Noname manuscript No.
(will be inserted by the editor)

Bounds on the Moments for an Ensemble of Random Decision Trees

Amit Dhurandhar

Received: Sep. 17, 2013 / Revised: Mar. 04, 2014 / Accepted: Jun. 30, 2014

Abstract An ensemble of random decision trees is a popular classification technique, especially known for its ability to scale to large domains. In this paper, we provide an efficient strategy to compute bounds on the moments of the generalization error computed over all datasets of a particular size drawn from an underlying distribution, for this classification technique. Being able to estimate these moments can help us gain insights into the performance of this model. As we will see in the experimental section these bounds tend to be significantly tighter than the state-of-the-art Breiman's bounds based on strength and correlation and, hence, more useful in practice.

Keywords bounds, random decision trees, moments

1 Introduction

An ensemble of random decision trees is a widely used classification technique [16,13, 22,14] , especially known for its ability to scale to large domains. This classification technique was introduced in [13] where the authors portrayed the efficacy of the method on real applications. According to the description in [13], the technique can be viewed as a special case of the random forest algorithm given in [7], where an attribute is chosen uniformly at random from the available list of attributes (i.e., from an uniform distribution over the available attributes) to form a node in the tree. In a random forest a set of attributes (say m attributes) are chosen uniformly at random as potential candidates for a node and then, based on statistical measures such as information gain (IG), gini gain (GG), etc., an attribute is selected from this set. Hence, an ensemble of random decision trees is the special case of a random forest, where $m = 1$. The primary motivation and the advantage of focusing on this variant is that it is highly scalable, since measures such as IG or GG do not have to be computed at every node in the tree. In addition, the prediction accuracy of this model is shown to be quite good in practice [16,13,22,23].

Amit Dhurandhar
IBM T.J. Watson
E-mail: adhuran@us.ibm.com

Realizing the widespread utility of an ensemble of random decision trees, it is important that we carefully analyze this model. In previous works [11], the authors provided a moment based methodology to accurately characterize classification models. In particular, they provided generic expressions to compute the first two moments of the generalization error – expected error over the entire input, given an underlying probability distribution. These moments were computed over $\mathcal{Z}(N)$, which is the set of all possible classifiers produced by training a given classification algorithm over datasets of size N, drawn independently and identically (i.i.d.) from some distribution. The challenge was then to customize these expressions to specific algorithms in order to meticulously study them. In [11, 10, 12], these expressions were customized for the Naive Bayes Classifier, Nearest Neighbor Classifier and (single) Random decision trees (not an ensemble). It was seen that such expressions have the potential to provide extremely tight characterizations of the behavior of individual classification algorithms as well as certain model selection techniques (viz. cross-validation, hold-out-set estimation). In addition to being accurate, these moments were also very efficient to compute. In [10], one of the future challenges that was laid out was to develop strategies to efficiently and accurately estimate these moments for an ensemble of random decision trees. The strategies and expressions presented in [10] are practical only for single random decision trees, since as we will later see, a straightforward extension of those ideas to an ensemble leads to highly inefficient characterizations.

Estimating moments to evaluate quality of classifiers is a standard procedure in machine learning [4, 5], where a dataset is split into multiple training and test sets, and the average error with a confidence interval or variance is reported as a measure of performance. Our methodology is an idealized version of this approach where rather than reporting the error averaged over a limited number of trials, we want to compute the expected error and maybe in some cases even the variance over all possible classifiers that would be produced by training over datasets of size N sampled from the underlying distribution. This as discussed in previous works [11, 10, 12] is a much more robust evaluation of classification algorithms. Recently, other such moment based methodologies [1–3] have also gained prominence for parameter estimation of models over traditional maximum likelihood and bayesian approaches, which are inefficient and have a tendency to get stuck in poor local minima.

It is important to stress the fact that the generic expressions provided in [11] are difficult to customize to specific learning techniques. The customization requires analyzing the inner workings of the techniques, which are unique to every learning algorithm. The situation here is analogous to PAC Bayes bounds literature [17, 15, 18], where many works have been published describing a way to compute them for different techniques. Moreover, in all the relevant previous works [11, 10, 12], exact and efficient to compute formulations were derived for the respective techniques. For an ensemble of random decision trees though, exact formulations based on extensions of the results in [10] lead to highly inefficient characterizations, which are exponential in the number of trees T. Thus, we in this paper, for the first time in this line of work, describe a novel way to find accurate and efficient to compute upper bounds. Non-trivial and efficient to compute upper bounds can be very useful in gaining confidence in a method [6]. In fact, the gain in time complexity is exponential in T. This is a significant improvement given that T is usually in the hundreds. The strategies employed to derive this bound bear little resemblance to previous analysis. Consequently, the contributions in this paper are by no means attainable by incremental extensions of prior works.

Table 1 Notation used in the paper.

Symbol	Meaning
X	Random vector modeling input.
$\mathcal{X}$	Domain of random vector (input space) X.
Y	Random variable modeling output.
$Y(x)$	Random variable modeling output for input x.
$\mathcal{Y}$	Set of class labels (output space).
N	Sample size
ζ	Classifier
$GE(\zeta)$	Generalization error of classifier ζ.
$\mathcal{Z}(N)$	The set of classifiers obtained by application of a classification algorithm to i.i.d. samples of size N.
$E_{\mathcal{Z}(N)}[]$	Expectation w.r.t. the space of classifiers built on a sample of size N.
d	Dimensionality of the input space.
h	Height of trees in the ensemble.
$path_p$	p^{th} possible path amongst the total number of allowed paths given by the tree building algorithm.
$\xi(path_p y)$	Count of the number of samples in the training set that correspond to $path_p$ and class y.
η_x^y	Denotes the condition that class y gets the maximum number of votes in classifying x through the respective paths in the ensemble.
$m_h(S)$	Number of paths of length h that classify x into class y in the sample S.

To summarize, we provide in this paper a method to efficiently obtain bounds on these moments for an ensemble of random decision trees grown to a specified height. We analyze the fixed height variant, since the main strength of this technique is being able to create an ensemble with high diversity [13]. This is true for most ensemble techniques as we want to cover different parts of the hypothesis space. Growing trees too deep limits this and, hence, restricting the height is desirable. Note that the case where trees are grown to their full height is just a special case in this setting and thus our analysis is still applicable. Moreover, our analysis does not place any restriction on the number of classes or the number of splits per attribute and is hence applicable to a large class of random decision tree ensembles. Through experiments on synthetic data and bootstrap distributions built on real data, we show that our (upper) bounds are tighter than the widely used strength and correlation bounds introduced in [7].

The rest of the paper is organized as follows. In Section 2, we first describe the ensemble algorithm and then briefly review the technical framework that our analysis is based on. In Section 3, we discuss the issues with directly extending the ideas used for characterizing single random decision trees to an ensemble. We then provide a solution by adopting a different perspective than the one in [10]. Moreover, as in the previous study, our analysis applies to categorical as well as continuous attributes with split points predetermined for each attribute. In Section 4, we compare the bounds derived from this analysis with Breiman's strength and correlation bounds on synthetic and real datasets. We discuss future lines of research and summarize the major developments in the paper in section 5.

2 Preliminaries

In this section we first provide a precise description of the algorithm for random decision trees that we analyze here. We then revisit the generic expressions provided in [11] that need to be customized to specific classification algorithms; in this case an ensemble

Algorithm 1 Procedure to build an ensemble of random decision trees.

Input: T, h, $F = \{f_1, ..., f_d\}$ {T is the # of trees in the ensemble, h is the height and the f_is are the d attributes.}
Output: W {The ensemble.}
Set $W = \phi$ {No trees in the ensemble initially.}
for $i = 1; i \leq T; i++$ **do**
  Randomly pick $f_i \in F$ {Pick the root.}
  $w = \{f_i\}$, $w_l = \{f_i\}$ {w denotes the tree built so far and w_l the attributes at the leaf of w.}
  for $j = 1; j < h; j++$ **do**
    $w_t = \phi$
    for $f \in w_l$ **do**
      If f has s splits, then randomly pick s attributes $F_s \subset F$ one for each split such that each attribute is distinct from its ancestors.
      $w_t = w_t \cup F_s$
    end for
    $w_l = w_t$
    $w = w \cup w_l$
  end for
  $W = W \cup w$
end for
Return W

of random decision trees. Finally, we provide the customized expressions for random decision trees (not ensemble) of prespecified height and discuss the intuitions in its derivation.

2.1 Ensemble of Random Decision Trees

In this paper, we consider the following procedure for building each random decision tree in an ensemble of T trees. The root of a tree is chosen uniformly at random from the available list of attributes. Nodes at the next level in the tree are chosen uniformly at random from a list of attributes that excludes the root. Hence, at each level of the tree a node is picked at random (uniform distribution) from the list of attributes that excludes its parents and ancestors. The tree is grown to a prespecified height h. This procedure is clearly elucidated in algorithm 1. It is assumed that the split points for continuous attributes are predetermined. After T trees have been built using the aforementioned procedure to form an ensemble, the class label of a datapoint is determined by taking a majority vote.

The procedure for building random decision trees considered here has been previously mentioned in [13, 10] and is shown to be efficient as well as accurate in practice.

2.2 Generic Expressions

We first introduce some notation that is used primarily in this section. X is a random vector modeling input whose domain is denoted by $\mathcal{X}$. Y is a random variable modeling output whose domain is denoted by $\mathcal{Y}$ (set of class labels). $Y(x)$ is a random variable modeling output for input x. ζ represents a particular classifier with its generalization error (GE) denoted by $GE(\zeta)$. Formally,

$$GE(\zeta) = E[\lambda(\zeta(X), Y)] \tag{1}$$

where $\lambda(.,.)$ is a zero-one loss function and the expectation is over the distribution on the $\mathcal{X} \times \mathcal{Y}$ space. $\mathcal{Z}(N)$ denotes a set of classifiers obtained by application of a classification algorithm to different samples of size N.

The fundamental concept behind the generic expressions is to characterize a class of classifiers induced by a classification algorithm and an i.i.d. sample of a particular size from an underlying distribution. We thus have a distribution over classifiers, with the GE of these classifiers being a random function that has its own distribution. Computing the entire distribution can be highly inefficient especially for a complicated function such as the GE of classifiers and hence we resort to efficiently computing the first few moments. With this we now revisit the expressions for the first two moments around zero of the GE of a classifier,

$$\begin{aligned} &E_{\mathcal{Z}(N)}\left[GE(\zeta)\right] = \\ &\int_{x \in \mathcal{X}} P\left[X = x\right] \sum_{y \in \mathcal{Y}} P_{\mathcal{Z}(N)}\left[\zeta(x) = y | x\right] P\left[Y(x) \neq y | x\right] dx, \end{aligned} \tag{2}$$

$$\begin{aligned} &E_{\mathcal{Z}(N) \times \mathcal{Z}(N)}\left[GE(\zeta) GE(\zeta')\right] = \\ &\int_{x \in \mathcal{X}} \int_{x' \in \mathcal{X}} P\left[X = x\right] P\left[X = x'\right] \cdot \\ &\sum_{y \in \mathcal{Y}} \sum_{y' \in \mathcal{Y}} P_{\mathcal{Z}(N) \times \mathcal{Z}(N)}\left[\zeta(x) = y \wedge \zeta'(x') = y' | x, x'\right] \cdot \\ &P\left[Y(x) \neq y | x\right] P\left[Y(x') \neq y' | x'\right] dx dx' \end{aligned} \tag{3}$$

It becomes clear from the above equations that to compute these moments we must be able to characterize the behavior of a classifier on each input independently for the first moment, and on pairs of inputs for the second moment. More specifically, we need be able to compute $P_{\mathcal{Z}(N)}\left[\zeta(x) = y\right]$ for the first moment and $P_{\mathcal{Z}(N) \times \mathcal{Z}(N)}\left[\zeta(x) = y \wedge \zeta'(x') = y'\right]$ for the second moment for the classification algorithm under consideration[1]. The other terms in these equations indicate the error of a single classifier or errors of two classifiers derived from the underlying distribution for the first and second moment respectively. This can be better understood from the following simple example. If the class prior for class 1 is p and that for class 2 is *1-p*, then the error of a classifier classifying data into class 1 is *1-p* and the error of a classifier classifying data into class 2 is p.

2.3 Customized Expressions for Random Decision Trees

As we saw in the previous subsection, in order to compute the moments we need to characterize $P_{\mathcal{Z}(N)}\left[\zeta(x) = y\right]$ for the first moment and $P_{\mathcal{Z}(N) \times \mathcal{Z}(N)}\left[\zeta(x) = y \wedge \zeta'(x') = y'\right]$ for the second moment for specific classification algorithms. To characterize these probabilities we have to mathematically model the manner in which a particular classification algorithm classifies a specific input (or pairs of inputs for the second moment). In

[1] These probabilities and $P\left[Y(x) \neq y\right]$ are conditioned on x. We omit explicitly writing the conditional since it improves readability and is obvious from the context.

the case of decision trees, a specific input is classified based on the relevant path in the tree from root to leaf with the majority class being chosen as the class for the input. Thus, the $P_{\mathcal{Z}(N)}\left[\zeta(x)=y\right]$ for a decision tree algorithm is given by,

$$\begin{aligned} &P_{\mathcal{Z}(N)}\left[\zeta(x)=y\right] = \\ &\sum_{p} P_{\mathcal{Z}(N)}[\xi(path_p y) > \xi(path_p y'), path_p exists, \\ &\qquad \forall y' \neq y,\ y, y' \in \mathcal{Y}] \end{aligned} \tag{4}$$

where p indexes all allowed paths by the decision tree algorithm in classifying input x. After the summation, the term $\xi(path_p y)$ is the number of (count of) inputs in the path "$path_p$" that lie in class y. A similar characterization exists for $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(x)=y \wedge \zeta'(x')=y'\right]$ where, rather than single paths, in each probability after the sum we have pairs of paths. The allowed paths are governed by the attribute selection method and the stopping criteria in the decision tree algorithm. For example, if we grow random decision trees of height h where there are total of d attributes the above probability would be,

$$\begin{aligned} &P_{\mathcal{Z}(N)}\left[\zeta(x)=y\right] = \\ &\sum_{p} \frac{P_{\mathcal{Z}(N)}[\xi(path_p y) > \xi(path_p y'),\ \forall y' \neq y,\ y, y' \in \mathcal{Y}]}{\binom{d}{h}} \end{aligned} \tag{5}$$

This is the case, since for any input there are $\binom{d}{h}$ possible paths and every path is equally likely, i.e., $P[path_p exists] = \frac{1}{\binom{d}{h}}\ \forall p$. The $\binom{d}{h}$ possible paths comes from the fact that as we build the tree there are d choices to pick the root, there are $d-1$ choices to pick each of the nodes at the next level and so on till we reach level h, where we have $d-h+1$ choices. Each of these $\binom{d}{h}$ paths is indexed by p and $path_p$ is the corresponding path. By similar arguments the probability for the second moment is given by,

$$\begin{aligned} &P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(x)=y \wedge \zeta'(x')=y'\right] = \\ &\frac{1}{\binom{d}{h}^2}\Big(\sum_{p,q} P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}[\xi(path_p y) > \xi(path_p y''), \\ &\qquad \xi(path_q y') > \xi(path_q y'''), \\ &\qquad \forall y \neq y'',\ \forall y' \neq y''',\ y, y', y'', y''' \in Y]\Big) \end{aligned} \tag{6}$$

An important thing to notice in the above formulas is that there are two sources of randomness, the first due to the tree building method (i.e., path exists) and the second due to the random samples (i.e., comparing counts) that are generated from the underlying distribution. As we will see later, this observation will help us in deriving bounds for the moments of an ensemble of random decision trees.

3 Characterizing an Ensemble of Random Decision Trees

In this section, we first show that naively extending the analysis for (single) random decision trees to an ensemble leads to highly inefficient formulations with no obvious way of deriving reasonable bounds on the moments. We then suggest a solution using the generic expressions that is efficient to compute and gives tighter bounds than directly finding bounds on the moments.

3.1 Straightforward Extension

An ensemble of random decision trees classify an input into a class based on the classes predicted by individual trees in the ensemble. In particular, any input is classified into the class that receives the maximum number of votes. To characterize $P_{\mathcal{Z}(N)}\left[\zeta(x)=y\right]$ we have to characterize all possible ways in which class y receives the maximum number of votes in classifying input x. Given that the ensemble has T trees with p_i indexing all allowed paths for an input x in the i^{th} tree ($i \leq T$) we have,

$$P_{\mathcal{Z}(N)}\left[\zeta(x)=y\right] = \frac{1}{\binom{d}{h}^T}\Big(\sum_{p_1,\ldots,p_T} P_{\mathcal{Z}(N)}[\eta_x^y]\Big) \tag{7}$$

where η_x^y denotes class y gets the maximum number of votes in classifying x through the respective paths of the T trees. Hence, given a set of paths, $P_{\mathcal{Z}(N)}[\eta_x^y]$ computes the probability of classifying input x into y over all samples of size N from an underlying distribution. η_x^y is characterized by comparing counts of various classes in the leaves of the trees in the ensemble. This is a generalization of the characterization for single trees shown in equation 5, where counts are compared in a single leaf to counts being compared in the corresponding leaves of all the trees in the ensemble. This leads to an exponential in T number of massive joint probabilities that need to be computed. Given that T is generally in the hundreds, exactly computing the above formula is impractical for even extremely small domains. It is also important to notice that the joint probabilities containing the counts from different trees cannot be factorized as a product of probabilities containing counts of individual trees, although the trees are built independently, the individual tree classifiers are correlated through the sample. There also does not seem to be any obvious way of deriving efficient and tight bounds on these expressions. The same issues are faced when estimating or bounding the probability for pairs of inputs used in the computation of the second moment.

3.2 Main Result

In the previous subsection we observed that a naive extension of the ideas used to characterize single random decision trees leads to highly inefficient formulations for an ensemble of random decision trees. As indicated before, there are two sources of randomness: the first due to the random tree building method and the second due to the random samples from the underlying distribution. To derive the main result in this paper, we characterize the second source of randomness, then fix it and compute probabilities based on the first source of randomness. This strategy leads to bounds that can be computed efficiently. With this we have the following theorem.

Theorem 1 *Consider a set of samples drawn from the underlying distribution that have cumulative measure/probability mass α and let S_{max} be a sample in this set for which the number of paths of length h ($\leq d$), that classify input x into class y is maximum. Let us denote this maximum number of paths by $m_h(S_{max})$. Then given that there are T trees in the ensemble and $q = \binom{d}{h}$ is the total number of paths of length h we have,*

$$P_{\mathcal{Z}(N)}\left[\zeta(x)=y\right] \leq \beta B(\frac{T}{2}, T, \frac{m_h(S_{max})}{q}) + (1-\beta)$$

where $0 \leq \beta \leq \alpha \leq 1$ *and* $B(\frac{T}{2}, T, \frac{m_h(S)}{q}) = \sum_{i=\lceil \frac{T}{2} \rceil}^{T} \binom{T}{i} (\frac{m_h(S)}{q})^i (1-\frac{m_h(S)}{q})^{(T-i)}$.

The bound on the probability for the second moment is analogous to the above bound with the only difference being that we consider pairs of inputs rather than just single inputs.

Before we describe the details in deriving the above bound and techniques to efficiently estimate the necessary parameters (viz. $m_h(S_{max})$ and β), we provide a brief overview of how these critical parameters can be estimated.

1. To compute $m_h(S_{max})$ for samples of size N with total measure α, find the (worst) sample for which the number of paths of length h that would classify input x into class y is maximum in this set.
2. Given this sample we can easily check to see how many of these q paths classify input x into class y. The higher the $m_h(S_{max})$ the looser the bound.
3. β can be estimated using Bonferronis inequality [21] for the given set of samples. The lower the β the looser the bound. Notice that there can be many sets of samples of size N with measure α. The best set of samples, i.e., the ones that would give the tightest bound would be the ones that have the lowest $m_h(S_{max})$. In this paper, we provide a strategy that finds a particular set of samples efficiently that are not necessarily the best. The set of samples our procedure finds seems reasonable, however, given that our bounds are mostly non-trivial and significantly tighter than Breiman's bounds, as witnessed in the experimental section. Deciphering the best set of samples efficiently is part of future research.

3.3 Derivation and Efficient Parameter Estimation of the Bound

We have seen that a straightforward extension of the ideas that were used to characterize single random decision trees cannot be used (due to computational considerations) to characterize an ensemble of random decision trees. We thus have to come up with a different strategy or perspective to tackle this problem. There are 3 key elements in coming up with this characterization, which are as follows:

1. First, characterize the second source of randomness, i.e., randomness due to the underlying distribution, and then conditioning on this source of randomness characterize the first source of randomness which is due to the tree building process.
2. Define a parameter $0 \leq \alpha < 1$ that denotes the measure of the samples of size N drawn from the underlying distribution. This parameter affects the tightness of the bounds.

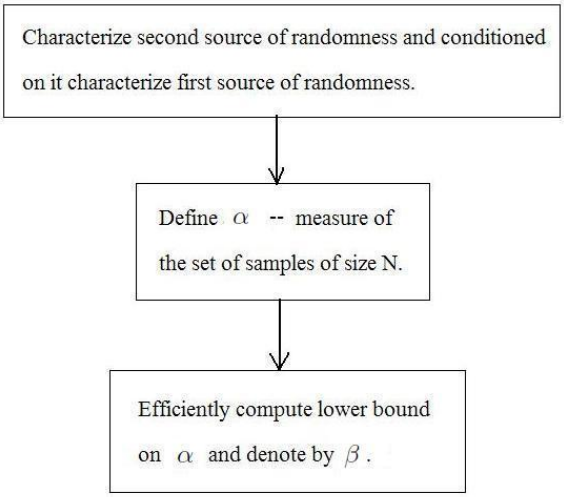

Fig. 1 The figure shows the key steps in deriving the bound for an ensemble of random decision trees.

3. Find a set of samples whose measure is exactly α can be computationally intensive and, hence, find samples with measure say β where $\beta \leq \alpha$ which can be done efficiently and still leads to an useful upper bound on the moments.

These elements are also mentioned in figure 1, and the details regarding them are given below.

Two Sources of Randomness: As mentioned before, there are two sources of randomness that are present: the first due to the tree building method (i.e., path exists) and the second due to the random samples (i.e., comparing counts) that are generated from the underlying distribution. In the characterization for random decision trees, we observe that we compute the probability for a certain path to exist and then, conditioned on this path, we compute the probability of classifying an input into a particular class over all samples of a particular size. In other words, we characterize the first source of randomness and then fix it in the computation of $P_{\mathcal{Z}(N)}[\zeta(x)=y]$ and $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(x)=y \wedge \zeta'(x')=y'\right]$. However, there is also the other possibility of characterizing the second source of randomness and fixing it in order to characterize these probabilities. In this scenario we would have,

$$P_{\mathcal{Z}(N)}[\zeta(x)=y] = \sum_{S\in D(N)} P[S]P_t[\eta_x^y|S] \tag{8}$$

where $D(N)$ denotes all datasets of size N from the underlying distribution, and $P_t[\eta_x^y|S]$ is the probability of classifying input x into class y based on the random tree generation process denoted by t given a particular dataset $S \in D(N)$. For a given dataset, the number of paths of length h that would classify an input x into class y is the same for all trees in the ensemble. In addition, the trees are built independently.

Hence, if $m_h(S)$ is the number of paths of length h that classify input x into class y given a dataset S and if T is the total number of trees in the ensemble, then $P_t[\eta_x^y|S]$ is given by a binomial cumulative distribution function (cdf),

$$P_t[\eta_x^y|S] = B(\frac{T}{2}, T, \frac{m_h(S)}{q}) \tag{9}$$

where $q = \begin{pmatrix} d \\ h \end{pmatrix}$ is the total number of paths of length h and $B(\frac{T}{2}, T, \frac{m_h(S)}{q}) = \sum_{i=\lceil \frac{T}{2} \rceil}^{T} \begin{pmatrix} T \\ i \end{pmatrix} (\frac{m_h(S)}{q})^i (1 - \frac{m_h(S)}{q})^{(T-i)}$. It is thus easy to see that given a sample, $P_t[\eta_x^y|S]$ can be computed efficiently. However, the number of samples will be huge for any reasonable problem size. Hence, directly computing the moments using this formulation also does not seem feasible. In fact, this formulation is infeasible even for single random decision trees. Nonetheless, this formulation does present us with an opportunity to compute bounds on the moments which we will now derive. The solution we provide is efficient and, as we will see in the experimental section turns out to be tighter than Breiman's bounds: $\kappa \frac{(1-s^2)}{s^2}$, where κ is the correlation between the random decision trees in an ensemble and s is the strength of the resultant classifier .[2]

Defining α: One way of upper bounding $P_{\mathcal{Z}(N)}[\zeta(x)=y]$, is to find the maximum $P_t[\eta_x^y|S]$ over all samples and then to substitute this value for each of the conditionals in equation 8. The maximum $P_t[\eta_x^y|S]$ would most likely be 1, since for arbitrary distributions there might exist samples where $m_h(S) = q$. This would make the bound trivial (i.e., 1), since $\sum_{S \in D(N)} P[S] = 1$. Hence, rather than choosing all samples, we can choose a fraction of them with measure α to upper bound the probabilities. $\alpha = 1$ corresponds to choosing all samples while lower values of α (viz. 0.95 or 0.99, etc.) correspond to choosing a subset of the samples with measure α. For this subset, we can find the sample which has the maximum $m_h(S)$ and, hence, the maximum $P_t[\eta_x^y|S]$ for that set. In the experimental section, we will see that our method of finding this sample and consequently the maximum $P_t[\eta_x^y|S]$ for that set leads to bounds that are significantly tighter than Breimans strength and correlation bounds. With this we have the following inequality,

$$\begin{aligned} P_{\mathcal{Z}(N)}[\zeta(x)=y] &= \sum_{S \in D(N)} P[S] P_t[\eta_x^y|S] \\ &\leq \alpha P_t[\eta_x^y|S_{max}] + (1-\alpha) \end{aligned} \tag{10}$$

S_{max} is the sample for which $P_t[\eta_x^y|S_{max}]$ has the maximum value in that set of samples with total measure α. Note that there can be many sets of samples with total measure α. Ideally, one would want to choose the set for which $P_t[\eta_x^y|S_{max}]$ is the least, since this would lead to the tightest possible bound among the other alternatives. At this point, it is not clear how this optimal set can be deciphered; however, we provide a solution where we find one of the feasible sets in an efficient manner and, as we will see later, this solution tends to give reasonably good bounds.

As mentioned before, our analysis applies to discrete attributes as well as continuous attributes with prespecified split points. Given this, any distribution over the

[2] For further details refer to [7] and [8].

Table 2 Example data distribution.

X ↓, Y →	y_1	y_2	$\cdots$	y_c	
$\bar{x_1}$	p_{11}	p_{12}	$\cdots$	p_{1c}	
$\bar{x_2}$	p_{21}	p_{22}	$\cdots$	p_{2c}	
$\vdots$					
$\bar{x_n}$	p_{n1}	p_{n2}	$\cdots$	p_{nc}	
					N

data can be represented as a multinomial with parameters $N, p_{11}, p_{12}, ..., p_{nc}$ as shown in table 2. Here, N is the sample size and $p_{11}, p_{12}, ..., p_{nc}$ are the probabilities for the corresponding input-output pairs (n distinct inputs and c classes) to occur with $\sum_{i=1,j=1}^{n,c} p_{ij} = 1$. Notice that bootstrap distributions built by resampling a dataset are subsumed by the given class of distributions. Given such a multinomial distribution, we want to find a set of samples/datasets of size N where the total measure of these datasets is equal to α.

Computing β: Attempting to directly find these samples would be computationally intensive, since we would have to sum the measures of each sample in the set in order to verify that the measure is α. However, since we want to find an upper bound on $P_{\mathcal{Z}(N)}[\zeta(x)=y]$, substituting a lower bound for α in equation 10 would give us an upper bound on $P_{\mathcal{Z}(N)}[\zeta(x)=y]$, as desired. Finding a lower bound on α can be done efficiently with the help of Bonferroni's inequality [21]. Bonferroni's inequality states that given g random variables $Z_1, ..., Z_g$ and $2g$ real numbers $a_1, ..., a_g, b_1, ..., b_g$ the following relationship holds,

$$\begin{aligned} &P[a_1 \leq Z_1 \leq b_1, ..., a_g \leq Z_g \leq b_g] \\ &\qquad \geq 1 - \sum_{i=1}^{g} \big(1 - P[a_i \leq Z_i \leq b_i]\big) \end{aligned} \tag{11}$$

Hence, the joint probability of variables can be lower bounded by a function of the marginals. The reason this result is useful is that one-dimensional confidence regions (i.e., confidence intervals or domain of marginals) of a particular measure can be efficiently computed as opposed to simultaneous (multidimensional) confidence regions which are much harder to exactly compute [19,20]. One easy way of obtaining confidence intervals for commonly used measures is by using standard formulas. For example, if we want the confidence interval of 0.95 measure for Z_i, we know that an approximation of it is given by $\mu_i + -1.94\sigma_i$, where μ_i and σ_i are the mean and standard deviation of Z_i. Similar formulas are known for other measures and, hence, one can compute confidence intervals for these measures in constant time. If we want the exact confidence interval, we can compute it as a binomial cdf, which can also be done in essentially constant time using series expansions for the incomplete regularized beta function. With this if β is the measure on the right hand side of equation 11, then the measure over the simultaneous confidence region or the relevant set of datasets for our problem is at least β. If α is the true measure over the set of datasets, but $\beta \leq \alpha$ is what we efficiently compute based on the above inequality, then we have,

Table 3 Collapsed view of the underlying distribution as required for bounding $P_{\mathcal{Z}(N)}[\zeta(x_i)=y]$.

X ↓, Y →	y_1	y_2	$\cdots$	y_c	
$x_1^{x_i}$	$p_{11}^{x_i}$	$p_{12}^{x_i}$	$\cdots$	$p_{1c}^{x_i}$	
$x_2^{x_i}$	$p_{21}^{x_i}$	$p_{22}^{x_i}$	$\cdots$	$p_{2c}^{x_i}$	
$\vdots$					
$x_k^{x_i}$	$p_{k1}^{x_i}$	$p_{k2}^{x_i}$	$\cdots$	$p_{kc}^{x_i}$	
R^{x_i}	$p_R^{x_i}$				N

$$
\begin{aligned}
P_{\mathcal{Z}(N)}[\zeta(x)=y] &= \sum_{S\in D(N)} P[S]P_t[\eta_x^y|S] \\
&\leq \alpha P_t[\eta_x^y|S_{max}] + (1-\alpha) \\
&\leq \beta P_t[\eta_x^y|S_{max}] + (1-\beta)
\end{aligned}
\tag{12}
$$

In our problem, the Z_i represent the number of datapoints corresponding to an input-output pair where the input has at least h attributes with the same value as the input x_i for which the above probability is to be upper bounded. This is seen in table 3, where $x_j^{x_i}$ denotes the j^{th} input which has at least h attributes with the same value as input x_i. If each of the d attributes can take v values[3] giving a total of $n = v^d$ distinct inputs, then $k = \sum_{i=h}^{d} \binom{d}{i} (v-1)^{d-i} < n$ and R^{x_i} denotes the entry in the table for the probability remaining after accounting for probabilities of each of the kc pairs i.e. $p_R^{x_i} = 1 - \sum_{j=1,l=1}^{k,c} p_{jl}^{x_i}$ and N as usual is the sample size. Hence, we have kc degrees of freedom i.e. $g = kc$, since the probabilities sum to 1 and the probability in the last cell ($p_R^{x_i}$) is determined by the other probabilities. With this we have Z_1 is the random variable corresponding to the count in the cell formed by the intersection of input $x_1^{x_i}$ and class y_1, Z_2 is the random variable corresponding to the count in the cell formed by the intersection of input $x_1^{x_i}$ and class y_2 and so on until Z_g, which is the random variable corresponding to the count in the cell formed by the intersection of input $x_k^{x_i}$ and class y_c. Consequently, one way of obtaining a confidence interval of measure 0.95 for the random variable representing the intersection of input $x_j^{x_i}$ and class y_l would be $Np_{jl}^{x_i} + -1.94(\sqrt{Np_{jl}^{x_i}(1-p_{jl}^{x_i})})$. We can however choose any confidence interval for each Z_i (not necessarily the same) so as to have the appropriate β, which as we know, lower bounds α. We also need to check that the individual confidence intervals produce legal datasets, since the total sample size is a fixed number N. This can be easily done by adding the upper limit of the intervals for each Z_i and verifying that the eventual sum is $\leq N$. If it is not, one can adjust the limits of one or more of the intervals so as to satisfy the condition. The confidence intervals for each Z_i together form a confidence region that represents the relevant set of datasets. From this set we now have to find S_{max}, i.e., the dataset for which the number of paths that classify input x into class y is the highest among the set. This dataset is the one formed by taking upper limits of the confidence intervals corresponding to class y and the lower limits of the remaining confidence intervals, with the appropriate value for

[3] This is after splitting the continuous attributes.

Algorithm 2 Overview of procedure to find S_{max} and $m_h(S_{max})$ for an input-output pair (x_i, y_b).

Input: h, $X = \{\bar{x}_1, ..., \bar{x}_n\}$, $Y = \{y_1, ..., y_c\}$, $\mathcal{P} = \{p_{11}, ..., p_{nc}\}$ {$\mathcal{P}$ is the estimated or given underlying probability distribution over $X \times Y$.}
Output: S_{max}, $m_h(S_{max})$
Let $x_j^{x_i}$ denote the j^{th} input which has at least h attributes with the same value as input x_i.
Let $p_{jl}^{x_i}$ be the (rolled-up) probability of being $(x_j^{x_i}, y_l)$, which is computed from $\mathcal{P}$.
Let $n_r = N$
for all $j \in \{1, ..., k\}, l \in \{1, .., c\}$ **do**
  if l=b {γ_{jl} below depends on desired β, which is the right side of equation 11, and N.} **then**
    Compute $n_{jl} = Np_{jl}^{x_i} + \gamma_{jl}(\sqrt{Np_{jl}^{x_i}(1 - p_{jl}^{x_i})})$
  else
    Compute $n_{jl} = Np_{jl}^{x_i} - \gamma_{jl}(\sqrt{Np_{jl}^{x_i}(1 - p_{jl}^{x_i})})$
  end if
  Compute $n_r = n_r - n_{jl}$
end for
$S_{max} = \{n_{11}, ..., n_{kc}, n_r\}$
Compute $m_h(S_{max})$ = Count # of paths of length h in S_{max}, where y_b is the majority class.
Return S_{max}, $m_h(S_{max})$

the cell corresponding to R^x, in order to have a total sample size of N. A dataset formed by this procedure would be S_{max}, since taking the upper and lower limits as indicated would indeed produce a dataset with maximum number of paths classifying the particular input into class y in the relevant set. An overview of this procedure is given in algorithm 2. With this, we have completely described a process that can be used to upper bound $P_{\mathcal{Z}(N)}[\zeta(x)=y]$. An analogous procedure can be used to upper bound $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(x)=y \wedge \zeta'(x')=y'\right]$ with the slight difference being we have to consider pairs of inputs rather than single inputs. The basic technique, however, remains the same.

3.4 Time Complexity

In the previous subsection we provided a strategy to upper bound $P_{\mathcal{Z}(N)}[\zeta(x)=y]$ and $P_{\mathcal{Z}(N)\times\mathcal{Z}(N)}\left[\zeta(x)=y \wedge \zeta'(x')=y'\right]$ $\forall x, x' \in X, \forall y, y' \in Y$ and consequently the moments. In this subsection, we analyze the time complexity of this strategy and compare it with the straightforward extension described in section 3.

Consider the following setup, where we have d input attributes each taking v values, a class attribute taking c values, a sample size of N and T trees in the ensemble each of height $h \leq d$. Given this, the straightforward extension would have a time complexity of $O(ncq^T N^{T+1})$ for the first moment and $O(n^2c^2q^{2T}N^{2T+2})$ for the second moment, where $n = v^d$ is the number of distinct inputs and $q = \binom{d}{h}$ is the total number of paths of length h for a particular input. The nc term in the time complexity comes from the fact that we have to estimate nc probabilities, the q^T term indicates we have to sum over all combinations of q paths for the T trees and N^{T+1} is the time it takes to compute each of the huge probabilities in equation 7. The term N^{T+1} can be reduced by approximating each of the probabilities using Monte Carlo sampling,

however, the q^T term is still very intimidating given that T is usually in the hundreds. The time complexity of our bound is $O(nkc^2qT)$ for the first moment and a quadratic function of this value for the second moment, where $k = \sum_{i=h}^{d} \binom{d}{i} (v-1)^{d-i} < n$. This time complexity comes from the fact that tables, such as table 3, can be created for each input in one pass of the dataset and then for each input-output pair we can find $\frac{m_h(S_{max})}{q}$ followed by the binomial probability in $O(kcqT)$ time. A key component enabling this exponential gain is the fact that the simultaneous confidence region can be lower bounded by bounding each variable independently and that $m_h(S_{max})$ can be found efficiently as shown before. Hence, there is a huge overall reduction in time complexity for practical applications as there is an exponential gain in T, with the sample size N not affecting the analysis. The increased $O(kc)$ factor is more than compensated for by these reductions. We have thus reduced the complexity from something that was practically impossible to compute even for small-scale problems to something that is reasonable to compute for medium-scale and even certain large-scale problems. Moreover, the bounds on the individual probabilities can be computed in parallel.

3.5 Practical Considerations

In the previous subsection we analyzed the worst-case time complexity of computing our bound. It however may be possible to further speed up its computation in practice. For instance, to upper bound the moments, we may not have to upper bound the probabilities for every input-output pair. For every input, it is sufficient to upper bound every probability starting from the one corresponding to the highest error (i.e., highest $P[X=x]\,P[Y(x)\neq y]\ \ \forall y \in Y$) to the ones with lower errors in a (descending) sorted fashion until the upper bounds exceed 1 or if we reach the probability with the lowest error, i.e., for the first moment the $P_{\mathcal{Z}(N)}[\zeta(x)=y]$ need not be upper bounded if $P[X=x]\,P[Y(x)\neq y]$ is the lowest error among all of Y and similarly for the second moment. The weight multiplied to the lowest error term corresponding to input x is $1 - \sum_{y' \in Y, y' \neq y} U(x, y')$ where $U(x, y')$ is the upper bound on $P_{\mathcal{Z}(N)}\left[\zeta(x)=y'\right]$ if the sum of $U(x, y')$ does not exceed 1. This can be done, since the probabilities for classifying an input into different classes form a convex combination. Conversely, if we first upper bound the probability with the lowest error for a particular input and then in a (ascending) sorted fashion upper bound probabilities with successively higher errors until one of the aforementioned conditions are met, we would get lower bounds on the moments. Such a strategy would give tighter bounds than blindly upper bounding every probability.

It is important to realize that, because of the generic expressions, we get tighter bounds when compared with directly bounding the moments using the procedure mentioned in the previous subsection. From Bonferonni's inequality in equation 11, we see that as the number of random variables increase, the bound (linearly) loosens. Hence, having fewer variables will invariably lead to tighter bounds. If we were to derive bounds directly, we would have to consider the entire dataset with all the cells without collapsing or aggregation of cells into fewer cells and hence fewer variables [11], as is the case using the generic expressions. The reason for this is that the generic expressions consider cells corresponding to an individual input or pairs of inputs rather than the entire input space.

4 Experiments

In the previous section we described a technique to derive bounds on the moments. In this section, we test the usefulness of these bounds by performing experiments on synthetic and real data. In particular, we compare the tightness of the upper bound derived by our method with the commonly used strength and correlation bounds given by Breiman, which are upper bounds on GE. As we will see, our bound tends to be significantly tighter than the alternative in most circumstances.

4.1 Setup

The experimental setup that we have here is very similar to the one in a previous study [10]. In each of the experiments on synthetic and real data that we describe below, we obtain a robust estimate of $E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$ by generating 1000 hold-out or test sets of 10000 datapoints and then averaging the error over these 1000 hold-out sets. Note that, even in the synthetic case, the $E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$ cannot be computed exactly, since the expectation is over all possible classifiers built over datasets of size N, which is computationally infeasible even for N in the hundreds. We set the confidence intervals per variable (Z_i) to 0.99. The number of trees in the ensemble (T) is set to a 100 for the real data and varied from 100 to 500 for the synthetic data. We also report the average computation time in seconds for the respective bounds.

The experimental setup for synthetic data is as follows: In our initial experiments we fix N to 100 and then increase it to 10000. The number of classes is fixed to two. We compute the upper bounds on $E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$ where the number of attributes is fixed to $d = 5$ with each attribute having 2 attribute values. We then increase the number of attribute values to 3 to observe the effect that increasing the number of split points has on the performance of the estimators. We also increase the number of attributes to $d = 8$ to study the effect that increasing the number of attributes has on the performance. With this we have a $d+1$ dimensional contingency table whose d dimensions are the attributes and the $(d+1)^{th}$ dimension represents the class labels. When each attribute has two values, the total number of cells in the table is $c = 2^{d+1}$ and with three values the total number of cells is $c = 3^d \times 2$. If we fix the probability of observing a datapoint in cell i to be p_i such that $\sum_{i=1}^{c} p_i = 1$ and the sample size to N the distribution that perfectly models this scenario is a multinomial distribution with parameters N and the set $\{p_1, p_2, ..., p_c\}$. In fact, irrespective of the value of d and the number of attribute values for each attribute, the scenario can be modeled by a multinomial distribution. In the studies that follow, the p_i's are varied and the amount of dependence between the attributes and the class labels (ρ) is computed for each set of p_i's using the Chi-square test [9]. More precisely, we sum over all i the squares of the difference of each p_i with the product of its corresponding marginals, with each squared difference being divided by this product, that is, correlation $= \sum_i \frac{(p_i - p_{im})^2}{p_{im}}$, where p_{im} is the product of the marginals for the i^{th} cell.

In the case of real data, we perform experiments by building bootstrap distributions on three UCI data sets that have been used previously in the context of decision trees [10] and six other industrial datasets obtained from diverse domains. The six proprietary datasets denoted by Oil, Semiconductor, Store, Spend, Airline and Invoice in table 6, are obtained from the petrochemical industry, the semiconductor industry,

the consumer products industry, the finance domain, the airline industry and the procurement domain respectively.

Oil: The Oil dataset has information obtained from a major oil corporation. There are a total of 9 measures in the dataset. These measures are obtained from the sensors of a 3-stage separator that separates oil, water, and gas. The 9 measures are composed of 2 measured levels of the oil water interface at each of the 3 stages and 3 overall measures. In particular, the measures are as follows:

1. 20LT_0017_mean (stage 1)
2. 20LT_0021_mean (stage 1)
3. 20LT_0081_mean (stage 2)
4. 20LT_0085_mean (stage 2)
5. 20LT_0116_mean (stage 3)
6. 20LT_0120_mean (stage 3)
7. Daily water in oil (Daily WIO)
8. Daily oil in water (Daily OIW)
9. Daily production normal

Our target measure is the last listed measure: Daily production normal (binary variable), which indicates if the daily oil production meets the desired level or not. The dataset size is 992.

Semiconductor: In the chip manufacturing industry predicting wafers lying within certain spec (which is a collection of chips) accurately ahead of time can be crucial in choosing the appropriate set of wafers to send forward for further processing. Eliminating faulty wafers can save the industry a huge amount of resources in terms of time and money.

The dataset we have has 175 features including the target. The target is a binary variable, which indicates if the wafer conforms to required spec or not. The other features are a combination of physical measurements and electrical measurements made on the wafer. The dataset size is 2361.

Store: In the consumer products industry, predicting when a certain item of a competitor brand may or may not be on promotion can be critical for strategic planning. The dataset we have has information about past promotions that we have carried out for a type of bread and, for the corresponding time periods, we also have information about 10 competitors that have carried out promotions. The target here is the binary time series corresponding to our closest competitor. The dataset size is 140 corresponding to 140 weeks of data.

Spend: The Spend dataset contains a couple of years worth of (spend) transactions spread across various categories belonging to a large corporation. There are 145963 transactions which are indicative of the companies expediture in this time frame. The dataset has 13 attributes namely; requestor name, cost center name, description code, material group, vendor name, business unit name, region, purchase order type, addressable, spend type, compliant, invoice spend amount. Our target is the compliance attribute, which indicates if a transaction was compliant or not. Given this, the goal is to identify the characteristics of a transaction that highly correlate with it being compliant/non-compliant. With this information the company would be able to put in

Table 4 The table shows $E_{\mathcal{Z}(N)}[GE(\zeta)]$ and the corresponding upper bounds for different levels of correlation (ρ) between the attributes and class labels for $T = 100$. The values before the commas in the cells under different levels of correlation correspond to our method while the values following the commas correspond to Brieman's strength and correlation bounds.

Parameter Settings	Split	Metric	$\rho = 0.9$	$\rho = 0.36$	$\rho = 0.11$	$\rho = 0.02$
$N = 100$,	binary	Bounds	0.22,0.79	0.33,1.23	0.66,1.27	0.57,1.34
$d = 5$, $h = 3$		$E_{\mathcal{Z}(N)}[GE(\zeta)]$	0.13	0.26	0.49	0.53
		Time	0.09,0.11	0.09,0.11	0.09,0.11	0.09,0.11
$N = 100$,	ternary	Bounds	0.48,1.37	0.8,2.49	0.67,1.77	0.57,1.7
$d = 5$, $h = 3$		$E_{\mathcal{Z}(N)}[GE(\zeta)]$	0.19	0.32	0.51	0.54
		Time	0.21,0.32	0.21,0.32	0.21,0.32	0.21,0.32
$N = 100$,	binary	Bounds	0.71,2.21	0.8,1.89	0.67,1.78	0.57,0.97
$d = 8$, $h = 3$		$E_{\mathcal{Z}(N)}[GE(\zeta)]$	0.33	0.42	0.48	0.49
		Time	0.35,0.56	0.35,0.56	0.35,0.56	0.35,0.56
$N = 10000$,	binary	Bounds	0.22,1.89	0.33,0.99	0.47,0.75	0.56,0.69
$d = 5$, $h = 3$		$E_{\mathcal{Z}(N)}[GE(\zeta)]$	0.11	0.23	0.41	0.50
		Time	0.10,0.12	0.10,0.12	0.10,0.12	0.10,0.12
$N = 10000$,	ternary	Bounds	0.26,1.15	0.36,3.9	0.49,5.41	0.50,1.42
$d = 5$, $h = 3$		$E_{\mathcal{Z}(N)}[GE(\zeta)]$	0.17	0.27	0.44	0.48
		Time	0.22,0.32	0.22,0.32	0.22,0.32	0.22,0.32
$N = 10000$,	binary	Bounds	0.68,1.86	0.8,1.21	0.67,4.86	0.57,0.62
$d = 8$, $h = 3$		$E_{\mathcal{Z}(N)}[GE(\zeta)]$	0.35	0.47	0.49	0.52
		Time	0.34,0.58	0.34,0.58	0.34,0.58	0.34,0.58

place appropriate policies and practices that could lead to potentially huge savings.

Airline: This dataset contains information about flights of a major airline carrier. In particular, the dataset has information about the date of departure and arrival of a flight, the airport the flight daparts from and arrives at, the pilots serial number, the type of aircraft, the dispatchers name, the cost and multiple fuel indicators signifying the amount of fuel put in the flight, the minimum required by the government and other safety fuel limits. We have information of over 100,000 flights and the goal is to identify factors that lead to the flight carrying excess fuel than what is required. We have a binary indicator, which depicts if the flight carried excess fuel. Controlling the factors which lead to this can help an airline carrier save money and improve safety.

Invoice: Large corporations have many of business units (BUs) viz. travel, marketing, auditing, information technology etc. with each BU consisting of various commodity councils (CCs) viz. office supplies, tech services, communication services, etc. Throughout a calendar year, each of the commodity councils carry out multiple transactions and record the corresponding invoices. It is extremely useful for these CCs and BUs at large to estimate in advance the invoice amounts that are likely to be registered in the near future. This dataset has 8 BUs with each BU consisting of 150 CCs. The data was collected daily for a year and, hence, the dataset has only 365 datapoints. Our target is the CC communication services[4] under the BU information technology, which has one of the highest invoice amounts and therefore is critical to business.

For each of the above datasets, we split the continuous attributes at the mean of the given data. We thus can form a contingency table representing each of the data

[4] Partitioned into 3 categories high, medium and low.

Table 5 The table shows $E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$ and the corresponding upper bounds for different levels of correlation (ρ) between the attributes and class labels for $T = 500$. The values before the commas in the cells under different levels of correlation correspond to our method while the values following the commas correspond to Brieman's strength and correlation bounds.

Parameter Settings	Split	Metric	$\rho = 0.9$	$\rho = 0.36$	$\rho = 0.11$	$\rho = 0.02$
$N = 100$,	binary	Bounds	0.23,0.76	0.35,1.13	0.68,1.17	0.60,1.29
$d = 5$, $h = 3$		$E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$	0.11	0.24	0.46	0.50
		Time	0.48,0.56	0.48,0.56	0.48,0.56	0.48,0.56
$N = 100$,	ternary	Bounds	0.49,1.17	0.84,2.25	0.67,1.77	0.56,1.8
$d = 5$, $h = 3$		$E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$	0.16	0.31	0.51	0.55
		Time	1.07,1.59	1.07,1.59	1.07,1.59	1.07,1.59
$N = 100$,	binary	Bounds	0.73,2.11	0.83,1.71	0.69,1.62	0.56,0.98
$d = 8$, $h = 3$		$E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$	0.30	0.40	0.47	0.48
		Time	1.78,2.89	1.78,2.89	1.78,2.89	1.78,2.89
$N = 10000$,	binary	Bounds	0.23,1.76	0.33,0.94	0.46,0.74	0.56,0.67
$d = 5$, $h = 3$		$E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$	0.11	0.21	0.38	0.49
		Time	0.49,0.60	0.49,0.60	0.49,0.60	0.49,0.60
$N = 10000$,	ternary	Bounds	0.29,1.02	0.34,3.5	0.49,5.39	0.50,1.41
$d = 5$, $h = 3$		$E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$	0.14	0.25	0.43	0.47
		Time	1.08,1.73	1.08,1.73	1.08,1.73	1.08,1.73
$N = 10000$,	binary	Bounds	0.71,1.67	0.78,1.17	0.65,4.74	0.56,0.61
$d = 8$, $h = 3$		$E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$	0.33	0.45	0.48	0.53
		Time	1.73,2.96	1.73,2.96	1.73,2.96	1.73,2.96

Table 6 The table shows $E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$ for the real datasets with the respective upper bounds and running time.

Dataset	$E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$	Our Bound, Time	Breiman's Bound, Time
Pima Indians	0.31	0.63, 0.37	2.01, 0.57
Balloon	0.24	0.29, 0.06	0.84, 0.08
Shuttle Landing Control	0.21	0.31, 0.09	0.91, 0.10
Oil	0.32	0.47, 0.41	0.83, 0.63
Semiconductor	0.09	0.16, 293.52	0.27, 342.73
Store	0.11	0.22, 0.53	0.25, 0.71
Spend	0.16	0.27, 267.17	0.86, 313.51
Airline	0.29	0.42, 238.82	0.93, 248.59
Invoice	0.43	0.79, 1226.78	1.37, 1892.29

sets. The counts in the individual cells divided by the data set size provide us with empirical estimates for the individual cell probabilities (p_i's). Thus, with the knowledge of N (data set size) and the individual p_i's, we have a multinomial distribution. Using this distribution we observe the behavior of the bounds.

The results are shown in tables 4, 5 and 6, where we also see the upper bounds computed using Breiman's formula [7]: $\kappa\frac{(1-s^2)}{s^2}$ described before. Estimating κ and s we find the upper bound on the GE for a particular classifier. Since we need an estimate of $E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$, we perform the above procedure multiple times thus building multiple ensembles and computing an upper bound on GE for each. We then average the upper bounds that we have computed and report the result as an estimate of the upper bound on $E_{\mathcal{Z}(N)}\left[GE(\zeta)\right]$.

4.2 Observations

In the synthetic experiments, our bound is the tightest when $d = 5$, $h = 3$ and when we have binary splits while it is the worst when $d = 8$, $h = 3$. This is because the number of variables is the least in the first scenario while it is the most in the second scenario. This trend is especially seen at moderate correlation (0.36), since the number of paths classifying inputs into the class with high errors for sample S_{max} increases more dramatically with the number of variables for this case than at high or low correlations. In any case, we do significantly better than Breiman's bound.

Increasing the sample size from 100 to 10000 has little effect on the time it takes to compute our bound while the quality of the bound remains intact, which is promising. Increasing the number of trees in the ensemble from 100 (table 4) to 500 (table 5) also maintains the quality of our bound though the time to compute it linearly increases. This is consistent with complexity analysis we provided in the previous section.

In the experiments on real data, we see a similar behavior where our bound outperforms Breiman's bound on all the nine datasets. These datasets are from diverse domains and exhibit varied characteristics in terms of sample size, dimensionality and type of attributes. The type of attributes are a mix of continuous and discrete, with the discrete attributes having a range of splits. We see in these as well as the synthetic experiments that the time to compute our bound increases as the number of attributes and the splitting factor increases, though it is still faster than computing Brieman's bound. The reason for this is that we do not have to explicitly build the trees to compute our bound.

Hence, from the experiments we see that our bound is never worse than that computed using Breimans's formula, though in certain situations it is close to being trivial, i.e., close to 1.

5 Discussion

In this paper we described a procedure to derive bounds for an ensemble of random decision trees. It would be interesting to extend such an analysis to the more general random forest algorithm. In order to derive bounds for the random forest algorithm, we would have to characterize the probability of choosing a particular attribute based on two criteria: first, that it is part of the subset of attributes that are randomly chosen and, second, that it has the maximum IG or GG in that set. For the ensemble of random decision trees, we just had to deal with the first criteria with a subset size of one. Deriving acceptable bounds for the random forest algorithm efficiently is thus a challenge for the future.

In the analysis presented in previous sections, one of the main ideas in computing the bound efficiently was to use Bonferroni's inequality. However, this inequality can lead to loose bounds with increasing dimensionality. In the future, it would be interesting to come up with better characterizations of the simultaneous confidence region leading to tighter and, hence, more useful bounds. The challenge, however, is to decipher these regions in a scalable fashion.

To conclude, we have presented a novel procedure to derive bounds on the moments of GE for an ensemble of random decision trees. This procedure is efficient and leads to tighter bounds than the well established Breiman's strength and correlation bounds. The derivation of these bounds required a perspective and a set of analytical tools

that are significantly different from the strategies used to characterize (single) random decision trees [10]. It would be interesting to see if the ideas used in analyzing the ensemble of random decision trees can be used to analyze other such ensembles in the future.

Acknowledgement

I would like to thank the editor and the anonymous reviewers for their constructive comments. I would also like to thank Katherine Dhurandhar for proofreading the paper.

References

1. A. Anandkumar, D. Foster, D. Hsu, S. Kakade and Y. Liu. A Spectral Algorithm for Latent Dirichlet Allocation. In *NIPS*, page 926-934, Lake Tahoe, USA, 2012.
2. B. Boots and G. Gordon. Two Manifold Problems with Applications to Nonlinear System Identification. In *ICML*, page 338, Edinburgh, Scotland, UK, 2012.
3. N. Bshouty and P. Long. Finding Planted Partitions in Nearly Linear Time using Arrested Spectral Clustering. In *ICML*, page 135-142, Haifa, Israel, 2010.
4. T. Hastie, R. Tibshirani and J. Friedman. *Elements of Statistical Learning*. Springer, 2 edition, 2001.
5. R. Duda, P. Hart, D. Stork. *Pattern Classification*. Wiley New York, 2 edition, 2001.
6. A. Dhurandhar and A. Dobra. Distribution free bounds for relational classification. *Knowledge and Information Systems*, 2012.
7. L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
8. S. Buttrey and I. Kobayashi. On strength and correlation in random forests. In *Proceedings of the 2003 Joint Statistical Meetings, Section on Statistical Computing*, 2003.
9. J. Connor-Linton. Chi square tutorial. http://www.georgetown.edu/faculty/ballc/webtools/web_chi_tut.html, 2003.
10. A. Dhurandhar and A. Dobra. Probabilistic characterization of random decision trees. *Journal of Machine Learning Research*, 9:2321–2348, 2008.
11. A. Dhurandhar and A. Dobra. Semi-analytical method for analyzing models and model selection measures based on moment analysis. *ACM Transactions on Knowledge Discovery and Data Mining*, 2009.
12. A. Dhurandhar and A. Dobra. Probabilistic characterization of nearest neighbor classifiers. *Intl. Journal on Machine Learning and Cybernetics*, 2012.
13. W. Fan, H. Wang, P. Yu, and S. Ma. Is random model better? on its accuracy and efficiency. In *ICDM '03: Proceedings of the Third IEEE International Conference on Data Mining*, page 51, Washington, DC, USA, 2003. IEEE Computer Society.
14. P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, 2006.
15. John Langford. Tutorial on practical prediction theory for classification. *J. Mach. Learn. Res.*, 6:273–306, December 2005.
16. F. Liu, K. Ting, and W. Fan. Maximizing tree diversity by building complete-random decision trees. In *PAKDD*, pages 605–610, 2005.
17. D. McAllester. Pac-bayesian model averaging. In *In Proceedings of the Twelfth Annual Conference on Computational Learning Theory*, pages 164–170. ACM Press, 1999.
18. David Mcallester. Simplified pac-bayesian margin bounds. In *In COLT*, pages 203–215, 2003.
19. S. Roy and R. Bose. Simultaneous confidence interval estimation. *Annals of Mathematical Statistics*, 24(3):513–536, 1953.
20. C. Sison and J. Glaz. Simultaneous confidence intervals and sample size determination for multinomial proportions. *JASA*, 90(429):366–369, 1995.
21. Y. Tong. *Probabilistic Inequalities for Multivariate Distributions*. Academic Press, 1 edition, 1980.
22. K. Zhang and W. Fan. Forecasting skewed biased stochastic ozone days: analyses, solutions and beyond. *Knowl. Inf. Syst.*, 14(3):299–326, 2008.
23. X. Zhang, Q. Yuan, S. Zhao, W. Fan, W. Zheng, and Z. Wang. Multi-label classification without the multi-label cost. In *SDM '10: Proceedings of the Siam Conference on Data Mining*, pages 778–789, 2010.

Author Biography

Fig. 2 Amit Dhurandhar is a research staff member in the Mathematical Sciences Dept. at IBM T.J. Watson. He received his B.E. in computer engineering from Pune University in 2004. He then received his Masters and P.h.d. in computer engineering from University of Florida in 2005 and 2009 respectively. He is the acting Knowledge Discovery and Data Mining Professional Interest Community (KDD PIC) chair at IBM T.J. Watson. Broadly speaking, Amit's research interests primarily span the areas of machine learning, data mining and computational neuroscience. He has authored several papers and has been a reviewer for many top quality conferences and journals.