Published in final edited form as:
Can J Stat. 2011 June 1; 39(2): 300–323. doi:10.1002/cjs.10105.

Robust penalized logistic regression with truncated loss functions

Seo Young Park[1] and **Yufeng Liu**[2,*]

[1]Department of Health Studies, Chicago, IL 60615, USA

[2]Department of Statistics and Operations Research, Carolina Center for Genome Sciences, Chapel Hill, NC 27599, USA

Abstract

The penalized logistic regression (PLR) is a powerful statistical tool for classification. It has been commonly used in many practical problems. Despite its success, since the loss function of the PLR is unbounded, resulting classifiers can be sensitive to outliers. To build more robust classifiers, we propose the robust PLR (RPLR) which uses truncated logistic loss functions, and suggest three schemes to estimate conditional class probabilities. Connections of the RPLR with some other existing work on robust logistic regression have been discussed. Our theoretical results indicate that the RPLR is Fisher consistent and more robust to outliers. Moreover, we develop estimated generalized approximate cross validation (EGACV) for the tuning parameter selection. Through numerical examples, we demonstrate that truncating the loss function indeed yields better performance in terms of classification accuracy and class probability estimation.

Key words and phrases

Classification; logistic regression; probability estimation; robustness; truncation

1. INTRODUCTION

The penalized logistic regression (PLR) is a commonly used classification method in practice. It is a generalization of the standard logistic regression with a penalty term on the coefficients. It is now known that the PLR can be fit in the regularization framework with *loss* + *penalty* (Wahba, 1999; Lin et al., 2000). The loss function controls goodness of fit of the model, and the penalization term helps avoid overfitting so that good generalization can be obtained.

The PLR uses the unbounded logistic loss. As a result, the resulting classifier can be sensitive to outliers. In this article, we propose the robust penalized logistic regression (RPLR), which uses truncated logistic loss function. Because truncation reduces the impact of misclassified outliers, the RPLR is more robust and accurate than the standard PLR. Connections of the proposed RPLR with some other existing robust logistic regression methods are also discussed.

One important aspect of classification is class probability estimation. Good class probability estimation can reflect the strength of classification. Thus, it is desirable in many applications. In the PLR, one can use the estimated classification function, that is, the

*Author to whom correspondence may be addressed. yfliu@email.unc.edu.

estimated logit function, to derive the corresponding probability estimation. When we replace the logistic loss by its truncated version, properties of the corresponding classification function may not preserve all class probability information any more. To solve this problem, we propose three different schemes for class probability estimation. Properties and performance of these three schemes are explored as well.

Although the original logistic loss function is convex, its truncated version becomes non-convex. Consequently, the corresponding minimization problem involves difficult non-convex optimization. To implement the RPLR, we decompose the non-convex truncated logistic loss function into the difference of two convex functions. Then, using this decomposition, we apply the difference convex (d.c.) algorithm to obtain the solution of the RPLR through iterative convex minimization.

The tuning parameter plays an important role in the RPLR implementation. To select an efficient tuning parameter, we develop the estimated generalized approximate cross validation (EGACV) procedure and compare its performance with the cross validation method.

In the following sections, we describe the new proposed method in more details with theoretical justification and numerical examples. Section 2 reviews the PLR and gives a maximum likelihood interpretation. In Section 3 we review some related robust logistic regression methods in the literature. In Section 4 we describe the RPLR and explore its theoretical properties. The methods for class probability estimation are also introduced. Section 5 develops the d.c. algorithm to solve the non-convex minimization problem for the RPLR. In Section 6 we discuss the issue of the tuning parameter selection. Numerical results are presented in Section 7 and Section 8 provides some discussion. The proofs of theorems and the detailed derivation of the tuning procedure are included in the Appendix Section.

2. PENALIZED LOGISTIC REGRESSION

In binary classification, we want to build a classifier based on a training sample $\{(\boldsymbol{x}_i, y_i)|i = 1, 2, \ldots, n\}$, where $\boldsymbol{x}_i \in \mathbf{R}^d$ is a vector of predictors, and $y_i \in \{+1, -1\}$ is its class label. Typically it is assumed that the training data are distributed according to an unknown probability distribution $P(\boldsymbol{x}, y)$. The goal is to find a classifier which minimizes the misclassification rate. Moreover, besides good classification accuracy, it is also desirable to estimate the class conditional probability.

For discussion, we first briefly review the PLR and its likelihood interpretation. In the standard logistic regression model for binary classification, one assumes that the logit can be modeled as a linear function in covariates. Specifically, the model can be written as follows:

$$\log\frac{Pr(Y=+1|\boldsymbol{X})}{Pr(Y=-1|\boldsymbol{X})}=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{X}+b \tag{1}$$

where $\boldsymbol{X}$ and Y denote the vector of explanatory variables and the class label, respectively. The coefficients of logistic regression $(\boldsymbol{w}, b)$ can be estimated by the method of maximum likelihood (McCullagh & Nelder, 1989). As one way of smoothing, le Cessie & van Houwelingen (1992) proposed PLR, which maximizes the log-likelihood subject to a constraint on the L_2 norm of the coefficients. Wahba (1999) showed the linear PLR is equivalent to finding b and $\boldsymbol{w}$ which solves

$$\min_{f\in\mathcal{F}}\sum_{i=1}^{n} l(y_i f(\boldsymbol{x}_i))+\lambda J(f) \tag{2}$$

where $\mathcal{F} = \{f : f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b\}$, $l(u) = \log(1 + e^{-u})$, $J(f)=1/2\|\boldsymbol{w}\|_2^2$, and $\lambda > 0$ is a tuning parameter. Once the classification function f is obtained, one can use sign($f(\boldsymbol{x})$) to estimate the label of $\boldsymbol{x}$, that is, $\hat{y} = +1$ if $f(\boldsymbol{x}) \geq 0$, and $\hat{y} = -1$ otherwise.

For a nonlinear problem, theory of reproducing kernel Hilbert spaces can be applied and then the kernel PLR has $\mathcal{F} = \{f : f(\boldsymbol{x}) = r(\boldsymbol{x}) + b, r(\boldsymbol{x}) \in \mathcal{H}_K\}$ and $J(f) = \| r \|_{\mathcal{H}_K}$ where $r(\boldsymbol{x})=\sum_{i=1}^{n} \upsilon_i K(\boldsymbol{x}_i, \boldsymbol{x})$ and K is the kernel function (Wahba, 1999). Properties of the reproducing kernel and the representer theorem imply that $\|r\|_{\mathcal{H}_K}^2 = \boldsymbol{\upsilon}^{\mathrm{T}}\boldsymbol{K}\boldsymbol{\upsilon}$ where $\boldsymbol{\upsilon} = (\upsilon_1, \ldots, \upsilon_n)^{\mathrm{T}}$ and $\boldsymbol{K}$ is an $n \times n$ positive definite matrix with its $i_1 i_2$ th element $K(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2})$ (Kimeldorf & Wahba, 1971).

Notice that the loss function $l(u)$ in (2) is a decreasing function as shown in the left panel of Figure 1 and in particular, its value grows rapidly as u goes to negative infinity. This causes high impact of outliers with very small (negative) value of $y_i f(\boldsymbol{x}_i)$. As a result, the coefficient estimates of the PLR can be affected by outliers far from their own classes. To further illustrate the effect of outliers on the PLR, we randomly generate two-dimensional separable data and apply the PLR to obtain a classification boundary. As shown in the left panel of Figure 2, the PLR works very well without outliers. However, if we randomly select one of the observations and move it away from its own class, then the classification boundary of the PLR is pulled towards to that outlier, as shown in the right panel of Figure 2. As a result, the corresponding misclassification rate will become higher. In contrast, our new proposed method is much more robust to the outlier so that its classification boundary is more accurate.

The effect of outliers on the PLR can also be interpreted using maximum likelihood. The likelihood function of unpenalized logistic regression can be written as

$$L(b, \boldsymbol{w})=\prod_{i=1}^{n} P(\boldsymbol{x}_i)^{(1+y_i)/2}(1 - P(\boldsymbol{x}_i))^{(1-y_i)/2} \tag{3}$$

where $P(\boldsymbol{x}) = Pr(y = +1|\boldsymbol{x})$. Then, we can plug in the logit function (1) into (3), and the corresponding maximizer of $L(b, \boldsymbol{w})$ is the solution of the logistic regression. Note that the ith term of the product in the likelihood is $P(\boldsymbol{x}_i)$ when $y_i = +1$, and $1 - P(\boldsymbol{x}_i)$ otherwise. Therefore, to maximize the likelihood, one needs to find $(\boldsymbol{w}, b)$ to make $P(\boldsymbol{x}_i)$ big when $y_i = 1$ and small when $y_i = -1$. However, this could be sensitive to outliers. To illustrate this further, assume there is one data point x_i with $y_i = +1$ which locates far from the other data points of class +1 but closer to data of class −1 as illustrated in the right panel of Figure 2. Using the solution $(\boldsymbol{w}, b)$ without the outlier, the corresponding $P(\boldsymbol{x}_i)$ for the outlier will be very small because $\boldsymbol{x}_i$ is closer to the data of class −1. Consequently, the ML method would select $(\boldsymbol{w}, b)$ which will make $P(\boldsymbol{x}_i)$ bigger to obtain larger likelihood at the expense of other entries' classification accuracy. This results in the boundary moving towards to the outlier. In the next section, we discuss some literature on robust logistic regression.

3. LITERATURE ON ROBUST LOGISTIC REGRESSION

There is a large literature on the robustness issue of the logistic regression. Most of the existing methods attempt to achieve robustness by downweighting observations which are far from the majority of the data, that is, outliers (Krasker & Welsch, 1982; Pregibon, 1982; Stefanski, Carroll & Ruppert, 1986; Copas, 1988; Künsch, Stefansk & Carroll, 1989; Morgenthaler, 1992; Carroll and Pederson, 1993; Bianco and Yohai, 1996; Bondell, 2005). Stefanski, Carroll & Ruppert (1986) and Künsch, Stefansk & Carroll (1989) modified original score function of the logistic regression to obtain bounded sensitivity, which is a concept introduced by Krasker & Welsch (1982). Morgenthaler (1992) used L_1-norm instead of L_2-norm in the likelihood, resulting in a weighted score function of the original score function. Cantoni & Ronchetti (2001b) focused on robustness of inference rather than the model.

Pregibon (1982) suggested resistant fitting methods which taper the standard likelihood to reduce the influence of extreme observations. In particular, he proposed to estimate $(\boldsymbol{w}, b)$ by solving

$$\min_{f\in\mathcal{F}} \sum_{i=1}^{n} h(\boldsymbol{x}_i)\rho\left(\frac{d_i}{h(\boldsymbol{x}_i)}\right) \tag{4}$$

where $\rho(u)$ is a tapering function, $h(\boldsymbol{x})$ is a factor which controls leverage of each observation, and d_i is negative log-likelihood, that is, $d_i = -[((1 + Y_i)/2) \log P(\boldsymbol{x}_i)+((1 + Y_i)/2) \log(1 - P(\boldsymbol{x}_i))]$. Note that this reduces to standard maximum likelihood estimation of the logistic regression when $h(\boldsymbol{x}) \equiv 1$ and $\rho(u) = u$. The particular tapering function Pregibon (1982) proposed to use is the Huber's loss function

$$\rho(u)=\begin{cases} u & \text{if } u \le H \\ 2(uH)^{1/2} - H & \text{otherwise} \end{cases} \tag{5}$$

where H is a prespecified constant. In order to compare with our new method, we provide a new view of the method by Pregibon (1982) in the loss function framework. In particular, with ρ in (5) and $h(\boldsymbol{x}) \equiv 1$, we can reduce (4) to

$$\min_{f\in\mathcal{F}} \sum_{i=1}^{n} l^{\text{Pregibon}}(y_i f(\boldsymbol{x}_i)) \tag{6}$$

where

$$l^{\text{Pregibon}}(u)=\rho(l(u))=\begin{cases} \log(1+e^{-u}) & \text{if } u \ge -\log(e^{H} - 1) \\ 2(H\ \log(1+e^{-u}))^{1/2} - H & \text{otherwise} \end{cases} \tag{7}$$

The estimate in (6) was shown to have approximately 95% asymptotic relative efficiency when $H = 1.345^2$. The loss function in (7) with $H = 1.345^2$ is plotted in the right panel of Figure 1 for comparison. As shown in the plot, $l^{\text{Pregibon}}(u)$ grows as u goes to negative infinity, but less rapidly than the loss function of the original logistic regression $l(u)$. Consequently, the resulting coefficient estimates become less sensitive to extreme observations. However, the value of $l^{\text{Pregibon}}(u)$ remains to be unbounded, hence the impact of outliers can still be large.

Bianco & Yohai (1996) proposed a consistent and more robust version of Pregibon's estimator, by adding a bias correction term. More specifically, they suggested to solve

$$\min_{f\in\mathcal{F}} \sum_{i=1}^{n} \rho(d_i) + C_i \tag{8}$$

with the d_i previously defined and the bias correction term C_i, where $C_i = G(P(\boldsymbol{x}_i)) + G(1 - P(\boldsymbol{x}_i)) - G(1)$, $G(t) = \int_0^t \rho'(-\log u)\mathrm{d}u$, and

$$\rho(t) = \begin{cases} t - \frac{t^2}{2c} & \text{if } t \le c \\ \frac{c}{2} & \text{otherwise} \end{cases} \tag{9}$$

where c is a constant. Croux & Haesbroeck (2003) pointed out that the minimizer of (8) with $\rho(t)$ in (9) does not exist quite often, in particular, the minimizer tends to be infinity. To overcome this problem, they suggested to use

$$\rho(t) = \begin{cases} t\mathrm{e}^{-\sqrt{d}} & \text{if } t \le d \\ -2\mathrm{e}^{-\sqrt{t}}(1+\sqrt{t}) + \mathrm{e}^{-\sqrt{d}}(2(1+\sqrt{d})+d) & \text{otherwise} \end{cases} \tag{10}$$

and

$$G(t) = \begin{cases} t\mathrm{e}^{-\sqrt{-\log t}} + \mathrm{e}^{1/4}\sqrt{\pi}\Phi\left(\sqrt{2}\left(\frac{1}{2} + \sqrt{-\log t}\right)\right) - \mathrm{e}^{1/4}\sqrt{\pi} & \text{if } t \le d \\ \mathrm{e}^{-\sqrt{d}}t - \mathrm{e}^{-1/4}\sqrt{\pi} + \mathrm{e}^{1/4}\sqrt{\pi}\Phi\left(\sqrt{2}\left(\frac{1}{2} + \sqrt{d}\right)\right) & \text{otherwise} \end{cases} \tag{11}$$

where d is a constant and Φ is the normal cumulative distribution function. To view the method by Croux & Haesbroeck (2003) in the loss function framework, we show that the problem (8) with $\rho(t)$ in (10) is equivalent to solving

$$\min_{f\in\mathcal{F}} \sum_{i=1}^{n} l^{\mathrm{CH}}(y_i f(\boldsymbol{x}_i)) \tag{12}$$

where

$$
\begin{aligned}
l^{\mathrm{CH}}(u) = {} & I_{\{u \geq -\log(e^d-1)\}} \left[\log(1+e^{-u}) e^{-\sqrt{d}} + e^{-\sqrt{d}} \frac{1}{1+e^{-u}} - e^{-1/4}\sqrt{\pi} + e^{1/4}\sqrt{\pi}\Phi\left(\sqrt{2}\left(\frac{1}{2}+\sqrt{d}\right)\right)\right] \\
& + I_{\{u < -\log(e^d-1)\}} \left[-2e^{-\sqrt{\log(1+e^{-u})}}\left(1+\sqrt{\log(1+e^{-u})}\right) + e^{-\sqrt{d}}\left(2(1+\sqrt{d})+d\right)\frac{1}{1+e^{-u}} e^{-\sqrt{\log(1+e^{-u})}} + e^{1/4}\sqrt{\pi}\Phi\left(\sqrt{2}\left(\frac{1}{2}+\right.\right.\right. \\
& + I_{\{u \geq \log(e^d-1)\}} \left[\frac{1}{1+e^{u}} e^{-\sqrt{\log(1+e^{u})}} + e^{1/4}\sqrt{\pi}\Phi\left(\sqrt{2}\left(\frac{1}{2}+\sqrt{\log(1+e^{u})}\right)\right) - e^{-1/4}\sqrt{\pi}\right] \\
& + I_{\{u < \log(e^d-1)\}} \left[e^{-\sqrt{d}} \frac{1}{1+e^{u}} - e^{-1/4}\sqrt{\pi} + e^{-1/4}\sqrt{\pi}\Phi\left(\sqrt{2}\left(\frac{1}{2}+\sqrt{d}\right)\right)\right]
\end{aligned}
\tag{13}
$$

The loss function (13) is plotted in Figure 1.

Another attempt to achieve robustness was made by Copas (1988), who modeled contamination of class labels in the training data. Specifically, it is assumed that the class label $y \in \{1, -1\}$ was transposed with a small probability γ. As a result, the response y can be 1 with probability $P^*(\boldsymbol{x})$, where

$$
P^*(\boldsymbol{x}) = (1-\gamma)P(\boldsymbol{x}) + \gamma(1-P(\boldsymbol{x})) \tag{14}
$$

Using (1) and (14), the log-likelihood with $P^*(\boldsymbol{x})$ becomes

$$
\begin{aligned}
& \sum_{i=1}^{n} \left[\frac{1+Y_i}{2} \log P^*(\boldsymbol{x}_i) + \frac{1-Y_i}{2} \log(1-P^*(\boldsymbol{x}_i)) \right] \\
& = \sum_{i=1}^{n} \left[\frac{1+Y_i}{2} \log \frac{1+\gamma(e^{-f(\boldsymbol{x}_i)}-1)}{1+e^{-f(\boldsymbol{x}_i)}} + \frac{1-Y_i}{2} \log \frac{1+\gamma(e^{f(\boldsymbol{x}_i)}-1)}{1+e^{f(\boldsymbol{x}_i)}} \right] \\
& = \sum_{i=1}^{n} \left[I_{(Y_i=1)} \log \frac{1+\gamma(e^{-Y_i f(\boldsymbol{x}_i)}-1)}{1+e^{-Y_i f(\boldsymbol{x}_i)}} + I_{(Y_i=-1)} \log \frac{1+\gamma(e^{-Y_i f(\boldsymbol{x}_i)}-1)}{1+e^{-Y_i f(\boldsymbol{x}_i)}} \right] = \sum_{i=1}^{n} \log \frac{1+\gamma(e^{-Y_i f(\boldsymbol{x}_i)}-1)}{1+e^{-Y_i f(\boldsymbol{x}_i)}}
\end{aligned}
\tag{15}
$$

To view this in the loss framework, we get the equivalent problem of log-likelihood maximization in (15) as follows

$$
\min_{f \in \mathcal{F}} \sum_{i=1}^{n} l^{\mathrm{Copas}}(y_i f(\boldsymbol{x}_i)) \tag{16}
$$

where $l^{\mathrm{Copas}}(u) = \log(1+e^{-u})/(1+\gamma(e^{-u}-1))$, which is plotted with $\gamma = 0.02$ in the right panel of Figure 1. With any γ smaller than 0.5, $l^{\mathrm{Copas}}(u)$ is decreasing in u, and bounded by

$-\log \gamma$. Though it reduces the impact of outliers, it heavily depends on the misclassification rate γ, which is often unknown and needs to be tuned.

Overall, despite progress on several variants of PLR to achieve robustness, there is still room for improvement as discussed earlier. In the next section, we propose a new classifier which effectively reduces the influence of outliers by truncating the logistic loss function.

4. ROBUST PENALIZED LOGISTIC REGRESSION

4.1. Truncated Loss for Robustness

Although most of the previous methods of robust logistic regression use the likelihood point of view, they can be transformed into the loss function framework as shown in the right panel of Figure 1. In this article, we propose a different approach to achieve robustness for the logistic regression. In particular, we develop a new classifier by truncating the loss function directly rather than modifying the log-likelihood function.

Our focus here is on outliers that are far from their own classes. Due to the unboundedness of the logistic loss function, it assigns large loss values for those outliers. Consequently, the resulting classifiers will be affected by them (Shen et al., 2003; Liu & Shen, 2006). To reduce the effect of outliers, we propose a novel robust version of the PLR(RPLR), which truncates the loss function of the PLR. Specifically, we propose to use the truncated logistic loss function $g_s(u) = \min(l(u), l(s))$ instead of $l(u)$. Here $s \leq 0$ represents the location of truncation. As illustrated on the left panel of Figure 1, $g_s(yf(\boldsymbol{x}))$ increases as $yf(\boldsymbol{x})$ decreases, but once $yf(\boldsymbol{x})$ is less than s, $g_s(yf(\boldsymbol{x}))$ becomes a constant. This implies that g_s becomes bigger as an observation gets further away from the classification boundary up to an upperbound. For outliers located further away from the boundary satisfying $yf(\boldsymbol{x}) \leq s$, the loss stays at a constant $l(s)$ so that the outliers cannot further influence the classification boundary. This is in contrast to the untruncated version whose impact grows to infinity. Furthermore, it differs from other methods discussed in the previous section in the sense that the effect of extreme observations stays the same once $yf(\boldsymbol{x})$ becomes less than s, while that of others keeps increasing. Note that s determines the level of truncation. When $s = -\infty$, no truncation occurs, thus the loss is the same as the original logistic loss. As s gets closer to 0, we have more truncation on the loss which may further reduce the effect of outliers. Therefore, $g_s(u)$ contains a group of loss functions indexed by s.

From the likelihood point of view, minimizing $\sum_{i=1}^{n} g_s(y_i f(\boldsymbol{x}_i))$ is equivalent to maximizing

$$\prod_{i=1}^{n} Q^{+}(\boldsymbol{x}_i)^{(1+y_i)/2}(1 - Q^{-}(\boldsymbol{x}_i))^{(1-y_i)/2} \tag{17}$$

where $Q^{+}(\boldsymbol{x}) = \max(P(\boldsymbol{x}), 1/(1 + e^{-s}))$ and $Q^{-}(\boldsymbol{x}) = \min(P(\boldsymbol{x}), 1/(1 + e^{s}))$. Interestingly, (17) has a similar form as that of the logistic regression in (3). The difference is that the ith factor is $Q^{+}(\boldsymbol{x}_i)$ or $1 - Q^{-}(\boldsymbol{x}_i)$, instead of $P(\boldsymbol{x}_i)$ or $1 - P(\boldsymbol{x}_i)$, depending on y_i. Hence, maximizing (17) is equivalent to finding $(\boldsymbol{w}, b)$ which gives big $Q^{+}(\boldsymbol{x})$ when $y = +1$ and small $Q^{-}(\boldsymbol{x})$ when $y = -1$. By definition, $Q^{+}(\boldsymbol{x})$ cannot get extremely small because it is lower bounded by $(1 + e^{-s})^{-1}$. Similarly, $Q^{-}(\boldsymbol{x})$ cannot get extremely big. Therefore, outliers may not influence (17) as much compared to (3). As a result, the maximizer of (17) can be less sensitive to outliers. For the toy example illustrated in Figure 2, the classification boundary of the original PLR deteriorates dramatically when there exists an extreme outlier in the dataset. In contrast, the RPLR boundary is very stable whether there is an outlier or not.

4.2. Fisher Consistency

In this section, we study Fisher consistency of robust logistic regression and its weighted version. Fisher consistency, also known as classification-calibration (Bartlett, Jordan & McAuliffe, 2006), requires that the population minimizer of a binary loss function has the same sign as $P(\boldsymbol{x}) - 1/2$ (Lin, 2004). Wu & Liu (2007) established the conditions of a truncated loss for Fisher consistency. In particular, the binary truncated logistic loss function $g_s(u) = \min(l(u), l(s))$ is Fisher consistent for any $s \le 0$. For the multicategory case with $k \ge 3$ classes, $g_s(u)$ is Fisher consistent for $s \in [-\log(2^{k/(k-1)} - 1), 0]$. In the binary case, the interval reduces to $s \in [-\log 3, 0]$. In this article, we consider three different truncation locations $s = 0, -\log 3$, and $-\log 7$ for the RPLR. The corresponding values of the logistic loss are $l(0)$, $2l(0)$, and $3l(0)$, respectively. Our numerical results suggest that $s = -\log 3$ with $l(s) = 2l(0) = 2 \log 2$ gives the best performance. This matches the Fisher consistency result for multicategory classification.

So far, we have focused on the standard case, that is, treating different types of misclassification equally. Sometimes, it can be natural to impose different costs for different types of misclassification. For example, it can be more severe to misclassify an observation of class +1 to class −1 than that of class −1 to +1. Then it is sensible to put a bigger cost for the first kind of misclassification than the second type. Lin, Lee & Wahba (2002) discussed the weighted SVM to deal with non-standard situations such as different misclassification costs for different classes. Recently, Wang, Shen & Liu (2007) applied weighted learning to large margin classifiers for probability estimation. In addition to Fisher consistency of non-weighted robust logistic regression, we investigate similar properties of the weighted robust logistic regression.

Let $(1 - \pi, \pi)$ with $0 < \pi < 1$ be the weights for class +1 and class −1, respectively, then the weighted version of the RPLR becomes

$$\min_{f \in \mathcal{H}_K} (1-\pi) \sum_{y_i=1} g_s(y_i f(\boldsymbol{x}_i)) + \pi \sum_{y_i=-1} g_s(y_i f(\boldsymbol{x}_i)) + \frac{\lambda}{2} J(f) \tag{18}$$

where $\lambda > 0$ balances the goodness of fit, measured by the loss function, and the smoothness of f. If $\lambda = 0$, the objective function in (18) reduces to the unpenalized robust logistic regression. Note that the expectation of the weighted loss part in (18) is $E[h_\pi(Y) g_s(Yf(\boldsymbol{X}))]$, where $h_\pi(1) = 1 - \pi$ and $h_\pi(-1) = \pi$.

To understand the RPLR further, we need to explore properties of weighted robust logistic regression. The following theorem discusses the theoretical minimizer of the truncated logistic loss.

Theorem 1. *The minimizer* f_π^* *of* $E[h_\pi(Y) g_s(Yf(\boldsymbol{X}))]$ *has the same sign as* $P(\boldsymbol{x}) - \pi$.

Theorem 1 indicates that the sign of f_π^* is the same as $\text{sign}(P(\boldsymbol{x}) - \pi)$. Thus, $\text{sign}(f_\pi^*)$ provides a natural estimate of $\text{sign}(P(\boldsymbol{x}) - \pi)$. In particular, if $f_\pi^* > 0$, then $P(\boldsymbol{x}) > \pi$, otherwise $P(\boldsymbol{x}) \le \pi$. This offers a natural procedure for class probability estimation. In particular, one can estimate f_π^* for many different π's $\in (0, 1)$ to obtain further information about $P(\cdot)$. Thus, it can be used for class probability estimation, as discussed further in Section 4.3.

4.3. Probability Estimation

Lin (2002) showed that under certain conditions the solution $\hat{f}_\pi$ of (18) approaches $f_\pi^* = \text{argmin}_f [h_\pi(Y) g_s(Yf(\boldsymbol{X}))]$. Therefore, we can use the property of f_π^* to design estimators

of class probabilities $\hat{P}(\boldsymbol{x})$. In the simplest scenario where $\pi = 1/2$ and $s = -\infty$, we use the regular logistic loss and (18) reduces to the ordinary PLR. In that case, it is well known that the minimizer of $E[l(Yf(X))]$ is $f = \log[p(X)/(1 - p(X))]$. Then a natural estimator of $P(\boldsymbol{x})$ is $e^{\hat{f}}/(1 + e^{\hat{f}})$.

When we use the truncated loss function, the minimizer of $E[h_\pi(Y)g_s(Yf(X))]$ does not always maintain enough information to obtain class probability estimation. The following theorem establishes the minimizer of $E[h_\pi(Y)g_s(Yf(X))]$.

Theorem 2. *Define* $H_1(\pi, P(\boldsymbol{x})) = \log[1 + 1/\tau(P(\boldsymbol{x}), \pi)] + [1/\tau(P(\boldsymbol{x}), \pi)] \log[1 + \tau(P(\boldsymbol{x}), \pi)]$, $H_2(\pi, P(\boldsymbol{x})) = \tau(P(\boldsymbol{x}), \pi) \log[1 + 1/\tau(P(\boldsymbol{x}), \pi)] + \log[1 + \tau(P(\boldsymbol{x}), \pi)]$, and $\tau(P(\boldsymbol{x}), \pi) = ((1 - \pi)P(\boldsymbol{x}))/(\pi(1 - P(\boldsymbol{x})))$. Then, for $t = g_s(s)$,

$$f_\pi^* = \begin{cases} \log \tau(P(\boldsymbol{x}), \pi) & \text{if } t > H_1(\pi, P(\boldsymbol{x})) \text{ and } t > H_2(\pi, P(\boldsymbol{x})) \\ -\infty & \text{if } t < H_1(\pi, P(\boldsymbol{x})) \text{ and } H_1(\pi, P(\boldsymbol{x})) > H_2(\pi, P(\boldsymbol{x})) \\ \infty & \text{if } t < H_2(\pi, P(\boldsymbol{x})) \text{ and } H_1(\pi, P(\boldsymbol{x})) < H_2(\pi, P(\boldsymbol{x})) \\ -\infty, \infty & \text{if } t < H_1(\pi, P(\boldsymbol{x})) = H_2(\pi, P(\boldsymbol{x})) \end{cases}$$

Theorem 2 implies that we can use f_π^* to express class probability only when $f_\pi^* = \log \tau(P(\boldsymbol{x}), \pi) = \log((1 - \pi)P(\boldsymbol{x}))/(\pi(1 - P(\boldsymbol{x})))$. Otherwise we cannot reconstruct $P(\boldsymbol{x})$ using f_π^*. To further illustrate the relationship between f_π^* and $P(\boldsymbol{x})$, we consider H_1 and H_2 in the case that $\pi = 1/2$ in Figure 3. When $P(\boldsymbol{x}) \in [p_1, p_2]$ with $t = H_1(\pi, p_1)$ and $t = H_2(\pi, p_2)$, then $f_\pi^* = \log((1 - \pi)P(\boldsymbol{x}))/(\pi(1 - P(\boldsymbol{x})))$. However, when $P(\boldsymbol{x}) \notin [p_1, p_2]$, f_π^* is either ∞ or $-\infty$, which does not have enough information to recover $P(\boldsymbol{x})$. For this reason, we need to explore other schemes to estimate $P(\boldsymbol{x})$.

To estimate the class probability, we propose the following three schemes.

Scheme 1: Since the RPLR works only for estimation of $P(\boldsymbol{x}) \in [p_1, p_2]$, we can consider utilizing it for those p, and using the ordinary PLR for $P(\boldsymbol{x}) \notin [p_1, p_2]$. Notice that this scheme is valid only for $t > 2 \log 2$, because if $t \leq 2 \log 2$, $p_1 = p_2$ and t is smaller than H_1 and H_2 for any $P(\boldsymbol{x})$ as shown in Figure 3. Thus, by Theorem 2, the RPLR does not work for estimation of any $P(\boldsymbol{x})$ when $t \leq 2 \log 2$.

This scheme is a valid approach in the sense that estimation of $P(\boldsymbol{x}) \in [p_1, p_2]$ is more critical than that of $P(\boldsymbol{x}) \notin [p_1, p_2]$. Usually the data points with very small $P(\boldsymbol{x})$ or very big $P(\boldsymbol{x})$ are easier to classify and we are more certain about the class membership of those points. However, class membership prediction for data points with $P(\boldsymbol{x})$ near 1/2 is not only difficult, but also highly affected by outliers. Thus, estimation of class probability becomes more important for those points. Therefore, we use the RPLR for estimation of $P(\boldsymbol{x}) \in [p_1, p_2]$, and use the ordinary PLR for $P(\boldsymbol{x}) \notin [p_1, p_2]$.

Scheme 2: The second scheme is motivated by the idea that we can shift p_1 and p_2 by changing π. Because H_1 and H_2 in Theorem 2 depend on π, different π's bring different estimable regions $[p_1, p_2]$. Hence, we can cover most of the $P(\boldsymbol{x}) \in [0, 1]$ using many different π's. Note that this method is applicable only when $t > 2 \log 2$, and here we illustrate the case with $t = 3 \log 2$. More specifically, we use seven different π's such as $\pi_1 = 1/2$, $\pi_2 = 1/5$, $\pi_3 = 4/5$, $\pi_4 = 1/20$, $\pi_5 = 19/20$, $\pi_6 = 1/91$, and $\pi_7 = 90/91$, which give different estimable regions for $P(\boldsymbol{x})$, [0.310, 0.690], [0.105, 0.358], [0.642, 0.899], [0.024, 0.101], [0.895, 0.976], [0.005, 0.024], and [0.976, 0.995]. Using $\hat{f}_j$ which denotes the solution from the RPLR with π_j, we can construct the estimator $\hat{P}_j(\boldsymbol{x}) = e^{\hat{f}_j}/(1 + e^{\hat{f}_j})$; $j = 1, \ldots, 7$, to estimate $P(\boldsymbol{x})$ in the corresponding region.

There are some drawbacks of the second scheme. First, there are overlaps between the estimable regions. Moreover, the RPLR with different π's can give contradictory inference about $P(\boldsymbol{x})$. To solve this, for given $\hat{P}^j(\boldsymbol{x})$, we consider $\hat{P}^1(\boldsymbol{x})$ first. If $\hat{P}^1(\boldsymbol{x}) \in [0.310, 0.690]$, then take $\hat{P}^1(\boldsymbol{x})$ as $\hat{P}(\boldsymbol{x})$. Otherwise, we consider $\hat{P}^2(\boldsymbol{x})$ or $\hat{P}^3(\boldsymbol{x})$ depending on whether $\hat{P}^1(\boldsymbol{x})$ is <0.310 or >0.690. Then take $\hat{P}^2(\boldsymbol{x})$ or $\hat{P}^3(\boldsymbol{x})$ as $\hat{P}(\boldsymbol{x})$ if it falls in the estimable region, otherwise, take $\hat{P}^4(\boldsymbol{x})$ or $\hat{P}^5(\boldsymbol{x})$ in the same manner as $\hat{P}(\boldsymbol{x})$ or use $\hat{P}^6(\boldsymbol{x})$ or $\hat{P}^7(\boldsymbol{x})$ likewise. If the RPLR with $\hat{P}^j(\boldsymbol{x})$ gives contradictory inference about $P(\boldsymbol{x})$ or none of them gives the estimate of $P(\boldsymbol{x})$ in the estimable region, then we use the PLR to estimate $P(\boldsymbol{x})$.

Scheme 3: Wang, Shen & Liu (2007) suggested to estimate the class probability for large margin classifiers via bracketing the probability using multiple weighted classifiers. We consider to apply the similar idea to the RPLR. First, we make equally spaced partitions of [0,1], that is, $0 = \pi_0 < \pi_1 < \ldots < \pi_m < \pi_{m+1} = 1$ such that $\pi_{j+1} - \pi_j$ is constant for any $i = 0, \ldots, m$. Then we can obtain $\hat{f}_j$ from the RPLR with π_j, $j = 1, \ldots, m$. By Theorem 1, $\hat{f}_j$ estimates whether the class probability is greater than π or not. Therefore, if we make the partition fine enough, then we can achieve probability estimation with the desired level of accuracy. To be more specific, we define $\pi^* = \arg\max_{\pi_j}\{\hat{f}_j > 0\}$ and $\pi_* = \arg\max_{\pi_j}\{\hat{f}_j < 0\}$, then $\hat{p}$ is obtained by $1/2(\pi^* + \pi_*)$.

This method is not restricted by the truncation location, that is, we can use this method for any $t > \log 2$, corresponding to $s \leq 0$. The larger m we use, the finer estimate we can get. However, larger m's require higher computational costs. As discussed in Wang, Shen & Liu (2007), this scheme provides consistent estimators for the class probability. Our numerical examples demonstrate that the third scheme works the best among the three schemes.

5. COMPUTATIONAL ALGORITHMS

Since the loss function g_s is not convex, the RPLR requires non-convex minimization. Note that g_s can be written as the difference of two convex functions as $g_s(u) = l(u) - l_s(u)$ as shown in the left panel of Figure 1.With this decomposition, we can solve the non-convex minimization via the d.c. algorithm (An & Tao, 1997; Liu, Shen & Doss, 2005). For each iteration, l_s is replaced by its linear approximation using the current solution. Then the problem becomes convex minimization. We iterate this until the objective function converges.

In the literature, Fan & Li (2001) introduced local quadratic approximation (LQA) to solve penalized likelihood optimization problems. Hunter & Li (2005) studied convergence of LQA as an instance of minorize–maximize or majorize–minimize (MM) algorithm. Considering a linear approximation of l_s as the affine minorization, the d.c. algorithm for RPLR is also a special case of the MM algorithm. Since the objective function in (18) is positive, our d.c. algorithm converges to an ε-local minimizer in finite iterations (An & Tao, 1997; Liu, Shen & Doss, 2005). In this section, we discuss the d.c. algorithm for the RPLR.

In linear learning with $f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$, (18) can be reduced to

$$\min_{b,\boldsymbol{w}} \sum_{i=1}^{n} h_\pi(y_i) g_s(y_i f(\boldsymbol{x}_i)) + \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2 \tag{19}$$

Using the fact that $g_s(u) = l(u) - l_s(u)$ with $l(u) = \log(1 + e^{-u})$ and $l_s(u) = [\log(1 + e^{-u}) - \log(1 + e^{-s})]_+$, (19) can be written as

$$\min_{\Theta} Q(\Theta)=\min_{\Theta} Q_{\text{vex}}(\Theta)+Q_{\text{cav}}(\Theta) \tag{20}$$

where $\Theta = (b, \boldsymbol{w})$,

$Q_{\text{vex}}(\Theta)^s=\frac{\lambda}{2}\|\boldsymbol{w}\|_2^2+\sum_{i=1}^n h(y_i)l(y_i\, f(\boldsymbol{x}_i))$ and $Q_{\text{cav}}(\Theta)^s=-\sum_{i=1}^n h(y_i)l_s(y_i\, f(\boldsymbol{x}_i))$. Then, at the $(m + 1)$th iteration, the d.c. algorithm minimizes

$$Q_{\text{vex}}(\Theta_m)^s+\langle\frac{\partial}{\partial \boldsymbol{w}}Q^s_{\text{cav}}(\Theta_m), \boldsymbol{w}\rangle+b\frac{\partial}{\partial b}Q^s_{\text{cav}}(\Theta_m)=\frac{\lambda}{2}\|\boldsymbol{w}\|_2^2+\sum_{i=1}^n h(y_i)\log(1+e^{-y_i f(\boldsymbol{x}_i)})+\sum_{i=1}^n h(y_i)\beta_i\frac{e^{-y_i f_m(\boldsymbol{x}_i)}}{1+e^{-y_i f_m(\boldsymbol{x}_i)}}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b) \tag{21}$$

where $f_m(\boldsymbol{x})=\boldsymbol{w}_m^{\mathrm{T}}\boldsymbol{x}+b_m$ and $\beta_i = 1$ if $y_i = 1$ and $f(\boldsymbol{x}_i) < s$, -1 if $y_i = -1$ and $f(\boldsymbol{x}_i) > -s$, and 0 otherwise. Problem (21) can then be solved using nonlinear convex minimization techniques.

The algorithm can be extended to nonlinear learning directly. Specifically, for kernel learning, (18) becomes

$$\min_{b,\boldsymbol{\upsilon}}\sum_{i=1}^n h_\pi(y_i)g_s(y_i f(\boldsymbol{x}_i))+\frac{\lambda}{2}\|f\|^2_{\mathscr{H}_K} \tag{22}$$

where $f(\boldsymbol{x})=\sum_{i=1}^n \upsilon_i K(\boldsymbol{x}_i, \boldsymbol{x})+b$ and $\boldsymbol{\upsilon} = (\upsilon_1, \ldots, \upsilon_n)$. Notice that $\sum_{i=1}^n \upsilon_i K(\boldsymbol{x}_i, \boldsymbol{x}) \in H_K$ and $\|f\|^2_{\mathscr{H}_K}=\langle\boldsymbol{\upsilon}, K\boldsymbol{\upsilon}\rangle$. Using $\Theta = (b, \boldsymbol{\upsilon})$ in (20) leads to a similar algorithm for the nonlinear kernel learning case.

6. TUNING PARAMETER SELECTION

The tuning parameter λ in (19) and (22) plays an important rule for the RPLR. In this section, we explore various ways to tune λ. We use the penalty term which measures smoothness of the model to avoid overfitting the data, and the tuning parameter λ decides how smooth our model will be. Thus, the choice of λ has a big impact on the resulting model.

There are numerous ways proposed to tune λ in the penalized likelihood literature and we employ some of those here for the RPLR. Some well known ones include the cross validation, AIC, and BIC. Among them, cross validation is probably one of the most commonly used method. Cantoni & Ronchetti (2001a) pointed out that choice of λ could be influenced by outliers. They proposed robust versions of cross validation and Mallows' C_p, which are essentially equivalent to modifying the loss function by imposing weights. In contrast, our RPLR automatically chooses robust λ without employing weights, because the loss function itself is already designed to reduce the effect of outliers. Since cross validation requires intensive computation, generalized approximate cross validation (GACV) can be a

good approximation. In this section, we explore how to generalize GACV to the RPLR problem.

Xiang & Wahba (1996) proposed GACV for the PLR, which estimates comparative Kullback–Leibler distance between the true linear predictor $f(\boldsymbol{x})$ and the estimated one for a particular λ. It starts with a leaving-out-one version, then uses Taylor expansion to get an estimate. This idea can be generalized here to get GACV of the RPLR. The details are as follows.

Let $f_\lambda(\boldsymbol{x})$ be the solution of the RPLR for a particular value of λ. The Kullback–Leibler distance $KL(f, f_\lambda)$ is

$$\mathrm{KL}(f, f_\lambda)=\frac{1}{n}\sum_{i=1}^{n} E\log\frac{\tilde{L}(y_i, f(\boldsymbol{x}_i))}{\tilde{L}(y_i, f_\lambda(\boldsymbol{x}_i))} \tag{23}$$

where $\tilde{L}(y_i, f(\boldsymbol{x}_i)) = P(\boldsymbol{x}_i)^{(1+y_i)/2}(1 - P(\boldsymbol{x}_i))^{(1-y_i)/2}$ for the PLR and $\tilde{L}(y_i, f(\boldsymbol{x}_i)) = Q^{+}(\boldsymbol{x}_i)^{(1+y_i)/2}(1 - Q^{-}(\boldsymbol{x}_i))^{(1-y_i)/2}$ for the RPLR. Since the true $f(\boldsymbol{x})$ is unknown and does not depend on λ, we define the comparative KL loss,

$$\mathrm{CKL}(\lambda)=\mathrm{KL}(f, f_\lambda) - \frac{1}{n}\sum_{i=1}^{n} E\log\tilde{L}(y_i, f(\boldsymbol{x}_i)) \tag{24}$$

to compare models with different λ. It can be shown that

$\mathrm{CKL}(\lambda)=1/n\sum_{i=1}^{n} E[-z_i f_\lambda(\boldsymbol{x}_i)+\log(1+e^{f_\lambda(\boldsymbol{x}_i)})]$ for the PLR, and

$\mathrm{CKL}(\lambda)=1/n\sum_{i=1}^{n} E[\min\{t, -z_i f_\lambda(\boldsymbol{x}_i)+\log(1+e^{f_\lambda(\boldsymbol{x}_i)})\}]$ for the RPLR, with $z_i = 1/2(1 + y_i)$. Then the remaining issue is how to estimate the CKL. After some derivation (the details are included in the Appendix Section), we define GACV for the RPLR as follows

$$\mathrm{GACV}(\lambda)=\frac{1}{n}\sum_{i=1}^{n}\min\left\{t, -z_i f_\lambda(\boldsymbol{x}_i)+\log(1+e^{f_\lambda(\boldsymbol{x}_i)})\right\}+\frac{\mathrm{tr}(H)}{n}\sum_{i=1}^{n} d_i\frac{h_{ii}z_i(z_i - P_\lambda(\boldsymbol{x}_i))}{n - \mathrm{tr}(W^{*1/2}HW^{*1/2})} \tag{25}$$

where $H = \{W(\boldsymbol{f}_\lambda) + n\lambda\Sigma\}^{-1}$ with Σ such that $\boldsymbol{f}^{\mathrm{T}}\Sigma\boldsymbol{f}$, h_{ii} is the ith diagonal entry of H, $P_\lambda(\boldsymbol{x}) = 1/(1 + e^{-f_\lambda(\boldsymbol{x})})$, and

$$d_i=\begin{cases} 1 & \text{if } t> \max(a_i+b_i, a_i) \\ 0 & \text{if } t< \min(a_i+b_i, a_i) \\ \frac{t-(a_i+b_i)}{-b_i} & \text{if } a_i+b_i<t<a_i \\ \frac{t-a_i}{b_i} & \text{if } a_i<t<a_i+b_i \end{cases} \tag{26}$$

with $a_i = -z_i f_\lambda(\boldsymbol{x}_i) + \log(1 + e^{f_\lambda(\boldsymbol{x}_i)})$ and $b_i = z_i(f_\lambda(\boldsymbol{x}_i) - f_\lambda^{(-i)}(\boldsymbol{x}_i))$, where $f_\lambda^{(-i)}(\cdot)$ is the solution of the RPLR with the ith data point omitted. Using the fact that $0 < d_i < 1$, we can bound GACV(λ). We use the average of the upper and lower bound of GACV. In particular, we define the EGACV

$$\mathrm{EGACV}(\lambda)=\frac{1}{n}\sum_{i=1}^{n}\min\left\{t, -z_i f_\lambda(\boldsymbol{x}_i)+\log(1+e^{f_\lambda(\boldsymbol{x}_i)})\right\}+\frac{\mathrm{tr}(H)}{2n}\sum_{i=1}^{n}\frac{h_{ii}z_i(z_i - P_\lambda(\boldsymbol{x}_i))}{n - \mathrm{tr}(W^{*1/2}HW^{*1/2})} \tag{27}$$

We use simulated data to illustrate the performance of EGACV(λ). The training set consists of 50 data points sampled from the uniform distribution over a unit disk $\{(x_1, x_2): x_1^2 + x_2^2 1\}$ and labeled as $y = 1$ if $x_1 \geq x_2$, $y = -1$ otherwise. The testing set has 10^5 data points which are sampled and labeled in the same manner as the training set. Using these datasets, we build a model using the RPLR with $t = 2 \log 2$ based on the training set and calculate CKL(λ) of the testing set for each λ such that $\log_{10} \lambda \in \{-3.0, -2.9, \ldots, 2.0\}$. Then we calculate EGACV(λ) using the training set only and plot it with CKL(λ) to see how close they are. We repeat this 100 times with a different training set each time and take average of EGACV(λ) and CKL(λ) and plot them. The left panel of Figure 4 illustrates typical curves of EGACV(λ) and CKL(λ) from one example, and the average curves of the 100 repetitions are plotted in the right panel. The solid line shows CKL(λ), the dashed line shows EGACV(λ), and the dotted lines show the upper and lower bounds of GACV(λ). As shown in Figure 4, EGACV(λ) reflects the variation of CKL(λ) quite well, thus EGACV(λ) can be a useful tool to tune λ.

7. NUMERICAL EXAMPLES

In this section, we examine performance of the RPLR. Using two simulated examples and two real data examples, we compute the PLR and RPLR to compare their classification errors as well as accuracy of class probability estimation.

7.1. Simulation

In the simulated examples, data are generated with the sample sizes of training, tuning and testing sets 100, 100, and 10^6, respectively. The training data sets are used to build classifiers, and λ is chosen by two different ways: by a grid search based on the tuning sets, and by a grid search based on the EGACV calculated from the training set. The testing errors and probability estimation errors are evaluated using the testing sets.

Example 1: The data are generated as follows. First, (x_1, x_2) is sampled from the uniform distribution over a unit disk $\{(x_1, x_2): x_1^2 + x_2^2 \leq 1\}$. Then, set $y = 1$ if $x_1 \geq x_2$, $y = -1$ otherwise. To demonstrate robustness of the RPLR, we randomly select v percent of the observations and change their class labels to the other classes, where $v = 0\%$, 5%, 10%, and 20%. For each value of v, we repeat the classification procedure 100 times to capture variation of the results. Since the true boundary is linear, we focus on linear learning in this example. For the RPLR, we use $s = 0$, $-\log 3$, and $-\log 7$ which correspond to $t = 2 \log 2$, $2 \log 2$, and $3 \log 2$, respectively. We also report misclassification rate of the RPLR when we tune s along with λ, as well as results of another version of logistic regression proposed by Croux & Haesbroeck (2003) for comparison. For class probability estimation, we apply Scheme 3 to each t, but Scheme 1 and Scheme 2 are used only for $t = 3 \log 2$ because they are valid only if $t > 2 \log 2$. To evaluate accuracy of probability estimation, we use $1/n' \sum_{i=1}^{n'} |\widehat{P}(\boldsymbol{x}_i) - P(\boldsymbol{x}_i)|$ to measure the probability estimation error, where n' is the size of the testing set.

Results are summarized in Tables 1 and 2. With no contamination, the RPLR and the PLR perform very similarly. As we increase the percent of contamination, the RPLR performs better than the PLR because the truncated loss is more robust against outliers.

The truncation location is an important issue. If the loss function is not truncated, it can be sensitive to outliers. If the loss function is truncated too much, we may under use the information of those data points close to the decision boundary. The performance of the RPLR with $t = \log 2$ corresponding to the most truncation is indeed suboptimal as shown in

Tables 1 and 2. The RPLR with $t = 3 \log 2$ works the best for the cases $v = 0$ and 5, but as the proportion of contamination grows, performance of the RPLR with $t = 2 \log 2$ becomes the best. This is reasonable because more truncation helps for data with more outliers. In general, we recommend to use $t = 2 \log 2$ for the truncation location of binary problems. This choice also has good theoretical justification as mentioned in Section 4.2 in terms of Fisher consistency. The results with $t = 2 \log 2$ are comparable to that using the tuned t, but a fixed t can be more efficient to compute.

Regarding to the choice of λ, the one chosen based on the tuning set performs better than the one by the EGACV. This may not be surprising because the first approach uses information from both the training set and the tuning set to choose λ, while the EGACV approach uses the training set only. Hence a direct comparison may not be fair considering the difference in the amount of information used between the two approaches. Nevertheless we can see that the EGACV approach works reasonably well in this example.

Note that the overall performance of the robust estimator of the logistic regression by Croux & Haesbroeck (2003) is not as good as that of the RPLR, especially when the data are highly contaminated.

As to the issue of class probability estimation, the RPLR with $t = 3 \log 2$ works the best for non-contaminated data, but $t = 2 \log 2$ becomes better as the rate of contamination increases. This agrees with the results of classification errors. In general, better classification performance can be translated into better class probability estimation. Thus, the RPLR yields more accurate class probability estimation than that of the PLR. Among three different schemes, Scheme 3 seems to perform the best overall.

To visualize the classification boundaries, we select a typical dataset and plot the corresponding boundaries yielded by the PLR and the RPLR on the left panel of Figure 5. Clearly, the RPLR is much less sensitive to outliers and deliver more accurate classification boundary than that of the PLR.

Example 2: We generate (x_1, x_2) uniformly from the unit disk $\{(x_1, x_2): x_1^2 + x_2^2 \leq 1\}$ with y being 1 if $(x_1 - x_2)(x_1 + x_2) < 0$, and −1 otherwise. Then we flip the class labels using the same strategy as in Example 1. Linear learning does not work here due to its generation. We use nonlinear learning with Gaussian kernel $K(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp(-\| \boldsymbol{x}_1 - \boldsymbol{x}_2 \|^2 / (2\sigma^2))$. We tune σ among the first quartile, the median, and the third quartile of the between-class pairwise Euclidean distances of training inputs (Wu & Liu, 2007). We use the same truncation location, class probability estimation schemes, and measure of probability estimation errors as in Example 1. Results are similar to Example 1 and not included to save space. The RPLR with $t = 2 \log 2$ works the best overall. When outliers exist in the data, truncation indeed improves both classification accuracy as well as class probability estimation. We also plot the results of one typical example on the right panel of Figure 5. Again, the RPLR is more robust and consequently its classification boundary is closer to the Bayes decision boundary.

Overall, based on these examples, we can conclude that the RPLR works better than the original PLR and is also competitive compared with the method of Croux & Haesbroeck (2003). We also explored the case when the logistic model is the true underlying model. In that case, the PLR works slightly better than that of the RPLR. When we contaminate the data with outliers, the RPLR works better than the PLR as expected.

7.2. Real Data

7.2.1. Leukaemia data—In this section, we apply the PLR and the RPLR to the leukaemia data set described in Golub et al. (1999). This data set is publicly available at:

www.broad.mit.edu/cgi-bin/cancer/datasets.cgi. It contains 72 samples with 7,129 gene expression values. The goal is to classify the patients into two types of leukaemia: acute myeloid leukaemia (AML) and acute lymphoblastic leukaemia (ALL). Since the number of genes is much higher than the sample size, we performed prescreening to choose a subset of genes. In particular, we used the ratios of between-groups to within-groups sum of squares of the genes to sort them and chose the top 40 genes. Similar procedure was done in Dudoit, Fridly & Speed (2002).

This data set includes a training set with 38 instances and a testing set with 34 instances. Heatmaps in Figure 6 are drawn for good visualization of the data sets. From the heatmap of the testing set, we can identify some observations that are difficult to classify. Indeed, there are two subjects that the PLR and the RPLR fail to classify to the correct classes. The training set is used for model building, then performance of the model is evaluated on the testing set. More specifically, the tuning parameter λ is chosen by fivefold cross validation on the training set. We also used EGACV and it gives very similar results. Using the RPLR coefficients estimated from the training set with the selected λ, class probability of each instance in the testing set is estimated. Both linear and nonlinear learning with Gaussian kernel have been performed. The results show that linear learning works better for this problem.

Figure 7 shows the results of the PLR and the RPLR with $t = 2 \log 2$. The results when $t = \log 2$ and $t = 3 \log 2$ are not reported because they are barely different from the case when $t = 2 \log 2$. The horizontal axis stands for the estimated value of linear predictor $f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$, and the vertical axis stands for the estimated probability. The observations of the classes ALL and AML are plotted as circles and squares, respectively, with a color scheme of blue for the training set (larger symbols) and red for the testing set (smaller symbols) for the online version of the plot. The solid and dashed lines are the estimated density curves of the values of linear predictors for the ALL and AML classes, respectively. Here, the class probabilities for the PLR were estimated by $\hat{P}(\boldsymbol{x}) = e^{\hat{f}}/(1 + e^{\hat{f}})$. For the RPLR, we use Scheme 3 to estimate the class probabilities. In both procedures of probability estimation, $\hat{f}(\boldsymbol{x}) > 0$ implies $\hat{P}(\boldsymbol{x}) > 0.5$, hence the $\operatorname{sign}(\hat{f}(\boldsymbol{x}))$ gives class prediction. As shown in Figure 7, there are two common misclassified observations by the PLR and RPLR. This is not surprising considering the nature of the data revealed by the heatmaps. Besides the two misclassified observations, the PLR and the RPLR show different patterns in class probability estimation. The estimated class probabilities by the RPLR are either very close to 1 or 0, while estimated probabilities by the PLR have more variability. This is because that these two classifiers have different sensitivity to outliers: since the PLR is sensitive to those two misclassified observations, the estimated probabilities of other observations are affected so that we lose some certainty about the class memberships for some of the other observations despite the clear pattern of the data. On the other hand, those two misclassified observations do not influence the RPLR as much, hence all the other class probabilities remain close to 0 or 1, which reflect the nature of the data better.

7.2.2. Lung cancer data—In this section, we apply the RPLR to the lung cancer data set described in Liu et al. (2008). The data set we use here has 12,625 genes of 188 lung cancer patients with 5 categories. There are five different categories: Adeno, Carcinoid, Colon, SmallCell, and Squamous with 128, 20, 13, 6, 21 patients, respectively. First, we calculate the ratio of the standard deviation and the sample mean of each gene, and choose 316 genes with the highest ratios. Then we standardize the genes so that each gene has sample mean 0 and sample standard deviation 1. Figure 8 is the biplot of the data after filtering and standardization on principal component analysis (PCA). Out of all five types of cancer, the Adeno group has the most broad spectrum and overlaps much with other types. This matches the biological knowledge that Adeno is a very heterogeneous lung cancer subtype

(Bhattacharjee et al., 2001). For that reason, we perform the RPLR to classify Adeno patients versus all other cancer patients.

Since there are 188 cancer patients in total, we randomly divide patients into training, tuning, and testing sets with sample sizes 63, 63, 62, respectively. Then we build a model for each value of λ and choose the λ that gives the smallest misclassification rate on the tuning set. Using the model with the selected λ, the misclassification rate on the testing set is calculated. This whole procedure is repeated for 10 times.

The results are reported in Table 3. We can see that although the difference is not very big, truncation indeed improves performance, and the truncation location that we suggest, $t = 2 \log 2$, gives the best result.

Overall, we can see that the RPLR yields competitive performance when the data are noisy with potential outliers. In practice, one may not know whether it is advantage to use robust methods for a particular application. Based on our experience, even when there are no outliers, the RPLR gives similar performance to that of the PLR. Thus, one may try both methods and compare the results.

8. DISCUSSION

In this article, we have proposed the RPLR, using the truncated logistic loss function to produce more robust classifiers to outliers than the standard PLR. Moreover, we have proposed three schemes of class probability estimation for the RPLR. Our theoretical investigation shows that the proposed RPLR is Fisher consistent and more robust than the original PLR. Numerical results demonstrate that truncation of the loss function indeed reduces the effect of outliers so that more accurate classification and class probability estimation can be obtained.

Our current study focuses on the loss function framework. It will be interesting to perform theoretical comparison of the proposed method with other existing robust logistic regression using the likelihood point of view. Future work includes the study of robustness versus efficiency as well as some comparison using the influence function as well as sensitivity curves.

We have used the L_2 penalty for the regularization term $J(f)$. It is now well known that one can use some other penalty functions to achieve variable selection. Examples of such penalty functions include the L_1 penalty (Tibshirani, 1996), the SCAD penalty (Fan & Li, 2001), the COSSO penalty (Lin & Zhang, 2006), etc. A natural extension of the RPLR is to use different penalty functions to achieve simultaneous variable selection and robust classification. Moreover, although we have focused on the binary case in this article, the truncated logistic loss is applicable for multicategory classification problems as well. The work of Zhu & Hastie (2005) can be useful here.

Acknowledgments

The authors are indebted to the editor, the associate editor, and two referees, whose helpful comments and suggestions led to a much improved presentation. This research was supported in part by National Science Foundation grant (DMS-0747575) and National Institutes of Health grant (NIH/NCI R01-CA149569).

APPENDIX

Proof of Theorem 1. Since $E[h_\pi(Y)g_s(Yf(\boldsymbol{X}))] = E[E[h_\pi(Y)g_s(Yf(\boldsymbol{X}))|\boldsymbol{X} = \boldsymbol{x}]]$, we can minimize $E[h_\pi(Y)g_s(Yf(\boldsymbol{X}))]$ by minimizing $E[h_\pi(Y)g_s(Yf(\boldsymbol{X}))|\boldsymbol{X} = \boldsymbol{x}]$ for every $\boldsymbol{x}$. Note that

$E[h_\pi(Y)g_s(Yf(\boldsymbol{X}))|\boldsymbol{X}=\boldsymbol{x}] = P(\boldsymbol{x})(1-\pi)g_s(f(\boldsymbol{x})) + (1-P(\boldsymbol{x}))\pi g_s(-f(\boldsymbol{x}))$. Because g_s is a non-increasing function, the minimizer f_π^* should satisfy that $f_\pi^* \geq 0$ if $P(\boldsymbol{x})(1-\pi) > (1-P(\boldsymbol{x}))\pi$, $f_\pi^* \leq 0$ otherwise. Note that $P(\boldsymbol{x})(1-\pi) > (1-P(\boldsymbol{x}))\pi$ is equivalent to $P(\boldsymbol{x}) > \pi$. Hence, it is sufficient to show that $f = 0$ is not a minimizer. We can assume $P(\boldsymbol{x}) > \pi$ without loss of generality. For $s = 0$, $E[h_\pi(Y)g_s(0)|\boldsymbol{X}=\boldsymbol{x}] = P(\boldsymbol{x})(1-\pi)g_s(0) + (1-P(\boldsymbol{x}))\pi g_s(0)$, and $E[h_\pi(Y)g_s(1)|\boldsymbol{X}=\boldsymbol{x}] = P(\boldsymbol{x})(1-\pi)g_s(1) + (1-P(\boldsymbol{x}))\pi g_s(-1)$. Hence $E[h_\pi(Y)g_s(0)|\boldsymbol{X}=\boldsymbol{x}] > E[h_\pi(Y)g_s(1)|\boldsymbol{X}=\boldsymbol{x}]$ because $g_s(0) > g_s(1)$ and $g_s(0) = g_s(-1)$. Thus, $f = 0$ is not a minimizer in this case. For $s < 0$,

$$\begin{aligned}
&\frac{d}{\mathrm{d}f(\boldsymbol{x})}E[\,h_\pi(Y)g_s(Yf(\boldsymbol{X}))|\boldsymbol{X}\\
&\quad=\boldsymbol{x}]|_{f(\boldsymbol{x})=0}=\frac{d}{\mathrm{d}f(\boldsymbol{x})}[\,P(\boldsymbol{x})(1\\
&\quad-\pi)g_s(f(\boldsymbol{x}))\\
&\quad+(1-P(\boldsymbol{x}))\pi g_s(\\
&\quad-f(\boldsymbol{x}))]|_{f(\boldsymbol{x})=0}\\
&\quad=P(\boldsymbol{x})(1\\
&\quad-\pi)g_s'(0)\\
&\quad+(1-P(\boldsymbol{x}))\pi g_s'(0)\\
&\quad=(P(\boldsymbol{x})\\
&\quad-\pi)g_s'(0)<0
\end{aligned}$$

because $g_s'(0)<0$. Thus, $f = 0$ is not a minimizer. Hence, $f_{pi}^*(\mathbf{x})$ has the same sign as $P(\boldsymbol{x}) - \pi$.

Proof of Theorem 2. Define $A(f) = E[h_\pi(Y)g_s(Yf(\boldsymbol{X}))|\boldsymbol{X}=\boldsymbol{x}]$. Observe that $A(f) = P(\boldsymbol{x})(1-\pi)\min(t, \log(1+e^{-f(\boldsymbol{x})})) + (1-P(\boldsymbol{x}))\pi\min(t, \log(1+e^{f(\boldsymbol{x})}))$, where $t = \log(1+e^{-s})$. We consider three cases, $s \leq f \leq -s$, $f < s$, and $f > -s$.

First, when $s \leq f \leq -s$,

$$A'(f)=\frac{d}{\mathrm{d}f(\boldsymbol{x})}[\,P(\boldsymbol{x})(1-\pi)\log(1+e^{-f})+(1-P(\boldsymbol{x}))\pi\,\log(1+e^{f})]=\frac{1}{1+e^{f}}[\,-P(\boldsymbol{x})(1-\pi)+(1-P(\boldsymbol{x}))\pi e^{f}],$$

and $A''(f) = (P(\boldsymbol{x})(1-\pi) + (1-P(\boldsymbol{x}))\pi)e^f/(1+e^f)^2$. Note that $A''(f) > 0$ for any $f \in [s, -s]$, and $A'(\tilde{f}) = 0$ when $\tilde{f} = \log((1-\pi)P(\boldsymbol{x}))/(\pi(1-P(\boldsymbol{x}))) = \log\tau(P(\boldsymbol{x}), \pi)$. Hence, $\tilde{f}$ is the minimizer of $A(f)$ for $f \in [s, -s]$. Note that $A(\tilde{f}) = (P(\boldsymbol{x})(1-\pi) + (1-P(\boldsymbol{x}))\pi)\log(P(\boldsymbol{x})(1-\pi) + (1-P(\boldsymbol{x}))\pi) - P(\boldsymbol{x})(1-\pi)\log(P(\boldsymbol{x})(1-\pi)) - (1-P(\boldsymbol{x}))\pi\log((1-P(\boldsymbol{x}))\pi)$.

Second, when $f < s$, note that $A(f) = P(\boldsymbol{x})(1-\pi)t + (1-P(\boldsymbol{x}))\pi\log(1+e^{f(\boldsymbol{x})})$ and it is an increasing function in f. Thus, the minimum of $A(f)$ in this case is $\lim_{f\to-\infty}A(f) = P(\boldsymbol{x})(1-\pi)t$.

Similarly, when $f > -s$, $A(f) = P(\boldsymbol{x})(1-\pi)\log(1+e^{-f(\boldsymbol{x})}) + (1-P(\boldsymbol{x}))\pi t$ and it is a decreasing function in f. Likewise, the minimum of $A(f)$ in this case is $\lim_{f\to\infty}A(f) = (1-P(\boldsymbol{x}))\pi t$.

Hence, $\tilde{f}$ is the minimizer of $A(f)$ if $A(\tilde{f}) < \lim_{f\to-\infty}A(f) = P(\boldsymbol{x})(1-\pi)t$ and $A(\tilde{f}) < \lim_{f\to\infty}A(f) = (1-P(\boldsymbol{x}))\pi t$. If $A(\tilde{f}) > \lim_{f\to-\infty}A(f) = P(\boldsymbol{x})(1-\pi)t$ and $\lim_{f\to\infty}A(f) = (1-P(\boldsymbol{x}))\pi t > \lim_{f\to-\infty}A(f) = P(\boldsymbol{x})(1-\pi)t$, $f = -\infty$ is the minimizer of $A(f)$. Similarly, $f = \infty$ is the minimizer of $A(f)$ if $A(\tilde{f}) > \lim_{f\to\infty}A(f) = P(\boldsymbol{x})(1-\pi)t$ and $\lim_{f\to\infty}A(f) = (1-P(\boldsymbol{x}))\pi t <$

$\lim_{f\to-\infty} A(f) = P(\boldsymbol{x})(1-\pi)t$. Finally, if $A(\tilde{f}) > \lim_{f\to\infty} A(f) = P(\boldsymbol{x})(1-\pi)t = \lim_{f\to-\infty} A(f) = P(\boldsymbol{x})(1-\pi)t$, then $f = -\infty, \infty$ is the minimizer of $A(f)$. The desired results can follow with that $H_1(\pi, P(\boldsymbol{x})) = tA(\tilde{f})/\lim_{f\to-\infty} A(f)$ and $H_2(\pi, P(\boldsymbol{x})) = tA(\tilde{f})/\lim_{f\to\infty} A(f)$.

Derivation of the GACV for the RPLR: First, let $f_\lambda^{(-i)}(\cdot)$ is the solution of the RPLR with the ith data point omitted. Adopting the leaving-out-cone cross validation function $\mathrm{CV}(\lambda)=1/n\sum_{i=1}^{n}[-z_i f_\lambda^{(-i)}(\boldsymbol{x}_i)+\log(1+e^{f_\lambda(\boldsymbol{x}_i)})]$ for data from general exponential family in Xiang & Wahba (1996), we define CV(λ) for the RPLR

$$\mathrm{CV}(\lambda)=\frac{1}{n}\sum_{i=1}^{n}\min\left\{t, -z_i f_\lambda^{(-i)}(\boldsymbol{x}_i)+\log(1+e^{f_\lambda(\boldsymbol{x}_i)})\right\} \tag{28}$$

Since it is computationally expensive to calculate $f_\lambda^{(-i)}(\boldsymbol{x}_i)$, we approximate CV(λ) using formulae introduced in Xiang & Wahba (1996), and Liu (1995). Specifically, from (28), we have

$$\mathrm{CV}(\lambda)=\frac{1}{n}\sum_{i=1}^{n}\min\left\{t, -z_i f_\lambda(\boldsymbol{x}_i)+\log(1+e^{f_\lambda(\boldsymbol{x}_i)})+z_i(f_\lambda(\boldsymbol{x}_i)-f_\lambda^{(-i)}(\boldsymbol{x}_i))\right\}=\frac{1}{n}\sum_{i=1}^{n}\min\{t, a_i+b_i\} \tag{29}$$

where $a_i = -z_i f_\lambda(\boldsymbol{x}_i) + \log(1 + e^{f_\lambda(\boldsymbol{x}_i)})$ and $b_i = z_i(f_\lambda(\boldsymbol{x}_i) - f_\lambda^{(-i)}(\boldsymbol{x}_i))$. Define

$$d_i=\begin{cases} 1 & \text{if } t> \max(a_i+b_i, a_i) \\ 0 & \text{if } t< \min(a_i+b_i, a_i) \\ \frac{t-(a_i+b_i)}{-b_i} & \text{if } a_i+b_i<t<a_i \\ \frac{t-a_i}{b_i} & \text{if } a_i<t<a_i+b_i. \end{cases} \tag{30}$$

Note that $0 < d_i < 1$. Now (29) becomes

$$\begin{aligned}\mathrm{CV}(\lambda)&=\frac{1}{n}\sum_{i=1}^{n}\left[\min\left\{t, -z_i f_\lambda(\boldsymbol{x}_i)+\log(1+e^{f_\lambda(\boldsymbol{x}_i)})\right\}+d_i z_i(f_\lambda(\boldsymbol{x}_i)-f_\lambda^{(-i)}(\boldsymbol{x}_i))\right]\\ &=\frac{1}{n}\sum_{i=1}^{n}\min\left\{t, -z_i f_\lambda(\boldsymbol{x}_i)+\log(1+e^{f_\lambda(\boldsymbol{x}_i)})\right\}\\ &\quad+\frac{1}{n}\sum_{i=1}^{n}d_i z_i\frac{f_\lambda(\boldsymbol{x}_i)-f_\lambda^{(-i)}(\boldsymbol{x}_i)}{z_i-P_\lambda^{(-i)}(\boldsymbol{x}_i)}\\ &\quad\times\frac{z_i-P_\lambda(\boldsymbol{x}_i)}{1-\frac{P_\lambda(\boldsymbol{x}_i)-P_\lambda^{(-i)}(\boldsymbol{x}_i)}{z_i-P_\lambda^{(-i)}(\boldsymbol{x}_i)}}\end{aligned} \tag{31}$$

where $P_\lambda(\boldsymbol{x}_i) = 1/(1 + e^{-f_\lambda(\boldsymbol{x}_i)})$ and $P_\lambda^{(-i)}(\boldsymbol{x}_i) = 1/(1 + e^{-f_\lambda(-i)(\boldsymbol{x}_i)})$. Let $b(f_\lambda(\boldsymbol{x}_i)) = \log(1 + e^{f_\lambda(\boldsymbol{x}_i)})$. Since $b'(f_\lambda(\boldsymbol{x}_i)) = P_\lambda(\boldsymbol{x}_i)$ and $b''(f_\lambda(\boldsymbol{x}_i)) = P_\lambda(\boldsymbol{x}_i)(1 - P_\lambda(\boldsymbol{x}_i))$,

$$\frac{P_\lambda(\boldsymbol{x}_i)-P_\lambda^{(-i)}(\boldsymbol{x}_i)}{z_i-P_\lambda^{(-i)}(\boldsymbol{x}_i)}=\frac{b'(f_\lambda(\boldsymbol{x}_i))-b'(f_\lambda^{(-i)}(\boldsymbol{x}_i))}{z_i-P_\lambda^{(-i)}(\boldsymbol{x}_i)}\approx b''(f_\lambda(\boldsymbol{x}_i))\frac{f_\lambda(\boldsymbol{x}_i)-f_\lambda^{(-i)}(\boldsymbol{x}_i)}{z_i-P_\lambda^{(-i)}(\boldsymbol{x}_i)}, \tag{32}$$

and (31) becomes

$$\mathrm{CV}(\lambda)=\frac{1}{n}\sum_{i=1}^{n}\min\ \left\{t,-z_i f_\lambda(\boldsymbol{x}_i)+\log(1+e^{f_\lambda(\boldsymbol{x}_i)})\right\}+\frac{1}{n}\sum_{i=1}^{n}d_i\frac{z_i(z_i-P_\lambda(\boldsymbol{x}_i))}{\frac{z_i-P_\lambda^{(-i)}(\boldsymbol{x}_i)}{f_\lambda(\boldsymbol{x}_i)-f_\lambda^{(-i)}(\boldsymbol{x}_i)}-P_\lambda(\boldsymbol{x}_i)(1-P_\lambda(\boldsymbol{x}_i))} \tag{33}$$

Now what is left is the calculation of $(z_i - P_\lambda^{(-i)}(\boldsymbol{x}_i))/(f_\lambda(\boldsymbol{x}_i) - f_\lambda^{(-i)}(\boldsymbol{x}_i))$. We modify the leaving-out-one lemma of Xiang & Wahba (1996), which is a generalized version of the leaving-out-one lemma of Craven & Wahba (1979).

Lemma 1 (*leaving-out-one lemma*). *Let* $-\tilde{l}(z_i, f(\boldsymbol{x}_i)) = \min\{t, -z_i f(\boldsymbol{x}_i) + \log(1 + e^{f(\boldsymbol{x}_i)})\}$ *and* $I_\lambda(f,z)= -\sum_{i=1}^{n}\tilde{l}(z_i, f(\boldsymbol{x}_i))+n\lambda J(f)$. *Suppose* $f^*(i, z^*, \cdot)$ *is the minimizer in* $\mathcal{H}$ *of* $I_\lambda(f, z^*)$, *where* $\boldsymbol{z}^* = (z_1, \ldots, z_{i-1}, z^*, z_{i+1}, \ldots, z_n)$. *Then,*

$$f^*(i, P_\lambda^{(-i)}(\boldsymbol{x}_i), \cdot)=f_\lambda^{(-i)}(\cdot)$$

where $f_\lambda^{(-i)}(\cdot)$ *is the minimizer of* $-\sum_{j\neq i}\tilde{l}(z_j, f(\boldsymbol{x}_j)) + n\lambda J(f)$, *and* $P_\lambda^{(-i)}(\boldsymbol{x}) = 1/(1 + e^{-f_\lambda^{(-i)}(\boldsymbol{x})})$.

Proof of Lemma 1. Let $\boldsymbol{z}^{(-i)} = (z_1, \ldots, z_{i-1}, P_\lambda^{(-i)}(\boldsymbol{x}_i), z_{i+1}, \ldots, z_n)^{\mathrm{T}}$, and $-\tilde{l}^*(z, \tau) = -z\tau + \log(1 + e^{\tau})$. Since $-(\partial\tilde{l}^*(z, \tau))/\partial\tau = -z + 1/(1 + e^{-\tau})$ and $-(\partial^2\tilde{l}^*(z, \tau))/\partial\tau^2 = e^{\tau}/(1 + e^{\tau})^2 0$, for any fixed z, the minimizer of $-\tilde{l}^*(z, \tau)$ is τ which satisfies $z = 1/(1 + e^{-\tau})$. Therefore, using $P_\lambda^{(-i)}(\boldsymbol{x}_i) = 1/(1 + e^{-f_\lambda^{(-i)}(\boldsymbol{x}_i)})$, we have $-\tilde{l}^*(P_\lambda^{(-i)}(\boldsymbol{x}_i), f_\lambda^{(-i)}(\boldsymbol{x}_i)) - \tilde{l}^*(P_\lambda^{(-i)}(\boldsymbol{x}_i), f_\lambda(\boldsymbol{x}_i))$. This implies

$$-\tilde{l}(P_\lambda^{(-i)}(\boldsymbol{x}_i), f_\lambda^{(-i)}(\boldsymbol{x}_i)) \leq -\tilde{l}(P_\lambda^{(-i)}(\boldsymbol{x}_i), f_\lambda(x_i)) \tag{34}$$

since $-\tilde{l}(z_i, f(\boldsymbol{x}_i)) = \min\{t, -\tilde{l}^*(z_i, f(\boldsymbol{x}_i))\}$. Hence, for any $\boldsymbol{f}$, we have

$$
\begin{aligned}
I_\lambda(\boldsymbol{f}, \boldsymbol{z}^{(-i)}) = & -\tilde{l}(P_\lambda^{(-i)}(\boldsymbol{x}_i), \\
& f(\boldsymbol{x}_i)) \\
& -\sum\nolimits_{j\neq i}\tilde{l}(z_j, \\
& f(\boldsymbol{x}_j)) \\
& +n\lambda J(f) \geq \\
& -\tilde{l}(P_\lambda^{(-i)}(\boldsymbol{x}_i), \\
& f^{(-i)}(\boldsymbol{x}_i)) \\
& -\sum\nolimits_{j\neq i}\tilde{l}(z_j, \\
& f(\boldsymbol{x}_j)) \\
& +n\lambda J(f) \geq \\
& -\tilde{l}(P_\lambda^{(-i)}(\boldsymbol{x}_i), \\
& f^{(-i)}(\boldsymbol{x}_i)) \\
& -\sum\nolimits_{j\neq i}\tilde{l}(z_j, \\
& f_\lambda^{(-i)}(\boldsymbol{x}_j)) \\
& +n\lambda J(f_\lambda^{(-i)})
\end{aligned}
$$

using (34) and the definition of $f_\lambda^{(-i)}$. Therefore, we have $f^*(i, P_\lambda^{(-i)}(\boldsymbol{x}_i), \cdot) = f_\lambda^{(-i)}(\cdot)$.

Now let $\boldsymbol{f}_\lambda = (f_\lambda(\boldsymbol{x}_1), \ldots, f_\lambda(\boldsymbol{x}_n))^{\mathrm{T}}$, $\boldsymbol{f}_\lambda^{(-i)} = (f_\lambda^{(-i)}(\boldsymbol{x}_1), \ldots, f_\lambda^{(-i)}(\boldsymbol{x}_n))^{\mathrm{T}}$, $\boldsymbol{z} = (z_1, \ldots, z_n)^{\mathrm{T}}$, and $\boldsymbol{z}^{(-i)} = (z_1, \ldots, z_{i-1}, P_\lambda^{(-i)}(\boldsymbol{x}_i), z_{i+1}, \ldots, z_n)^{\mathrm{T}}$. By the definition of f_λ, $(f_\lambda, \boldsymbol{z})$ is a local minimizer of $I_\lambda(f, \boldsymbol{z}^*)$. Also, $(f_\lambda^{(-i)}, \boldsymbol{z}^{(-i)})$ is a local minimizer of $I_\lambda(f, \boldsymbol{z}^*)$ by Lemma 1. Therefore, $(\partial I_\lambda(f, \boldsymbol{z}^*))/\partial f(f_\lambda, \boldsymbol{z}) = 0$ and $(\partial I_\lambda(f, \boldsymbol{z}^*))/\partial f(f_\lambda^{(-i)}, \boldsymbol{z}^{(-i)}) = 0$. Writing $J(f) = \boldsymbol{f}^{\mathrm{T}}\Sigma\boldsymbol{f}$ gives $I_\lambda = \min\{t, -z_i f(\boldsymbol{x}_i) + \log(1 + e^{f(\boldsymbol{x}_i)})\} + n\lambda\boldsymbol{f}^{\mathrm{T}}\Sigma\boldsymbol{f}$ (see Section 3.1 of Xiang & Wahba (1996) for computation of Σ). Since $I_\lambda 0$ is not differentiable, we approximate it with a differentiable function

$$
I_\lambda^* = \sum_{i=1}^{n} g^*(f_i, z_i, \boldsymbol{x}_i) + n\lambda\boldsymbol{f}^{\mathrm{T}}\Sigma\boldsymbol{f} \tag{36}
$$

with

$$
g^*(f, z, \boldsymbol{x}) = \begin{cases} t & \text{if } yf < -\log(e^t - 1) - \varepsilon \\ g^{**}(f, z, \boldsymbol{x}) & \text{if } -\log(e^t - 1) - \varepsilon \leq yf \leq -\log(e^t - 1) + \delta \\ -zf + \log(1 + e^f) & \text{if } yf > -\log(e^t - 1) + \delta(\varepsilon) \end{cases} \tag{37}
$$

where g^{**} is a quadratic function of f which makes g^* differentiable in f. Note that $I_\lambda^* \to I_\lambda$ as $\varepsilon \to 0$. Let σ_{ij} be the ijth element of Σ. Then,

$$
\frac{\partial I_\lambda^*}{\partial f(\boldsymbol{x}_i)} \xrightarrow{\varepsilon \to 0} \begin{cases} -z_i + 1/(1 + e^{-f(\boldsymbol{x}_i)}) + n\lambda\sum_j \sigma_{ij} f(\boldsymbol{x}_i) & \text{if } z_i f(\boldsymbol{x}_i) \geq -\log(e^t - 1) \\ n\lambda\sum_j \sigma_{ij} f(\boldsymbol{x}_i) & \text{otherwise} \end{cases} \tag{38}
$$

and

$$\frac{\partial^2 I_\lambda^*}{\partial f(\boldsymbol{x}_i)\partial f(\boldsymbol{x}_j)} \xrightarrow{\varepsilon\to 0} \begin{cases} n\lambda \sum_j \sigma_{ii} + I_{\{z_i f(\boldsymbol{x}_i) - \log(e^t - 1)\}} \frac{e^{f(\boldsymbol{x}_i)}}{(1+e^{f(\boldsymbol{x}_i)})^2} & \text{if } i=j \\ n\lambda \sum_j \sigma_{ij} & \text{if } i \neq j. \end{cases} \tag{39}$$

Therefore, defining

$$W(f) = \operatorname{diag}\left(I_{\{z_1 f(\boldsymbol{x}_1) \geq -\log(e^t-1)\}} \frac{e^{f(x_1)}}{(1+e^{f(\boldsymbol{x}_1)})^2}, \ldots, I_{\{z_n f(\boldsymbol{x}_n) - \log(e^t-1)\}} \frac{e^{f(\boldsymbol{x}_n)}}{(1+e^{f(\boldsymbol{x}_n)})^2}\right),$$

we have

$$\frac{\partial^2 I_\lambda^*}{\partial \boldsymbol{f} \partial \boldsymbol{f}^{\mathrm{T}}} \xrightarrow{\varepsilon\to 0} W + n\lambda\Sigma, \text{ and } \frac{\partial^2 I_\lambda^*}{\partial z \partial \boldsymbol{f}^{\mathrm{T}}} \xrightarrow{\varepsilon\to 0} -I.$$

Using Taylor expansion,

$$\begin{aligned} 0 = \frac{\partial I_\lambda^*}{\partial \boldsymbol{f}}(\boldsymbol{f}_\lambda{}^{(-i)}, & \\ & \boldsymbol{z}^{(-i)}) \\ & = \frac{\partial I_\lambda^*}{\partial \boldsymbol{f}}(\boldsymbol{f}_\lambda, \\ & \boldsymbol{z}) + \frac{\partial^2 I_\lambda^*}{\partial \boldsymbol{f} \partial \boldsymbol{f}^T}(\boldsymbol{f}_\lambda{}^{**}, \\ & \boldsymbol{z}^{**})(\boldsymbol{f}_\lambda{}^{(-i)} \\ & - \boldsymbol{f}_\lambda) \\ & + \frac{\partial^2 I_\lambda^*}{\partial z \partial \boldsymbol{f}^T}(\boldsymbol{f}_\lambda{}^{**}, z^{**})(\boldsymbol{z} \\ & - \boldsymbol{z}^{(-i)}) \xrightarrow{\varepsilon\to 0} 0 \\ & + \{W(\boldsymbol{f}^{**}) + n\lambda\Sigma\}(\boldsymbol{f}_\lambda{}^{(-i)} \\ & - \boldsymbol{f}_\lambda) \\ & - (\boldsymbol{z} - \boldsymbol{z}^{(-i)}) \end{aligned} \tag{40}$$

where $(\boldsymbol{f}_\lambda{}^{**}, \boldsymbol{z}^{**})$ is a point somewhere between $(\boldsymbol{f}_\lambda, \boldsymbol{z})$ and $(\boldsymbol{f}_\lambda{}^{(-i)}, \boldsymbol{z}^{(-i)})$. Approximating $W(\boldsymbol{f}_\lambda{}^{**})$ by $W(\boldsymbol{f}_\lambda)$ and letting $\varepsilon \to 0$ gives $\boldsymbol{f}_\lambda - \boldsymbol{f}_\lambda{}^{(-i)} = \{W(\boldsymbol{f}_\lambda{}^{**}) + n\lambda\Sigma\}^{-1}(\boldsymbol{z} - \boldsymbol{z}^{(-i)})$, that is,

$$\begin{pmatrix} f_\lambda(\boldsymbol{x}_1) - f_\lambda{}^{(-i)}(\boldsymbol{x}_1) \\ \vdots \\ f_\lambda(\boldsymbol{x}_i) - f_\lambda{}^{(-i)}(\boldsymbol{x}_i) \\ \vdots \\ f_\lambda(\boldsymbol{x}_n) - f_\lambda{}^{(-i)}(\boldsymbol{x}_n) \end{pmatrix} \simeq \{W(\boldsymbol{f}_\lambda{}^{**}) + n\lambda\Sigma\}^{-1} \begin{pmatrix} 0 \\ \vdots \\ z_i - P_\lambda{}^{(-i)}(\boldsymbol{x}_i) \\ \vdots \\ 0 \end{pmatrix} \tag{41}$$

Let $H = \{W(\boldsymbol{f}_\lambda) + n\lambda\Sigma\}^{-1}$ and h_{ii} be the ith diagonal entry of H. Then (41) implies

$$\frac{f_\lambda(\boldsymbol{x}_i) - f_\lambda^{(-i)}(\boldsymbol{x}_i)}{z_i - P_\lambda^{(-i)}(\boldsymbol{x}_i)} \simeq h_{ii} \tag{42}$$

Using (42), (33) becomes

$$\frac{1}{n}\sum_{i=1}^{n}\min\left\{t, -z_i f_\lambda(\boldsymbol{x}_i)+\log(1+\mathrm{e}^{f_\lambda(\boldsymbol{x}_i)})\right\}+\frac{1}{n}\sum_{i=1}^{n} d_i \frac{h_{ii} z_i (Z_i - P_\lambda(\boldsymbol{x}_i))}{1 - h_{ii} P_\lambda(\boldsymbol{x}_i)(1 - P_\lambda(\boldsymbol{x}_i))} \tag{43}$$

Replacing h_{ii} by tr(H)/n and replacing $h_{ii}P_\lambda(\boldsymbol{x}_i)(1 - P_\lambda(\boldsymbol{x}_i))$ by tr($W^{*1/2}HW^{*1/2}$)/n with

$$W^{*}=\mathrm{diag}\left(\frac{\mathrm{e}^{f(\boldsymbol{x}_1)}}{(1+\mathrm{e}^{f(\boldsymbol{x}_1)})^2},\ldots,\frac{\mathrm{e}^{f(\boldsymbol{x}_n)}}{(1+\mathrm{e}^{f(\boldsymbol{x}_n)})^2}\right),$$

we define

$$\mathrm{GACV}(\lambda)=\frac{1}{n}\sum_{i=1}^{n}\min\left\{t, -z_i f_\lambda(\boldsymbol{x}_i)+\log(1+\mathrm{e}^{f_\lambda(\boldsymbol{x}_i)})\right\}+\frac{\mathrm{tr}(H)}{n}\sum_{i=1}^{n} d_i \frac{h_{ii} z_i (z_i - P_\lambda(\boldsymbol{x}_i))}{n - \mathrm{tr}(W^{*1/2}HW^{*1/2})} \tag{44}$$

BIBLIOGRAPHY

An LTH, Tao PD. Solving a class of linearly constrained indefinite quadratic problems by d.c. algorithms. Journal of Global Optimization. 1997; 11:253–285.

Bartlett P, Jordan M, McAuliffe J. Convexity, classification, and risk bounds. Journal of the American Statistical Association. 2006; 101:138–156.

Bhattacharjee A, Richards WG, Staunton J, Li C, Monti S, Vasa P, Ladd C, Beheshti J, Bueno R, Gillette M, Loda M, Weber G, Mark EJ, Lander ES, Wong W, Johnson BE, Golub TR, Sugarbaker DJ, Meyerson M. Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses. Proceedings of the National Academy of Sciences of the United States of America. 2001; 98:13790–13795. [PubMed: 11707567]

Bianco, AM.; Yohai, VJ. Robust estimation in the logistic regression model. In: Rieder, H., editor. Robust Statistics, Data Analysis, and Computer Intensive Methods, Volume 109 of Lecture Notes in Statistics. New York: Springer-Verlag; 1996.

Bondell H. Minimum distance estimation for the logistic regression model. Biometrika. 2005; 92:724–731.

Cantoni E, Ronchetti E. Resistant selection of the smoothing parameter for smoothing splines. Statistics and Computing. 2001a; 11:141–146.

Cantoni E, Ronchetti E. Robust inference for generalized linear models. Journal of the American Statistical Association. 2001b; 96:1022–1030.

Carroll RJ, Pederson S. On robustness in the logistic regression model. Journal of the Royal Statistical Society, Series B (Methodological). 1993; 55:693–706.

Copas JB. Binary regression models for contaminated data (with discussion). Journal of the Royal Statistical Society, Series B (Methodological). 1988; 50:225–265.

Craven P, Wahba G. Smoothing noisy data with spline functions: Estimating the correct degree of smoothing by the method of generalized cross-validation. Numerische Mathematik. 1979; 31:377–403.

Croux C, Haesbroeck G. Implementing the bianco and yohai estimator for logistic regression. Computational Statistics and Data Analysis. 2003; 44:273–295.

Dudoit S, Fridly J, Speed TP. Comparison of discrimination methods for the classification of tumors using gene expression data. Journal of the American Statistical Association. 2002; 97:77–87.

Fan J, Li R. Variable selection via nonconcave penalized likelihood and it oracle properties. Journal of American Statistical Association. 2001; 96:1348–1360.

Golub T, Slonim D, Tamayo P, Huard C, Gaasenbeek M, Mesirov J, Coller H, Loh M, Downing J, Caligiuri M. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science. 1999; 286:531–537. [PubMed: 10521349]

Hunter D, Li R. Variable selection using mm algorithms. The Annals of Statistics. 2005; 33:1617–1642.

Kimeldorf G, Wahba G. Some results on Tchebycheffian spline functions. Journal of Mathematical Analysis and Applications. 1971; 33:82–95.

Krasker WS, Welsch RE. Efficient bounded-influence regression estimation. Journal of the American Statistical Association. 1982; 77:595–604.

Künsch HR, Stefanski LA, Carroll RJ. Conditionally unbiased bounded-influence estimation in general regression models, with applications to generalized linear models. Journal of the American Statistical Association. 1989; 84:460–466.

le Cessie S, van Houwelingen JC. Ridge estimators in logistic regression. Applied Statistics. 1992; 41:191–201.

Lin X, Wahba G, Xiang D, Gao F, Klein R, Klein B. Smoothing spline ANOVA models for large data sets with Bernoulli observations and the randomized GACV. The Annals of Statistics. 2000; 28:1570–1600.

Lin Y. Support vector machines and the bayes rule in classification. Data Mining and Knowledge Discovery. 2002; 6:259–275.

Lin Y. A note on margin-based loss functions in classification. Statistics and Probability Letters. 2004; 68:73–82.

Lin Y, Lee Y, Wahba G. Support vector machines for classification in nonstandard situations. Machine Learning. 2002; 46:191–202.

Lin Y, Zhang HH. Component selection and smoothing in smoothing spline analysis of variance models—Cosso. Annals of Statistics. 2006; 34:2272–2297.

Liu Y. Unbiased estimate of generalization error and model selection in neural network. Neural Networks. 1995; 8(2):215–219.

Liu Y, Hayes DN, Nobel A, Marron JS. Statistical significance of clustering for high dimension low sample size data. Journal of the American Statistical Association. 2008; 103:1281–1293.

Liu Y, Shen X. Multicategory ψ-learning. Journal of the American Statistical Association. 2006; 101:500–509.

Liu Y, Shen X, Doss H. Multicategory ψ-learning and support vector machine: Computational tools. Journal of Computation and Graphical Statistics. 2005; 14:219–236.

McCullagh, P.; Nelder, J. Generalized Linear Models. London: Chapman & Hall/CRC; 1989.

Morgenthaler S. Least-absolute-deviations fits for generalized linear models. Biometrika. 1992; 79:747–754.

Pregibon D. Resistant fits for some commonly used logistic models with medical applications. Biometrics. 1982; 38:485–498. [PubMed: 7115876]

Shen X, Tseng G, Zhang X, Wong W. On ψ-learning. Journal of the American Statistical Association. 2003; 98:724–734.

Stefanski LA, Carroll RJ, Ruppert D. Optimally hounded score functions for generalized linear models with applications to logistic regression. Biometrika. 1986; 73:413–424.

Tibshirani R. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society, Series B. 1996; 58:267–288.

Wahba, G. Support vector machines, reproducing kernel hilbert spaces and the randomized GACV. In: Bernhard, S.; Burges, CJS.; Smola, AJ., editors. Advances in Kernel Methods Support Vector Learning. Cambridge, MA: MIT Press; 1999. p. 69-88.

Wang J, Shen X, Liu Y. Probability estimation for large margin classifiers. Biometrika. 2007; 95:149–167.

Wu Y, Liu Y. Robust truncated hinge loss support vector machines. Journal of the American Statistical Association. 2007; 102:974–983.

Xiang D, Wahba G. A generalized approximate cross validation for smoothing splines with non-gaussian data. Statistica Sinica. 1996; 6:675–692.

Zhu J, Hastie T. Kernel logistic regression and the import vector machine. Journal of Computational and Graphical Statistics. 2005; 14:185–205.

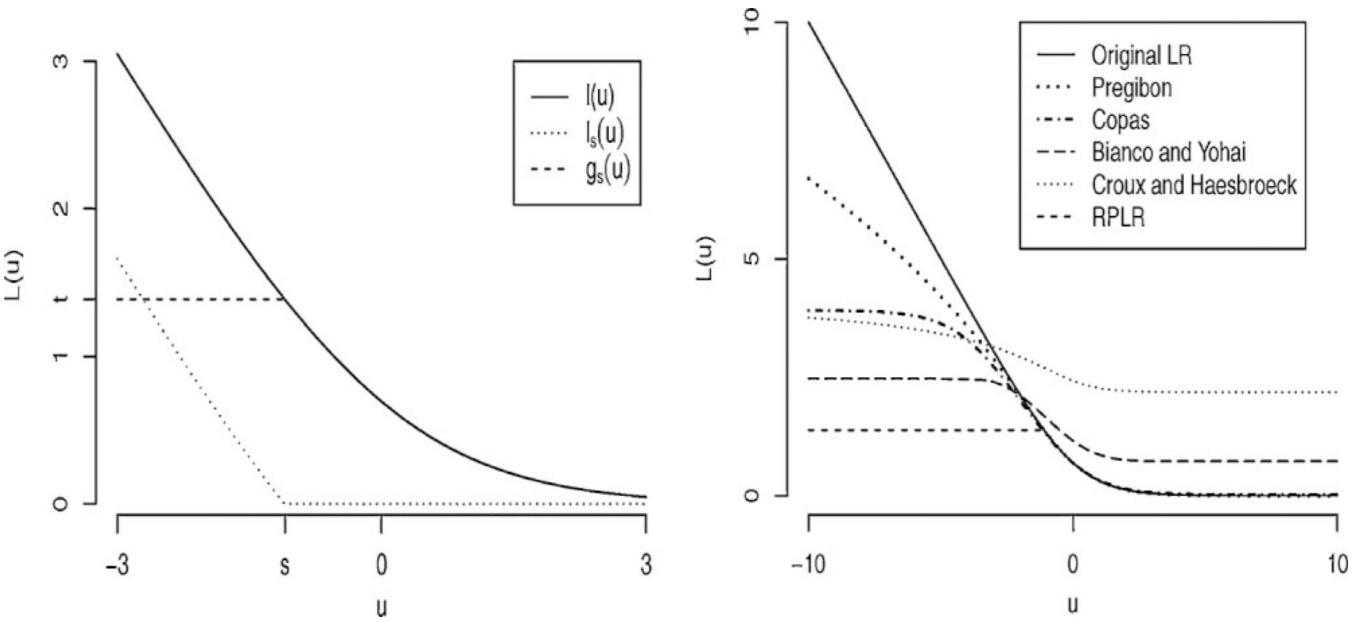

Figure 1.
Left: Plot of the functions $l(u)$, $l_s(u)$, and $g_s(u)$ with $l_s(u) = [l(u) - l(s)]_+$ and $g_s(u) = l(u) - l_s(u)$. Right: Plot of the loss functions of the original logistic regression, Pregibon's resistant fitting model, Copas' misclassification model, Bianco and Yohai's robust logistic regression, Croux and Haesbroeck's robust logistic regression and the proposed RPLR.

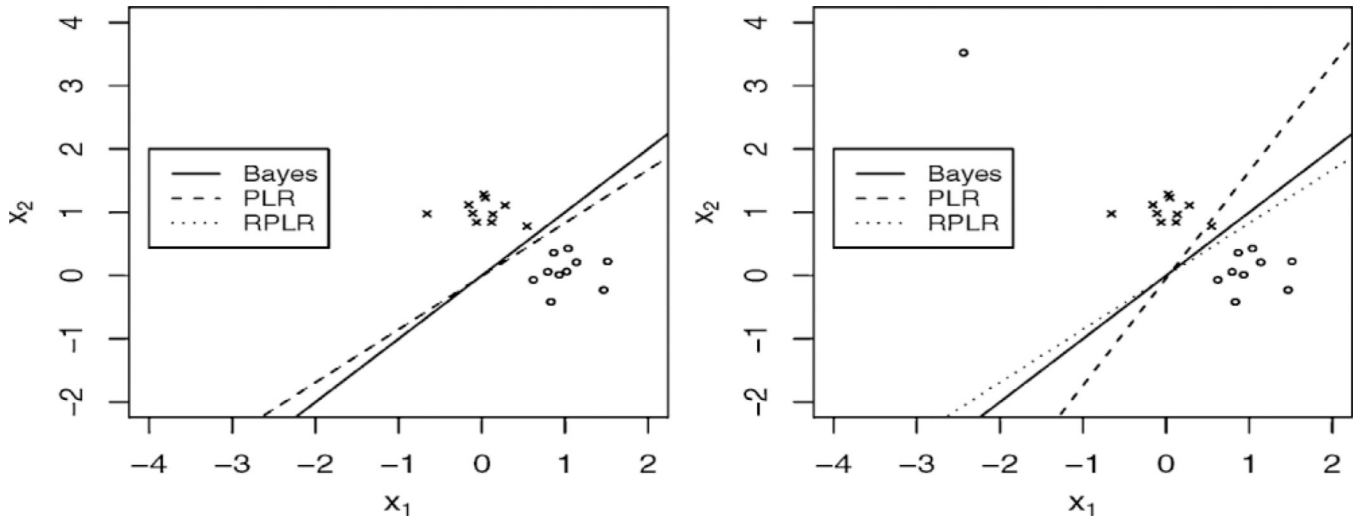

Figure 2.

Illustration plot of the effect of outliers with an outlier far away from its own class. The RPLR boundary is more robust than that of the original PLR. Note that on the left panel, the decision boundaries of the PLR and RPLR are identical.

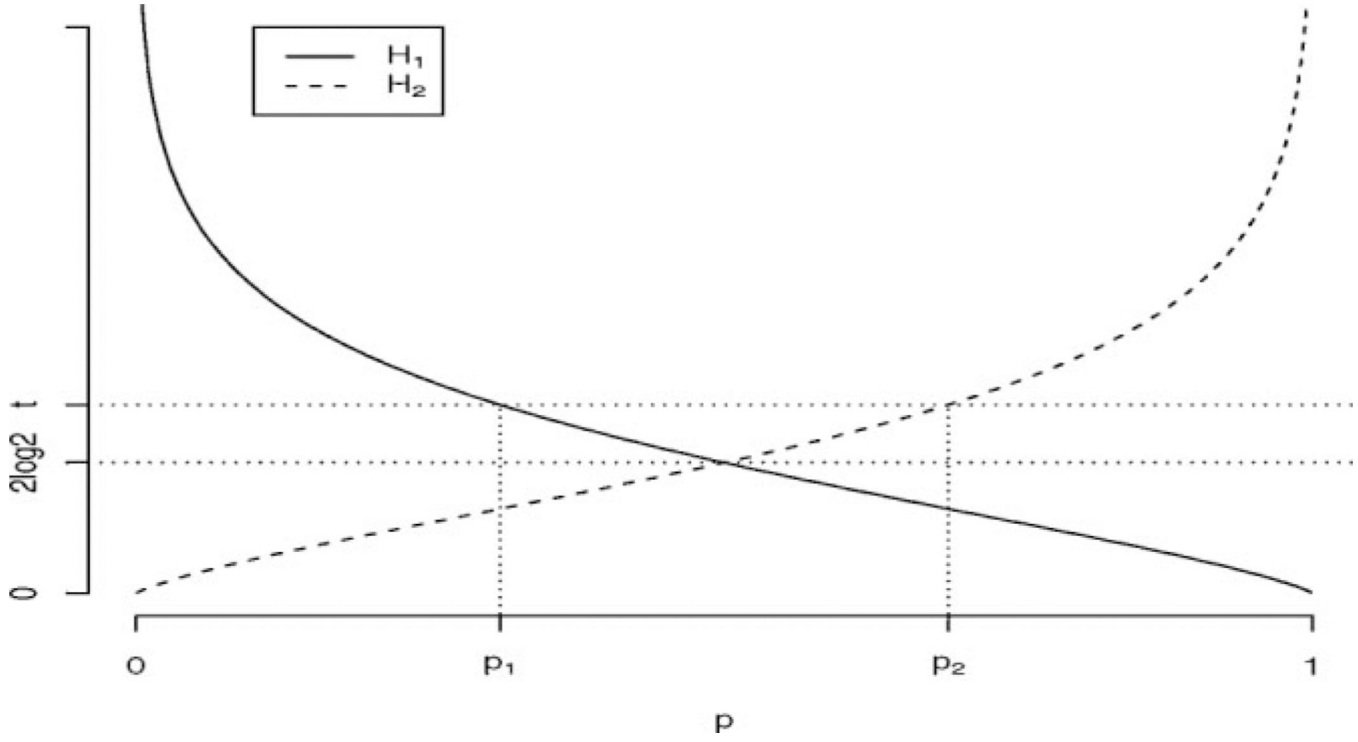

Figure 3.
Plot of H_1 and H_2 for Theorem 2 in Section 4.3. The conditions $t > H_1(\pi, p)$ and $t > H_2(\pi, p)$ only hold when $p \in [p_1, p_2]$.

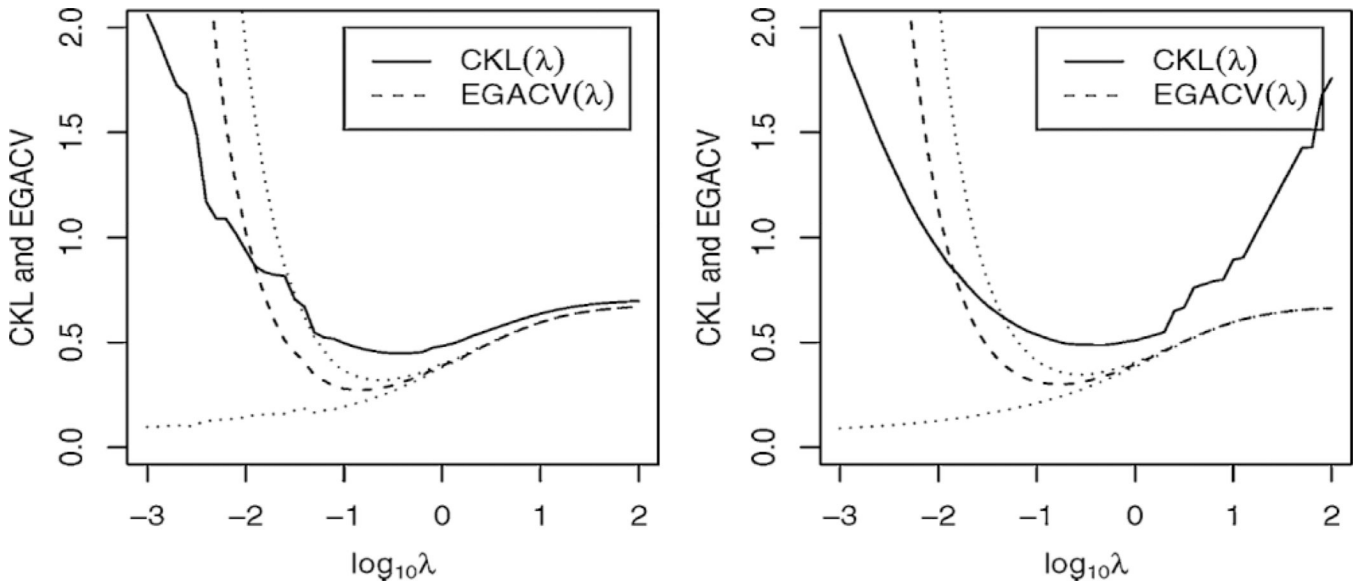

Figure 4.
Left: An illustration plot of CKL(λ) and EGACV(λ) from the example in Section 6. Right: Average curves of CKL(λ) and EGACV(λ) based on 100 replications.

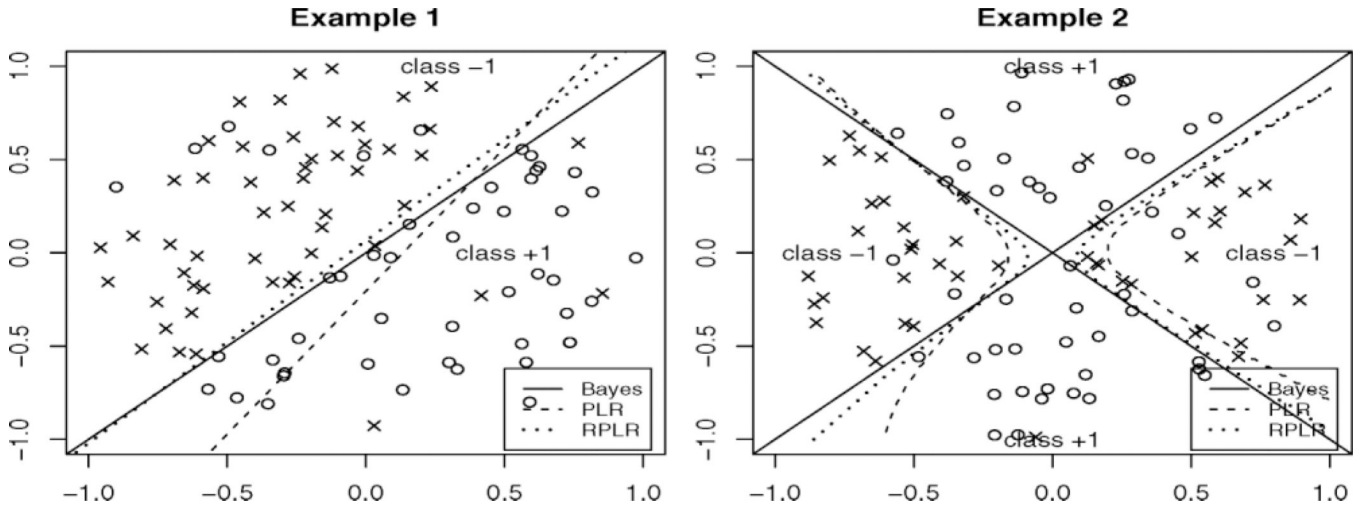

Figure 5.
Plot of typical training sets for Example 1 (the left panel) and Example 2 (the right panel) as well as the corresponding decision boundaries.

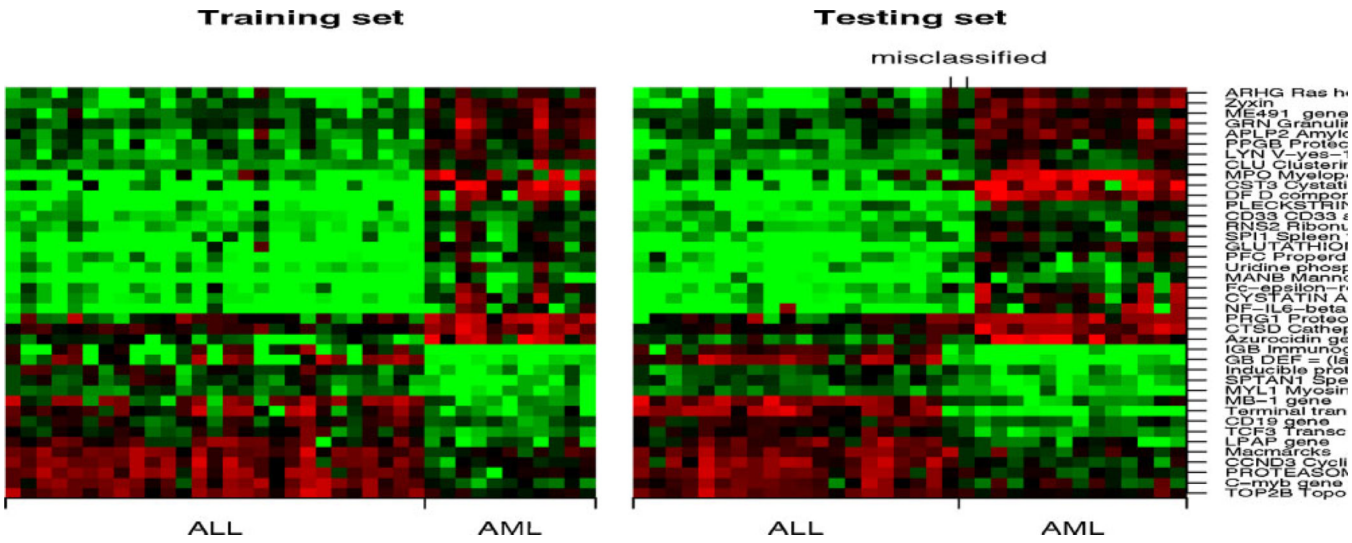

Figure 6.

Heat maps of the leukaemia data in Section 7.2.1. The left panel is for the training set and the right panel is for the testing set. The red and green colors of the online version represent high and low expression values, respectively. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com]

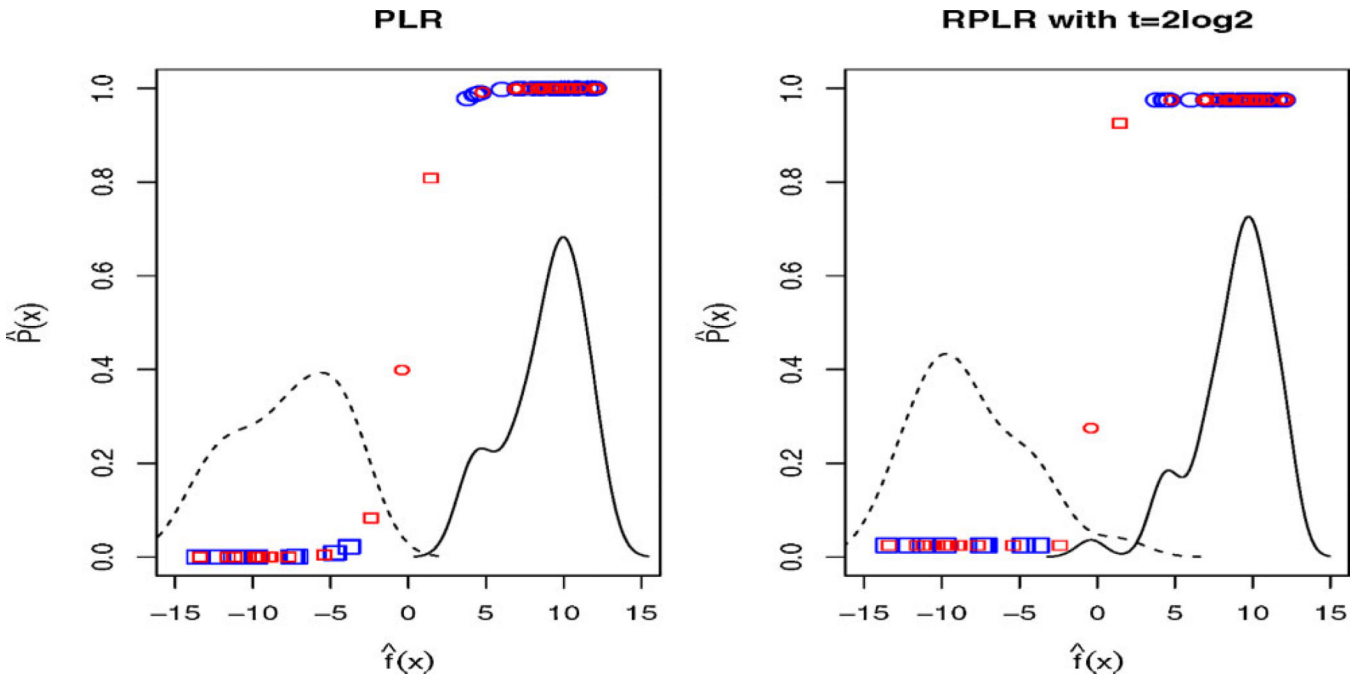

Figure 7.
Plot of the estimated class probabilities against the estimated values of the linear predictor $f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$ for the PLR and the RPLR with $t = 2 \log 2$. The solid and the dashed lines are the estimated density curves of the values of linear predictor for ALL and AML class, respectively. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com]

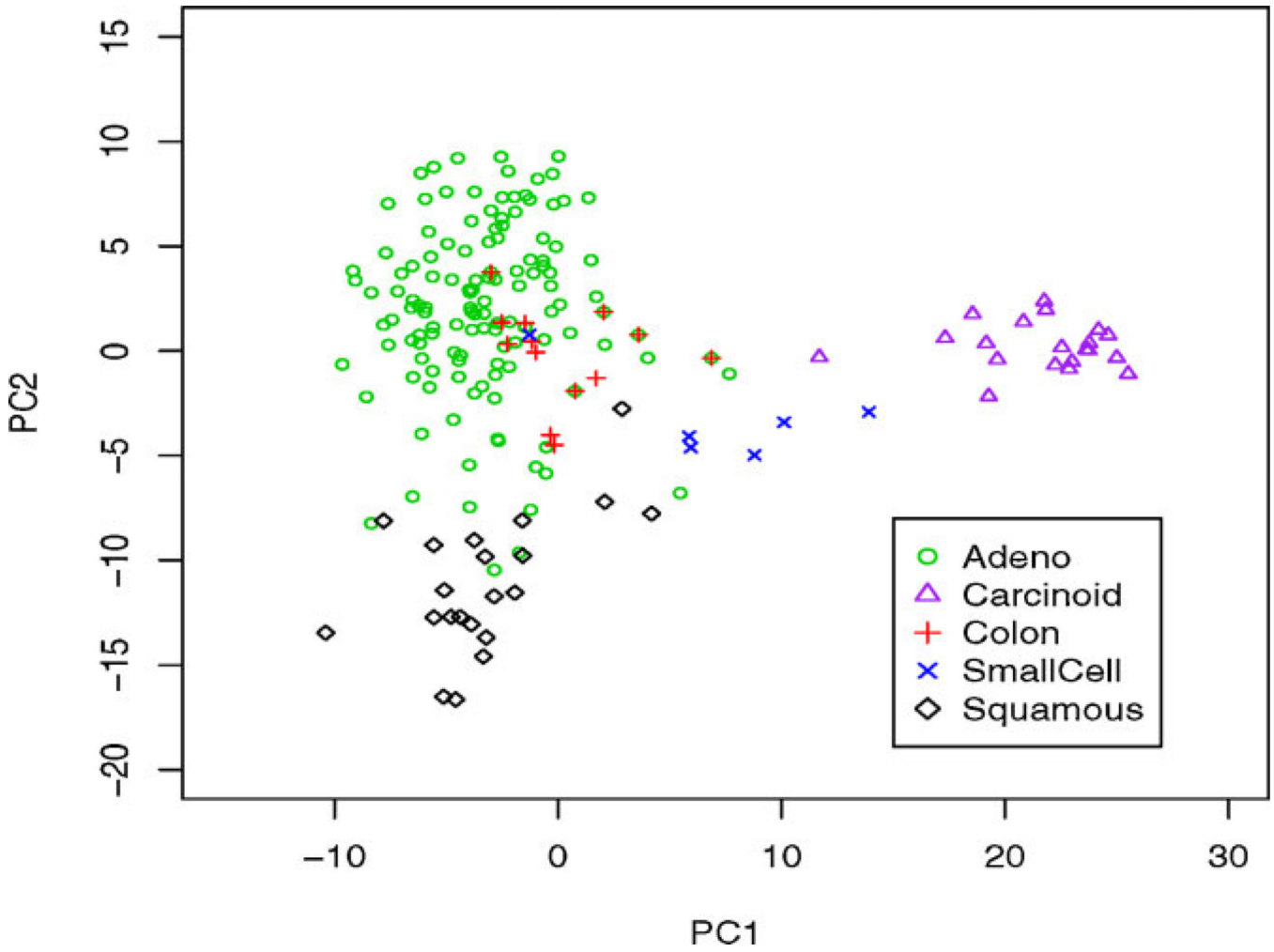

Figure 8.
Biplot on PCA of the lung cancer data in Section 7.2.2. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com]

Table 1

Testing errors of the simulated linear example (Example 1) in Section 7.1.

Method	$v = 0$	$v = 5$	$v = 10$	$v = 20$
PLR	0.0090 (0.0006)	0.0726 (0.0014)	0.1348 (0.0021)	0.2371 (0.0022)
RPLR (I)				
$t = 3 \log 2$	0.0061 (0.0005)	0.0606 (0.0009)	0.1172 (0.0015)	0.2271 (0.0022)
$t = 2 \log 2$	0.0090 (0.0006)	0.0613 (0.0008)	0.1161 (0.0012)	0.2198 (0.0017)
$t = \log 2$	0.0120 (0.0008)	0.0663 (0.0011)	0.1215 (0.0015)	0.2248 (0.0018)
Tuned	0.0097 (0.0007)	0.0612 (0.0008)	0.1150 (0.0011)	0.2205 (0.0016)
RPLR (II)				
$t = 3 \log 2$	0.0187 (0.0011)	0.0714 (0.0012)	0.1280 (0.0018)	0.2378 (0.0033)
$t = 2 \log 2$	0.0188 (0.0012)	0.0688 (0.0013)	0.1222 (0.0015)	0.2288 (0.0033)
$t = \log 2$	0.0306 (0.0019)	0.0782 (0.0046)	0.1301 (0.0042)	0.2447 (0.0067)
Croux and Haesbroeck	0.0104 (0.0009)	0.0658 (0.0010)	0.1286 (0.0019)	0.2335 (0.0021)
Bayes error	0.00	0.05	0.10	0.20

Here RPLR (I) and RPLR (II) refer to the RPLR results using the tuning set and EGACV for tuning parameter selection, respectively.

Table 2

Class probability estimation errors of the simulated linear example (Example 1) in Section 7.1.

Method	Scheme	$v = 0$	$v = 5$	$v = 10$	$v = 20$
PLR		0.0464 (0.0060)	0.1342 (0.0062)	0.1487 (0.0046)	0.1350 (0.0035)
RPLR (I)					
$t = 3 \log 2$	1	0.0207 (0.0049)	0.1101 (0.0029)	0.1350 (0.0029)	0.1289 (0.0030)
	2	0.0173 (0.0039)	0.0994 (0.0027)	0.1236 (0.0033)	0.1270 (0.0032)
	3	0.0438 (0.0037)	0.0686 (0.0034)	0.1022 (0.0041)	0.1184 (0.0041)
$t = 2 \log 2$	3	0.0614 (0.0050)	0.0676 (0.0032)	0.0934 (0.0035)	0.1053 (0.0041)
$t = \log 2$	3	0.0758 (0.0073)	0.0887 (0.0079)	0.1057 (0.0059)	0.1185 (0.0040)
RPLR (II)					
$t = 3 \log 2$	1	0.1152 (0.0008)	0.1248 (0.0015)	0.1323 (0.0015)	0.1279 (0.0026)
	2	0.0861 (0.0007)	0.1034 (0.0017)	0.1208 (0.0021)	0.1254 (0.0027)
	3	0.1053 (0.0010)	0.0975 (0.0019)	0.1084 (0.0028)	0.1230 (0.0040)
$t = 2 \log 2$	3	0.1193 (0.0011)	0.0982 (0.0028)	0.1054 (0.0026)	0.1053 (0.0034)
$t = \log 2$	3	0.1707 (0.0028)	0.1127 (0.0065)	0.1096 (0.0046)	0.1251 (0.0056)
Croux and Haesbroeck		0.0104 (0.0009)	0.0865 (0.0015)	0.1208 (0.0012)	0.1238 (0.0015)

Table 3

Testing errors of the lung cancer data example in Section 7.2.2.

Method	Testing Error
PLR	0.1274 (0.0052)
RPLR	
$t = 3 \log 2$	0.1242 (0.0051)
$t = 2 \log 2$	0.1210 (0.0046)
$t = \log 2$	0.1226 (0.0054)