SOME INFINITY THEORY FOR PREDICTOR ENSEMBLES

Leo Breiman
Statistics Department
University of California
Berkeley, California
leo@stat.berkeley.edu

Technical Report 577 (August 2000)

Abstract

To dispel some of the mystery about what makes tree ensembles work, they are looked at in distribution space i.e. the limit case of "infinite" sample size. It is shown that the simplest kind of trees are complete in D-dimensional space if the number of terminal nodes T is greater than D. For such trees we show that the Adaboost minimization algorithm gives an ensemble converging to the Bayes risk. Random forests which are grown using i.i.d random vectors in the tree construction are shown to be equivalent to a kernel acting on the true margin. The form of this kernel is derived for purely random tree growing and its properties explored. The notions of correlation and strength for random forests is reflected in the symmetry and skewness of the kernel

1 **Introduction**

Using ensembles of predictors for classification and regression has proved to give more accurate results than use of a single predictor. Many alternative algorithms have been introduced and tested over the last few years. But our knowledge so far is mainly empirical, with little theory to explain why and how the generalization error is so significantly lowered on most data sets.

For a while, the concepts of bias and variance, originating in regression and extended to classification showed some promise of helping our understanding. Later empirical work cast a shadow over this hope. While bagging (Breiman[1996]) was a simple and understandable algorithm, it did not perform as well as Adaboost, a more complex algorithm originated by Freund and Schapire[1996]), whose ability to lower generalization error is not yet fully understood. Efforts to interpret it have so far proved incomplete. (For

performance comparisons see Drucker, H. and Cortes, C. [1995], Bauer, E. and Kohavi, R. [1999] ,Dietterich, T. [1998])

1.1 Classification

In classification, there appear to be two distinct ways of forming ensembles. One consists of arcing algorithms (including Adaboost) which have the characteristics:

i) They function deterministically, with no random elements, growing the ensemble by iterative reweightings of the training set.
ii) Using weak classifiers i.e. small trees, to form the ensemble gives the best results.
iii) The voting is weighted.
iv) They can all be cast as down-the-gradient methods to minimize a target function defined on the training set.

Examples are Adaboost and its variants-- arc-x4 (Breiman[1998]), the various margin-function minimizers explored by Mason et al. [2000], the logit-boost methods of Friedman et al[1998] and others.

The other class of algorithms, called randomization algorithms, is characterized by:

i) Randomness is introduced into the training set or the tree construction.
ii) They work best using the largest tree possible.
iii) The voting is unweighted.
iv) The framework for their understanding is in terms of the correlation between members of the ensemble and the "strength" of each member.

Examples are bagging (Breiman[1996]), random split selection (Dieterrich[1998]), randomizing outputs (Breiman[1998a]), random feature selection (Amit and Geman[1997], Breiman[1999]), and perfect random trees (Cutler[1999]).

The accuracy's of these two different types of algorithms is comparable (Breiman[1999]). But they work differently.

1.3 My Kingdom for Some Good Theoretical Explanations

The area of ensemble algorithms is filled with excellent empirical results, but the understanding of how they work is a scarce commodity. To this day, we have little understanding of how Adaboost produces such low generalization error and does not overfit as more and more predictors are added to the ensemble (although I have some guesses).

Only recently has some theoretical work appeared {Bhulmann and Yu [2000]) which explains how bagging works in some cases. But this is far from understanding how more complex ensemble algorithms work. This paper attempts to fill a small space in the vacuum. The framework given in Section 2 is interesting but unrealistic. But it results in some increased understanding of the ensemble mechanisms.

In particular, we are able to pinpoint the differences between the two methods of constructing classifier ensembles--to see what the role of randomness is, as well as partially dispelling other mysteries.

2 **Theoretical Assumptions and Outline of results**

2.1 Framework

We work in distribution space. Put another way, in infinite sample space (hence the title). The outputs and inputs are represented by two random vectors Y,**X** whose distribution are known and **X** is assumed distributed on a finite closed D-dimensional Euclidean rectangle, R^D. The distribution of **X** is given by P(**dx**) which is assumed to be absolutely continuous w.r. to Lesbegue measure on D dimensions so that P(**dx**)=f(**x**)**dx.** The distribution of Y is given by P(y|**x**). Both distributions are assumed known.

Only the two class situation is considered. The members of the ensemble will be trees, all with the same number T of terminal nodes, formed by cuts parallel to the axes (although it will become clear how this generalizes). Also, they will be ±1trees. This means that to each terminal node is assigned the value either +1 or - 1. Ordinary trees with arbitrary values attached to their terminal nodes could be used, but in our context they offer no advantage over ±1trees.

2.2 Non-Technical Outline of Results

In section 3 we define a set of classifiers ς to be complete on R^D if every real-valued function on R^D is equal to a linear combination (perhaps infinite) of classifiers in ς. We show that the class of ± 1 trees with T terminal nodes is complete if T>D. A simple example shows that stumps, with T=2, is not complete if D=2.

The implication is that a linear combination of trees with T>D terminal nodes can achieve the minimal error rate (Bayes risk). The question this raises is how to compute the coefficients of a combination that will give minimum error. The fundamental down-the-gradient algorithm for arcing is defined in Section 4. This gives a method for evaluating coefficients that

minimize any target function. In Section 5 the minimization algorithm applied to an exponential function (Adaboost analog) is shown to converge to the minimum possible error (Bayes rate).

Section 6 looks at randomization algorithms for constructing ensembles. We show that using trees with a fixed number of terminal nodes the randomization produces a kernel type algorithm for separating the two classes. In Section 7 an analytical form is found for the kernel for a purely random tree construction. Its shape is studied to give some idea of random forest kernels.

In Section 8, we look at the relationship between the kernel shape and the two concepts of correlation and strength that have so far been used to understand the action of random forests. The symmetry of the kernel depends on correlation between trees in the forest--the lower the correlation, the more symmetry. Its skewness depends on the "strength" of the individual classifiers in the ensemble. We show, by example, the type of boundaries between classes for which skewness is necessary. The results also show why using the largest trees possible result in the best accuracy.

Section 9 is devoted to unanswered questions. Perhaps the most important is what light these results shed about prediction using ensembles on finite training sets. My conjectures about Adaboost are stated.

3. **Completeness**

First some familiar Hilbert space definitions and properties.

Definition 1. *Let $L_2(P)$ be the space of functions on R^D that are square-integrable with respect to $P(\boldsymbol{dx})$. A set of functions $\mathtt{F}$ in $L_2(P)$ will be called complete if the L_2 closure of the set of all finite linear combinations of functions in $\mathtt{F}$, denoted by $L_2(\mathtt{F})$, equals $L_2(P)$.*

Property 1. *A set of functions $\mathtt{F}$ is complete if and only if there is no non-zero function g in $L_2(P)$ such that $(g,f)=0$ for all $f\in\mathtt{F}$.*

Note: $(f,g)=\int f(\mathbf{x})g(\mathbf{x})P(\mathbf{d}x)$

Proposition 1. *A set of functions $\mathtt{F}$ is complete if $L_2(\mathtt{F})$ includes the indicators of all D-dimensional subrectangles of R^D.*

Proof: Every function in $L_2(P)$ can be L_2 approximated by a linear combination of indicator functions of disjoint rectangles. Hence the

proposition follows.

3.1 ±1 Trees.

The ±1 trees are that set of trees most commonly used in used in two-class classification.

Definition 2. *h(**x**) is a ±1tree if the T terminal nodes are formed by successive univariate splits of the input variables such that for all* **x** *in a given terminal node, h(**x**) is always+1 or always -1.*

Proposition 2. *If T>D the class of ±1trees is complete in* $L_2(P)$.

Proof: To form the indicator function of the rectangle

$$R_0 = \{r_1 < x_1 \le s_1, r_2 < x_2 \le s_2, \ldots, r_D < x_D \le s_D\}$$

proceed as follows: trees with T terminal nodes use T-1 splits. If $T>D$, the splits can be used to make a tree Tr_1 which contains the rectangle

$$R_1 = \{x_1 \le s_1, x_2 \le s_2, \ldots, x_D \le s_D\}$$

as one of its terminal nodes with the value +1. Let Tr_2 be a tree with exactly the same terminal nodes as the first but with the value of each node reversed except for R_1. Then $.5*Tr_1+.5*Tr_2$ is the indicator of R_1. Repeating the above process, construct the indicator function of

$$R_2 = \{x_1 \le r_1, x_2 \le s_2, \ldots, x_D \le s_D\}$$

Subtracting this from the indicator of R_1 gives the indicator of

$$R_3 = \{s_1 < x_1 \le r_1, x_2 \le s_2, \ldots, x_D \le s_D\}$$

and continuing this process leads to the indicator of R_0.

Proposition 2 is probably if and only if. For instance, the class of stumps (two terminal node trees) is not complete in D=2. Take P uniformly distributed on the square $(-1,+1)^2$. Take g(**x**) to equal +1 in the first and third quadrants, and -1 in the second and fourth. Then $(h,g)=0$ for every two node ±1tree h(**x**).

3.2 Implications for Predicting With Ensembles

The implications of completeness in terms of using ensembles for prediction are encouraging. Suppose that the loss function $L(Y,\phi(X))$ is a measure of the error in using $\phi(X)$ to predict Y. The expected loss is

$$L^*(\phi)=E_{Y,\mathbf{X}}L(Y,\phi(X))$$

where the subscripts indication expectation with respect to $Y,\mathbf{X}$. Assume that functions exist in $L_2(P)$ that minimize $L^*(\phi)$. Then:

Theorem 1 *In R^D there exist linear combinations of ± 1trees with $T>D$ that converges in $L_2(P)$ to any minimizer of $L^*(\phi)$ that is in $L_2(P)$.*

We illustrate with classification. Here:

$$L^*(\phi)=P_{Y,\mathbf{X}}(Y\neq\phi(X))$$

To put this into a more familiar ensemble context, assume that $\phi(\mathbf{x})$ predicts y=1 if $\phi(\mathbf{x})>0$, else predicts -1. Then

$$L^*(\phi)=P_{Y,\mathbf{X}}(Y\neq sign(\phi(X))).$$

Let $P(i|\mathbf{x})=P(Y=i|\mathbf{x})$. Then

$$L^*(\phi)=\int I(\phi(\mathbf{x})\leq 0)P(1|\mathbf{x})P(d\mathbf{x})+\int I(\phi(\mathbf{x})>0)P(-1|\mathbf{x})P(d\mathbf{x})$$

where I is the indicator function. Take finite combinations of ± 1trees

$$\sum_m c_m h_m(\mathbf{x})$$

that converge to a non-positive function on the set $P(1|\mathbf{x})<P(-1|\mathbf{x})$ and a positive function on the complement of the set. Then the loss converges to

$$\int \min(P(1|\mathbf{x}),P(-1|\mathbf{x}))P(d\mathbf{x}),$$

which is the the Bayes risk.

In the following sections we adopt the convention that "all trees" refers to all ± 1 trees with T terminals nodes where T>D.

4. **A Constructive Algorithm**

The results in Section 3 are non-constructive. They assert existence, but give no idea as to how to construct linear combinations of trees that converge to a desired function. There is an algorithm, first introduced in Breiman[1997], further elaborated in Breiman[1999] (and rediscovered many times) that offers a constructive method for producing such sequences. The finite-dimensional numerical optimization version of this algorithm has been around for a while and is called the Gauss-Southwell method (see Forsyth and Wasow[1963]).

The Gauss-Southwell method for minimizing a differentiable function $f(x_1,\ldots,x_m)$ of m real variables goes this way: at a point **x** compute all the partial derivatives $\partial f(x_1,\ldots,x_m)/\partial x_k$. Let the minimum of these be at x_j. Find the step of size α that minimizes $f(x_1,\ldots,x_j+\alpha,\ldots,x_m)$. Let the new **x** be $x_1,\ldots,x_j+\alpha,\ldots,x_m$ for the minimizing α value. If f is strictly convex on its domain, this algorithm converges to the global minimum.

The analog arcing algorithm produces a sequence of linear combinations of trees that minimize (under appropriate conditions) a given real valued target function of the sequence. Just as the Gauss-Southwell method, it finds the largest negative gradient at each **x** and then does a line minimization, but there are an infinite number of gradient directions.

4.1 Arcing Minimization Algorithm

To find the coefficients of a sum of trees that minimizes

$$E_{\mathbf{X}}\theta(\textstyle\sum_1^\infty c_m h_m(\mathbf{x}),\mathbf{x})$$

where $\theta(s,\mathbf{x})$ is real-valued, continuous and differentiable in s for each value of **x**, proceed as follows--After M steps:

i) Compute the minimum over all trees $h(\mathbf{x})$ *of*

$$E_{\mathbf{X}}[\theta_s(\textstyle\sum_1^M c_m h_m(\mathbf{x}),\mathbf{x})h(\mathbf{x})]$$

where θ_s *is the partial with respect to s. Denote a minimizing tree by* $h_{M+1}(\mathbf{x})$.

ii) Find the α *that minimizes*

$$E_{\mathbf{X}}\theta(\textstyle\sum_1^M c_m h_m(\mathbf{x})+\alpha h_{M+1}(\mathbf{x}),\mathbf{x})$$

and let c_{M+1} *equal the minimizing value of* α.

iii) Repeat until convergence.

Re i), it is simple to show that there exist minimizing trees. Uniqueness is difficult and may not hold.

As pointed out in Breiman[1997], this algorithm can be implemented with finite data and reweighting of the training set.

4.2 Heuristic Proof of Convergence

A sketch of a convergence proof is given with strong assumptions When the algorithm is applied in the next section to Adaboost, the assumptions will be lifted.

Assume:

i) *The domain of* $\mathbf{x}$ *is a closed finite rectangle.*

ii) *The functional* $E_{\mathbf{X}}\theta(f(\mathbf{x}),\mathbf{x})$ *is strictly convex on the set of functions in* $L_2(P)$

iii) *The sequence of functions* $s_M(\mathbf{x})=\sum_1^M c_m h_m(\mathbf{x})$ *is sequentially compact in* $L_2(P)$.

iv) *The function* $\theta(s,\mathbf{x})$ *and its first and second derivatives with respect to* s *are uniformly bounded.*

From ii) there is a unique function $s(\mathbf{x})$ minimizing $E_{\mathbf{X}}\theta(s(\mathbf{x}),\mathbf{x})$.

Theorem 2 *Under the above assumptions, the sequence* $s_M(\mathbf{x})$ *converges in* $L_2(P)$ *norm to* $s(\mathbf{x})$.

proof: Suppose that the minimum value over all trees h of

1) $$E_{\mathbf{X}}[\theta_s(s_M,\mathbf{x})h(\mathbf{x})]$$

is zero. Then the maximum value is also zero since the trees can be reversed. Since linear combinations of trees are dense in $L_2(P)$, then

$$E_{\mathbf{X}}[\theta_s(s_M,\mathbf{x})f(\mathbf{x})]=0$$

for all functions f in $L_2(P)$. This implies that for small ε and any f

$$E_{\mathbf{X}}[\theta(s_M+\varepsilon f(x),\mathbf{x})]\approx 0.$$

These are the first order necessary conditions for a minimum, The second order are automatically satisfied by the convexity. Thus, s_M equals the minimizing function s.

If the iterations do not stop, let h_{M+1} be the minimizing tree at step M+1, and $-b_{M+1}$ the minimum value of 1). Now

2) $$E_{\mathbf{X}}[\theta(s_M+\alpha h_{M+1},\mathbf{x})]\leq\theta_0-\alpha b_{M+1}+B\alpha^2$$

where B is the upper bound of the 2nd partial of θ and $\theta_0=E_{\mathbf{X}}\theta(s_M,\mathbf{x})$. From 2) it follows that the decrease in $E_{\mathbf{X}}\theta(s_M,\mathbf{x})$ given by the (M+1)st step is greater than $(b_{M+1})^2/2B$. Since the sum of the decreases must be finite, $b_M\to 0$.

Take a subsequence s_m that converges in 2-norm to a function g (assumption iii). For any tree h,

$$E_{\mathbf{X}}[\theta_s(s_m,\mathbf{x})h(\mathbf{x})]\to 0\ .$$

At the same time

$$E_{\mathbf{X}}[\theta_s(s_m,\mathbf{x})h(\mathbf{x})]\to E_{\mathbf{X}}[\theta_s(g,\mathbf{x})h(\mathbf{x})]$$

This gives the necessary conditions for the minimum and implies that $g=s$. There, the entire sequence s_M converges in norm to s.

However, the hard thing is showing the sequential compactness of the $\{s_M\}$ sequence--a difficulty we have assumed away.

5 **Adaboost Converges to the Bayes Risk**

For a sequence of trees $\{h_m\}$ define a voting function as

$$\phi(\mathbf{x})=\sum_m c_m h_m(\mathbf{x})$$

and vote for class 1 if $\phi>0$, else for class -1. As noted in Section 3, the expected error for ϕ is

$$L^*(\phi)=\int I(\phi(\mathbf{x})\leq 0)P(1|\mathbf{x})P(d\mathbf{x})+\int I(\phi(\mathbf{x})>0)P(-1|\mathbf{x})P(d\mathbf{x}).$$

This is not a pleasant functional to try and minimize. Using the inequalities

3) $$I(\phi\leq 0)\leq\exp(-\phi),\quad I(\phi>0)\leq\exp(\phi)$$

gives $L^*(\phi) \le L(\phi)$ where

4) $$L(\phi)=\int \exp(-\phi(\mathbf{x}))P(1|\mathbf{x})P(d\mathbf{x})+\int \exp(\phi(\mathbf{x}))P(-1|\mathbf{x})P(d\mathbf{x}).$$

This functional has a more workable form and is the functional that Adaboost minimizes using the arcing minimization algorithm given in Section 4.

Two assumptions are made:

i) The domain of $\mathbf{x}$ *is a closed finite D-dimensional rectangle.*
ii) The function $\log(p(-1|\mathbf{x})/p(1|\mathbf{x}))$ *is continuous on this rectangle.*

Let

$$r(\mathbf{x})=\log(p(-1|\mathbf{x})/p(1|\mathbf{x}))/2.$$

Then 4) can be rewritten as

$$L(\phi)=\int \cosh(\phi(\mathbf{x})+r(\mathbf{x}))[\sqrt{p(1|\mathbf{x})p(-1|\mathbf{x})}]P(d\mathbf{x})$$

Let the probability Q be defined by

$$Q(d\mathbf{x})=[\sqrt{p(1|\mathbf{x})p(-1|\mathbf{x})}]P(d\mathbf{x})/\int[\sqrt{p(1|\mathbf{x})p(-1|\mathbf{x})}]P(d\mathbf{x})$$

Up to a constant factor, then,

5) $$L(\phi)=\int \cosh(\phi(\mathbf{x})+r(\mathbf{x}))Q(d\mathbf{x})$$

Note that P and Q differ only by multiplication by a bounded positive function.

Apply the arcing algorithm to 5) building up a sum of trees for ϕ.

$$s_M(\mathbf{x})=\sum_1^M c_m h_m(\mathbf{x})$$

Theorem 3 $s_M(\mathbf{x})$ *converges in* $L_2(P)$ *norm to* $-r(\mathbf{x})$.

The proof is not difficult but has various details. It is deferred to Appendix I.

Proposition 4 *The error in using* $-r(\mathbf{x})$ *as a decision function is the Bayes risk.*

proof: Look at the loss

$$L(\phi)=\int I(-r(\mathbf{x})\le 0)P(1|\mathbf{x})P(d\mathbf{x})+\int I(-r(\mathbf{x})>0)P(-1|\mathbf{x})P(d\mathbf{x})$$

A simple computation shows that

$$L(\phi)=\int \min(P(-1|\mathbf{x}),P(1|\mathbf{x}))P(d\mathbf{x})$$

which is the Bayes risk.

The condition T>D is needed for r(**x**) to be equal to the limit of sums of trees. If r(**x**) can be well-approximated by sums of smaller trees, then the arcing algorithm using smaller trees will also converge to the Bayes rate. This may help to understand why Adaboost using stumps sometimes gives excellent results.

An open question is whether there are other algorithms of the arcing type that converge to the Bayes risk. I conjecture that there are many. Instead of using the exponential function in the bounds 3), one can use any monotonically increasing function g that is one at the origin. Then in 4) replace $\exp(-\phi(\mathbf{x}))$ by $g(-\phi(\mathbf{x}))$ and $\exp(\phi(\mathbf{x}))$by $g(\phi(\mathbf{x}))$. If one can show that the sequences of weighted sums of trees produced by the arcing algorithm applied to this new loss function are sequentially compact, then it would follow that these sequences converge to a function yielding the Bayes risk.

The device used to prove convergence in the exponential case does not generalize to other functions--at least not easily. But I do not see why sequential compactness should not hold for other functions beside the exponential. If so, this may shed light on why, in the finite data case, minimizing functions other than the exponential have given results comparable to Adaboost. See, for instance, Mason et al[2000], Friedman et al[1998], and Breiman[1998].

6. **Random Forests**

Random forests are formed by selecting i.i.d random vectors $\theta_1,\theta_2,,,$from a parent distribution and forming trees $h(\mathbf{x},\theta_m)$ that depend on the value of the vector θ_m. After forming M trees, the vote for y at **x** is defined as

$$\frac{1}{M}\sum_1^M I(y=h(\mathbf{x},\theta_m))$$

As $M\to\infty$ the limiting value, a.s. in **x,** of the vote for y is $P_\theta(y=h(\mathbf{x},\theta))$

where P_θ is the probability under the parent distribution of θ (see Breiman[1999]).

Although the tree construction is randomized, at each terminal rectangle R of each tree, the decision whether to label R +1or -1 is made as follows:

Labeling Rule *Label R as +1 if*

$$\int_R P(1|\mathbf{x})P(d\mathbf{x}) \geq \int_R P(-1|\mathbf{x})P(d\mathbf{x}).$$

Otherwise as -1.

This rule corresponds to the finite data decision rule. Let $D(\mathbf{x})=P(1|\mathbf{x})-P(-1|\mathbf{x})$. For each θ let $R_{\mathbf{x}}(\theta)$ be the terminal node containing $\mathbf{x}$. Then the vote for y=1 is

$$P_\theta\left(\int_{R_{\mathbf{x}}(\theta)} D(\mathbf{z})P(d\mathbf{z}) \geq 0\right)$$

and the vote for -1 is

$$P_\theta\left(\int_{R_{\mathbf{x}}(\theta)} D(\mathbf{z})P(d\mathbf{z}) < 0\right)$$

If the vote for +1 is >1/2, then the prediction is +1, else -1. Hence the error is

$$L=\int I\left(P_\theta\left(\int_{R_{\mathbf{x}}(\theta)} D(\mathbf{z})P(d\mathbf{z}) \geq 0\right) \leq 1/2\right)P(1|\mathbf{x})P(d\mathbf{x}) +$$
$$\int I\left(P_\theta\left(\int_{R_{\mathbf{x}}(\theta)} D(\mathbf{z})P(d\mathbf{z}) < 0\right) \leq 1/2\right)P(-1|\mathbf{x})P(d\mathbf{x})$$

Now

$$P_\theta\left(\int_{R_{\mathbf{x}}(\theta)} D(\mathbf{z})P(d\mathbf{z}) \geq 0\right) \leq 1/2 \Leftrightarrow \mathit{median}_\theta\left[\int_{R_{\mathbf{x}}(\theta)} D(\mathbf{z})P(d\mathbf{z})\right] \leq 0$$

Replace the median by the mean; i.e. assume that

$$\mathit{median}_\theta\left[\int_{R_{\mathbf{x}}(\theta)} D(\mathbf{z})P(d\mathbf{z})\right] \approx E_\theta\left[\int_{R_{\mathbf{x}}(\theta)} D(\mathbf{z})P(d\mathbf{z})\right].$$

This approximation is probably accurate, but difficult to rigorously justify. Let $K(\mathbf{x},\mathbf{z})$ be the probability, under P_θ that $\mathbf{x}$ and $\mathbf{z}$ are in the same terminal node. That is, let

6) $$K(\mathbf{x},\mathbf{z})=\lim_N \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{T} I(\mathbf{x},\mathbf{z}\in R_k(\theta_n))$$

(see Appendix II for proof of convergence). Then

$$E_\theta[\int_{R_{\mathbf{x}}(\theta)} D(\mathbf{z})P(d\mathbf{z})]=\int K(\mathbf{x},\mathbf{z})D(\mathbf{z})P(d\mathbf{z}).$$

Kernel Properties

i) $K(\mathbf{x},\mathbf{z})$ *is self-adjoint.*
ii) $K(\mathbf{x},\mathbf{x})=1$
iii) $K(\mathbf{x},\mathbf{z})$ *is positive definite.*
iv) $K(\mathbf{x},\mathbf{z})$ *does not depend on the class label of the terminal nodes*
v) $K(\mathbf{x},\mathbf{z})$ *is continuous under weak conditions.*

The first four properties can be seen from 6). For the fifth, see Appendix II.

The error is now represented as:

7) $$L=\int I(\int K(\mathbf{x},\mathbf{z})D(\mathbf{z})P(d\mathbf{z})\le 0)P(1|\mathbf{x})P(d\mathbf{x})+$$
$$\int I(\int K(\mathbf{x},\mathbf{z})D(\mathbf{z})P(d\mathbf{z})>0)P(-1|\mathbf{x})P(d\mathbf{x}).$$

Random forests are thus revealed as kernel constructors. There is an alternative procedure in which the kernel representation 7) is exact and not an approximation. Instead of voting the forest, some researchers, i.e. Amit and Geman[1997], grow random trees, each one of which gives an estimate for the probability that **x** belongs to class j. These estimates are averaged over all trees in the forest and **x** assigned to the class having the highest averaged probability. For this averaging procedure representation 7) is exact.

I have experimented on a variety of data sets using both voting and averaging and found little difference in generalization error between them.

7 A Kernel Example

To understand what is going on, it's useful to have a sense of what random forest kernels look like. A exact expression for a kernel can be gotten under the assumptions of a completely random construction:

Kernel Assumptions

i) The space of features is the M-dimensional unit cube, i.e. $[0,1]^M$.
ii) The probability measure is uniform on the cube
iii) At each stage a random choice is made of which node to split next
iv) The split variable m is randomly selected from the M candidates.
v) The split point is uniform on the length of the mth side of the node.

Stating the result requires some definitions. Let $S(k,w)$ be the probability that a Poisson variable with parameter w is greater than or equal to k and $P(k_1,...,k_M|K)$ a multinomial distribution for K trials with M outcomes at each trial. each outcome having probability 1/M. Take K to have the distribution of the sum of T-1 Bernoulli trials such that the probability of success at the Nth trial is 1/n. Then

Theorem 4 *Under the above assumptions, the kernel* $K(\mathbf{x},\mathbf{z})$ *equals*

8) $$\sum_K \sum_{k_1,...,k_M} P(k_1,...,k_M|K) \prod_{m=1}^{M} S(k_m,w_m) P(K)$$

where $w_m = \log(1/|x_m - z_m|)$.

Proof: See Appendix III.

This kernel is symmetric in the $|x_m - z_m|$, that is, permuting their values leaves the kernel unchanged. The kernel formula given in Theorem 4 is complex and not enlightening as to the form of the kernel. But an informative approximation can be made:

Corollary 5 For $M \geq 5$ and $T \leq \exp(M/2)$,

9) $$K(\mathbf{x},\mathbf{z}) \approx \exp(-\lambda \sum_1^M |x_m - z_m|)$$

where $\lambda = \log(T)/M$.

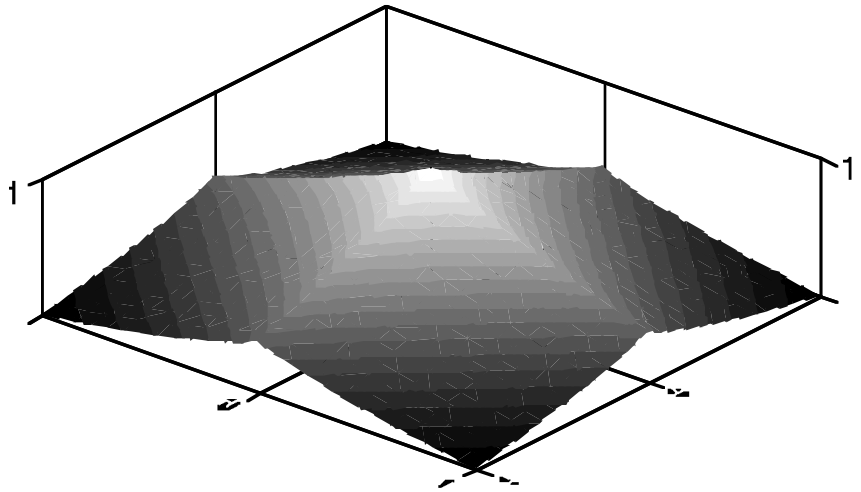
For the derivation see Appendix III. The parameter λ has the critical role in determining sharpness.

The level curves such that $K(\mathbf{x},\mathbf{z}) = h$ have the form

$$\sum_1^M |x_m - z_m| = -\log(h)/\lambda$$

The expression of the left is the city-block distance between **x** and **z.** Thus, the kernel centered at **z** has a pyramidal shape coming to a point at **z**, with the pyramid oriented sat 45% angles to the coordinate axes. Note that the smaller λ is, the wider the pyramid. Figure 1) is a graph of the kernel 8) for M=2,T=3.

Figure 1



In general, the sharper the kernel, the better. In the limit of sharpness, the loss in 7) goes to the Bayes risk. Sharpness is governed by the number of terminal nodes--the more there are, the sharper the kernel. This may help explain why the best results are gotten in random forests using the largest trees.

We note that Cutler [1999] derived an analytic expression for the classifiers given by ensembles they generated using extreme randomization, although not quite as random as assumed above. Interestingly, the derivation is in the finite data case.

8 <u>Correlation~Symmetry, Strength~Skewness</u>

Correlation and strength are concepts that have been useful in interpreting the behavior of random forests (Amit and Geman[1997]. Breiman[1999], Amit et al [1999], Cutler[1999]). Correlation is the average correlation between the margins of two different members of the forest. Strength is the average correct classification rate of the classifiers in the forest.

One can get a bound on the misclassification error of the forest in terms of correlation and strength, but it is a loose bound. On the other hand, keeping track of correlation and strength while varying the amount of randomness in the forest shows that the bound is closely connected with the generalization error.

The question is how these concepts translate or are translated from properties of the kernel. We advance the conjecture that they are reflections of the symmetry and skewness of the kernel, and illustrate this with some discussion and experimental results.

Empirically, random forests give accurate classification even in high dimensional problems. The reason for this is that the kernel can be a poor estimate of $D(\mathbf{z})$ as long as the error in the estimates of the boundary (the $D(\mathbf{z})=0$ contours) does not cause much of an error increase above the Bayes risk.

We argue that if the change in $D(\mathbf{x})$ moving across the $D(\mathbf{x})=0$ boundaries is not abrupt, the estimated boundary error is not large and the increase in error small.

First: If the forest is completely random i.e. if the splits have no association with the values of $D(\mathbf{z})$ in the nodes, then the kernel will tend to be symmetric in the sense that for fixed $\mathbf{z}$, the kernel will be a function of d($\mathbf{x}$-$\mathbf{z}$) where this function depends only on $|x_m - z_m|$. Then the kernel will be unbiased at linear boundaries in the sense that if a good approximation to $D(\mathbf{z})$ near a point $\mathbf{x}$ at which $D(\mathbf{x})=0$ is given by

$$D(\mathbf{z})=(\mathbf{x}-\mathbf{z})\cdot\nabla D(\mathbf{x})$$

then

$$\int K(\mathbf{x},\mathbf{z})D(\mathbf{z})P(d\mathbf{z})\approx 0$$

and the kernel will track the sign of $D(\mathbf{z})$ near the boundary.

Second: If S is the region such that:

8) $$\int K(\mathbf{x},\mathbf{z})D(\mathbf{z})P(d\mathbf{z}) > 0$$

and S^* the region such that $D(\mathbf{x})>0$ then the rise in error over the Bayes risk is less than:

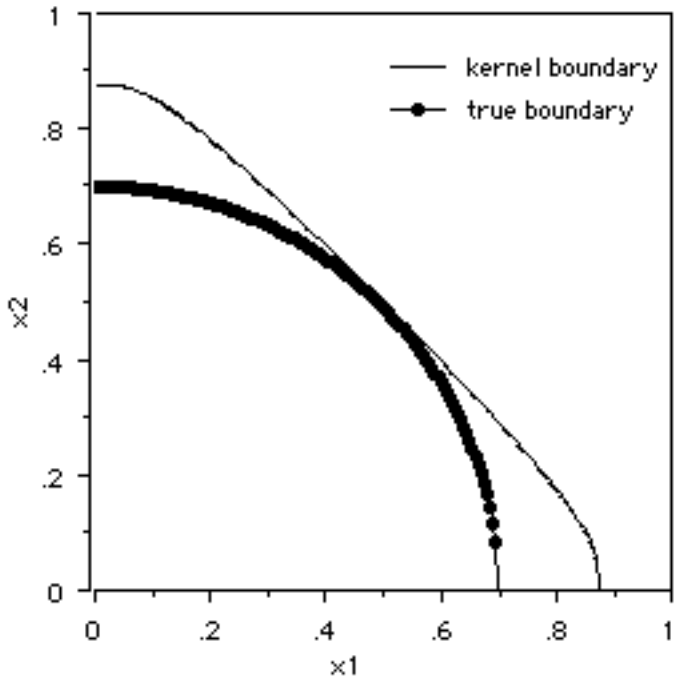
(9) $$\int_{S^*\nabla S} |D(\mathbf{z})| P(d\mathbf{z})$$

where $S^*\nabla S$ is the symmetric difference between the two regions.

Now $S^*\nabla S$ is the region between the estimated boundary and the true boundary. Since $D(\mathbf{z})=0$ on the true boundary, then if $D(\mathbf{z})$ is not rapidly changing in the neighborhood of the boundary, and if the boundary estimate does not differ too much from the true boundary, the increase in error will be small.

To illustrate, we ran the kernel 8) in the two dimensional square with three terminal nodes. Two Gaussions centered at (0,0) are used, one with standard deviation .71 and the other with sd 1.36. Both are normalized to have probability .5 on the square. The Bayes rate is 39.9%. The true boundaries and the boundaries given by the kernel are shown in Figure 2.

Figure 2

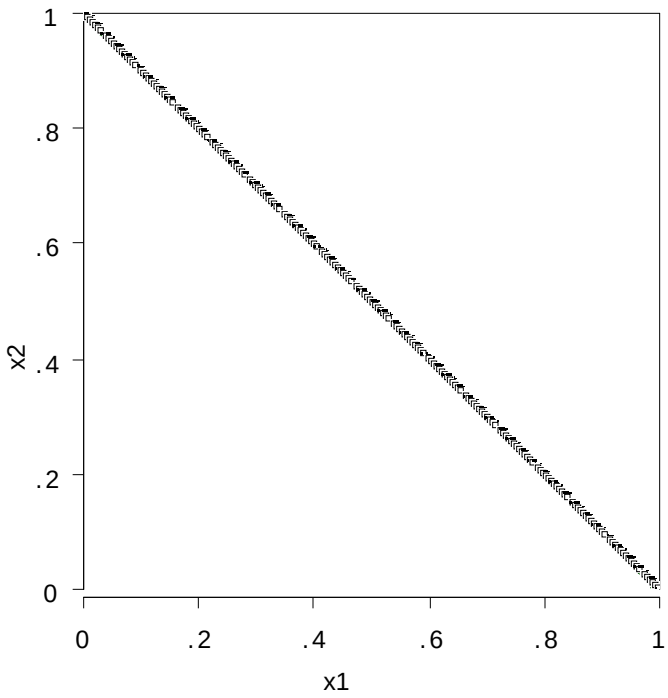


The boundaries do not match, but error rate given by the kernel is only .6% larger than the Bayes rate.

The problem, then, is with more abrupt changes near the boundary. However, if $D(\mathbf{z})$ is linear near the $D(\mathbf{z})=0$ boundary then the average over the kernel will be zero. To illustrate we again ran the kernel 8) with T=3 on the two dimensional problem with $P(1|\mathbf{x})=1$ if $x_1+x_2 \le 1$ and zero elsewhere.

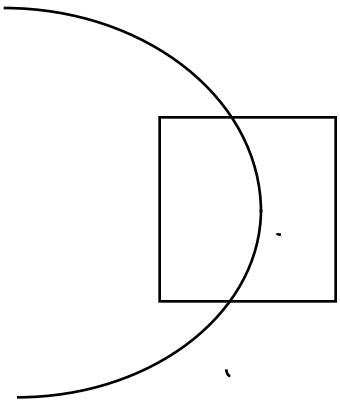
The resulting boundary is shown in Figure 3 together with the true boundary, Since they lie on top of each other, only one line is seen. The rise in error over the Bayes risk is .3%.

Figure 3



But if the boundary is abrupt and non-linear as in this picture:

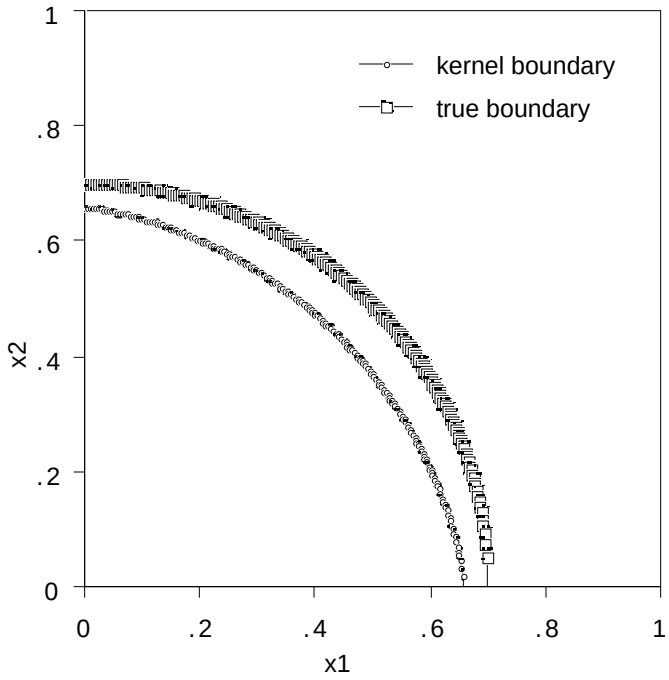
Figure 4



then the kernel average will not be zero on the $D(\mathbf{z})=0$ curve, and there will be a bias toward class #-1 instances.

To illustrate this situation, we used T=3 and $P(1|\mathbf{x})=1$ if $x_1^2+x_2^2 \leq .49$ and zero elsewhere. The resulting boundary and the true boundary are shown in Figure 5. The increase in error over the Bayes risk is 10.3%.

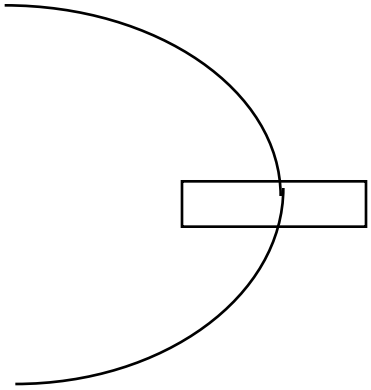
Figure 5



In the paper by Cutler1999], an extreme degree of randomization was used resulting in low correlation and low strength. The results on almost all data sets were competitive with Adaboost. But on the ringnorm data where the boundary is spherical and somewhat abrupt, the generalization error was well above that of algorithms that use more strength. The examples above may be a clue to their results.

To eliminate this curvature bias, the kernel needs to be skewed. If an element is introduced into the construction of the trees in the forest that favors purer nodes, then **x,z** will tend to occur more frequently together if they are of the same class. Thus, the kernel will tend to be skewed higher for instances **x**,**z** which are both in neighborhoods of higher positive D values or lower negative values. This is illustrated in Figure 6.

Figure 6

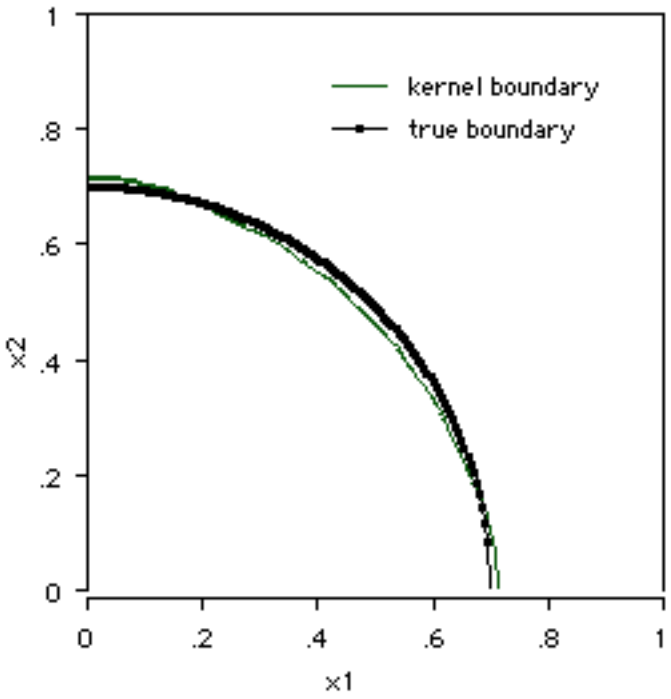


Thus, skewing the kernel will eliminate some of the bias due to curved boundaries.

To illustrate this, some strength or skewness was added was added and tested on the example with T=3 and $P(1|\mathbf{x})=1$ if $x_1^2+x_2^2 \leq .49$, zero otherwise. Similarities with the symmetric kernel are random choice of node to split and random choice of variable to split on. The difference is that instead of the split being uniformly distributed, the distribution was altered to favor cuts near the optimal split. The optimal split of a node is the split that minimizes the misclassification error. A triangular distribution was used for cut selection with peak at the optimal cut and dropping to zero at the endpoints of the node.

Since no analytic expression could be found for the kernel corresponding to this regime, it was programmed on a 200x200 grid and run 5000 times. The error dropped from 10.3% to 1.5%. The boundaries are graphed in Figure 7.

Figure 7



A good kernel for a problem has to balance symmetry against the skewness needed to cope with nonlinear boundaries. More research is necessary in this area. An eigen analysis of the kernel may be useful. A "perfect kernel" would reproduce a constant multiple of D(z), but it is not at clear how this could be accomplished by tree construction.

9. **Discussion**

The question is how much of the infinity theory applies to the practical issues of finite data sets. There are some puzzling issues.

9.1 Adaboost

The theoretical results indicate that as the sample size goes to infinity, the generalization error of Adaboost will converge to the Bayes risk. But on most data sets I have run, Adaboost does not converge. Instead it's behavior resembles an ergodic dynamical system. The mechanism producing this behavior is not understood.

Another facet of Adaboost's behavior is that it is more of an equalizer. In Adaboost, the instances most recently misclassified are more heavily weighted to get the next classifier grown to focus on classifying them correctly.

For some time it was believed that the reason for the success of this reweighting was to get the hard-to-classify instances classified correctly.

Recent work has shown that this is not the case. Adaboost is, instead, trying to roughly equalize the proportion of weighted votes that are incorrect. That is, it is trying to equalized the weighted fraction of times that each instance is misclassified. Put another way, it is approximately equalizing the margins of all of the instances. The weighting more heavily of instances recently misclassified increases their proportion of correct classifications and when they are brought into line Adaboost concentrates on other instances.

I also conjecture that it is this equalization property that gives Adaboost its ergodicity. Consider a finite number of classifiers $\{h_n\}$, each one having having an associated misclassification set Q_n. At each iteration the Q_n having the lowest weight (using the current normed weights) has its weight inceased to 1/2 while instances in the complement have their weights decreased. Thus, the Q_n selected moves to the top of the weight heap while other Q_n move down until they reach the bottom of the heap, when they are bounced to the top. It is this cycling among the Q_n that produces the ergodic behavior.

However, I do not understand the connection between the finite sample size equalization and what goes on in the infinity case. Why equalization combined with ergodicity produces low generalization error is a major unsolved problem in Machine Learning.

9.2 Random Forests

Numbers of years ago, efforts were made to do classification doing kernel density estimation. It didn't work out. Random forests also constructs a kernel, but a different kind. It is not designed to estimate densities, but to locate boundaries. Its appearance, through the medium of random forests, is a bit of a surprise.

With random forests, there is a stronger tie to the infinity behavior. The behavior in the finite sample and infinity situations appears more connected. Adjusting strength and correlation gives the results that one would expect from the infinity analysis. More work needs to be done. It would be satisfying to have a theoretical expressions for the kernel when some strength is applied to see how the skewness sets in.

Another puzzle is behavior for large dimension. Random forests has been run on data sets with intrinsically large dimensionality i.e. 256 and 1000, and has given low generalization error. The sample sizes are thin compared to the dimensionalty, so the kernel spread, given the number of terminal nodes,

is large. Still, the results on locating boundaries in two dimensions given a relatively large kernel spread, show that boundary determination may not be all that sensitive to kernel spread.

References

Amit, Y. and Geman, D. [1997] Shape quantization and recognition with randomized trees, Neural Computation 9,1545-1588

Amit, Y., Blanchard, G., and Wilder, K. [1999] Multiple Randomized Classifiers: MRCL Technical Report, Department of Statistics, University of Chicago

Bauer, E. and Kohavi, R. [1999] An Empirical Comparison of Voting Classification Algorithms, Machine Learning, 36, No. 1/2, 105-139

Bhulmann, P. and Yu, B. [2000] Explaining bagging, available at www.stat.Berkeley.EDU/users/binyu/publications.html

Breiman,L.[1999] Prediction Games and Arcing Algorithms, Neural Computation, 11, 1493-1517

Breiman, L. [1998], Arcing Classifiers, (discussion paper) Annals of Statistics, 26, 801-824

Breiman, L. [1997] Arcing the Edge, Technical Report 486, Statistics Department, University of California (available at www.stat.berkeley.edu)

Breiman. L.[1998a] Randomizing Outputs To Increase Prediction Accuracy. Technical Report 518, May 1, 1998, Statistics Department, UCB (in press Machine Learning)

Breiman,L. [1999] Random Forests, Technical Report, Statistics Department UCB (available at www.stat.berkeley.edu)

Breiman, L. [1996], Bagging Predictors, Machine Learning, 26, No. 2, 123-140

Cutler, A. (1999). Fast Classification Using Perfect Random Trees. Technical Report 5/99/99, Department of Mathematics and Statistics, Utah State University

Drucker, H. and Cortes, C. [1995] Boosting decision trees, Advances in Neural Information Processing Systems Vol. 8, pp. 479-485.

Dietterich, T. [1998] An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting and Randomization, Machine Learning 1-22

Freund, Y. and Schapire, R. [1996] Experiments with a new boosting algorithm, Machine Learning: Proceedings of the Thirteenth International Conference, pp. 148-156

Friedman,J., Hastie,T., and Tibshirani,R.[1998] Additive logistic regression: a statistical view of boosting. Technical Report, Statistics Department, Stanford University (to be published, Annals of Statistics)

Mason,L., Baxter,J.,Bartlett,P. and Frean,M. [2000] Boosting algorithms as gradient descent. Advances in Neural Information Processing Systems, 12, 512-518

Schapire, R. and Singer,Y.[1998] Improved boosting algorithms using confidence-rated predictions, Proceedings of the Eleventh Annual Conference on Computational Learning Theory

Appendix 1. Proof of Theorem 3.

a) If the arcing algorithm stops at a finite M, then

$$\int \sinh(r(\mathbf{x})+s_M(\mathbf{x}))h(\mathbf{x})Q(d\mathbf{x})=0$$

for all trees h. Hence for all functions f in $L_2(P)$

$$\int \sinh(r(\mathbf{x})+s_M(\mathbf{x}))\, f(\mathbf{x})Q(d\mathbf{x})=0,$$

which implies that $s_M(\mathbf{x})$ equals $-r(\mathbf{x})$ a.s.

b) If the algorithm continues indefinitely, let

$$b_{M+1}=-\inf_h \int \sinh(r(\mathbf{x})+s_M(\mathbf{x}))h(\mathbf{x})Q(d\mathbf{x})$$

and h_{M+1} a minimizing tree. Some algebra show that the α minimizing

$$\int \cosh(r(\mathbf{x})+s_M(\mathbf{x})+\alpha h_{M+1}(\mathbf{x}))Q(d\mathbf{x})$$

is:

1)
$$c_{M+1}=\frac{1}{2}\log((1+b_{M+1})/(1-b_{M+1}))$$
$$=b_{M+1}+O(b_{M+1}^2).$$

Denoting

$$I_M=\int \cosh(r(\mathbf{x})+s_M(\mathbf{x}))Q(d\mathbf{x})$$

then

$$I_{M+1}^2=I_M^2-b_{M+1}^2$$

Since I_M is bounded below, this implies that $\sum_1^\infty b_m^2<\infty$. We will use the inequality

2) $$|\int \sinh(r(\mathbf{x})+s_M(\mathbf{x}))s_M(\mathbf{x})Q(d\mathbf{x})| \le b_{M+1}\sum_1^M c_m .$$

c) There is a subsequence {m'} such that $\|r+s_{m'}\|\to 0$. To show this, note:

3) $$(b_{M+1}\sum_1^M c_m)^2 \le M b_{M+1}^2 \sum_1^M c_m^2$$

Suppose $\liminf_M (b_{M+1}\sum_1^M c_m)>0$, then since $\sum_1^\infty c_m^2<\infty$, for all M sufficiently large, $b_{M+1}^2 > a/M$, which cannot be. Thus, there is a subsequence such that the left hand side of 2) goes to zero.

For f any finite sum of trees, $\int \sinh(r(\mathbf{x})+s_M(\mathbf{x}))f(\mathbf{x})Q(d\mathbf{x})\to 0$. Since r(**x**) is continuous and bounded, for any $\varepsilon>0$ there is a finite sum of trees $\tilde{r}(\mathbf{x})$ such that $\sup_{\mathbf{x}\in S}|r(\mathbf{x})-\tilde{r}(\mathbf{x})|\le\varepsilon$. Hence

4) $$|\int \sinh(r(\mathbf{x})+s_M(\mathbf{x}))r(\mathbf{x})Q(d\mathbf{x})| \le |\int \sinh(r(\mathbf{x})+s_M(\mathbf{x}))\tilde{r}(\mathbf{x})Q(d\mathbf{x})| + \varepsilon\int |\sinh(r(\mathbf{x})+s_M(\mathbf{x}))|Q(d\mathbf{x})$$

The last term in 4) can be bounded by

$$\varepsilon\int \cosh(r(\mathbf{x})+s_M(\mathbf{x}))Q(d\mathbf{x}) \le C\varepsilon,$$

leading to the conclusion that $\int \sinh(r(\mathbf{x})+s_M(\mathbf{x}))r(\mathbf{x})Q(d\mathbf{x})\to 0$. Thus,

$$\int \sinh(r(\mathbf{x})+s_{m'}(\mathbf{x}))(r(\mathbf{x})+s_{m'}(\mathbf{x}))Q(d\mathbf{x})\to 0.$$

Since $x\sinh(x)\ge x^2$, this proves assertion c).

d) On the full sequence $\|r+s_M\|\to 0$. This follow by noting that $x\sinh(x)\ge\cosh x-1$. Hence, on the subsequence $I_{m'}\to 1$. Since I_M is non-increasing in M, on the whole sequence $I_M\to 1$. Now use $\cosh x-1\ge x^2$ to prove d).

Appendix II Continuity and Convergence

For a tree formed using the value θ of the random vector, let $B(\theta)$ be the boundaries of all of the terminal nodes in the tree that are in the interior of the domain of inputs.. For any **x** denote by $d(\mathbf{x},B(\theta))$ the Euclidean distance from **x** to the boundary. We impose on the tree formation process a continuity condition.

Condition C $\quad \nu(\varepsilon)=\sup_{\mathbf{x}} P_\theta(d(\mathbf{x},B(\theta))<\varepsilon) \to 0 \text{ as } \varepsilon \to \Theta$

This condition keeps mass from piling up on the boundaries of the terminal nodes.

Let Θ be the space of sequences $\underset{\sim}{\theta}=(\theta_1,\theta_2,...)$ of the random vectors. Then:

Theorem: *There is a set* $S\subset\Theta$ *with* $P(S)=1$ *such that for any sequence in S, for all* $\mathbf{x},z$,

$$\frac{1}{N}\sum_{n=1}^{N}\sum_{k} I(\mathbf{x}\in R_k(\theta_n))I(\mathbf{z}\in R_k(\theta_n)) \to K(\mathbf{x},\mathbf{z})$$

where $K(\mathbf{x},\mathbf{z})$ *is jointly continuous in* $\mathbf{x},z$,

Proof: Denote

$$W_N(\mathbf{x},\mathbf{z},\underset{\sim}{\theta})=\frac{1}{N}\sum_{n=1}^{N}\sum_{k} I(\mathbf{x}\in R_k(\theta_n))I(\mathbf{z}\in R_k(\theta_n))$$

Select S as follows: Take D to be countable and dense in the input domain. Then let S be all sequences $\underset{\sim}{\theta}$ such that

i) For all $\mathbf{x},\mathbf{z}\in D\otimes D$

$$W_N(\mathbf{x},\mathbf{z},\underset{\sim}{\theta}) \to E_\theta(\mathbf{x},\mathbf{z}\in R(\theta))=K(\mathbf{x},\mathbf{z})$$

ii) *Take* $\varepsilon_n \downarrow 0$. *For all* $\mathbf{x}\in D$ *and all* ε_n

$$\frac{1}{N}\sum_{n=1}^{N}\sum_{k} I(d(\mathbf{x},B(\theta_n))<\varepsilon_n) \to P_\theta(d(\mathbf{x},B(\theta))<\varepsilon_n)$$

Since both conditions are countable, they can be satisfied on a set S of probability one.

First: on $D\otimes D$, $K(\mathbf{x},\mathbf{z})$ is a continuous function. To see this, take $\mathbf{x}_n,\mathbf{z}_n \to \mathbf{x},\mathbf{z}$ in $D\otimes D$ such that $d(\mathbf{x},\mathbf{x}_n)\le\varepsilon_n$ and $d(\mathbf{z},\mathbf{z}_n)\le\varepsilon_n$. Use the inequality

$$|I(\mathbf{x}\in R_k(\theta))I(\mathbf{z}\in R_k(\theta))-I(\mathbf{x}_n\in R_k(\theta))I(\mathbf{z}_n\in R_k(\theta))|\le$$
$$I(d(\mathbf{x},B(\theta))<\varepsilon_n)+I(d(\mathbf{z},B(\theta))<\varepsilon_n)$$

So by condition ii)

$$|K(\mathbf{x},\mathbf{z})-K(\mathbf{x}_n,\mathbf{z}_n)|\le P_\theta(d(\mathbf{x},B(\theta))<\varepsilon_n)+P_\theta(d(\mathbf{z},B(\theta))<\varepsilon_n)$$

which proves the continuity.

Second: Now $K(\mathbf{x},\mathbf{z})$ defined on $D\otimes D$ can be extended to a continuous function on the whole rectangle of input space. Consider **x,z** not in $D\otimes D$ and take $\mathbf{x}_n,\mathbf{z}_n\to\mathbf{x},\mathbf{z}$ such that $\mathbf{x}_n,\mathbf{z}_n$ are in $D\otimes D$ with $d(\mathbf{x},\mathbf{x}_n)\le\varepsilon_n$ and $d(\mathbf{z},\mathbf{z}_n)\le\varepsilon_n$. Then using the inequality

$$|I(\mathbf{x}\in R_k(\theta))I(\mathbf{z}\in R_k(\theta))-I(\mathbf{x}_n\in R_k(\theta))I(\mathbf{z}_n\in R_k(\theta))|\le I(d(\mathbf{x}_n,B(\theta))<\varepsilon_n)+I(d(\mathbf{z}_n,B(\theta))<\varepsilon_n)$$

shows that for $\theta'\in S$

$$\limsup_N|W_N(\mathbf{x},\mathbf{z},\theta')-W_N(\mathbf{x}_n,\mathbf{z}_n,\theta')|\le 2\sup_{\mathbf{x}} P_\theta(d(\mathbf{x},B(\theta))<\varepsilon_n).$$

This implies that $W_N(\mathbf{x},\mathbf{z},\theta')\to K(\mathbf{x},\mathbf{z})$.

Appendix III Proof of Theorem 4 and Corollary 5.

Given two points **x** and **z,** the problem is to find the probability that they lie in the same terminal node, given that choices of the node to split on and the variable to split on are random and the splits chosen to be uniformly distributed on the current side of the node being split.

Theorem 4

Let the interval $d=[x_m,z_m]$ and let d stand for both the interval and its length. Suppose that the node containing **x,z** is split k times on the mth variable at the points $y_1,y_2,\ldots,y_k$. Then

1) $$P(y_k\notin d,y_{k-1}\notin d,\ldots,y_1\notin d)=\int_{y_1\notin d}\cdots\int_{y_k\notin d}p(dy_k|y_{k-1})\cdots p(dy_1)$$

Now given y_{k-1}, y_k is uniformly distributed over the interval $[0,y_{k-1}]$. Hence

$$P(y_k\notin d|y_{k-1})=(1-d/y_{k-1}).$$

The next step in the integration yields

$$1-\frac{d}{y_{k-2}}(1+\log(\frac{y_{k-2}}{d}))$$

Continuing for k steps evaluates the integral as

$$1-d\sum_{0}^{k-1}\frac{1}{j!}(\log(\frac{1}{d}))^j$$

Letting $w=\log(1/d)$, the integral is

$$S(k,w)=e^{-w}\sum_{j=k}^{\infty}\frac{w^j}{j!}$$

the probability that a Poisson variable with parameter w is greater than or equal to k. Now, denote $d_m=|x_m-z_m|$ and $w_m=\log(1/d_m)$. Let $P(k_1,...,k_M)$ be the probability that in growing the tree to T terminal nodes, there are $k_1,...,k_M$ splits on the variables 1,...,M on the sides of the terminal node containing **x,z.** Then the probability that none of these splits separates the points is:

2) $$\sum_{k_1,...,k_M}P(k_1,...,k_M)\prod_{m=1}^{M}S(k_m,w_m)$$

If the total number of splits on the node is K, then $P(k_1,...,k_M|K)$ is a multinomial distribution with probabilities 1/M. The distribution of K is that of a sum of T-1 Bernoulli variables such that the nth on has probability 1/n of being one. This results in the statement of Theorem 4.

Corollary 5.

Consider M large, say M>5 and $T\le\exp(M/2)$. If there are currently N terminal nodes, then the probability of selecting the node that **x,z** is in is 1/N. Further, the probability of selecting the mth variable to split on is 1/M. So k_m is the total achieved by T-1 trials, such that at each trial the probability is small. To get an approximation to 2) assume that the $k_1,...,k_M$ are independent Poissons with the same parameter λ.

Now λ is the average probability of success at each trial times the number of trials.. On the first trial, the probability of a hit is 1/M. On the second, 1/(2M). The sum of the probability of hits, totaled over the T-1 trials, is

$$\frac{1}{M}\sum_{N=1}^{T-1}\frac{1}{N}\approx\frac{\log(T)}{M}$$

Therefore, we take

$$\lambda = \frac{\log(T)}{M}.$$

Then, setting

$$U(\lambda, w) = e^{-\lambda} \sum_{0}^{\infty} \frac{1}{k!} \lambda^k S(k, w)$$

results in the approximation to 2):

$$\prod_{m=1}^{M} U(\lambda, w_m).$$

Now:

3) $$U(\lambda, w) = e^{-\lambda - w} \sum_{0}^{\infty} \frac{1}{k!} \lambda^k \sum_{k}^{\infty} \frac{1}{j!} w^j$$

Multiply 3) by $\exp(-ws)$ and integrate on s from 0 to ∞, thereby taking the Laplace transform of V on w and resulting in:

4) $$(e^{-\lambda} e^{\frac{\lambda}{s+1}})/s.$$

Let

$$v(\lambda, w) = e^{-(\lambda + w)} I_O(2\sqrt{\lambda w}).$$

Then 4) is recognizable as the Laplace transform of:

5) $$v(\lambda, w) + \int_{0}^{w} v(\lambda, u) du.$$

Integate the 2nd term in 5) by parts using $I_0(0)=1$ to get

6) $$e^{-\lambda} \int_{0}^{w} e^{-u} I_0'(2\sqrt{\lambda u}) du$$

As $w \to \infty$ 6) $\to 1$. Adding 1, subtracting 1 and taking logs gives that the log of 6) is approximated by

7) $$e^{-\lambda}\int_0^w e^{-u} I_0'(2\sqrt{\lambda u})du - 1$$

Now, approximate $\exp(-\lambda) \approx 1-\lambda$ and argue that since $I_0'(2\sqrt{\lambda u})$ is a slowly increasing function, it can be replaced in the integral by it's value at the origin which is λ. Using these approximations, 7) becomes

8) $$-\lambda e^{-w} - \lambda^2(1-e^{-w})$$

Neglecting the term in λ^2 gives the result that the log of 5) is

9) $$-\lambda d$$

leading directly to Corollary 5).

Since the steps from 5) to 9) seem to involve rather crude approximations, we used numerical integration to evaluate the -log of 5) as a function of d for d ranging from 0 to 1 and three values of λ, .1, .25. .5. The results are shown in Figure III-1 They verify the accuracy of the approximation 9).

Figure III-1