Boosting method for local learning in statistical classification

Masanori KAWAKITA

DOCTOR OF PHILOSOPHY

Department of Statistical Science
School of Mathematical and Physical Science
Graduate University for Advanced Studies

2005 (School Year)

Contents

1 Introduction **1**

2 Survey on ensemble learning **5**
2.1 Setup and Notations . . . 5
2.2 Some preliminaries . . . 6
2.2.1 Concentration inequalities . . . 6
2.2.2 Class of base classifiers and its VC dimension . . . 13
2.2.3 Bias-Variance theory of a classifier . . . 17
2.3 Bagging . . . 19
2.3.1 Bagged predictors . . . 19
2.3.2 Statistical aspects of bagging . . . 19
2.4 Boosting algorithm . . . 20
2.4.1 Base classifiers . . . 22
2.4.2 Ordinary boosting . . . 24
2.4.3 Regularized Boosting . . . 29
2.4.4 AsymBoost . . . 32
2.5 Statistical properties of boosting . . . 38
2.5.1 Least favorable error . . . 38
2.5.2 Bayes rule equivalence . . . 39
2.5.3 Training error of AdaBoost . . . 45
2.5.4 Property of generalization error . . . 46
2.5.5 Comparison between ordinary boosting and regularized boosting . . . 49

3 Application to shark bycatch data **53**
3.1 Graphical display of contribution of each feature . . . 55
3.2 Data sets . . . 57
3.3 Prediction by AdaBoost . . . 61
3.4 Comparison with logistic GAM . . . 66
3.5 Control of the balance between the false positive and negative ratios . . . 71
3.6 Discussion . . . 72

4 Local boosting method **74**
4.1 Derivation of the local boosting algorithm . . . 75
4.2 Statistical properties of local boosting . . . 80
4.2.1 Model associated with local boosting . . . 80
4.2.2 Bayes Risk Consistency . . . 83
4.2.3 Local least favorable error property . . . 101
4.3 Simulations . . . 103
4.4 Discussion . . . 115

5 Concluding remarks **116**

A Bayes classifier attains the minimum probability of misclassification **119**

B Center limit theorem **120**

C An equality on exponential function **120**

Abstract

The main objective is to study boosting methods in statistical classification. Several ensemble learning methods including boosting have attracted many researchers' interests in the last decade. In particular, it has been reported that the boosting methods perform well in many practical classification problems. The boosting algorithm constructs an accurate classifier by combining several base classifiers, which are often at most slightly more accurate than random guess. While many researchers have studied the boosting methods, their success has still some mysterious aspects. More intensive theoretical studies are required to clarify such mysteries.

We describe a survey on several ensemble learning methods. We set up the statistical classification problem and make some notations to develop discussion from learning theories. Some theoretical preliminaries for analyzing the performance of classification methods are also overviewed. Then, we survey some existing ensemble learning methods. In particular, we review theoretical properties of boosting methods, which have been clarified by several researchers.

The application of AdaBoost with decision stumps to shark bycatch data from the Eastern Pacific Ocean tuna purse-seine fishery is described. Generalized additive models (GAMs) are one of the most widely-used tools for analyzing fisheries data. It is well known that AdaBoost is closely connected to logistic GAMs when appropriate base classifiers are used. We compared results of AdaBoost to those obtained from GAMs. Compared to the logistic GAM, the prediction performance of AdaBoost was more stable, even with correlated features. Standard deviations of the test error were often considerably smaller for AdaBoost than for the logistic GAM. In addition, AdaBoost score plots, graphical displays of the contribution of each feature to the discriminant function, were also more stable than score plots of the logistic GAM, particularly in regions of sparse data. AsymBoost, a variant of AdaBoost developed for binary classification of a skewed response variable, was also shown to be effective at reducing the false negative ratio without substantially increasing the overall test error. Boosting with decision stumps, however, may not capture complicated structures in general since decision stumps are considerably simple classifiers. Use of more complicated base classifiers possibly improves the approximation ability of boosting. However, several literatures have pointed out that the use of complicated base classifiers may increase the generalization error of boosting.

In addition, it is difficult to find what types of base classifiers are appropriate to each problem without any prior knowledge.

To overcome these difficulties, we propose a new method, the *local boosting*, that is a localized version of boosting method based on the idea similar to but not the same as the local likelihood. Application of the local likelihood may improve the approximation ability considerably but also increases the computational cost, which makes the algorithm infeasible. The local boosting, however, includes a simple device for computational feasibility. We show that the local boosting has the Bayes risk consistency in the framework of PAC learning. It is seen that the estimation error increases compared to the ordinary boosting with simple base classifiers when we use the ordinary boosting with more complicated base classifiers or when we use the local boosting. However, the increase caused by the local boosting is not large. When same base classifiers are used, the local boosting attains the Bayes risk consistency in wider situations than the ordinary boosting by controlling the trade-off between estimation error and approximation error. Several simulations confirm the theoretical results and the effectiveness of the local boosting over the ordinary boosting in both binary and multiclass classifications.

Glossary

A_D^λ, A^λ	A_D^λ (A^λ) denotes an empirical (expected) loss functions
$B_\epsilon(\ell, h)$	Kernel sphere $B_\epsilon(\ell, h) = \{x \in \mathcal{X} \,\vert\, K_h(x, x_l) < \epsilon\,,\, x_\ell \in \mathcal{K}\}$
$\mathcal{C}$	$\mathcal{C} = \{f_j\}_{j=1}^J$ denote a class of all available base classifiers mapping from $\mathcal{X}$ to $\mathcal{Y}$.
D	Training data set $D = \{X_i, Y_i\}_{i=1}^n$ are i.i.d. samples generated from $P(x, Y = y)$.
$\mathcal{D}(\cdot,\cdot)$	Statistical divergence $\mathcal{D} : \mathcal{F} \times \mathcal{F} \to R_+$.
$E[\cdot]$	E denotes an expectation with respect to all random variables.
$\mathcal{F}$	$\mathcal{F} = \{F(x) \,\vert\, \mathcal{X} \to R\}$ denotes the set of all measurable discriminant functions.
g^*	Bayes classifier $g^*(x) = I(\eta(x) > 1/2) - I(\eta(x) \le 1/2)$.
$\mathcal{G}$	$\mathcal{G} = \{g : \mathcal{X} \to \mathcal{Y}\}$ denotes the set of all measurable classification functions.
H_f	For any $f \in \mathcal{C}$, $H_f = \{x \in \mathcal{X} \,\vert\, f(x) = 1\}$.
H	$H = \{H_f \,\vert\, f \in \mathcal{C}\}$
I_M	I_M denotes an M dimensional identity matrix.
$I(\cdot)$	$I(\cdot)$ denotes an indicator function.
J	J denotes the cardinality of $\mathcal{C}$.
$K_h(\cdot,\cdot)$	K_h is a kernel function mapping $\mathcal{X} \times \mathcal{X}$ to R_+ with the form $K_h(x, y) = k(\frac{\Vert x-y\Vert}{h})$.
$\mathcal{K}$	$\mathcal{K} = \{x_\ell \in \mathcal{X} \,\vert\, \ell = 1, 2, \cdots, N\}$
L,L_D	$L(g) = P(g(X) \neq Y)$. L_D denotes its empirical version over the training data D.
L^*	Bayes risk $L^* = \inf_{g \in G} L(g) = E[\min(P(Y = y \,\vert\, x), P(Y = -1 \,\vert\, x)]$
M	M indicates a number of available features.
n	n denotes the sample number in D.
N	N denotes a number of kernel center candidates, *i.e.*, $N = \vert\mathcal{K}\vert$.
P	$P(x, Y = y) = P(Y = y \,\vert\, x)P(x)$ is the underlying distributions of (X, Y).
R^M, R_+	R^M denotes an M-dimensional Euclidean space and R_+ denotes the set of all nonnegative real values.
$S_m(\cdot)$	Score functions of each feature $(x)_m$.
T	T denotes the iteration number in the boosting algorithm.
U, u, ξ	$U : R \to R$ is a differentiable, strictly convex and increasing function. We also denote U' by u and u^{-1} by ξ.
V	V denotes the VC dimension of base classifier class $\mathcal{C}$.
(X, Y)	(X, Y) is a pair of random variables taking values in $\mathcal{X} \times \mathcal{Y}$. The feature space is $X \subset R^M$. $\mathcal{Y}$ is the label set $\{1, 2, \cdots, G\}$.
ϕ	A cost function ϕ is a map from R to R satisfying some conditions.
$\mathrm{lin}(\mathcal{C})$	A linear hull of $\mathcal{C}$ is defined as $\mathrm{lin}(\mathcal{C}) = \{\sum_{j=1}^J \theta_j f_j(x) \,\vert\, f_j \in \mathcal{C}\,,\, \theta_j \ge 0\}$.
$\mathrm{conv}(\mathcal{C})$	A convex hull of $\mathcal{C}$, *i.e.*, $\mathrm{conv}(\mathcal{C}) = \{\sum_{j=1}^J \theta_j f_j(x) \,\vert\, f_j \in \mathcal{C}\,,\, \theta_j \ge 0\,,\, \sum_{j=1}^J \theta_j = 1\}$.
$\overline{\mathcal{M}}_\mathcal{K}$	The asymptotical model $\overline{\mathcal{M}}_\mathcal{K} = \{F(x) = \sum_{j=1}^J \bar{\theta}_j(x) f_j(x) \,\vert\, \bar{\theta}_j(x) = E[K_h(x, X)\theta_j(X)]\,,\, \sum_{j=1}^J E\theta_j(X) = 1\,,\, \forall j\,,\, f_j \in \mathcal{C}\,,\, \theta_j(x) \ge 0\}$.
$\mathcal{M}_\mathcal{K}$	The empirical model $\mathcal{M}_\mathcal{K} = \{F(x) = \sum_{j=1}^J \bar{\theta}_j(x) f_j(x) \,\vert\, \bar{\theta}_j(x) = \frac{1}{N}\sum_{\ell=1}^N K_h(x, x_\ell)\theta_{j\ell}\,,\, \sum_{j=1}^J \sum_{\ell=1}^N \theta_{j\ell} = N\,,\, \forall j,\, f_j \in \mathcal{C}, \theta_{j\ell} \ge 0\}$.
λ	λ is a regularization parameter in the regularized boosting algorithm.
$\mathcal{N}(\mu, \Sigma)$	$\mathcal{N}(\mu, \Sigma)$ denotes a normal density function with mean μ and covariance Σ.
Φ	$\Phi(x)$ denotes a normal distribution function with mean 0 and variance 1.
φ	φ denotes an empty set.

1 Introduction

The boosting method is one of the most attractive methods in statistical classification problems that have rapidly expanded in the last decade. The expansion added momentum to stimulating discussion in the theory of statistical learning. The open question of learnability for a set of (weak) base classifiers to be integrated into a single classifier with more accurate and efficient performance was presented by Kearns and Valiant (1988). That leads to the theoretical and experimental developments of boosting algorithms by way of several considerations including boost filtering (Schapire, 1990). (Freund and Schapire, 1997) developed the most famous and widely-used boosting algorithm, *AdaBoost*, in which base classifiers are combined by a linear coefficient vector that minimizes the exponential loss function. Then, several literatures proposed a generalized version of AdaBoost algorithm as a functional gradient descent of general convex loss functions (*e.g.*, Friedman et al., 2000; Mason et al., 1999; Collins et al., 2002; Murata et al., 2004). The exponential loss function apparently differs from classical loss functions discussed in statistics, in which the statistical interpretation of AdaBoost was given by Friedman et al. (2000) and Lebanon and Lafferty (2002). Friedman et al. (2000) pointed out the connection between AdaBoost and the maximum likelihood estimation for an additive logistic model associated with base classifiers. Lebanon and Lafferty (2002) also discussed the connection from the view of extended Kullbuck-Leibler divergence from the empirical distribution to the logistic model. This interpretation is extended to a boosting algorithm with a general loss function by the use of the Bregman U-divergence class (Murata et al., 2004). The only different point is that boosting works in the space of positive functions while the maximum likelihood estimation works in the space of probability functions. The success of boosting algorithm in practical situations is still mysterious. Freund and Schapire (1997) showed that the training error of AdaBoost and AdaBoost.M2 decreases exponentially. However, it is well known that a small training error does not indicate a small generalization error. Several researchers have tried to evaluate the generalization performance of boosting. One way to evaluate the generalization error is to derive its upperbound (Freund and Schapire, 1997; Schapire et al., 1998; Koltchinskii and Panchenko, 2002; Lugosi and Vayatis, 2004). Breiman (1998) took another way. He pointed out the relationship between bagging and boosting and studied their performance in view of bias-variance theory. However, there still remain unclear points in the success of boosting algorithm. We summarize several

ensemble learning methods, including boosting algorithms with their properties that have been already clarified in Section 2.

We demonstrate the application of boosting methods with decision stumps to fisheries data. Generalized linear models (GLMs) and Generalized additive models (GAMs) are tools used conventionally in fisheries data analysis to standardize bycatch and catch per unit effort data, as well as to identify factors leading to increased levels of bycatch (incidental mortality of non-target species) (*e.g.*, Punt et al., 2000; Bigelow et al., 1999; Swartzman et al., 1992; Lo et al., 1992) and to predict bycatch (Walsh et al., 2002). As described above, Friedman et al. (2000) elucidated that AdaBoost can be interpreted as a forward fitting algorithm to an additive logistic model. In fact, AdaBoost and the logistic GAM fits to the same target function, the half log-odds, up to the multiplicative constant. Therefore, we compare the prediction performance between AdaBoost and the logistic GAM in the application to the shark bycatch data that was provided by IATTC (Inter-American Tropical Tuna Commission). We use the decision stumps as base classifiers in this analysis. Decision stumps are most widely-used base classifiers. The use of decision stumps enables us to obtain a graphical tool, the score plot, for visualizing the dependence of bycatch on individual features. This tool is similar to that used to summarize additive contributions from a logistic GAM model. The results of the analysis indicate that the standard deviation of the test error is smaller than that of the logistic GAM. The results also indicate that score plots of AdaBoost are more stable than that of the logistic GAM. We observed the strong asymmetry between the false positive ratio (FPR) and the false negative ratio (FNR) in prediction performance of both methods. It is also demonstrated that the application of AsymBoost reduced the false negative ratio at cost of slight increase of the test error. One remarkable result in the analysis of shark bycatch data is that we observed some spatially local structures. Therefore, the classification rule that varies depending on location may improve the prediction performance.

Boosting methods with decision stumps, however, may not capture such complicated structures in general since decision stumps are too simple. Generally, when we use a base classifier that is based on only single feature, it is not difficult to find some examples where boosting methods perform poorly. A natural idea to avoid this issue is the use of complicated (stronger) classifiers. However, several literatures pointed out that the use of more complicated base classifiers may increase the generalization error of boosting method (Bartlett and Mendelson, 2002; Lugosi and Vayatis, 2004). In addition, it is difficult to

know what types of base classifiers are appropriate to each problem without any prior knowledge.

An alternative way to improve the approximation ability is application of localization techniques. Several statistical literatures have discussed localization techniques (*e.g.*, Hastie and Tibshirani, 1990; Vincent and Bengio, 2003; Roweis and Saul, 2000), which include the local likelihood method (Hjort and Jones, 1996; Fan and Gijbels, 1996; Eguchi and Copas, 1998). The local likelihood method improves the approximation ability without any prior knowledge about the underlying structure. However, the local likelihood method has some disadvantageous points. It requires separately to solve the likelihood equations that are localized by weight functions at many points. Thus, if we obtain the maximum local likelihood estimator, $\hat{\theta}_z$, for a given statistical model of probability density function, $P(x, \theta)$, and the target point, z, then we define the estimated density function as $ZP(x, \hat{\theta}_x)$ with the normalized factor, Z, in which the target point, z, is conformed to the point x where the density should be estimated. Practical applications of the local likelihood method are often infeasible in a case of a high-dimensional data space due to the following reason. The computation task for obtaining $\hat{\theta}_z$ increases exponentially with respect to the dimension because of dense evaluations over all grid points of the space. If we consider installing the local likelihood method in boosting, we have to implement boosting algorithms at all grid points separately.

We propose a localized version of boosting (denoted by the *local boosting* in the sequel) with localization similar to but not the same as the local likelihood method. The key idea is to localize the combination of base classifiers directly rather than to localize the exponential loss function as in the conventional local likelihood method. An advantageous aspect of this idea is that the resultant form of the combined base classifiers is naturally localized by the weight function. Thus, the expression enables us to reduce the implementation of boosting algorithms on all grid points to only that on empirical data points. As a result, the local boosting requires significantly less computational cost than the conventional local likelihood method, so the implementation is feasible even in a high-dimensional space. The local boosting constructs a single discriminant function over the entire region at one time. Throughout this paper, we confine ourselves to regularized boosting (*e.g.*, Mason et al., 1999) although the localization based on our idea may apply to an ordinary version of boosting.

We discuss the theoretical aspects of the local boosting from the viewpoint of the Bayes

risk consistency. Following the discussion of Lugosi and Vayatis (2004) on regularized boosting, we prove the theorem that states that the local boosting also has the Bayes risk consistency. The discussion on the theorem provides a useful understanding for comparing the local boosting with the ordinary boosting with respect to estimation error and approximation error. The estimation errors of both boosting methods are bounded in slightly different ways, but both decrease to zero asymptotically if the class of base classifiers have a finite VC dimension. Therefore, the Bayes risk consistency depends on whether their approximation errors decrease to zero. Inspection of the proof of that theorem indicates that the local boosting may reduce the approximation error considerably at the cost of the increase in estimation error. As a result, the local boosting is shown to have the Bayes risk consistency in wider situations than those of the usual boosting. A simulation study confirmed several theoretical results and demonstrated that the local AdaBoost overperformed AdaBoost in several situations where examples were simulated from probabilistic settings with strongly nonlinear or locally linear decision boundaries. We find it worth noting that our proposal is easily applied to AdaBoost.M2, which was developed for the multiclass classification by Freund and Schapire (1997). It was also demonstrated that a local AdaBoost.M2 overcame difficulties that faced AdaBoost.M2 in the simulation study.

This thesis is organized as follows. In Section 2, we set up our problem and then describe conventional statistical classification methods and their properties. Some theorems from PAC (Probably Approximately Correct) learning theory are also introduced in this chapter. In addition, we give geometrical interpretation of boosting methods. Analysis of shark bycatch data by AdaBoost with decision stumps, the logistic GAM and AsymBoost is shown in Section 3. We introduce score plots that are graphical displays of the dependence of the occurrence of shark bycatch on individual features in this chapter. Several merits of AdaBoost with decision stumps are illustrated. In addition, we show that the application of AsymBoost decreases FNR at cost of slight increase of test error. These results were published in Kawakita et al. (2005). In Section 4, we derive the local boosting algorithm and discuss its statistical properties. We prove the Bayes risk consistency of the local boosting in the framework of PAC learning. Inspection of its proof elucidates the difference between the local boosting and the ordinary boosting. Simulation studies illustrate the performance of the local AdaBoost compared with that of AdaBoost. These results were summarized by Kawakita and Eguchi (2005). Finally, some concluding

remarks are given in Section 5.

2 Survey on ensemble learning

In this chapter, we first introduce some notations and set up our problem. Second, we describe some theorems or definitions, which are used in later chapters. Then, several ensemble learning methods and their properties are discussed. We first review *bagging* (Breiman, 1996a) that constructs a strong classifier by *resampling*. Second, several conventional boosting methods are reviewed including U-Boost, regularized U-Boost, Asym-Boost. Boosting constructs a strong classifier by *reweighting* (adaptively resampling) (Breiman, 1998). Both methods decrease variance by combining base classifiers.

2.1 Setup and Notations

We use notations similar to those of Lugosi and Vayatis (2004) in this paper. Let (X, Y) be a pair of random variables taking values in $\mathcal{X} \times \mathcal{Y}$. $\mathcal{X}$ is a feature space in an M-dimensional Euclidean space. The label set $\mathcal{Y}$ is $\{-1, 1\}$ in binary case and is $\{1, 2, \cdots, G\}$ in multiclass case. We denote the m-th feature of feature vector $X \in \mathcal{X}$ by $(X)_m$ in the remainder of this thesis. For a given *training data* set, $D = \{(X_i, Y_i)\}_{i=1}^n$, consisting of n independent, identically distributed pairs having the same distribution as (X, Y), one is asked to construct an accurate classifier $g_n : \mathcal{X} \to \mathcal{Y}$. The probability of misclassification (generalization error) defined as

$$L(g_n) = P(g_n(X) \neq Y | D) \tag{1}$$

measures the performance of g_n. Let us denote the set of all measurable classification functions mapping $\mathcal{X}$ to $\mathcal{Y}$ by $\mathcal{G}$. In binary case, we often use a classifier of the form $\mathrm{sign}(F)$ where F is some function mapping from $\mathcal{X}$ to R, which is called *discriminant function*, sign takes 1 if its argument is non-negative and takes -1 otherwise. Denote the set of all discriminant functions mapping $\mathcal{X}$ to R by $\mathcal{F}$. The infimum of L over all classification function is obtained by the Bayes classifier,

$$g^*(x) = \underset{y \in \{1, 2, \cdots, G\}}{\mathrm{argmax}} P(Y = y \,|\, x). \tag{2}$$

Specifically, in binary case, g^* is written as

$$g^*(x) = I(\eta(x) \geq 1/2) - I(\eta(x) < 1/2), \tag{3}$$

remarks are given in Section 5.

2 Survey on ensemble learning

In this chapter, we first introduce some notations and set up our problem. Second, we describe some theorems or definitions, which are used in later chapters. Then, several ensemble learning methods and their properties are discussed. We first review *bagging* (Breiman, 1996a) that constructs a strong classifier by *resampling*. Second, several conventional boosting methods are reviewed including U-Boost, regularized U-Boost, Asym-Boost. Boosting constructs a strong classifier by *reweighting* (adaptively resampling) (Breiman, 1998). Both methods decrease variance by combining base classifiers.

2.1 Setup and Notations

We use notations similar to those of Lugosi and Vayatis (2004) in this paper. Let (X, Y) be a pair of random variables taking values in $\mathcal{X} \times \mathcal{Y}$. $\mathcal{X}$ is a feature space in an M-dimensional Euclidean space. The label set $\mathcal{Y}$ is $\{-1, 1\}$ in binary case and is $\{1, 2, \cdots, G\}$ in multiclass case. We denote the m-th feature of feature vector $X \in \mathcal{X}$ by $(X)_m$ in the remainder of this thesis. For a given *training data* set, $D = \{(X_i, Y_i)\}_{i=1}^n$, consisting of n independent, identically distributed pairs having the same distribution as (X, Y), one is asked to construct an accurate classifier $g_n : \mathcal{X} \to \mathcal{Y}$. The probability of misclassification (generalization error) defined as

$$L(g_n) = P(g_n(X) \neq Y | D) \tag{1}$$

measures the performance of g_n. Let us denote the set of all measurable classification functions mapping $\mathcal{X}$ to $\mathcal{Y}$ by $\mathcal{G}$. In binary case, we often use a classifier of the form $\mathrm{sign}(F)$ where F is some function mapping from $\mathcal{X}$ to R, which is called *discriminant function*, sign takes 1 if its argument is non-negative and takes -1 otherwise. Denote the set of all discriminant functions mapping $\mathcal{X}$ to R by $\mathcal{F}$. The infimum of L over all classification function is obtained by the Bayes classifier,

$$g^*(x) = \underset{y \in \{1, 2, \cdots, G\}}{\mathrm{argmax}} P(Y = y \,|\, x). \tag{2}$$

Specifically, in binary case, g^* is written as

$$g^*(x) = I(\eta(x) \geq 1/2) - I(\eta(x) < 1/2), \tag{3}$$

where I denotes an indicator function, *i.e.*,

$$I(A) = \begin{cases} 1 & \text{if } A \text{ is true} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

and $\eta(x)$ is the posterior probability $P(Y=1 \mid x)$. The proof appear in Appendix A. The probability of misclassification of this Bayes classifier is calculated as

$$L(g^*) = E \min\{\eta(X), 1 - \eta(X)\},$$

where E denotes the expectation taken under true distribution of X. We denote $L(g^*)$ by L^*, which is referred to as the *Bayes risk*. Our goal is to construct a classifier that has a probability of misclassification sufficiently close to L^*. We say that classification method has the *Bayes risk consistency* if its probability of misclassification converges to L^* as $n \to \infty$. In general, however, the probability of misclassification L of classifier g_n is not available since the underlying distribution of (X, Y) is unknown. Instead, we often minimize an empirical version of probability of misclassification, defined as

$$L_D(g_n) = \frac{1}{n} \sum_{i=1}^{n} I(g_n(X_i) \neq Y_i). \tag{5}$$

The empirical probability of misclassification $L_D(g_n)$ usually underestimates the prediction error $L(g_n)$ in general since the same data set is used for evaluation and training g_n. To evaluate the performance of g_n, we prepare sufficient *test data* that are independent and identically distributed with D. Then, we may approximate $L(g)$ by Eq. (5) over test data. We call Eq. (5) *training error* if it is computed over training data and call *test error* if it is computed over test data.

2.2 Some preliminaries

We introduce several theorems that are often used in the framework of PAC learning theory. There are many related references (Chernoff, 1952; Hoeffding, 1963; Ledoux and Talagrand, 1991; McDiarmid, 1989; Devroye and Lugosi, 2001).

2.2.1 Concentration inequalities

We evaluate the probability of the difference between a random variable and its expectation. Classical statistics develop some inequalities for bounding this probability. Note that we assume that any moment of random variable exists when necessary.

Let us begin with Markov's inequality.

Theorem 1 (Markov's inequality). *Let X be a nonnegative random variable. For any $\epsilon > 0$, we have*

$$P(X \leq \epsilon) \leq \frac{E[X]}{\epsilon}.$$

Proof. For simplicity, assume that X has a probability density function, denoted as $P(x)$. Then, we have

$$\begin{aligned} E[X] &= \int_0^\infty xP(x)dx = \int_0^\epsilon xP(x)dx + \int_\epsilon^\infty xP(x)dx \\ &\geq \int_\epsilon^\infty xP(x)dx \\ &\geq \epsilon \int_\epsilon^\infty P(x)dx \\ &= \epsilon P(X \leq \epsilon). \end{aligned}$$

□

Markov's inequality induces Chebyshev's inequality.

Theorem 2 (Chebyshev's inequality). *Let X be an arbitrary random variable. For any $\epsilon > 0$, we have*

$$P(|X - E[X]| \geq \epsilon) \leq \frac{Var[X]}{\epsilon^2}.$$

Proof. Clearly,

$$P(|X - E[X]| \geq \epsilon) = P(|X - E[X]|^2 \geq \epsilon^2).$$

Applying Markov's inequality to the nonnegative random variable $|X - E[X]|^2$, we have

$$P(|X - E[X]| \geq \epsilon) \leq E[(X - E[X])^2]/\epsilon^2.$$

□

We give an example. Let $\{X_i\}_{i=1}^n$ be independently and identically distributed (i.i.d.) random samples generated from distribution of X and S_n be the sample mean, *i.e.*, $S_n = (1/n)\sum_{i=1}^n X_i$. If $E[X]$ exists, we have, by Chebyshev's inequality,

$$P(|S_n - E[X]| \geq \epsilon) \leq \frac{\mathrm{Var}(S_n)}{\epsilon^2} = \frac{\mathrm{Var}(X)}{n\epsilon^2}.$$

To illustrate the weakness of Chebyshev's bound, assume that each X_i is i.i.d. Bernoulli(p) random variable (*i.e.*, $P(X = 1) = p = 1 - P(X = 0)$). In this case,

$$P(|S_n - p| \geq \epsilon) \leq \frac{p(1-p)}{n\epsilon^2}.$$

Let $\Phi(x) = \int_{-\infty}^{x} \exp(-t^2/2)/\sqrt{2\pi}dt$ be a normal distribution function. The center limit theorem (See Appendix G) implies that

$$P\left(\sqrt{\frac{n}{p(1-p)}}(S_n - p) \leq x\right) \to 1 - \Phi(x) \leq \frac{1}{\sqrt{2\pi}}\frac{\exp(-x^2/2)}{x}.$$

The last inequality follows from Lemma 3.

Lemma 3. *For any x,*

$$1 - \Phi(x) \leq \frac{1}{\sqrt{2\pi}}\frac{e^{-\frac{x^2}{2}}}{x}.$$

Proof. The left-hand side of the statement is greater than or equal to

$$\int_x^\infty \frac{1}{\sqrt{2\pi}}\ e^{-\frac{t^2}{2}}dt.$$

The right-hand side can be rewritten as

$$-\int_x^\infty -\frac{1}{\sqrt{2\pi}}(1 + \frac{1}{t^2})\ e^{-\frac{t^2}{2}}dt.$$

Thus,

$$\frac{1}{\sqrt{2\pi}}\frac{e^{-\frac{x^2}{2}}}{x} - (1 - \Phi(x)) \geq \int_x^\infty \frac{1}{\sqrt{2\pi}}\ \frac{e^{-\frac{t^2}{2}}}{t^2}dt \geq 0.$$

□

Therefore, by taking $\epsilon = x\sqrt{p(1-p)/n}$,

$$P(S_n - p \geq \epsilon) \to \frac{\sqrt{p(1-p)}}{\sqrt{2\pi n}}\frac{\exp(-n\epsilon^2/2p(1-p))}{\epsilon}.$$

We similarly obtain

$$P(S_n - p \geq -\epsilon) \to \frac{\sqrt{p(1-p)}}{\sqrt{2\pi n}}\frac{\exp(-n\epsilon^2/2p(1-p))}{\epsilon}.$$

This indicates that $P(|S_n - p| \geq \epsilon)$ decreases exponentially as $n \to \infty$. Clearly, Chebyshev's inequality is off the mark since its order is $1/n$.

An improvement on the upperbound is obtained by Chernoff's method. By Markov's inequality, we have

$$P(X \geq \epsilon) = P(\exp(\theta X) \geq \exp(\theta\epsilon)) \leq E[\exp(\theta X)]/\exp(\theta\epsilon)$$

for any random variable X, any $\theta > 0$, and any $\epsilon > 0$. Thus, replacing X with $S_n - E[S_n]$, we have

$$\begin{aligned} P(S_n - E[S_n] \geq \epsilon) &\leq \exp(-\theta\epsilon) E\Big[\exp\Big(\theta \sum_{i=1}^{n} (X_i - E[X_i])\Big)\Big] \\ &= \exp(-\theta\epsilon)\Pi_{i=1}^{n} E[\exp(\theta(X_i - E[X_i]))] \end{aligned}$$

To obtain tight bounds, we have to minimize the right-hand side of this inequality with respect to θ. Note that the search of such θ reduces to finding a tight upperbound for the moment generating function of the random variables $X_i - E[X_i]$. Hoeffding gave an elegant bound by using the following lemma.

Lemma 4. *Let X be a random variable such that $E[X] = 0$ and X takes its values on the interval $[a, b]$. Then, for any $\theta > 0$,*

$$E[\exp(\theta X)] \leq \exp(\frac{\theta^2 (b-a)^2}{8}).$$

Proof. Since X takes values on $[a, b]$, X can be written as

$$X = \tau a + (1 - \tau) b$$

where τ is random variable taking values on the interval $[0, 1]$. Due to the convexity of the exponential function, we have

$$\begin{aligned} E[\exp(\theta X)] &= E[\exp(\theta\tau a + \theta(1-\tau)b)] \\ &\leq E[\tau \exp(\theta a) + (1-\tau)\exp(\theta b)] \\ &= \exp(\theta a) E[\tau] + \exp(\theta b) E[1-\tau] \end{aligned}$$

for any fixed $\theta > 0$. Since $E[X] = 0$, we have

$$E[X] = aE[\tau] + bE[1-\tau] = aE[\tau] + b(1 - E[\tau]) = (a-b)E[\tau] + b = 0.$$

Then, $E[\tau] = b/(b-a)$. Denote $1 - E[\tau]$ by h, *i.e.*, $h = -a/(b-a)$. Therefore, we have

$$\begin{aligned} E[\exp(\theta X)] &\leq \exp(\theta a)(1-h) + \exp(\theta b) h \\ &= \{1 - h + \exp(\theta(b-a))h\} \exp(\theta a) \\ &= \{1 - h + h\exp(\theta(b-a))\} \exp(-\theta h(b-a)) \\ &= \exp(-\theta h(b-a) + \ln(1 - h + h\exp(\theta(b-a)))) \end{aligned}$$

Denote the argument of the exponential function in the last equality as ψ and $\theta(b-a)$ by ν, *i.e.*, $\psi(\nu) = -h\nu + \ln(1-h+he^{\nu})$. By Taylor-expansion around zero, we have

$$\psi(\nu) = \psi(0) + \psi'(0)\nu + \psi''(\xi)\frac{\nu^2}{2}$$

where $\xi \in [0, \nu]$.

$$\begin{aligned}
\psi(0) &= 0 \\
\psi'(0) &= -h + \frac{he^{\nu}}{1-h+he^{\nu}}\bigg|_{\nu=0} = 0 \\
\psi''(\xi) &= \frac{(he^{\nu})(1-h+he^{\nu}) - h^2e^{2\nu}}{(1-h+he^{\nu})^2}\bigg|_{\nu=\xi} \\
&= \frac{(he^{\nu})(1-h)}{(1-h+he^{\nu})^2}\bigg|_{\nu=\xi} \\
&= \{(\frac{e^{-\nu}}{h} + \frac{1}{1-h})(1-h+he^{\nu})\}^{-1}\bigg|_{\nu=\xi} \\
&= \{2 + \frac{1-h}{h}e^{-\nu} + \frac{h}{1-h}e^{\nu}\}^{-1}\bigg|_{\nu=\xi} \\
&\leq \frac{1}{4}.
\end{aligned}$$

The second last inequality is due to the inequality of arithmetic-geometric mean. Thus, we have

$$\psi(\nu) \leq \nu^2/8 = \theta^2(b-a)^2/8.$$

□

Combined with Chernoff's bound, Lemma 4 immediately leads to Hoeffding's inequality.

Theorem 5 (Hoeffding's inequality). *Let $\{X_i\}_{i=1}^n$ be i.i.d. random variables such that each X_i takes its value on the interval $[a_i, b_i]$ with probability one. Then for any $\epsilon > 0$, we have*

$$P(S_n - E[S_n] \geq \epsilon) \leq \exp(-2\epsilon^2 / \sum_{i=1}^{n}(b_i - a_i)^2)$$

and

$$P(S_n - E[S_n] \leq -\epsilon) \leq \exp(-2\epsilon^2 / \sum_{i=1}^{n}(b_i - a_i)^2).$$

Note that Chernoff (1952) and Okamoto (1958) proved this theorem for binomial random variables.

The following lemma is often useful by combining Lemma 4.

Lemma 6. *Let $\sigma > 0$, $n \geq 2$. Let $\{X_1, X_2, \cdots, X_n\}_{i=1}^n$ be arbitrary random variables satisfying $E[\exp(\theta X_i)] \leq \exp(\theta^2\sigma^2/2)$ for all $1 \leq i \leq n$ and $\theta > 0$. Then, we have*

$$E[\max_{1\leq i\leq n} \leq \sigma\sqrt{2\ln n}].$$

If, in addition, $E[\exp(\theta(-X_i))] \leq \exp(\theta^2\sigma^2/2)$ for all $1 \leq i \leq n$ and $\theta > 0$. Then, for any $1 \leq n$,

$$E[\max_{i\in\{1,2,\cdots,n\}} |X_i|] \leq \sigma\sqrt{2\ln(2n)}.$$

Proof. Due to the Jensen's inequality, we have

$$\begin{aligned}
\exp(\theta E[\max_{1\leq i\leq n} X_i]) &\leq E[\exp(\theta \max_{1\leq i\leq n} X_i)] \\
&= E[\max_{1\leq i\leq n} \exp(\theta X_i)] \\
&\leq \sum_{i=1}^n E[\exp(\theta X_i)] \\
&\leq n\exp(\theta^2\sigma^2/2).
\end{aligned}$$

Taking the logarithm of both side of this equation, we have

$$\begin{aligned}
\theta E[\max_{1\leq i\leq n} X_i] &\leq \ln n + \frac{\theta^2\sigma^2}{2}, \\
E[\max_{1\leq i\leq n} X_i] &\leq \frac{\ln n}{\theta} + \frac{\theta\sigma^2}{2}.
\end{aligned}$$

The parameter θ_* minimizing the right-hand side of this equation satisfies

$$-\ln n\theta_*^{-2} + \frac{\sigma^2}{2} = 0.$$

Thus, $\theta_* = \sqrt{\frac{2\ln n}{\sigma^2}}$ minimizes the right-hand side and yields the first inequality. The second inequality is obtained by applying the first inequality to $\{X_1, -X_1, X_2, -X_2, \cdots, X_n, -X_n\}$ since $\max_{1\leq i\leq n} |X_i| = \max_{1\leq i\leq n}\{X_1, -X_1, X_2, -X_2, \cdots, X_n, -X_n\}$. □

McDiarmid (1989) extended Hoeffding's inequality to general functions of independent random variables. First, we introduce the definition of the bounded difference condition.

Definition 7 (Bounded difference condition). *Let A be some set and g be a function mapping A^n to R. We say that g satisfies the bounded difference condition if there exists $\{c_i\}_{i=1}^n$ such that*

$$\sup_{x_1,x_2,\cdots,x_n,x_i'\in A} |g(x_1,x_2,\cdots,x_n) - g(x_1,\cdots,x_{i-1},x_i',x_{i+1},\cdots,x_n)| \leq c_i.$$

This condition implies that, when we change the i-th variable of g while all the other variables are kept, we cannot change the value of the function satisfying the bounded difference condition by more than c_i. Under this condition, we have the following useful theorem (McDiarmid, 1989; Devroye et al., 1996).

Theorem 8 (McDiarmid's inequality). *Let $\{X_i\}_{i=1}^n$ be random variables taking their values on A. Assume that a function g satisfies the bounded difference condition. Then, for all $\epsilon > 0$,*

$$P(g(X_1,\, X_2,\, \cdots,\, X_n) - E[g(X_1,\, X_2,\, \cdots,\, X_n)] \leq \epsilon) \leq \exp(-2\epsilon^2/\sum_{i=1}^n c_i^2),$$
$$P(g(X_1,\, X_2,\, \cdots,\, X_n) - E[g(X_1,\, X_2,\, \cdots,\, X_n)] \geq \epsilon) \leq \exp(-2\epsilon^2/\sum_{i=1}^n c_i^2).$$

Proof. Define $V = g - Eg$ and $V_i = E[g \,|\, X_1,\, X_2,\, \cdots,\, X_i] - E[g \,|\, X_1,\, X_2,\, \cdots,\, X_{i-1}]$ for $i = 1,\, 2,\, \cdots,\, n$. Then, clearly $V = \sum_{i=1}^n V_i$. Define $H_i(X_1,\, X_2,\, \cdots,\, X_i) = E[g \,|\, X_1,\, X_2,\, \cdots,\, X_i]$. Then, for any i,

$$V_i = H_i(X_1,\, X_2,\, \cdots,\, X_i) - \int_A H_i(X_1,\, X_2,\, \cdots,\, X_{i-1}, x) F_i(dx)$$

where F_i denotes the distribution of X_i. Then, due to the bounded difference condition, we have

$$\sup_{X_i} V_i - \inf_{X_i'} V_i = \sup_{X_i} \sup_{X_i'} \{H_i(X_1,\, X_2,\, \cdots,\, X_{i-1}, X_i) - H_i(X_1,\, X_2,\, \cdots,\, X_{i-1}, X_i')\} \leq c_i.$$

for each i. Therefore, lemma 4 implies that, for all $i = 1,\, 2,\, \cdots,\, n$,

$$E[\exp(\theta V_i \,|\, X_1,\, X_2,\, \cdots,\, X_{i-1})] \leq \exp(\theta^2 c_i^2/8).$$

Chernoff's bound is obtained for any $\theta > 0$,

$$\begin{aligned}
P(g - Eg \geq \epsilon) &\leq E[\exp(\theta V)]\exp(-\theta\epsilon) \\
&= E\Big[\exp\Big(\theta\sum_{i=1}^{n} V_i\Big)\Big]\exp(-\theta\epsilon) \\
&= E\Big[\exp\Big(\theta\sum_{i=1}^{n-1} V_i\Big)\Big]E[\exp(\theta V_n)\,|\,X_1,\,X_2,\,\cdots,\,X_{n-1}]\exp(-\theta\epsilon) \\
&\leq E\Big[\exp\Big(\theta\sum_{i=1}^{n-1} V_i\Big)\Big]\exp\Big(\frac{\theta^2 c_n^2}{8} - \theta\epsilon\Big) \\
&\vdots \\
&\leq \exp(\sum_{i=1}^{n}(\frac{\theta^2 c_i^2}{8}) - \theta\epsilon)
\end{aligned}$$

by repeating n times application of Lemma 4. The selection $\theta = 4\epsilon/\sum_{i=1}^{n} c_i^2$ minimizes the right-hand side and proves the first inequality. The proof of the second inequality is similar. □

We may deduce the following inequality directly from McDirmid's inequality.

$$P(|g(X_1,\,X_2,\,\cdots,\,X_n) - E[g(X_1,\,X_2,\,\cdots,\,X_n)]| \geq \epsilon) \leq 2\exp\Big(-2\epsilon^2/\sum_{i=1}^{n} c_i^2\Big)$$

2.2.2 Class of base classifiers and its VC dimension

Let $\mathcal{H}$ be a family of several subsets of R^M.

Definition 9 (VC shatter coefficient). *The VC shatter coefficient of $\mathcal{H}$ is defined as*

$$\mathcal{S}_{\mathcal{H}}(n) = \max_{x_1,\,x_2,\,\cdots,\,x_n \in R^M} |\{(x_1,\,x_2,\,\cdots,\,x_n) \cap H \,|\, H \in \mathcal{H}\}|.$$

Proposition 10 (Properties of VC shatter coefficient). *Let H be a family of subsets in R^M. The VC shatter coefficient of $\mathcal{H}$, $\mathcal{S}_{\mathcal{H}}(n)$, satisfies:*

(a) $\mathcal{S}_{\mathcal{H}}(n) \leq 2^n$

(b) *For any $n < n'$,* $\mathcal{S}_{\mathcal{H}}(n') \leq \mathcal{S}_{\mathcal{H}}(n)$

(c) $\mathcal{S}_{\mathcal{H}}(n+m) \leq \mathcal{S}_{\mathcal{H}}(n)\mathcal{S}_{\mathcal{H}}(m)$

(d) $\mathcal{S}_{\mathcal{H}\cup\mathcal{B}}(n) \leq \mathcal{S}_{\mathcal{H}}(n) + \mathcal{S}_{\mathcal{B}}(n)$

(e) *Let* $\mathcal{H}^c = \{A^c \,|\, A \in \mathcal{H}\}$. $\mathcal{S}_{\mathcal{H}^c} = \mathcal{S}_{\mathcal{H}}$.

These properties follow obviously from their definitions.

VC dimension of $\mathcal{H}$, denoted as V, is defined as the largest number n such that $\mathcal{S}_{\mathcal{H}}(n) = 2^n$. If, for any n, $\mathcal{S}_{\mathcal{H}} = 2^n$, then V is defined as $V = \infty$. In statistical classification, VC dimension can be interpreted as follows. Let us consider a binary classification, *i.e.*, $\mathcal{Y} = \{-1, 1\}$. Assume that $\mathcal{C}$ is a set of available classifiers, *i.e.*,

$$\mathcal{C} = \{f_j : \mathcal{X} \to \mathcal{Y} \,|\, j = 1, 2, \cdots, n\}.$$

Define H_f as $\{x \in \mathcal{X} \,|\, f(x) = 1\}$ for each $f \in \mathcal{C}$ and $\mathcal{H}$ as $\{H_f \,|\, f \in \mathcal{C}\}$. V is interpreted as the largest number of samples such that we may find the classifier that can classify all samples correctly for any labelling of $\{x_1, x_2, \cdots, x_n\}$.

The following lemma given by Sauer (1972) is an example illustrating the usefulness of VC dimension.

Lemma 11 (Sauer's lemma). *Let* $\mathcal{A}$ *be a family of some subsets of* R^M *with VC dimension* $V < \infty$. *Then, for all* n,

$$\mathcal{S}_{\mathcal{A}}(n) \leq \sum_{i=0}^{V} \left(\begin{array}{c} n \\ i \end{array} \right).$$

Before jumping the proof of this lemma, we introduce some notations. For fixed $\{x_1, x_2, \cdots, x_n\}$, the finite set $\mathcal{A}(\boldsymbol{x}_n)$ is defined as

$$\mathcal{A}(\boldsymbol{x}_n) = \{(b_1, b_2, \cdots, b_n) \in \{0, 1\}^n \,|\, \exists A \in \mathcal{A}, b_i = I(x_i \in A)\,,\, i = 1, 2, \cdots, n\},$$

where $\boldsymbol{x}_n = \{x_1, x_2, \cdots, x_n\}$. Then, the shatter coefficient of $\mathcal{A}$ is rewritten as

$$\mathcal{S}_{\mathcal{A}}(n) = \max_{x_1, x_2, \cdots, x_n \in R^M} |\mathcal{A}(\boldsymbol{x}_n)|.$$

Definition 12. *Let* B *be an arbitrary subset of* $\{0, 1\}^n$. *We denote an arbitrary subset of* $\{1, 2, \cdots, n\}$ *by* $S = \{s(1), s(2), \cdots, s(n')\}$. *Note that we denote by* S *and* s *different things in other chapters. We say that* B *shatters a set* $S = \{s(1), s(2), \cdots, s(n')\} \subset \{1, 2, \cdots, n\}$ *if the restriction of* B *to the components* $\{s(1), s(2), \cdots, s(n')\}$ *is the full* n' *dimensional binary hypercube, i.e.,*

$$\{(b_{s(1)}, b_{s(2)}, \cdots, b_{s(n')}) \,|\, b = (b_1, b_2, \cdots, b_n) \in B\} = \{0, 1\}^{n'}.$$

Using these notations, the proof of Lemma 11 is given below.

Proof. Fix $\boldsymbol{x}_n$ such that $\mathcal{S}_{\mathcal{A}}(n) = \max_{x_1, x_2, \cdots, x_n \in R^M} |\mathcal{A}(\boldsymbol{x}_n)|$. Denote $B_0 = \mathcal{A}(\boldsymbol{x}_n)$. It suffices to show that the cardinality of any set B_0 that cannot shatter any set of size $n' > V$, is at most $\sum_{i=0}^{V} \left(\begin{array}{c} n \\ i \end{array} \right)$. To show this, we transform B_0 into a set B_n with $|B_n| = |B_0|$ such that any set shattered by B_n is also shattered by B_0.

For every vector $b = (b_1, b_2, \cdots, b_n) \in B_0$, if $b_1 = 1$, then flip b_1 to zero unless $(0, b_2, b_3, \cdots, b_n)$ is already in B_0. Keep b unchanged if $b_1 = 0$. Clearly, the set B_1 of vectors obtained in this way has the same cardinality as B_0. In addition, if B_1 shatters a set $S = \{s(1), s(2), \cdots, s(n')\} \subset \{1, 2, \cdots, n\}$, B_0 also shatters S. If $1 \notin S$, this is trivial. If $1 \in S$, then we assume that $s(1) = 1$ without loss of generality. The fact that B_1 shatters S implies that, for any $v \in \{0,1\}^{n'-1}$, there exists a $b \in B_1$ such that $b_1 = 1$ and $(b_{s(2)}, \cdots, b_{s(n')}) = v$. By the construction of B_1 this is possible only if, for any vector $v \in \{0,1\}^{\{n'-1\}}$, both the vectors $(0, b_2, \cdots, b_n)$ and $(1, b_2, \cdots, b_n)$ where $(b_{s(2)}, \cdots, b_{s(n')}) = v$ are in B_0. This means that B_0 also shatters S.

Next, execute the same transformation of B_1 on the second component of each vector. That is, for each vector $b \in B_1$, if $b_2 = 1$, flip b_2 to zero unless $(b_1, 0, b_2, \cdots, b_n)$ is in B_1. Then, the obtained set B_2 also has the same cardinality, and any set shattered by B_2 is also shattered by B_1.

Repeating this transformation over all components, we obtain B_n such that B_n does not shatter sets of cardinality larger than V, since otherwise B_0 would necessarily shatter sets of the same size. In addition, it is easily seen that, for any vector $b \in B_n$, all vectors of the form $c = (c_1, c_2, \cdots, c_n)$ such that $c_i \in \{b_i, 0\}$. Then, B_n is a subset of the set

$$\mathcal{T} = \{b \in \{0,1\}^n \,|\, b \text{ has at most } V \text{ ones}\}.$$

The cardinality of $\mathcal{T}$ is clearly $\sum_{i=0}^{V} \left(\begin{array}{c} n \\ i \end{array} \right)$. The statement follows from

$$\mathcal{S}_{\mathcal{A}}(n) = |B_0| = |B_n| \leq |\mathcal{T}| = \sum_{i=0}^{V} \left(\begin{array}{c} n \\ i \end{array} \right).$$

□

We note that in the proof of Lemma 11 it does not hold that any set shattered by B_0 is necessarily shattered by B_1 (and therefore also by B_n).

The following proposition gives a more transparent upperbound of shatter coefficient.

Proposition 13. *Let $\mathcal{A}$ be a family of some subsets of R^M. Assume that $\mathcal{A}$ has a VC dimension $V < \infty$. Then,*

$$\mathcal{S}_{\mathcal{A}}(n) \leq (n+1)^V,$$

and for all $n \geq V$,

$$\mathcal{S}_{\mathcal{A}}(n) \leq \left(\frac{ne}{V}\right)^V.$$

Proof. From Lemma 11, we have

$$\mathcal{S}_{\mathcal{A}}(n) \leq \sum_{i=1}^{V} \left(\begin{array}{c} n \\ i \end{array} \right).$$

Thus, we have

$$\begin{aligned} (n+1)^V &\geq \sum_{i=0}^{V} \left(\begin{array}{c} V \\ i \end{array} \right) n^i \\ &\geq \sum_{i=1}^{V} \frac{V!}{i!(V-i)!} n^i \geq \sum_{i=1}^{V} \frac{n^i}{i!} \geq \sum_{i=1}^{V} \frac{n^i}{i!(n-i)!} \\ &= \sum_{i=1}^{V} \left(\begin{array}{c} n \\ i \end{array} \right) \geq \mathcal{S}_{\mathcal{A}}(n). \end{aligned}$$

If $V/n \leq 1$, then

$$\left(\frac{V}{n}\right)^V \sum_{i=0}^{V} \left(\begin{array}{c} n \\ i \end{array} \right) \leq \sum_{i=0}^{V} \left(\frac{V}{n}\right)^i \left(\begin{array}{c} n \\ i \end{array} \right) \leq \sum_{i=0}^{n} \left(\frac{V}{n}\right)^i \left(\begin{array}{c} n \\ i \end{array} \right) = \left(1 + \frac{V}{n}\right)^n \leq e^V.$$

The last inequality follows from Lemma 43 in Appendix. Therefore,

$$\mathcal{S}_{\mathcal{A}}(n) \leq \sum_{i=0}^{V} \left(\begin{array}{c} n \\ i \end{array} \right) \leq \left(\frac{ne}{V}\right)^V.$$

□

Vapnik (1982) showed an upperbound of generalization error. Before jumping to the theorem, we introduce some notations. Let $H_f = \{x \in R^M \mid f(x) = 1\}$ for each $f \in \mathcal{C}$. Define $\mathcal{H} = \{H_f \mid f \in \mathcal{C}\}$.

Theorem 14.

$$P\left(\sup_{f \in \mathcal{C}} |L_D(f) - L(f)| > \epsilon\right) \leq 6\mathcal{S}_{\mathcal{H}}(2n) \exp\left(-\frac{\epsilon^2 n}{4}\right).$$

2.2.3 Bias-Variance theory of a classifier

We overview the bias-variance theory discussed by Breiman (1998) in statistical classification. Consider a multiclass classification problem with a label set $\mathcal{Y} = \{1, 2, \cdots, G\}$. Now let D be a set of random variables. The probability of misclassification is redefined as

$$L(g) = P(g(X; D) \neq Y \,|\, D).$$

Write

$$Q(y \,|\, x) = P(g(x; D) = y)$$

for each $y \in \mathcal{Y}$. Then Q can be regarded as the probability of that g based on independent replicas of D assigns the label y at x. This corresponds to the bagged classifier (See Section 2.3.1). Then, the probability of misclassification of g is

$$L(g) = E\left[\int_{\mathcal{X}} \left\{\sum_{y \in \mathcal{Y}} (1 - Q(y \,|\, x)) P(Y = y \,|\, x)\right\} P(x) dx\right].$$

Clearly,

$$\sum_{y \in \mathcal{Y}} (1 - Q(y \,|\, x)) P(Y = y \,|\, x) = 1 - \sum_{y \in \mathcal{Y}} Q(y \,|\, x) P(Y = y \,|\, x) \geq 1 - \max_{y \in \mathcal{Y}} P(Y = y \,|\, x).$$

with equality if and only if

$$Q(y \,|\, x) = \begin{cases} 1 & \text{if } y = \operatorname{argmax}_{y' \in \mathcal{Y}} P(Y = y' \,|\, x) \\ 0 & \text{otherwise} \end{cases}.$$

Obviously, the Bayes classifier g^* leads to such $Q(y \,|\, x)$ and attains the minimum probability of misclassification:

$$L(g) = 1 - \int_{\mathcal{X}} \max_{y \in \mathcal{Y}} P(Y = y \,|\, x) P(x) dx.$$

Definition 15 (Unbiasedness of classifier). *Define an aggregated classifier* $g_A^*(x) = \operatorname{argmax}_{y \in \mathcal{Y}} Q(y \,|\, x)$. *$g(x; D)$ is unbiased at x if* $g_A^*(x) = g^*(x)$.

Note that the unbiasedness of $g(x; D)$ at x does not necessarily mean that g predicts a label of x accurately. For instance in binary classification, $P(Y = 1 \,|\, x) = 0.9$, $P(Y = 2 \,|\, x) = 0.1$ and $Q(Y = 1 \,|\, x) = 0.6$, $Q(Y = 2 \,|\, x) = 0.4$. Let $\mathcal{U}$ be the set of all x at which g is unbiased, *i.e.*,

$$\mathcal{U} = \{x \in \mathcal{X} \,|\, g_A^*(x) = g^*(x)\}.$$

and call $\mathcal{U}$ the *unbiased set*. The complement of $\mathcal{U}$ is denoted by $\overline{\mathcal{U}}$ and is called the *biased set*. Clearly the aggregated (bagged) classifier is the best (Bayes) classifier at any point $x \in \mathcal{U}$.

Definition 16 (Bias and variance). *The bias of a classifier $g(x; D)$ is defined as*

$$\text{Bias}(g) = P(g^*(X) = Y, X \in \overline{\mathcal{U}}) - E[P(g(X; D) = Y, X \in \overline{\mathcal{U}} \mid D)], \tag{6}$$

and its variance is defined as

$$\text{Var}(g) = P(g^*(X) = Y, X \in \mathcal{U}) - E[P(g(X; D) = Y, X \in \mathcal{U} \mid D)]. \tag{7}$$

These definitions lead to the *Bias-variance decomposition*:

$$L(g) = L^* + \text{Bias}(g) + \text{Var}(g). \tag{8}$$

This decomposition follows from the following observation:

$$\begin{aligned} L^* + \text{Bias}(g) + \text{Var}(g) &= (1 - P(g^*(X) = Y, X \in \mathcal{U})) + (1 - P(g^*(X) = Y, X \in \overline{\mathcal{U}})) \\ &\quad + P(g^*(X) = Y, X \in \overline{\mathcal{U}}) - E[P(g(X; D) = Y, X \in \overline{\mathcal{U}} \mid D)] \\ &\quad + P(g^*(X) = Y, X \in \mathcal{U}) - E[P(g(X; D) = Y, X \in \mathcal{U} \mid D)] \\ &= E[P(g(X; D) \neq Y, X \in \overline{\mathcal{U}} \mid D)] + E[P(g(X; D) \neq Y, X \in \mathcal{U} \mid D)] \\ &= E[P(g(X; D) \neq Y \mid D)] = L(g). \end{aligned}$$

Proposition 17 (Properties of bias and variance). *Bias and variance has the following properties:*

1. *Bias and variance are always nonnegative.*
2. *The variance of $g_A(x)$ is necessarily zero.*
3. *If $g(x; D)$ is deterministic, i.e., does not depend on D, then its variance is zero.*
4. *The bias of g^* is zero.*

The proofs of $1 - 4$ are immediate from definitions of bias and variance. Breiman (1996a) and Breiman (1998) established a clear discussions about bagging and arcing (boosting) from the view of bias-variance theory, which will be reviewed later.

Friedman (1997) gave a thoughtful analysis of the meaning of bias and variance in binary case. Other definitions of bias and variance in classification are given in (Kong and Dietterich, 1995; Kohavi and Wolpert, 1996; Tibshirani, 1996).

2.3 Bagging

Breiman (1996a) discussed the bagging predictors. Breiman (1996b) pointed out the instability of some recently developed methods, including neural networks, CART (Morgan and Sonquist, 1963; Breiman et al., 1984; Quinlan, 1993), and subset selection in linear regression. Bagging improve prediction performance of such instable methods.

2.3.1 Bagged predictors

Assume that we have a training data set D consisting of n samples. Statistical method constructs a predictor $g : \mathcal{X} \times \mathcal{X}^n \to \mathcal{Y}$ based on the information of D. Therefore, write g by $g(x; D)$. We denote the b-th bootstrapped data set from D by $D^{(b)}$ for $b = 1, 2, \cdots, B$. Then, the bagged classifier for regression is defined as

$$g_A(x) = \frac{1}{B} \sum_{b=1}^{B} g(x; D^{(b)}).$$

The bagged classifier for multiclass classification is defined as

$$g_A(x) = \underset{y \in \mathcal{Y}}{\operatorname{argmax}} \frac{1}{B} \sum_{b=1}^{B} I(g(x; D^{(b)}) = y).$$

2.3.2 Statistical aspects of bagging

Some theoretical observations explain how the aggregated predictor constructed by bagging works well.

Regression ($\mathcal{Y} = R$) Assume that the sample size n of D is sufficiently large such that $g_A^*(x) = E[g(x; D)]$ (recall that D is a set of random variables). For a fixed $(x, y) \in (\mathcal{X}, \mathcal{Y})$, we have

$$E[(y - g(x; D))^2] = y^2 - 2yE[g(x, D)] + E[g^2(x; D)].$$

By applying Jensen's inequality to the third term, *i.e.*, $E[g^2(x; D)] \geq E[g(x; D)]^2$, we have

$$E[(y - g(x; D))^2] \geq (y - g_A^*(x))^2.$$

Taking expectation of both sides with respect to the joint underlying distribution of (X, Y), the mean-squared error of g_A^* is equal to or less than that of g, *i.e.*,

$$E[(Y - g(X; D))^2] \geq E[(Y - g_A^*(X))^2].$$

The extent of improvement obtained by g_A^* depends on how unequal the two sides of

$$E[g^2(x;D)] \geq E[g(x;D)]^2 = g_A^*(x)^2.$$

The role of instability of predictor is clear. If $g(x;D)$ changes largely with replicate D, the aggregation improves $g(x;D)$ largely. Otherwise, the aggregation would not improve it so much. In any way, g_A^* always improve g.

The bagged predictor, however, is not equal to g_A^* in general since the underlying distribution is unknown. Indeed, the bagged predictor approximates $g_A^*(x)$ by bootstrap method. Thus, if $g(x;D)$ is stable predictor, bagging predictor may lead to worse prediction because of the uncertainty in the estimation of g_A^*. Thus, there may exist a trade-off between the stability and the estimation error.

Classification in $\mathcal{Y} = \{1, 2, \cdots, G\}$ In a classification problem, a predictor $g(x;D)$ predicts a label $y \in \mathcal{Y}$. Define an aggregated classifier $g_A(x;D) = \text{argmax}_{y \in \mathcal{Y}} P(g(x;D) = y)$. As was already seen, the aggregated classifier $g_A(x)$ decreases its variance, defined in Eq. (7), to zero. However, it is not guaranteed that the bias of g_A decreases. Therefore, if the unbiased set dominates in $\mathcal{X}$, the bagged classifier performs well. Otherwise, bagging classifier may make the prediction performance worse.

2.4 Boosting algorithm

Boosting methods combine (weak) base classifiers to obtain a strong classifier. A classifier is called a *weak classifier* if its error rate is slightly better than random guessing and is called a *strong classifier* if it is very accurate. We will use this terminology for the remainder of this thesis. Let $\mathcal{C}$ be a set of available base classifiers. We describe details of $\mathcal{C}$ in Section 2.4.1. In binary classification, for example, boosting constructs a resultant classifier such that $g(x) = \text{sign}(F(x))$, where $F(x)$ is found in the linear hull of $\mathcal{C}$ and sign denotes the sign of its argument. Denote the set of all linear combination of arbitrary classifiers in $\mathcal{C}$ by $\text{lin}(\mathcal{C})$. Note that $F(x)$ takes value on not only $\{-1, 1\}$. Thus, $F(x)$ is a discriminant function. In this case, our goal is to search the discriminant function, F, from $\text{lin}(\mathcal{C})$ that minimizes $L(\text{sign}(F'))$ in F'. The minimization of $L(\text{sign}(F))$ over a linear hull of $\mathcal{C}$, however, is an NP-hard problem in general (Höffgen et al., 1995). Instead, the classification algorithm often minimizes an increasing, differentiable, and convex function that places an upperbound on L. Let ϕ be a function mapping R to R. Instead of L,

define a loss function $A : \mathcal{F} \to R$ as

$$A(F) = E[\phi(-YF(X))].$$

Unless otherwise stated, we always assume that ϕ satisfies the following condition throughout this thesis.

Condition 18 (Conditions for cost function). *Let $\phi : R \to R$ be a cost function.*

1. *ϕ is strictly convex and strictly increasing.*
2. *$\phi(0) = 1$.*
3. *$\lim_{x \to -\infty} \phi(x) = 0$.*
4. *ϕ is differentiable.*

The first condition is essential. Without the second and third conditions, boosting algorithm is still Bayes risk consistent even though A is not an upperbound of L. Not few cost functions of existing methods satisfy Condition 18. If ϕ satisfies Condition 18, A is upperbound for L, *i.e.*, for any $F \in \mathcal{F}$,

$$L(F) \leq A(F)$$

since always $I(F(X) \neq Y) \leq \phi(-YF(X))$. Therefore, we may expect minimizing the probability of misclassification by these methods since their upperbounds are minimized. For example, AdaBoost is designed to find a classifier minimizing the exponential loss $\phi = \exp$, and F consists of a linear combination of $\mathcal{C}$. Because the underlying distribution of X and Y is unknown in an actual situation, boosting methods iteratively find a minimizer of the empirical version of the loss function,

$$A_D(F) = \frac{1}{n} \sum_{i=1}^{n} \phi(-Y_i F(X_i)). \tag{9}$$

Subsequent sections describe how boosting methods minimize the empirical loss function iteratively.

Let $F^* = \operatorname{argmin}_{F' \in \mathcal{F}} A(F')$, where the infimum is taken over all measurable functions F' mapping from $\mathcal{X}$ to R. We know that the classifier, $\operatorname{sign}(F^*(x))$, equals to the Bayes classifier, $g^*(x)$ (Section 2.5.2). Thus, boosting method may construct a Bayes classifier

that has a probability of misclassification L^* asymptotically if $F^*(x)$ is in $\mathrm{lin}(\mathcal{C})$, a scaled linear hull of $\mathcal{C}$. Otherwise, AdaBoost does not guarantee an accurate prediction, which sometimes occurs even when we use widely-used base classifiers. To overcome this situation, we propose a localized version of boosting that attains a higher approximation performance even together with the same base classifiers in Section 4.

2.4.1 Base classifiers

We describe base classifiers to be combined by boosting methods. First, we introduce several assumptions for a class of base classifiers. Let $\mathcal{C}$ be a set of available base classifiers:

$$\mathcal{C} = \{f_j : \mathcal{X} \to \mathcal{Y} \,|\, j = 1, 2, \cdots, J\}.$$

Definition 19 (Negation closedness). *We say that a class of base classifiers is negation closed if $-f \in \mathcal{C}$ for any base classifier f in $\mathcal{C}$.*

We assume that $\mathcal{C}$ is negation closed and has a finite VC dimension, denoted by V. Most widely-used base classifiers to be combined by boosting method is *decision stump*. Decision stump $f^{\mathrm{s}}(x; m, b, s)$ is a type of weak classifier and is defined as follows:

$$f^{\mathrm{s}}(x; m, b, s) = s \cdot \mathrm{sign}((x)_m - b), \tag{10}$$

where $m = 1, 2, \cdots, M$, $(x)_m$ is the m-th element of x, $b \in R$ is a threshold value, and s is a sign variable taking values of 1 or -1. It is easily seen that, for fixed values of s and b, the decision stump is a shifted step function that assigns x a label based on only the m-th feature $(x)_m$. $\mathcal{C}_{\mathrm{ds}}$ is defined as the set of $f^{\mathrm{s}}(x; m, b, s)$ for all possible combinations of m, s, and b. Note that the cardinality of $\mathcal{C}$ is infinity when b may take a value in R, but we often prepare finite candidates of b in practical use as follows. Given a training data set $D = \{(X_1, Y_1), (X_2, Y_2), \cdots, (X_n, Y_n)\}$, we prepare a collection of decision stumps for each feature $(X)_m$ in the following manner.

(a) Sort all unique values of the m-th feature $(X)_m$ as $\{(X_{i'})_m\}_{i'=1,2,\cdots n_m}$ where n_m is the number of unique values of the m-th feature;

(b) Find all mid-points between sequential pairs of points in this sorted collection;

(c) For each mid-point (indicated by b), prepare two candidate decision stumps $f^{\mathrm{s}}(x; m, b, 1)$ and $f^{\mathrm{s}}(x; m, b, -1)$ whose the discontinuity points b are at the mid-points.

Finally, we construct $\mathcal{C}_{\rm ds}$ by gathering all classifiers prepared in the step (c) for all $(x)_m$. Therefore J, the number of classifiers contained in $\mathcal{C}_{\rm ds}$, is $\sum_{m=1}^{M} 2(n_m - 1)$.

Any strong classifiers can be used as well as base classifiers in boosting. However, boosting with $\mathcal{C}_{\rm ds}$ has several advantages. Since decision stumps are simple classifiers, its computational cost is not much and $\mathcal{C}_{\rm ds}$ has a small VC dimension as described below. The upperbounds of generalization error in Section 2.5.4 or 4.2.2 indicates that, if a base classifier class $\mathcal{C}$ has a smaller VC dimension, boosting method attains sometimes a smaller generalization error since it avoids overfitting. Data analysis in Section 3.3 also demonstrated that it is unlikely that AdaBoost with $\mathcal{C}_{\rm ds}$ overfits to the training data. Another advantage is that predictions of boosting with $\mathcal{C}_{\rm ds}$ are invariant under any one-to-one transformation of each feature. Thus, centralization or normalization, which are often needed by conventional discrimination methods, become unnecessary. In addition, graphical displays of the contribution of individual feature to the value of discriminant function, *score plots*, that are introduced in Section 3.1 are easily constructed. Finally, boosting with decision stumps may perform well with correlated features. Considering these merits, we use *decision stumps* as classifiers to be combined by AdaBoost in all simulations in Section 3.3 and 4.3.

The following proposition indicates that $\mathcal{C}_{\rm ds}$ has a VC dimension that is necessarily finite and is relatively small in general.

Proposition 20 (Finiteness of VC dimension of $\mathcal{C}_{\rm ds}$). *The VC dimension of $\mathcal{C}_{\rm ds}$ is less than or equal to $\lfloor 2(1 + \log_2 M) \rfloor$.*

Proof. Denote the shatter coefficient of $\mathcal{C}_{\rm ds}$ by

$$\mathcal{S}_{\mathcal{C}_{\rm ds}}(n) = \max_{x_1, x_2, \cdots, x_n \in R^M} |\{\{x_1, x_2, \cdots, x_n\} \cap \{x \mid f^{\rm s}(x; m, s, b) = 1\} \mid f^{\rm s}(x; m, s, b) \in \mathcal{C}_{\rm ds}\}|.$$

The VC dimension of $\mathcal{C}_{\rm ds}$ is defined as

$$V = \max_{n'} \{n' \mid \mathcal{S}_{\mathcal{C}_{\rm ds}}(n') = 2^{n'}\}.$$

For each feature $(x)_m$ ($m = 1, 2, \cdots, M$), decision stumps yield at most $2n$ different subsets of $\{x_1, x_2, \cdots, x_n\}$. Thus, we have

$$\mathcal{S}_{\mathcal{C}_{\rm ds}}(n) \leq 2Mn.$$

V is less than any n satisfying that

$$2Mn < 2^n,$$

or equivalently,

$$1 + \log_2 M < n - \log_2 n.$$

For any positive integer n',

$$\frac{n'}{2} \leq n' - \log_2 n'.$$

Therefore, V is at most $2(1 + \log_2 M)$, *i.e.*, $V \leq \lfloor 2(1 + \log_2 M) \rfloor$. □

In multiclass case, we modify decision stumps slightly:

$$f^{\mathrm{s}}(x; m, s, b, g) = \begin{cases} g & \text{if } s \cdot \mathrm{sign}((x)_m - b) = 1 \\ \{1, 2, \cdots, G\} \backslash g & \text{otherwise} \end{cases},$$

where g may take values in $m \in \{1, 2, \cdots, M\}$ and s is a sign variable as defined before. We prepare such classifiers with b in the same manner as that in the binary case for all possible combinations of feature m, class g.

2.4.2 Ordinary boosting

The algorithm of the ordinary boosting with general loss function is described. Boosting method with general loss function has been discussed by Mason et al. (1999), Friedman et al. (2000), Murata et al. (2004), and so on. Literatures often interpret boosting method as general functional gradient descent algorithm. We also describe the algorithm of boosting method from this viewpoint.

Binary case The algorithm of the ordinary boosting algorithm iteratively minimizes its loss function by adding a linear term of base classifier. Let $F_0(x) \equiv 0$ be an initial discriminant function. For a given current discriminant function, F_{t-1}, AdaBoost chooses a new base classifier, f, and its coefficient, α, iteratively as follows.

$$f = \operatorname*{argmin}_{f' \in \mathcal{C}} A_D(F_{t-1} + \alpha' f') \quad \text{for any positive } \alpha' \tag{11}$$

$$\alpha = \operatorname*{argmin}_{\alpha' > 0} A_D(F_{t-1} + \alpha' f) \tag{12}$$

Then, the discriminant function is updated as

$$F_t(x) = F_{t-1}(x) + \alpha f(x). \tag{13}$$

The final classifier is obtained as $g(x) = \mathrm{sign}(F_T(x))$ after T repetitions in this process. Note that we may assume that α is always positive since $\mathcal{C}$ is negation closed.

The optimization in Eq. (11) should depend on two variables f' and α', which make boosting algorithm computationally infeasible. The following one-dimensional approximation removes this difficulty of the optimization in Eq. (11). By Taylor-expansion around $\alpha' = 0$, we have

$$\begin{aligned} f &= \operatorname*{argmin}_{f' \in \mathcal{C}} A_D(F_{t-1} + \alpha' f') \\ &\approx \operatorname*{argmin}_{f' \in \mathcal{C}} A_D(F_{t-1}) + \left. \frac{\partial A_D(F_{t-1} + \alpha' f')}{\partial \alpha'} \right|_{\alpha'=0} \alpha' \\ &= \operatorname*{argmin}_{f' \in \mathcal{C}} \sum_{i=1}^{n} \phi'(-Y_i F_{t-1}(X_i))(-Y_i f'(X_i))\alpha' \\ &= \operatorname*{argmin}_{f' \in \mathcal{C}} \sum_{i=1}^{n} \phi'(-Y_i F_{t-1}(X_i))(2I(Y_i \neq f'(X_i)) - 1). \\ &= \operatorname*{argmin}_{f' \in \mathcal{C}} \sum_{i=1}^{n} \phi'(-Y_i F_{t-1}(X_i)) I(Y_i \neq f'(X_i)). \end{aligned} \tag{14}$$

As a result, the optimization in Eq. (11) reduces to the optimization Eq. (14) that does not depend on α'.

We may not have the explicit solution α in the optimization in Eq. (12) in general. Therefore, some numerical optimization technique should be applied. The optimization in Eq. (12) is over one variable and thus many existing strong techniques may be used.

The summary of boosting algorithm with general loss function is described below.

---

1. Initialize weights on sample $\{w_0(i)\}_{i=1}^n$ as $1/n$ for each i and $F_0 \equiv 0$.

2. For $t = 1, 2, \cdots, T$, repeat the following process.

   (a) Find a new best classifier its coefficient, denoted as $f_{j(t)}$ and α_t, as follows:

$$j(t) = \operatorname*{argmin}_{j \in \{1,2,\cdots,J\}} \epsilon_t(f_j) \tag{15}$$

$$\alpha_t = \operatorname*{argmin}_{\alpha \in R_+} A_D(F_{t-1} + \alpha f_{j(t)})\} \tag{16}$$

   with $\alpha_1 = 1$ where the weighted error rate, $\epsilon_t(f)$, is defined as

$$\epsilon_t(f) = \sum_{i=1}^{n} w_{t-1}(i) I(Y_i \neq f(X_i)). \tag{17}$$

(b) Update the discrimination function F_{t-1}, and weights $\{w_{t-1}(i)\}_{i=1}^n$ as follows.

$$\begin{aligned} F_{t+1}(x) &= F_t(x) + \alpha_t f_{j(t)}(x) \\ w_t(i) &= \frac{1}{Z_t}\phi'(-Y_i F_t(X_i)) \end{aligned}$$

where Z_t is a normalization constant such that $\sum_{i=1}^n w_t(i) = 1$.

3. Finally, we obtain a resultant classifier $g(x) = \mathrm{sign}(F_T(x))$.

With $\phi = \exp$, we obtain AdaBoost algorithm that is the boosting algorithm proposed by Freund and Schapire (1997). In AdaBoost algorithm, the minimization in Eq. (11) is exactly independent of choice of α'. In addition, the optimization in Eq. (16) has an explicit solution:

$$\alpha_t = \frac{1}{2}\ln\frac{1-\epsilon_t(f_t)}{\epsilon_t(f_t)}.$$

Therefore, AdaBoost algorithm is rather simpler than others as seen below.

---

1. Initialize weights on sample $\{w_0(i)\}_{i=1}^n$ as $1/n$ for each i and $F_0 \equiv 0$.

2. For $t = 1, 2, \cdots, T$, repeat the following process.

(a) Find a new best classifier and its coefficient, denoted as $f_{j(t)}$ and α_t, as follows:

$$\begin{aligned} j(t) &= \operatorname*{argmin}_{j\in\{1,2,\cdots,J\}} \epsilon_t(f_j) \\ \alpha_t &= \frac{1}{2}\ln\frac{1-\epsilon_t(f_{j(t)})}{\epsilon_t(f_{j(t)})} \end{aligned}$$

$\epsilon_t(f)$, is defined as

$$\epsilon_t(f) = \sum_{i=1}^n w_{t-1}(i) I(Y_i \neq f(X_i)).$$

(b) Update $\{w_{t-1}(i)\}_{i=1}^n$ and the discrimination function, F_{t-1}, as follows.

$$\begin{aligned} w_t(i) &= \frac{1}{Z_t} w_{t-1}(i)\exp(-Y_i\alpha_t f_{j(t)}(X_i)\} \\ F_t(x) &= F_{t-1}(x) + \alpha_t f_{j(t)}(x) \end{aligned} \tag{18}$$

where Z_t is a normalization constant such that $\sum_{i=1}^n w_t(i) = 1$.

3. Finally, we obtain a resultant classifier $g(x) = \mathrm{sign}(F_T(x))$.

Different loss functions yield different boosting algorithm. We list loss functions of existing boosting methods in Table 1. Properties of these methods are discussed in Section 2.5.

Table 1: List of loss function ϕ, its derivative ϕ', and the inverse of the derivative $\xi = (\phi')^{-1}$. Note that η in η-boost does not denote $P(Y=1 \mid x)$ but denote just a real value in the interval $[0, 1]$.

Method	ϕ	ϕ'	ξ
AdaBoost	$\exp(z)$	$\exp(z)$	$\ln(z)$
LogitBoost	$\ln\{1+\exp(2z)\}$	$\frac{2\exp(2z)}{1+\exp(2z)}$	$\frac{1}{2}\ln\frac{z}{2-z}$
η-Boost	$(1-\eta)\exp(z)+\eta z$	$(1-\eta)\exp(z)+\eta$	$\ln\frac{z-\eta}{1-\eta}$
β-Boost	$\frac{1}{\beta+1}(1+\beta z)^{(\beta+1)/\beta}$	$(\beta z+1)^{(1/\beta)}$	$\frac{z^\beta-1}{\beta}$
MadaBoost	$\begin{cases} \frac{1}{2}\exp(2z) & (z<0) \\ z+(1/2) & (\text{otherwise}) \end{cases}$	$\begin{cases} \exp(2z) & (z<0) \\ 1 & \text{otherwise} \end{cases}$	$(1/2)\ln(z)\ z \in (0,1]$

Extension to multiclass problem We extend the boosting method for binary classification to a multiclass case. In statistical classification, it is natural to aim at modeling the posterior probability $P(Y=y \mid x)$. Therefore, we model the underlying probability $P(Y=y \mid x)$ by discriminant function $F(x, y)$. Then, similarly to the Bayes classifier g^*, we predict a label of x as

$$\hat{y} = \operatorname*{argmax}_{y' \in \mathcal{Y}} F(x, y').$$

There is no necessity to normalize $F(x, y)$ by $\sum_{y\in\mathcal{Y}} F(x, y) = 1$ because of the virtue of boosting method. Similarly to binary case, boosting method for multiclass case constructs $F(x, y)$ by combining base classifiers linearly. For each classifier $f_j \in \mathcal{C}$, define $f_j(x, y) = I(y \in f_j(x))$. A loss function for multiclass classification is defined as:

$$\begin{aligned} A(F) &= E[\sum_{y\in\mathcal{Y}} \phi(F(X,y) - E[F(X,Y) \mid X])] \\ A_D(F) &= \frac{1}{n}\sum_{i=1}^{n}\sum_{y\in\mathcal{Y}} \phi(F(X_i,y) - F(X_i,Y_i)). \end{aligned}$$

It should be noted that A and A_D denote either binary loss functions or multiclass loss functions depending on their context. Denote a Dirac's delta function by δ. Then, em-

pirical versions of $P(x)$ and $P(Y=y\,|\,x)$ are defined as

$$\hat{P}(x) = \frac{1}{n}\sum_{i=1}^{n}\delta(x, X_i)$$

and

$$\hat{P}(Y = y\,|\,x) = \sum_{i=1}^{n}\delta(x, X_i)\delta(y, Y_i)/\sum_{i=1}^{n}\delta(x, X_i).$$

Note that replacing $P(x)$ and $P(Y=y\,|\,x)$ in $A(F)$ with their empirical version alone does not yield A_D. One more assumption is needed. That is, for any $x \in \mathcal{X}$, there is at most one $y \in \mathcal{Y}$ for which $\hat{P}(Y = y\,|\,x) > 0$. With this assumption, A_D is derived as follows.

$$\begin{aligned} A(F)|_{P=\hat{P}} &= \int_{\mathcal{X}}\sum_{y\in\mathcal{Y}}\hat{P}(x)\phi(F(x,y) - \sum_{y'\in\mathcal{Y}}F(x,y')\hat{P}(Y = y'\,|\,x))dx \\ &= \frac{1}{n}\sum_{i=1}^{n}\sum_{y\in\mathcal{Y}}\phi(F(X_i,y) - \sum_{y'\in\mathcal{Y}}F(X_i,y')\hat{P}(Y = y'\,|\,X_i)) \\ &= \frac{1}{n}\sum_{i=1}^{n}\sum_{y\in\mathcal{Y}}\phi(F(X_i,y) - F(X_i,Y_i)). \end{aligned}$$

In real world, it may not be expected that the employed assumption is satisfied. However, the resultant algorithm works well in several situations. The derivation of the algorithm follows from the iterative minimization of empirical loss function in a similar way with Eq. (11) and (12). Therefore, we omit its derivation and show the resultant algorithm below.

---

1. Initialize weights on each sample $\{w_0(i,y)\}_{i=1}^{n}$ as $\frac{I(y\neq Y_i)}{n(G-1)}$ for each i and $F_0 \equiv 0$.

2. For $t = 1, 2, \cdots, T$, repeat the following process.

   (a) Find a new best classifier and its coefficient, denoted as $f_{j(t)}$ and α_t, as follows:

$$\begin{aligned} j(t) &= \operatorname*{argmin}_{j\in\{1,2,\cdots,J\}}\epsilon_t(f_j) \\ \alpha_t &= \operatorname{argmin}_{\alpha\in R_+}A_D(F_{t-1} + \alpha f_{j(t)}) \end{aligned}$$

   where the weighted error rate, $\epsilon_t(f)$, is defined as

$$\epsilon_t(f) = \frac{1}{2}\sum_{i=1}^{n}\sum_{y\in\mathcal{Y}}w_t(i,y)\{f(X_i,y) - f(X_i,Y_i) + 1\}I(y \neq Y_i) \qquad (19)$$

(b) Update $\{w_{t-1}(i,y)\}_{i=1}^n$ and the discrimination function, F_{t-1}, as follows.

$$\begin{aligned} F_t(x,y) &= F_{t-1}(x,y) + \alpha_t f_{j(t)}(x,y) \\ w_t(i,y) &= \frac{1}{Z_t}\phi'(F_t(X_i,y) - F_t(X_i,Y_i)) \end{aligned}$$

where Z_t is a normalization constant such that $\sum_{i=1}^n \sum_{y\in\mathcal{Y}} w_t(i,y)I(y \neq Y_i) = 1$.

3. Finally, we obtain a classifier $F(x) = \mathrm{argmax}_{y\in\mathcal{Y}} F_T(x,y)$.

Note that the above algorithm reduces to the binary boosting algorithm if we take $f(x,y) = (yf(x)+1)/2$. Thus, the binary boosting algorithm is a special case of the above algorithm.

When $\phi = \exp$, this algorithm reduces to AdaBoost.M2 that was proposed by Freund and Schapire (1997). Similarly to the binary case, computational cost of AdaBoost.M2 is relatively low due to its simplicity.

2.4.3 Regularized Boosting

A regularized version of boosting method is introduced in this section. In general, unregularized minimization of empirical probability of misclassification is apt to suffer from overfitting. The minimizer of empirical probability of misclassification is the empirical Bayes classifier $\hat{g}^*(x)$ defined as

$$\hat{g}^*(x) = 2I(\hat{P}(Y = y \,|\, x) \geq \frac{1}{2}) - 1.$$

Note that $\hat{g}^*(x)$ is not unique since prediction of $\hat{g}^*(x)$ at any x not in the training data has no effect on empirical probability of misclassification. Considering that there usually exist at most one datum for a fixed x, $\hat{g}^*(x)$ predicts a label of x based on only the training data. We say that a datum (X_i, Y_i) is *mislabelled* if its label Y_i differs from the label that the Bayes classifier, Eq. (3), predicts, *i.e.*, $Y_i \neq g^*(X_i)$. Since the training data contain mislabelled data in general, $\hat{g}^*(x)$ may perform poorly.

The ordinary boosting methods may also suffer from overfitting. The ordinary boosting methods does not put any restriction on the sum of coefficients of combined classifiers. This causes that the resultant classifier would inevitably overfit to the training data, even

though the overfitting sometimes does not occur until long iterations in boosting algorithm. This was also suggested by theoretical discussions. The upperbound of generalization error of the ordinary boosting that was obtained by Freund and Schapire (1997) indicates that the generalization error may increase as the iteration number of boosting goes to infinity. Note that an iterative gradient descent in boosting algorithm converges significantly slowly and then the overfitting will not occur until a large number of iterations in some practical cases. However, the overfitting occurs if the number of training data is quite few, or the training data contains many mislabelled data.

There are two ways for avoiding the overfitting. One is *early stopping* that was discussed by Jiang (2004) or Zhang and Yu (2005). The other way is to restrict sum of coefficients of combined classifiers, which was suggested by, *e.g.*, Mason et al. (1999). In this version, the sum of coefficients of combined classifier is always restricted to a some constant $\lambda > 0$. This implies that base classifiers are combined not linearly but convexly. We focus on this regularized boosting method in this section.

The regularized boosting algorithm is derived similarly to the ordinary boosting. Let $\mathcal{C}$ be a set of all available classifiers again. Define

$$\begin{aligned} A^\lambda(F) &= E\Big[\sum_{y\in\mathcal{Y}} \phi(-\lambda Y F(X))\Big] \\ A_D^\lambda(F) &= \frac{1}{n}\sum_{i=1}^n \sum_{y\in\mathcal{Y}} \phi(-\lambda Y_i F(X_i)). \end{aligned}$$

Note that $A(F)$ under the restriction that $\sum_{t=1}^T \alpha_t = \lambda$ in which $F(x) = \sum_{t=1}^T \alpha_t f_{j(t)}(x)$ is equivalent to $A^\lambda(F)$ under the restriction that $\sum_{t=1}^T \alpha_t = 1$. We denote the set of all convex combination of arbitrary classifiers in $\mathcal{C}$ by $\mathrm{conv}(\mathcal{C})$ in the sequel. Let $F_0(x) \equiv 0$ be an initial discriminant function. For a given current discriminant function, F_{t-1}, regularized boosting chooses a new base classifier, f, and its coefficient, α, iteratively as follows.

$$f = \operatorname*{argmin}_{f'\in\mathcal{C}} A_D^\lambda(F_{t-1} + \alpha'(f' - F_{t-1})) \quad \text{for any positive } \alpha' \tag{20}$$

$$\alpha = \operatorname*{argmin}_{\alpha'>0} A_D^\lambda(F_{t-1} + \alpha'(f - F_{t-1})) \tag{21}$$

Then, the discriminant function is updated as

$$F_t(x) = (1-\alpha)F_{t-1}(x) + \alpha f(x). \tag{22}$$

The final classifier is obtained as $g(x) = \mathrm{sign}(F_T(x))$ after T repetitions of this process. The difference from the ordinary boosting is only the way of combining base classifiers. Thus, almost same arguments hold for regularized boosting. We describe the complete summary of regularized boosting below.

---

1. Determine a smoothing parameter $\lambda > 0$.

2. Initialize weights on sample $\{w_0(i)\}_{i=1}^n$ as $1/n$ for each i and $F_0 \equiv 0$.

3. For $t = 1, 2, \cdots, T$, repeat the following process.

   (a) Find a new best classifier and its coefficient, denoted as $f_{j(t)}$ and α_t, as follows:

$$\begin{aligned} j(t) &= \operatorname*{argmin}_{j \in \{1,2,\cdots,J\}} \epsilon_t(f_j) \\ \alpha_t &= \operatorname*{argmin}_{\alpha \in R_+} A_D^{\lambda}(F_{t-1} + \alpha(f_{j(t)} - F_{t-1})\} \quad (t \geq 2) \end{aligned} \tag{23}$$

   with $\alpha_1 = 1$ where the weighted error rate, $\epsilon_t(f)$, is defined as

$$\epsilon_t(f) = \sum_{i=1}^{n} w_t(i) I(Y_i \neq f(X_i)).$$

   (b) Update $\{w_{t-1}(i)\}_{i=1}^n$ and the discrimination function, F_{t-1}, as follows.

$$F_t(x) = (1 - \alpha_t) F_{t-1}(x) + \alpha_t f_{j(t)}(x) \tag{24}$$

$$w_t(i) = \frac{1}{Z_t} \phi'(-\lambda Y_i F_t(X_i)) \tag{25}$$

   where Z_t is a normalization constant such that $\sum_{i=1}^n w_t(i) = 1$.

4. Finally, we obtain a resultant classifier $g(x) = \mathrm{sign}(F_T(x))$.

It holds that $\sum_{t=1}^T \alpha_t = 1$. This implies that $F_T(x)$ necessarily takes its value in the closed interval $[-1, 1]$.

We also describe the summary of regularized boosting for multiclass classification for the completeness.

---

1. Determine a smoothing parameter, $\lambda > 0$.

2. Initialize weights on each sample $\{w_0(i,y)\}_{i=1}^n$ as $\frac{I(y\neq Y_i)}{n(|\mathcal{Y}-1|)}$ for each i and $F_0 \equiv 0$.

3. For $t = 1, 2, \cdots, T$, repeat the following process.

   (a) Find a new best classifier and its coefficient, denoted as $f_{j(t)}$ and α_t, as follows:

$$\begin{aligned} j(t) &= \underset{j\in\{1,2,\cdots,J\}}{\operatorname{argmin}} \epsilon_t(f_j) \\ \alpha_t &= \begin{cases} \operatorname{argmin}_{\alpha\in R_+} A_D^{\lambda}(F_{t-1} + \alpha(f_{j(t)} - F_{t-1})\} & (t > 1) \\ 1 & (t = 1) \end{cases} \end{aligned}$$

   where the weighted error rate, $\epsilon_t(f)$, is defined as

$$\epsilon_t(f) = (1/2)\sum_{i=1}^{n}\sum_{y\in\mathcal{Y}} w_t(i,y)\{f(X_i,y) - f(X_i,Y_i) + 1\}I(y \neq Y_i)$$

   (b) Update $\{w_{t-1}(i,y)\}_{i=1}^n$ and the discrimination function, F_{t-1}, as follows.

$$\begin{aligned} F_t(x,y) &= (1-\alpha_t)F_{t-1}(x,y) + \alpha_t f_{j(t)}(x,y) \\ w_t(i,y) &= \frac{1}{Z_t}\phi'(\lambda(F_t(X_i,y) - F_t(X_i,Y_i))) \end{aligned}$$

   where Z_t is a normalization constant such that $\sum_{i=1}^n \sum_{y\in\mathcal{Y}} w_t(i,y)I(y \neq Y_i) = 1$.

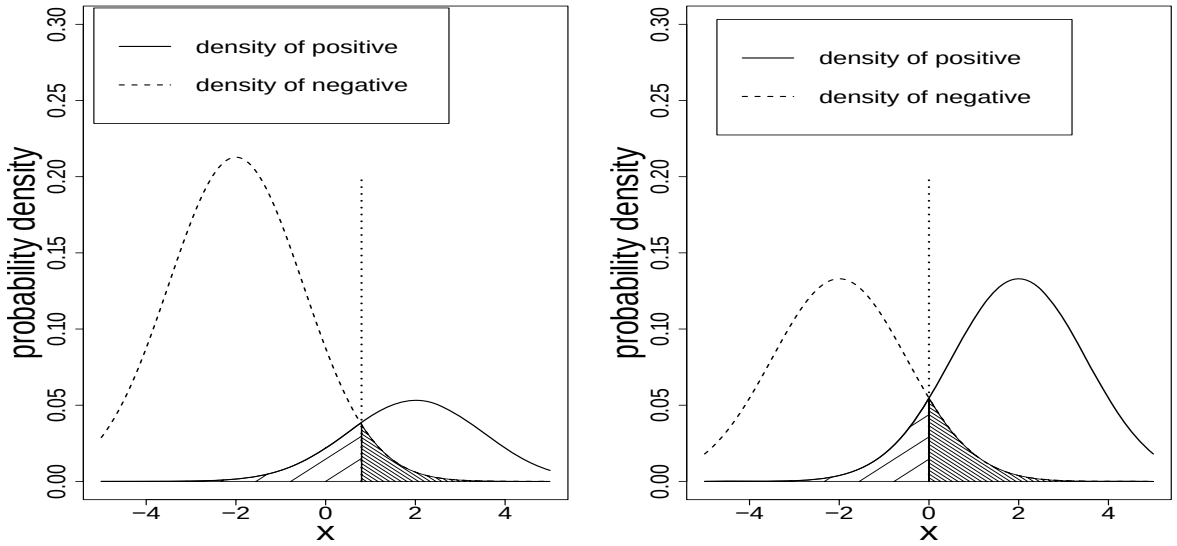
4. Finally, we obtain a classifier $F(x) = \operatorname{argmax}_{y\in\mathcal{Y}} F_T(x,y)$.

Lugosi and Vayatis (2004) proved the Bayes risk consistency of regularized boosting method under appropriate choice of λ. They suggested λ should be chosen as a function of the sample size of the training data. Instead of description of their result, we note that Section 4 provides a complete information about this issue since our proposed method that appear in Section 4 contains the ordinary regularized boosting as a special case. See, however, (Mason et al., 1999; Lugosi and Vayatis, 2004) for the role of λ in details. We show a simple example to illustrate the effectiveness of this type of regularized boosting in Section 2.5.5.

2.4.4 AsymBoost

AsymBoost (Viola and Jones, 2001) is a variant of AdaBoost, developed specifically for binary classification where the distributions of positive and negative samples are highly

skewed. To illustrate this issue, consider an example of binary classification with one feature (Figure 1), with a simple discriminant function $F(x) = \text{sign}(x - b)$ where $x = b$ corresponds to the decision boundary. In the asymmetric case (a), the decision boundary minimizing prediction error is shifted to the right compared to the symmetric case (b) since negative samples are much more likely to occur than positive samples. As a result, the false negative ratio for the asymmetric case (a) is much larger than that of the symmetric case (b), where the false positive ratio (FPR) and the false negative ratio (FNR) are defined as:



(a) (b)

Figure 1: The probability density functions $P(x\,, Y = 1)$ (solid line) and $P(x\,, Y = -1)$ (dotted line) in a binary classification problem are shown. Panel (a) is the asymmetric case ($P(Y = 1) \ll P(Y = -1)$) and panel (b) is the symmetric case ($P(Y = 1) = P(Y = -1)$) in the same setting. Given a simple discriminant function $F(x) = \text{sign}(x-b)$, where $x = b$ corresponds to its decision boundary, the dotted vertical line shows the Bayes optimal decision boundary which minimizes the probability of misclassification. The false negative ratio corresponds to the proportion of the light solid area to the under area of $P(x\,, Y = 1)$, while the false positive ratio corresponds to the proportion of the the dense solid area to the under area of $P(x\,, Y = -1)$. The probability of misclassification of F corresponds to the sum of the light solid are and the dense solid area.

$$\begin{aligned} \text{FPR} &= P\left(F(x) \geq 0 \,|\, Y = -1\right), \\ \text{FNR} &= P\left(F(x) < 0 \,|\, Y = 1\right). \end{aligned} \tag{26}$$

FPR denotes the probability of false prediction given a negative sample, while FNR denotes the probability of false prediction given a positive sample. AsymBoost enables us to attain an arbitrary balance between the FPR and the FNR. This is achieved by modifying the symmetric loss function used by the ordinary boosting. The AsymBoost algorithm seeks to minimize the following asymmetric empirical loss function:

$$A_D^{\text{asym}}(F) = \frac{1}{n} \sum_{i=1}^{n} (\sqrt{k})^{Y_i} \exp(-Y_i F(X_i)) \tag{27}$$

where $k > 0$ is a balance parameter. It is easily extended to boosting with general loss function even though the original AsymBoost proposed by Viola and Jones (2001) is an asymmetric version of AdaBoost. As easily seen in (27), the loss for positive samples are weighted k times more than that for negative samples. It should be mentioned that AsymBoost cannot be regarded as one of the ordinary boosting method since the asymmetric loss function in Eq. (27) does not satisfy Condition 18. AsymBoost minimizes this asymmetric loss iteratively in a manner similar to AdaBoost:

$$\begin{aligned} f &= \operatorname*{argmin}_{f' \in \mathcal{C}} A_D^{\text{asym}}(F_{t-1} + \alpha' f') \quad \text{for any positive} \alpha' \\ \alpha &= \operatorname*{argmin}_{\alpha' > 0} A_D^{\text{asym}}(F_{t-1} + \alpha' f). \end{aligned}$$

Define

$$w_t(i) = \frac{1}{Z_t} \exp(-Y_i F_{t-1}(X_i) + Y_i \ln \sqrt{k}),$$

where

$$Z_t = \sum_{i'=1}^{n} \exp(-Y_{i'} F_{t-1}(X_{i'}) + Y_{i'} \ln \sqrt{k}).$$

The loss function $A_D^{\mathrm{asym}}(F_{t-1}+\alpha' f')$ can be written as

$$\begin{aligned} A_D^{\mathrm{asym}}(F_{T-1}+\alpha' f') &= \frac{1}{n}\sum_{i=1}^{n}(\sqrt{k})^{Y_i}\exp(-Y_i(F_{t-1}(X_i)+\alpha' f'(X_i))) \\ &= \frac{1}{n}\sum_{i=1}^{n}\exp(-Y_i F_{t-1}(X_i)+Y_i\ln\sqrt{k})\exp(-Y_i\alpha' f'(X_i)) \\ &= \frac{1}{n}\sum_{i=1}^{n}Z_t w_t(i)\exp(-Y_i\alpha' f'(X_i)) \\ &= \frac{1}{n}\sum_{i=1}^{n}Z_t w_t(i)\{I(Y_i\neq f'(X_i))e^{\alpha'}+I(Y_i=f'(X_i))e^{-\alpha'}\} \\ &= \frac{Z_t}{n}\sum_{i=1}^{n}w_t(i)\{I(Y_i\neq f'(X_i))(e^{\alpha'}-e^{-\alpha'})+e^{-\alpha'}\}. \end{aligned}$$

Therefore, the search of $f=\mathrm{argmin}_{f'\in\mathcal{C}}=A_D^{\mathrm{asym}}(F_{T-1}+\alpha' f')$ reduces to

$$f=\underset{f'\in\mathcal{C}}{\mathrm{argmin}}\sum_{i=1}^{n}w_t(i)I(Y_i\neq f'(X_i)),$$

regardless of the value of positive α'. Thus, it is natural to define the weighted error rate $\epsilon_t(f)$ as

$$\epsilon_t(f)=\sum_{i=1}^{n}w_t(i)I(Y_i\neq f(X_i)).$$

In addition, it is easy to see that $\alpha=\mathrm{argmin}_{\alpha'\in R_+}=A_D^{\mathrm{asym}}(F_{T-1}+\alpha' f')$ for any fixed $f'\in\mathcal{C}$ is calculated as

$$\frac{1}{2}\ln\frac{1-\epsilon_t(f')}{\epsilon_t(f')}.$$

We show the complete summary of AsymBoost algorithm below.

---

1. Initialize weights on sample $\{w_0(i)\}_{i=1}^n$ as $1/n$ for each i and $F_0\equiv 0$.

2. For $t=1,\,2,\,\cdots,\,T$, repeat the following process.

   (a) Find a new best classifier and its coefficient, denoted as $f_{j(t)}$ and α_t, as follows:

$$\begin{aligned} j(t) &= \underset{j\in\{1,2,\cdots,J\}}{\mathrm{argmin}}\ \epsilon_t(f_j) \\ \alpha_t &= \frac{1}{2}\ln\frac{1-\epsilon_t(f_{j(t)})}{\epsilon_t(f_{j(t)})}, \end{aligned}$$

where $\epsilon_t(f)$ is defined as

$$\epsilon_t(f) = \sum_{i=1}^{n} w_{t-1}(i) I(Y_i \neq f(X_i)).$$

(b) Update $\{w_{t-1}(i)\}_{i=1}^n$ and the discrimination function, F_{t-1}, as follows.

$$\begin{aligned} w_t(i) &= \frac{1}{Z_t} w_{t-1}(i) \exp\left(-Y_i \alpha_t f_{j(t)}(X_i) + \frac{Y_i}{T} \ln \sqrt{k}\right) \\ F_t(x) &= F_{t-1}(x) + \alpha_t f_{j(t)}(x) \end{aligned} \tag{28}$$

where Z_t is a normalization constant such that $\sum_{i=1}^n w_t(i) = 1$.

3. Finally, we obtain a resultant classifier $g(x) = \mathrm{sign}(F_T(x))$.

Thus, the AsymBoost algorithm is quite similar to that of AdaBoost. The difference is only the weight update rule Eq. (28). Note that the reason why $Y_i \ln \sqrt{k}$ is divided by the iteration number T is to assign the influence of the asymmetric term $\sqrt{k}^{Y_i}$ to all base classifiers chosen in the learning process equally.

The value of k determines the balance between the FPR and the FNR. If $k > 1$, false prediction of positive samples are penalized stronger and vice versa. Intuitively, the larger k, the larger the FPR and the smaller the FNR. To understand this point, we follow a similar argument of Friedman et al. (2000). The population minimizer of the asymmetric loss Eq. (27), denoted by F^*_{asym}, is calculated as:

Proposition 21. *Let*

$$A^{\mathrm{asym}}(F) = E[\sqrt{k}^{Y} \exp(-YF(X))].$$

Then, the population minimizer of $A(\cdot)$ is obtained as

$$F^*_{\mathrm{asym}}(x) = \operatorname*{argmin}_{F \in \mathcal{F}} A^{\mathrm{asym}}(F) = \frac{1}{2} \ln \frac{P(Y=1 \mid x)}{P(Y=-1 \mid x)} + \ln \sqrt{k}.$$

Proof. Due to the decomposition of expectation, we have

$$\begin{aligned} A^{\mathrm{asym}}(F) &= E[\sqrt{k}^{Y} \exp(-YF(X))] \\ &= EE[\sqrt{k}^{Y} \exp(-YF(x)) | x] \\ &= E[P(Y=1 \mid x)\sqrt{k} \exp(-F(x)) + P(Y=-1 \mid x) \frac{1}{\sqrt{k}} \exp(F(x))]. \end{aligned}$$

Write $F(x)$ as c and the inside of $E[\cdot]$ as $\hat{A}(c)$. The constant, c, that minimizes $\hat{A}(c)$ satisfies

$$\frac{\hat{A}(c)}{dc} = -P(Y=1 \mid x)\sqrt{k}e^{-c} + P(Y=-1 \mid x)\frac{1}{\sqrt{k}}e^{c} = 0.$$

Thus, we have

$$c = F(x) = \frac{1}{2}\ln k\frac{P(Y=1 \mid x)}{P(Y=-1 \mid x)}.$$

$F^*_{\text{asym}}(x)$ satisfying this relationship for any x minimizes $A^{\text{asym}}(F)$. □

This implies that $F^*_{\text{asym}} = F^*_{\text{ada}} + \ln\sqrt{k}$ where F^*_{ada} denotes the target discriminant function of AdaBoost (See Section 2.5.2). Therefore, $\text{sign}(F^*_{\text{asym}}(x))$ is not equal to the Bayes classifier in general. Note that the discriminant function estimated by AsymBoost differs from that estimated by AdaBoost by the intercept $\ln\sqrt{k}$ in general. They coincide with each other if the base classifier class $\mathcal{C}$ contains a constant classifier $f(x) \equiv 1$. In such a case, AdaBoost or AsymBoost algorithm may directly estimate the intercept term and their estimates are same except the intercept. The value of discriminant function $F^*_{\text{asym}}(x) \geq 0$ increases as k becomes large. This results in an increase in the FPR and a decrease in the FNR. As a specific example, suppose we estimate $k = P(Y=-1)/P(Y=1)$ as:

$$k = \frac{n_n}{n_p} \tag{29}$$

where n_p is the number of positive samples and n_n is the number of negative samples in a training data set D. In this way, we can obtain the FPR and the FNR that we would have obtained had our sample contained equal numbers of positive and negative values ($n_p = n_n$). With k estimated as Eq. (29), we have

$$F^*_{\text{asym}}(x) = \frac{1}{2}\ln\frac{P(x \mid Y=1)}{P(x \mid Y=-1)}.$$

In the case that the form of the densities of positive and negative samples are as illustrated in Figure 1 (identical shapes and point-symmetric[1]), the FPR and the FNR given by AsymBoost will be equal.

[1]A function $f(x)$ is point-symmetric if the value of $f(x)$ depends only on the distance from a certain point $\boldsymbol{m}$, *i.e.*, f can be rewritten as $f(x) = g(\|x - \boldsymbol{m}\|)$ for some function g. If f is unimodal, m is the mode.

2.5 Statistical properties of boosting

Several literatures have tried to explain why the boosting method is successful. Freund and Schapire (1997) showed that the upperbound of training error given by AdaBoost and AdaBoost.M2 decreases exponentially. However, it is commonly known that the minimization of training error does not necessarily lead to the minimization of generalization error. To elucidate why boosting methods perform well, it is needed to evaluate the generalization error of boosting. There are two approaches for evaluating generalization performance of boosting method. One is the bias-variance theory approach. The other is the evaluation of the upperbound of generalization error. Breiman (1998) took the former approach by regarding the boosting method as ARCING algorithm. As a result, it was elucidated that boosting methods reduced the variance of decision tree algorithm more than bagging. However, Schapire et al. (1998) pointed out that variance reduction was not sufficient as an explanation of the success of boosting. Several researchers have taken the latter approach (Freund and Schapire, 1997; Schapire et al., 1998; Koltchinskii and Panchenko, 2002; Bartlett and Mendelson, 2002; Lugosi and Vayatis, 2004) although this approach does not explain the success of boosting perfectly. Still now, intensive theoretical research on boosting is required. In addition, we discuss the Bayes rule equivalence, the least favorable error property and the geometrical interpretation of boosting method in this section.

2.5.1 Least favorable error

The ordinary boosting has an interesting property of the weighted error rate. The weighted error $\epsilon_t(f)$ defined in Eq. (17) and (19) has an important role in the ordinary boosting algorithm.

Proposition 22 (Least favorable error property). *At each iteration step t in the ordinary boosting algorithm, it holds that*

$$\epsilon_t(f_{j(t-1)}) = 1/2.$$

Proof. It suffices to prove this statement for multiclass case since the ordinary boosting for binary case is obtained as a special case of multiclass algorithm. Since α_t minimizes $A_D(F_{t-1} + \alpha f_t)$, α_t must satisfy

$$\frac{\partial}{\partial \alpha} A_D(F_{t-1} + \alpha f_t) = 0.$$

This implies that

$$\sum_{y \in \mathcal{Y}} \phi'(F_t(X_i, y) - F_t(X_i, Y_i))(f_t(X_i, y) - f_t(X_i, Y_i)) = 0.$$

It is easy to see that the left-hand side of this equation equals to $2\epsilon_t(f_{j(t)}) - 1$. □

The weighted error rate in binary case can be interpreted as follows. Recall that the sum of all $\{w_t(i)\}_{i=1}^n$ is always restricted to one. Thus, we can regard $w_t(i)$ as the probability assigned to the i-th sample (X_i, Y_i). Suppose that a pair of new random variables (X_t', Y_t') is resampled from D according to the probabilities $\{w_t(i)\}_{i=1}^n$. Inspection of Eq. (17) indicates that each weight on sample is proportional to $\phi'\{-Y_i F_{t-1}(X_i)\}$ in the search of $f_{j(t)}$. Thus, (X_t', Y_t') takes its values in either i-th sample (X_i, Y_i) on D with a higher probability if a larger $F_{t-1}(X_i)$ predicts a label of X_i incorrectly, *i.e.*, the larger that $-Y_i F(X_i)$ is. Therefore the weighted error rate $\epsilon_t(f)$ can be regarded as the probability of false prediction over (X_t', Y_t') given by f. Similar interpretation is given for multiclass case by Murata et al. (2004). The least favorable error property indicates that the chosen classifier $f_{j(t)}$ is the best classifier on the resampled data sets that is least favorable for $f_{j(t-1)}$ in the sense that the error rate of $f_{j(t-1)}$ is worst among all base classifiers. Intuitively, $f_{j(t)}$ is the base classifier that improves the prediction performance of $f_{j(t-1)}$ most.

2.5.2 Bayes rule equivalence

The population minimizer of loss function with cost function ϕ satisfying Condition 18 necessarily yields the Bayes classifier. Friedman et al. (2000) showed that the population minimizer of exponential loss function ($\phi = \exp$) over $\mathcal{F}$ in binary classification is the half log-odds. Clearly the sign of the half log-odds is the Bayes classifier. Similar calculations indicate that the population minimizer of general cost function satisfying Condition ϕ necessarily yields the Bayes classifier.

Binary case

Proposition 23. *Let $\phi : R \to R$ be a cost function satisfying Condition 18. Define $F^*(x) = \mathrm{argmin}_{F \in \mathcal{F}} A(F)$.*

Case (a) *Assume that $\eta(x) \in (0, 1)$ holds almost everywhere. Then $g_F(x) = \mathrm{sign}(F^*(x))$ is the Bayes classifier.*

Case (b) *Assume that $\eta(x) \in \{0,1\}$ is almost everywhere. Then,*

$$\inf_{F \in \mathcal{F}} A(F) = 0.$$

Proof. **Case (a)** Assume that $\eta(x) \notin \{0,1\}$ with probability one. Due to the decomposition of expectation, we have

$$A(F) = E[\phi(-YF(X))] = E[\eta(X)\phi(-F(X)) + (1-\eta(X))\phi(F(X))].$$

F^* should minimize the inside of the last expectation for each $x \in \mathcal{X}$. Define $h(\eta, \alpha) = \eta\phi(-\alpha) + (1-\eta)\phi(\alpha)$. Thus, for a fixed x, $F^*(x)$ satisfies

$$F^*(x) = \operatorname*{argmin}_{\alpha \in R} h(\eta(x), \alpha).$$

Since $\eta(x) \notin \{0,1\}$ and ϕ is strictly increasing and convex, $h(\eta, \alpha)$ has the unique minimum in α. Since ϕ is differentiable, $F^*(x)$ must satisfy

$$-\eta(x)\phi(-F^*(x)) + (1-\eta(x))\phi(F^*(x)) = 0.$$

Therefore, we have

$$\frac{\phi(F^*(x))}{\phi(-F^*(x))} = \frac{\eta(x)}{1-\eta(x)}.$$

This implies that $F^*(x) > 0$ if and only if $\eta(x) > 1/2$, proving that $\operatorname{sign}(F^*(x))$ is the Bayes classifier.

Case (b) Assume that $\eta(x) \in \{0,1\}$ with probability one. Clearly, there is no minimizer of $h(\eta, \alpha)$ in α. Define f_n taking values n if $\eta(x) = 1$ and $-n$ if $\eta(x) = 0$. Taking the limit in n combined with the fact $A(F) \geq 0$ leads to $\inf_{f \in \mathcal{F}} A(F) = 0$.

□

Multiclass case Similarly to the binary case, the population minimizer of loss functions in multiclass classification also yields the Bayes classifier.

Proposition 24. *Let $\phi : R \to R$ be a cost function satisfying Condition 18. Define $F^*(x) = \operatorname{argmin}_{F \in \mathcal{F}} A(F)$.*

Case (a) *Assume that $P(Y = y \,|\, x) \neq 1$ for any y almost everywhere. Then $g_F(x) = \operatorname{argmax}_{y \in \mathcal{Y}}(F^*(x, y))$ is the Bayes classifier.*

Case (b) *Assume that there exists a single* $\bar{y} \in \mathcal{Y}$ *such that* $P(Y = \bar{y} \,|\, x) = 1$ *almost everywhere. Then,*

$$\inf_{F \in \mathcal{F}} A(F) = 1.$$

Proof. **Case (a)** Assume that $P(Y = y \,|\, x)\{0, 1\}$ with probability one. The loss function for multiclass classification is written as

$$A(F) = E\Big[\sum_{y \in \mathcal{Y}} \phi(F(X, y) - E[F(X, Y) \,|\, X])\Big].$$

Write the inside of the expectation as $A|_{X=x}(F)$. For any fixed x and y', the minimizer of $A(F)$, denoted by F^*, should satisfy

$$\begin{aligned} \frac{\partial}{\partial F(x, y')} A|_{X=x}(F^*) &= (1 - P(Y = y' \,|\, x))\phi'(F^*(x, y') - E[F^*(x, Y) \,|\, x]) + \\ &\quad \sum_{y \neq y'} P(Y = y' \,|\, x)\phi'(F^*(x, y) - E[F^*(x, Y) \,|\, x]) \\ &= 0. \end{aligned}$$

Therefore, the posterior $P(Y = y \,|\, x)$ can be written as

$$P(Y = y \,|\, x) = \frac{\phi'(F^*(x, y) - E[F^*(x, Y) \,|\, x])}{\sum_{y' \in \mathcal{Y}} \phi'(F^*(x, y') - E[F^*(x, Y)])} \tag{30}$$

for any fixed x and y. This implies that there must exist a positive function $c(x)$ such that $\phi'(F^*(x, y) - E[F^*(x, Y) \,|\, x]) = c(x)P(Y = y \,|\, x)$. Due to the strict monotonicity of ϕ', we have

$$\begin{aligned} \operatorname*{argmax}_{y \in \mathcal{Y}} F^*(x, y) &= \operatorname*{argmax}_{y \in \mathcal{Y}} F^*(x, y) - E[F^*(x, Y) \,|\, x] \\ &= \operatorname*{argmax}_{y \in \mathcal{Y}} \phi'(F^*(x, y) - E[F^*(x, Y) \,|\, x]) \\ &= \operatorname*{argmax}_{y \in \mathcal{Y}} c(x)P(Y = y \,|\, x) \\ &= \operatorname*{argmax}_{y \in \mathcal{Y}} P(Y = y \,|\, x). \end{aligned}$$

Case (b) Assume that there exists a single $\bar{y} \in \mathcal{Y}$ such that $P(Y = \bar{y} \,|\, x) = 1$ with probability one. For any $y \neq \bar{y}$, $P(Y = y \,|\, x) = 0$. Clearly, there is no F^* satisfying Eq. (30) since ϕ' is strictly positive. Since $E[F(x, Y) \,|\, x] = F(x, \bar{y})$, $A(F)$ has the

form:

$$\begin{aligned} A(F) &= E\Big[\sum_{y\in\mathcal{Y}} \phi(F(x,y) - E[F(x,Y)\,|\,x])\Big] \\ &= E\Big[\sum_{y\in\mathcal{Y}} \phi(F(x,y) - F(x,\bar{y}))\Big] \\ &= E\Big[\sum_{y\neq\bar{y}} \phi(F(x,y) - F(x,\bar{y}))\Big] + 1. \end{aligned}$$

Define $F_n(x,y)$ taking values n if $y = \bar{y}$ and taking values $-n$ otherwise for any x. Taking the limit in n combined with the fact $A(F) \geq 1$ leads to $\inf_{f\in\mathcal{F}} A(F) = 1$.

□

We derive the population minimizers of several loss functions in binary classification.

AdaBoost The population minimizer of the expected exponential loss function is equal to the half log-odds.

Proposition 25. *Let $\phi = \exp$. Then, the population minimizer of $A(\cdot)$ is obtained as*

$$F^*_{\text{ada}}(x) = \operatorname*{argmin}_{F\in\mathcal{F}} A(F) = \frac{1}{2}\ln\frac{P(Y=1\,|\,x)}{P(Y=-1\,|\,x)} \tag{31}$$

where $E[\cdot]$ denotes the expectation with respect to the true joint distribution of X and Y.

Proof. Due to the decomposition of expectation, we have

$$\begin{aligned} A(F) &= E\exp(-YF(X)) \\ &= EE[\exp(-YF(x))|x] \\ &= E[P(Y=1\,|\,x)\exp(-F(x)\} + P(Y=-1\,|\,x)\exp(F(x)\}]. \end{aligned}$$

Write $F(x)$ as c and the inside of $E[\cdot]$ as $\hat{A}(c)$. The constant, c, that minimizes $\hat{A}(c)$ satisfies

$$\frac{\hat{A}(c)}{dc} = -P(Y=1\,|\,x)\exp(-c) + P(Y=-1\,|\,x)\exp(c) = 0.$$

Thus, we have

$$c = F(x) = \frac{1}{2}\ln\frac{P(Y=1\,|\,x)}{P(Y=-1\,|\,x)}.$$

$F^*(x)$ satisfying this relationship for any x minimizes $A(F)$. □

We may rewrite the above equation as

$$P(Y=y\,|\,x)=\frac{-(y+1)F^*_{\text{ada}}(x)}{1+\exp(-2F^*_{\text{ada}}(x))}. \tag{32}$$

This indicates that the AdaBoost algorithm is interpreted as a forward fitting of this logistic model.

LogitBoost

Proposition 26. *Let $\phi(x)=\ln(1+\exp(-2x))$. Then, the population minimizer of $A(\cdot)$ is obtained as*

$$F^*_{\text{logit}}(x)=\operatorname*{argmin}_{F\in\mathcal{F}} A(F)=\frac{1}{2}\ln\frac{P(Y=1\,|\,x)}{P(Y=-1\,|\,x)}.$$

Proof. Due to the decomposition of expectation, we have

$$\begin{aligned}
A(F) &= E[\ln(1+\exp(-2YF(X)))] \\
&= EE[\ln(1+\exp(-YF(x)))\,|\,x] \\
&= E[P(Y=1\,|\,x)\ln(1+\exp(-2F(x)))+P(Y=-1\,|\,x)\ln(1+(\exp(2F(x))))\,|\,x].
\end{aligned}$$

Write $F(x)$ as c and the inside of $E[\cdot]$ as $\hat{A}(c)$. The constant, c, that minimizes $\hat{A}(c)$ satisfies

$$\frac{\hat{A}(c)}{dc}=P(Y=1\,|\,x)\frac{-2\exp(-2c)}{1+\exp(-2c)}+P(Y=-1\,|\,x)\frac{2\exp(2c)}{1+\exp(2c)}=0.$$

Thus, we have

$$\begin{aligned}
\frac{P(Y=1\,|\,x)}{P(Y=-1\,|\,x)} &= \frac{e^{4c}(1+e^{-2c})}{1+e^{2c}} \\
&= \frac{e^{4c}(1+e^{-2c})}{1+e^{2c}} \\
&= \frac{(e^{4c}+e^{2c})}{1+e^{2c}}.
\end{aligned}$$

Denoting the left-hand side by u, this equation is rewritten as

$$e^{4c}+(1-u)e^{2c}-u=0.$$

By solving this quadratic equation, we have

$$c=F(x)=\frac{1}{2}\ln\frac{P(Y=1\,|\,x)}{P(Y=-1\,|\,x)}.$$

$F^*(x)$ satisfying this relationship for any x minimizes $A(F)$. □

It is easy to see that the minimization of $\phi(x) = \ln(1 + \exp(2x))$ is equal to the maximization of the log-likelihood with exponential model defined in Eq. (32). In fact, LogitBoost is deeply connected to AdaBoost in view of extended KL-divergence (Lebanon and Lafferty, 2002).

β-Boost

Proposition 27. *Let*

$$\phi(x) = \frac{1}{\beta+1}(1+\beta x)^{\frac{\beta+1}{\beta}}.$$

Then, the population minimizer of $A(\cdot)$ is obtained as

$$F^*_\beta(x) = \operatorname*{argmin}_{F\in\mathcal{F}} A(F) = \frac{1}{\beta}\frac{\left(\frac{P(Y=1\,|\,x)}{P(Y=-1\,|\,x)}\right)^\beta - 1}{\left(\frac{P(Y=1\,|\,x)}{P(Y=-1\,|\,x)}\right)^\beta + 1}.$$

Proof. Due to the decomposition of expectation, we have

$$\begin{aligned} A(F) &= E\left[\frac{1}{\beta+1}(1-\beta YF(X))^{\frac{\beta+1}{\beta}}\right] \\ &= EE\left[\frac{1}{\beta+1}(1-\beta YF(x))^{\frac{\beta+1}{\beta}}\,|\,x\right] \\ &= E\left[\frac{1}{\beta+1}\{P(Y=1\,|\,x)(1-\beta F(X))^{\frac{\beta+1}{\beta}} + \right. \\ &\qquad \left. P(Y=-1\,|\,x)P(Y=1\,|\,x)(1+\beta F(X))^{\frac{\beta+1}{\beta}}\}\right]. \end{aligned}$$

Write $F(x)$ as c and the inside of $E[\cdot]$ as $\hat{A}(c)$. The constant, c, that minimizes $\hat{A}(c)$ satisfies

$$\frac{\hat{A}(c)}{dc} = \frac{1}{\beta}\{-P(Y=1\,|\,x)(1-\beta c)^{\frac{1}{\beta}} + P(Y=-1\,|\,x)(1+\beta c)^{\frac{1}{\beta}}\} = 0.$$

Thus, we have

$$\begin{aligned} \frac{P(Y=1\,|\,x)}{P(Y=-1\,|\,x)} &= \left(\frac{1+\beta c}{1-\beta c}\right)^{\frac{1}{\beta}} \\ \frac{1+\beta c}{1-\beta c} &= \left(\frac{P(Y=1\,|\,x)}{P(Y=-1\,|\,x)}\right)^\beta \\ c &= \frac{1}{\beta}\frac{\left(\frac{P(Y=1\,|\,x)}{P(Y=-1\,|\,x)}\right)^\beta - 1}{\left(\frac{P(Y=1\,|\,x)}{P(Y=-1\,|\,x)}\right)^\beta + 1}. \end{aligned}$$

The population minimizer $F^*_\beta(x) = c$ must satisfy this equation for all $x \in \mathcal{X}$. □

η-Boost η-loss has also unique population minimizer of the expected loss function, denoted by F_η^*. The minimizer F_η^*, however has a complicated form so that we may not interpret easily. Instead, we show its relationship to the underlying distribution.

Proposition 28. *Let $\phi(x) = (1-\eta)\exp(x) + \eta x$ where η is a real value in the interval $[0,1]$. Define $F_\eta^* = \operatorname{argmin}_{F\in\mathcal{F}} A(F)$. Then, the population minimizer F_η^* satisfies*

$$P(Y=y \mid x) = \frac{(1-\eta)\exp(yF_\eta^*(x))}{\sum_{y'\in\{-1,1\}}(1-\eta)\exp(y'F(x)) + \eta}.$$

Proof. Due to the decomposition of expectation, we have

$$\begin{aligned} A(F) &= E[(1-\eta)\exp(-YF(X)) + \eta(-YF(X))] \\ &= EE[(1-\eta)\exp(-YF(x)) + \eta(-YF(x)) \mid x] \\ &= E[P(Y=1 \mid x)\{(1-\eta)\exp(-F(X)) - \eta F(X)\} + \\ &\quad P(Y=-1 \mid x)\{(1-\eta)\exp(F(X)) + \eta F(X)\}]. \end{aligned}$$

Write $F(x)$ as c and the inside of $E[\cdot]$ as $\hat{A}(c)$. The constant, c, that minimizes $\hat{A}(c)$ satisfies

$$\frac{\hat{A}(c)}{dc} = P(Y=1 \mid x)\{-(1-\eta)\exp(-c) - \eta\} + P(Y=-1 \mid x)\{(1-\eta)\exp(c) + \eta\} = 0.$$

Thus, the population minimizer $F_\eta^*(x)$ satisfies the following equations:

$$\begin{aligned} \frac{P(Y=1 \mid x)}{P(Y=-1 \mid x)} &= \frac{(1-\eta)\exp(F_\eta^*) + \eta}{(1-\eta)\exp(-F_\eta^*) + \eta} \\ P(Y=1 \mid x) &= \frac{(1-\eta)\exp(F_\eta^*) + \eta}{\sum_{y'\in\{-1,1\}}(1-\eta)\exp(y'F_\eta^*) + \eta}. \end{aligned}$$

This completes the proof. □

2.5.3 Training error of AdaBoost

Freund and Schapire (1997) shows that the training error of the ordinary AdaBoost decreases exponentially.

Theorem 29. *Let $F_T(x)$ be a discriminant function constructed by the ordinary AdaBoost algorithm. Then, the training error of* $\operatorname{sign}(F_T(x))$ *is bounded above by*

$$L_D(\operatorname{sign}(F_T(x))) \le 2^T \Pi_{t=1}^T \sqrt{\epsilon_t(f_{j(t)})(1-\epsilon_t(f_{j(t)}))}.$$

Proof. In the AdaBoost algorithm, the discriminant function is constructed as

$$F_T(x) = \sum_{t=1}^{T} \alpha_t f_{j(t)}(x).$$

The coefficient α_T satisfies

$$\alpha_T = \frac{1}{2} \ln \frac{1 - \epsilon_T(f_{j(T)})}{\epsilon_T(f_{j(T)})}$$

where $\epsilon_t(f) = \sum_{i=1}^{n} I(Y_i \neq f(X_i)) w_{t-1}(i)$. Recall that the weight $w_{T-1}(i)$ is written as

$$w_{T-1}(i) = \frac{1}{Z_t} \exp(-Y_i F_{T-1}(X_i)),$$

where

$$Z_T = \sum_{i=1}^{n} \exp(-Y_i F_{T-1}(X_i)) = n A_D(F_{T-1}).$$

Then, we have

$$\begin{aligned}
A_D(F_T) &= \frac{1}{n} \sum_{i=1}^{n} \exp(-Y_i F_T(X_i)) \\
&= \frac{1}{n} \sum_{i=1}^{n} I(Y_i \neq f_{j(T)}(X_i)) \exp(\alpha_T) Z_T w_{T-1}(i) + I(Y_i = f_{j(T)}) \exp(-\alpha_T) Z_t w_{T-1}(i) \\
&= \frac{Z_T}{n} \exp(\alpha_T) \epsilon_T + \frac{Z_T}{n} \exp(-\alpha_T)(1 - \epsilon_T) \\
&= \frac{Z_T}{n} \left\{ \sqrt{\frac{1 - \epsilon_T}{\epsilon_T}} \epsilon_T + \sqrt{\frac{\epsilon_T}{1 - \epsilon_T}} (1 - \epsilon_T) \right\} \\
&= A_D(F_{T-1}) 2 \sqrt{(1 - \epsilon_T(f_{j(T)})) \epsilon_T(f_{j(T)})}.
\end{aligned}$$

Therefore, we have

$$A_D(F_T) = 2^T \Pi_{t=1}^{T} \sqrt{(1 - \epsilon_t(f_{j(t)})) \epsilon_t(f_{j(t)})}.$$

Combined with the fact that $L_D(F_T) \leq A_D(F_T)$, this completes the proof. □

Similar upperbound of training error can be obtained for AdaBoost.M2.

2.5.4 Property of generalization error

Preliminary results on the bound of generalization error is derived by Freund and Schapire (1997).

Freund and Schapire (1997) derived an upperbound of generalization error of AdaBoost.

Theorem 30 (Upperbound of generalization error). *Let* $\Theta_T(\mathcal{C})$ *be*

$$\Theta_T(\mathcal{C}) = \{\text{sign}(\sum_{t=1}^{T} \alpha_t f_{j(t)} - b) \,|\, a, b, \alpha_1, \alpha_2, \cdots, \alpha_T \in R\,, \forall t,\, f_{j(t)} \in \mathcal{C}\}$$

With at least probability $1 - \epsilon$*,*

$$\sup_{F \in \Theta_T(\mathcal{C})} |L_D(F) - L(F)| \leq 2\sqrt{\frac{d(\ln(\frac{2n}{d} + 1)) + \ln\frac{9}{\epsilon}}{n}}$$

where $d = 2(V + 1)(T + 1)\log_2(e(T + 1))$*.*

Schapire et al. (1998) derived the upperbound of generalization error that does not depend on T.

Theorem 31 (Margin bound of boosting). *Let* $\theta > 0$ *and* $\delta \in (0, 1]$*. For any* $f \in \text{conv}(\mathcal{C})$*,*

$$P(Yf(X) \leq 0) \leq P_D(Yf(X) \leq \theta) + 2\exp(-T\theta^2/8) + \sqrt{\frac{\ln(T(T+1)^2 J^T/2\delta)}{2n}}.$$

where $P_D(\cdot)$ *denotes the empirical probability of its argument.*

Proof. $f \in \text{conv}(\mathcal{C})$ has the form as

$$f(x) = \sum_{j=1}^{J} \alpha_t f_j(x)$$

where $f_j \in \mathcal{C}$ and the weights $\{\alpha_j\}$ are positive and satisfy $\sum_{j=1}^{J} \alpha_j = 1$. Therefore, $\{\alpha_j\}_{j=1}^{J}$ can be regarded as probabilities on $\mathcal{C}$. Define $\overline{\mathcal{F}}$ as

$$\overline{\mathcal{F}} = \{F(x) = \frac{1}{T}\sum_{t=1}^{n} f_{j(t)}(x) \,|\, f_{j(t)} \in \mathcal{C}\}$$

where $j(t)$ is independently chosen at random from $\{1, 2, \cdots, J\}$ according to the probability $\{\alpha_j\}_{j=1}^{J}$. Thus, f can be associated with a distribution over $\overline{\mathcal{F}}$ as defined by the coefficients $\{\alpha_j\}_{j=1}^{J}$.

In general, for two events A and B,

$$P(A) = P(A \cap B) + P(A \cap B^c) \leq P(A) + P(A \cap B^c).$$

Then, we have, for any fixed $g \in \overline{\mathcal{F}}$ and $\theta > 0$,

$$P(Yf(X) \leq 0) \leq P(Yg(X) \leq \theta/2) + P(Yg(X) > \theta/2 \cap Yf(X) < 0).$$

Taking both side of this inequality with respect to the distribution on $\overline{\mathcal{F}}$, we have

$$\begin{aligned} P(Yf(X) \leq 0) &\leq E[P(Yg(X) \leq \theta/2 \,|\, g)] + E[P(Yg(X) > \theta/2 \cap Yf(X) \leq 0 \,|\, g)] \\ &= E[P(Yg(X) \leq \theta/2 \,|\, g)] + E[P(Yg(X) > \theta/2 \cap Yf(X) \leq 0 \,|\, X, Y)] \\ &\leq E[P(Yg(X) \leq \theta/2 \,|\, g)] \\ &\quad + E[P(Yg(X) > \theta/2 \,|\, Yf(X) \leq 0, X, Y)] \end{aligned} \tag{33}$$

Due to Hoeffding's inequality (Theorem 5), the inside of the expectation of the second term is upperbounded as

$$\begin{aligned} P(Yg(X) > \theta/2 \,|\, Yf(X) \leq 0 \,|\, X, Y) &\leq P(Yg(X) - Yf(X) > \theta/2 \,|\, Yf(X) \leq 0 \,|\, X, Y) \\ &\leq \exp(-2\frac{\theta^2}{4}/(\sum_{t=1}^{T}\left(\frac{2}{T}\right)^2)) \\ &= \exp(-T\theta^2/8). \end{aligned}$$

The right-hand side does not depend on X and Y. Thus, the second term in (33) has the same upperbound. The inside of the expectation of the first term in (33) is upperly bounded as follows: By applying Hoeffding's inequality, for a fixed $g \in \overline{\mathcal{F}}$ and $\theta > 0$, we have

$$P\left(P(Yg(X) \leq \theta/2) - P_D(Yg(X) \leq \theta/2) > \epsilon\right) \leq \exp(-2n\epsilon^2).$$

The number of candidates of g (the number of possible combinations) is $|J|^T$ where J is $|\mathcal{C}|$ as was described before. The candidates of θ is $\{2t/T\}_{t=0}^{T}$ and thus its number is $(T+1)$. Therefore, the probability of that there exist at least $g \in \overline{\mathcal{F}}$ and $\theta > 0$ such that

$$P(Yg(X) \leq \theta/2) - P_D(Yg(X) \leq \theta/2) > \epsilon$$

is at most $(T+1)|\mathcal{C}|^T \exp(-2n\epsilon^2)$. This implies with at least probability $1-\epsilon$,

$$P(Yg(X) \leq \theta/2) \leq P_D(Yg(X) \leq \theta/2) + \sqrt{\frac{\ln((T+1)J^T/\epsilon)}{2n}}.$$

The empirical probability in the first term is bounded as we did above:

$$\begin{aligned} P_D(Yg(X) \leq \theta/2) &\leq P_D(Yf(X) \leq \theta) + P_D(Yg(X) \leq \theta/2 \,|\, Yf(X) \leq \theta) \\ &\leq P_D(Yf(X) \leq \theta) + P_D(Yg(X) \leq \theta/2 \,|\, Yf(X) \leq \theta) \\ &\leq P_D(Yf(X) \leq \theta) + \exp(-T\theta^2/8). \end{aligned}$$

Combined these results, with at least probability $1-\epsilon$

$$P(Yf(X) \leq 0) \leq P_D(Yf(X) \leq \theta) + 2\exp(-T\theta^2/8) + \sqrt{\frac{\ln((T+1)J^T/\epsilon)}{2n}}.$$

This inequality holds for any $T' = 1, 2, \cdots, T$ with at least probability $1 - \sum_{T'=1}^{T} \epsilon$ since the probability of the occurrence of several events is at most the sum of the probability of the occurrence of each event. Therefore, taking $\epsilon = 2\delta/T(T+1)$, we have, with at least probability $1-\delta$,

$$P(Yf(X) \leq 0) \leq P_D(Yf(X) \leq \theta) + 2\exp(-T\theta^2/8) + \sqrt{\frac{\ln(T(T+1)^2J^T/2\delta)}{2n}}.$$

□

2.5.5 Comparison between ordinary boosting and regularized boosting

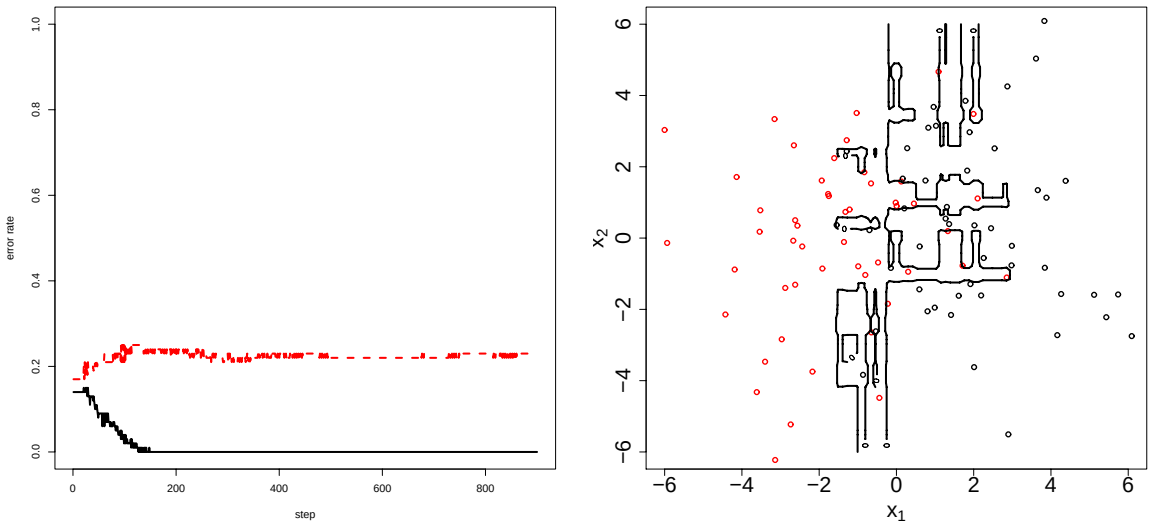
We show a simple example of binary classification to illustrate the effectiveness of this type of regularized boosting. Suppose that the prior probability $P(Y=1) = P(Y=-1) = 1/2$ and that $P(x \,|\, Y=1) = \mathcal{N}(\mu_1, 3I_2)$ and $P(x \,|\, Y=-1) = \mathcal{N}(\mu_{-1}, 3I_2)$ where $\mu_1 = (-2, 0)^T$ and $\mu_{-1} = (2, 0)$. Here, $\mathcal{N}(\mu, \sigma^2)$ denotes a normal density function with mean μ and covariance matrix Σ and I_M denotes an M-dimensional identity matrix. In this case, the target discriminant function $F^*_{\mathrm{ada}}(x)$ defined in Eq. (31) is easily calculated as

$$F^*_{\mathrm{ada}}(x) = \frac{1}{12}\{\|x-\mu_{-1}\|^2 - \|x-\mu_1\|^2\}.$$

The Bayes classifier g^* in this example is easily calculated as

$$g^*(x) = -\mathrm{sign}((x)_1).$$

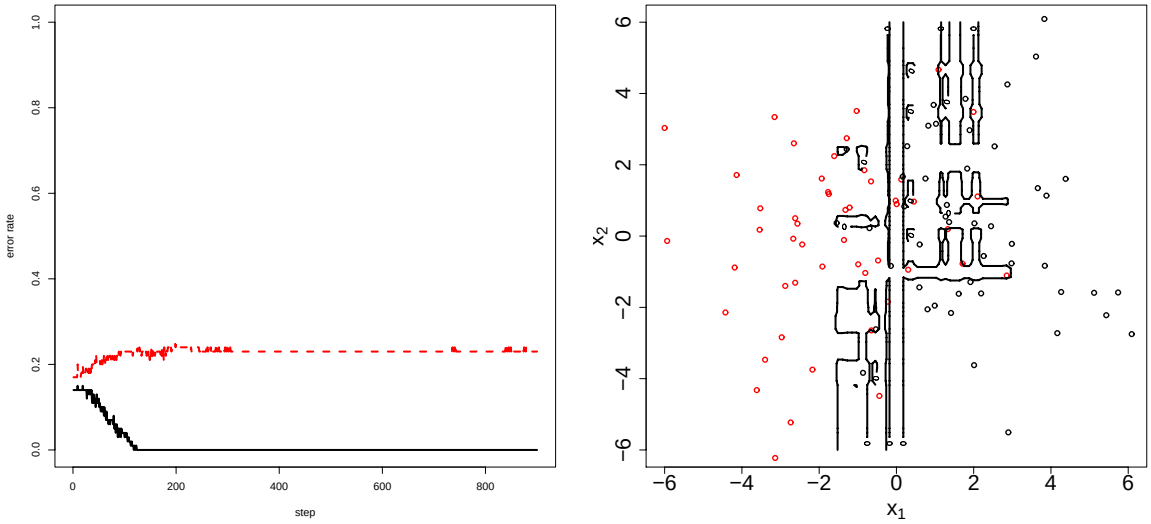
When we use decision stump $g^* = f^{\mathrm{s}}(x; 1, 0, -1)$. Thus, this example is considerably easy problem for AdaBoost. However, inspection of Figure 2 shows that AdaBoost performs somewhat poorly in this example. AdaBoost found the optimal base classifier $f^{\mathrm{s}}(x; 1, 0, -1)$ at the first step and then overfitted to the training data as the step increases. As a result, the decision boundary of AdaBoost was too complicated (Figure 2).



(a) (b)

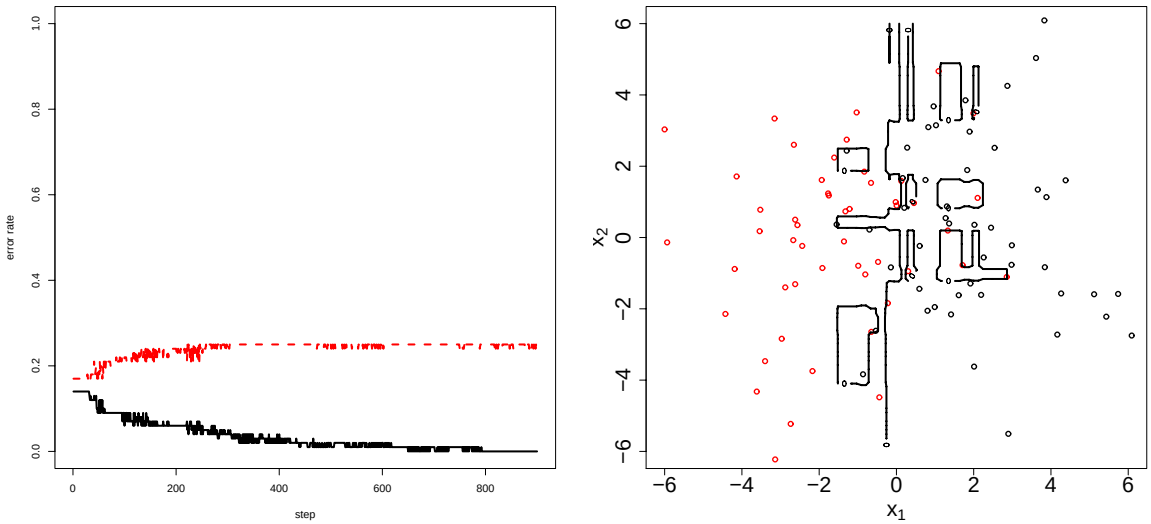
Figure 2: The panel (a) shows the plot of error rates of AdaBoost prediction for gaussian data. The panel (b) shows the plot of AdaBoost prediction over training data. Each color corresponds to each class label. Black line shows the decision boundary given by AdaBoost.

AdaBoost.L1 (See Section 2.4.3) performs better when its regularization parameter λ is appropriately chosen (Figure 3-7). When λ is sufficiently large, the decision boundary of AdaBoost.L1 is quite similar to that of AdaBoost. As λ becomes smaller, AdaBoost.L1 constructed more smooth decision boundary. Specifically, AdaBoost.L1 with $\lambda = 1$ yielded the Bayes classifier. Therefore, we may control the smoothness of decision boundary by tuning the regularization parameter λ. One way to choose λ was discussed by Lugosi and Vayatis (2004). The other way is to choose λ yielding the best test error estimated by cross-validation.



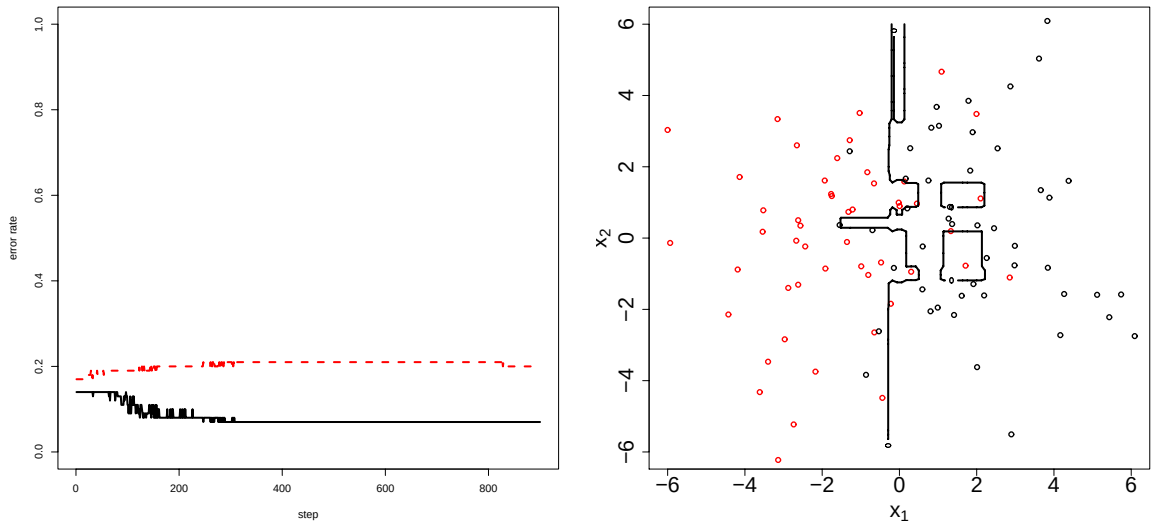
(a) (b)

Figure 3: The panel (a) shows the plot of error rates of AdaBoost.L1 with $\lambda = 500$ for gaussian data. The panel (b) shows the training data (each color corresponds to each class label) with the decision boundary (black line) that was constructed by AdaBoost.L1.



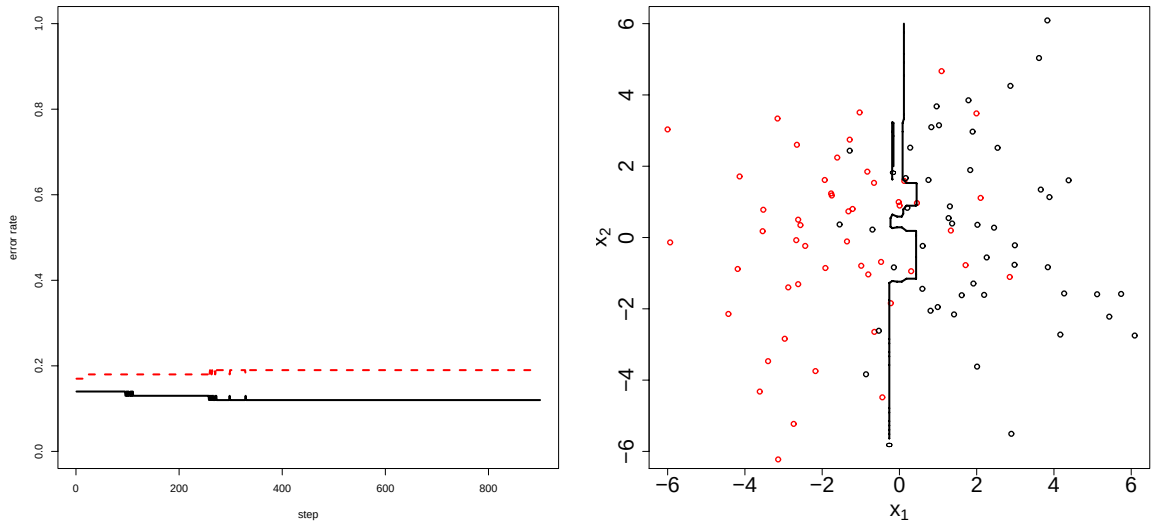
(a) (b)

Figure 4: The panel (a) shows the plot of error rates of AdaBoost.L1 with $\lambda = 20$ for gaussian data. The panel (b) shows the training data (each color corresponds to each class label) with the decision boundary (black line) that was constructed by AdaBoost.L1.



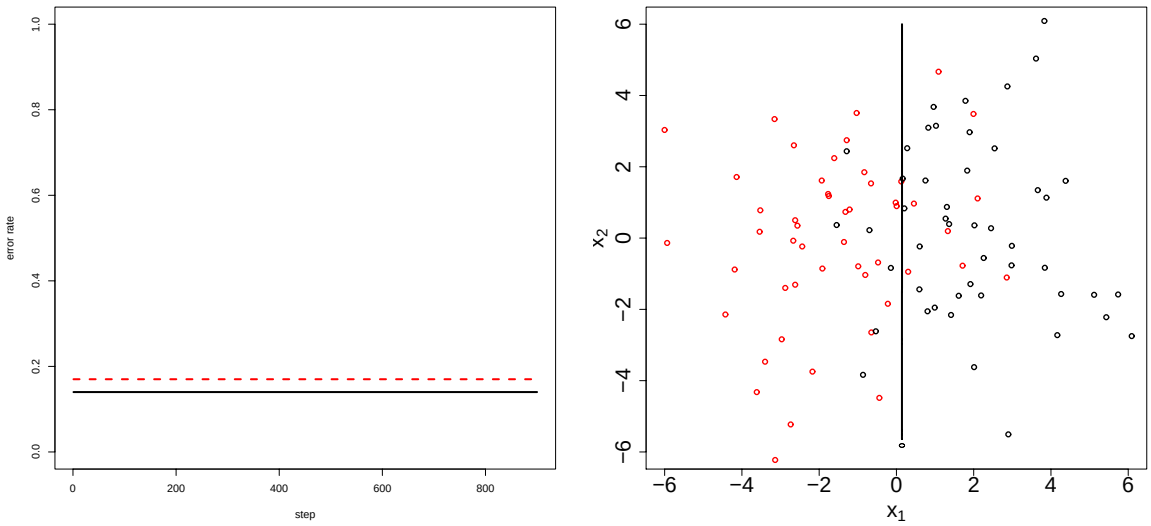
(a) (b)

Figure 5: The panel (a) shows the plot of error rates of AdaBoost.L1 with $\lambda = 10$ for gaussian data. The panel (b) shows the training data (each color corresponds to each class label) with the decision boundary (black line) that was constructed by AdaBoost.L1.



(a) (b)

Figure 6: The panel (a) shows the plot of error rates of AdaBoost.L1 with $\lambda = 4$ for gaussian data. The panel (b) shows the training data (each color corresponds to each class label) with the decision boundary (black line) that was constructed by AdaBoost.L1.

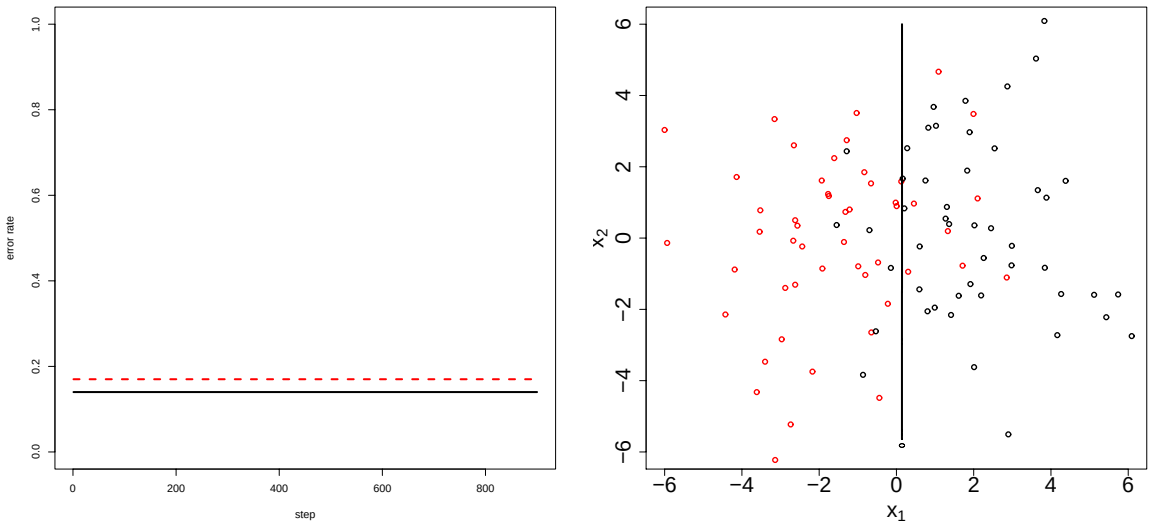


(a) (b)

Figure 7: The panel (a) shows the plot of error rates of AdaBoost.L1 with $\lambda = 1$ for gaussian data. The panel (b) shows the training data (each color corresponds to each class label) with the decision boundary (black line) that was constructed by AdaBoost.L1.

3 Application to shark bycatch data

There is a growing body of statistical literature on new algorithmic methods that are good predictive techniques with complex data, such as the types of data frequently used in fisheries analyses. Such methods include *neural networks*, *random forests* and *boosting* (Hastie et al., 2001; Breiman, 2001; Breiman, 1999). These algorithmic techniques are not model based as are generalized linear models (GLMs) (McCullagh and Nelder, 1989) and generalized additive models (GAMs) (Hastie and Tibshirani, 1990). GLMs and GAMs are tools used in fisheries data analysis to standardize bycatch and catch per unit effort data, as well as to identify factors leading to increased levels of bycatch (incidental mortality of non-target species) and to predict bycatch. The fishery-dependent data used in these analyses are usually collected opportunistically. Such data are typically characterized by a lack of a balanced sampling design and may contain correlated and/or weak features. These aspects can make analysis with GLM and GAM techniques problematic. With the increasing sizes of databases of fishery-dependent data, opportunities exist for exploring statistical techniques that can yield stable predictions or serve as exploratory analysis tools for these types of data.



(a) (b)

Figure 7: The panel (a) shows the plot of error rates of AdaBoost.L1 with $\lambda = 1$ for gaussian data. The panel (b) shows the training data (each color corresponds to each class label) with the decision boundary (black line) that was constructed by AdaBoost.L1.

3 Application to shark bycatch data

There is a growing body of statistical literature on new algorithmic methods that are good predictive techniques with complex data, such as the types of data frequently used in fisheries analyses. Such methods include *neural networks*, *random forests* and *boosting* (Hastie et al., 2001; Breiman, 2001; Breiman, 1999). These algorithmic techniques are not model based as are generalized linear models (GLMs) (McCullagh and Nelder, 1989) and generalized additive models (GAMs) (Hastie and Tibshirani, 1990). GLMs and GAMs are tools used in fisheries data analysis to standardize bycatch and catch per unit effort data, as well as to identify factors leading to increased levels of bycatch (incidental mortality of non-target species) and to predict bycatch. The fishery-dependent data used in these analyses are usually collected opportunistically. Such data are typically characterized by a lack of a balanced sampling design and may contain correlated and/or weak features. These aspects can make analysis with GLM and GAM techniques problematic. With the increasing sizes of databases of fishery-dependent data, opportunities exist for exploring statistical techniques that can yield stable predictions or serve as exploratory analysis tools for these types of data.

Few examples of the application of supervised learning methods to fisheries data are available in the published literature. *Decision trees* have been used to relate species groups to environmental variables and sampling locations (Taquet et al., 1997; Tserpes et al., 1999). Some earlier methods for constructing decision trees have been found to be unstable in the sense that a small perturbation in the data causes large changes in predictions (Breiman, 1996b). As described in Section 2, recently developed methods for constructing decision trees that make use of ensembles of classifiers or iterative reweighting schemes tend to provide more stable predictions than earlier methods. Bagging is an effective technique for reducing instability of decision trees, which is based on *resampling* the data. *Boosting* is a recently developed method that employs an *adaptively resampling and combining* algorithm to yield stable predictions. As a result, it was repoted by Breiman (1998) that decision trees constructed by boosting are often more stable than those constructed by bagging. In addition, one interesting thing is that boosting method can be effective at yielding stable predictions for problems involving correlated features where some conventional methods may fail to perform well. This property makes boosting an attractive tool for analysis of fisheries bycatch data when problems can be cast in a presence/absence context.

In many situations, bycatch data can be meaningfully studied in terms of presence/absence. For example, presence/absence often adequately describes the bycatch of rare species, such as turtles. For vulnerable species, such as sharks with delayed maturation and low reproductive potential, a precautionary approach to bycatch mitigation may be desirable, and thus, it may be preferable to reduce the data to presence/absence. In addition, if counting error is severe, bycatch data may only be informative on the occurrence of bycatch, not the exact magnitude. If bycatches are large but rare, there may be insufficient observations of non-zero bycatch to build a good predictive model for the magnitude of bycatch.

In this section we demonstrate the use of AdaBoost with shark bycatch data from the eastern Pacific Ocean tuna purse-seine fishery. In Section 3.1, we present a graphical tool, the score plot, for visualizing the dependence of bycatch on individual features. This tool is similar to that used to summarize additive contributions from a logistic GAM model. We explain the details of shark bycatch data in Section 3.2. In Section 3.3, AdaBoost with decision stumps is applied to shark bycatch presence/absence data. We compare the results of AdaBoost to those of the logistic GAM in Section 3.4. In Section 3.5, we

demonstrate that AsymBoost with decision stumps reduces FNR at cost of slight increase in test error. Several discussions appear in Section 3.6.

3.1 Graphical display of contribution of each feature

Score plots are a graphical representation of the contribution of each feature to the discriminant function. The discriminant function $F_T(\boldsymbol{x})$ built using AdaBoost with decision stumps can be written as the sum of contributions from each of the m features:

$$\begin{aligned} F_T(x) &= \sum_{t=1}^{T} \alpha_t f^{\mathrm{s}}_{j(t)}(x; m_t, b_t, s_t) \\ &= \sum_{\{t|m_t=1\}} \alpha_t f^{\mathrm{s}}_{j(t)}(x; 1, b_t, s_t) \\ &\quad + \sum_{\{t|m_t=2\}} \alpha_t f^{\mathrm{s}}_{j(t)}(x; 2, b_t, s_t) \\ &\quad \vdots \\ &\quad + \sum_{\{t|m_t=M\}} \alpha_t f^{\mathrm{s}}_{j(t)}(x; M, b_t, s_t) \\ &= \sum_{m=1}^{M} S_m((x)_m) \end{aligned} \tag{34}$$

The last equality is obtained by defining $S_m((x)_m)$, called the *score function of* $(x)_m$, as

$$S_m((x)_m) = \sum_{\{t|m_t=m\}} \alpha_t f^{\mathrm{s}}_{j(t)}(x; m_t, b_t, s_t) \tag{35}$$

where $\{t \,|\, m_t = m\}$ denotes the collection of values of the index t where the t-th classifier $f^{\mathrm{s}}_{j(t)}$ is based on the m-th feature. A plot of $S_m((x)_m)$ versus $(x)_m$ is referred to as a *score plot*. Given new data x, Eq. (34) shows that $F_T(x)$ predicts a label according to the sign of the sum of the scores of each feature.

We introduce the interpretation of $S_m((x)_m)$ based on the results of Friedman et al. (2000). The logistic GAM models the log-odds by a linear function of x:

$$\ln \frac{P(Y=1 \,|\, x)}{P(Y=-1 \,|\, x)} = \sum_{m=1}^{M} s_m((x)_m; \theta_m) \tag{36}$$

where $P(Y = 1 \,|\, x)$ denotes the probability that $Y = 1$ conditional on x, $P(Y = -1 \,|\, x) = 1 - P(Y=1 \,|\, x)$, and $s_m((x)_m; \theta_m)$ denotes a function of $(x)_m$ specified by the

parameter vector θ_m. For *linear logistic regression*, $s_m((x)_m; a_m) = a_m(x)_m$ where a_m is a scalar parameter. For logistic GAM in our use, s_m represents a smoothing spline. Given a feature vector x, the prediction of the logistic GAM is the label $\{-1, 1\}$ with the greatest conditional probability. This prediction rule is the same as assigning the label according to the sign of the right-hand side of Eq. (34). Therefore, the right-hand side of the Eq. (36) is equivalent to Eq. (34) in terms of the prediction rule. The right-hand side of the Eq. (36) is referred to as the *discriminant function of the logistic GAM* and each term $s_m((x)_m)$ the *score of* $(x)_m$. For AdaBoost the true log-odds, denoted by F^*_{ada}, minimizes the expected exponential loss function as described in Section 2.5.2. Thus, logistic GAM and AdaBoost aim at the same goal, aside from a positive constant multiplier, which has no effect on prediction. The score function of each feature given by the logistic GAM approximates the relationship between log-odds and each feature with a smoothing spline, whereas AdaBoost with decision stumps approximates the relationship with a linear sum of shifted step functions. Thus, it is meaningful to compare the properties of the score function given by the logistic GAM with that given by AdaBoost. We present such a comparison in Section 3.4.

The score plot provides a graphical means of screening features. A feature is informative if its score function has a large absolute value over most of the range of the feature. Given a value of even a single feature, the prediction based on it is relatively sure if the score of the feature takes on a large absolute value and if the pointwise confidence region for the score function does not include zero at many points. Thus, inspection of score plots can indicate which features are informative.

The *leave-one-out test error* can also be used to measure the informativeness of each feature. The leave-one-out test error for feature $(x)_m$ is defined as the test error when $(x)_m$ is excluded from building the classifier. Inspection of score plot shows the informativeness of each feature based on only training data. By contrast, inspection of the leave-one-out test error is based on both training and test data since leave-one-out test error is the test error of the discriminant function that was constructed on the training data. This implies that the leave-one-out test error can be a more effective means of determining which features are informative for prediction because it is not affected by overfitting.

The score function also allows us to determine features which are unlikely to be useful for discrimination. A feature is not likely to be useful for discrimination if its score is almost zero. To see this quantitatively, we propose the following rough criterion, denoted

as $|S_m|_{\max}$.

$$|S_m|_{\max} = \frac{\max_{x'} |S_m(x')|}{(1/n) \sum_{i=1}^{n} |\sum_{m'=1}^{M} S_{m'}((X_i)_{j'})|}$$

where $\{X_1, X_2, \cdots, X_n\}$ are test data. $|S_m|_{\max}$ is the ratio of the maximum absolute value of m-th feature's score to the averaged value of the discriminant function over the test data set. Features that are unlikely to be useful for discrimination will have relatively smaller values of $|S_m|_{\max}$. Note that we cannot know which features are most informative for prediction entirely from $|S_m|_{\max}$ because the score function is based only on the training data and can be affected by overfitting. For this reason, $|S_m|_{\max}$ is not useful for selecting informative features because large values of $S_m((x)_m)$ over a small subset of the values of $(x)_m$ will lead to a large $|S_m|_{\max}$, even if $S_m((x)_m)$ is close to zero over the rest of the values of $(x)_m$. However, it is unlikely that overfitting leads to large absolute values of $S_m((x)_m)$ over the majority of the range of $(x)_m$. For this reason, $|S_m|_{\max}$ and the leave-one-out test error will often agree on a feature's utility. Features associated with very large $|S_m|_{\max}$ are typically informative in terms of the leave-one-out test error, while features that do not have very large values of $|S_m|_{\max}$ are not necessarily informative in terms of the leave-one-out test error.

3.2 Data sets

Observers of IATTC aboard large fishing vessels of the international tuna purse-seine fishery record bycatches of several species, including sharks, that are incidentally caught as part of fishing operations (Bayliff, 2001). This surface fishery operates primarily in oceanic waters of the eastern Pacific Ocean (Watters, 1999). In addition to estimating the amounts of tuna catch and bycatch, observers record details about the local environment (*e.g.*, sea surface temperature) and details of the fishing operations and fishing gear. Silky sharks (*Carcharhinus falciformis*) dominate the shark bycatch in this fishery[2], with most bycatch occurring in purse-seine sets on tunas associated with floating objects ('floating object' sets)(IATTC, 2004). This analysis focuses on the bycatch of large silky sharks (>150 cm total length), which are most likely sub-adult and adult animals (Oshitani et al., 2003). Large silky sharks typically comprise about one-third of the silky shark

[2]Unpublished data of the Inter-American Tropical Tuna Commission, 8604 La Jolla Shores Drive, La Jolla, California 92037-1508. Contact person: Marlon Roman-Verdesoto.

bycatch.

As an example of the application of AdaBoost to fisheries data, we regard the silky shark bycatch problem as a binary classification problem. We use AdaBoost to develop a classifier for predicting the occurrence of large silky shark bycatch (presence:$Y = 1$; absence:$Y = -1$) in floating objects sets. To build this classifier, we use data on $3,772$ floating objects sets from 2001. Thirty percent of these sets had bycatch of one or more large silky sharks. In spite of the predominance of sets with no sharks, shark bycatch is distributed throughout the area occupied by the fishery (Figure 8). Candidate features of the occurrence of silky shark bycatch include descriptors of the set location (latitude, longitude), local environment (*e.g.*, sea surface*e.g.*, temperature), characteristics of the floating object and the purse-seine net, and characteristics of the community at the floating object (*e.g.*, biomass of other groups of animals). In total, 16 features were included in the analysis. Details of these features are presented in Table 2. Some features are correlated (Table 3), which may lead to instability of several classical classification techniques.

Table 2: List of available features (with their abbreviations), the feature type and the models in which each feature was used. Each model contains the features marked with $\surd$. The feature *shp* takes on the values as as (1=cylindrical 2=polygonal 4=irregular 5=aggregated 6=other 7=spherical) depending on the shape of the floating object. The feature *col* takes on the values (1=red 2=green 3=orange 4=blue 5=yellow 6=black 7=white) depending on the color of the floating object.

Feature	abbrev.	type	Model I	Model II	Model III
Julian date	dat	Continuous	$\surd$	$\surd$	$\surd$
latitude (decimal degrees)	lat	Continuous	$\surd$	$\surd$	$\surd$
longitude (decimal degrees)	lon	Continuous	$\surd$	$\surd$	$\surd$
amount of small fishes (numbers of animals)	sml	Continuous	$\surd$	$\surd$	$\surd$
amount of tunas that could have been prey for sharks (metric tons)	pry	Continuous	$\surd$	$\surd$	$\surd$
amount of target species of tunas (metric tons)	tgt	Continuous	$\surd$	$\surd$	$\surd$
amount of non-target species of tunas (metric tons)	ntgt	Continuous	$\surd$		
amount of bycatch of species other than tunas (number of animals, excluding silky sharks)	ntn	Continuous	$\surd$		
start time of the purse-seine set (local time)	time	Continuous	$\surd$		
percentage of the floating object covered with epibiota	epi	Continuous	$\surd$	$\surd$	
depth of the floating object (meters)	dev	Continuous	$\surd$	$\surd$	
depth of the purse-seine net (fathoms)	net	Continuous	$\surd$	$\surd$	
sea surface temperature (degrees Centigrade)	tmp	Continuous	$\surd$	$\surd$	
Beaufort sea state	beau	Categorical	$\surd$	$\surd$	
shape of the floating object	shp	Categorical	$\surd$	$\surd$	
color of the floating object	col	Categorical	$\surd$	$\surd$	

Table 3: Sample correlations between several features in the shark bycatch data.

	dat	lat	lon	pry
lat	0.36			
lon	-0.41	-0.33		
pry	-0.10	-0.03	-0.02	
tgt	-0.20	-0.19	0.08	0.78

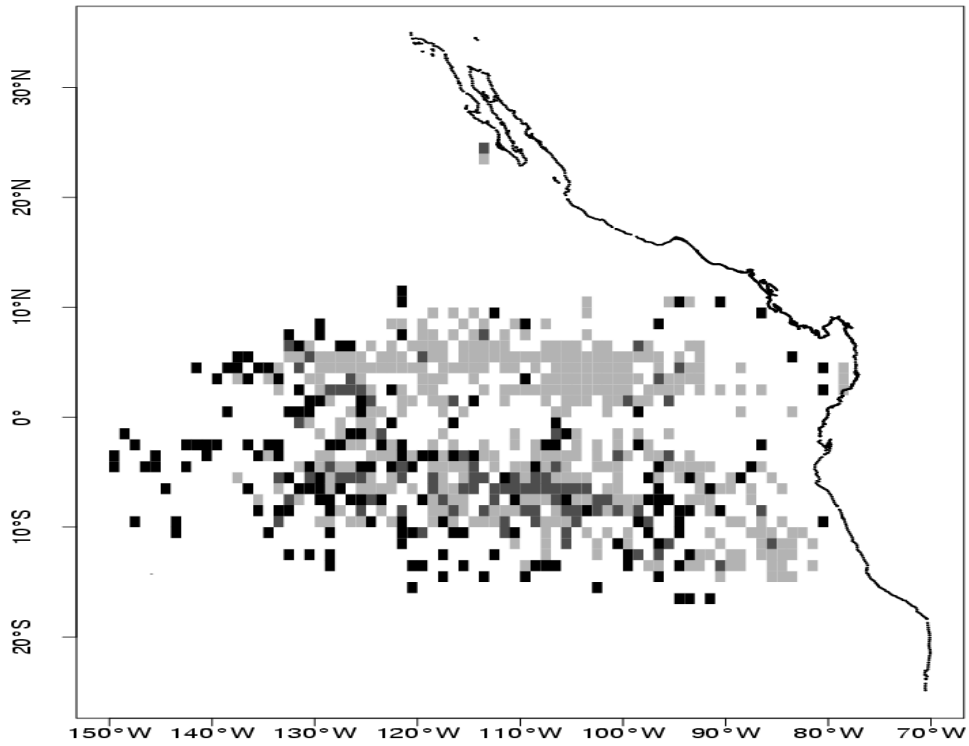


Figure 8: Proportion of sets with shark bycatch by one degree square area. Light gray indicates values of 0, dark gray indicates values between 0 and 0.75, and black indicates values of greater than 0.75.

We develop classifiers for the occurrence of shark bycatch using AdaBoost, AsymBoost and logistic GAM for three models of increasing complexity (Table 2). Error rates, their standard deviations, $|S_m|_{\max}$ and the leave-one-out test error were calculated by averaging the results of 100 trials. The training data and the test data sets for each trial were

independently constructed by dividing the original data set as follows. Each observation of the original data set is assigned to either the training data set or the test data set with probability 1/2. For the purpose of illustrating feature effects, score functions for selected trials are presented. Approximate pointwise 95% confidence regions for score functions were computed from the bootstrap resampled data as follows. We made 200 bootstrapped data sets from the training data set that was used in that trial. By applying AdaBoost to each, we obtain 200 estimated discriminant functions. The upper bound of 95% confidence region is the plot of the fifth largest score and the lower bound is the plot of the fifth smallest score at each point.

3.3 Prediction by AdaBoost

Application of AdaBoost to the shark bycatch data demonstrates the point that AdaBoost with decision stumps gives very stable predictions with respect to the test error, even as model complexity increases (Table 4). It can be seen that there is a slight reduction in the average test error as the number of parameters increases. However, more importantly in this specific example, the standard deviations of the test error remain largely the same as the model complexity increases, illustrating the point that AdaBoost predictions remain relatively stable. Inspection of Table 4 also shows that the FNR was much larger than the FPR. Recall that the FPR is the probability that AdaBoost predicts the occurrence of shark bycatch when no sharks were caught, while the FNR is the probability that AdaBoost predicts no shark bycatch when in fact sharks were caught (see Eq. (26)). Although this situation would be expected given the predominance of sets in the data with no shark bycatch, it is not desirable from the point of view of identifying options for bycatch mitigation. We illustrate an improvement to this situation with application of AsymBoost in Section 3.5.

Table 4: AdaBoost test error rates and their standard deviations for each of the three models.

	Test Error(%)	False Positive Ratio(%)	False Negative Ratio(%)
model I	25.77 ± 0.76	10.75 ± 1.26	61.14 ± 3.06
model II	25.88 ± 0.76	10.63 ± 1.36	61.79 ± 2.98
model III	26.45 ± 0.70	9.48 ± 1.48	66.39 ± 3.36

Overfitting was not apparent with these data. k-fold cross validation (we used $k =$

10) is an effective technique for estimating the test error without referring to the test data set. Generally, k-fold cross validation should be used to determine the optimal iteration number (T), even though it requires additional computational cost. However, with AdaBoost the choice of the exact value of T is often not a critical issue since the increase of the test error caused by overfitting is relatively slow. In fact, the average test error in our analysis does not vary much after it reaches its minimum value at around step 130 (Figure 9). Therefore, even if the boosting iterations were stopped at step 200 (chosen heuristically), the average test errors are 26.77, 27.04 and 27.85 for model I, II and III, respectively. These are equal to or only slightly worse than those obtained when ten-fold cross-validation was used to select T. Note that this does not always hold when the number of samples is significantly small or when other types of base classifiers are used.

$|S_m|_{\max}$ and the leave-one-out test errors indicate the existence of two types of non-informative features. Inspection of Figure 10 shows that the following features, 'amount of non-target species of tunas' (ntgt), 'Beaufort sea state' (beau), 'shape of the floating object' (shp) and 'color of the floating object' (col), all have very low values of $|S_m|_{\max}$, suggesting that they are the least informative features for the training data. From Figure 11 we can see that 'amount of small fishes' (sml), 'amount of tunas that could have been prey for sharks' (pry), 'amount of target species of tunas' (tgt), 'amount of non-target species of tunas' (ntgt), 'start time of the purse-seine set' (time), 'percentage of the floating object covered with epibiota' (epi), 'Beaufort sea state' (beau), 'shape of the floating object' (shp) and 'color of the floating object' do not increase the test error significantly when they are removed. $|S_m|_{\max}$ tells us which features are non-informative for the training data, however, with the leave-one-out test error tells us which features are non-informative for the test data set. In particular, the features, 'amount of tunas that could have been prey for sharks' (pry), 'start time of the purse-seine set' (time), 'percentage of the floating object covered with epibiota' (epi) are slightly informative for the training data set, but are not informative for the test data set. This implies that these features contribute to overfit to the training data set. The features 'Beaufort sea state' (beau) and 'shape of the floating object' (shp) are neither informative for the training data set nor for the test data set. These features are of little use for classification of shark bycatch with this data set.

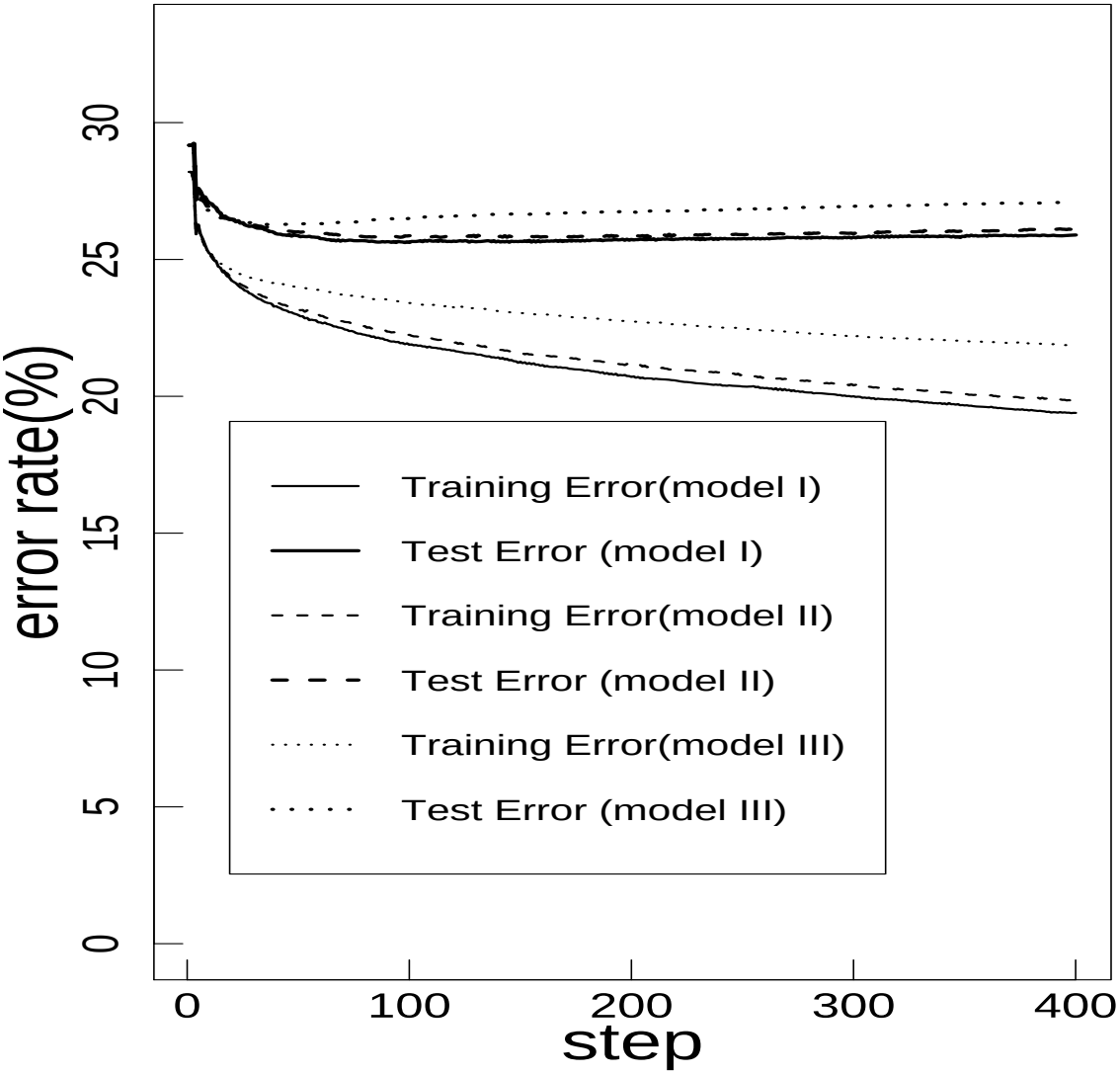


Figure 9: AdaBoost training and test errors for each of the three models as a function of the iteration number (step).

Informative features are most easily identified from the leave-one-out test error. Inspection of Figure 11 shows that the features: 'Julian date' (dat), 'longitude' (lon), 'latitude' (lat), have relatively large leave-one-out test error, well above the 95% confidence region of the original test error, and are thus the informative features for this data set. Most features were not found to be very informative with these data, although the features 'amount of bycatch of species other than tunas' (ntn) , 'depth of the floating object' (dev)

and ‘depth of the purse-seine net’ (net) and ‘sea surface temperature’ (tmp) are slightly informative. This indicates that, in this data set, there are no prominent features except ‘dat’, ‘lon’ and ‘lat’ that are useful for the prediction of the occurrence of bycatch. Highly informative features can also be identified from the score function, but care is needed. Informative features are those that have large score values over most range of the feature and narrow confidence regions with few zero-crossings. Inspection of Figure 12 shows that, for example, the score function of the features ‘Julian date’ (dat) and ‘longitude’ (lon) have these characteristics.

The shape of the score function shows how each feature contributes to the discriminant function. Inspection of Figure 12 shows that sharks were likely to be caught in sets made during the first part of the year, in sets made far from the coast and in sets with a large amount of bycatch of species other than tunas. The score plots also show when the prediction based on each feature is not reliable. For example, for purse-seine sets between about $90°W$ and $130°W$, the feature ‘longitude’ (lon) does not contribute strongly to the discriminant function since the confidence region crosses the zero line and the score function takes relatively small absolute values. Finally, we may easily see from the bottom right panel of Figure 12 that the feature ‘shape of the floating object’ (shp) does not contribute to the discriminant function over the entire range of its values, as already indicated by $|S_m|_{\max}$.

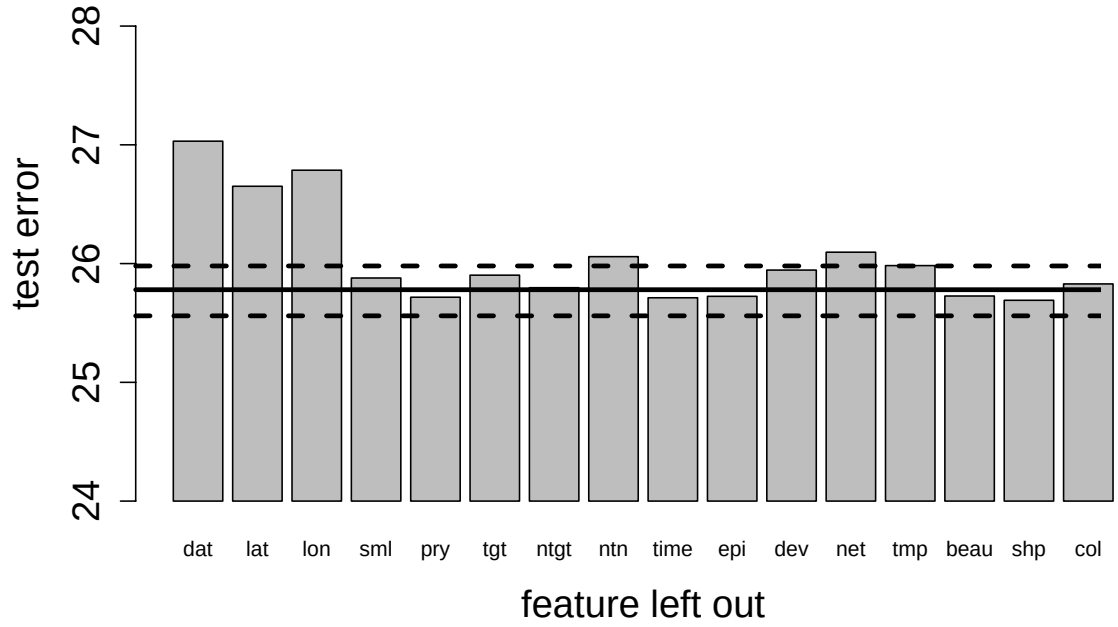


Figure 10: The average value of $|S_m|_{\max}$ for each feature of AdaBoost (averaged over 100 trials.)

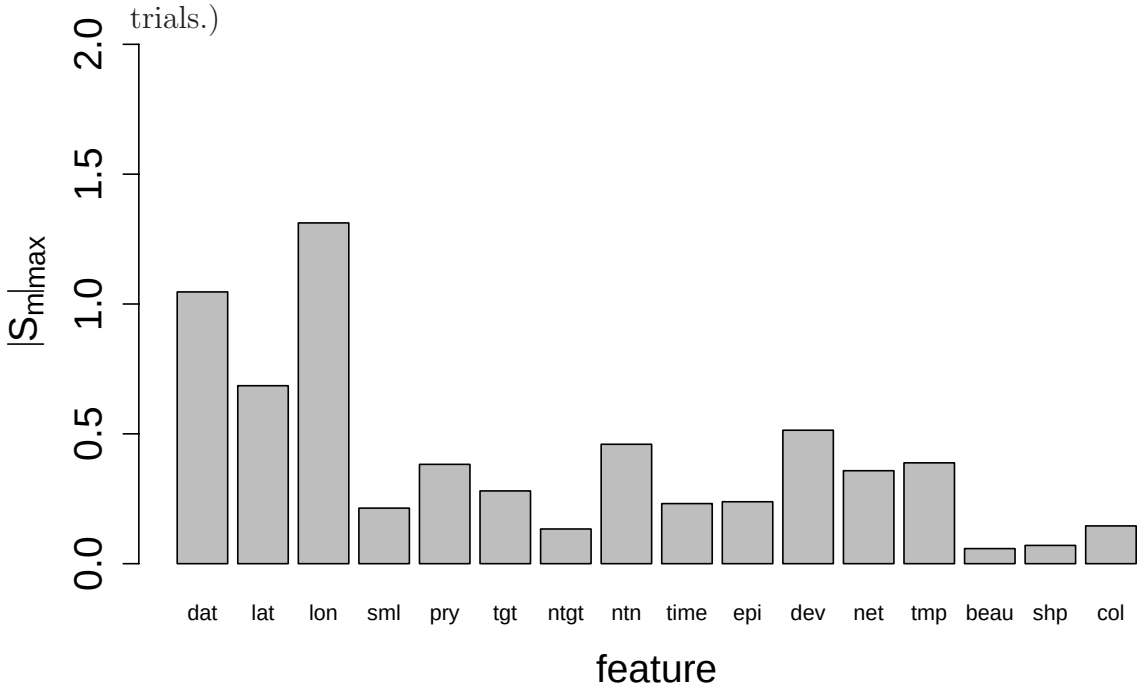


Figure 11: The average value of the leave-one-out test error of AdaBoost (averaged over 100 trials.) The thick horizontal line indicates the test error for model I (averaged over 100 trials) and the dashed lines indicate the upperbound and the lowerbound of its 95% confidence region (calculated by bootstrap).

3.4 Comparison with logistic GAM

Comparison of the prediction performance and the score functions of AdaBoost and logistic GAM shows that those of AdaBoost are more stable than those of logistic GAM. To obtain logistic GAM predictions for this comparison, we used the R package *mgcv* (version 0.9-6) in the R language. The mgcv implementation of GAM uses thin-plate regression splines (Wood, 2003) to represent the smooth functions and solves the multivariate smoothing parameter estimation problem by using the Generalized Cross Validation (GCV) criterion (Hastie et al., 2001, pp.216-217) with stable and efficient computational methods (Wood, 2000; Wood, 2004). The thin plate regression splines are optimal low rank smooths which do not have knots, thus avoiding the problems associated with "knot placement".

Table 5: Logistic GAM test error rates and their standard deviations for each of the three models.

	Test Error(%)	False Positive Ratio(%)	False Negative Ratio(%)
model I	26.28 ± 3.91	11.65 ± 6.60	60.74 ± 4.31
model II	26.48 ± 4.47	11.67 ± 8.4	61.29 ± 7.18
model III	26.44 ± 0.75	9.09 ± 1.18	67.25 ± 2.41

Inspection of Tables 4 and 5 shows that the prediction performance of AdaBoost and of the logistic GAM are almost equal in terms of the average test error; both techniques yielded an average test error of about 26%. However, it also shows that the standard deviation of the test error of AdaBoost is smaller than that of logistic GAM. There are two reasons why the prediction performance of logistic GAM varies. First, the logistic GAM suffers from high correlation between features, even if they are informative, as do some other classical methods. In fact, more careful inspection of Table 5 shows that the standard deviations of the test error of model I and II given by the logistic GAM are considerably larger than that of model III. We should note that '*mgcv*' package was already improved to be resistant to the high correlation between features. For our analysis, the '*mgcv*' package performed well in most trials, however, in a few trials, the test error of logistic GAM was extremely large. Such trials occurred four times for model I and five times for model II. By contrast, the test error of AdaBoost was stable for all trials. Second, the shape of the score function of the logistic GAM varies considerably ('edge effect'). We discuss edge effects from the point of view of the score plot below.

Comparison of the score functions of AdaBoost and of the logistic GAM shows that, in the presence of correlated features, the score functions of AdaBoost are more stable than those of the logistic GAM. Recall that the score function given by the logistic GAM approximates the relationship between the log-odds and each feature with a smoothing spline, whereas AdaBoost with decision stumps approximates this relationship with a linear sum of shifted step functions. Therefore, the shapes of the score functions given by the two methods would be expected to be somewhat different. However, in the trials where the test error of logistic GAM increased dramatically between model II and model I, the score functions of logistic GAM also took excessively large values and lost their shape, while those of AdaBoost remained very stable. This is illustrated in Figure 13 with the feature 'amount of target species of tunas' (tgt). The variability in the logistic GAM score function between trials indicates its instability, while AdaBoost attains better prediction performance with consistent score shape. We note that these variations are seen only in the score plots of weakly informative features and only in the region where the data are sparse. Thus, these variations do not cause much increase in the average test error, but they do inflate the standard deviation of the test error (Table 5). Note that the true score function of feature may change its shape in general when other features are added, unless conditional on the label y, that feature and the additional features are statistically independent.

Even for a specified set of features the score shape of the logistic GAM can be quite variable, while that of AdaBoost is generally stable. Inspection of the right panel of Figure 14 shows that the score shape of logistic GAM varies most in the region where the data are sparse. This is because the logistic GAM tries to estimate parameters based on only a few data points, which may vary considerably depending on the data set. This phenomenon is called an *edge effect*. Unbounded features, for example, 'amount of target species tunas' (tgt), tend to suffer from edge effects, regardless of their informativeness or the other features in the model, because they have regions where the data are sparse. By contrast, inspection of Figure 14 shows that the AdaBoost score functions have more stable shapes across the entire range of values of the feature. Note that, in the region where the data are dense, the score shape of AdaBoost and of the logistic GAM are not as variable and their shapes are similar, both indicating that the occurrence of shark bycatch increased with the amount of catch of target species of tunas. Because edge effects are most pronounced in regions where data are sparse, edge effects do not typically lead to large increases in

the average test error. However, given new data which has an outlier value for a feature that suffers from the edge effect, the prediction can be unreliable because the large value of score of that feature may dominate the value of discriminant function. In this sense, the score functions of AdaBoost are more tractable than those the logistic GAM. We note that the range of AdaBoost score was half of that of logistic GAM because AdaBoost approximates half the log-odds, while the logistic GAM approximates the log-odds (See section 3.1).

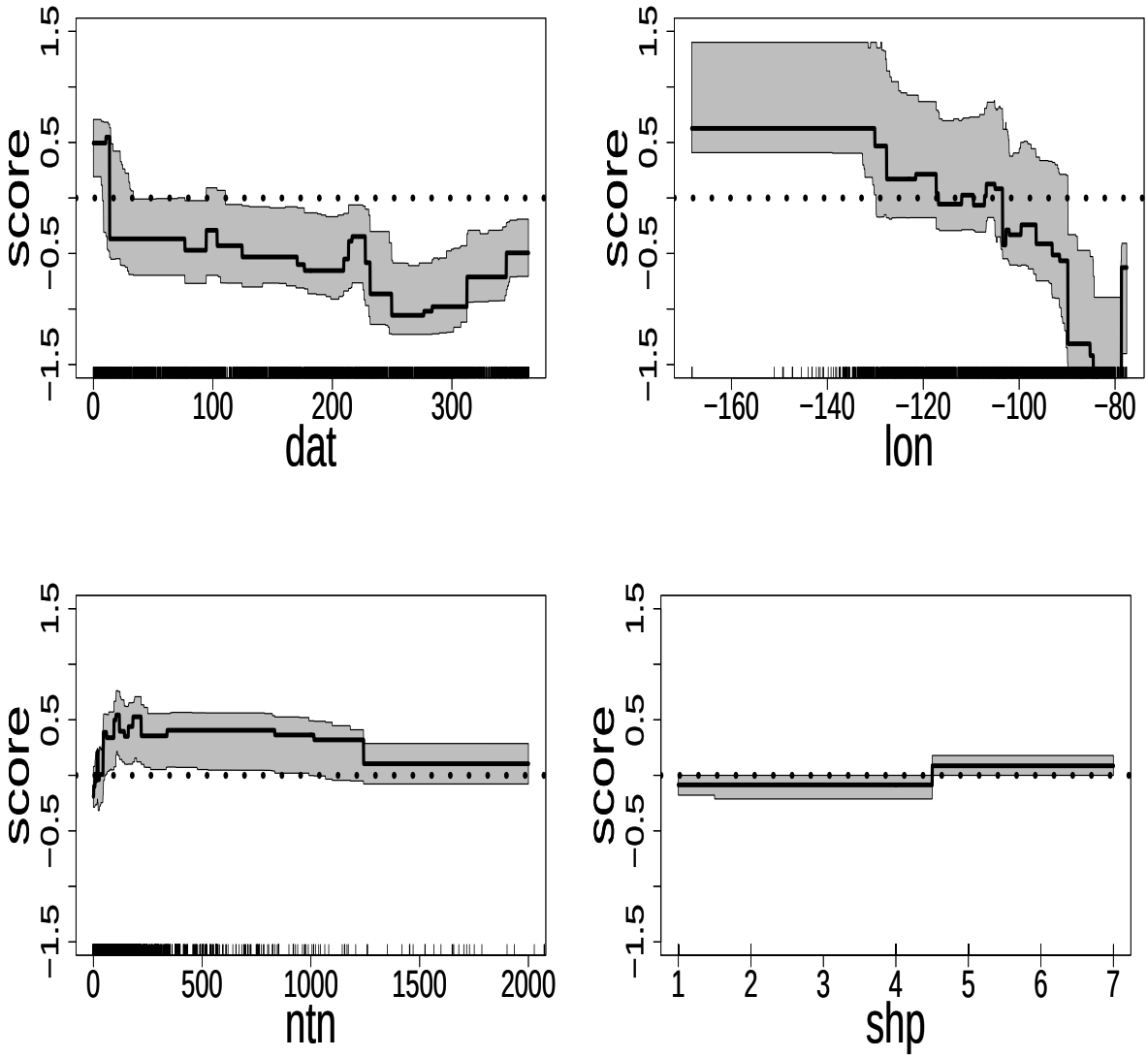


Figure 12: Score plots obtained from the bycatch prediction by AdaBoost. The solid line shows the score function The gray shaded region indicates the 95% confidence region. The dotted horizontal line shows the zero level. The rug plot in each panel shows the distribution of the data with respect to the range of each feature.

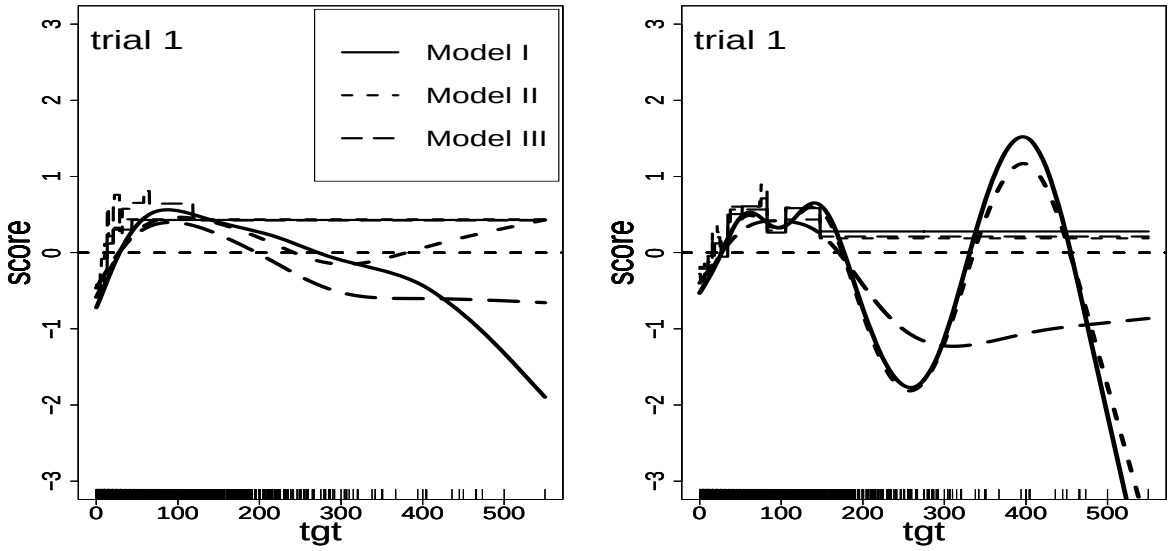
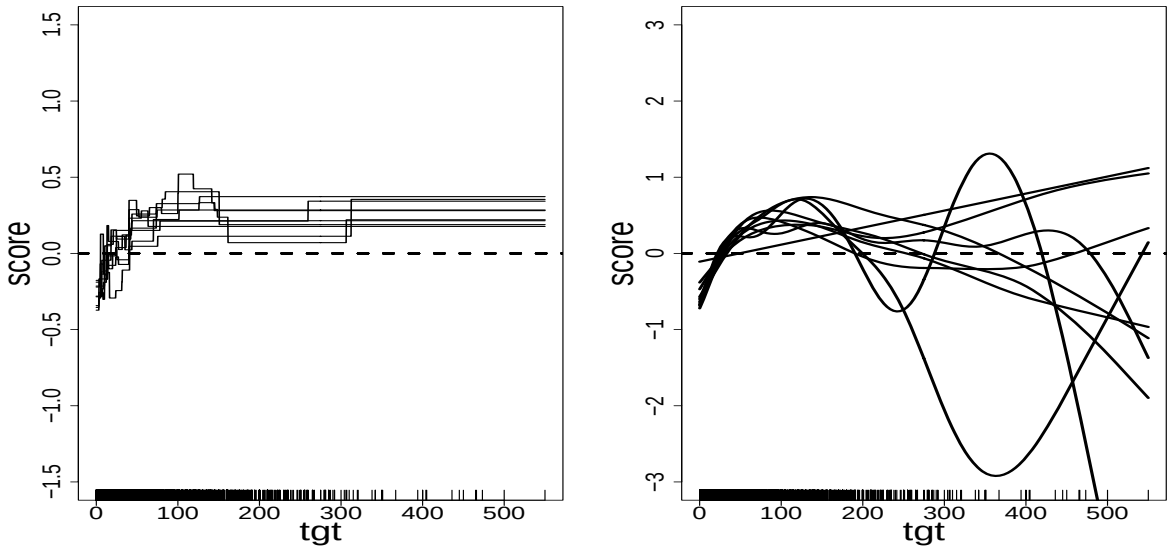


Figure 13: Score plots of 'amount of target species of tunas' (tgt) for AdaBoost and logistic GAM predictions from models I, II and III in two separate trials. The rug plot shows the distribution of the data with respect to 'amount of target species of tunas' (tgt). Note that the value of the score of AdaBoost in this figure is doubled for the comparison to the logistic GAM.



(a) (b)

Figure 14: Score plots of 'amount of target species of tunas' (tgt) for AdaBoost (a) and logistic GAM (b) predictions of bycatch from model I for nine trials. The rug plot shows the distribution of the data with respect to the range of the feature.

3.5 Control of the balance between the false positive and negative ratios

Highly skewed proportions of positive and negative samples in the data can strongly influence the FPR and FNR. The relationship between the probability of misclassification and FPR and FNR is given by the following equation:

$$\text{probability of misclassification } L(F) = P(Y = 1) \cdot \text{FNR} + P(Y = -1) \cdot \text{FPR} \qquad (37)$$

When n_n dominates in the data, $P(Y = 1) \ll P(Y = -1)$. In this case, the probability of misclassification is minimized when FNR $\gg$ FPR. Because of the preponderance of sets with no shark bycatch in our data set (70% of sets had no shark bycatch), the imbalance between the FPR and the FNR obtained from AdaBoost (Table 4) and the logistic GAM (Table 5) is expected. This imbalance is, however, not desirable if the goal is to accurately predict the occurrence of the shark bycatch. On average, 61% of the time the AdaBoost model I predicted no bycatch given that bycatch had in fact occurred (Table 4).

Table 6: AsymBoost test error rates and their standard deviations for each of the three models.

	Test Error(%)	False Positive Ratio(%)	False Negative Ratio(%)
model I	29.53 ± 0.77	27.67 ± 1.48	33.90 ± 2.02
model II	30.40 ± 1.20	28.82 ± 2.07	34.15 ± 2.34
model III	31.71 ± 1.37	31.10 ± 2.68	33.15 ± 3.29

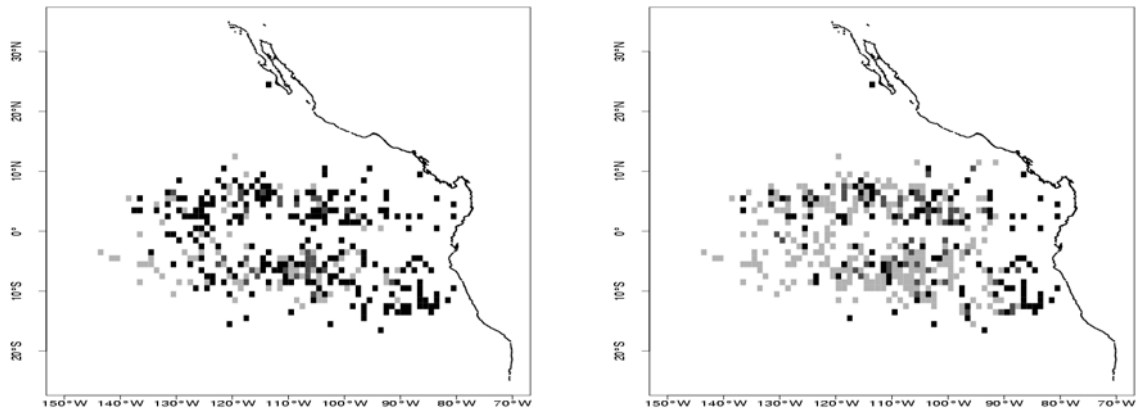


Figure 15: The proportion of the false negative predictions is shown by one degree square area. The left panel shows results from AdaBoost, the right panel shows results from AsymBoost. Light gray indicates values of 0, dark gray indicates values between 0 and 0.75, and black indicates values of greater than 0.75.

Application of AsymBoost to the shark bycatch data decreased the FNR considerably as compared to that of AdaBoost with only a slight increase in the overall test error. Application of AsymBoost enables us to control the trade-off between the FPR and the FNR whereby the FNR can be decreased at the expense of the FPR or *vice versa*. The ideal trade-off depends on the purpose of the analysis. As a specific example, application of AsymBoost to the shark bycatch data with its parameter k determined as per Eq. (29) shows that the FNR of AsymBoost decreased to about half that of the FNR of AdaBoost, with only a 3% increase in the test error (Table 6). Inspection of Figure 15 shows that the proportion of false negative predictions by AsymBoost remains large in the eastern and north-eastern region of the fishery, while it decreased in the south-western region. This implies that the discriminant function of AdaBoost often took on small absolute values for positive samples in this south-western region.

3.6 Discussion

We have demonstrated the use of a new predictive technique AdaBoost on the shark bycatch data. First, when decision stumps are used as base classifiers, AdaBoost can still yield stable predictions, even with many correlated features. This is not necessarily true for classical discriminant techniques, such as logistic regression (Ryan, 1997, Section 9.9). Unlike most classical techniques, AdaBoost exploits all features together and can

achieve more accurate predictions. Second, AdaBoost with sufficiently base classifiers is also relatively resistant to over-fitting. This means that the test error increases slowly for t greater than the optimal iteration number (T).Thus, the prediction accuracy is not overly sensitive to the stopping rule. Finally, with base classifiers based on only one feature, the contribution of each feature to the discriminant function $F(x)$ can be explored graphically using score plots, an option not available for some other algorithmic techniques such as neural networks.

Our analyses show that, compared to the logistic GAM, AdaBoost with decision stumps gives stable predictions even with problematic data. We have illustrated that AdaBoost performs well with correlated features, whereas the test error of the logistic GAM was sometimes large. Generally, the use of logistic GAM requires extra effort to remove the high correlation between features in the situation where many features are available and their predictive utility is unknown. AdaBoost with decision stumps will yield more stable predictions in such a situation. It was also demonstrated that, compared with logistic GAM, score functions of AdaBoost were relatively consistent both with respect to variability within the training data set and/or with increasing numbers of features. It should be mentioned that these shortcomings of the logistic GAM stem from the fact that the logistic GAM was not developed specifically for discrimination. In fisheries science, GAMs have been mostly employed for standardization of indices of relative abundance, and for modeling the effects of various features (*e.g.*, temperature) on such indices. In these cases, GAMs have been proven beneficial. It is still unknown whether boosting methods will perform well in such applications. Comparison of both methods for standardization of indices of relative abundance remains a challenge of future work.

The use of more complicated base classifiers with AdaBoost may improve some aspects of the prediction performance. There exist decision boundaries which cannot be approximated with linear sums of decision stumps. A simple example of this is when the true decision boundary, (*i.e.*, the true log-odds) is defined by products of the features:

$$\ln \frac{P(Y=1 \mid x)}{P(Y=-1 \mid x)} = \Pi_{m=1}^{M}(x)_m.$$

In such cases, other types of classifiers are required. However, care is needed because the properties of boosting depend strongly on the choice of the classifiers. AdaBoost with

classifiers other than decision stumps may not have all the favorable properties illustrated in this thesis. Thus, the type of classifiers should be determined after due consideration of the properties of the data.

Our results suggest that post-stratification of data such as the shark bycatch data may improve the prediction performance of AdaBoost. As indicated by the results of AsymBoost, the prediction performance of AdaBoost is not spatially uniform (Figure 15). We found that discriminant function of AdaBoost often takes on small absolute values in the south-western region of the fishery (see Section 3.5). This indicates that predicting shark bycatch in this area is more difficult than in other areas. Thus, there is the possibility of improving the prediction performance of AdaBoost by stratifying the data spatially and then applying AdaBoost separately to each stratum. Note that however division of the data set may lead to increased overfitting because it decreases the number of samples in each data set. Some improvements to the stopping rule may be needed to mitigate overfitting.

Our results also suggest that bycatch prediction with such the shark bycatch data may be improved by including information on the bycatch of nearby sets. Date (dat) and location (lat, lon) were by far the most informative features of the occurrence of shark bycatch for this data set (Figures 10-11). Similar results have been obtained from other studies using were conventional techniques (*e.g.*, Bigelow et al., 1999; Walsh and Kleiber, 2001). These features undoubtedly serve as proxies for characteristics of the local environment and community structure that were not adequately captured by sea surface temperature (tmp) and the various measures of the size of the community at the floating object (sml, pry, tgt, ntgt, and ntn). Although no theory currently exists for including values of the label y in nearby sets at earlier time periods as features in AdaBoost, our results suggest that this would be worthy of further study. Results of such analyses might lead to both improved prediction of bycatch and a better understanding of the important spatial and temporal scales for bycatch.

4 Local boosting method

We propose a new boosting method for improving the approximation error. As was illustrated in the previous section, AdaBoost with decision stumps performs well in several situations. However, much complicated decision boundary cannot be approximated by

classifiers other than decision stumps may not have all the favorable properties illustrated in this thesis. Thus, the type of classifiers should be determined after due consideration of the properties of the data.

Our results suggest that post-stratification of data such as the shark bycatch data may improve the prediction performance of AdaBoost. As indicated by the results of AsymBoost, the prediction performance of AdaBoost is not spatially uniform (Figure 15). We found that discriminant function of AdaBoost often takes on small absolute values in the south-western region of the fishery (see Section 3.5). This indicates that predicting shark bycatch in this area is more difficult than in other areas. Thus, there is the possibility of improving the prediction performance of AdaBoost by stratifying the data spatially and then applying AdaBoost separately to each stratum. Note that however division of the data set may lead to increased overfitting because it decreases the number of samples in each data set. Some improvements to the stopping rule may be needed to mitigate overfitting.

Our results also suggest that bycatch prediction with such the shark bycatch data may be improved by including information on the bycatch of nearby sets. Date (dat) and location (lat, lon) were by far the most informative features of the occurrence of shark bycatch for this data set (Figures 10-11). Similar results have been obtained from other studies using were conventional techniques (*e.g.*, Bigelow et al., 1999; Walsh and Kleiber, 2001). These features undoubtedly serve as proxies for characteristics of the local environment and community structure that were not adequately captured by sea surface temperature (tmp) and the various measures of the size of the community at the floating object (sml, pry, tgt, ntgt, and ntn). Although no theory currently exists for including values of the label y in nearby sets at earlier time periods as features in AdaBoost, our results suggest that this would be worthy of further study. Results of such analyses might lead to both improved prediction of bycatch and a better understanding of the important spatial and temporal scales for bycatch.

4 Local boosting method

We propose a new boosting method for improving the approximation error. As was illustrated in the previous section, AdaBoost with decision stumps performs well in several situations. However, much complicated decision boundary cannot be approximated by

the boosting method when base classifiers are too simple. In fact, it is easy to find many examples in which the boosting method with decision stumps performs poorly.

A simple way to overcome this difficulty is to use more complicated base classifiers. For example, decision stumps on a product of some pair of feature, decision trees, and so on. It is, however, not desirable in general since the upperbound on the generalization error increases as the complexity of base classifier class increases (See Section 2.5.4). In addition, the use of complicated base classifiers may make it impossible to interpret discriminant functions. For examples, we may not obtain score plots (See Section 3.1) when base classifiers based on more than one feature are used.

These observations motivated us to develop a localized version of boosting method, which is referred to as the *local boosting*. The local boosting is derived based on an idea similar to but not same as the local likelihood. A direct application of the local likelihood approach to boosting method would increase the computational cost of boosting method significantly in high-dimensional case, which often makes the implementation of the algorithm infeasible. The local boosting, however, does not require much increase of computational cost. The localization improves the approximation error of the ordinary boosting method in complicated situations even when decision stumps are used. We give the proof of the Bayes risk consistency of local boosting. Inspection of the proof elucidates that the local boosting improves the approximation error at cost of slight increase of estimation error, compared to the ordinary boosting. Simulation studies illustrate the theoretical results and the advantageous aspects of the local boosting.

4.1 Derivation of the local boosting algorithm

We now derive the algorithm of the local boosting for binary and multiclass classification. By borrowing an idea from the local likelihood methods, the local boosting algorithm is derived by localizing the base classifier. For this purpose, we introduce kernel functions $K_h : R^M \times R^M \to R_+$ with the form $K_h(x, y) = k(\|x-y\|/h)$ such that $\max_{x\in\mathcal{X}} K(x, x') \leq 1$ and $\int_{\mathcal{X}} K_h(x, x')dx < \infty$ for any $x' \in \mathcal{X}$, where $h > 0$ is a bandwidth, R_+ denotes the set of nonnegative real numbers, $\|\cdot\|$ is an arbitrary norm on $\mathcal{X}$ and $k : R_+ \to R_+$ is a function such that $K(0) = 1$ and $\lim_{z\to\pm\infty} k(z) = 0$. A few examples of function k are

listed below.

$$\begin{aligned} k_{\mathrm{rec}}(z) &= I(|z| \leq 1) \text{ (rectangular kernel)} \\ k_{\mathrm{gau}}(z) &= \exp(-z^2) \text{ (gaussian kernel)} \\ k_{\mathrm{tri}}(z) &= I(|z| \leq 1)(1-|z|^3)^3 \text{ (tricube kernel)} \end{aligned}$$

Let $\mathcal{K}$ be a set of kernel centers:

$$\mathcal{K} = \{x_\ell \in \mathcal{X} \,|\, \ell = 1, 2, \cdots, N\}.$$

The set $\mathcal{K}$ should cover the whole range of $\mathcal{X}$ as densely as possible. However, covering $\mathcal{X}$ in a high-dimensional case is difficult. A reasonable choice of $\mathcal{K}$ will be discussed in Sections 4.2.1 and 4.2.2. We here derive a local version of regularized boosting though our localization idea may also apply to the ordinary boosting. Let us first consider a binary classification. Recall that the usual regularized boosting algorithm is derived from the optimizations in Eq. (20) and (21). The local boosting algorithm is obtained by replacing the base classifier, f, in these optimizations by $K_h(\cdot, x')f$ as follows.

$$f = \operatorname*{argmin}_{f' \in \mathcal{C}} A_D^\lambda(F_{t-1} + \alpha'(K_h(\cdot, x')f' - F_{t-1})), \quad \text{for any positive } \alpha' \tag{38}$$

$$\alpha = \operatorname*{argmin}_{\alpha' > 0} A_D^\lambda(F_{t-1} + \alpha'(K_h(\cdot, x')f - F_{t-1})). \tag{39}$$

where x' is selected randomly from $\mathcal{K}$. Then, the discriminant function is updated as $F_t(x) = (1-\alpha)F_{t-1}(x) + \alpha K_h(x, x')f(x)$. We may assume the positiveness of α because of the assumption that $\mathcal{C}$ is negation closed. We calculate f approximately by using Taylor expansion with respect to α' as follows.

$$\begin{aligned} f &= \operatorname*{argmin}_{f' \in \mathcal{C}} A_D^\lambda(F_{t-1} + \alpha'(K_h(\cdot, x')f' - F_{t-1})) \\ &\approx \operatorname*{argmin}_{f' \in \mathcal{C}} A_D^\lambda(F_{t-1}) + \left.\frac{\partial A_D^\lambda(F_{t-1} + \alpha'(K_h(\cdot, x')f' - F_{t-1}))}{\partial \alpha'}\right|_{\alpha'=0} \alpha' \\ &= \operatorname*{argmin}_{f' \in \mathcal{C}} \sum_{i=1}^n \phi'(-\lambda Y_i F_{t-1}(X_i))(-Y_i)(K_h(X_i, x')f'(X_i) - F_{t-1}(X_i))\lambda\alpha' \\ &= \operatorname*{argmin}_{f' \in \mathcal{C}} \sum_{i=1}^n K_h(X_i, x')\phi'(-\lambda Y_i F_{t-1}(X_i))(-Y_i f'(X_i))\lambda\alpha' \\ &= \operatorname*{argmin}_{f' \in \mathcal{C}} \sum_{i=1}^n K_h(X_i, x')\phi'(-\lambda Y_i F_{t-1}(X_i))I(Y_i \neq f'(X_i)). \end{aligned} \tag{40}$$

As a result, the optimization in Eq. (38) does not depend on α'. It is worth noting the difference compared to the direct application of the local likelihood approach to boosting. The conventional local likelihood approach would suggest that Eq. (11) would be localized as

$$f = \underset{f' \in \mathcal{C}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} K_h(X_i, x') \phi(-Y_i(F_{t-1}(X_i) + \alpha'(f'(X_i) - F_{t-1}(X_i))))$$

and similarly for α. This approach would require a high computational cost because the boosting algorithm is repeated N times for each kernel center $x' \in \mathcal{K}$. In addition, this approach does not construct a single discriminant function but different discriminant functions for all $x' \in \mathcal{K}$. The proposed localization overcomes these difficulties with minor changes made to the usual boosting algorithm. The complete summary of the algorithm of the local boosting is described below.

---

1. Prepare $\mathcal{K}$ and determine a bandwidth h and a smoothing parameter λ.

2. Initialize weights on sample $\{d_0(i)\}_{i=1}^n$ as $1/n$ for each i and $F_0 \equiv 0$.

3. For $t = 1, 2, \cdots, T$, repeat the following process.

   (a) Choose a new kernel center $x_{\ell(t)}$ from $\mathcal{K}$ such that $\ell(t)$ is randomly chosen from $\{1, 2, \cdots, N\}$. Set

   $$w_t(i) = \frac{1}{Z_t} d_{t-1}(i) K_h(X_i, x_{\ell(t)}) \tag{41}$$

   for each i where Z_t is a normalization constant such that $\sum_{i=1}^n w_t(i) = 1$.

   (b) Find a new locally best classifier around $x_{\ell(t)}$ and its coefficient, denoted as $f_{j(t)}$ and α_t, as follows:

   $$\begin{aligned} j(t) &= \underset{j \in \{1,2,\cdots,J\}}{\operatorname{argmin}} \epsilon_t(f_j) \\ \alpha_t &= \underset{\alpha \in R_+}{\operatorname{argmin}} A_D^{\lambda}(F_{t-1} + \alpha(K_h(\cdot, x_{\ell(t)}) f_{j(t)} - F_{t-1})\} \quad (t \geq 2) \end{aligned} \tag{42}$$

   with $\alpha_1 = 1$ where the localized weighted error rate, $\epsilon_t(f)$, is defined as

   $$\epsilon_t(f) = \sum_{i=1}^{n} w_t(i) I(Y_i \neq f(X_i)).$$

(c) Update $\{d_{t-1}(i)\}_{i=1}^n$ and the discrimination function, F_{t-1}, as follows.

$$F_t(x) = (1-\alpha_t)F_{t-1}(x) + \alpha_t K_h(x, x_{\ell(t)}) f_{j(t)}(x) \quad (43)$$

$$d_t(i) = \phi'(-\lambda Y_i F_t(X_i)) \quad (44)$$

4. Finally, we obtain a resultant classifier $g(x) = \mathrm{sign}(F_T(x))$.

It holds that $\sum_{t=1}^T \alpha_t = 1$. This implies that $F_T(x)$ necessarily takes its value in the closed interval $[-1, 1]$.

In the local boosting algorithm, $f_{j(t)}$ is a locally best classifier around $x_{\ell(t)}$ in the following sense. To see this, we may interpret the local boosting algorithm from the view of resampling, similarly to that of the usual boosting (Section 2.5.1). From the fact that the sum of all $\{w_t(i)\}_{i=1}^n$ is always restricted to one, we can regard $w_t(i)$ as the probability assigned to the i-th sample (X_i, Y_i). Suppose that a pair of new random variables (X_t', Y_t') is resampled from D according to the probabilities $\{w_t(i)\}_{i=1}^n$. Inspection of Eq. (41) and (44) indicates that each weight on sample $w_t(i)$ is proportional to $K_h(X_i, x_{\ell(t)})\phi'(-\lambda Y_i F_{t-1}(X_i))$. Thus, (X_t', Y_t') takes its values on either samples $\{(X_i, Y_i)\}$ in D with a higher probability if X_i is nearer to $x_{\ell(t)}$ and/or if a larger $F_{t-1}(X_i)$ predicts a label of X_i incorrectly, *i.e.*, the larger that $-Y_i F(X_i)$ is. The locally weighted error rate $\epsilon_t(f)$ can be regarded as the probability of misclassification over (X_t', Y_t') given by f. Intuitively, Eq. (42) corresponds to the search of f that most improves the prediction performance of F_{t-1} locally around $x_{\ell(t)}$. Finally, $f_{j(t)}$ is added to F_{t-1} as in Eq. (43) after being localized in the same way as that in Eq. (41).

Bandwidth h controls the extent of localization of boosting. Infinite bandwidth ($h \to \infty$) introduces no localization and reduces the local boosting to the usual boosting because $K_h(\cdot, \cdot)$ always takes the value one, and then, a base classifier is chosen in Eq. (42) based on all the training data. When a discriminant function consisting of only a single convex combination of base classifiers performs poorly over all the training data, h should be small. If h is smaller, the base classifier must works in smaller area and may perform better. The final discriminant function is constructed by combining such locally accurate classifiers and then may perform well. The selection of h will be discussed in Sections 4.2.1 and 4.2.2.

We should mention that our idea about localization can directly apply to the multiclass case. It is easy to see that regularized boosting for multiclass classification is localized in

the same way. Therefore, we omit the redundant derivation and show just its algorithm below.

---

1. Prepare $\mathcal{K}$ and determine a bandwidth, h, and a smoothing parameter, λ.

2. Initialize weights on each sample $\{d_0(i,y)\}_{i=1}^n$ as $\frac{I(y\neq Y_i)}{n(|\mathcal{Y}|-1)}$ for each i and $F_0 \equiv 0$.

3. For $t = 1, 2, \cdots, T$, repeat the following process.

   (a) Choose a new kernel center $x_{\ell(t)}$ from $\mathcal{K}$ such that $\ell(t)$ is randomly chosen from $\{1, 2, \cdots, N\}$. Set

   $$w_t(i,y) = \frac{1}{Z_t} d_t(i,y) K(X_i, x_{\ell(t)})$$

   for each i where Z_t is a normalization constant such that $\sum_{i=1}^n \sum_{y\in\mathcal{Y}} w_t(i,y) I(y \neq Y_i) = 1$.

   (b) Find a new locally best classifier around $x_{\ell(t)}$ and its coefficient, denoted as $f_{j(t)}$ and α_t, as follows:

   $$\begin{aligned} j(t) &= \operatorname*{argmin}_{j\in\{1,2,\cdots,J\}} \epsilon_t(f_j) \\ \alpha_t &= \begin{cases} \operatorname{argmin}_{\alpha\in R_+} A_D^\lambda(F_{t-1} + \alpha(K_h(\cdot, x_{\ell(t)}) f_{j(t)} - F_{t-1})\} & (t>1) \\ 1 & (t=1) \end{cases} \end{aligned}$$

   where the localized weighted error rate, $\epsilon_t(f)$, is defined as

   $$\epsilon_t(f) = (1/2) \sum_{i=1}^n \sum_{y\in\mathcal{Y}} w_t(i,y)(f(X_i,y) - f(X_i,Y_i) + 1) I(y \neq Y_i)$$

   (c) Update $\{d_{t-1}(i,y)\}_{i=1}^n$ and the discrimination function, F_{t-1}, as follows.

   $$\begin{aligned} F_t(x,y) &= (1-\alpha_t) F_{t-1}(x,y) + \alpha_t K_h(x, x_{\ell(t)}) f_{j(t)}(x,y) \\ d_t(i,y) &= \phi'(-\lambda Y_i (F_t(X_i,y) - F_t(X_i,Y_i))) \end{aligned}$$

4. Finally, we obtain a classifier $F(x) = \operatorname{argmax}_{y\in\mathcal{Y}} F_T(x,y)$.

4.2 Statistical properties of local boosting

We discuss several statistical properties of the local boosting, compared to the usual boosting. First, we derive statistical models associated with both boosting methods. Then, we prove the Bayes risk consistency of the local boosting. Inspection of the proof provides a useful viewpoint for understanding how the local boosting improves the prediction performance of the usual boosting.

4.2.1 Model associated with local boosting

The ordinary boosting and the local boosting are connected to different models respectively made from the same class of base classifiers.

For comparison, we first show the model associated with usual boosting. Let us consider the global model, $\mathcal{M}$, defined as

$$\mathcal{M} = \{F(x) = \sum_{j=1}^{J} \theta_j f_j(x) \mid \sum_{j=1}^{J} \theta_j = 1, \forall j,\ f_j \in \mathcal{C},\ \theta_j \geq 0\}.$$

The global model $\mathcal{M}$ is the set of discriminant functions consisting of any convex combination of $\mathcal{C}$. The usual boosting searches for the minimizer of A_D^λ from $\mathcal{M}$ as seen below. The update rule, Eq. (22), indicates that a discriminant function constructed by the usual boosting has the form:

$$F_T(x) = \sum_{t=1}^{T} \alpha_t f_{j(t)}(x),$$

such that $\sum_{t=1}^{T} \alpha_t = 1$. Let $\Gamma_j = \{t \mid f_{j(t)} = f_j\}$. Defining $\theta_j = \sum_{t \in \Gamma_j} \alpha_t$, $F_T(x)$ can be rewritten as

$$F_T(x) = \sum_{j=1}^{J} \theta_j f_j(x).$$

Thus, any discriminant function constructed by usual boosting can be regarded as an element of the global model, $\mathcal{M}$.

The local boosting is connected to a local model, $\mathcal{M}_\mathcal{K}$, defined as

$$\begin{aligned} \mathcal{M}_\mathcal{K} \;=\; & \{F(x) = \sum_{j=1}^{J} \bar{\theta}_j(x) f_j(x) \mid \bar{\theta}_j(x) = \frac{1}{N} \sum_{\ell=1}^{N} K_h(x, x_\ell)\theta_{j\ell}\,, \\ & \sum_{j=1}^{J} \sum_{\ell=1}^{N} \theta_{j\ell} = N\,,\ \forall j,\ f_j \in \mathcal{C}, \theta_{j\ell} \geq 0\}. \end{aligned} \tag{45}$$

Any discriminant function constructed by the local boosting is an element of $\mathcal{M}_{\mathcal{K}}$ as follows. The update rule, Eq. (43), indicates that the local boosting constructs a discriminant function with the form

$$F_T(x) = \sum_{t=1}^{T} K_h(x, x_{\ell(t)}) \alpha_t f_{j(t)}(x) \tag{46}$$

such that $\ell(t)$ is randomly chosen from $\{1, 2, \cdots, N\}$ and $\sum_{t=1}^{T} \alpha_t = 1$. Let $\Gamma_{j,\ell} = \{t \,|\, f_{j(t)} = f_j \,,\, \ell(t) = \ell\}$. Taking $\theta_{j\ell} = \sum_{t \in \Gamma_{j,\ell}} N\alpha_t$, the discriminant function, Eq. (46), can be rewritten as

$$F_T(x) = \frac{1}{N} \sum_{\ell=1}^{N} \sum_{j=1}^{J} \theta_{j\ell} K_h(x, x_\ell) f_j(x).$$

Clearly, the discriminant function, $F_T(x)$, is in $\mathcal{M}_{\mathcal{K}}$. A discriminant function in $\mathcal{M}_{\mathcal{K}}$ consists of different convex combinations depending on the location in $\mathcal{X}$ while a discriminant function in $\mathcal{M}$ consists of a single convex combination of base classifiers. Therefore, $\mathcal{M}_{\mathcal{K}}$ can be more greedy than $\mathcal{M}$. This richness of $\mathcal{M}_{\mathcal{K}}$ causes the local boosting to be Bayes risk consistent in a variety of situations than that of the usual boosting.

The selection of $\mathcal{K}$ and a bandwidth h plays an important role in the local boosting algorithm. A $(\mathcal{K}, h)$ pair is referred to as a *localizing factor* in the remainder of this thesis. A localizing factor controls the trade-off between the approximation ability of $\mathcal{M}_{\mathcal{K}}$ and an overfitting. We discuss this issue roughly here and will discuss it again more theoretically from the view of the Bayes risk consistency in Section 4.2.2. $\mathcal{M}_{\mathcal{K}}$ with $h \to \infty$ reduces to $\mathcal{M}$ regardless of $\mathcal{K}$. As described in the previous section, $h \to \infty$ reduces the algorithm of the local boosting to that of usual boosting. In fact, taking $h \to \infty$, $K_h(\cdot, x_\ell) \equiv 1$ for any $x_\ell \in \mathcal{K}$, and then, $\bar{\theta}_j(x)$ does not depend on x because

$$\lim_{h \to \infty} \bar{\theta}_j(x) \to \frac{1}{N} \sum_{\ell=1}^{N} \theta_{j\ell}.$$

Clearly, discriminant function $F(x) = \sum_{j=1}^{J} \bar{\theta}_j(x) f_j(x)$ is an element of $\mathcal{M}$ in this case. Thus, $\mathcal{M}_{\mathcal{K}}$ reduces to $\mathcal{M}$ when $h \to \infty$. Decreasing h enhances the approximation ability of $\mathcal{M}_{\mathcal{K}}$ but also suffers from overfitting to the training data and requires a more dense $\mathcal{K}$. To see that, we introduce some notation. Let $B_\epsilon(\ell, h)$ be the region where the value of a kernel with center $x_\ell \in \mathcal{K}$ takes a value larger than ϵ, *i.e.*,

$$B_\epsilon(\ell, h) = \{x \in \mathcal{X} \,|\, K_h(x, x_\ell) > \epsilon\}.$$

We also denote the union of $\{B_\epsilon(\ell, h)\}_{\ell=1}^n$ by $\mathcal{B}_\epsilon(\mathcal{K}, h)$ and denote the region $\{x \in \mathcal{X} \mid x \notin \mathcal{B}_\epsilon(\mathcal{K}, h)\}$ by $\overline{\mathcal{B}}_\epsilon(\mathcal{K}, h)$. For a given kernel center, x_ℓ, a discriminant function, F_{t-1}, is updated in the local boosting algorithm such that only training data in $B_\epsilon(\ell, h)$ with $0 < \epsilon \ll 1$ are more accurately classified. Roughly, a discriminant function constructed by the local boosting consists of local discriminant functions that are constructed by the usual boosting based on training data in $B_\epsilon(\ell, h)$. Therefore, $\mathcal{B}_\epsilon(\mathcal{K}, h)$ should cover the whole region of $\mathcal{X}$. When h is large, $\mathcal{B}_\epsilon(\mathcal{K}, h)$ can cover $\mathcal{X}$ even with small $\mathcal{K}$. In particular, when $h \to \infty$, $\mathcal{B}_\epsilon(\mathcal{K}, h)$ covers $\mathcal{X}$ even with a single point set $\mathcal{K}$ and then F_T consists of only a single convex combination classifier. In contrast, small h and dense $\mathcal{K}$ partition $\mathcal{X}$ into small areas finely and then yield a more complicated discriminant function. However, an h that is too small requires a significantly large $\mathcal{K}$ because $\mathcal{B}_\epsilon(\mathcal{K}, h)$ should cover $\mathcal{X}$, which is often infeasible in a high-dimensional case. Let us focus on a specific $B_\epsilon(\ell, h)$. If h is smaller, $B_\epsilon(\ell, h)$ is narrower and there will be fewer training data in $B_\epsilon(\ell, h)$. In particular, when h approaches zero, several problems may occur. When a kernel function with compact support is used, special care is needed. If a kernel center x' such that $x' \neq X_i$ for any i is selected, then the local boosting algorithm with too small h comes to a halt because $w_t(i) = 0$ for all i in Eq. (41). Even when we use a kernel function with infinite support, $B_\epsilon(\ell, h)$ includes only the training data on x_ℓ, *i.e.*, at most one datum in general. In this case, the weights $\{w_t(i)\}$ are zero except on indices in which X_i is the point nearest to x_ℓ. (Note that this holds exactly for any kernel function with infinite support satisfying $\lim_{n\to\infty} k(x_n)/k(y_n) \to 0$ for any $\{x_n, y_n\}_{n=1}^\infty$ such that $x_n, y_n \to \infty$ and $y_n - x_n \to \infty$ as $n \to \infty$.) In this case, the local boosting is significantly overfit to the training data. When the training data nearest to x_ℓ are mislabels (defined in Section 2.4.3), local boosting may perform poorly over the region around x_ℓ. In addition, if there are training data in $\overline{\mathcal{B}}_\epsilon(\mathcal{K}, h)$, they have almost no effect through the construction of a discriminant function. A localizing factor should be selected by considering the trade-off between approximation ability and overfitting.

A practical choice of $\mathcal{K}$ is the training data itself. In actual situations, samples are often generated in a specific region of $\mathcal{X}$. It is not necessary to cover the whole region of $\mathcal{X}$ but only the support of X:

$$\mathrm{supp}(X) = \{x \in \mathcal{X} \mid P(x) > 0\},$$

where $P(x)$ denotes the underlying marginal distribution of X. One simple and practical

choice of $\mathcal{K}$ based on this idea is a set of the points in the given training data, $D = \{(X_i, Y_i)\}_{i=1}^n$, *i.e.*,

$$\mathcal{K}_* = \{x_\ell = X_\ell \,|\, \ell = 1, 2, \cdots, n\}. \tag{47}$$

$\mathcal{B}_\epsilon(\mathcal{K}_*, h)$ covers a region where data are likely to be generated and asymptotically cover $\mathrm{supp}(X)$. In addition, there are no training data in $\overline{\mathcal{B}}_\epsilon(\mathcal{K}_*, h)$. Thus, we used this $\mathcal{K}_*$ in the simulation studies in Section 4.3. Note that the local boosting with small h still tends to overfit to training data even when $\mathcal{K}_*$ is used. As $n \to \infty$, the coefficient $\bar{\theta}_j(x)$ in the model $\mathcal{M}_\mathcal{K}$ with $\mathcal{K}_*$ converges to $E[K_h(x, X)\theta_j(X)]$ due to the law of large numbers (Recall that $\theta_{j\ell} = \theta_j(x_\ell)$). Thus, the local boosting with $\mathcal{K}_*$ is asymptotically associated with the asymptotical model, $\overline{\mathcal{M}}_\mathcal{K}$, defined as

$$\begin{aligned}\overline{\mathcal{M}}_\mathcal{K} \;=\; & \{F(x) = \sum_{j=1}^J \bar{\theta}_j(x) \,|\, \bar{\theta}_j(x) = E[K_h(x, X)\theta_j(X)] f_j(x), \\ & \sum_{j=1}^J E\theta_j(X) = 1 \,,\, \forall j \,,\, f_j \in \mathcal{C} \,,\, \theta_j(x) \geq 0\}.\end{aligned}$$

Note that $\theta_j(x)$ is defined on $\mathrm{supp}(X)$ because any element of $\mathcal{K}_*$ is necessarily in $\mathrm{supp}(X)$. We may use other types of $\mathcal{K}$ if it satisfies the condition discussed in Section 4.2.2. For example, we may use the points generated from a uniform distribution over the support of X for decreasing duplicates of kernel centers if $\mathrm{supp}(X)$ is known in advance. In addition, it is not necessary to increase the number of points in $\mathcal{K}$ as $n \to \infty$.

4.2.2 Bayes Risk Consistency

The risk of the local boosting converges to Bayes risk L^* almost surely in a variety of situations, compared to that of the usual boosting. We confine ourselves to binary classification in this section. The convergence of the risk to L^* is referred to as the *Bayes risk consistency* (Lugosi and Vayatis, 2004). The Bayes risk consistency of the local boosting is shown in Theorem 32. The local boosting iteratively minimizes A_D^λ as the iteration number T increases infinitely. Therefore, we survey properties of an estimator given by the local boosting, $\hat{F}$, defined as

$$\hat{F} = \operatorname*{argmin}_{F' \in \mathcal{M}_\mathcal{K}} A_D^\lambda(F'),$$

where $\mathcal{M}_\mathcal{K}$ is defined in Eq. (45). Note that we often design to find an appropriate T to avoid overfitting in practical uses. We use the notion of Rademacher complexity in the

proof (*e.g.*, Bartlett and Mendelson, 2002; Bartlett et al., 2003). For a given set, $\mathcal{F}$, of mappings from $\mathcal{X}$ to R, Rademacher complexity of $\mathcal{F}$ is defined as

$$R_n(\mathcal{F}) = \frac{2}{n} E \sup_{F \in \mathcal{F}} |\sigma_i F(X_i)|,$$

where σ_i is a Rademacher variable for each i, *i.e.*, a random variable that is independent of all other random variables and that takes the values 1 or -1 with probability $1/2$. One can find various upperbounds of Rademacher complexity in many literatures (*e.g.*, Ledoux and Talagrand, 1991; van der Vaart and Wellner, 1996). Although several steps in the proof of the theorem are the same as those in (Koltchinskii and Panchenko, 2002; Lugosi and Vayatis, 2004), we describe all steps of the proof for completeness.

Theorem 32. *Let ϕ be a strictly convex and strictly increasing cost function such that $\phi(0) = 1$ and $\lim_{x \to -\infty} \phi(x) = 0$. For each n, let $\mathcal{K}$ be a set of fixed points in $\mathcal{X}$,*

$$\mathcal{K} = \{x_\ell \in \mathcal{X} \,|\, \ell = 1, 2, \cdots, N_n\},$$

such that N_n, the cardinality of $\mathcal{K}$, is less than or equal to n^β for a finite $\beta > 0$. Assume that the class of base classifier, $\mathcal{C}$, has a finite VC dimension, V, is negation closed, and that the distribution of (X, Y) satisfies

$$\lim_{\lambda \to \infty} \inf_{F \in \mathcal{M}_\mathcal{K}} A^\lambda(F) - A^* = 0, \tag{48}$$

where $A^* = \inf_{f': \mathcal{X} \to R} A(f')$. Let λ_n be a sequence of positive numbers such that

$$\lambda_n \to \infty, \ \lambda_n \phi'(\lambda_n) \sqrt{\frac{\ln n}{n}} \to 0$$

as $n \to \infty$ and define the estimator, $\hat{F}_n$, as

$$\hat{F}_n = \operatorname*{argmin}_{F' \in \mathcal{M}_\mathcal{K}} A_D^{\lambda_n}(F').$$

Then, $g_{\hat{F}_n} = \operatorname{sign}(\hat{F}_n)$ is strongly Bayes risk consistent, that is,

$$\lim_{n \to \infty} L(g_{\hat{F}_n}) = L^* \text{ almost surely.}$$

Before jumping to the proof of Theorem 32, we introduce some lemmas that were given by Lugosi and Vayatis (2004).

Lemma 33. *Let $h(\eta, \alpha) = \eta\phi(-\alpha) + (1 - \eta)\phi(\alpha)$ and $H(\eta) = \inf_{\alpha \in R} h(\eta, \alpha)$. $H(\eta)$ is a strictly concave function on the interval $(0, 1)$.*

Proof. Since $\eta \in (0,1)$, there exists uniquely α that minimizes $h(\eta,\alpha)$ for any fixed η from Proposition 23. Write α minimizing $h(\eta,\alpha)$ as $\alpha(\eta)$. It is easy to show that, for any $\eta_1 \neq \eta_2$, $\alpha(\eta_1) \neq \alpha(\eta_2)$. Therefore, for any fixed $\theta \in [0,1]$, we have

$$\begin{aligned}
H(\theta\eta_1 + (1-\theta)\eta_2) &= \inf_\alpha h(\theta\eta_1 + (1-\theta)\eta_2, \alpha) \\
&= \inf_\alpha h(\theta\eta_1 + (1-\theta)\eta_2, \alpha) \\
&= \inf_\alpha (\theta\eta_1 + (1-\theta)\eta_2)\phi(-\alpha) + (1 - \theta\eta_1 - (1-\theta)\eta_2)\phi(\alpha) \\
&= \inf_\alpha \theta(\eta_1\phi(-\alpha) + (1-\eta_1)\phi(\alpha)) + (1-\theta)(\eta_2\phi(-\alpha) + (1-\eta_2)\phi(\alpha)) \\
&= \inf_\alpha \theta h(\eta_1,\alpha) + (1-\theta)h(\eta_2,\alpha) \\
&> \theta \inf_\alpha h(\eta_1,\alpha) + (1-\theta)\inf_\alpha h(\eta_2,\alpha) \\
&= \theta H(\eta_1) + (1-\theta)H(\eta_2).
\end{aligned}$$

□

Lemma 34 (Lemma 4 of Lugosi and Vayatis, 2004). *Let $\phi : R \to R$ be a cost function satisfying Condition 18. Then the function*

$$H(\eta) = \inf_{\alpha \in R} \eta\phi(-\alpha) + (1-\eta)\phi(\alpha)$$

defined for $\eta \in [0,1]$, is strictly concave, symmetric around $1/2$, and $H(0) = H(1) = 0$, $H(1/2) = 1$.

Proof. Concavity follows from Lemma 33. Define $h(\eta,\alpha) = \eta\phi(-\alpha) + (1-\eta)\phi(\alpha)$. Then, $H(\eta) = \inf_{\alpha\in R} h(\eta,\alpha)$. Concavity of H follows from Lemma 33. For any $\eta' \in [0,1/2]$,

$$\begin{aligned}
H(1/2-\eta') &= \inf_{\alpha\in R} h(1/2-\eta',\alpha) \\
&= \inf_{\alpha\in R} (1/2-\eta')\phi(-\alpha) + (1/2+\eta')\phi(\alpha) \\
&= \inf_{\alpha\in R} (1/2+\eta')\phi(\alpha) + (1-(1/2+\eta'))\phi(-\alpha) \\
&= \inf_{\alpha\in R} h(1/2+\eta',-\alpha) \\
&= H(1/2+\eta').
\end{aligned}$$

Thus, H is symmetry around $1/2$. Clearly $H(0) = H(1) = 0$. Due to the strict convexity

of ϕ, we have,

$$\begin{aligned} H(1/2) &= \inf_{\alpha \in R} h(1/2, \alpha) \\ &= \inf_{\alpha \in R} \frac{1}{2}\{\phi(-\alpha) + \phi(\alpha)\} \\ &\leq \phi\left(\frac{1}{2}(-\alpha) + \frac{1}{2}(\alpha)\right) = \phi(0) = 1. \end{aligned}$$

□

Lemma 35 (Lemma 5 of Lugosi and Vayatis, 2004). *Let ϕ be a cost function satisfying Condition 18. Let $\{f_n : \mathcal{X} \to R\}$ be an arbitrary sequence such that*

$$\lim_{n \to \infty} A(f_n) - A^* = 0$$

where $A^* = \inf_{f \in \mathcal{F}} A(f)$. Then, $g_n(x) = \mathrm{sign}(f_n(x))$ has a probability of misclassification converging to L^*.

Proof. The difference of probability of misclassification can be rewritten as follows (Devroye et al., 1996).

$$\begin{aligned} L(g_n) - L^* &= E[I(g_n(X) \neq Y) - I(g^*(X) \neq Y)] \\ &= E[I(g_n(X) \neq g^*(X))\{I(g_n(X) \neq Y) - I(g^*(X) \neq Y)\}] \\ &= E[I(g_n(X) \neq g^*(X))\{\eta(X)(I(g_n(X) \neq 1) - I(g^*(X) \neq 1)) \\ &\quad +(1 - \eta(X))(I(g_n(X) \neq -1) - I(g^*(X) \neq -1))\}] \end{aligned}$$

For a fixed x such that $\eta(x) \geq 1/2$, the inside of the expectation in the last equality is calculated as

$$\begin{aligned} &I(g_n(x) \neq g^*(x))\{\eta(x)(I(g_n(x) \neq 1) - I(g^*(x) \neq 1)) \\ &\quad +(1 - \eta(x))(I(g_n(x) \neq -1) - I(g^*(x) \neq -1))\} &= I(g_n(x) \neq g^*(x))\{\eta(x) - (1 - \eta(x))\} \\ &&= I(g_n(x) \neq g^*(x))(2\eta(x) - 1). \end{aligned}$$

For a fixed x such that $\eta(x) < 1/2$,

$$\begin{aligned} I(g_n(x) \neq g^*(x))\{\eta(x)(I(g_n(x) \neq 1) - I(g^*(x) \neq 1)) & \\ +(1-\eta(x))(I(g_n(x) \neq -1) - I(g^*(x) \neq -1))\} &= I(g_n(x) \neq g^*(x))\{-\eta(x) + (1-\eta(x))\} \\ &= I(g_n(x) \neq g^*(x))(1 - 2\eta(x)) \end{aligned}$$

Thus,

$$L(g_n) - L^* = E[I(g_n(X) \neq g^*(X))|2\eta(X) - 1|].$$

Define $S_\delta = \{x \,|\, \eta(x) \in [1/2 - \delta, 1/2 + \delta]\}$. Clearly,

$$L(g_n) - L^* \leq 2\delta + P(X \notin S_\delta \text{ and } g_n(X) \neq g^*(X)).$$

Proceeding by contradiction, assume that $P(X \notin S_\delta \text{ and } g_n(X) \neq g^*(X))$ does not vanish as $n \to \infty$. Then, there necessarily exists a sequence of sets $K_n \subset \bar{S}_\delta$ such that $g_n(x) \neq g^*(x)$ on B_n and $\liminf_n P(X \in B_n) > 0$. Without loss of generality, we may assume that $g_n(x) = 1$ and $g^*(x) = -1$ on B_n due to the symmetry. Then, $f_n(x) \geq 0$ and $f^*(x) < 0$. Note that $f^*(x) < 0$ implies that $\eta(x) < 1/2 - \delta$.

Define $h(\eta, \alpha) = \eta\phi(-\alpha) + (1-\eta)\phi(\alpha)$. Then, the difference $A(f_n) - A^*$ can be written as

$$A(f_n) - A^* = E[h(\eta(X), f_n(X)) - \inf_{f' \in \mathcal{F}} h(\eta(X), f'(X))]. \tag{49}$$

Write, for any set $B \subset \mathcal{X}$,

$$A|_B(f) = E[I(X \in B)\{\eta(X)h(\eta(X), f(X))\}].$$

Since the inside of the expectation in Eq. (49) is a positive function, we then have, for any $B \subset \mathcal{X}$,

$$A(f_n) - A^* \geq A|_B(f_n) - A|_B(f^*).$$

Since $h(\eta(x), \cdot)$ is a strictly convex function taking its minimum at $f^*(x)$, we have that $h(\eta(x), f_n(x)) > h(\eta(x), 0) = \phi(0) = 1$. Then, we have

$$A|_{B_n}(f_n) = E[I(X \in B_n)h(\eta(X), f_n(X))] > E[I(X \in B_n)] = P(X \in B_n).$$

On the other hand, $H(\eta) = \inf_{\alpha \in R} h(\eta, \alpha)$ is strictly increasing on the interval $[0, 1/2]$ due to Lemma 34. Then, we have

$$A|_{B_n}(f^*) = E[I(X \in B_n)H(\eta(X))] \geq E[I(X \in B_n)H(1/2 - \delta)] = H(1/2 - \delta)P(X \in B_n).$$

Thus,

$$A(f_n) - A^* \geq A|_B(f_n) - A|_B(f^*) > (1 - H(1/2 - \delta))P(X \in B_n).$$

Because of the concavity of H (Lemma 34), we have $\liminf_{n\to\infty} A(f_n) - A^* > 0$, which is a contradiction. □

We also use Theorem 4.12 of Ledoux and Talagrand (1991). Before jumping to this theorem, we define *contraction*.

Definition 36 (Contraction). *Let ψ be a function mapping from R to R. We say that ψ is a contraction if $|\psi(s) - \psi(t) \leq |s - t|$ for any $s, t \in R$.*

Theorem 37 (Theorem 4.12 of Ledoux and Talagrand 1991). *Let $F : R_+ \to R_+$ be convex and increasing. Let further $\psi_i : R \to R$ $(i = 1, 2, \cdots, n)$ be contractions such that $\psi_i(0) = 0$. Then, for any bounded subset T in R^n,*

$$EF\left(\frac{1}{2}\sup_{t\in T}\left|\sum_{i=1}^n \sigma_i\psi_i(t_i)\right|\right) \leq EF\left(\sup_{t\in T}\left|\sum_{i=1}^n \sigma_i t_i\right|\right).$$

Proof. Define $(A)^+$ as $AI(A > 0)$ and $(A)^-$ as $A(I(A < 0))$. Then, we have

$$\begin{aligned}
\frac{1}{2}\sup_{t\in T}\left|\sum_{i=1}^n \sigma_i\psi_i(t_i)\right| &= \frac{1}{2}\sup_{t\in T}\left(\sum_{i=1}^n \sigma_i\psi_i(t_i)\right)^+ - \left(\sum_{i=1}^n \sigma_i\psi_i(t_i)\right)^- \\
&\quad (\text{since } |A| = (A)^+ - (-A)^-) \\
&\leq \frac{1}{2}\sup_{t\in T}\left(\sum_{i=1}^n \sigma_i\psi_i(t_i)\right)^+ - \frac{1}{2}\sup_{t\in T}\left(\sum_{i=1}^n \sigma_i\psi_i(t_i)\right)^-
\end{aligned}$$

Since F is convex and increasing, we have

$$\begin{aligned}
EF\left(\frac{1}{2}\sup_{t\in T}\left|\sum_{i=1}^n \sigma_i\psi_i(t_i)\right|\right) &\leq EF\left(\frac{1}{2}\sup_{t\in T}\left(\sum_{i=1}^n \sigma_i\psi_i(t_i)\right)^+ - \frac{1}{2}\sup_{t\in T}\left(\sum_{i=1}^n \sigma_i\psi_i(t_i)\right)^-\right) \\
&\leq \frac{1}{2}EF\left(\sup_{t\in T}\left(\sum_{i=1}^n \sigma_i\psi_i(t_i)\right)^+\right) + \frac{1}{2}EF\left(\sup_{t\in T} -\left(\sum_{i=1}^n \sigma_i\psi_i(t_i)\right)^-\right).
\end{aligned}$$

Using the fact that $(A)^- = -(-A)^+$ and that $-\sigma_i$ has the same distribution of as σ_i, we have

$$\begin{aligned}
EF\left(\sup_{t\in T} -\left(\sum_{i=1}^n \sigma_i\psi_i(t_i)\right)^-\right) &= EF\left(\sup_{t\in T}\left(\sum_{i=1}^n (-\sigma_i)\psi_i(t_i)\right)^+\right) \\
&= EF\left(\sup_{t\in T}\left(\sum_{i=1}^n \sigma_i\psi_i(t_i)\right)^+\right)
\end{aligned}$$

Thus,

$$EF\left(\frac{1}{2}\sup_{t\in T}\left|\sum_{i=1}^{n}\sigma_i\psi_i(t_i)\right|\right) \le EF\left(\sup_{t\in T}\left(\sum_{i=1}^{n}\sigma_i\psi_i(t_i)\right)^+\right)$$

The statement follows from the following lemma with $G = F((\cdot)^+)$.

Lemma 38. *If $G : R \to R$ be convex and increasing, then*

$$EG\left(\sup_{t\in T}\sum_{i=1}^{n}\sigma_i\psi_i(t_i)\right) \le EG\left(\sup_{t\in T}\sum_{i=1}^{n}\sigma_i t_i\right).$$

Proof. By conditioning and iteration, it suffices to show that if T is a subset of R^2 and ψ is a contraction on R such that $\psi(0) = 0$, then

$$EG\left(\sup_{t\in T}\sum_{i=1}^{n}t_1 + \sigma_2\psi(t_2)\right) \le EG\left(\sup_{t\in T}\sum_{i=1}^{n}t_1 + \sigma_2 t_2\right).$$

Instead of proving this inequality directly, we show that, for any $t, s \in T$,

$$EG\left(\sup_{t\in T}\sum_{i=1}^{n}t_1 + \sigma_2 t_2\right) \ge \frac{1}{2}G(t_1 + \psi(t_2)) + \frac{1}{2}G(s_1 + \psi(s_2))$$

, which implies the above statement. We may assume that

$$\begin{aligned} (*) \quad t_1 + \psi(t_2) &\le s_1 + \psi(s_2) \\ (**) \quad s_1 - \psi(s_2) &\le t_1 - \psi(t_2) \end{aligned}$$

without loss of generality. We distinguish the following four cases.

Case: $t_2 \ge 0$**,** $s_2 \ge 0$ First, Assume that $s_2 \le t_2$. It suffices to show that

$$\frac{1}{2}G(t_1 + \psi(t_2)) + \frac{1}{2}G(s_1 - \psi(s_2)) \le \frac{1}{2}G(t_1 + t_2) + \frac{1}{2}G(s_1 - s_2)$$

since the right-hand side is obviously less than or equal to $EG(\sup_{t\in T}(t_1 + \sigma_2 t_2))$. Set $a = s_1 - \psi(s_2)$, $b = s_1 - s_2$, $a' = t_1 + t_2$, $b' = t_1 + \psi(t_2)$ so that we sould like to prove that

$$G(a) - G(b) \le G(a') - G(b').$$

Since ϕ is a contraction with $\psi(0) = 0$ and $s_2 \ge 0$, $|\psi(s_2)| \le s_2$. Therefore,

$$a - b = s_2 - \psi(s_2) \ge 0.$$

Due to $(*)$, we also have

$$b' - b = t_1 + \psi(t_2) - s_1 + s_2 \geq s_1 + \psi(s_2) - s_1 + s_2 = \psi(s_2) + s_2 \geq 0.$$

Furthermore, again by contraction and $s_2 \leq t_2$,

$$a - b = s_2 - \psi(s_2) \leq t_2 - \psi(t_2) = a' - b'.$$

Since G is convex and increasing, the map $G(\cdot + x) - G(\cdot)$ is also increasing for any positive x. Taking x as $a - b \geq 0$, we have

$$\begin{aligned} G(a) - G(b) &= G(b + (a-b)) - G(b) \\ &\leq G(b' + (a-b)) - G(b') \\ &\quad \text{(due to that } b \leq b') \\ &\leq G(a') - G(b'). \\ &\quad \text{(using that } a - b \leq a' - b') \end{aligned}$$

When $s_2 \geq t_2$, the argument is similar changing s into t and ψ into $-\psi$.

Case: $t_2 \leq 0$**,** $s_2 \leq 0$ It is completely similar to the preceding case.

Case: $t_2 \geq 0$**,** $s_2 \leq 0$ Since $\psi(t_2) \leq t_2$, $-\psi(s_2) \leq -s_2$, we have directly

$$\frac{1}{2}G(t_1 + \psi(t_2)) + \frac{1}{2}G(s_1 - \psi(s_2)) \leq \frac{1}{2}G(t_1 + t_2) + \frac{1}{2}G(s_1 - s_2). \tag{50}$$

Case: $t_2 \leq 0$**,** $s_2 \geq 0$ Similar to the preceding case.

□

□

Now we turn back to the proof of Theorem 32.

Proof. It suffices to show that $A^{\lambda_n}(\hat{F}_n)$ converges to A^* almost surely since the statement follows from Lemmas 34 and 35. Denote an element of $\mathcal{M}_{\mathcal{K}}$ that minimizes A^{λ_n} by $\bar{F}_n$. Then, we have

$$\begin{aligned} A^{\lambda_n}(\hat{F}_n) - A^* &= A^{\lambda_n}(\hat{F}_n) - A^{\lambda_n}(\bar{F}_n) + A^{\lambda_n}(\bar{F}_n) - A^* \\ &= \{A^{\lambda_n}(\hat{F}_n) - A^{\lambda_n}(\bar{F}_n)\} + \inf_{F \in \mathcal{M}_{\mathcal{K}}} \{A^{\lambda_n}(\bar{F}_n) - A^*\}. \end{aligned} \tag{51}$$

The second term on the right-hand side converges to zero due to the assumption. Due to lemma 8.2 of Devroye et al. (1996), the first term has an upperbound as follows:

$$\begin{aligned} A^{\lambda_n}(\hat{F}_n) - A^{\lambda_n}(\bar{F}_n) &= A^{\lambda_n}(\hat{F}_n) - A_D^{\lambda_n}(\hat{F}_n) + A_D^{\lambda_n}(\hat{F}_n) - A^{\lambda_n}(\bar{F}_n) \\ &\leq \sup_{F\in\mathcal{M}_\mathcal{K}} \left|A^{\lambda_n}(F) - A_D^{\lambda_n}(F)\right| + \{A_D^{\lambda_n}(\bar{F}_n) - A^{\lambda_n}(\bar{F}_n)\} \\ &\leq 2 \sup_{F\in\mathcal{M}_\mathcal{K}} \left|A^{\lambda_n}(F) - A_D^{\lambda_n}(F)\right|. \end{aligned} \tag{52}$$

McDiarmid's inequality (Theorem 8) implies that for any $\delta > 0$

$$P(\sup_{F\in\mathcal{M}_\mathcal{K}} \left|A^{\lambda_n}(F) - A_D^{\lambda_n}(F)\right| - E \sup_{F\in\mathcal{M}_\mathcal{K}} \left|A^{\lambda_n}(F) - A_D^{\lambda_n}(F)\right| > \delta) \leq \exp\left\{\frac{-n\delta^2}{2(\lambda_n\phi'(\lambda_n))^2}\right\}.$$

Or equivalently, at least with probability $1-\delta$,

$$\sup_{F\in\mathcal{M}_\mathcal{K}} \left|A^{\lambda_n}(F) - A_D^{\lambda_n}(F)\right| \leq E \sup_{F\in\mathcal{M}_\mathcal{K}} \left|A^{\lambda_n}(F) - A_D^{\lambda_n}(F)\right| + \lambda_n\phi'(\lambda_n)\sqrt{\frac{2\ln(1/\delta)}{n}}. \tag{53}$$

The first term on the right-hand side of Eq. (53) has an upperbound with respect to Rademacher complexity of $\mathcal{M}_\mathcal{K}$ as follows.

Let $D' = \{X_i', Y_i'\}_{i=1}^n$ be an independent copy of random variables $\{X_i, Y_i\}_{i=1}^n$. Due to the standard symmetrization technique, we have

$$\begin{aligned} E \sup_{F\in\mathcal{M}_\mathcal{K}} \left|A^{\lambda_n}(F) - A_D^{\lambda_n}(F)\right| &= E \sup_{F\in\mathcal{M}_\mathcal{K}} \left|E[A_{D'}^{\lambda_n}(F)] - A_D^{\lambda_n}(F)\right| \\ &\leq E \sup_{F\in\mathcal{M}_\mathcal{K}} \left|A_{D'}^{\lambda_n}(F) - A_D^{\lambda_n}(F)\right| \\ &= \frac{1}{n} E \sup_{F\in\mathcal{M}_\mathcal{K}} \left|\sum_{i=1}^n \phi(-\lambda_n Y_i' F(X_i')) - \phi(-\lambda_n Y_i F(X_i))\right| \\ &= \frac{1}{n} E \sup_{F\in\mathcal{M}_\mathcal{K}} \left|\sum_{i=1}^n \sigma_i\{\phi(-\lambda_n Y_i' F(X_i')) - \phi(-\lambda_n Y_i F(X_i))\}\right| \\ &= \frac{1}{n} E \sup_{F\in\mathcal{M}_\mathcal{K}} \left|\sum_{i=1}^n \sigma_i\{(\phi(-\lambda_n Y_i' F(X_i')) - 1) - (\phi(-\lambda_n Y_i F(X_i)) - 1)\}\right| \\ &\leq \frac{2}{n} E \sup_{F\in\mathcal{M}_\mathcal{K}} \left|\sum_{i=1}^n \sigma_i(\phi(-\lambda_n Y_i F(X_i)) - 1)\right| \\ &\leq \frac{4}{n}\lambda_n\phi'(\lambda_n) E \sup_{F\in\mathcal{M}_\mathcal{K}} \left|\sum_{i=1}^n \sigma_i(-Y_i F(X_i))\right| \\ &= \frac{4}{n}\lambda_n\phi'(\lambda_n) E \sup_{F\in\mathcal{M}_\mathcal{K}} \left|\sum_{i=1}^n \sigma_i F(X_i)\right| \\ &= 2\lambda_n\phi'(\lambda_n) R_n(\mathcal{M}_\mathcal{K}). \end{aligned} \tag{54}$$

The inequality in the third-last line follows from Theorem 37. The second-last equality follows because $Z_i = -\sigma_i Y_i$ is equal to σ_i. The last equality is obtained just by the definition of Rademacher complexity. Define $\mathcal{K} * \mathcal{C}$ as

$$\mathcal{K} * \mathcal{C} = \{K_h(x, x_\ell) f_j(x) \,|\, x_\ell \in \mathcal{K}\,,\ f_j \in \mathcal{C}\}.$$

Rademacher complexity of $\mathcal{M}_\mathcal{K}$ is equal to that of $\mathcal{K} * \mathcal{C}$, which follows from

$$\begin{aligned}
R_n(\mathcal{M}_\mathcal{K}) &= \frac{2}{n} E \sup_{F \in \mathcal{M}_\mathcal{K}} \left| \sum_{i=1}^n \sigma_i F(X_i) \right| \\
&= \frac{2}{n} E \sup_{\{\theta_{j\ell}\}} \left| \sum_{i=1}^n \sigma_i \frac{1}{N_n} \sum_{j=1}^J \sum_{\ell=1}^{N_n} K_h(X_i, x_\ell) \theta_{j\ell} f_j(X_i) \right| \\
&= \frac{2}{n} E \sup_{\{\theta_{j\ell}\}} \left| \sum_{j=1}^J \sum_{\ell=1}^{N_n} \frac{\theta_{j\ell}}{N_n} \sum_{i=1}^n \sigma_i K_h(X_i, x_\ell) f_j(X_i) \right| \\
&= \frac{2}{n} E \sup_{x_\ell \in \mathcal{K}} \sup_{f \in \mathcal{C}} \left| \sum_{i=1}^n \sigma_i K_h(X_i, x_\ell) f(X_i) \right| \\
&= R_n(\mathcal{K} * \mathcal{C}).
\end{aligned}$$

The second-last equality follows because a convex combination takes its extremal value (supremum or infimum) when only one coefficient takes the value of one and others are zero.

Rademacher complexity of $\mathcal{K} * \mathcal{C}$ has an upperbound with respect to V as follows. Let

$$\widehat{R}_n(\mathcal{K} * \mathcal{C}) = \frac{2}{n} E \left[\sup_{F \in \mathcal{K} * \mathcal{C}} \left| \sum_{i=1}^n \sigma_i F(X_i) \right| |X_1, X_2, \cdots, X_n \right].$$

Then, $R_n(\mathcal{K} * \mathcal{C})$ can be written as $R_n(\mathcal{K} * \mathcal{C}) = E\widehat{R}_n(\mathcal{K} * \mathcal{C})$. Define H_f as $\{x \,|\, f(x) = 1\,,\ x \in \mathcal{X}\}$ for any $f \in \mathcal{C}$ and $\mathcal{H}$ as $\{H_f \,|\, f \in \mathcal{C}\}$. Then, $\widehat{R}_n(\mathcal{K} * \mathcal{C})$ is written as

$$\widehat{R}_n(\mathcal{K} * \mathcal{C}) = \frac{2}{n} E \left[\max_{x_\ell \in \mathcal{K}} \sup_{H_f \in \mathcal{H}} \left| \sum_{i=1}^n \sigma_i K_h(X_i, x_\ell) (I(X_i \in H_f) - I(X_i \notin H_f)) \right| |X_1, X_2, \cdots, X_n \right].$$

Then, we have

$$\begin{aligned}
\widehat{R}_n(\mathcal{K} * \mathcal{C}) &= \frac{2}{n} E \left[\max_{x_\ell \in \mathcal{K}} \sup_{H_f \in \mathcal{H}} \left| \sum_{i=1}^n \sigma_i K_h(X_i, x_\ell) (I(X_i \in H_f) - I(X_i \notin H_f)) \right| |X_1, X_2, \cdots, X_n \right] \\
&= \frac{2}{n} E \left[\max_{x_\ell \in \mathcal{K}} \max_{H_f \in \hat{\mathcal{H}}} \left| \sum_{i=1}^n \sigma_i K_h(X_i, x_\ell) (I(X_i \in H_f) - I(X_i \notin H_f)) \right| |X_1, X_2, \cdots, X_n \right],
\end{aligned}$$

where $\hat{\mathcal{H}}$ denotes the subset of $\mathcal{H}$ in which any two elements in $\hat{\mathcal{H}}$ have different intersections with $\{X_i\}_{i=1}^n$. Lemma 4 implies that

$$E\left[\exp(\theta\sum_{i=1}^{n}\sigma_i K_h(X_i,x_\ell)(I(X_i\in H_f)-I(X_i\notin H_f)))|X_1,X_2,\cdots,X_n\right]\leq\exp(n\theta^2/2)$$

Thus, by applying Lemma 6, we have

$$E\left[\max_{x_\ell\in\mathcal{K}}\max_{H_f\in\hat{\mathcal{H}}}\left|\sum_{i=1}^{n}\sigma_i K_h(X_i,x_\ell)(I(X_i\in H_f)-I(X_i\notin H_f))\right||X_1,X_2,\cdots,X_n\right]\leq\sqrt{2n\ln(2N_n|\hat{\mathcal{H}}|)}.$$

Let us denote the VC shatter coefficient of $\mathcal{H}$ by $\mathcal{S}_{\mathcal{H}}$. By applying Proposition 13, we have

$$\begin{aligned}
\widehat{R}_n(\mathcal{K}*\mathcal{C}) &\leq \frac{2}{n}\sqrt{2n\ln 2n^\beta\mathcal{S}_{\mathcal{H}}(n)}\\
&\leq 2\sqrt{\frac{2n\ln 2n^\beta(n+1)^V}{n}}\\
&\leq 2\sqrt{\frac{2\ln 2(n+1)^{V+\beta}}{n}}\\
&\leq 2\sqrt{\frac{2\ln(2(n+1))^{V+\beta}}{n}}\\
&= 2\sqrt{\frac{2(V+\beta)\ln 2(n+1)}{n}}.
\end{aligned}$$

By taking expectations of both sides, we have

$$R_n(\mathcal{K}*\mathcal{C})\leq 2\sqrt{\frac{2(V+\beta)\ln 2(n+1)}{n}}. \tag{55}$$

Eqs. (52), (53), and (55) imply that the first term of Eq. (51) is bounded with at least probability $1-\delta$ as

$$A^{\lambda_n}(\hat{F}_n)-A^{\lambda_n}(\bar{F}_n)\leq 8\lambda_n\phi'(\lambda_n)\sqrt{\frac{2(V+\beta)\ln 2(n+1)}{n}}+2\lambda_n\phi'(\lambda_n)\sqrt{\frac{2\ln(1/\delta)}{n}}. \tag{56}$$

□

Inspection of the proof of Theorem 32 provides a useful viewpoint for comparison between the usual boosting and the local boosting. The left-side of Eq. (51) is referred to as *risk bias*. Risk bias can be decomposed into two parts as in Eq. (51). We write the decomposition here again for convenience.

$$A^{\lambda_n}(\hat{F}_n)-A^*=\{A^{\lambda_n}(\hat{F}_n)-A^{\lambda_n}(\bar{F}_n)\}+\inf_{F\in\mathcal{M}_{\mathcal{K}}}\{A^{\lambda_n}(\bar{F}_n)-A^*\}.$$

The first term is referred to as *estimation error* and the second term is referred to as *approximation error*. Clearly, both terms always take nonnegative values. First, we should mention what this theorem indicates about the usual boosting. As described in the previous section, the usual boosting can be regarded as the local boosting with $h \to \infty$ and $\mathcal{K}$ consisting of an arbitrary single point ($N = 1$). In this case, the assumption in Eq. (48) about the approximation error reduces to

$$\lim_{\lambda \to \infty} \inf_{F \in \mathcal{M}} A^{\lambda}(F) - A^* = 0$$

and the upperbound of the estimation error reduces to

$$A^{\lambda_n}(\hat{F}_n) - A^{\lambda_n}(\bar{F}_n) \leq 8\lambda_n \phi'(\lambda_n) \sqrt{\frac{2V \ln 2(n+1)}{n}} + 2\lambda_n \phi'(\lambda_n) \sqrt{\frac{2\ln(1/\delta)}{n}}$$

because $N \equiv 1$ implies that $\beta = 0$. The usual boosting and the local boosting have different characteristics in view of the estimation error and the approximation error.

The estimation error of the local boosting may be worse than that of the usual boosting when the same base classifiers are used. The estimation error reflects the risk difference between the minimizer of expected loss and empirical loss in model $\mathcal{M}_{\mathcal{K}}$. If a model is more complicated, the estimation error may be larger because the model may be more overfit to the training data more. In fact, the estimation error relates to the model complexity as was seen in Eq. (54). Inspection of Eq. (56) indicates that the estimation error of both boosting methods approach zero as $n \to \infty$ when $\mathcal{C}$ has a finite VC dimension. The remarkable difference is the increase, β, in V, which depends on the number of kernel center candidates. Therefore, the estimation error of the local boosting may be worse than that of usual boosting in practical situations ($n < \infty$). However, the extent of the increase is not significant in general because $\beta = \ln N_n / \ln n$. Specifically, $\beta = 1$ when $\mathcal{K} = \mathcal{K}_*$, which is defined in Eq. (47). If we use the usual boosting with more complicated base classifiers instead of employing the local boosting, then the corresponding VC dimension, V, itself increases. Compared to this case, $\beta = 1$ is the least increase.

The local boosting may improve the approximation error significantly compared to the usual boosting. The approximation error reflects the model's approximation ability. As described above, the estimation errors of both boosting methods with appropriate $\mathcal{C}$ are reduced to zero as $n \to \infty$. Therefore, the Bayes risk consistency of both boosting methods depends on whether or not their approximation error reduces to zero as $n \to \infty$, *i.e.*, the assumption in Eq. (48). The local boosting may have a considerably smaller

approximation error than that of the usual boosting. To see this simply, suppose that $\phi = \exp$ in the remainder of this section. As described in Section 2.5.2, Friedman et al. (2000) shows that A^* is attained by half log-odds denoted by F^*, *i.e.*,

$$F^*(x) = \frac{1}{2} \ln \frac{P(Y=1 \mid x)}{P(Y=-1 \mid x)}.$$

The approximation error is reduced to zero as $n \to \infty$ if there exists $F(x) \in \mathrm{lin}(\mathcal{C})$ such that $F^*(x) = F(x)$ where $\mathrm{lin}(\mathcal{C})$ denotes the set of all linear combinations of $\mathcal{C}$. Note that the sum of coefficients is not restricted to one here due to the fact that $\lambda_n \to \infty$ as $n \to \infty$ combined with the following proposition.

Proposition 39. *Assume that $F^*(x)$ takes its values in the closed interval $[-u, u]$, and $\mathcal{C}$ is negation closed. If there exists a real number $r > 0$ and $F(x) \in \mathcal{M}$ such that $F^*(x) = rF(x)$, then there necessarily exists $F'(x) \in \mathcal{M}$ such that $F^*(x) = \lambda F'(x)$ for any $\lambda > r$.*

Proof. Because $\mathcal{C}$ is negation closed, it can necessarily be partitioned into two complementary sets: $\{f_j^+(x)\}_{j=1}^{J/2}$ and $\{f_j^-(x)\}_{j=1}^{J/2}$ such that $f_j^+ = -f_j^-$ for any $j \in \{1, 2, \cdots, J/2\}$. Without loss of generality, there exists a set of nonnegative coefficients $\{\theta_j^*\}_{j=1}^{J}$ such that $F^*(x) = r \sum_{j=1}^{J/2} \theta_j^{*+} f_j^+(x)$ and $\sum_{j=1}^{J/2} \theta_j^{*+} = 1$. Now $F'(x)$ can be written as

$$F'(x) = \sum_{j=1}^{J/2} \lambda \theta_j^+ f_j^+(x) + \sum_{j=1}^{J/2} \lambda \theta_j^- f_j^-(x).$$

It suffices to find nonnegative $\{\theta_j^+, \theta_j^-\}_{j=1}^{J/2}$ such that $\sum_{j=1}^{J/2} \theta_j^+ + \theta_j^- = 1$ and $F^*(x) = \lambda F'(x)$. Let $\theta_j^+ = (1-u)\theta_j^{*+}$ and $\theta_j^- = u\theta_j^{*+}$ for some $0 < u < 1$. Clearly, these $\{\theta_j^+, \theta_j^-\}_{j=1}^{J/2}$ satisfy $\sum_{j=1}^{J/2} \theta_j^+ + \theta_j^- = 1$. Due to the condition that $\lambda F'(x) = F^*(x)$, we have the equations

$$r\theta_j^{*+} = \lambda\{(1-u)\theta_j^{*+}\}$$

for any j. Because $\lambda > r$, this equation has a unique solution $u = \frac{\lambda - r}{2\lambda}$. Then, taking

$\theta_j^+ = \frac{\lambda+r}{2\lambda}\theta_j^{*+}$ and $\theta_j^- = \frac{\lambda-r}{2\lambda}\theta_j^{*+}$, we have

$$\begin{aligned}
\lambda F'(x) &= \lambda \sum_{j=1}^{J/2} \theta_j^+ f_j^+(x) + \lambda \sum_{j=1}^{J/2} \theta_j^- f_j^-(x) \\
&= \lambda \sum_{j=1}^{J/2} (\theta_j^+ - \theta_j^-) f_j^+(x) \\
&= \lambda \sum_{j=1}^{J/2} \frac{r}{\lambda} \theta_j^{*+} f_j^+(x) \\
&= r \sum_{j=1}^{J/2} \theta_j^{*+} f_j^+(x) \\
&= F^*(x)
\end{aligned}$$

□

The fact that $F(x) \in \mathrm{lin}(\mathcal{C})$ is equivalent to the fact that the underlying distribution $P(Y=y \,|\, x)$ is included in the exponential model associated with $\mathcal{M}$, which is defined as

$$\mathcal{P}(\mathcal{M}) = \left\{ \frac{\exp(-(y+1)\sum_{j=1}^J \theta_j f_j(x))}{1+\exp(-2\sum_{j=1}^J \theta_j f_j(x))} \;\middle|\; \forall j \,,\, \theta_j \geq 0 \right\}. \tag{57}$$

In contrast, the approximation error of the local boosting is reduced to zero as $n \to \infty$ if $P(Y=y \,|\, x)$ is included in the local exponential model associated with $\mathcal{C}$, which is defined as

$$\mathcal{P}(\mathcal{M}_\mathcal{K}) = \left\{ \frac{\exp(-(y+1)\sum_{j=1}^J \bar{\theta}_j(x) f_j(x))}{1+\exp(-2\sum_{j=1}^J \bar{\theta}_j(x) f_j(x))} \;\middle|\; \forall j \,,\, \bar{\theta}_j(x) = \frac{1}{N_n}\sum_{\ell=1}^{N_n} K_h(x, x_\ell)\theta_{j\ell} \,,\, \theta_j \geq 0 \right\}. \tag{58}$$

Specifically, the following asymptotical local exponential model associated with $\overline{\mathcal{M}}_\mathcal{K}$ replaces Eq. (58) when $\mathcal{K} = \mathcal{K}_*$.

$$\mathcal{P}(\overline{\mathcal{M}}_\mathcal{K}) = \left\{ \frac{\exp(-(y+1)\sum_{j=1}^J \bar{\theta}_j(x) f_j(x))}{1+\exp(-2\sum_{j=1}^J \bar{\theta}_j(x) f_j(x))} \;\middle|\; \forall j \,,\, \bar{\theta}_j(x) = E[K_h(x, X')\theta_j(X')] \,,\, \theta_j(x) \geq 0 \right\} \tag{59}$$

There also exists no restriction on the sum of coefficients in models Eq. (58) and (59) due to the same reason as that in the case of Eq. (57). Suppose now that $\mathcal{P}(\mathcal{M})$ is a misspecified model, *i.e.*, $P(Y=y \,|\, x) \notin \mathcal{P}(\mathcal{M})$ (or equivalently, a single linear combination of $\mathcal{C}$ cannot approximate F^*). In this case, the approximation error of the usual boosting

does not vanish even in the asymptotical sense. The local boosting, however, may reduce its approximation error to zero if $\mathcal{P}(\mathcal{M})$ is locally correct. To see this formally, let $\{\mathcal{X}_q\}_{q=1}^Q$ be a partition of $\mathcal{X}$, *i.e.*, $\mathcal{X} = \cup_{q=1}^Q \mathcal{X}_q$ and $\mathcal{X}_q \cap \mathcal{X}_{q'} = \varphi$ for any $q \neq q'$ where φ denotes an empty set. $\mathcal{P}(\mathcal{M})$ is locally correct means that there exist positive $\{\theta_{jq}\}$ such that, for a partition $\{X_q\}$,

$$P(Y=y \,|\, x) = \sum_{q=1}^{Q} I(x \in \mathcal{X}_q) \frac{\exp(-(y+1)F_q(x))}{1+\exp(-2F_q(x))}$$

where $F_q(x) \in \mathcal{M}$ for each q, or, equivalently,

$$F^*(x) = \sum_{q=1}^{Q} I(x \in \mathcal{X}_q) F_q(x). \tag{60}$$

In this case, $F^*(x)$ can be approximated by the local boosting as follows. Let $\Gamma_q = \{\ell \,|\, x_\ell \in \mathcal{X}_q\}$ and $\mathcal{I}_q(x) = (1/N)\sum_{\ell \in \Gamma_q} K_h(x, x_\ell)$. Then, the local boosting may construct a discriminant function such that

$$F(x) = \sum_{q=1}^{Q} \mathcal{I}_q(x) F_q(x).$$

When $\mathcal{K}$ is distributed with sufficient density in each $\mathcal{X}_q$, and an appropriate h is selected, $\mathcal{I}_q(x)$ can approximate $I(x \in \mathcal{X}_q)$ to some extent. Therefore, this localized discriminant function, $F(x)$, can approximate $F^*(x)$ more accurately than discriminant functions in $\mathrm{lin}(\mathcal{C})$. In particular, restricting our goal to decrease only approximation error, h should be as small as possible. In fact, $h \to 0$ enhances the approximation ability of $\mathcal{P}(\mathcal{M}_\mathcal{K})$ as well as that of nonparametric models when $n \to \infty$. To see this, assume that $K_h(\cdot,\cdot)$ is a standard gaussian kernel and that $\mathcal{K} = \mathcal{K}_*$. Taking $h_n = (1/\pi)\lambda_n^{-2/M}$ as the bandwidth of the kernel, we have

$$\begin{aligned}
\lim_{n\to\infty} \lambda_n \bar{\theta}_j(x) &= \lim_{n\to\infty} \frac{\lambda_n}{n} \sum_{\ell=1}^{n} K_{h_n}(x, X_\ell)\theta_{j\ell} \\
&= \lim_{n\to\infty} \frac{1}{n} \sum_{\ell=1}^{n} \mathcal{N}(X_\ell; x, \frac{h_n}{2} I_M)\theta_{j\ell} \\
&= E[\mathcal{N}(X; x, \frac{h_n}{2} I_M)\theta_j(X)] \\
&= \theta_j(x) P(x),
\end{aligned}$$

where $\mathcal{N}(\mu, \Sigma)$ is a normal distribution with mean μ and covariance Σ. I_M denotes an

M-dimensional identity matrix. Thus, $\mathcal{P}(\mathcal{M}_{\mathcal{K}})$ approaches the set

$$\left\{ \frac{\exp(-(y+1)\sum_{j=1}^{J}\tilde{\theta}_j(x)f_j(x))}{1+\exp(-2\sum_{j=1}^{J}\tilde{\theta}_j(x)f_j(x))} \;\middle|\; \forall j\,,\, \tilde{\theta}_j(x) \geq 0 \right\}, \tag{61}$$

where $\tilde{\theta}_j(x) = \theta_j(x)P(x)$. In this case, a discriminant function has the form $F(x) = \sum_{j=1}^{J}\tilde{\theta}_j(x)f_j(x)$. Clearly, this model can approximate $F^*(x)$ with the form Eq. (60) with any complicated or fine partition and then can decrease the approximation error to zero in general. In practical situations, however, the local boosting with $h \to 0$ performs poorly due to the following reason. Recall the same notations defined in the previous section. If h is smaller, $\overline{\mathcal{B}}_{\epsilon}(\mathcal{K}, h)$ is larger for fixed $\mathcal{K}$. In $\overline{\mathcal{B}}_{\epsilon}(\mathcal{K}, h)$, the absolute value of $F(x)$ is less than ϵ. If h is too small, there may exist a large area such that

$$\{x \,|\, \lambda_n \epsilon \ll F^*(x)\}.$$

Therefore, an excessively small h increases the approximation error in general unless $\mathcal{K}$ is sufficiently dense such that $\mathcal{B}_{\epsilon}(\mathcal{K}, h)$ covers $\mathrm{supp}(X)$. However, the enlargement of $\mathcal{K}$ increases the upperbound of the estimation error as was seen above. Specifically, when $h \to 0$ the upperbound of the estimation error diverges to infinity because $\mathcal{K}$ is required to be an infinite set. The localizing factor controls this trade-off between the estimation error and the approximation error. In other words, the localizing factor connects the parametric model, $\mathcal{P}(\mathcal{M})$, with the nonparametric model in Eq. (61) smoothly. A similar interpretation is discussed by Eguchi et al. (2003) in the framework of the local likelihood. Some references studied the optimal selection of bandwidth h in this framework (Fan and Gijbels, 1995; Fan et al., 1998). Such studies may apply to the local boosting to select the optimal localizing factor. We note that these observations may apply similarly to a general function, ϕ, that satisfies Condition 18.

Case	Bandwidth h	Kernel Centers $\mathcal{K}$	Approximation Error	Estimation Error
(a)	small	dense	small	large
(b)	large	sparse	large if $P(Y=y \,|\, x) \notin \mathcal{P}(\mathcal{M})$	small
(c)	large	dense	large if $P(Y=y \,|\, x) \notin \mathcal{P}(\mathcal{M})$	large
(d)	small	sparse	surely large	small

Table 7: Summary of trade-off between estimation error and approximation error.

As a result, the local boosting may have the Bayes risk consistency in wider situations than those of the usual boosting when a localizing factor is appropriately set. The dependence of the trade-off between the estimation error and the approximation error on a

localizing factor is summarized in Table 7. The selection of a localizing factor in Cases (c) and (d) is undesirable, which, however, may sometimes occur in practical applications. Case (d), *i.e.*, selection of a small bandwidth, h, and a sparse $\mathcal{K}$ causes a large $\overline{\mathcal{B}}_\epsilon(\mathcal{K}, h)$. Thus, the approximation error is significantly large unless $F^*(x)$ is almost zero for any $x \in \overline{\mathcal{B}}_\epsilon(\mathcal{K}, h)$. Under the restriction that $\overline{\mathcal{B}}_\epsilon(\mathcal{K}, h)$ is small, if a bandwidth, h, is smaller, $\mathcal{K}$ should be more dense. We assume that h and $\mathcal{K}$ pairs in Cases (a)-(c) are designed to satisfy this restriction. Therefore, the approximation errors in Cases (b) and (c) are small if $\mathcal{P}(\mathcal{M})$ is correct. In Case (c), however, the estimation error may be large because $\mathcal{K}$ is dense. Case (c) is the worst case when $\mathcal{P}(\mathcal{M})$ is a misspecified model. Cases (a) and (b) illustrate proper selections of localizing factors. The usual boosting is a special case of Case (b). If $\mathcal{P}(\mathcal{M})$ is correct, Case (b) illustrates the best selection. However, the approximation error is large when $\mathcal{P}(\mathcal{M})$ is not correct, even though the estimation error is small. Here, Case (a) illustrates the best case. The localizing factor should be selected such that h is as large as possible and $\mathcal{K}$ is as sparse as possible, unless the approximation error increases. We recommend the use of $\mathcal{K}_*$ as $\mathcal{K}$. When $\mathcal{K} = \mathcal{K}_*$, the upperbound of estimation error for the local boosting increases the least compared to the use of more complicated base classifiers. In addition, $\mathcal{K}_*$ with a proper bandwidth h may also decrease the approximation error asymptotically due to the following reason. When we use $\mathcal{K}_*$ with a proper h, $\overline{\mathcal{B}}_\epsilon(\mathcal{K}, h)$ may be small since $\mathcal{K}_*$ is distributed all over $\mathrm{supp}(X)$ according to the underlying distribution. In particular, the local boosting with $\mathcal{K}_*$ may have the strongest approximation ability as $n \to \infty$ since we can reduce the bandwidth h to zero without increase of $\overline{\mathcal{B}}_\epsilon(\mathcal{K}, h)$. The use of $\mathcal{K}_*$ will also be supported by several simulations in the next section.

We illustrate the situation where $P(Y=y \,|\, x) \notin \mathcal{P}(\mathcal{M})$ when decision stumps $\mathcal{C}_{\mathrm{ds}}$ (See Section 2.4.1) are used as base classifiers. Suppose the following XOR situation where the Bayes risk is not zero:

$$\begin{aligned} P(Y=1 \,|\, x) &= (1-\rho)\{I(x \in \mathcal{X}_1) + I(x \in \mathcal{X}_3)\} + \rho\{I(x \in \mathcal{X}_2) + I(x \in \mathcal{X}_4)\} \\ P(Y=-1 \,|\, x) &= \rho\{I(x \in \mathcal{X}_1) + I(x \in \mathcal{X}_3)\} + (1-\rho)\{I(x \in \mathcal{X}_2) + I(x \in \mathcal{X}_4)\} \end{aligned}$$

where $\mathcal{X}_q$ $(q = 1, 2, 3, 4)$ are those in Fig. 16 and $0 < \rho < 1/2$. The half log-odds in this case is easily calculated:

$$F^*(x) = \frac{1}{2} \ln \frac{1-\rho}{\rho} \sum_{q=1}^{4} (-1)^{q+1} I(x \in \mathcal{X}_q). \tag{62}$$

The Bayes risk is also easily calculated as $L^* = \rho$. Suppose that we use decision stumps as base classifiers. Any discriminant function consisting of a linear combination of decision stumps can be written with the form

$$F(x) = \sum_{m=1}^{M} S_m((x)_m),$$

where $S_m((x)_m)$ is a score function defined in Eq. (35) However, F^* in Eq. (62) cannot be written with such an additive form. Clearly, $F^* \notin \mathrm{lin}(\mathcal{C}_{\mathrm{ds}})$ even though $\mathcal{C}_{\mathrm{ds}}$ is widely used base classifiers. In fact, the best classifier in $\mathrm{lin}(\mathcal{C}_{\mathrm{ds}})$ is

$$F(x) = \frac{1}{4} \ln \frac{1-\rho}{\rho} (f^{\mathrm{s}}(x; 1, -1, 0) + f^{\mathrm{s}}(x; 2, -1, 0)),$$

in which $L(\mathrm{sign}(F)) = \rho + \frac{1}{4}(1 - 2\rho)$. $\mathcal{M}_{\mathcal{K}}$ may construct a discriminant function that is closer to F^*. Let $\mathcal{K}$ be a set of each center of $\mathcal{X}_q$, *i.e.*, $\{x_q\}_{q=1}^4$ in Fig. 16. We use a rectangular kernel function (*i.e.*, $k_{\mathrm{rec}}(z)$ and L_1 norm) and the bandwidth h is set to v. Denote $f_1(x) = f^{\mathrm{s}}(x; 2, 1, 0)$ and $f_2(x) = f^{\mathrm{s}}(x; 2, -1, 0)$. Set $\{\theta_{jq}\}$ as

$$\theta_{jq} = \begin{cases} \frac{1}{2} \ln \frac{1-\rho}{\rho} \{I(x_q \in \mathcal{X}_1) + I(x_q \in \mathcal{X}_4)\} & (j = 1) \\ \frac{1}{2} \ln \frac{1-\rho}{\rho} \{I(x_q \in \mathcal{X}_2) + I(x_q \in \mathcal{X}_3)\} & (j = 2) \end{cases}.$$

Then, we have

$$F^*(x) = \sum_{j=1}^{2} \sum_{q=1}^{4} K_v(x, x_q) \theta_{jq} f_j(x).$$

Thus, the local boosting has the Bayes risk consistency. In practical cases, choosing such an opportune $\mathcal{K}$ and h is difficult, particularly in high-dimensional cases. We, however, expected that $\mathcal{K}_*$ with a proper h performs well. In more detail, $\{\theta_{j\ell}\}$ is well estimated if $B_\epsilon(\ell, h)$ does not include the origin. Otherwise, the estimation may fail due to the same reason as that in the case of $\mathrm{lin}(\mathcal{C})$. If h is small, the number of latter cases is reduced. Thus, when a sufficiently large training data is given such that $\mathcal{B}_\epsilon(\mathcal{K}_*, h)$ covers $\mathcal{X}$, the local boosting may construct the Bayes classifier.

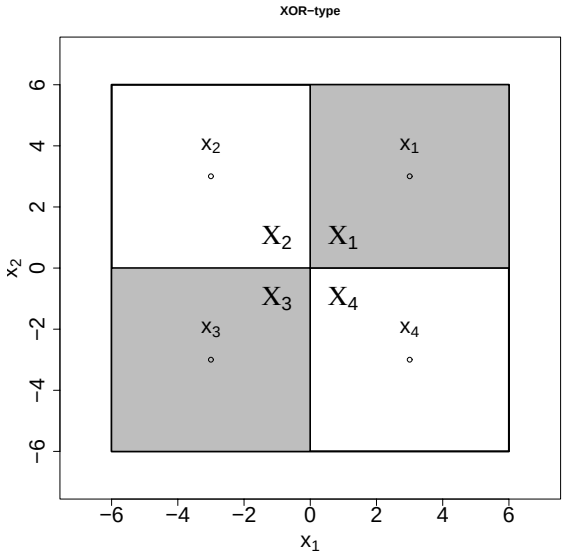


Figure 16: XOR problem where the Bayes risk is not zero is illustrated with $v = 6$. Positive samples are distributed in shaded areas, $\mathcal{X}_1$ and $\mathcal{X}_3$, with probability $1 - \rho$ and in other areas, $\mathcal{X}_2$ and $\mathcal{X}_4$, with probability ρ. Vice versa for negative samples.

4.2.3 Local least favorable error property

The local AdaBoost has the least favorable property of error rate, that is similar to AdaBoost. The algorithm of AdaBoost is characterized in terms of the weighted error rate. At each step, it holds that the current chosen classifier has always the worst weighted error rate 1/2 at the next step, that is, $\epsilon_t(f_{j(t-1)}) = 1/2$. We may show that the local AdaBoost has this property locally.

Proposition 40. *$f_{j(t-1)}$ that is the locally best classifier when the kernel center is $x_{\ell(t-1)}$ has necessarily the least favorable weighted error rate $1/2$ at the earliest step when the kernel center backs to $x_{\ell(t-1)}$ if either of the following cases occur:*

Case (1) *The kernel center does not move, that is, $x_{\ell(t)} = x_{\ell(t-1)}$.*

Case (2) *The kernel center comes back to $x_{\ell(t-1)}$ at $t+1$ after moving to $x_{\ell(t)}$ such that $B_\epsilon(\ell(t-1), h) \cap B_\epsilon(\ell(t), h) = \varphi$, where φ denotes an empty set.*

Note that sufficiently small ϵ means that, for any $x \notin B_{\ell,h}$, we may treat $K_h(x, x_\ell)$ as zero.

Proof. **Case (1)** α_{t-1} satisfies

$$\frac{1}{n}\sum_{i=1}^{n}\phi'(-Y_iF_{t-1}(X_i))(-Y_iK_h(X_i,x_{\ell(t-1)})f_{j(t-1)}(X_i))=0. \tag{63}$$

since α_{t-1} should be the solution of Eq. (39). Since $x_{\ell(t)}=x_{\ell(t-1)}$, we have, by replacing $K_h(X_i,x_{\ell(t-1)})$ with $K_h(X_i,x_{\ell(t)})$ for any i,

$$\frac{1}{n}\sum_{i=1}^{n}\phi'(-Y_iF_{t-1}(X_i))(-Y_iK_h(X_i,x_{\ell(t)})f_{j(t-1)}(X_i))=0.$$

By dividing the both side of this equation by $\frac{1}{n}\sum_{i'=1}^{n}\phi'(-Y_{i'}F_{t-1}(X_{i'}))K_h(X_{i'},x_{\ell(t)})$, we have

$$\sum_{i=1}^{n}w_t(i)(-Y_if_{j(t-1)}(X_i))=0.$$

This completes the proof since we can rewrite the left-hand side of this equation as

$$\begin{aligned}\sum_{i=1}^{n}w_t(i)(-Y_if_{j(t-1)}(X_i)) &= \sum_{i=1}^{n}w_t(i)\{I(Y_i\neq f_{j(t-1)}(X_i))-(1-I(Y_i\neq f_{j(t-1)}(X_i)))\}\\ &= 2\epsilon_t(f_{j(t-1)})-1.\end{aligned}$$

Case (2) Eq. (63) also holds for this case. By assumption, we have, for any $i\in\{1,2,\cdots,n\}$,

$$\begin{aligned}\phi'(-Y_iF_t(X_i))K_h(X_i,x_{\ell(t+1)}) &= \phi'(-Y_i(F_{t-1}(X_i)+\alpha_tK_h(X_i,x_{\ell(t)})f_{j(t)}))K_h(X_i,x_{\ell(t-1)})\\ &= \begin{cases}0 & (X_i\notin B_\epsilon(\ell(t-1),h))\\ \phi'(-Y_iF_{t-1}(X_i))K_h(X_i,x_{\ell(t-1)}) & (\text{otherwise})\end{cases}\\ &= \phi'(-Y_iF_{t-1}(X_i))K_h(X_i,x_{\ell(t-1)})\end{aligned}$$

Thus, it holds that

$$\frac{1}{n}\sum_{i=1}^{n}\phi'(-Y_iF_t(X_i))(-Y_iK_h(X_i,x_{\ell(t+1)})f_{j(t-1)}(X_i))=0. \tag{64}$$

Similarly to the Case (1), this completes the proof.

□

It is worth mentioning that the proof of the Case (2) implies that the local least favorable property holds also when the kernel center comes backs to $x_{\ell(t-1)}$ after any number of steps t' such that $B_\epsilon(\ell(t-1),h)\cap\{B_\epsilon(\ell(t+1),h)\cup B_\epsilon(\ell(t+2),h)\cup\cdots\cup B_\epsilon(\ell(t+t'),h)\}=\varphi$.

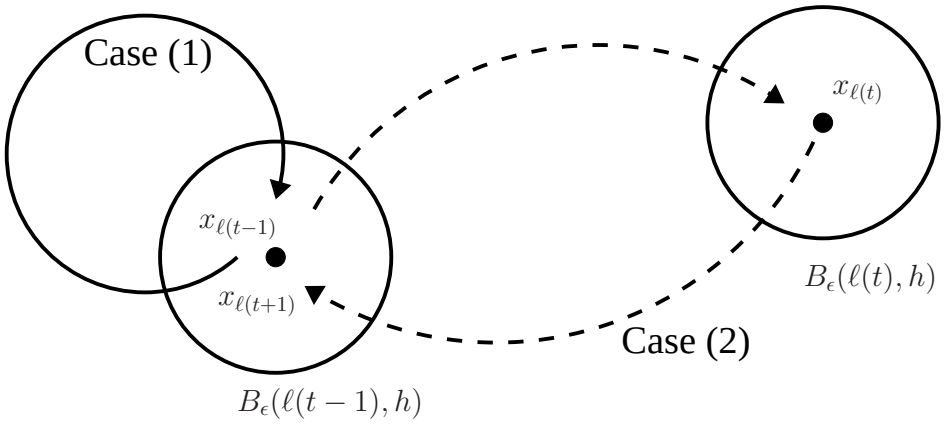


Figure 17: Move of kernel center in Case (1) and (2) of Proposition 40.

4.3 Simulations

Several simulations demonstrate the performance of the local AdaBoost, compared to the usual regularized AdaBoost. In simulations, we used decision stumps as base classifiers for both methods in the manner as explained in Section 2.4.1. We also used $\mathcal{K}_*$ and a gaussian kernel (*i.e.*, K_h with k_{gau}). The optimization over the coefficient of the current classifier in Eq. (39) was performed by a quasi-Newton method implemented in the function "*optim*" written in R language. The software code of the local boosting will appear in `http://www.ism.ac.jp/~eguchi/homepage/myhomepage.html`. Through all simulations, the underlying distribution of X is

$$P(x) = \frac{1}{4v^2} \sum_{q=1}^{4} I(x \in \mathcal{X}_q),$$

where $v = 6$. The situation in which $P(Y = y \,|\, x) \in \mathcal{P}(\mathcal{M})$ is illustrated in Figs. 18 and 19. The difference between them is only the size of the training data, while test data of both cases consist of 600 samples. The usual AdaBoost (Case (b) in Table 7) constructs an accurate decision boundary consistently in both cases. The local AdaBoost with $h = 3$ (Case (a) in Table 7) performs poorly in Fig. 18 because the local AdaBoost constructs a decision boundary that is too flexible and overfit to the training data. This reflects the increase in the estimation error as was discussed in the previous section. Fig. 19, however, indicates that the local boosting also performs well when the size of the training data is sufficient.

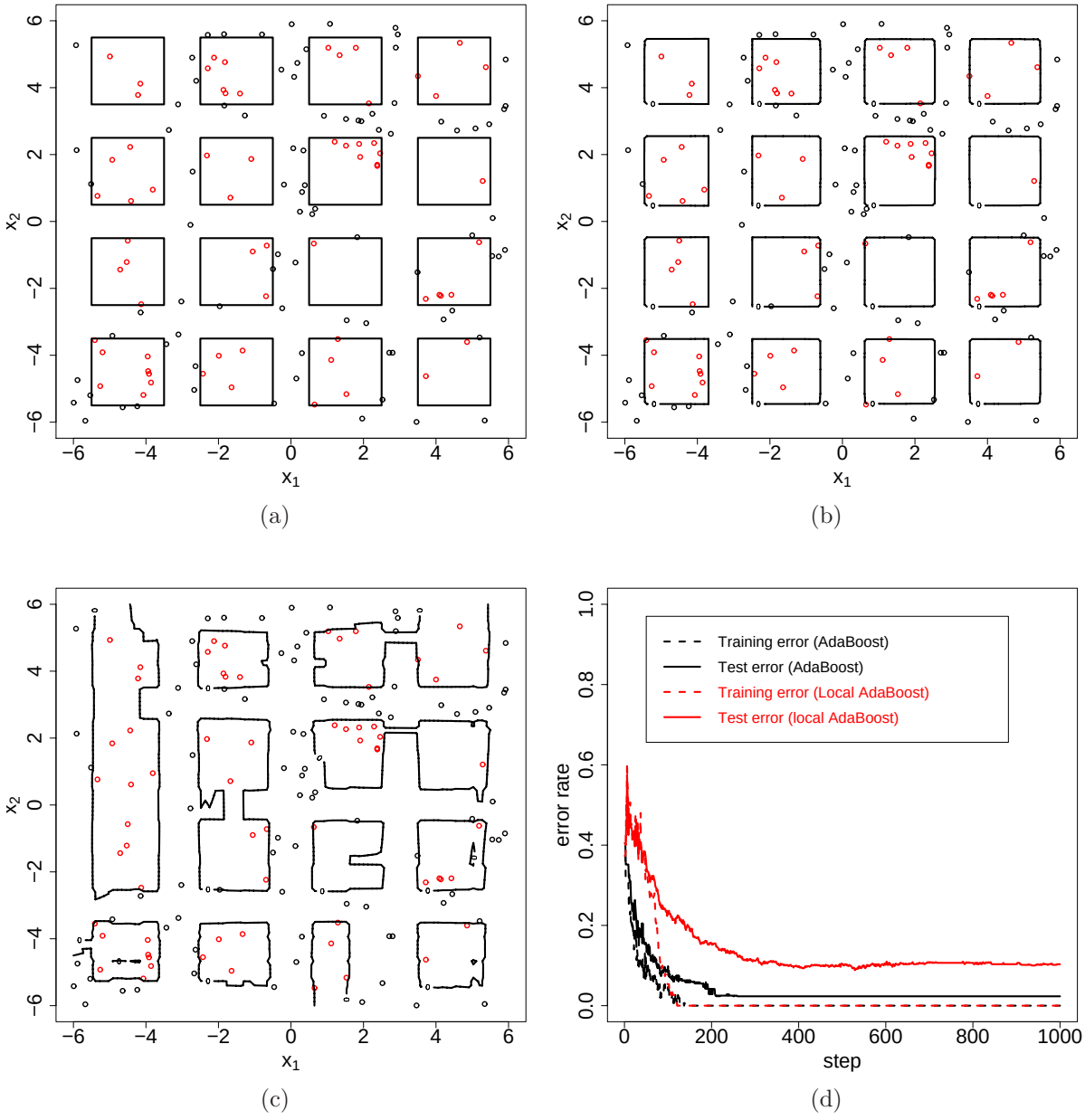


Figure 18: Squares data: positive samples are distributed inside squares, while negative samples are distributed out side of squares. The panel (a) shows a training data set with optimal decision boundary. The panel (b) shows a decision boundary that was estimated by AdaBoost, while the panel (c) shows that of the local AdaBoost. The panel (d) shows plots of error rates on training data and test data against step. The training data set consists of 150 samples, while the test data set consists of 600 samples.

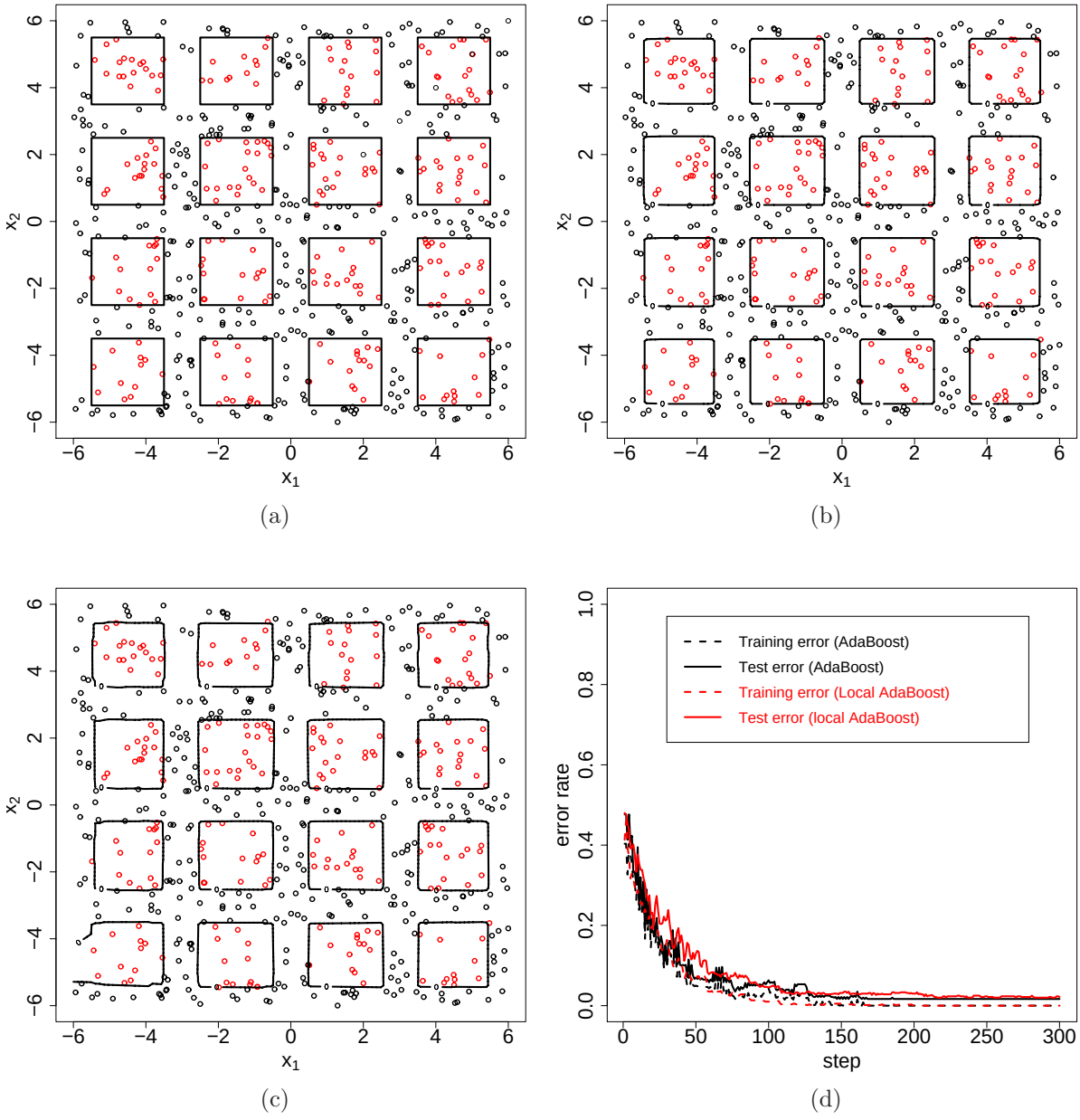


Figure 19: Squares data: positive samples are distributed inside squares, while negative samples are distributed out side of squares. The panel (a) shows a training data set with optimal decision boundary. The panel (b) shows the decision boundary that was estimated by AdaBoost, while the panel (c) shows that of the local AdaBoost. The panel (d) shows plots of error rates on the training data set and the test data set against step. Both data sets consist of 600 samples.

The XOR example, which was already introduced in the previous section, is illustrated

in Fig. 20. In our simulations, this is a unique example where the Bayes risk is not zero but 0.1. The local AdaBoost improved the performance of the usual AdaBoost although it is overfit to the training data to some extent. The test error ($T = 400$) of the local AdaBoost for various h in the same problem is shown in Fig. 21. From Fig. 21, we see that the performance of the local AdaBoost may be worse when h is large or too small. When h is large, there are many kernel centers x_ℓ such that $B_\epsilon(\ell, h)$ includes the origin (Case (b) in Table 7). When h is too small, the local AdaBoost suffers from significant overfitting (Case (d) in Table 7). These reflect the discussion in the previous section. In contrast, the XOR example with the Bayes risk equal to zero is illustrated in Fig. 22. In this case, local AdaBoost performs significantly better.

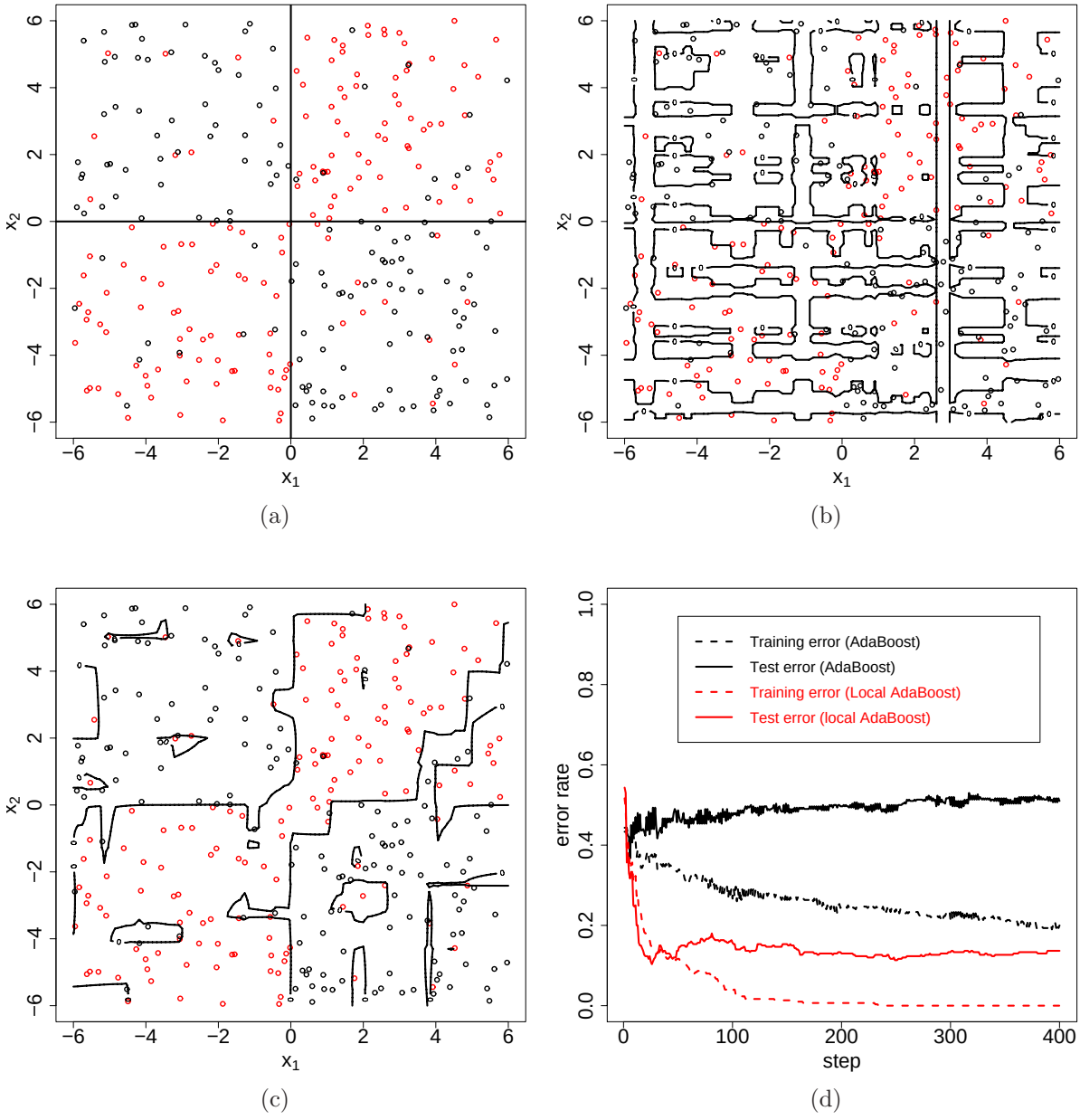


Figure 20: XOR data: Samples are generated in XOR setting with $\rho = 0.1$, which was explained in Section 4.2.2 and Fig. 16. The panel (a) shows a training data set with optimal decision boundary. The panel (b) shows the decision boundary that was estimated by AdaBoost with the training data, while the panel (c) shows that of the local AdaBoost. The panel (d) shows plots of error rates on the training data set and the test data set against step. Both data sets consist of 300 samples.

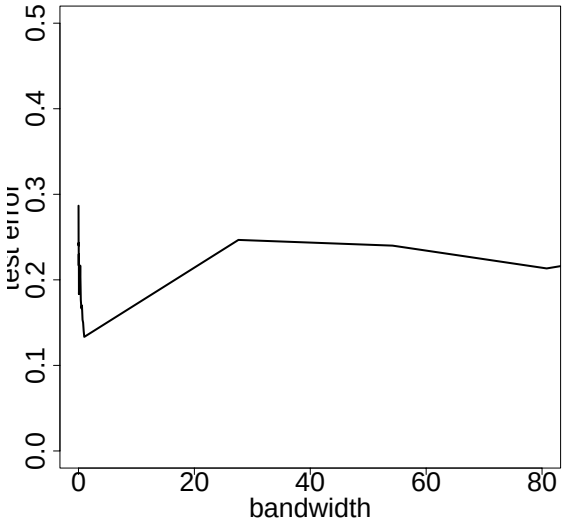
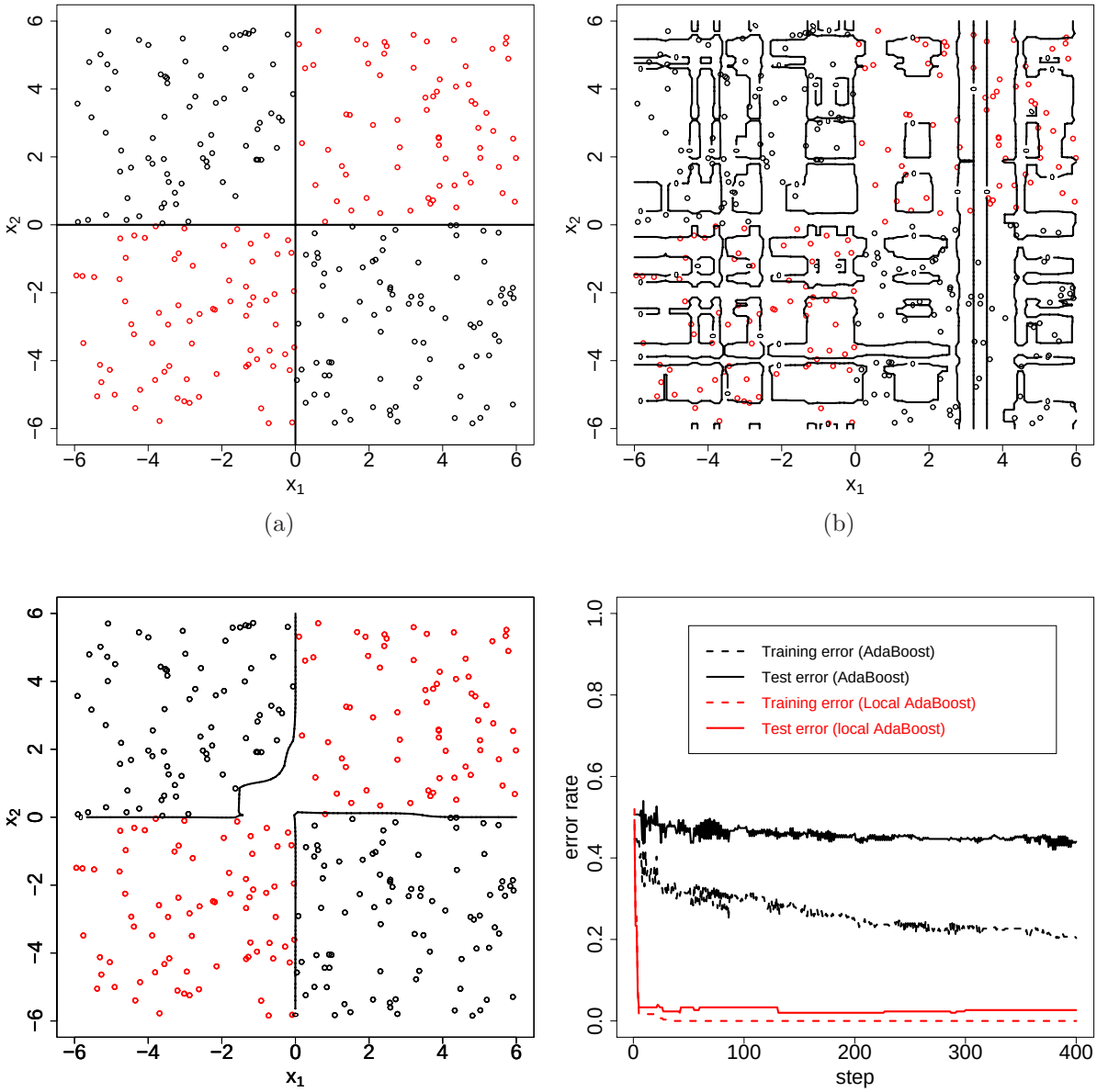


Figure 21: Plot of test error of the local AdaBoost with $\mathcal{K}_*$ at $T = 400$ for various bandwidth h in XOR problem with $\rho = 0.1$ (Fig. 16).



(a) (b) (c) (d)

Figure 22: XOR data: Samples are generated in XOR setting with $\rho = 0$, which was explained in Section 4.2.2 and Fig. 16. The panel (a) shows a training data set with optimal decision boundary. The panel (b) shows the decision boundary that was estimated by AdaBoost, while the panel (c) shows that of the local AdaBoost. The panel (d) shows plots of error rates on the training data set and the test data set against step. Both data sets consist of 300 samples.

A few more examples where $P(Y = y \mid x) \notin \mathcal{P}(\mathcal{M})$ and the local AdaBoost improved

prediction performance (Case (a) in Table 7) are demonstrated in Figs. 23-25.

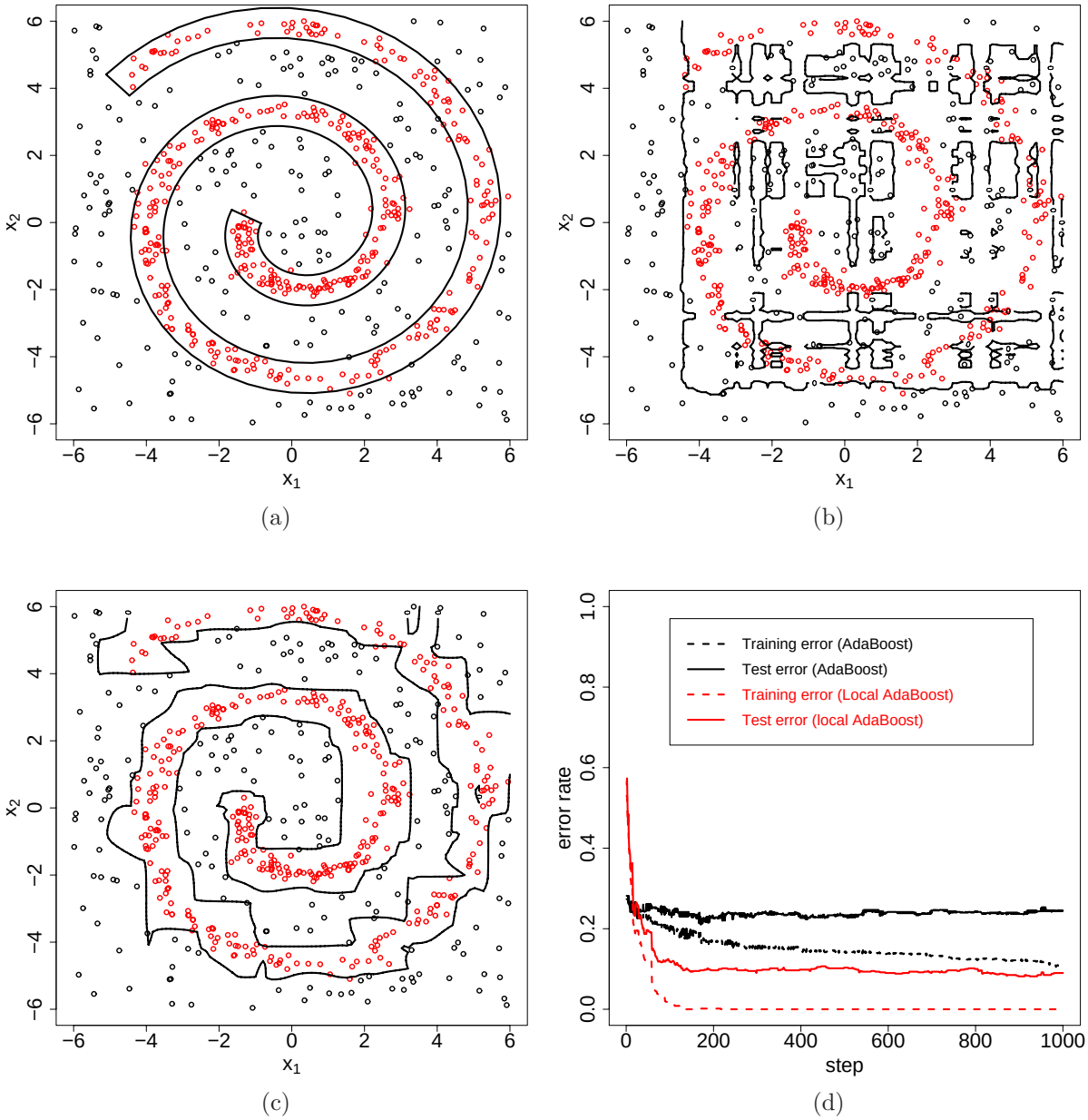


Figure 23: Spiral data: positive samples are distributed inside a spiral tube, while negative samples are distributed out side of the tube. The panel (a) shows a training data set with optimal decision boundary. The panel (b) shows the decision boundary that was estimated by AdaBoost, while the panel (c) shows that of the local AdaBoost. The panel (d) shows plots of error rates on the training data set and the test data set against step. Both data sets consist of 600 samples.

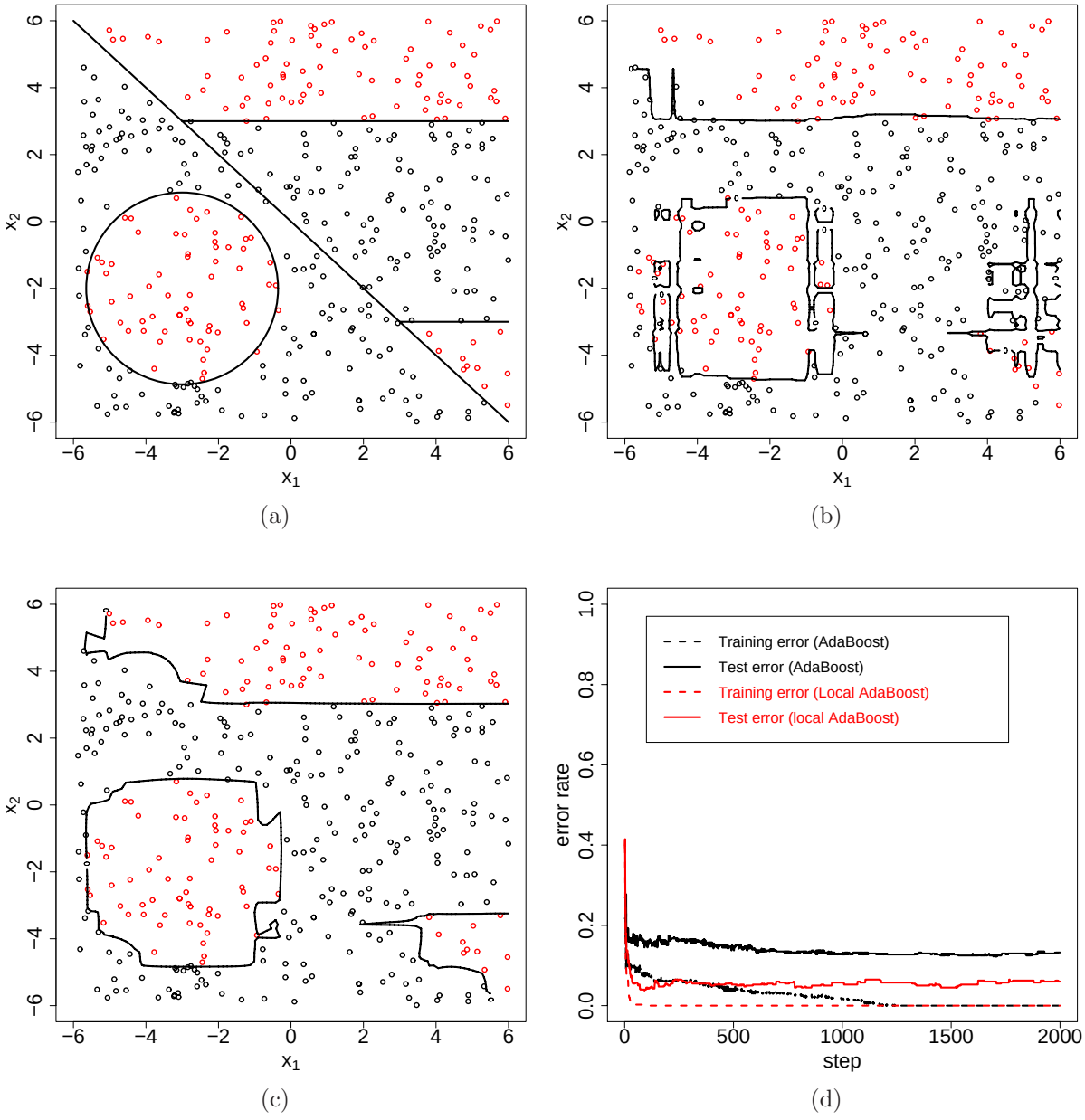


Figure 24: Island1 data: positive samples are distributed inside several diagrams, while negative samples are distributed out side of them. The panel (a) shows a training data set with optimal decision boundary. The panel (b) shows the decision boundary that was estimated by AdaBoost, while the panel (c) shows that of the local AdaBoost. The panel (d) shows plots of error rates on the training data set and the test data set against step. Both data sets consist of 400 samples.

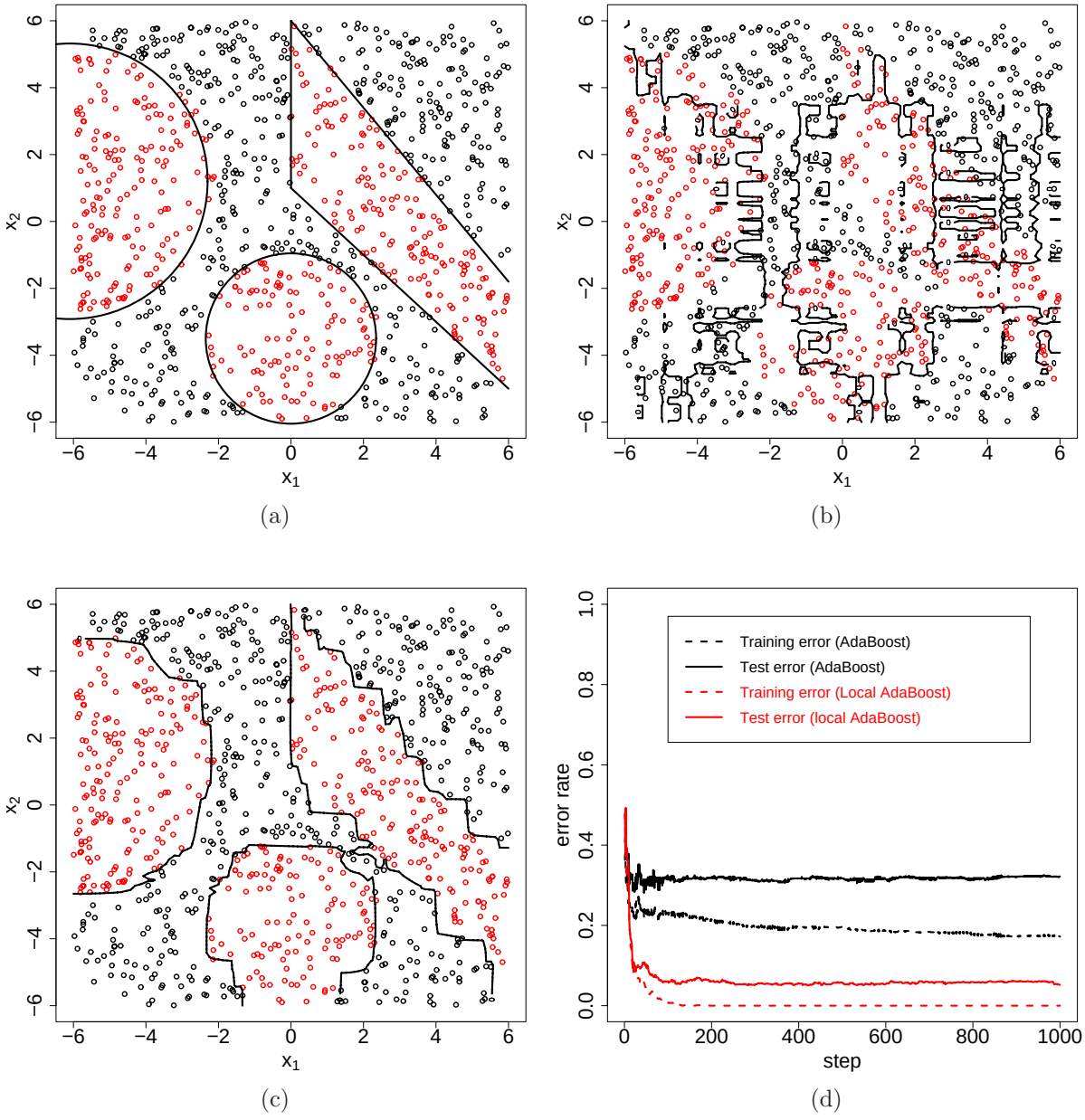


Figure 25: Island2 data: positive samples are distributed inside several diagrams, while negative samples are distributed out side of them. The panel (a) shows a training data set with optimal decision boundary. The panel (b) shows the decision boundary that was estimated by AdaBoost, while the panel (c) shows that of the local AdaBoost. The panel (d) shows plots of error rates on the training data set and the test data set against step. Both data sets consist of 1000 samples.

Finally, we demonstrate that the local AdaBoost.M2 also improves the performance

of AdaBoost.M2. A complicated multiclass classification problem is shown in Fig. 26. We use decision stumps for multiclass case, as described in Section 2.4.1. It is also seen in Fig. 26 that the local AdaBoost performs significantly better than AdaBoost.

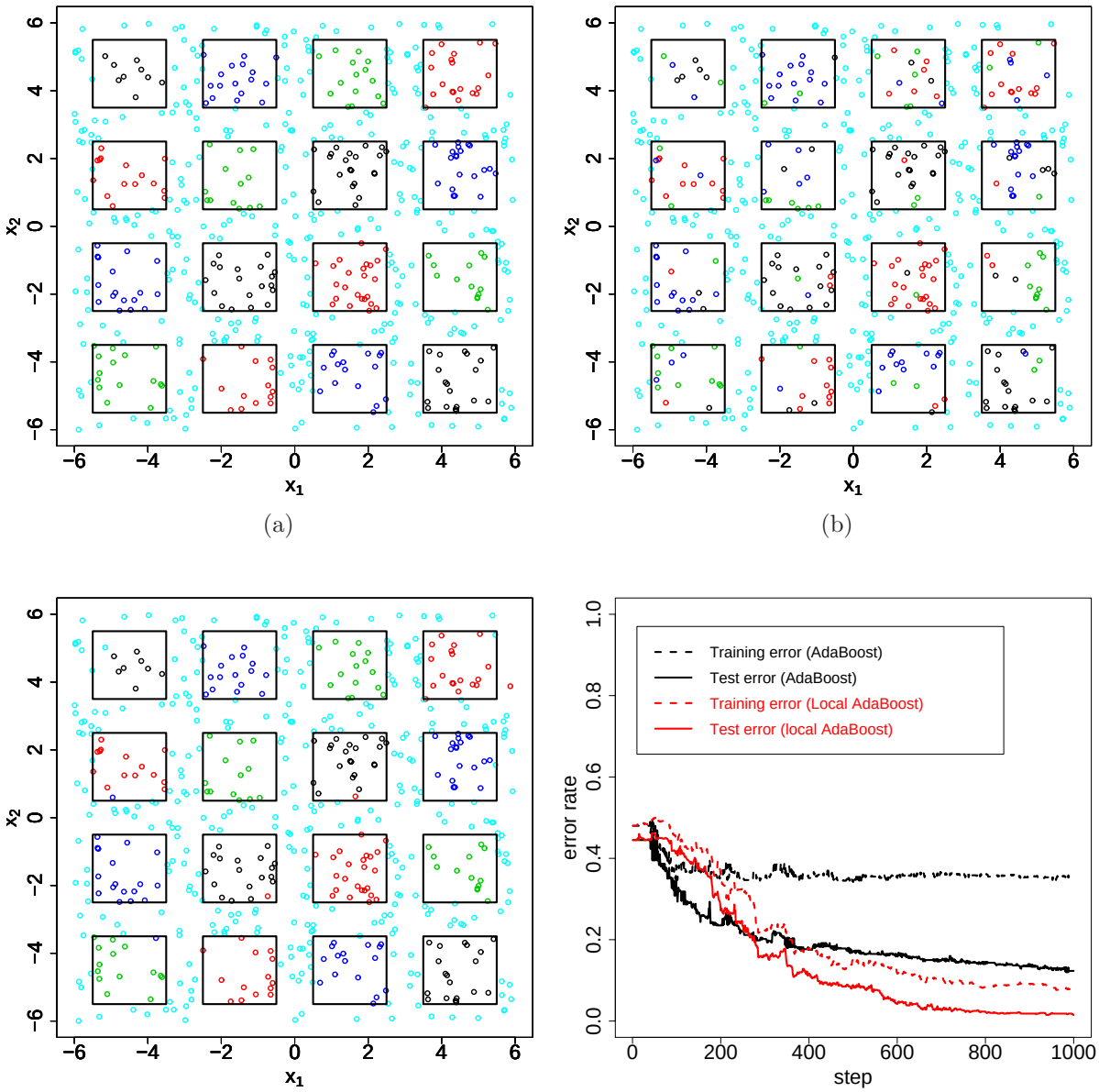


Figure 26: Check data: this figure illustrates a five-class classification problem. Each color corresponds to each class. The panel (a) shows a training data set with optimal decision boundary. The panel (b) shows the predicted class of each sample by AdaBoost with the training data, while the panel (c) shows that by the the local AdaBoost. The panel (d) shows plots of error rates on the training data set and the test data set against step. Both data sets consist of 1000 samples.

4.4 Discussion

We proposed a boosting method for local learning based on an idea that is similar to that of the local likelihood method. There are many cases where the usual boosting with widely-used base classifiers performs poorly because of poor approximation ability. In such cases, one of the simplest strategies is to use more complicated base classifiers that have a possibly larger VC dimension. The use of such base classifiers increases not only computational cost but also the probability of misclassification (generalization error). In statistics, the local likelihood methods are known to improve the approximation ability of a parametric model. However, the local likelihood methods have several difficulties. They require high computational cost and do not construct a single discriminant function. We proposed the local boosting by introducing kernel localization that is slightly different from that of the conventional local likelihood method. The local boosting overcomes such difficulties. Following the discussion in Lugosi and Vayatis (2004), we proved the theorem stating that the local boosting has the Bayes risk consistency under several conditions. Inspection of the proof of this theorem elucidates the property of the local boosting from the view of the estimation error and the approximation error. Lugosi and Vayatis (2004) derived the distribution-free, nonasymptotical, probabilistic upperbound of the estimation error of the usual boosting with respect to V (VC dimension of class of base classifiers). The estimation error of the local boosting has the same upperbound except the increase β in V. The increase, β, depends on the number of kernel center candidates, N since $\beta = \ln N / \ln n$. One interpretation is that localization increases model complexity by β. This is not a steep increase in V against N. If we use more complicated base classifiers, V itself directly increases by an integer. Compared to this case, $\beta = \ln N / \ln n$ is a small increase in general. In any case, the estimation errors of the usual boosting and the local boosting decrease to zero in the asymptotical case ($n \to \infty$) if the class of base classifiers has a finite VC dimension. Thus, the Bayes risk consistency of both boosting methods depends on whether their approximation errors decrease to zero or not. Even if we have sufficient training data, the usual boosting with, for example, $\phi = \exp$ has a large approximation error if $\mathcal{P}(\mathcal{M})$ is a misspecified model. The local boosting may reduce the approximation error at the cost of increasing the estimation error. Thus, the local boosting may have the Bayes risk consistency in wider situations than those of the usual boosting. In actual situations ($n < \infty$), however, the increase in the estimation error is

not a small issue in some cases. Specifically, the local boosting performs relatively poorly in the situation where the approximation error of the usual boosting may decrease to zero because of this increase. The discussion in Section 4.2.2 suggests that the localizing factor $(h,\ \mathcal{K})$ should be selected such that h is as large as possible and that $\mathcal{K}$ is as sparse as possible under the restriction that $\mathcal{P}(\mathcal{M}_{\mathcal{K}})$ includes the underlying distribution $P(Y=y\,|\,x)$. However, in general, selecting a localizing factor satisfying such a restriction without any prior knowledge is difficult. One practical selection of $\mathcal{K}$ is $\mathcal{K}_*$. When n approaches infinity, $\mathcal{K}_*$ covers $\mathrm{supp}(X)$ densely, and then, we may select a small h for decreasing the approximation error. In addition, β is equal to one in this case. It is the least increase in the VC dimension compared to uses of other types of base classifiers. Several simulations illustrate the advantage of the local boosting using $\mathcal{K}_*$.

Parameter λ should be selected by cross validation or trial-and-error. The discussion in Section 4.2.2 indicates that λ should satisfy the following three conditions. First, $\lambda \to \infty$ and $\lambda\phi'(\lambda)\sqrt{\ln n/n} \to 0$ as $n \to \infty$. This condition is necessary for guaranteeing that the upperbound of the estimation error decreases to zero as $n \to \infty$. When $\phi = \exp$, a candidate of $\{\lambda_n\}$ satisfying this condition is $\ln n^{\rho}$ for any $0 < \rho < 1/2$. Second, $\lambda > \max_{x\in\mathcal{X}} F^*(x)$ is the necessary condition for reducing the approximation error. However, satisfying this condition is difficult in general because F^* is unknown. Third, λ should be as small as possible. This condition follows from the finding that large λ may increase the estimation error as seen in Eq. (56). Finding the optimal λ satisfying all the above conditions without any prior knowledge seems difficult. Therefore, we need to determine λ by trial-and-error or cross validation in general.

5 Concluding remarks

We study two topics about booting method. First, we applied AdaBoost with decision stumps to the shark bycatch data from the Eastern Pacific Ocean tuna purse-seine fishery. Compared to the logistic GAM, which has been widely-used in fisheries data analysis, the prediction performance of AdaBoost was stable. In addition, AdaBoost manages many features without any preprocessing. We also presented a graphical tool, *score plot*, to explore the relationship between label and each feature. We may obtain more stable score plots from AdaBoost than from the logistic GAM.

One of the keys of these favorable properties of AdaBoost is the use of decision stumps.

not a small issue in some cases. Specifically, the local boosting performs relatively poorly in the situation where the approximation error of the usual boosting may decrease to zero because of this increase. The discussion in Section 4.2.2 suggests that the localizing factor $(h,\ \mathcal{K})$ should be selected such that h is as large as possible and that $\mathcal{K}$ is as sparse as possible under the restriction that $\mathcal{P}(\mathcal{M}_{\mathcal{K}})$ includes the underlying distribution $P(Y=y\,|\,x)$. However, in general, selecting a localizing factor satisfying such a restriction without any prior knowledge is difficult. One practical selection of $\mathcal{K}$ is $\mathcal{K}_*$. When n approaches infinity, $\mathcal{K}_*$ covers $\mathrm{supp}(X)$ densely, and then, we may select a small h for decreasing the approximation error. In addition, β is equal to one in this case. It is the least increase in the VC dimension compared to uses of other types of base classifiers. Several simulations illustrate the advantage of the local boosting using $\mathcal{K}_*$.

Parameter λ should be selected by cross validation or trial-and-error. The discussion in Section 4.2.2 indicates that λ should satisfy the following three conditions. First, $\lambda \to \infty$ and $\lambda\phi'(\lambda)\sqrt{\ln n/n} \to 0$ as $n \to \infty$. This condition is necessary for guaranteeing that the upperbound of the estimation error decreases to zero as $n \to \infty$. When $\phi = \exp$, a candidate of $\{\lambda_n\}$ satisfying this condition is $\ln n^{\rho}$ for any $0 < \rho < 1/2$. Second, $\lambda > \max_{x\in\mathcal{X}} F^*(x)$ is the necessary condition for reducing the approximation error. However, satisfying this condition is difficult in general because F^* is unknown. Third, λ should be as small as possible. This condition follows from the finding that large λ may increase the estimation error as seen in Eq. (56). Finding the optimal λ satisfying all the above conditions without any prior knowledge seems difficult. Therefore, we need to determine λ by trial-and-error or cross validation in general.

5 Concluding remarks

We study two topics about booting method. First, we applied AdaBoost with decision stumps to the shark bycatch data from the Eastern Pacific Ocean tuna purse-seine fishery. Compared to the logistic GAM, which has been widely-used in fisheries data analysis, the prediction performance of AdaBoost was stable. In addition, AdaBoost manages many features without any preprocessing. We also presented a graphical tool, *score plot*, to explore the relationship between label and each feature. We may obtain more stable score plots from AdaBoost than from the logistic GAM.

One of the keys of these favorable properties of AdaBoost is the use of decision stumps.

Note that Friedman et al. (2000) pointed out the close relationship between AdaBoost and GAM. The important difference between them comes from their models. AdaBoost uses the model consisting of linear combination of weak classifiers, while GAM uses smoothing splines. Decision stumps are weak classifiers and also are stable (or hard) classifiers with respect to its shape. Therefore, decision stumps obviously play an important role on the stability of AdaBoost.

However, we found many examples where boosting method with decision stumps performs poorly because of the shortage of approximation ability. We proposed a new boosting method, *local boosting*, which attains the sufficient approximation ability even with decision stumps. The local boosting was derived from an idea of the local likelihood approach. The local boosting includes a simple device to overcome the computational difficulties. We proved its Bayes risk consistency under certain conditions. Several theoretical inspections indicate that the local boosting improves the approximation error of usual boosting at cost of the slight increase of the estimation error. Some simulations supported the theoretical results.

Finally, we remark that the local AdaBoost did not improve the original AdaBoost much with respect to test error in the shark bycatch problem. We applied the local AdaBoost with decision stumps to the shark bycatch data. We apply the localization to only 'date', 'latitude' and 'longitude'. As a result, the test error of the local AdaBoost was less than the results in Section 3.3 by approximately 1(%). Therefore, the reason why AdaBoost or GAM test errors are not small may be less related to the approximation ability.

Acknowledgement

I appreciate deeply Prof. S. Eguchi (ISM[3]) for supervising me in the whole process of this research. I also thank C. E. Lennert-Cody (IATTC) and M. Minami (ISM) for many advices and cooperations on the analysis of shark bycatch data. I thank the IATTC for providing the shark bycatch data. I also appreciate Prof. S. Kuriki and H. Fujisawa for supporting me. Prof. Ikeda gave me useful comments, which improved this thesis much. I thank him. My family supported me through the whole process of the doctor course. I could not complete this dissertation without their support. Members in the student room

[3]Institute of Statistical Mathematics

in ISM also supported me very much. I appreciate all of them. Finally, I thank myself. He led me to this area, he did not give up this hard course and he helped me getting Ph. D.

A Bayes classifier attains the minimum probability of misclassification

Proposition 41. *Let* $g^*(x) = \text{argmax}_{y\in\mathcal{Y}} P(Y = y \mid x)$ *be the Bayes classifier. Then,*

$$\inf_{g\in\mathcal{G}} L(g) = L(g^*).$$

Proof. The proof appears separately for binary and multiclass case.

Binary case Let $g : \mathcal{X} \to \{1, -1\}$ be an arbitrary classifier.

$$\begin{aligned} L(g) &= P(g(X) \neq Y) = E[I(g(X) \neq Y)] \\ &= E[\eta(X)I(g(X \neq 1)) + (1 - \eta(X))I(g(X) \neq -1)] \\ &= E[\eta(X)(1 - I(g(X = 1))) + (1 - \eta(X))I(g(X) = 1)] \\ &= E[\eta(X) + I(g(X) = 1)(1 - 2\eta(X))] \\ &= P(Y = 1) + E[I(g(X) = 1)(1 - 2\eta(X))] \end{aligned}$$

The first term in the last equality is not dependent on g. The second term in the last equality is minimized when

$$g(x) = \begin{cases} 1 & \text{if } (1 - 2\eta(x) <= 0) \\ -1 & \text{otherwise} \end{cases}$$

This directly implies that $g(x)$ is the Bayes classifier $g^*(x)$.

Multiclass case Let $\mathcal{Y}$ be $\{1, 2, \cdots, G\}$ and $g : \mathcal{X} \to \mathcal{Y}$ be an arbitrary classifier.

$$\begin{aligned} L(g) &= P(g(X) \neq Y) = E[I(g(X) \neq Y)] \\ &= E\Big[\sum_{y=1}^{G} I(g(X) \neq y)P(Y = y|x)\Big] \\ &= E\Big[\sum_{y=1}^{G}(1 - I(g(X) = y))P(Y = y|x)\Big] \\ &= 1 - E\Big[\sum_{y=1}^{G} I(g(X) = y))P(Y = y|x)\Big] \end{aligned}$$

Thus, $\text{argmin}_g L(g) = \text{argmax}_g E[\sum_{y=1}^{G} I(g(X) = y)P(Y = y|x)]$. Clearly, $g(x) = \text{argmax}_{y\in\mathcal{Y}} P(Y = y|x)$ minimizes L and is just the Bayes classifier.

□

B Center limit theorem

Restricted to this section, we denote an imaginary unit by j, *i.e.*, $j^2 = -1$.

Theorem 42 (Center limit theorem). *Let $\{X_i\}_{i=1}^n$ be a sequence of i.i.d. M-dimensional random variable with mean μ and covariance Σ. Define $S_n = \frac{1}{n}\sum_{j=1}^n \sqrt{n}\Sigma^{-1/2}(X_i - \mu)$. Then, as $n \to \infty$,*

$$P(S_n \leq x) \to \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{t^2}{2}} dt.$$

Proof. Let $Z_i = \sqrt{n}\Sigma^{-1/2}(X_i - \mu)$. Then, $S_n = \frac{1}{\sqrt{n}} Z_i$. Let θ be an arbitrary M-dimensional vector. Considering that $E[Z_i] = 0$ and $E(Z_i Z_i^T) = I_M$ for any i, the characteristic function of S_n is obtained as follows.

$$\begin{aligned}
E[\exp(j\theta^T S_n)] &= \Pi_{i=1}^n E[\exp(j\theta^T Z_i/\sqrt{n})] \\
&= \{E[\exp(j\theta^T X/\sqrt{n})]\}^n \\
&= \left\{ E\left[1 + \frac{i\theta^T Z}{\sqrt{n}} - \frac{\theta^T Z Z^T \theta}{2n} + o\left(\frac{\|\theta\|^2}{\sqrt{n}} \right) \right] \right\}^n \\
&= \left\{ 1 - \frac{\|\theta\|^2}{2n} + o\left(\frac{\|\theta\|^2}{\sqrt{n}} \right) \right\}^n
\end{aligned}$$

where Z is an independent copy of Z_i. Due to the property of exponential function exp, this converses to $\exp(-\|\theta\|^2/2)$ as $n \to \infty$ for any fixed t. The statement follows from the Levy's continuous theorem. □

C An equality on exponential function

Lemma 43. *Let $n = 1, 2, \cdots, \infty$ and x be a real positive variable. For any n,*

$$\left(1 + \frac{x}{n}\right)^n \geq e^x.$$

Proof. By Taylor-expansion, we have

$$\begin{aligned}
e^x - (1 + \frac{x}{n})^n &= \sum_{k=0}^{\infty} \frac{x^k}{k!} - \sum_{k=0}^{n} \left(\begin{array}{c} n \\ k \end{array} \right) \left(\frac{x}{n}\right)^k \\
&= \sum_{k=0}^{\infty} \frac{x^k}{k!} - \sum_{k=0}^{n} \frac{n!}{n^k k!(n-k)!} x^k
\end{aligned}$$

For each $k \leq n$,

$$\begin{aligned}
\frac{x^k}{k!} - \frac{n! x^k}{n^k k!(n-k)!} &= \frac{x^k}{k!}\left(1 - \frac{n!}{n^k(n-k!)}\right) \\
&= \frac{x^k}{k!}\left(1 - \frac{n}{n} \cdot \frac{n-1}{n} \cdot \cdots \cdot \frac{n-k+1}{n} \cdot \frac{n-k}{n-k} \cdot \frac{n-k-1}{n-k-1} \cdots \frac{2}{2} \cdot \frac{1}{1}\right) \\
&= \frac{x^k}{k!}\left(1 - \frac{n}{n} \cdot \frac{n-1}{n} \cdot \cdots \cdot \frac{n-k+1}{n}\right) \\
&\geq 0.
\end{aligned}$$

□

References

Bartlett, P. L., I., J. M., McAuliffe, J. D., 2003. Convexity, classification, and risk bounds. `http://stst-www.berkeley.edu/tech-reports/638.pdf`.

Bartlett, P. L., Mendelson, S., 2002. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research* 3, 463–482.

Bayliff, W. H., 2001. Organization, functions and achievements of the inter-american tropical tuna commission. Special Report 13. IATTC, La Jolla, CA, pp. 122.

Bigelow, K., Boggs, C., He, X., 1999. Environmental effects on swordfish and blue shark catch rates in the us north pacific longline fishery. *Fisheries Oceanography* 8, 178–198.

Breiman, L., 1996a. Bagging predictors. *Machine Learning* 26, 123–140.

Breiman, L., 1996b. The heuristics of instability in model selection. *Annals of Statistics* 24, 2350–2383.

Breiman, L., 1998. Arcing classifiers. *Annals of Statistics* 26 (3), 801–849.

Breiman, L., 1999. Prediction games and arcing algorithms. *Neural Computation* 11 (7), 1493–1518.

Breiman, L., 2001. Random forests. Machine Learning 45 (1), 5–32.

Breiman, L., Friedman, J., Olshen, R., Stone, C., 1984. Classification and Regression Trees. *Wadsworth, Belmont.*

Chernoff, H., 1952. A measure of asymptotic efficiency of test of a hypothesis based on the sum of observations. *Annals of mathematical statistics* 23, 493–507.

Collins, M., Schapire, R., Singer, Y., 2002. Logistic regression, adaboost and bregman distances. In *Machine Learning*, 253–285.

Devroye, L., Györfi, L., Lugosi, G., 1996. A probabilistic theory of pattern recognition. Springer-Verlag, New York.

Devroye, L., Lugosi, G., 2001. Combinatorial Methods in Density Estimation. Springer-Verlag, New York.

Efron, B., Tibshirani, R., 1993. An introduction to the Bootstrap. *Chapman & Hall, New York*.

Eguchi, S., Copas, J., 1998. A class of local likelihood methods and near-parametric asymptotics. *Journal of Royal Statistical Society B* 60, 709–724.

Eguchi, S., Kim, T.-Y., Park, B. U., 2003. Local likelihood method and theory for a bridge between parametric and nonparametric regression. *Journal of Nonparametric Statistics* 15, 665–683.

Fan, J., Farmen, M., Gijbels, I., 1998. Local maximum likelihood estimation and inference. *Journal of Royal Statistical Society B*.

Fan, J., Gijbels, I., 1995. Data-driven bandwidth selection in local polynomial fitting: variable bandwidth and spatial adaption. *Journal of Royal Statistical Society B* 57, 371–394.

Fan, J., Gijbels, I., 1996. Local polynomial modelling and its applications. Chapman & Hall, London.

Freund, Y., Schapire, R. E., 1997. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55 (1), 119–139.

Friedman, J., 1997. On bias, variance, 0/1-loss, and the curse of dimensionality. *Journal of knowledge discovery and data mining* 1, 55–77.

Friedman, J. H., Hastie, T., Tibshirani, R., 2000. Additive logistic regression: A statistical view of boosting. Annals of Statistics 28, 337–407.

Hastie, T., Tibshirani, R., Friedman, J., 2001. The elements of statistical learning: Data mining, inference and prediction. Springer-Verlag.

Hastie, T. J., Tibshirani, R. J., 1990. Generalized Additive Models. Chapman & Hall, London.

Hjort, N. L., Jones, M. C., 1996. Locally parametric nonparametric density estimation. *Annals of Statistics* 24 (4), 1619–1647.

Hoeffding, W., 1963. Probability inequalities for sums of bounded random variables. *Journal of the american statistical association* 58, 13–30.

Höffgen, K. U., Simon, H. U., van Horn, K. S., 1995. Robust trainability of single neurons. Journal of Computer and System Sciences 50, 114–125.

IATTC, 2004. Annual report of the inter-american tropical tuna commission, 2002. Inter-American Tropical Tuna Commission, La Jolla, CA.

Jiang, W., 2004. Process consistency for adaboost. *Annals of Statistics* 32, 13–29.

Kawakita, M., Cleridy, E., Minami, M., Eguchi, S., 2005. An introduction to the predictive technique adaboost with a comparison to generalizaed additive models. *Fishries Research* 73, 328–343.

Kawakita, M., Eguchi, S., 2005. Boosting method for local learning in statistical pattern recognition. *In revision*.

Kearns, M., Valiant, L. G., 1988. Learning boolean formulae or finite automata is as hard as factoring. Technical Report TR-14-88, Harvard University Aiken Computation Laboratory.

Kohavi, R., Wolpert, D. H., 1996. Bias plus variance decomposition for zero-one loss functions. In *Machine Learning: Proceedings of the Thirteenth International Conference* (L. Saitta, ed.) Morgan Kaufmann, San Francisco, 275–283.

Koltchinskii, V., Panchenko, D., 2002. Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics* 30 (1), 1–30.

Kong, E., Dietterich, T. G., 1995. Error-correcting output coding corrects bias and variance. In *proceeding of the Twelfth International Conference on Machine Learning* (A.Prieditis and S. Russell, eds.) Morgan Kaufmann, San Francisco, 313–321.

Lebanon, G., Lafferty, J., 2002. Boosting and maximum likelihood for exponential models. *Advances in Neural Information Processing Systems* 14.

Ledoux, M., Talagrand, M., 1991. Probability in Banach Spaces: *isoperimetry and processes*. Springer-Verlag, New York.

Lo, N. C., Jacobson, L. D., Squire, J. L., 1992. Indices of relative abundance from fish spotter data based on delta-lognormal models. *Canadian Journal of Fisheries and Aquatic Sciences* 49, 2515–2526.

Lugosi, G., Vayatis, N., 2004. On the Bayes-risk consistency of regularized boosting methods. *Annals of Statistics* 32, 30–55.

Mason, L., Baxter, J., Bartlett, P. L., Frean, M., 1999. Functional gradient techniques for combining hypotheses. In A. J. Smola, P. L. Bartlett, B. Schölkopf, and D. Schuurmans editors, *Advances in Large Margin Classifiers*, MIT Press, Cambridge, MA, 221–247.

McCullagh, P., Nelder, J. A., 1989. Generalized Linear Models. Chapman & Hall.

McDiarmid, C., 1989. On the method of bounded differences. In *Surveys in Combinatorics*. Cambridge University Press.

Morgan, N. N., Sonquist, J. A., 1963. Problems in the analysis of survey data, and a proposal. *Journal of the american statistical association*, 415–434.

Murata, N., Takenouchi, T., Kanamori, T., Eguchi, S., 2004. Information geometry of U-Boost and Bregman divergence. *Neural Computation* 16, 1437–1481.

Okamoto, M., 1958. Some inequalities relating to the partial sum of binomial probabilities. *Annals of the Institute of Statistical Mathematics* 10, 29–35.

Oshitani, S., Nakauo, H., Tanaka, S., 2003. Age and growth of the silky shark *carcharhinus falciformis* from the pacific ocean. Fisheries Science 69, 456–464.

Punt, A., Walker, T., Taylor, B., Pribac, F., 2000. Standardization of catch and effort data in a spatially-structured shark fishery. *Fisheries Research* 45, 129–145.

Quinlan, R., 1993. C4.5: Programs for machine learning. *Morgan Kaufmann, San Mateo*.

Roweis, S., Saul, L., 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290 (5500), 2323–2326.

Ryan, T. P., 1997. Modern Regression Methods. *Wiley Inter-Science*.

Sauer, N., 1972. On the density of families of sets. *Journal of combinatorial theory Series A* 13, 145–147.

Schapire, R., 1990. The strength of the weak learnability. *Machine Learning* 5, 197–227.

Schapire, R. E., Freund, Y., Bartlett, P., Lee, W. S., 1998. Boosting the margin: a new explantion for the effectiveness of voting methods. *Annals of Statistics* 26, 1651–1686.

Swartzman, G., Huang, C., Kaluzny, S., 1992. Spatial analysis of bering sea groundfish survey data using generalized additive models. *Canadian Journal of Fisheries and Aquatic Sciences* 49, 1366–1378.

Taquet, M., Gaertner, J.-C., Bertrand, J., 1997. Typologie de la flottille chalutière de sète: formalisation par une méthode de segmentation. *Aquatic Living Resources* 10, 137–148.

Tibshirani, R., 1996. Bias, variance, and prediction error for classification rules. *Technical Report, Dept. Statistics, Univ. Toronto*.

Tserpes, G., Peristeraki, P., Potamias, G., Tsimenides, N., 1999. Species distribution in the southern aegean sea based on bottom-trawl surveys. *Aquatic Living Resources* 12 (3), 167–175.

van der Vaart, A. W., Wellner, J. A., 1996. Weak convergence and Empirical processes. With Applications to Statistics. Springer-Verlag, New York.

Vapnik, V. N., 1982. Estimation of dependences based on empirical data. Springer-Verlag, New York.

Vincent, P., Bengio, Y., 2003. Locally weighted full covariance gaussian density estimation. Technical report 1240.

Viola, P., Jones, M., December 2001. Fast and robust classification using asymmetric adaboost and a detector cascade. *Neural Information Processing Systems* 14.

Walsh, W., Kleiber, P., 2001. Generalized additive models and regression tree analyses of blue shark (prionace glauca) catch rates by the hawaii-based commercial longline fishery. *Fisheries Research* 53, 115–131.

Walsh, W. A., Kleiber, P., McCracken, W., 2002. Comparison of logbook reports of incidental blue shark catch rates by hawaii-based longline vessels to fishery observer data by application of a generalized additive model. *Fisheries Research* 58, 79–94.

Watters, G. M., 1999. Geographical distributions of effort and catches of tunas by purse-seine vessels in the eastern pacific ocean during 1965-1998. IATTC data Report 10. IATTC, La Jolla, CA.

Wood, S. N., 2000. Modelling and smoothing parameter estimation with multiple quadratic penalties. *Journal of the Royal Statistical Society: Series B* 62, 413–428.

Wood, S. N., 2003. Thin plate regression splines. *Journal of the Royal Statistical Society: Series B* 65 (1), 95–114.

Wood, S. N., 2004. Stable and efficient multiple smoothing parameter estimation for generalized additive models. *Journal of the American Statistical Association* 99 (467), 673–686.

Zhang, T., Yu, B., 2005. Boosting with early stopping: Convergence and consistency. *Annals of statistics* 33, 1538–1579.