Neural Comput & Applic (2017) 28:1057–1067
DOI 10.1007/s00521-015-2118-2

ORIGINAL ARTICLE

Sparse learning of maximum likelihood model for optimization of complex loss function

Ning Zhang[1] · **Prathamesh Chandrasekar**[2]

Received: 1 July 2015 / Accepted: 4 November 2015 / Published online: 21 November 2015

Abstract Traditional machine learning methods usually minimize a simple loss function to learn a predictive model and then use a complex performance measure to measure the prediction performance. However, minimizing a simple loss function cannot guarantee an optimal performance. In this paper, we study the problem of optimizing the complex performance measure directly to obtain a predictive model. We proposed to construct a maximum likelihood model for this problem, and to learn the model parameter, we minimize a complex loss function corresponding to the desired complex performance measure. To optimize the loss function, we approximate the upper bound of the complex loss. We also propose to impose the sparsity to the model parameter to obtain a sparse model. An objective was constructed by combining the upper bound of the loss function and the sparsity of the model parameter, and we develop an iterative algorithm to minimize it by using the fast iterative shrinkage-thresholding algorithm framework. The experiments on optimization on three different complex performance measures, including F-score, receiver operating characteristic curve, and recall precision curve break-even point, over three real-world applications, aircraft event recognition of civil aviation safety, intrusion detection in wireless mesh networks, and image classification, show the advantages of the proposed method over state-of-the-art methods.

Keywords Machine learning · Complex multivariate performance · Sparse learning · Maximum likelihood · Civil aviation safety

1 Introduction

Machine learning aims to train a predictive model from a training set of input-out pairs and then use the model to predict an unknown output from a given test input [1, 13, 14, 17, 31, 34, 37]. In this paper, we focus on the machine learning problem of binary pattern classification. In this problem, each input is a feature vector of a data point, and each output is a binary class label of a data point, either positive or negative [5, 19, 21, 26, 28–30]. To learn the predictive model, i.e., the classification model, we usually compare the true class label of data point against the predicted label using a loss function, for example, hinge loss, logistic loss, and squared ℓ_2 norm loss. By minimizing the loss functions over the training set with regard to the parameter of the classification model, we can obtain an optimal classification model. To evaluate the performance of the model, we apply it to a set of test data points to predict their class labels and then compare the predicted class labels to their true class labels. This comparison can be conducted by using some multivariate performance measures, for example, prediction accuracy, F-score, area under receiver operating characteristic curve (AUROC) [2, 7, 22, 23], and precision–recall curve break-even point (PRBEP) [18, 20, 27, 33]. A problem of such machine learning procedure is that in the training process, we optimize a simple loss function, such as hinge loss, but in

✉ Ning Zhang
zhangning115@yahoo.com

Prathamesh Chandrasekar
prathameshchandrasekar@yahoo.com

[1] Guangzhou Civil Aviation College, Guangzhou 510403, China

[2] Uttar Pradesh Technical University, Lucknow, Uttar Pradesh 226021, India

the test process, we use a different and complex performance measure to evaluate the prediction results. It is obvious that the optimization of the loss function cannot lead to an optimization of the performance measure. For example, in the formulation of support vector machine (SVM), the hinge loss is minimized, but usually in the test procedure, the AUROC is used as a performance measure. However, in many real-world applications, the optimization of a specific performance measure is desired. To solve this problem, direct optimization of some complex loss functions corresponding to some desired performance measures is studied. These methods try to optimize a complex loss function in the objective function, and the loss functions are corresponding to the performance measure directly. By minimizing the loss function directly to obtain the predictive model, the desired performance measure can be optimized by the predictive model directly. In this paper, we study this problem and propose a novel method based on sparse learning and maximum likelihood optimization.

1.1 Related works

Some existing works proposed to optimize a complex multivariate loss function are briefly introduced as follows.

- Joachims [8] proposed to learn a support vector machine to optimize a complex loss function. In the proposed model, the complexity of the predictive model is reduced by minimizing squared ℓ_2 norm of the model parameter. To minimize the complex loss, its upper bound is approximated and minimized.
- Mao and Tsang [16] improved the Joachims's work by integrating feature selection to support vector machine for complex loss optimization. A weight is assigned to each feature before the predictive model is learned. Moreover, the feature weights and the predictive model parameter are learned jointly in an iterative algorithm.
- Li et al. [12] proposed a classifier adaptation method to extend Joachims's work. The predictive model is a combination of a base classifier and an adaptation function, and the learning of the optimal model is transferred to the learning of the parameter of the adaptation function.
- Zhang et al. [36] proposed a novel smoothing strategy by using Nesterov's accelerated gradient method to improve the convergence rate of the method proposed by Joachims [8]. This method, according to the results reported in [36], converges significantly faster than Joachims's method [8], but it does not scarify generalization ability.

Almost all the existing methods are limited to the support vector machine for multivariate complex loss function. This method uses a linear function to construct the predictive model and seek both the minimum complexity and loss.

1.2 Contribution

In this paper, we propose a novel predictive model to optimize a complex loss function. This model is based on the likelihood of a positive or negative class given an input feature vector of a data point. The likelihood function is constructed based on a sigmoid function of a linear function. Given a group of data points, we organize them as a data tuple, and the predicted class label tuple is the one that maximizes the logistic likelihood of the data tuple. The learning target is to learn a predictive model parameter, so that with the corresponding predicted class label tuple, the complex loss function can be minimized. Moreover, we also hope the model parameter can be as sparse as possible, so that only the useful can be kept in the model. To this end, we construct an objective function, which is composed of two terms. The first term is the ℓ_1 norm of the parameter to impose the sparsity of the parameter, and the second term is the complex loss function to seek the optimal desired performance measure. The problem is transferred to a minimization problem of the objective function with regard to the parameter. To solve this problem, we first approximate the upper bound of the complex as a logistic function of the parameter and then optimize it by using the fast iterative shrinkage-thresholding algorithm (FISTA). The novelty of this paper is summarized as follows:

1. For the first time, we propose to use the maximum likelihood model to construct a predictive model for the optimization of complex losses.
2. We construct a novel optimization problem for the learning of the model parameter by considering the sparsity of the model and the minimization of the complex loss jointly.
3. We develop a novel iterative algorithm to optimize the proposed minimization problem, and a novel method to approximate the upper bound of the complex loss. The approximation of the upper bound of the complex loss is obtained as a logistic function, and the problem is optimized by a FISTA algorithm.

1.3 Paper organization

This paper is organized as follows: In Sect. 2, we introduce the proposed method, in Sect. 3, we evaluate the proposed method on two real-world applications, and in Sect. 4, the paper is concluded and some future works are given.

2 Proposed method

2.1 Problem formulation

Suppose we have a data set of n data points, and we denote them as $\{(\mathbf{x}_i, y_i)\}|_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^d$ is the d-dimensional feature vector of the ith data point and $y_i \in \{+1, -1\}$ is it corresponding class label. We consider the data points as a data tuple, $\overline{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, and their corresponding class labels as a label tuple, $\overline{y} = (y_1, \ldots, y_n)$. Under the framework of complex performance measure optimization, we try to learn a multivariate mapping function to map the data tuple $\overline{\mathbf{x}}$ to a class label tuple $\overline{y}^* = (y_1^*, \ldots, y_n^*) \in \mathcal{Y}$, where $y_i^* \in \{+1, -1\}$ is the predicted label of the ith data point and $\mathcal{Y} = \{+1, -1\}^n$. To measure the performance of the multivariate mapping function, $\overline{h}(\overline{\mathbf{x}})$, we use a predefined complex loss function $\Delta(\overline{y}, \overline{y}^*)$ to compare the true class label tuple $\overline{y}$ against the predicted class label tuple $\overline{y}^*$.

To construct the multivariate mapping function $\overline{h}(\overline{\mathbf{x}})$, we proposed to apply a linear discriminate function to match the ith data point $\mathbf{x}_i$ against the ith class label y_i' in a candidate tuple $\overline{y}' = (y_1', \ldots, y_n')$,

$$f_{\mathbf{w}}(\mathbf{x}_i, y_i') = y_i' \mathbf{w}^\top \mathbf{x}_i, \tag{1}$$

where $\mathbf{w} = [w_1, \ldots, w_d] \in \mathbb{R}^d$ is the parameter vector of the function. And then we apply a sigmoid function to the response of this function to impose it to a range of [0, 1],

$$\begin{aligned} g(\mathbf{x}_i, y_i') &= \frac{1}{1 + \exp(-f(\mathbf{x}_i, y_i'))} \\ &= \frac{1}{1 + \exp(-y_i' \mathbf{w}^\top \mathbf{x}_i)}. \end{aligned} \tag{2}$$

Moreover,

$$\begin{aligned} g(\mathbf{x}_i, +1) &= \frac{1}{1 + \exp(-\mathbf{w}^\top \mathbf{x}_i)} \\ &= \frac{(1 + \exp(-\mathbf{w}^\top \mathbf{x}_i)) - \exp(-\mathbf{w}^\top \mathbf{x}_i))}{1 + \exp(-\mathbf{w}^\top \mathbf{x}_i)} \\ &= 1 - \frac{\exp(-\mathbf{w}^\top \mathbf{x}_i))}{1 + \exp(-\mathbf{w}^\top \mathbf{x}_i)} \\ &= 1 - \frac{1}{1 + \exp(\mathbf{w}^\top \mathbf{x}_i)} \\ &= 1 - g(\mathbf{x}_i, -1), \end{aligned} \tag{3}$$

thus we can treat $g(\mathbf{x}_i, y_i)$ as the conditional probability of $y = y_i'$ given $\mathbf{x} = \mathbf{x}_i$,

$$Pr(y = y_i' | \mathbf{x} = \mathbf{x}_i) = g(\mathbf{x}_i, y_i'). \tag{4}$$

We also assume that the data points in the tuple $\overline{\mathbf{x}}$ are conditionally independent from each other, and thus the conditional probability of $\overline{y} = \overline{y}'$ given the $\overline{\mathbf{x}}$ is

$$\begin{aligned} Pr(\overline{y} = \overline{y}' | \overline{\mathbf{x}}) &= \prod_{i=1}^n Pr(y = y_i' | \mathbf{x} = \mathbf{x}_i) \\ &= \prod_{i=1}^n \frac{1}{1 + \exp(-y_i' \mathbf{w}^\top \mathbf{x}_i)}. \end{aligned} \tag{5}$$

To constructed the complex mapping function, we map the data tuple to the class tuple $\overline{y}^*$ which can give the maximum log-likelihood,

$$\begin{aligned} y^* \leftarrow \overline{h}(\overline{\mathbf{x}}) &= \arg\max_{\overline{y}' \in \mathcal{Y}} \log(Pr(\overline{y} = \overline{y}' | \overline{\mathbf{x}})) \\ &= \arg\max_{\overline{y}' \in \mathcal{Y}} \log\left(\prod_{i=1}^n \frac{1}{1 + \exp(-y_i' \mathbf{w}^\top \mathbf{x}_i)} \right). \end{aligned} \tag{6}$$

In this way, we seek the maximum likelihood estimator of the class label tuple as the mapping result for a data tuple.

To learn the parameter of the linear discriminative function, $\mathbf{w}$, so that the complex loss function $\Delta(\overline{y}, \overline{y}^*)$ can be minimized, we consider the following problems,

- *Encouraging sparsity of* $\mathbf{w}$ We assume that in a feature vector a data point, only a few features are useful, while most of the remaining features are useless. Thus, we need to conduct a feature selection procedure to remove the useless features and keep the useful features, so that we can obtain a parse feature vector. In our method, instead of seeking sparsity of the feature vectors, we seek the sparsity of the parameter vector $\mathbf{w}$. With a sparse $\mathbf{w}$, we can also control the sparsity of the feature effective to the prediction results. To encourage the sparsity of $\mathbf{w}$, we use the ℓ_1 norm of $\mathbf{w}$ to present its sparsity, and minimize the ℓ_1,

$$\begin{aligned} \min_{\mathbf{w}} \Bigg\{ & \frac{1}{2} \|\mathbf{w}\|_1 = \frac{1}{2} \sum_{j=1}^d |w_j| \\ & = \frac{1}{2} \sum_{j=1}^d \frac{w_j^2}{|w_j|} = \frac{1}{2} \mathbf{w}^\top diag\left(\frac{1}{|w_1|}, \ldots, \frac{1}{|w_d|} \right) \mathbf{w} \\ & = \frac{1}{2} \mathbf{w}^\top \Lambda \mathbf{w} \Bigg\}, \end{aligned} \tag{7}$$

  where $diag\left(\frac{1}{|w_1|}, \ldots, \frac{1}{|w_d|}\right) \in \mathbb{R}^{d \times d}$ is a diagonal matrix with its diagonal elements as $\frac{1}{|w_1|}, \ldots, \frac{1}{|w_d|}$, and

$$\Lambda = diag\left(\frac{1}{|w_1|}, \ldots, \frac{1}{|w_d|} \right) \tag{8}$$

  when the ℓ_1 norm of $\mathbf{w}$ is minimized, most elements of $\mathbf{w}$ will shrink to zeros and lead a sparse $\mathbf{w}$.
- *Minimizing complex performance lose* $\Delta(\overline{y}, \overline{y}^*)$ Given the predicted label tuple $\overline{y}^*$, we can measure the prediction performance by comparing it against the true label tuple $\overline{y}$ by using a complex performance measure. To obtain an optimal mapping function, we minimize a

corresponding complex loss of a complex performance measure, $\Delta(\overline{y}, \overline{y}^*)$,

$$\min_{\mathbf{w}} \Delta(\overline{y}, \overline{y}^*) \tag{9}$$

Due to its complexity, we minimize its upper boundary instead of itself. We have the following theorem to define the upper boundary of $\Delta(\overline{y}, \overline{y}^*)$.

Theorem 1 *$\Delta(\overline{y}, \overline{y}^*)$ satisfies*

$$\begin{aligned}\Delta(\overline{y}, \overline{y}^*) \leq & \max_{\overline{y}' \in \mathcal{Y}} \left\{ \log\left(\prod_{i=1}^{n} \frac{1}{1+\exp(-y_i' \mathbf{w}^\top \mathbf{x}_i)}\right) \right. \\ & \left. - \log\left(\prod_{i=1}^{n} \frac{1}{1+\exp(-y_i \mathbf{w}^\top \mathbf{x}_i)}\right) + \Delta(\overline{y}, \overline{y}') \right\} \\ = & \left\{ \log\left(\prod_{i=1}^{n} \frac{1}{1+\exp(-y_i'' \mathbf{w}^\top \mathbf{x}_i)}\right) \right. \\ & \left. - \log\left(\prod_{i=1}^{n} \frac{1}{1+\exp(-y_i \mathbf{w}^\top \mathbf{x}_i)}\right) + \Delta(\overline{y}, \overline{y}'') \right\},\end{aligned} \tag{10}$$

where $\overline{y}' = (y_1', \ldots, y_n')$, *and* $\overline{y}'' = (y_1', \ldots, y_n')$,

$$\begin{aligned}\overline{y}'' = \arg\max_{\overline{y}' \in \mathcal{Y}} & \left\{ \log\left(\prod_{i=1}^{n} \frac{1}{1+\exp(-y_i' \mathbf{w}^\top \mathbf{x}_i)}\right) \right. \\ & \left. - \log\left(\prod_{i=1}^{n} \frac{1}{1+\exp(-y_i \mathbf{w}^\top \mathbf{x}_i)}\right) + \Delta(\overline{y}, \overline{y}') \right\}\end{aligned} \tag{11}$$

The proof of this theorem is found in Appendix section.

After we have the upper bound of the loss function, we minimize it instead of $\Delta(\overline{y}, \overline{y}^*)$ to obtain the mapping function parameter, $\mathbf{w}$,

$$\begin{aligned}\min_{\mathbf{w}} & \left\{ \log\left(\prod_{i=1}^{n} \frac{1}{1+\exp(-y_i'' \mathbf{w}^\top \mathbf{x}_i)}\right) \right. \\ & - \log\left(\prod_{i=1}^{n} \frac{1}{1+\exp(-y_i \mathbf{w}^\top \mathbf{x}_i)}\right) + \Delta(\overline{y}, \overline{y}'') \\ & \left. = \sum_{i=1}^{n} \log\left(\frac{1+\exp(-y_i \mathbf{w}^\top \mathbf{x}_i)}{1+\exp(-y_i'' \mathbf{w}^\top \mathbf{x}_i)}\right) + \Delta(\overline{y}, \overline{y}'') \right\}.\end{aligned} \tag{12}$$

Please note that $\overline{y}''$ is also a function of $\mathbf{w}$.

The overall optimization problem is obtained by combining the problems in (7) and (12),

$$\begin{aligned}\min_{\mathbf{w}} & \left\{ f(\mathbf{w}) = \frac{1}{2} \mathbf{w}^\top \Lambda \mathbf{w} \right. \\ & \left. + C \left[\sum_{i=1}^{n} \log\left(\frac{1+\exp(-y_i \mathbf{w}^\top \mathbf{x}_i)}{1+\exp(-y_i'' \mathbf{w}^\top \mathbf{x}_i)}\right) + \Delta(\overline{y}, \overline{y}'') \right] \right\}\end{aligned} \tag{13}$$

where C is a trade-off parameter. Please note that in this objective, both Λ and $\overline{y}''$ are functions of $\mathbf{w}$. In the first term of the objective, we impose the sparsity of the $\mathbf{w}$, and in the second term, we minimize the upper bound of $\Delta(\overline{y}, \overline{y}^*)$.

2.2 Optimization

To solve the problem of (13), we try to employ the FISTA algorithm with constant step size to minimize the objective $f(\mathbf{w})$. This algorithm is an iterative algorithm, and in each iteration, we first update a search point according to a previous solution of the parameter vector and then update the next parameter vector based on the search point. The basic procedures are summarized as the two following steps:

1. *Search point step* In this step, we assume the previous solution of $\mathbf{w}$ is $\mathbf{w}_{pre}$, and seek a search point $\mathbf{v} \in \mathbb{R}^d$ based on $\mathbf{w}$ is $\mathbf{w}_{pre}$ and a step size L.
2. *Weighting factor step* In this step, we assume we have a weighting factor of previous iteration, τ_{pre}, and we update it to a new weighting factor τ_{cur}.
3. *Solution update step* In this step, we update the new solution of the variable according to the search point. The updated solution is a weighted version of the previous search points, weighted by the weighting factors.

In the follows, we will discuss how to implement these three steps.

2.2.1 Search point step

In this step, when we want to minimize an objective function $f(\mathbf{w})$ with regard to a variable vector $\mathbf{w}$ with a step size L and a previous solution $\mathbf{w}_{pre}$, we seek a search point $\mathbf{u}^*$ as follows,

$$\mathbf{u}^* = \arg\min_{\mathbf{u}} \left\{ \frac{L}{2} \left\| \mathbf{u} - \left(\mathbf{w}_{pre} - \frac{1}{L} \nabla f(\mathbf{w}_{pre}) \right) \right\|_2^2 \right\}, \tag{14}$$

where $\nabla f(\mathbf{w})$ is the gradient function of $f(\mathbf{w})$. Due to the complexity of function $f(\mathbf{w})$, the close form of gradient function $\nabla f(\mathbf{w})$ is difficult to obtain. Thus, instead of seeking gradient function directly, we seek the sub-gradient of this function. At this end, we use the EM algorithm strategy. In each iteration, we first fix $\mathbf{w}$ as $\mathbf{w}_{pre}$ and calculate Λ according to (8), and $y_i''|_{i=1}^n$ according to (11). Then we fix Λ and $y_i''|_{i=1}^n$ and seek the sub-gradient $\nabla f(\mathbf{w})$,

$$\begin{aligned}\nabla f(\mathbf{w}) = \Lambda \mathbf{w} + C \sum_{i=1}^{n} & \left(\frac{y_i'' \mathbf{x}_i \exp(-y_i'' \mathbf{w}^\top \mathbf{x}_i)}{1+\exp(-y_i'' \mathbf{w}^\top \mathbf{x}_i)} \right. \\ & \left. - \frac{y_i \mathbf{x}_i \exp(-y_i \mathbf{w}^\top \mathbf{x}_i)}{1+\exp(-y_i \mathbf{w}^\top \mathbf{x}_i)} \right).\end{aligned} \tag{15}$$

After we have the sub-gradient function $\nabla f(\mathbf{w})$, we substitute it to (14), and we have

$$\begin{aligned}\mathbf{u}^* &= \arg\min_{\mathbf{u}}\left\{\frac{L}{2}\left\|\mathbf{u}-\left(\mathbf{w}_{pre}-\frac{1}{L}\nabla f(\mathbf{w}_{pre})\right)\right\|_2^2\right\}\\ &= \arg\min_{\mathbf{u}}\left\{\frac{L}{2}\left\|\mathbf{u}-\left[\mathbf{w}_{pre}-\frac{1}{L}\left(\Lambda\mathbf{w}_{pre}\right.\right.\right.\right.\\ &\quad +C\sum_{i=1}^{n}\left(\frac{y_i''\mathbf{x}_i\exp(-y_i''\mathbf{w}_{pre}^\top\mathbf{x}_i)}{1+\exp(-y_i''\mathbf{w}_{pre}^\top\mathbf{x}_i)}\right.\\ &\quad \left.\left.\left.\left.-\frac{y_i\mathbf{x}_i\exp(-y_i\mathbf{w}_{pre}^\top\mathbf{x}_i)}{1+\exp(-y_i\mathbf{w}_{pre}^\top\mathbf{x}_i)}\right)\right)\right]\right\|_2^2\right\}\\ &= \arg\min_{\mathbf{u}}\left\{\frac{L}{2}\left\|\mathbf{u}-\left[\left(I-\frac{1}{L}\Lambda\right)\mathbf{w}_{pre}\right.\right.\right.\\ &\quad -\frac{C}{L}\sum_{i=1}^{n}\left(\frac{y_i''\mathbf{x}_i\exp(-y_i''\mathbf{w}_{pre}^\top\mathbf{x}_i)}{1+\exp(-y_i''\mathbf{w}_{pre}^\top\mathbf{x}_i)}\right.\\ &\quad \left.\left.\left.-\frac{y_i\mathbf{x}_i\exp(-y_i\mathbf{w}_{pre}^\top\mathbf{x}_i)}{1+\exp(-y_i\mathbf{w}_{pre}^\top\mathbf{x}_i)}\right)\right]\right\|_2^2=g(\mathbf{u})\right\}. \end{aligned} \tag{16}$$

To solve this problem, we set the gradient function of the objective function $g(\mathbf{u})$ to zero,

$$\begin{aligned}\nabla g(\mathbf{u}) &= L\left\{\mathbf{u}-\left[\left(I-\frac{1}{L}\Lambda\right)\mathbf{w}_{pre}\right.\right.\\ &\quad -\frac{C}{L}\sum_{i=1}^{n}\left(\frac{y_i''\mathbf{x}_i\exp(-y_i''\mathbf{w}_{pre}^\top\mathbf{x}_i)}{1+\exp(-y_i''\mathbf{w}_{pre}^\top\mathbf{x}_i)}\right.\\ &\quad \left.\left.-\frac{y_i\mathbf{x}_i\exp(-y_i\mathbf{w}_{pre}^\top\mathbf{x}_i)}{1+\exp(-y_i\mathbf{w}_{pre}^\top\mathbf{x}_i)}\right)\right]\right\}=0\\ \Rightarrow \mathbf{u}^* &= \left(I-\frac{1}{L}\Lambda\right)\mathbf{w}_{pre}-\frac{C}{L}\sum_{i=1}^{n}\left(\frac{y_i''\mathbf{x}_i\exp(-y_i''\mathbf{w}_{pre}^\top\mathbf{x}_i)}{1+\exp(-y_i''\mathbf{w}_{pre}^\top\mathbf{x}_i)}\right.\\ &\quad \left.-\frac{y_i\mathbf{x}_i\exp(-y_i\mathbf{w}_{pre}^\top\mathbf{x}_i)}{1+\exp(-y_i\mathbf{w}_{pre}^\top\mathbf{x}_i)}\right). \end{aligned} \tag{17}$$

In this way, we obtain the search point $\mathbf{u}^*$.

2.2.2 Weighting factor step

We assume that weighting factor of previous iteration is τ_{pre}, and we can obtain the weighting factor of current iteration, τ_{cur}, as follows,

$$\tau_{cur}=\frac{1+\sqrt{1+4\tau_{pre}{}^2}}{2}. \tag{18}$$

2.2.3 Solution update step

After we have the search point of this current iteration, $\mathbf{u}^*$, the search point of previous iteration, $\mathbf{u}^*{}_{pre}$, and the weighting factor of this iteration and previous iteration, τ_{cur} and τ_{pre}, we can have the following update procedure for the solution of this iteration,

$$\begin{aligned}\mathbf{w}_{cur} &= \mathbf{u}^*+\left(\frac{\tau_{pre}-1}{\tau_{cur}}\right)\left(\mathbf{u}^*-\mathbf{u}^*_{pre}\right)\\ &= \left(\frac{\tau_{cur}+\tau_{pre}-1}{\tau_{cur}}\right)\mathbf{u}^*-\left(\frac{\tau_{pre}-1}{\tau_{cur}}\right)\mathbf{u}^*_{pre}. \end{aligned} \tag{19}$$

In this equation, we can see that the updated solution of $\mathbf{w}_{cur}$ is a weighted version of the current search point, $\mathbf{u}^*$, and the previous search point, $\mathbf{u}^*_{pre}$.

2.3 Iterative algorithm

With the optimization in the previous section, we summarize the iterative algorithm to optimize the problem in (13). The iterative algorithm is given in Algorithm 1.

Algorithm 1: FISTA with constant stepsize to optimize (13)

1. **Input**: L, a constant step-size;
2. **Step 0**: Take $\mathbf{w}_1=\mathbf{u}_0$, $\tau_1=1$.
3. **Step** k ($k\geq 1$):
   (a) Update Λ_k according to (8) by fixing $\mathbf{w}=\mathbf{w}_k$.
   (b) Update $y''_{i\,k}|_{i=1}^n$ according to (11) by fixing $\mathbf{w}=\mathbf{w}_k$.
   (c) Update $\mathbf{u}_k$ according to (17) by fixing $\mathbf{w}_{pre}=\mathbf{w}_k$, $\Lambda=\Lambda_k$, and $y_i''=y''_{i\,k}, i=1,\cdots,n$.
   (d) Updating τ_k according to (18) by fixing $\tau_{k-1}=\tau_{pre}$.
   (e) Updating $\mathbf{w}_k$ according to (19) by fixing $\mathbf{u}^*=\mathbf{u}_k$, $\mathbf{u}^*_{pre}=\mathbf{u}_{k-1}$, $\tau_{cur}=\tau_k$, and $\tau_{pre}=\tau_{k-1}$.
4. **Output**: $\mathbf{w}_k$

In this algorithm, we can see that in each iteration, we first update Λ and $y_i''|_{i=1}^n$ and then use them to update the search point. With the search point and an updated weighting factor, we update the mapping function parameter vector, $\mathbf{w}$. This algorithm is called learning of sparse maximum likelihood model (SMLM).

2.4 Scaling up to big data based on Hadoop

In this section, we discuss how to fit the proposed algorithm to big data set. We assume that the number of the training data points, n, is extremely large. One single machine is not able to store the entire data set, and the data set is split into m subsets and stored in m different clusters. The clusters are managed by a big data platform, Hadoop [4, 10, 25, 35]. Hadoop is a software of distributed data management and processing. Given a large data set, it splits it into subsets and stores them in different clusters. To process the data and obtain a final output, it uses a MapReduce framework [3, 6, 15, 24]. This framework requires a Map program and a Reduce program from the users. The Hadoop software delivers the Map program to each cluster and uses it to process the subset to produce

some median results and then uses the Reduce program to combine the median results to produce the final outputs. Using the MapReduce framework, by defining our own Map and Reduce functions, we can implement the critical steps in Algorithm 1. For example, in the sub-step (c) of step k, we need to calculate $\mathbf{u}_k$ from (17). In this step, the most time-consuming step is to calculate the summation of a function over all the data points,

$$output = \sum_{i=1}^{n}\left(\frac{y_i''\mathbf{x}_i \exp(-y_i''\mathbf{w}_{pre}^{\top}\mathbf{x}_i)}{1+\exp(-y_i''\mathbf{w}_{pre}^{\top}\mathbf{x}_i)} - \frac{y_i\mathbf{x}_i \exp(-y_i\mathbf{w}_{pre}^{\top}\mathbf{x}_i)}{1+\exp(-y_i\mathbf{w}_{pre}^{\top}\mathbf{x}_i)}\right) = \sum_{i=1}^{n} function(\mathbf{x}_i, y_i, y_i''), \quad (20)$$

where $function(\mathbf{x}_i, y_i, y_i'') = \frac{y_i''\mathbf{x}_i \exp(-y_i''\mathbf{w}_{pre}^{\top}\mathbf{x}_i)}{1+\exp(-y_i''\mathbf{w}_{pre}^{\top}\mathbf{x}_i)} - \frac{y_i\mathbf{x}_i \exp(-y_i\mathbf{w}_{pre}^{\top}\mathbf{x}_i)}{1+\exp(-y_i\mathbf{w}_{pre}^{\top}\mathbf{x}_i)}$ is the function applied to each data point. Since the entire data set is split into m subsets, $\mathcal{X}_m|_{j=1}^{m}$, we can design a Map function to calculate the summation over each subset and then design a Reduce function to combine them to obtain the final output. The Map and Reduce functions are as follows.

Map function applied to the j-th subset

1. **Input**: Data points of the jth subset, $\{(\mathbf{x}_i, y_i, y_i'')\}|_{i:\mathbf{x}_i\in\mathcal{X}_j}$.
2. **Input**: Previous parameter, $\mathbf{w}_{pre}$.
3. **Initialize**: $Output_j = 0$.
4. **For** $i : \mathbf{x}_i \in \mathcal{X}_j$
   (a) $Output_j = Output_j + function(\mathbf{x}_i, y_i, y_i'')$;
5. **Endfor**
6. **Output**: $Output_j$

Reduce function to calculate the final output

1. **Input**: Median outputs of m Map functions, $Output_j|_{j=1}^{m}$.
2. **Initialize**: $Output = 0$.
3. **For** $j = 1, \ldots, m$
   (a) $Output = Output + Output_j$;
4. **Endfor**
5. **Output**: $Output$

3 Experiment

In this section, we evaluate the proposed SMLM for the optimization of complex loss function. Three different applications are considered, which are aircraft event recognition, intrusion detection in wireless mesh networks, and image classification.

3.1 Aircraft event recognition

Recognizing aircraft event of aircraft landing is an important problem in the area of civil aviation safety research. This procedure provides important information for fault diagnosis and structure maintenance of aircraft [32]. Given a landing condition, we want to predict whether it is normal and abnormal. To this end, we extract some features and use them to predict the aircraft event of normal or abnormal. In this experiment, we evaluate the proposed algorithm in this application and use it as a model for the prediction of aircraft event recognition.

3.1.1 Data set

In this experiment, we collect a data set of 160 data points. Each data point is a landing condition, and we describe the landing condition by five features, including vertical acceleration, vertical speed, lateral acceleration, roll angle, and pitch rate. The data points are classified into two classes, normal class and abnormal. The normal class is treated as positive class, while the abnormal class is treated as negative class. The number of positive data points is 108, and the number of negative data points is 52.

3.1.2 Experiment setup

In this experiment, we use the tenfold cross-validation. The data set is split into tenfolds randomly, and each fold contains 16 data points. Each fold is used as a test set in turn, and the remaining tenfolds are combined and used as training set. The proposed model is training over the training set and then used to predict the class labels of the testing data points in the test set. The prediction results are evaluated by a performance measurement. This performance measurement is used to compare the true class labels of the test data points against the predicted class labels. In the training procedure, a complex loss function corresponding to the performance measurement is minimized.

In our experiments, we consider three performance measurements, which are F-score, area under receiver operating characteristic curve (AUROC), and precision–recall curve break-even point (PRBEP). To define these performance measures, we first need to define the following items,

- true positive (TP), the number of correctly predicted positive data points,
- true negative (TN), the number of correctly predicted negative data points,
- false positive (FP), the number of negative data points wrongly predicted to positive data points, and

- false negative (FN), the number of positive data points wrongly predicted to negative data points.

With these measures, we can define F-score as follows,

$$F = \frac{2 \times TP}{2 \times TP + FP + FN}. \quad (21)$$

Moreover, we can also define true positive rate (TPR) and the false positive rate (FPR) as follows,

$$TPR = \frac{TP}{TP + FN}, \quad FPR = \frac{FP}{FP + TN}. \quad (22)$$

With different thresholds, we can have different pair of TPR and FPR. By plotting TPR against FPR values, we can have a curve of receiver operating characteristic (ROC). The area under this curve is obtained as AUROC. The recall and precision are defined as follows,

$$recall = \frac{TP}{TP + FN}, \quad precision = \frac{TP}{TP + FP}. \quad (23)$$

With different thresholds, we can also have different pair of recall and precision values. We can obtain a recall–precision (RP) curve, by plotting different precision values against recall values. PRBEP is the value of the point of the RP curve where recall and precision are equal to each other.

3.1.3 Experiment result

We compare the proposed algorithm, SMLM, against several state-of-the-art complex loss optimization methods, including support vector machine for multivariate performance optimization (SVM_{multi}) [9], classifier adaptation for multivariate performance optimization (CAPO) [12], and features selection for multivariate performance optimization (FS_{multi}) [16]. The boxplots of the optimized F-scores of tenfold cross-validation of different algorithms on the aircraft event recognition problem are given in Fig. 1, these of optimized AUROC are given in Fig. 2, and these of the optimized PRBEP are given in Fig. 3. From these figures, we can see that the proposed method, SMLM, outperforms the compared algorithms on three different optimized performances. For example, in Fig. 3, we can see that the boxplot of PRBEP of SMLM is significantly higher than that of other methods, the median value is almost 0.6, while that of other methods is much lower than 0.6. In Fig. 2, we can also have similar observation, and the overall AUROC values optimized by SMLM are much higher than those of other methods. A reason for this outperforming is that our method seeks the maximum likelihood and sparsity of the model simultaneously.

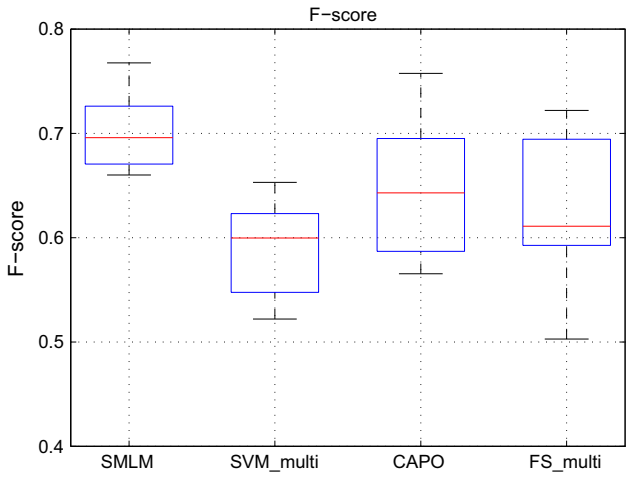

Fig. 1 Boxplots of F-score of compared method on aircraft event recognition problem

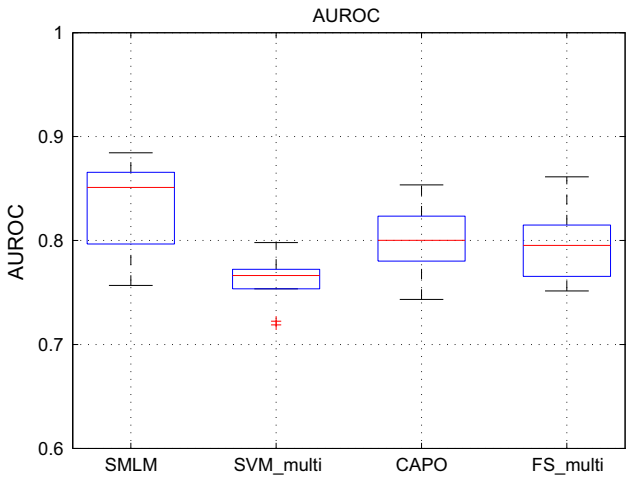

Fig. 2 Boxplots of AUROC of compared method on aircraft event recognition problem

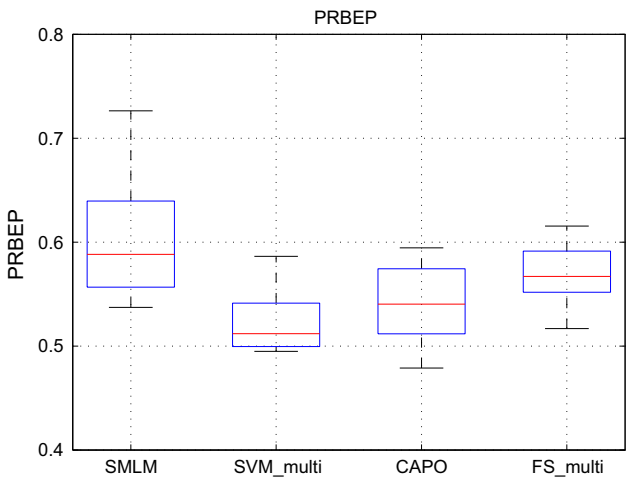

Fig. 3 Boxplots of PRBEP of compared method on aircraft event recognition problem

3.2 Intrusion detection in wireless mesh networks

Wireless mesh network (WMN) is a new generation technology of wireless networks, and it has been used in many different applications. However, due to its openness in wireless communication, it is vulnerable to intrusions, and thus, it is extremely important to detect intrusion in WMN. Given an attack record, the problem of intrusion detection is to classify it to one of the following classes, denial service attacks, detect attacks, obtain root privileges and remote attack unauthorized access attacks. In this paper, we use the proposed method, SMLM, for the problem intrusion detection,

3.2.1 Data set

In this experiment, we use the KDD CPU1999 data set. This data set contains 40,000 attack records, and for each class, there are 10,000 records. For each record, we first preprocess the record and then convert the features into digital signature as the new features.

3.2.2 Experiment setup

In this experiment, we also use the tenfold cross-validation, and we also use the *F*-score, AUROC, and PRBEP performance measures.

3.2.3 Experiment result

The boxplots of the optimized *F*-scores of tenfold cross-validation are given in Fig. 4, the boxplots of AUROC are given in Fig. 5, and the boxplots of PRBEP are given in Fig. 6. Similar to the results on aircraft event recognition problem, the outperforming of the proposed algorithm, SMLM, over other methods is also significant. This is a strong evidence of the advantages of sparse learning and maximum likelihood.

3.3 ImageNet image classification

In the third experiment, we use a large image set to test the performance of the proposed algorithm with big data.

3.3.1 Data set

In this experiment, we use a large data set, ImageNet [11]. This data set contains over 15 million images, and the images belong to 22,000 classes. These images are from Web pages and are labeled by people manually. The entire

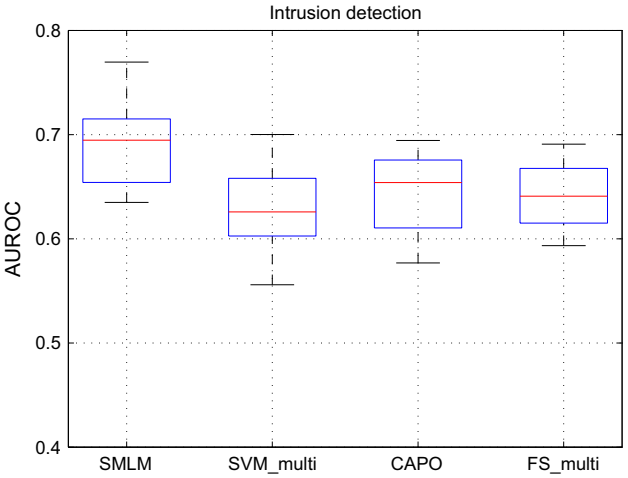

Fig. 5 Boxplots of AUROC of compared method on aircraft event recognition problem

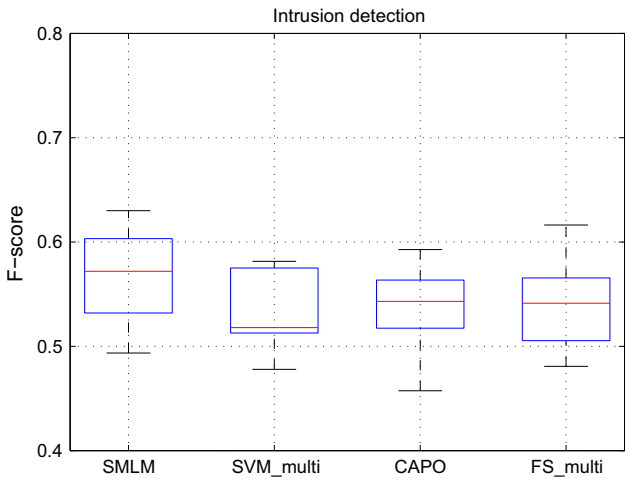

Fig. 4 Boxplots of *F*-score of compared method on intrusion detection problem

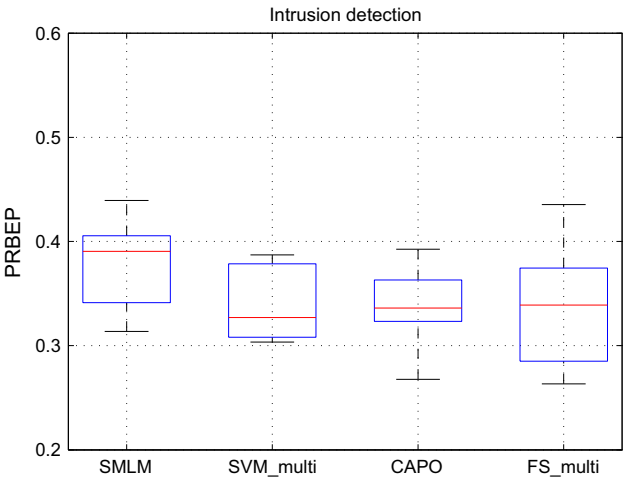

Fig. 6 Boxplots of PRBEP of compared method on aircraft event recognition problem

data set is split into three subsets, which are one training set, one validation set, and one testing set. The training set contains 1.2 million images, the validation set contains 50,000 images, and the testing set contains 150,000 images. To represent each image, we use the bag-of-features method. Local SIFT features are extracted from each image and quantized to a histogram. The features can be downloaded directly from http://image-net.org/download-features.

3.3.2 Experiment setup

In this experiment, we do not use the tenfold cross-validation, but use the given training/validation/testing set splitting. We first perform the proposed algorithm to the training set to learn the classifier, then use the validation set to justify the optimal trade-off parameters, and finally test the classifier over the testing set. The performances of *F*-score, AUROC, and PRBEP are considered in this experiment. To handle the multi-classification problem, we have a binary classification problem for each class, and in this problem, the considered class is a positive class, while the combination of all other classes is a negative class.

3.3.3 Experiment results

The boxplots of the optimized *F*-score, AUROC, and PRBEP of different classes are given in Figs. 7, 8, and 9. From these figures, we clearly see that the proposed algorithm outperforms the competing methods. This is another strong evidence of the effectiveness of the SMLM algorithm. Moreover, it also shows that the proposed algorithm also works well over the big data.

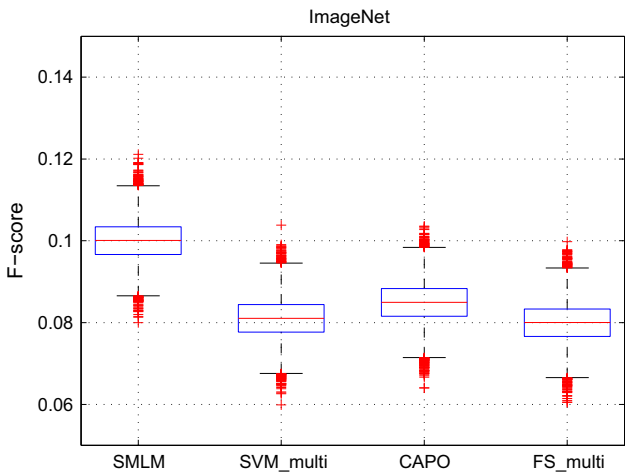

Fig. 7 Boxplots of *F*-score of compared method on ImageNet image classification problem

3.4 Running time

The running time of the proposed algorithm on the three used data sets is given in Fig. 10. It can be observed from this figure that the first two experiments do not consume much time, while the third large-scale data set-based experiment costs a lot of time. This is natural, because in each iteration of the algorithm, we have a function for each data point, and a summation over the responses of this function.

4 Conclusion

In this paper, we investigate the problem of optimization of complex corresponding to a complex multivariate performance measure. We propose a novel predicative model to solve this problem. This model is based on the maximum

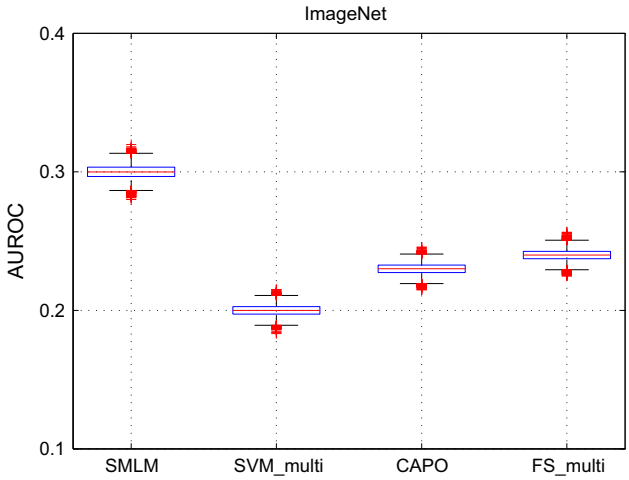

Fig. 8 Boxplots of AUROC of compared method on ImageNet image classification problem

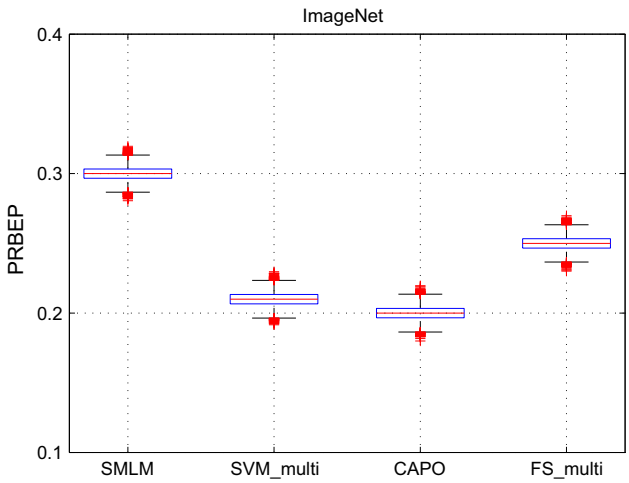

Fig. 9 Boxplots of PRBEP of compared method on ImageNet image classification problem

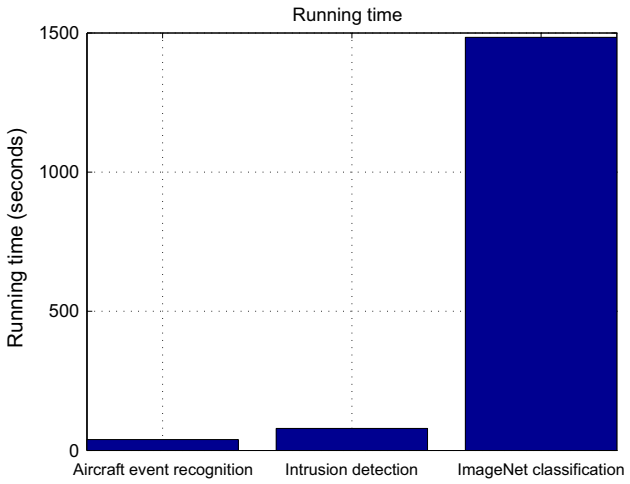

Fig. 10 Running time of SMLM algorithm on three experiments

likelihood of a class label tuple given an input data tuple. To solve the model parameter, we propose an optimization problem based on the approximation of the upper bound of the loss function and the sparsity of the model. Moreover, an iterative algorithm is developed to solve it. Experiments on two real-world applications show its advantages over state of the art.

Appendix

Proof of Theorem 1 According to (6), we have

$$
\begin{aligned}
&\bar{y}^* = \arg\max_{\bar{y}\in\mathcal{Y}} \log\left(\prod_{i=1}^{n}\frac{1}{1+\exp(-y_i'\mathbf{w}^\top\mathbf{x}_i)}\right),\\
&\Rightarrow \log\left(\prod_{i=1}^{n}\frac{1}{1+\exp(-y_i^*\mathbf{w}^\top\mathbf{x}_i)}\right) \geq \log\left(\prod_{i=1}^{n}\frac{1}{1+\exp(-y_i'\mathbf{w}^\top\mathbf{x}_i)}\right), \forall\bar{y}'\in\mathcal{Y},\\
&\Rightarrow \log\left(\prod_{i=1}^{n}\frac{1}{1+\exp(-y_i^*\mathbf{w}^\top\mathbf{x}_i)}\right) \geq \log\left(\prod_{i=1}^{n}\frac{1}{1+\exp(-y_i\mathbf{w}^\top\mathbf{x}_i)}\right),\\
&\Rightarrow \log\left(\prod_{i=1}^{n}\frac{1}{1+\exp(-y_i^*\mathbf{w}^\top\mathbf{x}_i)}\right) - \log\left(\prod_{i=1}^{n}\frac{1}{1+\exp(-y_i\mathbf{w}^\top\mathbf{x}_i)}\right) \geq 0,\\
&\Rightarrow \log\left(\prod_{i=1}^{n}\frac{1}{1+\exp(-y_i^*\mathbf{w}^\top\mathbf{x}_i)}\right) - \log\left(\prod_{i=1}^{n}\frac{1}{1+\exp(-y_i\mathbf{w}^\top\mathbf{x}_i)}\right)\\
&\quad + \Delta(\bar{y},\bar{y}^*) \geq \Delta(\bar{y},\bar{y}^*),\\
&\Rightarrow \max_{\bar{y}'\in\mathcal{Y}}\left\{\log\left(\prod_{i=1}^{n}\frac{1}{1+\exp(-y_i'\mathbf{w}^\top\mathbf{x}_i)}\right) - \log\left(\prod_{i=1}^{n}\frac{1}{1+\exp(-y_i\mathbf{w}^\top\mathbf{x}_i)}\right)\right.\\
&\quad \left.+\Delta(\bar{y},\bar{y}')\right\} \geq \Delta(\bar{y},\bar{y}^*),
\end{aligned}
\tag{24}
$$

and thus we have (10).

References

1. Barbosa R, Batista B, Bario C, Varrique R, Coelho V, Campiglia A, Barbosa F (2015) A simple and practical control of the authenticity of organic sugarcane samples based on the use of machine-learning algorithms and trace elements determination by inductively coupled plasma mass spectrometry. Food Chem 184:154–159
2. Bhattacharya B, Hughes G (2015) On shape properties of the receiver operating characteristic curve. Stat Probab Lett 103:73–79. doi:10.1016/j.spl.2015.04.003
3. Csar T, Pichler R, Sallinger E, Savenkov V (2015) Using statistics for computing joins with map reduce. CEUR Workshop Proc 1378:69–74
4. Feller E, Ramakrishnan L, Morin C (2015) Performance and energy efficiency of big data applications in cloud environments: a hadoop case study. J Parallel Distrib Comput 79–80:80–89
5. He Y, Sang N (2013) Multi-ring local binary patterns for rotation invariant texture classification. Neural Comput Appl 22(3–4): 793–802
6. Irudayasamy A, Arockiam L (2015) Scalable multidimensional anonymization algorithm over big data using map reduce on public cloud. J Theor Appl Inf Technol 74(2):221–231
7. Jaime-Prez J, Garca-Arellano G, Mndez-Ramrez N, Gonzlez-Llano T, Gmez-Almaguer D (2015) Evaluation of hemoglobin performance in the assessment of iron stores in feto-maternal pairs in a high-risk population: Receiver operating characteristic curve analysis. Rev Bras Hematol Hemoter 37(3):178–183
8. Joachims T (2005) A support vector method for multivariate performance measures. In: ICML 2005—proceedings of the 22nd international conference on machine learning, pp 377–384
9. Joachims T (2005) A support vector method for multivariate performance measures. In: Proceedings of the 22nd international conference on machine learning, pp 377–384. ACM
10. Kim M, Lee Y, Park HH, Hahn S, Lee CG (2015) Computational fluid dynamics simulation based on hadoop ecosystem and heterogeneous computing. Comput Fluids 115:1–10
11. Krizhevsky A, Sutskever I, Hinton G (2012) Imagenet classification with deep convolutional neural networks. Adv Neural Inf Process Syst 2:1097–1105
12. Li N, Tsang IW, Zhou ZH (2013) Efficient optimization of performance measures by classifier adaptation. IEEE Trans Pattern Anal Mach Intell 35(6):1370–1382
13. Liu C, Yang S, Deng L (2015) Determination of internal qualities of newhall navel oranges based on nir spectroscopy using machine learning. J Food Eng 161:16–23
14. Liu X, Wang J, Yin M, Edwards B, Xu P (2015) Supervised learning of sparse context reconstruction coefficients for data representation and classification. Neural Comput Appl. doi:10.1007/s00521-015-2042-5
15. Maitrey S, Jha C, Jha C (2015) Handling big data efficiently by using map reduce technique. In: Proceedings—2015 IEEE international conference on computational intelligence and communication technology, CICT 2015, pp 703–708
16. Mao Q, Tsang IWH (2013) A feature selection method for multivariate performance measures. IEEE Trans Pattern Anal Mach Intell 35(9):2051–2063
17. Neoh S, Zhang L, Mistry K, Hossain M, Lim C, Aslam N, Kinghorn P (2015) Intelligent facial emotion recognition using a layered encoding cascade optimization model. Appl Soft Comput J 34:72–93
18. Ozenne B, Subtil F, Maucort-Boulch D (2015) The precision–recall curve overcame the optimism of the receiver operating characteristic curve in rare diseases. J Clin Epidemiol 68(8):855–859
19. Ryu J, Hong S, Yang H (2015) Sorted consecutive local binary pattern for texture classification. IEEE Trans Image Process 24(7):2254–2265
20. Saito T, Rehmsmeier M (2015) The precision–recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. PLoS One 10(3):e0118432

21. Sanchez-Valdes D, Alvarez-Alvarez A, Trivino G (2015) Walking pattern classification using a granular linguistic analysis. Appl Soft Comput J 33:100–113
22. Schlattmann P, Verba M, Dewey M, Walther M (2015) Mixture models in diagnostic meta-analyses—clustering summary receiver operating characteristic curves accounted for heterogeneity and correlation. J Clin Epidemiol 68(1):61–72
23. Sedgwick P (2015) How to read a receiver operating characteristic curve. BMJ. doi:10.1136/bmj.h2464 (**Online**)
24. Shanoda M, Senbel S, Khafagy M (2015) Jomr: multi-join optimizer technique to enhance map-reduce job. In: 2014 9th international conference on informatics and systems, INFOS 2014, pp PDC80–PDC87
25. Shi X, Chen M, He L, Xie X, Lu L, Jin H, Chen Y, Wu S (2015) Mammoth: gearing hadoop towards memory-intensive mapreduce applications. IEEE Trans Parallel Distrib Syst 26(8):2300–2315
26. Tian Y, Zhang Q, Liu D (2014) υ-nonparallel support vector machine for pattern classification. Neural Comput Appl 25(5):1007–1020
27. Villmann T, Kaden M, Lange M, Sturmer P, Hermann W (2015) Precision–recall-optimization in learning vector quantization classifiers for improved medical classification systems. In: Proceedings 2014 IEEE symposium on computational intelligence and data mining (CIDM), pp 71–77. doi:10.1109/CIDM.2014.7008150
28. Wang H, Wang J (2014) An effective image representation method using kernel classification. In: 2014 IEEE 26th international conference on tools with artificial intelligence (ICTAI 2014), pp 853–858
29. Wang J, Wang H, Zhou Y, McDonald N (2015) Multiple kernel multivariate performance learning using cutting plane algorithm. In: Systems, Man and Cybernetics (SMC), 2015 IEEE International Conference on. IEEE
30. Wang J, Zhou Y, Wang H, Yang X, Yang F, Peterson A (2015) Image tag completion by local learning. In: Advances in Neural Networks–ISNN 2015. Springer
31. Wang J, Zhou Y, Yin M, Chen S, Edwards B (2015) Representing data by sparse combination of contextual data points for classification. In: Advances in Neural Networks–ISNN 2015. Springer
32. Wang X, Shu P (2014) Incremental support vector machine learning method for aircraft event recognition. In: Proceedings—2nd international conference on enterprise systems, ES 2014, pp 201–204
33. Wen Z, Zhang R, Ramamohanarao K (2014) Enabling precision/recall preferences for semi-supervised svm training. In: Proceedings of the 23rd ACM international conference on information and knowledge management, pp 421–430
34. Xu Z, Qi Z, Zhang J (2014) Learning with positive and unlabeled examples using biased twin support vector machine. Neural Comput Appl 25(6):1303–1311
35. Yin J, Liao Y, Baldi M, Gao L, Nucci A (2015) Gom-hadoop: a distributed framework for efficient analytics on ordered datasets. J Parallel Distrib Comput 83:58–69
36. Zhang X, Saha A, Vishwanathan S (2012) Smoothing multivariate performance measures. J Mach Learn Res 13(1):3623–3680
37. Zhao J, Zhou Z, Cao F (2014) Human face recognition based on ensemble of polyharmonic extreme learning machine. Neural Comput Appl 24(6):1317–1326