Ensemble

김지수 (Jisu KIM)

통계적 기계학습(Statistical Machine Learning), 2025 1학기 (spring)

The lecture note is a minor modification of the lecture notes from Prof. Yongdai Kim's "Statistical Machine Learning", and Prof Larry Wasserman and Ryan Tibshirani's "Statistical Machine Learning". Also, see Section 5, 6, 10 from [?].

1 Review

1.1 Basic Model for Supervised Learning

- Input(입력) / Covariate(설명 변수) : $x \in \mathbb{R}^d$, so $x = (x_1, \dots, x_d)$.
- Output(출력) / Response(반응 변수): $y \in \mathcal{Y}$. If y is categorical, then supervised learning is "classification", and if y is continuous, then supervised learning is "regression".
- Model(모형) :
$$y \approx f(x), \qquad f \in \mathcal{M}.$$
If we include the error ϵ to the model, then it can be also written as
$$y = \phi(f(x), \epsilon).$$
For many cases, we assume additive noise, so
$$y = f(x) + \epsilon.$$
- Assumption(가정): f belongs to a family of functions $\mathcal{M}$. This is the assumption of a model: a model can be still used when the corresponding assumption is not satisfied in your data.
- Loss function(손실 함수): $\ell(y, a)$. A loss function measures the difference between estimated and true values for an instance of data. The most common ones are:
  - square: $\ell(y, a) = (y - a)^2$.
  - $0 - 1$: $\ell(y, a) = I(y \neq a)$.
- Training data(학습 자료): $\mathcal{T} = \{(y_i, x_i), i = 1, \dots, n\}$, where (y_i, x_i) is a sample from a probability distribution P_i. For many cases we assume i.i.d., or x_i's are fixed and y_i's are i.i.d..
- Goal(목적): we want to find f that minimizes the expected prediction error,
$$f^0 = \arg\min_{f \in \mathcal{F}} \mathbb{E}_{(Y,X)\sim P}\left[\ell(Y, f(X))\right].$$
Here, $\mathcal{F}$ can be different from $\mathcal{M}$; $\mathcal{F}$ can be smaller than $\mathcal{M}$.
- Prediction model(예측 모형): f^0 is unknown, so we estimate f^0 by $\hat{f}$ using data. For many cases we minimizes on the empirical prediction error, that is taking the expectation on the empirical distribution $P_n = \frac{1}{n}\sum_{i=1}^n \delta_{(Y_i, X_i)}$.
$$\hat{f} = \arg\min_{f \in \mathcal{F}} \mathbb{E}_{P_n}\left[\ell(Y, f(X))\right] = \arg\min_{f \in \mathcal{F}} \frac{1}{n}\sum_{i=1}^n \ell(Y_i, f(X_i)).$$
- Prediction(예측): if $\hat{f}$ is a predicted function, and x is a new input, then we predict unknown y by $\hat{f}(x)$.

1.2 Bayes Classifier

For the classification so that $y \in \mathcal{Y} = \{0, \dots, K-1\}$, the risk (위험) is

$$R(f) = \mathbb{E}_{(Y,X)\sim P}\left[\ell(Y, f(X))\right].$$

Theorem 1. *The rule f that minimizes $R(f)$ is*

$$f^*(x) = \arg\min_{k=0,\dots,K-1} \sum_{j=0}^{K-1} \ell(j,k)\mathbb{P}\left(Y = j | X = x\right), \tag{1}$$

and in particular when $\ell(y,a) = I(y \neq a)$,

$$f^*(x) = \arg\max_{k=0,\dots,K-1} \mathbb{P}\left(Y = k | X = x\right).$$

This optimal rule f^* is called the *Bayes rule / Bayes classifier* (베이즈분류 / 베이즈모형). The risk $R^* = R(f^*)$ is called the *Bayes risk* (베이즈위험).

2 Introduction

Ensemble (앙상블) is a generic term for methods of constructing many learners (학습기) (eg. classifiers (분류기)) and combining them to make the final strong (i.e. highly accurate) learner (학습기). Examples of ensemble are:

- Bagging (배깅) (Breiman 1996)
- Boosting (부스팅) (Freund and Schapire 1997)
- Random Forest (랜덤포레스트) (Breiman 2004)

Empirically, ensemble (앙상블) methods perform much better than the best single model. It is very simple and effective but there is still a large gap between theory and practice.

3 Partitions and Trees

We begin by reviewing trees. As with nonparametric regression, simple and interpretable classifiers can be derived by partitioning the range of X. Let $\Pi_n = \{A_1, \dots, A_N\}$ be a partition of $\mathcal{X}$. Let A_j be the partition element that contains x. Then

$$\hat{h}(x) = \begin{cases} 1 & \text{if } \sum_{X_i \in A_j} Y_i \geq \sum_{X_i \in A_j}(1 - Y_i), \\ 0 & \text{otherwise.} \end{cases}$$

This is nothing other than the plugin classifier based on the partition regression estimator

$$\hat{f}(x) = \sum_{j=1}^{N} \overline{Y}_j I(x \in A_j)$$

where $\overline{Y}_j = n_j^{-1}\sum_{i=1}^n Y_i I(X_i \in A_j)$ is the average of the Y_i's in A_j and $n_j = \#\{X_i \in A_j\}$. (We define $\overline{Y}_j$ to be 0 if $n_j = 0$.)

Recall from the results on regression that if $m \in H_1(1, L)$ and the binwidth b of a regular partition satisfies $b \asymp n^{-1/(d+2)}$ then

$$\mathbb{E}||\hat{f} - f^0||_P^2 \leq \frac{c}{n^{2/(d+2)}}. \tag{2}$$

We conclude that the corresponding classification risk satisfies $R(\hat{h}) - R(h_*) = O(n^{-1/(d+2)})$.

Regression trees and classification trees (also called decision trees) are partition classifiers where the partition is built recursively. For illustration, suppose there are two covariates, X_1 = age and X_2 = blood pressure. Figure 1 shows a classification tree using these variables.

The tree is used in the following way. If a subject has Age ≥ 50 then we classify it as $Y = 1$. If a subject has Age < 50 then we check his blood pressure. If systolic blood pressure is < 100 then we classify him as $Y = 1$, otherwise we classify him as $Y = 0$. Figure 2 shows the same classifier as a partition of the covariate space.

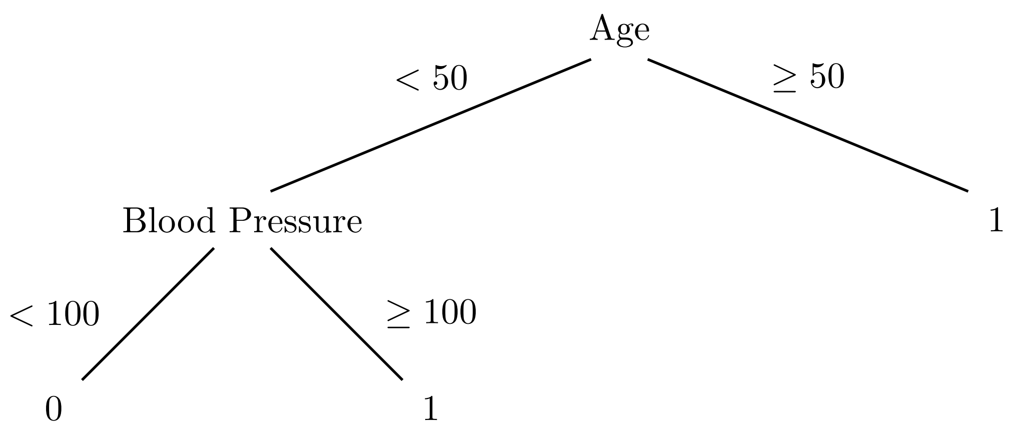

Figure 1: A simple classification tree.

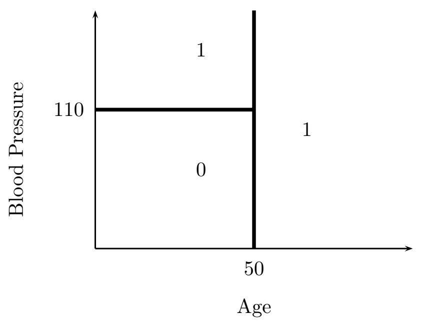

Figure 2: Partition representation of classification tree.

Method	Test Error
Logistic regression	0.23
SVM (Gaussian Kernel)	0.20
Kernel Regression	0.24
Additive Model	0.20
Reduced Additive Model	0.20
11-NN	0.25
Trees	0.20

Table 1: Various methods on the MAGIC data. The reduced additive model is based on using the three most significant variables from the additive model.

Here is how a tree is constructed. First, suppose that there is only a single covariate X. We choose a split point t that divides the real line into two sets $A_1 = (-\infty, t]$ and $A_2 = (t, \infty)$. Let $\overline{Y}_1$ be the mean of the Y_i's in A_1 and let $\overline{Y}_2$ be the mean of the Y_i's in A_2.

For continuous Y (regression), the split is chosen to minimize the training error. For binary Y (classification), the split is chosen to minimize a surrogate for classification error. A common choice is the impurity defined by $I(t) = \sum_{s=1}^{2} \gamma_s$ where

$$\gamma_s = 1 - [\overline{Y}_s^2 + (1 - \overline{Y}_s)^2]. \tag{3}$$

This particular measure of impurity is known as the *Gini index.* If a partition element A_s contains all 0's or all 1's, then $\gamma_s = 0$. Otherwise, $\gamma_s > 0$. We choose the split point t to minimize the impurity. Other indices of impurity besides the Gini index can be used, such as entropy. The reason for using impurity rather than classification error is because impurity is a smooth function and hence is easy to minimize.

Now we continue recursively splitting until some stopping criterion is met. For example, we might stop when every partition element has fewer than n_0 data points, where n_0 is some fixed number. The bottom nodes of the tree are called the *leaves.* Each leaf has an estimate $\hat{f}(x)$ which is the mean of Y_i's in that leaf. For classification, we take $\hat{h}(x) = I(\hat{f}(x) > 1/2)$. When there are several covariates, we choose whichever covariate and split that leads to the lowest impurity.

The result is a piecewise constant estimator that can be represented as a tree.

3.1 Example

The following data are from simulated images of gamma ray events for the Major Atmospheric Gamma-ray Imaging Cherenkov Telescope (MAGIC) in the Canary Islands. The data are from archive.ics.uci.edu/ml/datasets/MAGIC+Gamma+Teles
The telescope studies gamma ray bursts, active galactic nuclei and supernovae remnants. The goal is to predict if an event is real or is background (hadronic shower). There are 11 predictors that are numerical summaries of the images. We randomly selected 400 training points (200 positive and 200 negative) and 1000 test cases (500 positive and 500 negative). The results of various methods are in Table 1. See Figures 3, 4, 5, 6.

4 Bagging

Trees are useful for their simplicity and interpretability. But the prediction error can be reduced by combining many trees. A common approach is called bagging (배깅), which stands for ***B**ootstrap **agg**regat**ing***. The algorithm is as follows. Let $\mathcal{L} = \{(y_i, x_i), i = 1, \ldots, n\}$ be the dataset.

1. Make bootstrap samples:$\{\mathcal{L}^{(b)}, b = 1, \ldots, B\}$ (data sets obtained by with replacement sampling from $\mathcal{L}$).
2. Form predictors: $\{f(x, \mathcal{L}^{(b)}), b = 1, \ldots, B\}$ learned on each bootstrap samples $\mathcal{L}^{(b)}$.
3. If y is numeric, use $f_B(x) = \frac{1}{B}\sum_{j=1}^{B} f(x, \mathcal{L}^{(b)})$ for the predictor
4. If y is categorical (class label), let the $\{f(x, \mathcal{L}^{(b)}), b = 1, \ldots, B\}$ vote to form $f_B(x)$. That is, $f_B(x) = \text{argmax}_j N_j$ where $N_j = \#\{b : f(x, \mathcal{L}^{(b)}) = j\}$.

The performance of a single tree ($\bar{e}_S$: error for a single tree) vs bagging ($\bar{e}_B$: error for bagging) is as follows.

Bagging Classification Trees:

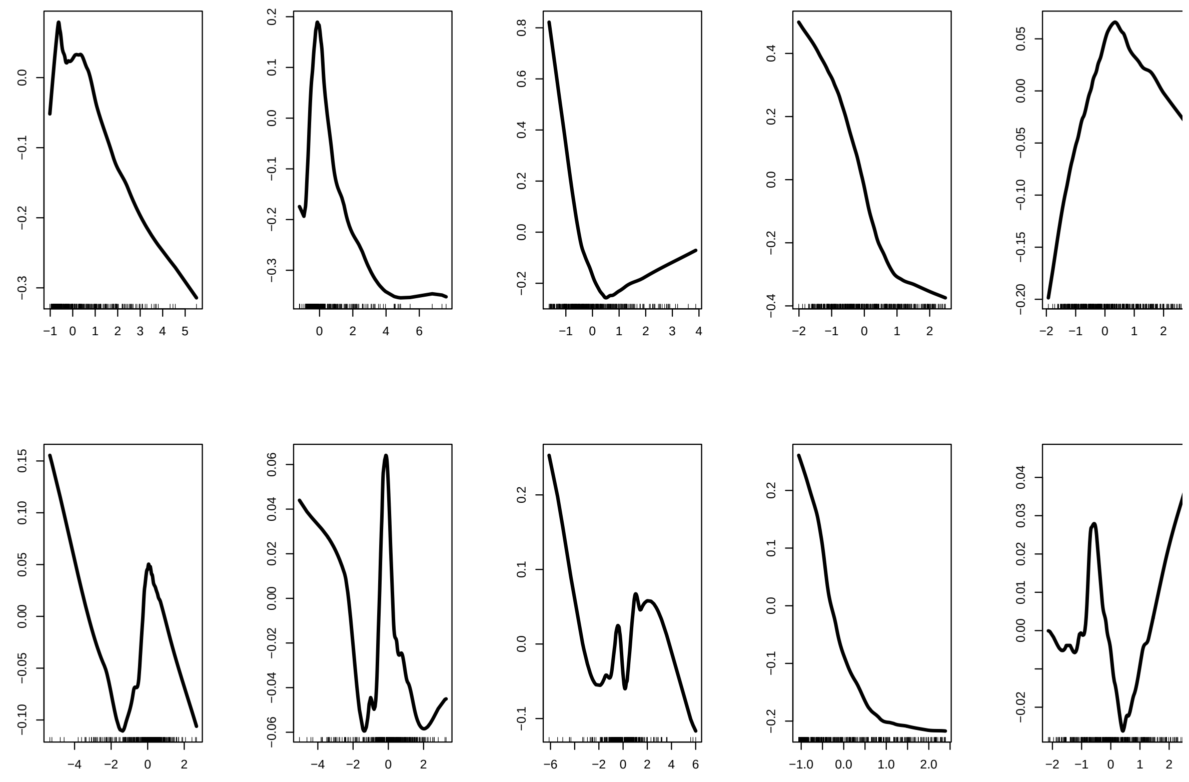
Figure 3: Estimated functions for additive model.

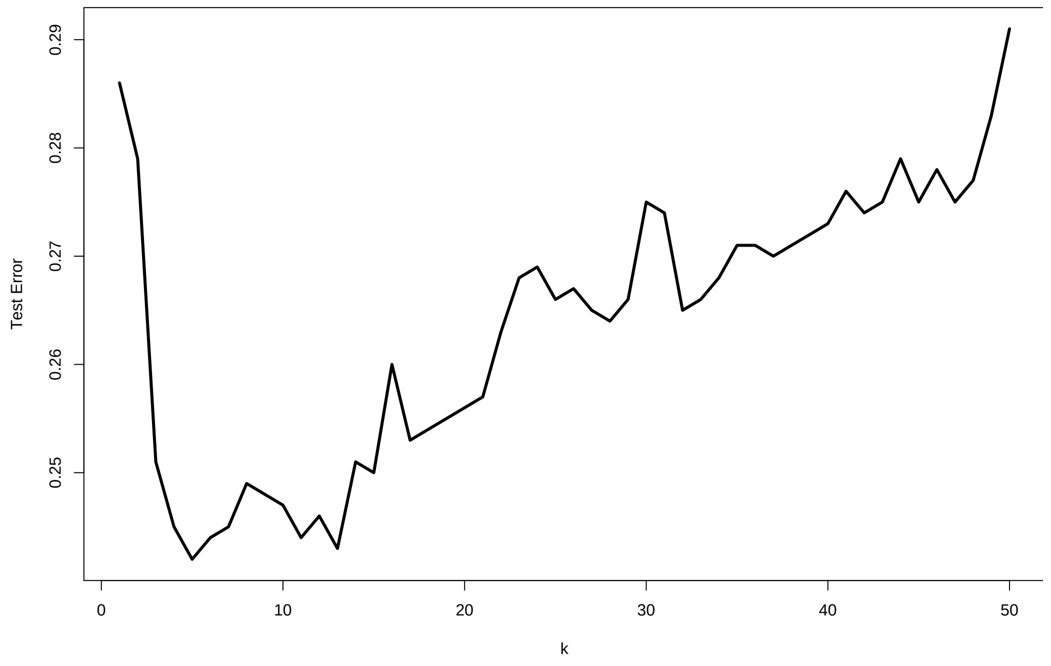

Figure 4: Test error versus k for nearest neighbor estimator.

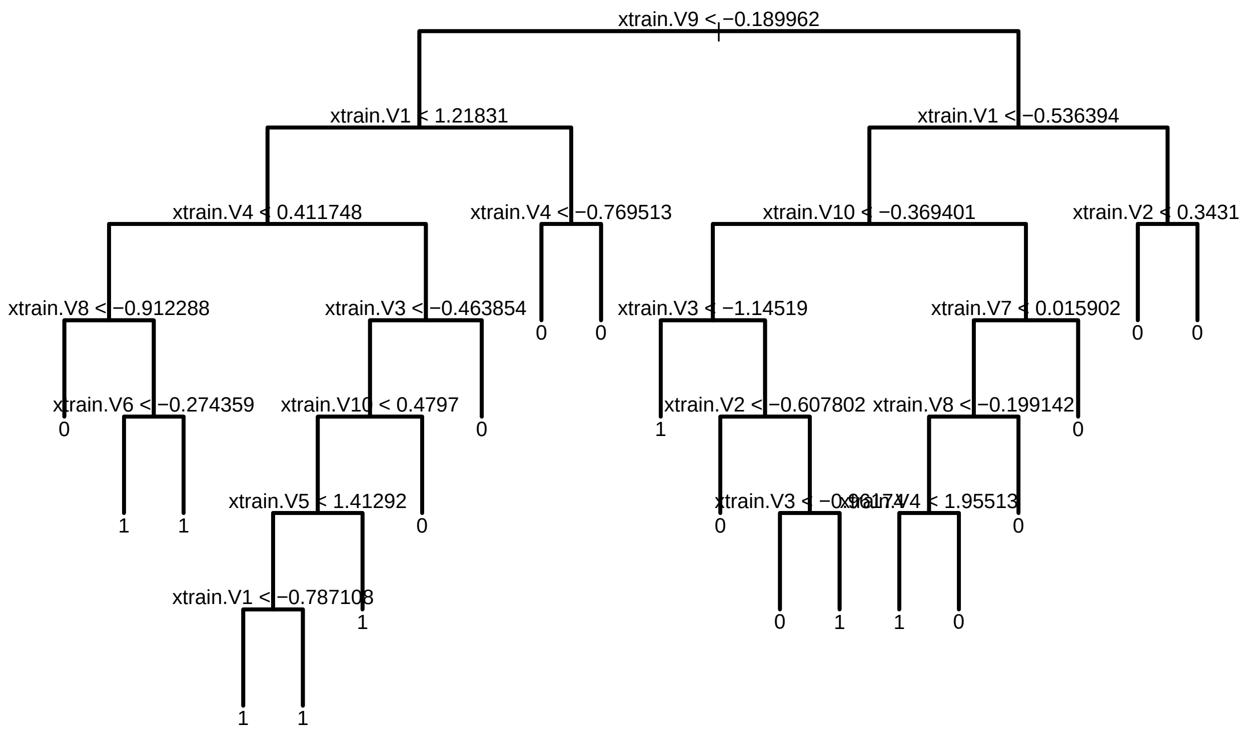

Figure 5: Full tree.

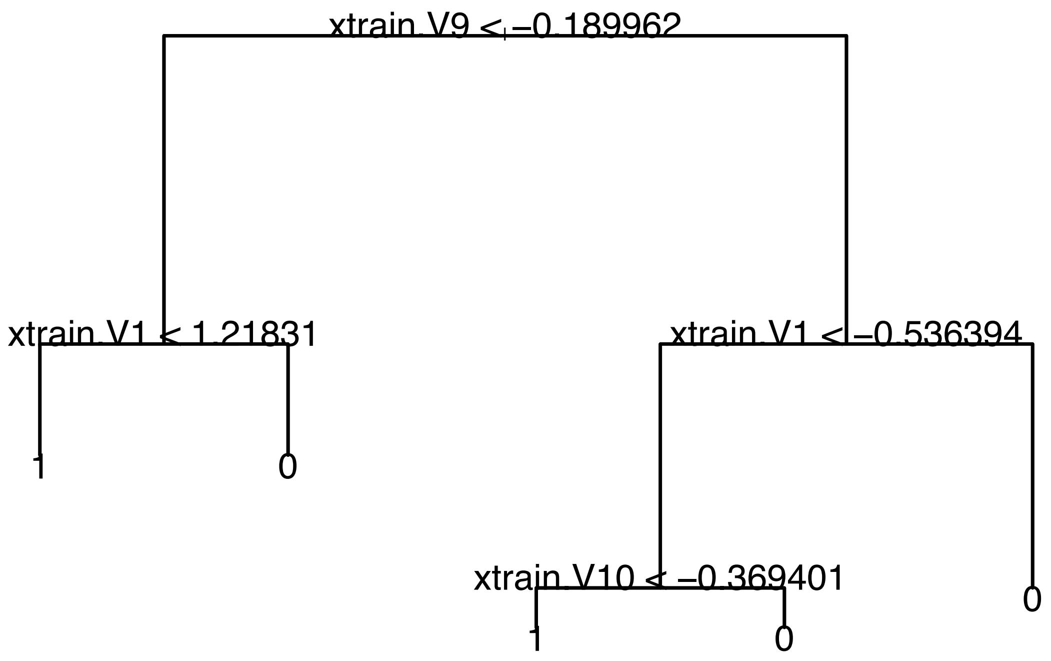

Figure 6: Classification tree. The size of the tree was chosen by cross-validation.

Data Set	Samples	Variables	Classes	$\bar{e}_S$	$\bar{e}_B$	**Decrease**
waveform	300	21	3	29.0	19.4	33%
heart	1395	16	2	10.0	5.3	47%
breast cancer	699	9	2	6.0	4.2	30%
ionosphere	351	34	2	11.2	8.6	23%
diabetes	1036	8	2	23.4	18.8	20%
glass	214	9	6	32.0	24.9	22%
soybean	307	35	19	14.5	10.6	27%

Bagging Regression Trees:

Data Set	$\bar{e}_S$	$\bar{e}_B$	**Decrease**
Boston Housing	19.1	11.7	39%
Ozone	23.1	18.0	22%
Friedman # 1	11.4	6.2	46%
Friedman # 2	30,800	21,700	30%
Friedman # 3	.0403	.0249	38%

4.1 Explanation of Bagging

To get some intuition about why bagging is useful, one explanation comes from Breiman (1996). As before, let $f(x, \mathcal{L})$ be a predictor learned on a random training dataset $\mathcal{L}$. Then the aggregated predictor is

$$f_A(x, P) = \mathbb{E}_{\mathcal{L}} f(x, \mathcal{L}).$$

And the average prediction error in $f(x, \mathcal{L})$ under $0-1$ loss is

$$e = \mathbb{E}_{\mathcal{L}} E_{Y,X}(Y - f(X, \mathcal{L}))^2,$$

where $(Y, X) \perp \mathcal{L}$ and $(Y, X) \sim P$. On the other hand, the average prediction error in f_A is

$$e_A = E_{Y,X}(Y - f_A(X, P))^2.$$

Then e and e_A are compared using Jensen's inequality as

$$\begin{aligned} e &= E_{Y,X}Y^2 - 2E_{Y,X}Y\mathbb{E}_{\mathcal{L}} f(X, \mathcal{L}) + E_{Y,X}\mathbb{E}_{\mathcal{L}} f^2(X, \mathcal{L}) \\ &\geq E_{Y,X}Y^2 - 2E_{Y,X}Y f_A(X, P) + E_{Y,X} f_A^2(X, P) \\ &= E_{Y,X}(Y - f_A(X, P))^2 = e_A. \end{aligned}$$

Note that $e - e_A$ depends on how unequal of the two sides of

$$f_A^2(X, P) = [\mathbb{E}_{\mathcal{L}} f(X, \mathcal{L})]^2 \leq \mathbb{E}_{\mathcal{L}} f^2(X, \mathcal{L}).$$

So, If $f(x, \mathcal{L})$ is stable with respect to $\mathcal{L}$, e and e_A are close, and vice versa. Conclusively, f_A always improves f, but it improves more for unstable predictors.

The bagging f_B can be thought to be an estimator of f_A provided that $\mathcal{L}$ is similar to the population (i.e. P).

To sum up, Bagging works well with unstable learners such as decision trees, variable selection, Neural networks etc. However, Bagging does not improve much stable learners such as linear models, regularized learners, nearest neighbor etc. Question is, When and Why is $Var_{\mathcal{L}}(f(X, \mathcal{L}))$ large ?

Now, we consider this example from Buhlmann and Yu (2002). Suppose that $x \in \mathbb{R}$ and use the notation $\mathcal{L} = \{Y_1, \ldots, Y_n\}$ here. Consider the simple decision rule

$$f(x, \mathcal{L}) = I(\overline{Y}_n \leq x).$$

Let $\mu = \mathbb{E}[Y_i]$ and for simplicity assume that $\text{Var}(Y_i) = 1$. Suppose that x is close to μ relative to the sample size. We can model this by setting $x \equiv x_n = \mu + c/\sqrt{n}$. Then central limit theorem implies $\sqrt{n}(\bar{Y}_n - \mu) \xrightarrow{d} N(0, 1)$, so

$$f(x, \mathcal{L}) = I(\sqrt{n}(\bar{Y}_n - \mu) \leq c) \approx I(Z \leq c),$$

where $Z \sim N(0, 1)$. So the limiting mean and variance of $f(x, \mathcal{L})$ are $\Phi(c)$ and $\Phi(c)(1 - \Phi(c))$. Now the bootstrap distribution of $\overline{Y}^*$ (conditional on $Y_1, \ldots, Y_n$) is approximately $N(\overline{Y}, 1/n)$. That is, $\sqrt{n}(\overline{Y}^* - \overline{Y})|Y_1, \ldots, Y_n \xrightarrow{d}$

$N(0,1)$. Let $\mathbb{E}^*$ denote the average with respect to the bootstrap randomness. Then, if f_B is the bagged estimator (with the ideal case where $B \to \infty$), we have

$$\begin{aligned} f_B(x_n) &= \mathbb{E}^*[I(\overline{Y}^* \le x_n)] = \mathbb{E}^*\left[I\left(\sqrt{n}(\overline{Y}^* - \overline{Y}) \le \sqrt{n}(x_n - \overline{Y})\right)\right] \\ &\approx \Phi(\sqrt{n}(x_n - \overline{Y})) \approx \Phi(c + Z), \end{aligned}$$

where $Z \sim N(0,1)$, and again we used the fact that $\sqrt{n}(\bar{Y}_n - \mu) \xrightarrow{d} N(0,1)$.

To summarize, $f(x,\mathcal{L}) \approx I(Z \le c)$ while $f_B \approx \Phi(c+Z)$ which is a smoothed version of $I(Z \le c)$. In other words, bagging is a smoothing operator. In particular, suppose we take $c = 0$. Then $f(x,\mathcal{L})$ converges to a Bernoulli with mean $1/2$ and variance $1/4$. The bagged estimator f_B converges to $\Phi(Z) = \text{Unif}(0,1)$ which has mean $1/2$ and variance $1/12$. The reduction in variance is due to the smoothing effect of bagging.

4.2 Remarks

We see how bagging works on Boston Housing Market data.

- Objective: Find a housing value equation that relates the median value of homes in a Boston area to air pollution and to 12 other variables.
- Base Classifier: (unpruned) decision trees
- Bagging results in 39% improvement upon the best single tree.

From Figure 4.2, we can say that bootstrapped predictors have at least five different structures. So, the bagged predictor is a linear combination of these five trees.

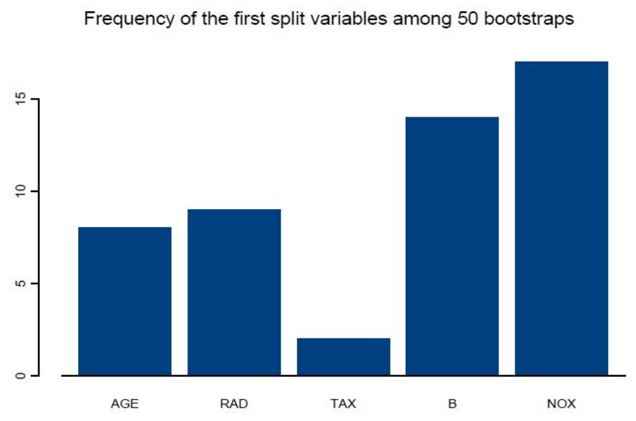

Figure 7: 50 Bootstrapped trees on Boston Housing Market data.

So far we have seen the bagging on regression. On the contrary, bagging on classification does not always work. This is mainly because the loss function (typically 0-1) is not convex. So no nice bias-variance decomposition exists. Simple example is:

- $\Pr(Y = 1|X = x) = 0.1$
- $f(x,\mathcal{L}) = 0.95$ w.p. 0.5 and $= 0.1$ w.p. 0.5
- Then $f_A(x) = 0.525$ and the missclassification rate is 0.9, which is worse than that of a single predictor (error rate= 0.5)

Bagging can make a good classifier better but can make a bad classifier worse (Breiman 1998, Tibshirani 1996, Friedman, 1996). However, in practice, Bagging works well for classification but no explanation for it.

5 Random Forests

Finally we get to random forests. These are bagged trees except that we also choose random subsets of features for each tree. The estimator can be written as

$$\hat{f}(x) = \frac{1}{M}\sum_j \hat{f}_j(x)$$

where $\hat{f}_j$ is a tree estimator based on a subsample (or bootstrap) of size a using p randomly selected features. The trees are usually required to have some number k of observations in the leaves. There are three tuning parameters: a, p and k. You could also think of M as a tuning parameter but generally we can think of M as tending to ∞.

For each tree, we can estimate the prediction error on the un-used data. (The tree is built on a subsample.) Averaging these prediction errors gives an estimate called the *out-of-bag* error estimate.

5.1 Theories

Unfortunately, it is very difficult to develop theory for random forests since the splitting is done using greedy methods. Much of the theoretical analysis is done using simplified versions of random forests.

We first do analysis using somewhat Bayesian approach. suppose each tree is $T_k(x) = T(x:\theta_k)$ and $\theta_k \overset{iid}{\sim} \pi(\theta)$. Let

$$mg(x,y) = \frac{1}{M}\sum_{k=1}^{M} I(T_k(x) = y) - \frac{1}{M}\sum_{k=1}^{M} I(T_k(x) = -y).$$

Let PE_M^* be the prediction error. Then

$$PE_M^* = P_{(X,Y)}(mg(X,Y) < 0).$$

By SLLN, as $M \to \infty$,

$$PE_M^* \to PE^* = P_{(X,Y)}\left\{P_\theta(T(X:\theta) = Y) - P_\theta(T(X:\theta) = -Y) < 0\right\}.$$

We will derive the upper bound of PE^* as a function of the correlation and strength.

Let

$$mr(X,Y) = P_\theta(T(X:\theta) = Y) - P_\theta(T(X:\theta) = -Y)$$

Then the strength of a single tree is

$$s = \mathbb{E}_{(X,Y)}(mr(X,Y)).$$

Chebychev's inequality gives

$$PE^* \le Var(mr)/s^2.$$

Now we calculate $Var(mr).mr(X,Y)$ can be expanded as

$$\begin{aligned} mr(X,Y) &= \mathbb{E}_\theta(I(T(X:\theta) = Y) - I(T(X:\theta) = -Y)) \\ &= \mathbb{E}_\theta(rmg(\theta, X, Y)), \end{aligned}$$

where $rmg(\theta,X,Y) = I(T(X:\theta) = Y) - I(T(X:\theta) = -Y)$. Note that

$$mr(X,Y)^2 = \mathbb{E}_{\theta,\theta'} rmg(\theta,X,Y) rmg(\theta^{'},X,Y),$$

where θ and θ' are independent and $\theta, \theta^{'} \sim \pi(\theta)$. Then

$$\begin{aligned} Var(mr) &= \mathbb{E}_{\theta,\theta'} cov(rmg(\theta,X,Y), rmg(\theta^{'},X,Y)) \\ &= \mathbb{E}_{\theta,\theta'}\left(\rho(\theta,\theta')sd(\theta)sd(\theta')\right), \end{aligned}$$

where

$$\rho(\theta,\theta^{'}) = \mathrm{corr}_{(X,Y)}\left(rmg(\theta,X,Y), rmg(\theta^{'},X,Y)|\theta,\theta^{'}\right)$$

and $sd(\theta) = \mathrm{std}_{(X,Y)}(rmg(\theta,X,Y)|\theta)$. Then

$$Var(mr) = \bar{\rho}\left(\mathbb{E}_\theta(sd(\theta))\right)^2 \le \bar{\rho}\mathbb{E}_\theta(sd^2(\theta))$$

where

$$\bar{\rho} = \mathbb{E}_{\theta,\theta'}\left(\rho(\theta,\theta^{'})sd(\theta)sd(\theta^{'})\right)/\mathbb{E}_{\theta,\theta'}(sd(\theta)sd(\theta^{'})).$$

Write

$$\begin{aligned}\mathbb{E}_\theta(sd^2(\theta)) &\le \mathbb{E}_\theta(\mathbb{E}_{(X,Y)}(rmg(\theta,X,Y)^2)) - \mathbb{E}_\theta(\mathbb{E}_{(X,Y)}(rmg(\theta,X,Y)))^2 \\ &\le 1 - s^2.\end{aligned}$$

To sum up, we have

$$PE^* \le \bar{\rho}(1-s^2)/s^2.$$

It shows that the two ingredients involved in the generalization error for random forests are the strength of the individual classifiers in the forest, and the correlation between them in terms of the raw margin functions. It is the first of its kind to explain that an important ingredient in success of ensemble methods is diversity of base learners. For example, random perturbation can be understood as a tool of increasing the diversity.

One example of a simplified tree is the *centered forest* is defined as follows. Suppose the data are on $[0,1]^d$. Choose a random feature, split in the center. Repeat until there are k leaves. This defines one tree. Now we average M such trees. Breiman (2004) and Biau (2002) proved the following.

Theorem. *If each feature is selected with probability* $1/d$, $k = o(n)$ *and* $k \to \infty$ *then*

$$\mathbb{E}[|\hat{f}(X) - f^0(X)|^2] \to 0$$

as $n \to \infty$.

Under stronger assumptions we can say more:

Theorem. *Suppose that* m *is Lipschitz and that* m *only depends on a subset* S *of the features and that the probability of selecting* $j \in S$ *is* $(1/S)(1+o(1))$. *Then*

$$\mathbb{E}|\hat{f}(X) - f^0(X)|^2 = O\left(\frac{1}{n}\right)^{\frac{3}{4|S|\log 2+3}}.$$

This is better than the usual Lipschitz rate $n^{-2/(d+2)}$ if $|S| \le p/2$. But the condition that we select relevant variables with high probability is very strong and proving that this holds is a research problem.

A significant step forward was made by Scornet, Biau and Vert (2015). Here is their result.

Theorem. *Suppose that* $Y = \sum_j f_j(X(j)) + \epsilon$ *where* $X \sim \text{Uniform}[0,1]^d$, $\epsilon \sim N(0,\sigma^2)$ *and each* f_j *is continuous, with* $f^0(X) = \sum_j f_j(X(j))$. *Assume that the split is chosen using the maximum drop in sums of squares. Let* t_n *be the number of leaves on each tree and let* a_n *be the subsample size. If* $t_n \to \infty$, $a_n \to \infty$ *and* $t_n(\log a_n)^9/a_n \to 0$ *then*

$$\mathbb{E}[|\hat{f}(X) - f^0(X)|^2] \to 0$$

as $n \to \infty$.

Again, the theorem has strong assumptions but it does allow a greedy split selection. Scornet, Biau and Vert (2015) provide another interesting result. Suppose that (i) there is a subset S of relevant features, (ii) $p = d$, (iii) f_j is not constant on any interval for $j \in S$. Then with high probability, we always split only on relevant variables.

5.2 Connection to Nearest Neighbors

Lin and Jeon (2006) showed that there is a connection between random forests and k-NN methods. We say that X_i is a *layered nearest neighbor* (LNN) of x If the hyper-rectangle defined by x and X_i contains no data points except X_i. Now note that if tree is grown until each leaf has one point, then $\hat{f}(x)$ is simply a weighted average of LNN's. More generally, Lin and Jeon (2006) call X_i a k-potential nearest neighbor $k - PNN$ if there are fewer than k samples in the the hyper-rectangle defined by x and X_i. If we restrict to random forests whose leaves have k points then it follows easily that $\hat{f}(x)$ is some weighted average of the $k - PNN$'s.

Let us know return to LNN's. Let $\mathcal{L}_n(x)$ denote all LNN's of x and let $L_n(x) = |\mathcal{L}_n(x)|$. We could directly define

$$\hat{f}(x) = \frac{1}{L_n(x)}\sum_i Y_i I(X_i \in \mathcal{L}_n(x)).$$

Biau and Devroye (2010) showed that, if X has a continuous density,

$$\frac{(d-1)!\mathbb{E}[L_n(x)]}{2^d(\log n)^{d-1}} \to 1.$$

Moreover, if Y is bounded and m is continuous then, for all $p \geq 1$,

$$\mathbb{E}|\hat{f}(X) - f^0(X)|^p \to 0$$

as $n \to \infty$. Unfortunately, the rate of convergence is slow. Suppose that $\mathrm{Var}(Y|X = x) = \sigma^2$ is constant. Then

$$\mathbb{E}|\hat{f}(X) - f^0(X)|^p \geq \frac{\sigma^2}{\mathbb{E}[L_n(x)]} \sim \frac{\sigma^2(d-1)!}{2^d(\log n)^{d-1}}.$$

If we use k-PNN, with $k \to \infty$ and $k = o(n)$, then the results Lin and Jeon (2006) show that the estimator is consistent and has variance of order $O(1/k(\log n)^{d-1})$.

As an aside, Biau and Devroye (2010) also show that if we apply bagging to the usual 1-NN rule to subsamples of size k and then average over subsamples, then, if $k \to \infty$ and $k = o(n)$, then for all $p \geq 1$ and all distributions P, we have that $\mathbb{E}|\hat{f}(X) - f^0(X)|^p \to 0$. So bagged 1-NN is universally consistent. But at this point, we have wondered quite far from random forests.

5.3 Connection to Kernel Methods

There is also a connection between random forests and kernel methods (Scornet 2016). Let $A_j(x)$ be the cell containing x in the j^{th} tree. Then we can write the tree estimator as

$$\hat{f}(x) = \frac{1}{M}\sum_j \sum_i \frac{Y_i I(X_i \in A_j(x))}{N_j(x)} = \frac{1}{M}\sum_j \sum_i W_{ij} Y_j$$

where $N_j(x)$ is the number of data points in $A_j(x)$ and $W_{ij} = I(X_i \in A_j(x))/N_j(x)$. This suggests that a cell A_j with low density (and hence small $N_j(x)$) has a high weight. Based on this observation, Scornet (2016) defined kernel based random forest (KeRF) by

$$\hat{f}(x) = \frac{\sum_j \sum_i Y_i I(X_i \in A_j(x))}{\sum_j N_j(x)}.$$

With this modification, $\hat{f}(x)$ is the average of each Y_i weighted by how often it appears in the trees. The KeRF can be written as

$$\hat{f}(x) = \frac{\sum_i Y_i K(x, X_i)}{\sum_s K_n(x, X_s)}$$

where

$$K_n(x, z) = \frac{1}{M}\sum_j I(x \in A_j(x)).$$

The trees are random. So let us write the j^{th} tree as $T_j = T(\Theta_j)$ for some random quantity Θ_j. So the forests is built from $T(\Theta_1), \ldots, T(\Theta_M)$. And we can write $A_j(x)$ as $A(x, \Theta_j)$. Then $K_n(x, z)$ converges almost surely (as $M \to \infty$) to $\kappa_n(x, z) = P_\Theta(z \in A(x, \Theta))$ which is just the probability that x and z are connected, in the sense that they are in the same cell. Under some assumptions, Scornet (2016) showed that KeRF's and forests are close to each other, thus providing a kernel interpretation of forests.

Recall the centered forest we discussed earlier. This is a stylized forest — quite different from the forests used in practice — but they provide a nice way to study the properties of the forest. In the case of KeRF's, Scornet (2016) shows that if $f^0(x)$ is Lipschitz and $X \sim \mathrm{Unif}([0, 1]^d)$ then

$$\mathbb{E}[(\hat{f}(x) - f^0(x))^2] \leq C(\log n)^2 \left(\frac{1}{n}\right)^{\frac{1}{3 + d \log 2}}.$$

This is slower than the minimax rate $n^{-2/(d+2)}$ but this probably reflects the difficulty in analyzing forests.

5.4 Variable Importance

Let $\hat{f}$ be a random forest estimator. How important is feature $X(j)$?

LOCO. One way to answer this question is to fit the forest with all the data and fit it again without using $X(j)$. When we construct a forest, we randomly select features for each tree. This second forest can be obtained by simply average the trees where feature j was not selected. Call this $\hat{f}_{(-j)}$. Let $\mathcal{H}$ be a hold-out sample of size m. Then let

$$\hat{\Delta}_j = \frac{1}{m}\sum_{i\in\mathcal{H}} W_i$$

where

$$W_i = (Y_i - \hat{f}_{(-j)}(X_i))^2 - (Y_i - \hat{f}(X_i))^2.$$

Then Δ_j is a consistent estimate of the prediction risk inflation that occurs by not having access to $X(j)$. Formally, if $\mathcal{T}$ denotes the training data then,

$$\mathbb{E}[\hat{\Delta}_j|\mathcal{T}] = \mathbb{E}\left[(Y - \hat{f}_{(-j)}(X))^2 - (Y - \hat{f}(X))^2 \,\middle|\, \mathcal{T}\right] \equiv \Delta_j.$$

In fact, since $\hat{\Delta}_j$ is simply an average, we can easily construct a confidence interval. This approach is called LOCO (Leave-Out-COvariates). Of course, it is easily extended to sets of features. The method is explored in Lei, G'Sell, Rinaldo, Tibshirani, Wasserman (2017) and Rinaldo, Tibshirani, Wasserman (2015).

Permutation Importance. A different approach is to permute the values of $X(j)$ for the out-of-bag observations, for each tree. Let $\mathcal{O}_j$ be the out-of-bag observations for tree j and let $\mathcal{O}_j^*$ be the out-of-bag observations for tree j with $X(j)$ permuted.

$$\hat{\Gamma}_j = \frac{1}{M}\sum_j\sum_i W_{ij}$$

where

$$W_{ij} = \frac{1}{m_j}\sum_{i\in\mathcal{O}_j^*}(Y_i - \hat{f}_j(X_i))^2 - \frac{1}{m_j}\sum_{i\in\mathcal{O}_j}(Y_i - \hat{f}_j(X_i))^2.$$

This avoids using a hold-out sample. This is estimating

$$\Gamma_j = \mathbb{E}[(Y - \hat{f}(X_j'))^2] - \mathbb{E}[(Y - \hat{f}(X))^2]$$

where X_j' has the same distribution as X except that $X_j'(j)$ is an independent draw from $X(j)$. This is a lot like LOCO but its meaning is less clear. Note that $\hat{f}_j$ is not changed when $X(j)$ is permuted. Gregorutti, Michel and Saint Pierre. (2013) show that, when (X,ϵ) is Gaussian, that $\text{Var}(X) = (1-c)I + c\mathbf{1}\mathbf{1}^T$ and that $\text{Cov}(Y, X(j)) = \tau$ for all j then

$$\Gamma_j = 2\left(\frac{\tau}{1-c+dc}\right)^2.$$

It is not clear how this connects to the actual importance of $X(j)$. In the case where $Y = \sum_j f_j(X(j)) + \epsilon$ with $\mathbb{E}[\epsilon|X] = 0$ and $\mathbb{E}[\epsilon^2|X] < \infty$, they show that $\Gamma_j = 2\text{Var}(f_j(X(j))$.

5.5 Inference

Using the theory of infinite order U-statistics, Mentch and Hooker (2015) showed that $\sqrt{n}(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])/\sigma$ converges to a Normal(0,1) and they show how to estimate σ.

Wager and Athey (2017) show asymptotic normality if we use sample splitting: part of the data are used to build the tree and part is used to estimate the average in the leafs of the tree. Under a number of technical conditions — including the fact that we use subsamples of size $s = n^\beta$ with $\beta < 1$ — they show that $(\hat{f}(x) - f^0(x))/\sigma_n(x) \rightsquigarrow N(0,1)$ and they show how to estimate $\sigma_n(x)$. Specifically,

$$\hat{\sigma}_n^2(x) = \frac{n-1}{n}\left(\frac{n}{n-s}\right)^2 \sum_i (\text{Cov}(\hat{f}_j(x), N_{ij})^2$$

where the covariance is with respect to the trees in the forest and $N_{ij} = 1$ of (X_i, Y_i) was in the j^{th} subsample and 0 otherwise.

5.6 Empirical Results

Random forest vs single tree, Bagging, Boosting:

Data Set	Single	Bagging	AdaBoost	RF1	RF1-L
waveform	29.0	19.4	18.2	17.2	16.1
breast cancer	6.0	5.3	3.2	2.9	2.9
ionosphere	11.2	8.6	5.9	7.1	5.7
diabetes	23.4	18.8	20.2	24.2	23.1
glass	32.0	24.9	22.0	20.6	23.5

Moreover, random forest is robust to noise. If we inject 5% noise to output labels, then increases in error rates due to noise are:

Data Set	AdaBoost	RF1
breast cancer	43.2	1.8
ionosphere	27.7	3.8
diabetes	6.8	1.8
glass	1.6	0.4

5.7 Summary

Random forests are considered one of the best all purpose classifiers. But it is still a mystery why they work so well. The situation is very similar to deep learning. We have seen that there are now many interesting theoretical results about forests. But the results make strong assumptions that create a gap between practice and theory. Furthermore, there is no theory to say why forests outperform other methods. The gap between theory and practice is due to the fact that forests — as actually used in practice — are complex functions of the data.

6 Boosting

Boosting is a method of combing weak learners (learners slightly better than random guess) to produce a strong committee. Freund and Schapire (1997) first proposed a practically usable boosting algorithm called "AdaBoost (Adaptive Boost)". Since then, many researches have been done to understand and extend boosting.

A motivating example is the horse race. Suppose there are 100 gamblers who claim that they are expert in predicting the horse race results. Since they spend lots of time to study horse race, we can admit that their winning probabilities are slightly better than random guess (i.e. 50%). Now, the question is "Is it possible to combine 100 predictions of the 100 experts to make a better prediction?". Surprisingly, it is possible, which is called *weak learnability*. AdaBoot is the first algorithm to implement the idea of weak learnability.

Let $Z_i = (X_i, Y_i)$ where $Y_i \in \{-1, +1\}$. We make the weak learning assumption: for some $\gamma > 0$ we have an algorithm returns $h \in \mathcal{H}$ such that, for all P,

$$P(R(h) \leq 1/2 - \gamma) \geq 1 - \delta$$

where $\gamma > 0$ is the edge.

AdaBoost algorithm is as follows:

1. Set $D_1(i) = 1/n$ for $i = 1, \ldots, n$.
2. Repeat for $t = 1, \ldots, T$:
   (a) Let $h_t = \text{argmin}_{h \in \mathcal{H}} P_{D_t}(Y_i \neq h(X_i))$.
   (b) $\epsilon_t = P_{D_t}(Y_i \neq h_t(X_i))$.
   (c) $\alpha_t = (1/2) \log((1 - \epsilon_t)/\epsilon_t)$.
   (d) Let
   $$D_{t+1}(i) = \frac{D_t(i) e^{-Y_i \alpha_t h_t(X_i)}}{Z_t}$$
   where Z_t is a normalizing constant.

3. Set $g(x) = \sum_t \alpha_t h_t(x)$.
4. Return $h(x) = \text{sign} g(x)$.

AdaBoost increases the weights for misclassified observations and decreases the weights for correctly classified observations.

6.1 Empirical Results

Data Set	Single	AdaBoost	**Decrease**	Bagging
waveform	29.0	18.2	37%	19.4
breast cancer	6.0	3.2	47%	5.3
ionosphere	11.2	5.9	47%	8.6
diabetes	23.4	20.2	14%	18.8
glass	32.0	22.0	31%	24.9

AdaBoost outperforms the single best model. Moreover, AdaBoost is more accurate than Bagging in most cases (except **diabetes**).

6.2 Training Error

$h_t, t = 1, \ldots, T$, called *base learners*, are typically slightly better than random guess. In particular, Freund and Schapire (1997) showed that the training error converges to 0 exponentially fast if

$$\text{err}_t < 0.5 - \gamma$$

for some $\gamma > 0$. They conjectured that the overfitting emerges if T is too large since the model is too complex.

Lemma. *We have*

$$Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}.$$

Proof. Since $\sum_i D_t(i) = 1$ we have

$$\begin{aligned} Z_t &= \sum_i D_t(i) e^{-\alpha_t Y_i h_t(X_i)} = \sum_{Y_i h_t(X_i)=1} D_t(i) e^{-\alpha_t} + \sum_{Y_i h_t(X_i)=-1} D_t(i) e^{\alpha_t} \\ &= (1-\epsilon_t) e^{-\alpha_t} + \epsilon_t e^{\alpha_t} = 2\sqrt{\epsilon_t(1-\epsilon_t)}. \end{aligned}$$

since $\alpha_t = (1/2)\log((1-\epsilon_t)/\epsilon_t)$. □

Theorem. *Suppose that $\gamma \leq (1/2) - \epsilon_t$ for all t. Then*

$$\hat{R}(h) \leq e^{-2\gamma^2 T}.$$

Hence, the training error goes to 0 quickly.

Proof. Recall that $D_1(i) = 1/n$. So

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i) e^{-\alpha_t Y_i h_t(X_i)}}{Z_t} = \frac{D_{t-1}(i) e^{-\alpha_{t-1} Y_i h_{t-1}(X_i)} e^{-\alpha_t Y_i h_t(X_i)}}{Z_t Z_{t-1}} \\ &= \cdots = \frac{e^{-Y_i \sum_t \alpha_t h_t(X_i)}}{n \prod_t Z_t} = \frac{e^{-Y_i g(X_i)}}{n \prod_t Z_t} \end{aligned}$$

which implies that

$$e^{-Y_i g(X_i)} = n D_{T+1}(i) \prod_t Z_t. \tag{4}$$

Since $I(u \le 0) \le e^{-u}$ we have

$$\begin{aligned}
\hat{R}(h) &= \frac{1}{n}\sum_i I(Y_i g(X_i) \le 0) \le \frac{1}{n}\sum_i e^{-Y_i g(X_i)} = \frac{1}{n}\sum_i n(\prod_t Z_t) D_{T+1}(i) = \prod_{t=1}^{T} Z_t \\
&= \prod_t 2\sqrt{\epsilon_t(1-\epsilon_t)} = \prod_t \sqrt{1 - 4(1/2-\epsilon_t)^2} \\
&\le \prod_t e^{-2(1/2-\epsilon_t)^2} \qquad \text{since } 1 - x \le e^{-x} \\
&= e^{-2\sum_t (1/2-\epsilon_t)^2} \le e^{-2\gamma^2 T}.
\end{aligned}$$

□

6.3 Generalization Error

The training error gets small very quickly. But how well do we do in terms of prediction error?

Let

$$\mathcal{F} = \left\{ \text{sign}(\sum_t \alpha_t h_t) :\ \alpha_t \in \mathbb{R},\ h_t \in \mathcal{H} \right\}.$$

For fixed $h = (h_1, \ldots, h_T)$ this is just a set of linear classifiers which has VC dimension T. So the shattering number is

$$\left(\frac{en}{T}\right)^T.$$

If $\mathcal{H}$ is finite then the shattering number is

$$\left(\frac{en}{T}\right)^T . |\mathcal{H}|^T.$$

If $\mathcal{H}$ is infinite but has VC dimension d then the shattering number is bounded by

$$\left(\frac{en}{T}\right)^T \left(\frac{en}{d}\right)^{dT} \preceq n^{Td}.$$

By the VC theorem, with probability at least $1-\delta$,

$$R(\hat{h}) \le \hat{R}(h) + \sqrt{\frac{Td \log n}{n}}.$$

Unfortunately this depends on T. We can fix this using margin theory.

Margins. Consider the classifier $h(x) = \text{sign}(g(x))$ where $g(x) = \sum_t \alpha_t h_t(x)$. The classifier is unchanged if we multiply g by a scalar. In particular, we can replace g with $\tilde{g} = g/||\alpha||_1$. This form of the classifier is a convex combination of the h_t's.

We define the *margin at* x of $g = \sum_t \alpha_t h_t$ by

$$\rho(x) = \frac{yg(x)}{||\alpha||_1} = y\tilde{g}(x).$$

Think of $|\rho(x)|$ as our confidence in classifying x. The margin of g is defined to be

$$\rho = \min_i \rho(X_i) = \min_i \frac{Y_i g(X_i)}{||\alpha||_1}.$$

Note that $\rho \in [-1, 1]$.

To proceed we need to review Radamacher complexity. Given a class of functions $\mathcal{F}$ with $-1 \leq f(x) \leq 1$ we define

$$\mathcal{R}_n(\mathcal{F}) = \mathbb{E}_\sigma \left[\sup_{f \in \mathcal{F}} \frac{1}{n} \sum_i \sigma_i f(Z_i) \right]$$

where $P(\sigma_i = 1) = P(\sigma_i = -1) = 1/2$. If $\mathcal{H}$ is finite then

$$\mathcal{R}_n(\mathcal{H}) \leq \sqrt{\frac{2 \log |\mathcal{H}|}{n}}.$$

If $\mathcal{H}$ has VC dimension d then

$$\mathcal{R}_n(\mathcal{H}) \leq \sqrt{\frac{2d \log(en/d)}{n}}.$$

We will need the following two facts. First,

$$\mathcal{R}_n(\text{conv}(\mathcal{H})) = \mathcal{R}_n(\mathcal{H})$$

where conv$(\mathcal{H})$ is the convex hull of $\mathcal{H}$. Second, if

$$|\phi(x) - \phi(y)| \leq L||x - y||$$

for all x, y then

$$\mathcal{R}_n(\phi \circ \mathcal{F}) \leq L \mathcal{R}_n(\mathcal{F}).$$

The set of margin functions is

$$\mathcal{M} = \{ y f(x) : \ f \in \text{conv}(\mathcal{H}) \}.$$

We then have

$$\mathcal{R}_n(\mathcal{M}) = \mathcal{R}_n(\text{conv}(\mathcal{H})) = \mathcal{R}_n(\mathcal{H}).$$

A key result is that, with probability at least $1 - \delta$, for all $f \in \mathcal{F}$,

$$\mathbb{E}[f(Z)] \leq \frac{1}{n} \sum_i f(Z_i) + 2\mathcal{R}_n(\mathcal{F}) + \sqrt{\frac{2 \log(1/\delta)}{n}}. \tag{5}$$

Now fix a number ρ and define the margin-sensitive loss function

$$\phi(u) = \begin{cases} 1 & u \leq 0 \\ 1 - \frac{u}{\rho} & 0 \leq \rho \\ 0 & u \geq \rho. \end{cases}$$

Note that

$$I(u \leq 0) \leq \phi(u) \leq I(u \leq \rho).$$

Assume that $\mathcal{H}$ has VC dimension d. Then

$$\mathcal{R}_n(\phi \circ \mathcal{M}) \leq L \mathcal{R}_n(\mathcal{M}) \leq L \mathcal{R}_n(\mathcal{H}) \leq \frac{1}{\rho} \sqrt{\frac{2d \log(en/d)}{n}}.$$

Now define the empirical margin sensitive loss of a classifer f by

$$\hat{R}_\rho = \frac{1}{n} \sum_i I(Y_i f(X_i) \leq \rho).$$

Theorem. *With probability at least* $1 - \delta$,

$$R(g) \leq \hat{R}_\rho(g/||\alpha||_1) \leq \frac{1}{\rho} \sqrt{\frac{2d \log(en/d)}{n}} + \sqrt{\frac{2 \log(2/\delta)}{n}}.$$

Proof. Recall that $I(u \leq 0) \leq \phi(u) \leq I(u \leq \rho)$. Also recall that g and $\tilde{g} = g/||\alpha||_1$ are equivalent classifiers. Then using (5) we have

$$\begin{aligned} R(g) = R(\tilde{g}) = P(Y\tilde{g}(X) \leq 0) &\leq \frac{1}{n}\sum_i \phi(Y_i\tilde{g}(X_i)) + 2\mathcal{R}_n(\phi \circ \mathcal{M}) + \sqrt{\frac{2\log(2/\delta)}{n}} \\ &\leq \frac{1}{n}\sum_i \phi(Y_i\tilde{g}(X_i)) + \frac{1}{\rho}\sqrt{\frac{2d\log(en/d)}{n}} + \sqrt{\frac{2\log(2/\delta)}{n}} \\ &= \hat{R}_\rho(g/||\alpha||_1) + \frac{1}{\rho}\sqrt{\frac{2d\log(en/d)}{n}} + \sqrt{\frac{2\log(2/\delta)}{n}}. \end{aligned}$$

□

Next we bound $\hat{R}_\rho(g/||\alpha||_1)$.

Theorem. *We have*

$$\hat{R}_\rho(g/||\alpha||_1) \leq \prod_{t=1}^T \sqrt{4\epsilon_t^{1-\rho}(1-\epsilon_t)^{1+\rho}}.$$

Proof. Since $\phi(u) \leq I(u \leq \rho)$ we have

$$\begin{aligned} \hat{R}_\rho(g/||\alpha||_1) &\leq \frac{1}{n}\sum_i I(Y_i g(X_i) - \rho||\alpha||_1 \leq 0) \\ &\leq e^{\rho||\alpha||_1}\frac{1}{n}\sum_i e^{-Y_i g(X_i)} \\ &= e^{\rho||\alpha||_1}\frac{1}{n}\sum_i nD_{T+1}(i)\prod_t Z_t = e^{\rho||\alpha||_1}\prod_t Z_t \\ &= \prod_{t=1}^T \sqrt{4\epsilon_t^{1-\rho}(1-\epsilon_t)^{1+\rho}} \end{aligned}$$

since $Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}$ and $\alpha_t = (1/2)\log((1-\epsilon_t)/\epsilon_t)$. □

Assuming $\gamma \leq (1/2 - \epsilon_t)$ and $\rho < \gamma$ then it can be shown that $\sqrt{4\epsilon_t^{1-\rho}(1-\epsilon_t)^{1+\rho}} \equiv b < 1$. So $\hat{R}_\rho(g/||\alpha||_1) \leq b^T$. Combining with the previous result we have, with probability at least $1-\delta$,

$$R(g) \leq b^T + \frac{1}{\rho}\sqrt{\frac{2d\log(en/d)}{n}} + \sqrt{\frac{2\log(2/\delta)}{n}}.$$

This shows that we get small error even with T large (unlike the earlier bound based only on VC theory).

7 References

Biau, Devroye and Lugosi. (2008). Consistency of Random Forests and Other Average Classifiers. *JMLR*.

Biau, Gerard, and Scornet. (2016). A random forest guided tour. Test 25.2: 197-227.

Biau, G. (2012). Analysis of a Random Forests Model. arXiv:1005.0208.

Buhlmann, P., and Yu, B. (2002). Analyzing bagging. Annals of Statistics, 927-961.

Gregorutti, Michel, and Saint Pierre. (2013). Correlation and variable importance in random forests. arXiv:1310.5726.

Lei J, G'Sell M, Rinaldo A, Tibshirani RJ, Wasserman L. (2017). Distribution-free predictive inference for regression. Journal of the American Statistical Association.

Lin, Y. and Jeon, Y. (2006). Random Forests and Adaptive Nearest Neighbors. *Journal of the American Statistical Association*, 101, p 578.

L. Mentch and G. Hooker. (2015). Ensemble trees and CLTs: Statistical inference for supervised learning. Journal of Machine Learning Research.

Rinaldo A, Tibshirani R, Wasserman L. (2015). Uniform asymptotic inference and the bootstrap after model selection. arXiv preprint arXiv:1506.06266.

Scornet E. Random forests and kernel methods. (2016). IEEE Transactions on Information Theory. 62(3):1485-500.

Wager, S. (2014). Asymptotic Theory for Random Forests. arXiv:1405.0352.

Wager, S. (2015). Uniform convergence of random forests via adaptive concentration. arXiv:1503.06388.

Wager, S. and Athey, S. (2017). Estimation and inference of heterogeneous treatment effects using random forests. Journal of the American Statistical Association.