of this convex function is itself also a function.

5. SM Algorithm for LogitBoost

In this section, we apply the SM algorithm to LogitBoost. From (1) and (3), we can see that LogitBoost is equivalent to maximizing the conditional log likelihood. In the following subsections, we present several SM algorithms for LogitBoost based on four different methods for constructing the surrogate function.

5.1. Using Jensen's Inequality

Using g_{ij} and α_i as defined in (6) and (7), we can rewrite $L_l(\boldsymbol{\lambda})$ in (3) as

$$\begin{aligned} L_l(\boldsymbol{\lambda}) &= \sum_{i=1}^{n} \ln \Big\{ 1 + \\ &\qquad e^{\sum_{j=1}^{m} |g_{ij}| \left[\frac{g_{ij}}{|g_{ij}|}(\lambda_j - \lambda_j(t)) + \boldsymbol{\lambda}(t)^T \mathbf{g}_i \right] + (1-\alpha_i)\boldsymbol{\lambda}(t)^T \mathbf{g}_i} \Big\}. \end{aligned}$$

Since $\frac{d^2 \ln(1+\exp(u))}{du^2} = \frac{\exp(u)}{(1+\exp(u))^2} > 0$, $\ln(1+\exp(\cdot))$ is convex, and hence

$$\begin{aligned} & L_l(\boldsymbol{\lambda}) \\ \leq & \sum_{i=1}^{n}(1-\alpha_i)\ln(1+e^{\boldsymbol{\lambda}(t)^T\mathbf{g}_i}) \\ & + \sum_{i=1}^{n}\left\{\sum_{j=1}^{m}|g_{ij}|\ln\left[1+e^{\frac{g_{ij}}{|g_{ij}|}(\lambda_j-\lambda_j(t))+\boldsymbol{\lambda}(t)^T\mathbf{g}_i}\right]\right\} \\ \equiv & Q_l(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t)). \end{aligned} \quad (8)$$

It is easy to show that $Q_l(\boldsymbol{\lambda}(t)|\boldsymbol{\lambda}(t)) = L_l(\boldsymbol{\lambda}(t))$. Hence, $Q_l(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ can be used as a surrogate function of $L_l(\boldsymbol{\lambda})$. As in Section 5, we can minimize $Q_l(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ w.r.t. the λ_j's, by setting the partial derivative

$$\begin{aligned} \frac{\partial Q_l(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))}{\partial\lambda_j} = & \sum_{i\in S_j^+}|g_{ij}|\frac{e^{\boldsymbol{\lambda}(t)^T\mathbf{g}_i}e^{\lambda_j-\lambda_j(t)}}{1+e^{\boldsymbol{\lambda}(t)^T\mathbf{g}_i}e^{\lambda_j-\lambda_j(t)}} \\ & - \sum_{i\in S_j^-}|g_{ij}|\frac{e^{\boldsymbol{\lambda}(t)^T\mathbf{g}_i}e^{\lambda_j(t)-\lambda_j}}{1+e^{\boldsymbol{\lambda}(t)^T\mathbf{g}_i}e^{\lambda_j(t)-\lambda_j}} \end{aligned}$$

to zero. However, a closed-form solution cannot be found. There are two methods to tackle this problem. One is to employ a strategy similar to the generalized EM algorithm (Dempster et al., 1977), leading to a generalized SM algorithm. Alternatively, we can resort to a gradient SM algorithm analogous to the gradient EM algorithm (Lange, 1995). Here, we employ this strategy for LogitBoost. Using

$$\begin{aligned} \left.\frac{\partial Q_l(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))}{\partial\lambda_j}\right|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}(t)} = & \sum_{i=1}^{n}p_i(\boldsymbol{\lambda}(t))g_{ij}, \\ \left.\frac{\partial^2 Q_l(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))}{\partial\lambda_j^2}\right|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}(t)} = & \sum_{i=1}^{n}p_i(\boldsymbol{\lambda}(t))(1-p_i(\boldsymbol{\lambda}(t)))|g_{ij}|, \end{aligned}$$

where $p_i(\boldsymbol{\lambda}) = \frac{\exp(\boldsymbol{\lambda}^T\mathbf{g}_i)}{1+\exp(\boldsymbol{\lambda}^T\mathbf{g}_i)}$, we update the current parameter estimate $\lambda_j(t)$ to

$$\begin{aligned} \lambda_j(t+1) = & \lambda_j(t) - \left\{\sum_{i=1}^{n}p_i(\boldsymbol{\lambda}(t))(1-p_i(\boldsymbol{\lambda}(t)))|g_{ij}|\right\}^{-1} \\ & \times\sum_{i=1}^{n}p_i(\boldsymbol{\lambda}(t))g_{ij}. \end{aligned} \quad (9)$$

5.2. Using First-Order Taylor Approximation

Our point of departure is from that the function $f(u) = \ln\cosh\sqrt{u}$ for $u \in [0,\infty)$ is concave (Jaakkola & Jordan, 1997). Noting the following relationship

$$\ln(1+e^{\boldsymbol{\lambda}^T\mathbf{g}_i}) = \ln 2 + \frac{\boldsymbol{\lambda}^T\mathbf{g}_i}{2} + \ln\cosh(\frac{\boldsymbol{\lambda}^T\mathbf{g}_i}{2}), \quad (10)$$

and using the concavity of $f(u) = \ln\cosh\sqrt{u}$, we have

$$\begin{aligned} \ln\cosh(\frac{\boldsymbol{\lambda}^T\mathbf{g}_i}{2}) \leq & \ln\cosh(\frac{\boldsymbol{\lambda}(t)^T\mathbf{g}_i}{2}) \\ & + \frac{1}{4}(\boldsymbol{\lambda}-\boldsymbol{\lambda}(t))^T\beta_i(t)\mathbf{g}_i\mathbf{g}_i^T(\boldsymbol{\lambda}+\boldsymbol{\lambda}(t)), \end{aligned}$$

where $\beta_i(t)$ stands for the derivative of $\ln\cosh\sqrt{u}$ at $\sqrt{u} = |\boldsymbol{\lambda}(t)^T\mathbf{g}_i/2|$, and $\beta_i(t) = \frac{\tanh(\boldsymbol{\lambda}(t)^T\mathbf{g}_i/2)}{|\boldsymbol{\lambda}(t)^T\mathbf{g}_i|}$ when $\boldsymbol{\lambda}(t)^T\mathbf{g}_i \neq 0$ and $\beta_i(t) = \frac{1}{2}$ otherwise. Thus, we obtain a quadratic surrogate function

$$\begin{aligned} Q_f(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t)) = & n\ln 2 + \sum_{i=1}^{n}\left\{\frac{\boldsymbol{\lambda}^T\mathbf{g}_i}{2} + \ln\cosh(\frac{\boldsymbol{\lambda}(t)^T\mathbf{g}_i}{2})\right\} \\ & + \frac{1}{4}(\boldsymbol{\lambda}-\boldsymbol{\lambda}(t))^T\left\{\sum_{i=1}^{n}\beta_i(t)\mathbf{g}_i\mathbf{g}_i^T\right\}(\boldsymbol{\lambda}+\boldsymbol{\lambda}(t)). \end{aligned}$$

Minimization of $Q_f(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ w.r.t. $\boldsymbol{\lambda}$ results in a new one-step SM algorithm

$$\boldsymbol{\lambda}(t+1) = -\left\{\sum_{i=1}^{n}\beta_i(t)\mathbf{g}_i\mathbf{g}_i^T\right\}^{-1}\sum_{i=1}^{n}\mathbf{g}_i. \quad (11)$$

Since this SM algorithm follows exactly the setting of the standard SM algorithm, its convergence is naturally guaranteed. It is worth noting that (10) shows we indeed express the objective as the difference of two convex functions because a linear function is both convex and concave and the minus of a concave function is convex. The use of differences of convex (d.c.) functions is a very important strategy in convex optimization and has received much attention recently in machine learning.

5.3. Using the Low Quadratic Bound Principle

The original idea of the low quadratic bound principle was proposed by Böhning and Lindsay (1988). More specifically, let $L(\boldsymbol{\theta})$ be the objective function to be maximized, $\nabla L(\boldsymbol{\theta})$ the Fisher score vector and $\nabla^2 L(\boldsymbol{\theta})$ the Hessian matrix at $\boldsymbol{\theta} \in \mathbb{R}^r$. The low quadratic bound algorithm aims to find a negative definite $r \times r$ matrix $\mathbf{B}$ such that $\nabla^2 L(\boldsymbol{\theta}) \succeq \mathbf{B}$ for all $\boldsymbol{\theta}$.[1] Thus, one can define the surrogate function $Q(\boldsymbol{\theta} \mid \boldsymbol{\phi})$ of $L(\boldsymbol{\theta})$ as

$$Q(\boldsymbol{\theta}|\boldsymbol{\phi}) = L(\boldsymbol{\phi}) + (\boldsymbol{\theta}-\boldsymbol{\phi})^T\nabla L(\boldsymbol{\phi}) + \frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{\phi})^T\mathbf{B}(\boldsymbol{\theta}-\boldsymbol{\phi}).$$

Clearly, $L(\boldsymbol{\theta}) - Q(\boldsymbol{\theta} \mid \boldsymbol{\phi})$ attains its minimum at $\boldsymbol{\theta} = \boldsymbol{\phi}$. Since $Q(\boldsymbol{\theta} \mid \boldsymbol{\phi})$ is a quadratic function, its convexity implies that it has only one maximum. If we let $\boldsymbol{\phi}$ be the tth estimate of $\boldsymbol{\theta}$, denoted $\boldsymbol{\theta}(t)$, then maximizing $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}(t))$ w.r.t. $\boldsymbol{\theta}$ yields the $(t+1)$th estimate of $\boldsymbol{\theta}$ as

$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) - \mathbf{B}^{-1}\nabla L(\boldsymbol{\theta}(t)). \quad (12)$$

[1] Here $\mathbf{C} \succeq \mathbf{D}$ means $\mathbf{C} - \mathbf{D}$ is positive semi-definite.

Convergence of this iterative algorithm has also been proven (Böhning & Lindsay, 1988).

We now apply the low quadratic bound principle to LogitBoost. First, we compute the Fisher score vector and Hessian matrix as

$$\begin{aligned}\nabla L_l(\boldsymbol{\lambda}) &= \sum_{i=1}^{n} p_i(\boldsymbol{\lambda})\mathbf{g}_i, \\ \nabla^2 L_l(\boldsymbol{\lambda}) &= \sum_{i=1}^{n} p_i(\boldsymbol{\lambda})(1-p_i(\boldsymbol{\lambda}))\mathbf{g}_i\mathbf{g}_i^T.\end{aligned}$$

Since $p_i(\boldsymbol{\lambda})(1-p_i(\boldsymbol{\lambda})) \leq \frac{1}{4}$, we have

$$\nabla^2 L_l(\boldsymbol{\lambda}) \preceq \frac{1}{4}\mathbf{G}\mathbf{G}^T,$$

where $\mathbf{G} = [\mathbf{g}_1, \ldots, \mathbf{g}_n]$. Now, given the tth iterates $\lambda_j(t)$'s of λ_j's, we can define a surrogate function of $L_l(\boldsymbol{\lambda})$ as

$$\begin{aligned}Q_q(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t)) &= L_l(\boldsymbol{\lambda}(t)) + (\boldsymbol{\lambda}-\boldsymbol{\lambda}(t))^T \nabla L_l(\boldsymbol{\lambda}(t)) \\ &\quad + \frac{1}{8}(\boldsymbol{\lambda}-\boldsymbol{\lambda}(t))^T \mathbf{G}\mathbf{G}^T(\boldsymbol{\lambda}-\boldsymbol{\lambda}(t)).\end{aligned}$$

Then minimization of $Q_q(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ gives rise to the $(t+1)$th iterate of $\boldsymbol{\lambda}$, as:

$$\boldsymbol{\lambda}(t+1) = \boldsymbol{\lambda}(t) - 4(\mathbf{G}\mathbf{G})^{-1}\nabla L_l(\boldsymbol{\lambda}(t)). \tag{13}$$

We can see that the assumption $\sum_{j=1}^{m} |g_{ij}| \leq 1$ is not necessary for the SM algorithm.

5.4. Using Multiple Approaches

Usually the three approaches discussed above are separately used to construct a surrogate function depending upon the problem at hand. However, in some cases, it may be useful to combine multiple approaches together. Here we present a method for LogitBoost by combining Jensen's inequality and first-order Taylor approximation.[2]

Consider the $Q_l(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ in (8) and again work on $\ln(\cdot)$ with first-order Taylor approximation. Then

$$\begin{aligned}&\ln\left[1 + e^{\frac{g_{ij}}{|g_{ij}|}(\lambda_j - \lambda_j(t)) + \boldsymbol{\lambda}(t)^T\mathbf{g}_i}\right] \\ &\leq \ln\left[1 + e^{\boldsymbol{\lambda}(t)^T\mathbf{g}_i}\right] + \frac{\left(e^{\frac{g_{ij}}{|g_{ij}|}(\lambda_j - \lambda_j(t))} - 1\right)e^{\boldsymbol{\lambda}(t)^T\mathbf{g}_i}}{1 + e^{\boldsymbol{\lambda}(t)^T\mathbf{g}_i}}.\end{aligned}$$

By combining this with the expression for $Q_l(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$, we obtain a new surrogate function for $L_l(\boldsymbol{\lambda})$:

$$\begin{aligned}&Q_c(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t)) \\ &= \sum_{i=1}^{n} \ln\left(1 + e^{\sum_{j=1}^{m}\lambda_j(t)g_{ij}}\right) \\ &+ \sum_{i=1}^{n} p_i(\boldsymbol{\lambda}(t)) \sum_{j=1}^{m} |g_{ij}| \left\{ e^{\frac{g_{ij}}{|g_{ij}|}(\lambda_j - \lambda_j(t))} - 1 \right\}.\end{aligned} \tag{14}$$

The partial derivative of $Q_c(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ w.r.t. λ_j is

$$\begin{aligned}\frac{\partial Q_c(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))}{\partial \lambda_j} &= \sum_{i=1}^{n} p_i(\boldsymbol{\lambda}(t)) g_{ij} e^{\frac{g_{ij}}{|g_{ij}|}(\lambda_j - \lambda_j(t))} \\ &= \sum_{i \in S_j^+} p_i(\boldsymbol{\lambda}(t))|g_{ij}|e^{\lambda_j - \lambda_j(t)} - \\ &\quad \sum_{i \in S_j^-} p_i(\boldsymbol{\lambda}(t))|g_{ij}|e^{\lambda_j(t) - \lambda_j}.\end{aligned}$$

It is easy to find an exact analytical solution of $\operatorname{argmin}_{\boldsymbol{\lambda}} Q_c(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ as

$$\lambda_j(t+1) = \lambda_j(t) + \frac{1}{2}\ln\left(\frac{\sum_{i \in S_j^-}|g_{ij}|p_i(\boldsymbol{\lambda}(t))}{\sum_{i \in S_j^+}|g_{ij}|p_i(\boldsymbol{\lambda}(t))}\right). \tag{15}$$

Clearly, this is a standard SM algorithm and thus its convergence is guaranteed. Notice that this algorithm is equivalent to the parallel Bregman distance algorithm for LogitBoost proposed by Collins et al. (2002). However, our derivation is much simpler because we only utilize Jensen's inequality with the convexity of $\ln(1 + \exp(u))$ and first-order Taylor approximation with the concavity of $\ln(u)$.

5.5. Analysis and Discussion

From the previous subsections, we can see that multiple surrogate functions can be derived for the same objective function and hence multiple SM algorithms are resulted. A natural question to ask is what criteria should be used to guide the design of a good surrogate function. One intuitive criterion is the closeness of a surrogate function to the original objective function. Specifically, the closer is the surrogate function to the objective function, the better it will be. Another possible criterion is the tractability of the M-step. Obviously, a closed-form update equation is desirable.

Going back to the LogitBoost example above, it can be shown that

$$L_l(\boldsymbol{\lambda}) \leq Q_l(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t)) \leq Q_c(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t)),$$

and thus the surrogate function $Q_l(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ proposed in Section 5.1 is superior to $Q_c(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ proposed in

[2] Other combinations also exist. Due to space limit, these possibilities will be reported in a separate paper.

Section 5.4. On the other hand, while $Q_l(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ does not have a closed-form solution for the M-step, it is easy to show that an exact analytical solution exists for $Q_c(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$. Similarly, we can show that[3]

$$L_l(\boldsymbol{\lambda}) \leq Q_f(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t)) \leq Q_q(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t)).$$

From (11) and (13), we can see that both the SM algorithms based on $Q_f(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ and $Q_q(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ essentially amount to minimizing $L_l(\boldsymbol{\lambda})$ by Newton's method, but with the Hessian matrix $\nabla^2 L_l(\boldsymbol{\lambda})$ replaced by an approximation matrix. They can avoid the non-convergent problem of standard Newton's method. Since the latter method uses a constant matrix (i.e., $\mathbf{B}$), it only needs to compute the inverse of this constant matrix once during the whole iterative process. However, the former method has the same computational cost as Newton's method. Thus, in general, there has to be a tradeoff between the two criteria.

In passing, note that for $\boldsymbol{\lambda}(t+1)$ given in (15), we have

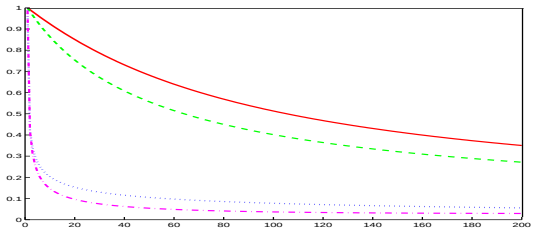
$$\begin{aligned} &Q_l(\boldsymbol{\lambda}(t+1)|\boldsymbol{\lambda}(t)) \leq Q_c(\boldsymbol{\lambda}(t+1)|\boldsymbol{\lambda}(t)) \\ &\leq Q_c(\boldsymbol{\lambda}(t)|\boldsymbol{\lambda}(t)) = L_l(\boldsymbol{\lambda}(t)) = Q_l(\boldsymbol{\lambda}(t)|\boldsymbol{\lambda}(t)). \end{aligned}$$

So the iterative procedure based on (15) defines a generalized SM algorithm for either the surrogate function $Q_c(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$ or the surrogate function $Q_l(\boldsymbol{\lambda}|\boldsymbol{\lambda}(t))$.

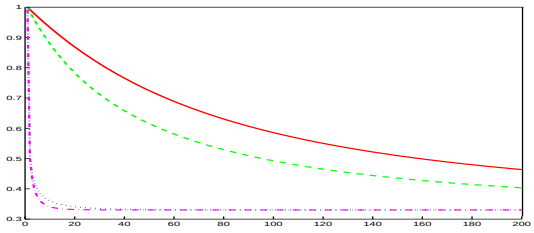
6. Experiments

In this section, we evaluate empirically several SM algorithms for LogitBoost, i.e., those defined by (9), (11), (13) and (15). For convenience, we refer to them as SM-J, SM-F, SM-Q and SM-C, respectively. We use synthetic data sets for two-class classification problems similar to those used by Collins et al. (2002). The first data set consists of 3,000 data points $\mathbf{x}_i \in \mathbb{R}^{100}$ sampled randomly from the normal distribution with zero mean and identity covariance matrix. To label these points, we first randomly generate a 100-dimensional hyperplane represented by a vector $\mathbf{w} \in \mathbb{R}^{100}$ subject to $\|\mathbf{w}\| = 1$ and then assign the label $y_i = \text{sgn}(\mathbf{w}^T \mathbf{x}_i)$ to each $\mathbf{x}_i$. After this labeling step, we perturb each point $\mathbf{x}_i$ by adding a random noise term $\varepsilon_i \sim \mathcal{N}(\mathbf{0}, 0.2 * \mathbf{I})$, leading to a new noisy data point $\mathbf{z}_i$. We use 1,000 points for training and the remaining 2,000 points for testing.

We run our experiments using two data sets, i.e., $\{\mathbf{x}_i\}$ with noise and $\{\mathbf{z}_i\}$ without noise. Specifically, we set $h_j(\mathbf{x}_i) = x_{ij}$ and $h_j(\mathbf{z}_i) = z_{ij}$, respectively, for the two data sets. First, for $i = 1, \ldots, 1000$ and $j = 1, \ldots, 100$, we calculate $g_{ij} = -y_i h_j(\mathbf{x}_i)$ (or $g_{ij} = -y_i h_j(\mathbf{z}_i)$) and set $g_{ij} = \frac{g_{ij}}{\sum_{j=1}^{100} |g_{ij}|}$ such that $\sum_{j=1}^{100} |g_{ij}| \leq 1$. Figure 1 shows the training losses, whose values are normalized to 1 when $\boldsymbol{\lambda}(0) = \mathbf{0}$, corresponding to the four SM algorithms. We can see that for both data sets, the convergence of SM-F and SM-Q is faster while the convergence of SM-J and SM-C is slower. On the noisy data set, both SM-F and SM-Q converge to a fixed point after about 15 iterations. The loss values of SM-J and SM-C, on the other hand, still decrease slowly even after 200 iterations. Note that these results are in line with our discussions above in Section 5.5.



(a) Without noise



(b) With noise

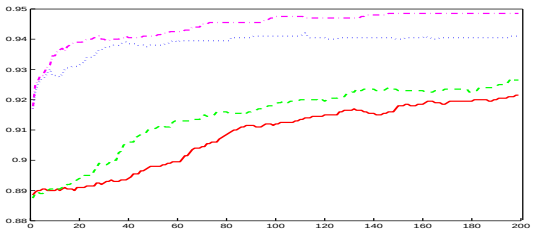
Figure 1. Training loss vs. number of iterations (SM-C: red solid; SM-J: green dashed; SM-Q: blue dotted; SM-F: magenta dash-dot).

We also report the classification accuracies. On the test data without noise, both SM-F and SM-Q outperform SM-J and SM-C. Moreover, SM-F outperforms SM-Q while SM-J outperforms SM-C. However, on the test data with noise, the classification accuracies of both SM-F and SM-Q decrease as the number of iterations increases. This shows the occurrence of overtraining for these two algorithms. Contrarily, SM-J and SM-C are rather robust to noise.

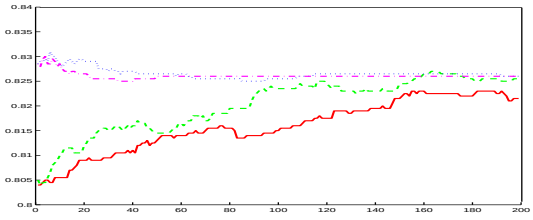
7. Concluding Remarks

In this paper, we have demonstrated the successful application of SM algorithms to boosting for two-class problems, particularly LogitBoost. SM algorithms can also be applied to boosting for multi-class problems. However, due to space limit, this more general case will be reported in an extended version of this paper.

[3]Due to space limit, the proof is omitted in the paper.



(a) Without noise



(b) With noise

Figure 2. Testing accuracy vs. number of iterations (SM-C: red solid; SM-J: green dashed; SM-Q: blue dotted; SM-F: magenta dash-dot).

Like EM algorithms for missing data problems, SM algorithms are gaining popularity in computational statistics for problems without missing data. Although EM algorithms are commonly used for solving many machine learning problems, SM algorithms are still rarely used. We hope this paper is successful in demonstrating the power of SM algorithms and will lead to wide applications in machine learning.

Acknowledgments

This research has been partially supported by the Research Grants Council of the Hong Kong Special Administrative Region under grants HKUST6195/02E and DAG03/04.EG36.

References

Becker, M. P., Yang, I., & Lange, K. (1997). EM algorithms without missing data. *Statistical Methods in Medical Research*, *6*, 38–54.

Böhning, D., & Lindsay, B. G. (1988). Monotonicity of quadratic-approximation algorithms. *Annals of the Institute of Statistical Mathematics*, *40*, 641–663.

Bühlmann, P., & Yu, B. (2003). Boosting with the L_2-loss: Regression and classification. *Journal of the American Statistical Association*, *98*, 324–339.

Collins, M., Schapire, R. E., & Singer, Y. (2002). Logistic regression, AdaBoost and Bregman distances. *Machine Learning*, *47*, 253–285.

Darroch, J. N., & Ratcliff, D. (1972). Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, *43*, 1470–1480.

Della Pietra, S., Della Pietra, V., & Lafferty, J. (1997). Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*, 380–393.

Della Pietra, S., Della Pietra, V., & Lafferty, J. (2001). *Duality and auxiliary functions for Bregman distances* (Technical Report CMU-CS-01-109). School of Computer Science, Carnegie-Mellon University.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B*, *39*, 1–38.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, *29*, 1189–1232.

Friedman, J. H., Hastie, T., & Tibshirani, R. (2000). Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, *28*, 337–374.

Jaakkola, T., & Jordan, M. (1997). A variational approach to Bayesian logistic regression models and their extensions. *The Sixth International Workshop on Artificial Intelligence and Statistics*.

Kivinen, J., & Warmuth, M. K. (1999). Boosting as entropy projection. *The Twelfth Annual Conference on Computational Learning Theory*.

Lafferty, J. (1999). Additive models, boosting and inference for generalized divergences. *The Twelfth Annual Conference on Computational Learning Theory* (pp. 125–133). Santa Cruz, CA.

Lange, K. (1995). A gradient algorithm locally equivalent to the EM algorithm. *Journal of the Royal Statistical Society, Series B*, *57*, 425–437.

Lange, K., Hunter, D. R., & Yang, I. (2000). Optimization transfer using surrogate objective functions with discussion. *Journal of Computational and Graphical Statistics*, *9*, 1–59.

Meng, X.-L. (2000). Discussion on "Optimization transfer using surrogate objective functions". *Journal of Computational and Graphical Statistics*, *9*, 35–43.

Rockafellar, T. (1970). *Convex analysis*. Princeton, New Jersey: Princeton University Press.