Improving Boosting Methods With A Stable Loss Function Handling Outliers

WangChao (WangChaoAnHui@126.com)
LiBo
WangLei
PengPai

Research Article

Keywords: Boosting Methods, Robust Classification, Ensemble Learning

Posted Date: March 3rd, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1402125/v1

Improving Boosting Methods With A Stable Loss Function Handling Outliers

Wang Chao [1], Li Bo [1], Wang Lei [1], Peng Pai [1]

[1] Department of Mathematic and Statistics, Central China Normal University, Wuhan, Hubei, China

---

Address for correspondence: Wang Chao, Department of Mathematic and Statistics, Central China Normal University, Wuhan, Hubei, China.

E-mail: WangChaoAnHui@126.com.

Phone: (+86) 132 9663 3780.

Fax: (+1) 999 888 666.

Abstract: In classification problems, the occurrence of abnormal observations is often encountered. How to obtain a stable model to deal with outliers has always been a subject of widespread concern. In this article, we draw on the ideas of the AdaBoosting algorithm and propose a asymptotically linear loss function, which makes the output function more stable for contaminated samples, and two boosting algorithms were designed, based on two different way of updating, to handle outliers. In addition, a skill for overcoming the instability of Newton's method when dealing with weak convexity is introduced. Several samples, where outliers were artificially added, show that the Discrete L-AdaBoost and Real L-AdaBoost Algorithms find the boundary of each category consistently under the condition where data is contaminated. In

real world data sets, we show the effectiveness of suggested algorithms by comparison with other two ensemble learning methods, especially for large dataset.

Key words: Boosting Methods; Robust Classification; Ensemble Learning

1 Introduction

For the classification problem, we learn from Bayes theorem that all we need is $P(y = j|x)$, the posterior or conditional class probabilities. One could transfer many regression machinery across to the classification domain by simply noting that $E(1_{[y=j]}|x) = P(y = j|x)$, where $1_{[y=j]}$ is the 0/1 indicator variable representing class j. A celebrated example is $E(1_{[y=j]}|x) = \beta_j^T x$, where linear regression is used for estimation and the response confined to $[0, 1]$ is ignored. While this works fairly well in general, several problems have been noted for constrained regression problems (Hastie et al., 1994). One popular approach used in statistics for overcoming these problems is logit transformation. For a two-class problem, where $y \in \{-1, 1\}$, a logistic model has the form

$$\log \frac{P(y = 1|x)}{P(y = -1|x)} = F(x) \tag{1.1}$$

The monotone *logit* transformation on the left guarantees that for any values of $F(x) \in \mathbb{R}$, the probability estimation lies in $[0, 1]$. Therefore, many loss functions dealing with classification problems have been proposed. The exponential criterion $E[e^{-yF(x)}]$, which appeared in (Friedman et al., 2000; Pu and Rao, 2018; Gao et al., 2019), is often used to solve classification problems. In this paper, a robust criterion is suggested for dealing with long-tail error outliers.

Robust estimation is often desirable in the presence of outliers. Outlier observations will bring severe challenges to parameter estimation and function estimation. Many robust methods are designed to deal with exceptional situations(Kong et al., 2018; Wang et al., 2020, 2019). One of the robust approaches is the so-called M-estimation approach (Huber, 1972), which relies on minimizing a dispersion function that is slowly varying instead of the squared residuals and the latest development research can be found in Sun et al. (2020). From the part of Comparison, we will see that M-estimation is not suitable for classifications. We found that the negative binomial log-likelihood not only can effectively deal with the classification problem, whose population solution coincident with model (1.1),but also retains the robustness of M-estimation.

We consider the following stochastic optimization problems:

$$\underset{F(\cdot)}{minimize} \quad J(F) := E\left[\log\left(1+e^{yF(x)}\right) - yF(x)\right], \tag{1.2}$$

where, $F : \mathbb{R}^p \to \mathbb{R}$ is a function to be evaluated, $y \in \{-1, 1\}$ denotes the response variable, $x \in \mathcal{X} \subset \mathbb{R}^p$ denotes the predictor (or covariate), and the expectation is over the joint distribution of (y, x). The above minimization is commonly encountered in statistical learning. When $F(x)$ is assumed to be a linear combination of x, as $F(x) = \beta^T x$, the problem turns into logical regression. But in this paper, we don't make any assumptions about the form of $F(x)$, except for additivity, that is

$$F(x) = f_1(x) + f_2(x) + \cdots + f_M(x), M \in \mathbb{N}^+.$$

Generalized additive models (GAMs) is a very popular function estimation model for functional estimation(Daraghmi et al., 2014; Lv et al., 2017; Rana et al., 2018). With the assumption of additivity, estimating $F(x)$ is equivalent to estimating $f_i(x)$. Before giving the estimation of $F(x)$, we shall explain what $F(x)$ stands for or the

significance of solving the minimization (1.2). We have Lemma 1.1.

Lemma 1.1 *Let $F^{pop}(x)$ denote the true minimizer of the population risk given in (1.2), then we have*

$$P(y=1|x) = \frac{e^{F^{pop}(x)}}{1+e^{F^{pop}(x)}} \tag{1.3}$$

$$F^{pop}(x) = \log \frac{P(y=1|x)}{P(y=-1|x)} \tag{1.4}$$

Proof. Take first derivative of $J(F)$ on $F(x)$. For fixed x, $F^{pop}(x)$ shall satisfy

$$E\left[\frac{ye^{yF^{pop}(x)}}{1+e^{yF^{pop}(x)}} - y\middle| x\right] = 0.$$

As $y \in \{1, -1\}$, we get

$$\frac{e^{yF^{pop}(x)}}{1+e^{yF^{pop}(x)}}P(y=-1|x) - \frac{1}{1+e^{yF^{pop}(x)}}P(y=1|x) = 0.$$

Combined by $P(y=1|x) + P(y=-1|x) = 1$, we get (1.3) (1.4).

From Lemma 1.1, if we have an estimate of $F^{pop}(x)$, we could calculate $P(y=1|x)$ by (1.3). This gives the significance of solving the minimization (1.2), that is, providing us a method for classification problems. (i)First, get an estimate of $F^{pop}(x)$, denoted by $\hat{F}(x)$;(ii)Then calculate $P(y=1|x)$ by (1.3). In two-class problems, only $sign(\hat{F}(x))$ is needed, for $P(y=1|x) > 0.5$ is equivalent to $F^{pop}(x) > 0$. This knowledge will be used for the prediction of $y|x$ in the following algorithms.

The exact minimization of the stochastic optimization problem (1.2) requires the knowledge of the underlying distribution of the variables (y, x). In practice, however, the joint distribution of the pair is not available; therefore, after observing n independent data points (y_i, x_i), one standard approach is to minimize the following

surrogate of (1.2), often referred to as empirical risk approximation

$$\underset{F(\cdot)}{minimize} \quad J(F) := \frac{1}{n}\sum_{i=1}^{n}\left[\log(1+e^{y_i F(x_i)}) - y_i F(x_i)\right]. \tag{1.5}$$

In generalized linear models, $F(x)$ is assumed to be a linear combination of x. In practice, the true minimizer $F^{pop}(x)$ is nearly impossible to be linear and any parameterized form of $F(x)$ will greatly reduced the domain of $F(x)$. Therefore, in this paper, the boosting method is chosen to solve this additive model. Boosting, originally proposed by Schapire (1990) and Freund (1995), is devoted to producing a strong composite learner from a given class of weak learner. Some boosting algorithms can be interpreted from a viewpoint of statistical gradient descent to solve optimization problems with different loss functions (Friedman et al., 2000; Friedman, 2001). Since then, more and more boosting algorithms have been designed by solving different loss functions, including LS_Boost which optimizes least-squares, LAD_TreeBoost which optimizes least absolute deviation, M_TreeBoost which optimizes Huber loss, L_K_TreeBoost which optimizes logistic binomial log-likelihood and so on. Recently, Lev. V designed a software reliability boosting method to avoid overfitting (Utkin and Coolen, 2020). By comparison, we found that most of the boosting algorithms are more designed for regression problems than classification problems, for their optimization goals are not suitable for classification models. Therefore, the goal of this paper is to design a robust function estimation algorithm to deal with classification problems, where abnormal observation occurs. In this paper, only additivity of $F(x)$ is required. Notice that we do not make any assumption of the distribution of (y, x). The additive models have a long history in statistics, and so we give only two forms which we focus on. The first discrete form is

$$F(x) = \sum_{m}^{M} c_m f_m(x), \tag{1.6}$$

where $c_m \in \mathbb{R}^+$, $f_m(x) \in \{-1, 1\}$. The second real form is

$$F(x) = \sum_{m}^{M} f_m(x), \tag{1.7}$$

where $f_m(x) \in \mathbb{R}$. The difference between the discrete and real form additive models comes from the codomain of the components f_m. The *backfitting algorithm* (Friedman and Stuetzle, 1981; Buja et al., 1989) is a convenient modular "Gauss-Seidel" algorithm for fitting additive models. On our minimization problem, a backfitting update is

$$f_m(x) \leftarrow \underset{f_m(x)}{argmin} \quad J\left(\sum_{k \neq m} f_k(x) + f_m(x)\right) \quad for \quad m = 1, 2, \cdots, M, 1, \cdots. \tag{1.8}$$

Any method or algorithm for estimating a function of x can be used to obtain an estimate of the minimizer in (1.8). The methods include nonparametric algorithms, such as local regression, smoothing splines or tree regression. In the right-hand side, all the latest versions of the functions f_m are used in forming the minimization problem. The backfitting cycles are repeated until convergence. Alternatively, one can use a "greedy" forward stepwise approach,

$$f_m(x) \leftarrow \underset{f_m(x)}{argmin} \quad J\left(F_{m-1}(x) + f_m(x)\right) \quad for \quad m = 1, 2, \cdots, M, \cdots, \tag{1.9}$$

where,

$$F_m(x) = \sum_{i=1}^{m} f_m(x).$$

This approach is used by Mallat and Zhifeng Zhang (1993) in matching pursuit. In this paper, we estimate $F(x)$ by this forward stepwise approach.

Our main contributions are as follows:

- We proposed a loss function to deal with classification problems, where abnormal observations occurs. By comparison, we analyzed the effectiveness of the proposed loss function.

- Under the assumption that the model is additive, we derived two iterative algorithms, which will generate two models for classification tasks.
- Simulation experiments and real data experiments are designed to verify the effectiveness of the algorithm. Through experiments, we found that when the boosting algorithm is used to fit a model, decision committee of simpler classifiers can often achieve better prediction performances.

2 Main results

2.1 Comparison & robust discussion

In this subsection, we give the reason of the suggestion, that's the negative log-likelihood criterion (1.2) is suitable for classification where long-tail error outlier occurs. Many generalization of Boosting algorithms have been proposed. Friedman(2000) suggests AdaBoost can be derived as a method for fitting an additive model $\sum_m f_m(x)$ in a forward stagewise manner(Friedman et al., 2000). Friedman(2001) derived a Gradient Boosting Machine, a general framework for function estimation (Friedman, 2001). As applications of Gradient Boosting, Least-squares regression (2.1), M-estimation (2.2) and negative binomial log-likelihood (2.3) were discussed. In figure 1, four kinds of loss were plot, where $y \in \{-1, 1\}$ and $yF(x)$ measures the consistence between y and $F(x)$.

$$L(y, F) = (y - F)^2/2 \tag{2.1}$$

$$L(y, F) = \begin{cases} \frac{1}{2}(y - F)^2, & |y - F| \leq \delta \\ \delta(|y - F| - \delta/2) & |y - F| > \delta \end{cases} \tag{2.2}$$

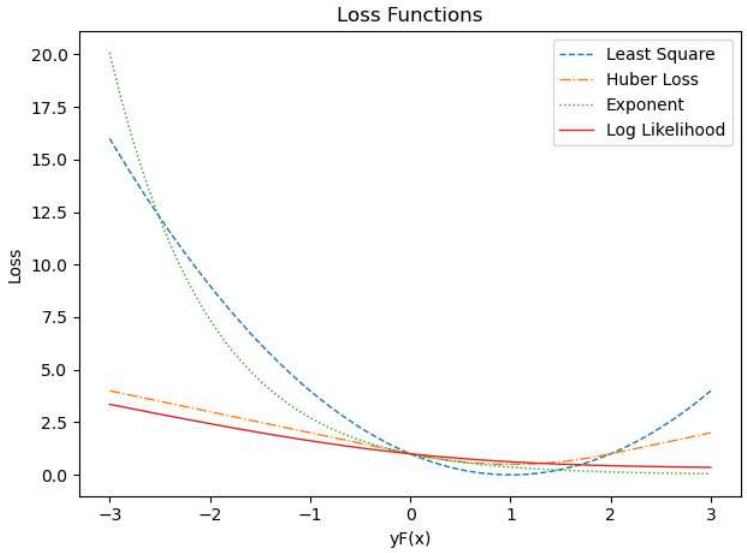

Figure 1: The loss goes by $yF(x)$. Least Square stands for equation (2.1), Huber Loss for (2.2) where $\delta = 1$, Exponent for e^{-yF}, and Log likelihood for (1.2). All curves pass through point $(0, 1)$, by panning up and down.

$$L(y, F) = \log(1 + e^{-2yF}) \tag{2.3}$$

We can see that (2.3) is equivalent to (1.2) up to a factor 2. In Friedman (2001), only Newton's method was used for solving (2.3). Although Newton's method can achieve fast convergency in theory, it is easy to cause instability in the calculation, especially when the Decision Tree is used as the basic learner. And the discussion of robustness of these losses was ignored.

For binomial classification, y takes value in $\{-1, 1\}$. We take the prediction mechanism to be $sign\,(F(x))$. For a given $F(x)$, $yF(x) > 0$ stands a correct prediction. The larger $yF(x)$ means the better the estimation $F(x)$, vice versa. Here are some discusses:

(a) Least Square loss is often used to deal with regression problems. But for classification problems, the least square method will no longer be suitable. Because it imposes

a greater penalty on samples that are classified too correctly. When $yF(x) \to \infty$, the Least Square loss grows. M-regression techniques attempt resistance to long-tailed error distributions and outliers while maintaining high efficiency for normally distributed errors. It also encounters the same problem of Least Square loss. Its advantage is that the way to increase the loss of misclassified samples is linear, which makes M-regression robust when dealing with long-tailed error outliers.

(b)M-regression uses absolute loss instead of square loss, which gives the same weight δ on the observations where $|y - F| > \delta$, while takes square loss when $|y - F| < \delta$. This is the main reason for success of Huber loss. But for classification problems, less penalty is needed for $yF > 0$, as $yF > 0$ stands for correct prediction. As we can see,

$$yF = \frac{1}{2}(y - F)^2 - \frac{1}{2} - \frac{1}{2}F^2,$$

where y take value in $\{1, -1\}$. There is no special to make $y = 1$ and $F > 0$, then $yF > 0$ always stands for correct prediction. But for large F, $|y - F| > \delta$, Huber loss always takes penalty δ on those observations, which is not reasonable. We should make less penalty on the observations where $yF > 0$. As our suggestion, the penalty goes as

$$\frac{e^{yF}}{1 + e^{yF}} - 1 \to 0 \quad as \quad yF \to +\infty.$$

Although, Huber loss makes no suitable for classification problems. Its succession gives some enlighten, such as take the same penalty on the observations where $|y - F| > \delta$. So in our suggestion, the penalty goes as

$$\frac{e^{yF}}{1 + e^{yF}} - 1 \to 1 \quad as \quad yF \to -1.$$

For wrong predictions, which stands for outliers when the function proposed for F is correct, the penalty weight will no be larger than 1. This assume us takes not so much concern on the observations where $yF \to -\infty$. This property inherit the

success of Huber loss.

(c) Exponential criterion appeared in Schapire and Singer (1998); Breiman (1999). Friedman(2000) shows that the AdaBoost algorithms, appeared in Freund et al. (1996), can be interpreted as stagewise estimation procedures for fitting an additive logistic regression model (Friedman et al., 2000). The Exponential criterion avoids imposing the greater penalty on the too correct samples. But its punishment for errors increases exponentially, which makes it non-robust when dealing with long-tailed error outliers. In this paper, the Log-likelihood loss (1.2) is suggested and discussed in detail. Because this loss avoids imposing greater penalty on the too correct samples. It inherits the advantage of Huber loss. When $yF(x) \to -\infty$, which stands for wrong prediction, the loss function gradually becomes linear, $\log\left(1 + e^{yF(x)}\right) - yF(x) \to -yF(x)$, which gives the loss some robustness when dealing with long-tailed error outliers.

In the following sections, we derive the estimation algorithm of $F(x)$. As $F(x)$ is additive, we update it through the following two ways.

1. $F(x) \leftarrow F(x) + cf(x), f(x) \in \{-1, 1\}, c$ is a positive constant.
2. $F(x) \leftarrow F(x) + f(x), f(x) \in \mathbb{R}$.

The two updating ways correspond to two different algorithms, namely, Discrete L-AdaBoost and Real L-AdaBoost, where L stands for *logit* transformation.

2.2 Discrete L-AdaBoost

Suppose we have a current estimate $F(x)$ and seek an improved estimate,

$$F(x) \leftarrow F(x) + cf(x),$$

where $c > 0$, $f(x) \in \{-1, 1\}$.

Proposition 2.1 *The Discrete L-AdaBoost algorithm (population version) builds an additive logistic regression model via Gradient descent updates for minimizing the negative log-likelihood criterion (1.2).*

Proof. Let $J(F) = E\left[\log\left(1 + e^{yF(x)}\right) - yF(x)\right]$. For fixed c (and x), we expand $J(F + cf(x))$ to second order about $f(x) = 0$,

$$\begin{aligned} J(F + cf) &= E\left[\log\left(1 + e^{yF(x)+cyf(x)}\right) - yF(x) - cyf(x)\right] \\ &\approx E\left[\log\left(1 + e^{yF(x)}\right) - yF(x) - \frac{1}{1 + e^{yF(x)}}cyf(x) + \frac{e^{yF(x)}}{2(1 + e^{yF(x)})^2}cy^2f^2(x)\right] \\ &= E\left[\log\left(1 + e^{yF(x)}\right) - yF(x) - \frac{1}{1 + e^{yF(x)}}cyf(x) + \frac{e^{yF(x)}}{2(1 + e^{yF(x)})^2}c^2\right], \end{aligned}$$

since $y^2 = 1$ and $f^2(x) = 1$. We seek $f(x)$ to minimize $J(F + cf)$ is equivalent to minimize this expectation,

$$-E\left[\frac{1}{1 + e^{yF(x)}}yf(x)\right].$$

As $yf(x) = -\frac{1}{2}(y - f(x))^2 + 1$, the above minimization problem turns to be

$$\underset{f(x)}{minimize} \quad E\left[\frac{1}{1 + e^{yF(x)}}(y - f(x))^2\right].$$

Then the minimizer $f(x)$ is

$$f(x) \leftarrow E_w(y) := \frac{E\left[\frac{1}{1+e^{yF(x)}}y\right]}{E\left[\frac{1}{1+e^{yF(x)}}\right]},$$

where $w = \frac{1}{1+e^{yF(x)}}$. Given $f(x) \in \{-1, 1\}$, we can directly minimize $J(F + cf)$ to determine c,

$$c = \underset{c}{argmin} \quad E\left[\log\left(1 + e^{yF(x)+cyf(x)}\right) - yF(x) - cyf(x)\right].$$

Taking derivation of the right-hand formula on c, the minimizer c satisfies

$$E\left[\frac{yf(x)}{1+e^{yF(x)+cyf(x)}}\right]=0.$$

After observing n points of datasets, c can be estimated by the root of the following equation,

$$\sum_{i=1}^{n}\frac{y_i f(x_i)}{1+e^{y_i F(x_i)+cy_i f(x_i)}}=0.$$

Newton's root-find method can be used for seeking the root.

Gradient descent method is a very popular function estimation method(Dicker et al., 2013; Jiang et al., 2017). In our derivation, the second-order Taylor expansion of the loss function is used. This is essentially using the gradient descent method, but the step size is determined by optimizing the objective function again. Summarizing the estimation process of $F(x)$ from **proposition** 2.1, we get Algorithm 1 for classification problems.

It should be noted that how to estimate

$$f(x)\leftarrow E_w(y|x). \tag{2.4}$$

In practice, any method or algorithm for estimating a function of x can be used to obtain an estimate of the weighted conditional expectation in (2.4). Here we choose tree based classifier as an estimate of $f(x)$, for it's widely used for basic functions (Freund et al., 1996; Schapire and Singer, 1998; Ke et al., 2017).

2.3 Overcoming the instability.

Given a loss function $J(F):=E[L(yF)]$. According to Newton's method, $F(x)$ can be updated by

$$F\leftarrow F(x)-\frac{E[L'(yF)y]}{E[L''(yF)]},$$

Algorithm 1 Discrete L-AdaBoost

1. Start with the training data $\{x_i, y_i\}_{i=1}^n$ and weights $w_i = \frac{1}{n}, i = 1, \cdots, n$, $F_0(x) = 0$.

2. Repeat for $m = 1, \cdots, M$:

   (a) Fit the classifier $f_m(x) \in \{-1, 1\} := E_w y$ using weights w_i on the training data. Here, we use Decision tree as an estimate of $f_m(x)$.

   (b) Solve the following equation for $c \in \mathbb{R} : \sum_{i=1}^n \frac{y_i f_m(x_i)}{1+e^{y_i F_{m-1}(x)+c y_i f_m(x_i)}} = 0$. Use Newton's root-finding method:

   initialize $c = 0$;

   Repeat until convergence:

$$c \leftarrow c + \frac{\sum_i \frac{y_i f_m(x_i)}{1+e^{y_i F_{m-1}(x_i)+c y_i f_m(x_i)}}}{\sum_i \frac{e^{y_i F_{m-1}(x_i)+c y_i f_m(x_i)}}{(1+e^{y_i F_{m-1}(x_i)+c y_i f_m(x_i)})^2}}.$$

   (c) Update $F_m(x) \leftarrow F_{m-1}(x) + c f_m(x)$;

   Update the weights:

   $w_i \leftarrow \frac{1}{1+e^{y_i F_m(x_i)}}$ and renormalize so that $\sum_i w_i = 1$.

3. Output the classifier $\text{sign}(F_M(x))$.

where L', L'' represent the first and second derivatives of L, respectively. In our case, Decision Tree is used for the update,

$$f \leftarrow E_w[\tilde{y}],$$

where $\tilde{y} = -\frac{L'(F(x))}{L''(F(x))}y$, $w = L''(F(x))$. For strictly convex optimization problems, this weighted expectation will not cause much trouble. But for the loss function in this paper, its second derivative tends to 0, when $|F(x)|$ increases. Especially in M-regression, the second derivative is exactly 0 for $|y - F| > \delta$. This greatly limits the scope of application of Newton's method. In fact, we find that although the second derivative tends to zero at some observations, its expectation $E[L''(yF)]$ tends to be much larger than zero. From the point of view of optimization, the expectation of the second derivative is used as a step to control the size of the gradient. Therefore, the update can be seen as,

$$f \leftarrow \frac{E[-L'(F(x))]}{E[L''(x)]}E_w(y),$$

where $w = -L'(F(x))$. Decision Tree is used to learn $E_w[y]$, then the trained Decision Tree is given a multiplier $\frac{E[-L'(F(x))]}{E[L''(x)]}$. In this way, we not only take advantage of the fast convergence of Newton's method, but also avoid the instability caused by the second derivative too small.

2.4 Real L-AdaBoost

Suppose we have a current estimate $F(x)$ and seek an improved estimate, like

$$F(x) \leftarrow F(x) + f(x),$$

where $f(x) \in \mathbb{R}$.

Proposition 2.2 *The Real L-AdaBoost algorithm fits an additive logistic regression model via Newton like updates and approximate optimization of (1.2).*

Proof. Let $J(F) = E\left[\log\left(1 + e^{yF(x)}\right) - yF(x)\right]$.

$$\left.\frac{\partial J(F(x) + f(x))}{\partial f(x)}\right|_{f(x)=0} = -E\left[\frac{y}{1 + e^{yF(x)}}|x\right],$$

$$\left.\frac{\partial^2 J(F(x) + f(x))}{\partial f(x)^2}\right|_{f(x)=0} = E\left[\frac{e^{yF(x)}}{(1 + e^{yF(x)})^2}|x\right] \; since \quad y^2 = 1.$$

Hence the Newton update is

$$\begin{aligned} F(x) &\leftarrow F(x) + \frac{E\left[\frac{y}{1+e^{yF(x)}}|x\right]}{E\left[\frac{e^{yF(x)}}{(1+e^{yF(x)})^2}|x\right]} \\ &= F(x) + \frac{E\left[\frac{1}{1+e^{yF(x)}}|x\right]}{E\left[\frac{e^{yF(x)}}{(1+e^{yF(x)})^2}|x\right]} E_w\left[y|x\right]. \end{aligned}$$

where $w(x, y) = \frac{1}{1+e^{yF(x)}}$,

$$E_w(x) \triangleq \frac{E\left[wx\right]}{E\left[w\right]}.$$

In Algorithm 2, we takes adaptive Newton steps for updating $F(x)$. The main difference between this and the Discrete L-AdaBoost algorithm is how it uses its estimates of the weighted expectations to update the functions. With more detailed analysis, we found that the multiplier are updated differently. In Discrete L-AdaBoost, the multiplier c solves $\sum_{i=1}^{n} \frac{y_i f_m(x_i)}{1+e^{y_i F_{m-1}(x) + c y_i f_m(x_i)}} = 0$, where root-finding method is used. While in Real L-AdaBoost, the multiplier

$$E\left[\frac{1}{1 + e^{yF(x)}}|x\right] / E\left[\frac{e^{yF(x)}}{(1 + e^{yF(x)})^2}|x\right]$$

is estimated by $\frac{E\left[\frac{1}{1+e^{yF(x)}}\right]}{E\left[\frac{e^{yF(x)}}{(1+e^{yF(x)})^2}\right]}$, which is calculated according the original function $F(x)$.

Algorithm 2 Real L-AdaBoost

1. Start with the training data $\{x_i, y_i\}_{i=1}^n$, and weights $w(x_i, y_i) = \frac{1}{n}, i = 1, \cdots, n$, $F_0(x) = 0$.

2. Repeat for $m = 1, \cdots, M$:

   (a) Compute $c_m = E\left[\frac{1}{1+e^{yF_{m-1}(x)}}\right] / E\left[\frac{e^{yF_{m-1}(x)}}{(1+e^{yF_{m-1}(x)})^2}\right]$.

   (b) Fit the regression function $f_m(x) \in \mathbb{R}$ by weighted least-squares of y_i to x_i with w_i

   (c) Update $F_m(x) \leftarrow F_{m-1}(x) + c_m f_m(x)$.

   (d) Recompute weights $w(y_i, x_i) = \frac{1}{1+e^{y_i F(x_i)}}, i = 1, \cdots, n$ and renormalize so that $\sum_i w_i = 1$.

3. Output the classifier: $\text{sign}(F_M(x))$.

2.5 Multiclass case

For the multiclass case, several generalizations of AdaBoost is presented in Schapire and Singer (1998), among which their AdaBoost.MH seemed to dominate the others in the empirical studies. Therefore, we use AdaBoost.MH algorithm (Algorithm 3) here to handle the multiclass problems. The key to the algorithm is converting the J classes problem into fitting a two-class classifier J times. The reasonableness relies on that this algorithm minimizes the loss function $\sum_{j=1}^{J} El\big(y_j, F_j(x)\big)$, which is equivalent to minimizing each loss function $El\big(y_j, F_j(x)\big)$ separately.

Algorithm 3 AdaBoost.MH (Schapire and Singer, 1998)

1. Expand the original N observations into $n \times J$ pairs $\big((x_i, 1), y_{i1}\big)$, $\big((x_i, 2), y_{i2}\big)$, $\cdots$, $\big((x_i, J), y_{iJ}\big)$, $i = 1, \cdots, n$. Here y_{ij} is the $\{-1, 1\}$ response for class j and observation i .

2. Apply Real L-AdaBoost to the augmented dataset, producing a function $F : \mathcal{X} \times (1, \cdots, J) \mapsto \mathbb{R}$; $F(x, j) = \sum_m f_m(x, j)$.

3. Output the classifier $\arg\max_j F(x, j)$.

3 Discussion of stability where outliers occur with some artificial examples

We call sample separable, if there exists a criterion that can separate each categories. For example, $y = 2\mathbf{1}_{x>0.5} - 1$ is a separable sample, where $\mathbf{1}_{(\cdot)}$ is indicator of $x > 0.5$. In separable sample, outliers may come from measuring tools or manual record error.

A stable classification algorithm should find a true criterion or a criterion close to the truth, under the interference of outliers. Another outliers occurrence comes from inseparable samples. For example, $P(y=1)=x$ and $P(y=-1)=1-x$, $x\in[0,1]$. Here, we can not find a criterion that separate $y=1$ and $y=-1$. By maximizing Likelihood, we can still find a proper boundary for classification, that's $y=2\mathbf{1}_{x>0.5}-1$. In this situation, outliers come from the indistinguishability of each categories. Under the second scenario, the distribution of sample is very critical for finding a stable boundary, that should be close to the boundary of maximizing Likelihood. In this paper, we presume,

$$Y=\mathbf{1}_{F(x)>0}. \tag{3.1}$$

And $F(x)$ is proposed to be additive, that's $F(x)=f_1(x)+f_2(x)+\cdots$. The additivity of functions is not the originality of this article. It is a classic function estimation hypothesis, which is widely popular due to its simple and reasonable structure(Feng et al., 2018; Sundararajan and Ollis, 2021).

In this section, we give several samples, where outliers occur, to show the fineness of Discrete L-Adaboost and Real L-Adaboost.

Example 3.1 *(**One-sided contamination sample**) Draw* 20000 *observation points for* $x_1,x_2\sim(0,1)$ *uniformly, and assign category* $y=1$ *for each point if* $x_1-x_2>0$, *else assign category* $y=-1$. *This sample is separable by criterion* $x_1-x_2>0$. *Then contaminate* α *proportion of the observations by assigning* $y=1$ *for the randomly chosen* α *proportion of the* $y=-1$ *points. The contaminated data set is presented at first row of Figure 2.*

Example 3.2 *(**Three classification**) Let* (x_1,x_2) *scatter in an unit circle evenly,*

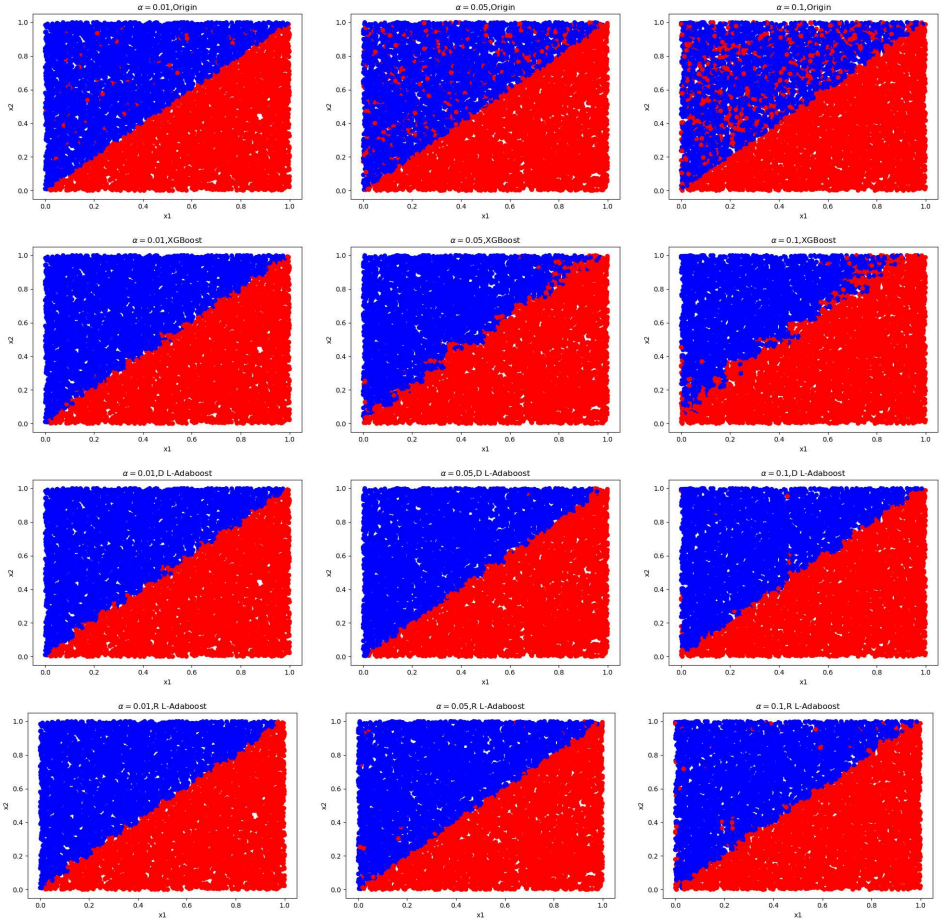

Figure 2: Example1. The boundary learned by classification algorithms, Discrete L-Adaboost, Real L-Adaboost and XGBoost.

that's $x_1 = r\cos(\theta), x_2 = r\sin(\theta)$, $r \in (0,1), \theta \in (0, 2\pi]$. *Presume we have* 45000 *observations. Separate all points into three categories by* $r \in (0, \frac{1}{3}]$, $r \in (\frac{1}{3}, \frac{2}{3}]$ *and* $r \in (\frac{2}{3}, 1]$. *Let* $G = 0, 1, 2$ *stands for categories. Select* α *proportion of points from group* $G = 0, 2$, *and assign them as group* $G = 1$. *Then we get a contaminated data set, at first row of Figure 3.*

Example 3.3 ***(Mixture Gaussian Distribution)*** *Select* 20000 *points from two*

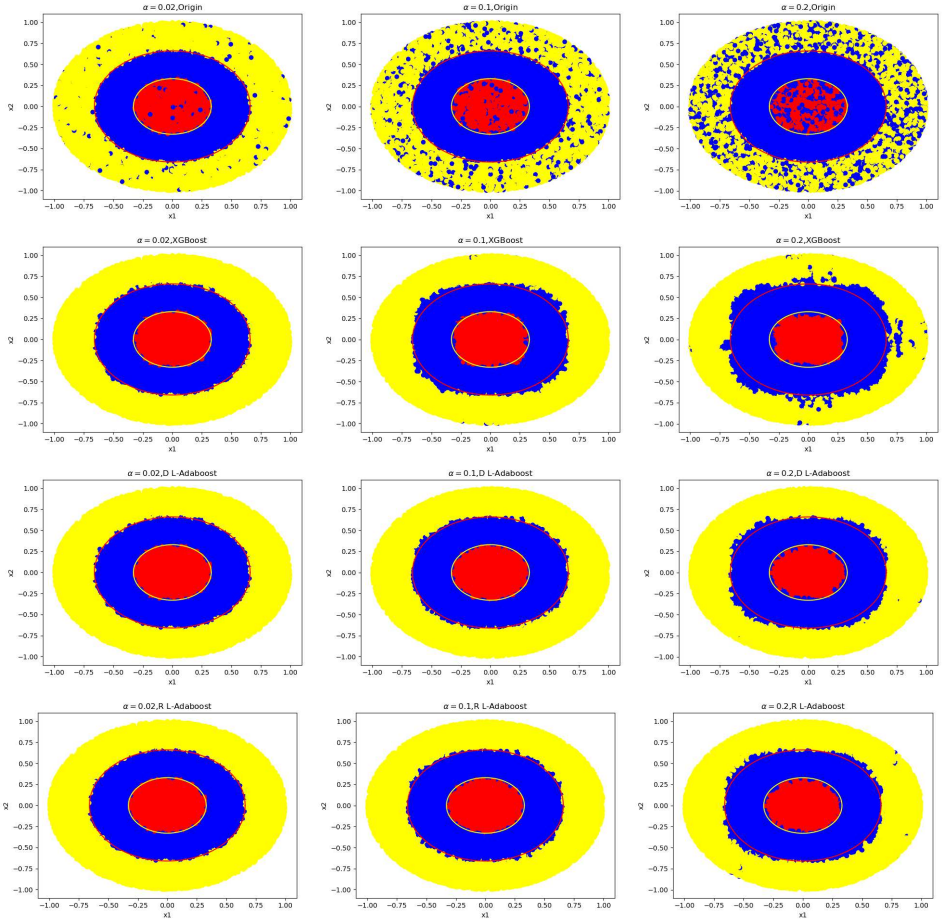

Figure 3: Example2. The boundary learned by classification algorithms, Discrete L-Adaboost, Real L-Adaboost and XGBoost.

Gaussian distribution,

$$N\left(\begin{bmatrix}\frac{1}{2}L\\0\end{bmatrix},\begin{bmatrix}1&0\\0&1\end{bmatrix}\right),N\left(\begin{bmatrix}-\frac{1}{2}L\\0\end{bmatrix},\begin{bmatrix}1&0\\0&1\end{bmatrix}\right)$$

7200 *from the first distribution and* 12800 *from the second distribution. Assign the points from first distribution as first category denoted as* $Y=1$*, and the points from second distribution as second category denoted a* $Y=-1$*. Here we get three sample by setting* $L=0.05,1,2$*, which measuring the distance of centers between these two distribution. See Figure 4.*

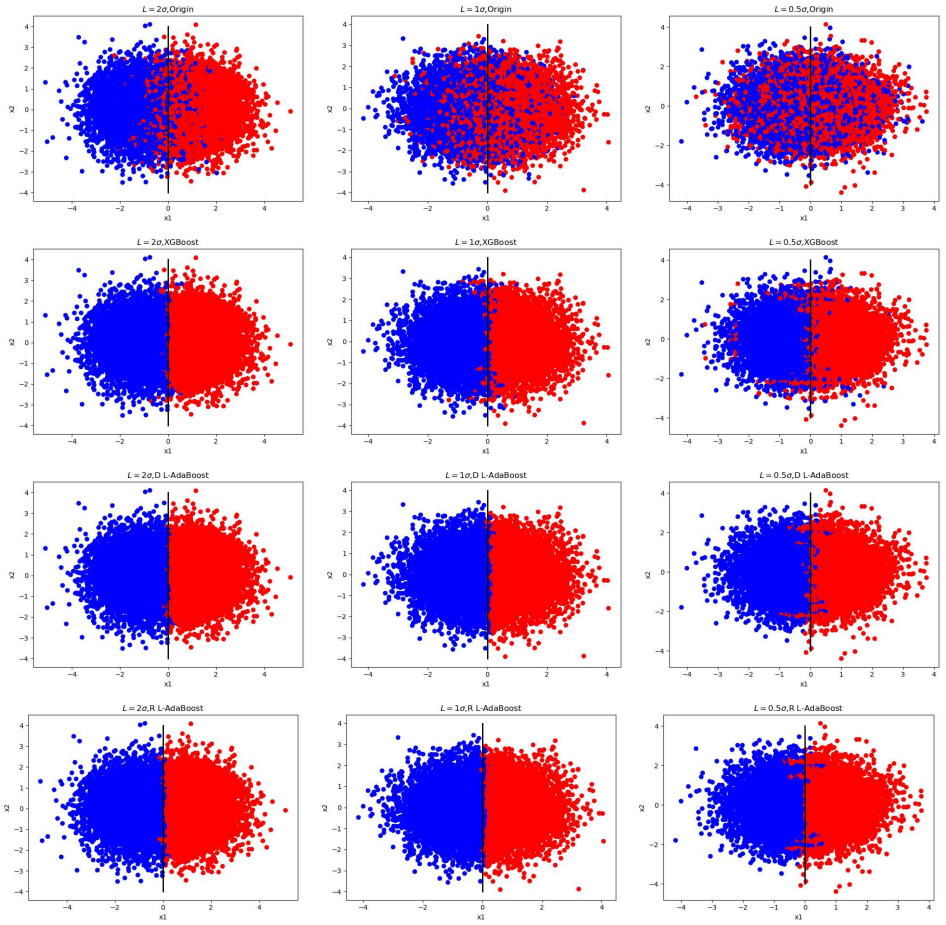

Figure 4: Example3. The boundary learned by classification algorithms, Discrete L-Adaboost, Real L-Adaboost and XGBoost.

In Example 3.1 and 3.3, the max_depth of Cart Tree was set $max\_depth = 1$. In Example 3.2, the max_depth is 2. For all classifiers, number of iterations is 1000. To test the immunity of outliers, we gradually increase the proportion of abnormalities. By increasing the imbalance of noise, we found that most boosting algorithms will be affected. We choose the XGBoost(Chen et al. (2015)) algorithm as a typical comparison, as its popularity. In Example 3.1, 3.2, the boundary for classification moves to the part of contaminated category. In Example3.2, XGBoost shows its invasion at $\alpha = 0.2$, standing for 20% points have be contaminated. While Discrete

L-AdaBoost and Real L-AdaBoost maintain good stability. Boosting algorithm is greedy, as it believes residual learnable theory (He et al. (2016)). This can be seen from Example 3.3, where XGBoost tries to learn noises. Our recommended algorithm behaves more calmly in the face of this kind of outliers.

4 Experiments with real world data

In this section, we compared the performance of the Discrete L-AdaBoost method and Real L-AdaBoost method with other two popular Boosting methods, that's Discrete AdaBoost(first appeared in Freund et al. (1996)) and Random Forest, which stands for milestone along the process of Boosting methods developments. For brevity, some simplification is need, as follows:

DAB : Discrete AdaBoost(Freund et al., 1996).

RF : Random Forest.

DLAB : Discrete L-AdaBoost - Algorithm 1.

RLAB : Real L-AdaBoost - Algorithm 2.

All the algorithms use the AdaBoost.MH when dealing with the multiclass problems. The data sets used in the experiments are summarized in Table 1.

From Table 2, boosting methods, DAB, DLAB, RLAB, outperform Random Forests in the four selected data sets. The booting algorithms have an average accuracy of at least 5 percentage points higher than that of the random forest algorithm on the dataset 'Ionosphere' and data set'Glass', 15% higher on data set 'Segmentation'.

We check the performance of RLAB on the data sets "Ionosphere" and "Wavefor-

Table 1: The real world data used in the experiments.

Data set	Train	Test	Inputs	Classes
Ionosphere	351	5-fold CV	34	2
Glass	214	5-fold CV	10	6
Segmentation	210	5-fold CV	20	7
Waveform	5000	5-fold CV	22	3

m". With the improvement of the learning ability of the basic learner, that is, from the trumps to Decision Tree with four terminal nodes, the accuracy of the RLAB algorithm has not been significantly improved, even lower. The same happens with the DLAB algorithm on the data set "Glass" and the DAB algorithm on the data set "Waveform". This shows that for the boosting algorithms, the simpler the basic learner, the better the performance. That's the weaker learner with low variance shall be chosen.

The DLAB (Freund et al., 1996) algorithm as a classic boosting algorithm is often used as a comparison object. According to the comparison of the loss functions, we found that negative log-likelihood is more advantageous when dealing with long-tailed error outliers. This view can be clearly displayed by the accuracy rate in Table 2.This improvement is more stable on dataset "Waveform", which contains 5000 observations. This gives us confidence in the effectiveness of the suggested algorithms, especially when the sample size is large.

Table 2: Error rates of real world data sets.

	2 Terminal Nodes			4 Terminal Nodes		
Method	30	60	90	30	60	90
-Ionosphere	Cart error =0.105					
RF	0.157	0.154	0.151	0.091	0.094	0.097
DAB	0.083	0.080	0.086	0.077	0.082	0.085
DLAB	0.081	0.074	0.073	0.070	0.075	0.078
RLAB	0.074	0.063	0.071	0.069	0.074	0.077
-Glass	Cart error =0.440					
RF	0.435	0.440	0.392	0.445	0.436	0.435
DAB	0.383	0.382	0.364	0.378	0.373	0.373
DLAB	0.345	0.345	0.331	0.369	0.360	0.336
RLAB	0.344	0.345	0.345	0.365	0.360	0.335
-Segmentation	Cart error =0.186					
RF	0.281	0.214	0.257	0.229	0.200	0.171
DAB	0.081	0.067	0.071	0.071	0.071	0.067
DLAB	0.065	0.056	0.060	0.067	0.057	0.057
RLAB	0.060	0.055	0.056	0.062	0.055	0.054
-Waveform	Cart error =0.267					
RF	0.330	0.323	0.312	0.211	0.213	0.210
DAB	0.165	0.152	0.148	0.159	0.158	0.155
DLAB	0.159	0.151	0.145	0.145	0.147	0.148
RLAB	0.154	0.145	0.142	0.145	0.143	0.146

5 Conclusion

This paper has presented two boosting algorithms, Discrete L-AdaBoost and Real L-AdaBoost, based on additive logistic models. These two algorithms are supposed to generate a strong composite learner from weak learner and Decision Trees were used as weak learners in experiment studies. By comparison of different loss function, we show the suitability of negative log-likelihood criterion for classification problems, where abnormal observations occurs. And based on this loss function, we derived two stable algorithms to fit the additive logistic model for classification tasks. Through simulation and real data experiments, we verified the effectiveness of the proposed algorithms.

Acknowledgements

Our thanks go to Professor Bo Li for stimulating this research. Our research is supported by NSF of China (Grant No. 61877023) and the Fundamental Research Funds for the Central Universities (Grant No. CCNU19TD009).

References

Breiman, L. (1999). Prediction games and arcing algorithms. *Neural computation*, **11**(7), 1493–1517.

Buja, A., Hastie, T., and Tibshirani, R. (1989). Linear smoothers and additive models. *The Annals of Statistics*, **17**(2), 453–510. ISSN 00905364. URL http:

//www.jstor.org/stable/2241560.

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., et al. (2015). Xgboost: extreme gradient boosting. *R package version 0.4-2*, **1**(4), 1–4.

Daraghmi, Y., Yi, C., and Chiang, T. (2014). Negative binomial additive models for short-term traffic flow forecasting in urban areas. *IEEE Transactions on Intelligent Transportation Systems*, **15**(2), 784–793. doi: 10.1109/TITS.2013.2287512.

Dicker, L., Huang, B., and Lin, X. (2013). Variable selection and estimation with the seamless-l 0 penalty. *Statistica Sinica*, **23**(2), 929–962. ISSN 10170405, 19968507. URL http://www.jstor.org/stable/24310368.

Feng, Y., Yang, Q., Tong, X., and Chen, L. (2018). Evaluating land ecological security and examining its relationships with driving factors using gis and generalized additive model. *Science of the total environment*, **633**, 1469–1479.

Freund, Y. (1995). Boosting a weak learning algorithm by majority. *Information and Computation*, **121**(2), 256 – 285. ISSN 0890-5401. doi: https://doi.org/10.1006/inco.1995.1136. URL http://www.sciencedirect.com/science/article/pii/S0890540185711364.

Freund, Y., Schapire, R. E., et al. (1996). Experiments with a new boosting algorithm. In *icml*, volume 96, pages 148–156. Citeseer.

Friedman, J., Hastie, T., and Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *Ann. Statist.*, **28**(2), 337–407. doi: 10.1214/aos/1016218223. URL https://doi.org/10.1214/aos/1016218223.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, **29**(5), 1189–1232. ISSN 00905364. URL http://www.jstor.org/stable/2699986.

Friedman, J. H. and Stuetzle, W. (1981). Projection pursuit regression. *Journal of the American Statistical Association*, **76**(376), 817–823. doi: 10.1080/01621459.1981.10477729. URL https://www.tandfonline.com/doi/abs/10.1080/01621459.1981.10477729.

Gao, Y., Wen, J., and Peng, L. (2019). New exponential stability criterion for switched linear systems with average dwell time. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, **233**(8), 935–944. doi: 10.1177/0959651818807607. URL https://doi.org/10.1177/0959651818807607.

Hastie, T., Tibshirani, R., and Buja, A. (1994). Flexible discriminant analysis by optimal scoring. *Journal of the American Statistical Association*, **89**(428), 1255–1270. doi: 10.1080/01621459.1994.10476866. URL https://doi.org/10.1080/01621459.1994.10476866.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Huber, P. J. (1972). The 1972 wald lecture robust statistics: A review. *Ann. Math. Statist.*, **43**(4), 1041–1067. doi: 10.1214/aoms/1177692459. URL https://doi.org/10.1214/aoms/1177692459.

Jiang, B., Wu, T.-Y., Zheng, C., and Wong, W. H. (2017). Learning summary statistic for approximate bayesian computation via deep neural network. *Statistica Sinica*, **27**(4), 1595–1618. ISSN 10170405, 19968507. URL http://www.jstor.org/stable/26384090.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30, pages 3146–3154. Curran Associates, Inc. URL https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf.

Kong, D., Bondell, H. D., and Wu, Y. (2018). Fully efficient robust estimation, outlier detection and variable selection via penalized regression. *Statistica Sinica*, **28** (2), 1031–1052. ISSN 10170405, 19968507. URL http://www.jstor.org/stable/44841936.

Lv, J., Pawlak, M., and Annakkage, U. D. (2017). Prediction of the transient stability boundary based on nonparametric additive modeling. *IEEE Transactions on Power Systems*, **32**(6), 4362–4369. doi: 10.1109/TPWRS.2017.2669839.

Mallat, S. G. and Zhifeng Zhang (1993). Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, **41**(12), 3397–3415. doi: 10.1109/78.258082.

Pu, Z. and Rao, R. (2018). Exponential stability criterion of high-order bam neural networks with delays and impulse via fixed point approach. *Neurocomputing*, **292**,

63 – 71. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2018.02.081. URL http://www.sciencedirect.com/science/article/pii/S0925231218302480.

Rana, P., Vilar, J., and Aneiros, G. (2018). On the use of functional additive models for electricity demand and price prediction. *IEEE Access*, **6**, 9603–9613. doi: 10.1109/ACCESS.2018.2805819.

Schapire, R. E. (1990). The strength of weak learnability. *Machine learning*, **5**(2), 197–227.

Schapire, R. E. and Singer, Y. (1998). Boostexter: A system for multiclass multi-label text categorization. *Machine Learning*, **39**(2/3), 135–168.

Sun, Q., Zhou, W.-X., and Fan, J. (2020). Adaptive huber regression. *Journal of the American Statistical Association*, **115**(529), 254–265. doi: 10.1080/01621459.2018.1543124. URL https://doi.org/10.1080/01621459.2018.1543124. PMID: 33139964.

Sundararajan, A. and Ollis, B. (2021). Regression and generalized additive model to enhance the performance of photovoltaic power ensemble predictors. *IEEE Access*, **9**, 111899–111914.

Utkin, L. V. and Coolen, F. P. A. (2020). A new boosting-based software reliability growth model. *Communications in Statistics - Theory and Methods*, **0**(0), 1–28. doi: 10.1080/03610926.2020.1740736. URL https://doi.org/10.1080/03610926.2020.1740736.

Wang, L., Zheng, C., Zhou, W., and Zhou, W.-X. (2020). A new principle for tuning-free huber regression. *Statistica Sinica*. URL https://eprints.soton.ac.uk/441508/.

Wang, Y., Jiang, Y., Zhang, J., Chen, Z., Xie, B., and Zhao, C. (2019). Robust variable selection based on the random quantile lasso. *Communications in Statistics - Simulation and Computation*, **0**(0), 1–11. doi: 10.1080/03610918.2019.1643886. URL https://doi.org/10.1080/03610918.2019.1643886.