Interpreting Blackbox Models via Model Extraction

Osbert Bastani
Stanford University
obastani@cs.stanford.edu

Carolyn Kim
Stanford University
ckim@cs.stanford.edu

Hamsa Bastani
Stanford University
hsridhar@stanford.edu

Abstract

Interpretability has become an important issue as machine learning is increasingly used to inform consequential decisions. We propose an approach for interpreting a blackbox model by extracting a decision tree that approximates the model. Our model extraction algorithm avoids overfitting by leveraging blackbox model access to actively sample new training points. We prove that as the number of samples goes to infinity, the decision tree learned using our algorithm converges to the exact greedy decision tree. In our evaluation, we use our algorithm to interpret random forests and neural nets trained on several datasets from the UCI Machine Learning Repository, as well as control policies learned for three classical reinforcement learning problems. We show that our algorithm improves over a baseline based on CART on every problem instance. Furthermore, we show how an interpretation generated by our approach can be used to understand and debug these models.

1 Introduction

Recent advances in machine learning have revolutionized our ability to use data to inform critical decisions, such as medical diagnosis [30, 16, 44], bail decisions for defendants [28, 27] and the design of aircraft collision avoidance systems [42]. At the same time, machine learning algorithms have been shown to exhibit unexpected defects when deployed in the real world; examples include causality (i.e., inability to distinguish causal effects from correlations) [34, 16], fairness (i.e., internalizing prejudices present in training data) [22, 26], and algorithm aversion (i.e., lack of trust by end users) [19].

Interpretability is a promising approach to address these challenges [38, 21]—in particular, we can help human users diagnose issues and verify correctness of machine learning models by providing insight into the model's reasoning [43, 8, 37, 31, 29]. For example, suppose the user is trying to train a model that does not depend on a prejudiced feature (e.g., gender or ethnicity). Omitting the feature might not suffice to avoid prejudice, since the model could reconstruct that feature from other features [35]. A better approach might be to train the model with the prejudiced feature, and then assess the dependence of the model on that feature. This approach requires the ability to understand the model's reasoning process, i.e., how model predictions are affected by changing the prejudiced feature [21]. Similarly, the user may want to determine whether dependence on a feature is causal, or understand the high-level structure of the model to gain confidence in its correctness.

In this paper, we propose an technique that we call *model extraction* for interpreting the overall reasoning process performed by a model. Given a model $f : \mathcal{X} \to \mathcal{Y}$, the interpretation produced by our algorithm is an approximation $T(x) \approx f(x)$, where T is an interpretable model. In this paper, we take T to be a decision tree, which has been established as highly interpretable [31, 37, 8]. Intuitively, if T is a sufficiently good approximation of f, then any issues in f should be reflected in T as well. Thus, the user can understand and debug f by examining T; then, the original model f can be deployed so that performance is not sacrificed.

Previous model extraction approaches have focused on interpreting specific families of models such as random forests [45, 18, 46], enabling them to leverage domain-specific knowledge about the internal structure of the model. In contrast, our approach is fully *blackbox*, i.e., it is agnostic to the

internal model structure and only requires the ability to run the model f on an input $x \in \mathcal{X}$ and obtain its predicted output $f(x) \in \mathcal{Y}$. This way, not only is our approach very flexible in that it can work with any model family, but it is furthermore independent of the implementation details of a specific model family.

The key challenge to learning accurate decision trees is that they often overfit and obtain poor performance, whereas complex models such as random forests and deep neural nets are better regularized [9]. For example, random forests use ensembles of trees to avoid overfitting to specific features or training points.

Thus, our algorithm uses active learning to construct T from f—it actively samples a large number of training points, and determines the corresponding label simply by computing $y = f(x)$. The large quantity of data ensures that T does not overfit to the small set of initial training points. We prove that under mild assumptions, by generating a sufficient quantity of data, the extracted tree T converges to the *exact* greedy decision tree, i.e., it avoids overfitting since the sampling error goes to zero.

We implement our algorithm and evaluate its performance in two experiments. First, we use our algorithm to produce interpretations of random forests and neural nets, as well as for control policies trained using reinforcement learning. We show that our active learning approach substantially improves over using CART [11], a standard decision tree learning algorithm. Second, we demonstrate how the decision trees extracted can be used to debug issues with these models, for example, to assess the dependence on prejudiced features, to determine why certain models perform worse, and to understand the high-level structure of a learned control policy.

1.1 Related Work

A number of approaches for improving interpretability have been proposed, which largely fall into three directions: (i) learning more interpretable models, (ii) providing explanations for individual model predictions, and (iii) providing an overall interpretation of a model.

CART [11] is considered highly interpretable, but its accuracy is often lacking in practice; [31, 8] tackle this concern by producing sparse rule lists that resemble decision trees yet are non-greedy, thus enabling them to learn more accurate models without sacrificing interpretability. Alternatively, [43] proposes supersparse linear integer models, which are sparse linear models where the coefficients are integer valued, thus resembling risk-scoring systems constructed manually by humans for applications such as medical diagnosis or criminal rescidivism. This approach is extended by [27] for classification problems with binary features. Finally, [15] propose generalized additive models, which are linear combinations of arbitrarily complex single-feature models, as interpretable models.

However, in many domains, the highest performing models are complex blackbox models with limited interpretability, e.g., neural nets or random forests. While interpretable models may achieve nearly comparable performance, sacrificing even a small amount of accuracy can be undesirable in settings with critical consequences such as medicine. One proposed approach is to use the blackbox model, but generate interpretations for every new test point along with the predicted output. For instance, given a new test point x, [37] generates an interpretation for the prediction $f(x)$ by fitting a simple model locally around x and using this simple model to explain the prediction. Similarly, [32] uses the Shapley value to determine the most influential features for a given prediction. These approaches can help the user understand a specific prediction, but they cannot help understand the model as a whole, making it less useful for diagnosing problems with the model itself.

Our paper takes the third approach, i.e., it interprets the model as a whole. Past techniques have focused on simply identifying influential features. For instance, the *relative influence* scores the contribution of each feature in tree-based models (e.g., random forests) [24]. Similarly, [39] proposes a method for computing influence scores for features in deep neural nets. However, these approaches cannot help understand more complex reasoning performed by the model, as is needed to understand the dependence of a model on prejudiced or potentially non-causal features.

Finally, our model extraction algorithm is closely related to the idea of *model compression* [13], where a larger, more complex model is used to guide the training of a smaller, more efficient model. Our algorithm can be thought of as a model compression algorithm where the target model is a decision tree; it leverages active learning to avoid overfitting the decision tree.

2 Problem Formulation

Our algorithm learns axis-aligned decision trees. We start by establishing notation. An *axis-aligned constraint* is a constraint $C = (x_i \leq t)$, where $i \in [d]$ and $t \in \mathbb{R}$. More general constraints can be built from existing constraints using negations $\neg C$, conjunctions $C_1 \wedge C_2$, and disjunctions $C_1 \vee C_2$. The *feasible set* of C is $\mathcal{F}(C) = \{x \in \mathcal{X} \mid x \text{ satisfies } C\}$.

A *decision tree* T is a binary tree. Each *internal node* $N = (N_L, N_R, C)$ of T has a left child node N_L and a right child node N_R, and is labeled with an axis-aligned constraint $C = (x_i \leq t)$. Each *leaf node* $N = (y)$ of T is associated with a label $y \in \mathcal{Y}$. We use N_T to denote the root node of T.

The decision tree is interpreted as a function $T : \mathcal{X} \to \mathcal{Y}$ in the usual way. More precisely, a leaf node $N = (y)$ is interpreted as a function $N(x) = y$, an internal node $N = (N_L, N_R, C)$ is interpreted as a function $N(x) = N_L(x)$ if $x \in \mathcal{F}(C)$, and $N(x) = N_R(x)$ otherwise. Then, $T(x) = N_T(x)$.

For a node $N \in T$, we let C_N denote the conjunction of the constraints along the path from the root of T to N. More precisely, C_N is defined recursively: for the root N_T, we have $C_{N_T} = \text{True}$, and for an internal node $N = (N_L, N_R, C)$, we have $C_{N_L} = C_N \wedge C$ and $C_{N_R} = C_N \wedge \neg C$.

Then, given a training set $X_{\text{train}} \subseteq \mathcal{X}$ and blackbox access to a function $f : \mathcal{X} \to \mathcal{Y}$, our goal is to learn a decision tree $T : \mathcal{X} \to \mathcal{Y}$ that approximates f. We focus on the case $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{Y} = [m] = \{1, ..., m\}$ (i.e., classification); our approach easily generalizes to the case where $\mathcal{X}$ contains categorical dimensions, and to the case $\mathcal{Y} = \mathbb{R}$ (i.e., regression).

For classification, we measure performance using accuracy relative to f on a held out test set, i.e., $\frac{1}{|X_{\text{test}}|} \sum_{x \in X_{\text{test}}} \mathbb{I}[T(x) = f(x)]$. For binary classification, we use F_1 score, and for regression, we use mean-squared error.

3 Decision Tree Extraction Algorithm

Our algorithm is greedy, both for scalability and because it is a natural fit for interpretability, since more relevant features occur higher in the tree. We first describe the input distribution $\mathcal{P}$ constructed by our algorithm to generate data. Then, we describe the *exact greedy decision tree* T^*, where all decisions are made exactly according to the distribution $\mathcal{P}$ (albeit greedily); note that we cannot construct T^* since we treat f as a blackbox. Finally, we describe how our algorithm estimates T^* using i.i.d. samples from $\mathcal{P}$. In Section 4, we show the estimated tree converges to T^* asymptotically.

3.1 Input Distribution

Our algorithm constructs a distribution $\mathcal{P}$ over $\mathcal{X}$ by fitting a mixture of axis-aligned Gaussian distributions to the training data using expectation maximization. It has parameters $\phi \in \mathbb{R}^k$ defining a categorical distribution over $[K]$, and parameters $\mu \in \mathbb{R}^{Kd}$ and $\Sigma \in \mathbb{R}^{Kd^2}$, where $\mu_j \in \mathbb{R}^d$ and $\Sigma_j \in \mathbb{R}^{d^2}$ (where Σ_j is diagonal), for $j \in [K]$, defining the jth Gaussian distribution in the mixture. To sample $x \sim \mathcal{P}$, first sample $j \sim \text{Categorical}(\phi)$ and then sample $x \sim \mathcal{N}(\mu_j, \Sigma_j)$.

3.2 Exact Greedy Decision Tree

We describe the *exact greedy decision tree* T^* of size k, which closely mirrors CART [11]. It is initialized to a tree with a single leaf node $N_{T^*} = (y)$, where y is the majority label according to $\mathcal{P}$. At each iteration, a leaf node $N = (y)$ in the current tree is replaced with an internal node $N' = (N_L, N_R, C)$, where $N_L = (y_L)$ and $N_R = (y_R)$ are leaf nodes, and $C = (x_{i^*} \leq t^*)$ where

$$(i^*, t^*) = \operatorname*{arg\,max}_{i \in [d], t \in \mathbb{R}} G(i, t),$$

where the gain

$$G(i, t) = \text{Gain}(f, C_N \wedge (x_i \leq t)) + \text{Gain}(f, C_N \wedge (x_i > t)) - \text{Gain}(f, C_N) \tag{1}$$

$$\text{Gain}(f, C) = \left(1 - \sum_{y \in \mathcal{Y}} \Pr_{x \sim \mathcal{P}}[f(x) = y \mid C]^2\right) \cdot \Pr_{x \sim \mathcal{P}}[C] \tag{2}$$

is the (weighted) Gini impurity. The gain can be replaced with other gains such as information gain, or MSE for regression. The leaf node labels are

$$y_L = \arg\max_{y \in \mathcal{Y}} \Pr_{x \sim \mathcal{P}}[f(x) = y \mid C_N \wedge (x_i \leq t)] \tag{3}$$

$$y_R = \arg\max_{y \in \mathcal{Y}} \Pr_{x \sim \mathcal{P}}[f(x) = y \mid C_N \wedge (x_i > t)]. \tag{4}$$

The node chosen to be replaced is that with the highest potential gain (1) in the current tree.

To construct the exact greedy decision tree of size k, this process is repeated $k-1$ times. If at any iteration, the gain (1) is zero, then the process is terminated (so T^* may have fewer than k nodes).

3.3 Estimated Greedy Decision Tree

Given $n \in \mathbb{N}$, our algorithm constructs a greedy decision tree the same way as the construction of the exact greedy decision tree, except that (1) and (3) are estimated using n i.i.d. samples $x \sim \mathcal{P} \mid C_N$ each. Recall that at each iteration, we have to select the node N with the highest potential gain to replace. Our algorithm does so by estimating the gain for all nodes; these estimates are constructed using a separate set of i.i.d. samples to ensure unbiasedness of the tree parameters.

Next, we describe how our algorithm samples $x \sim \mathcal{P} \mid C$, where C is a conjunction of axis-aligned constraints:

$$C = (x_{i_1} \leq t_1) \wedge ... \wedge (x_{i_k} \leq t_k) \wedge (x_{j_1} > s_1) \wedge ... \wedge (x_{j_h} > s_h).$$

Some inequalities in C may be redundant. First, suppose there are two constraints $x_i \leq t$ and $x_i \leq t'$. Assume without loss of generality that $t \leq t'$; then, the first constraint implies the second, so we can discard the latter. For a pair of constraints $x_i > s$ and $x_i > s'$, we can similarly discard one of them. Second, given two constraints $x_i \leq t$ and $x_i > s$, we can assume that $t \geq s$; otherwise C is unsatisfiable, so the gain (1) would have been zero and the algorithm would have terminated. In summary, we can assume C contains at most one inequality $(x_i \leq t)$ and at most one inequality $(x_i > s)$ per $i \in [d]$, and if both are present, then the two are not mutually exclusive. For simplicity, we assume C contains both inequalities for each $i \in [d]$:

$$C = (s_1 \leq x_1 \leq t_1) \wedge ... \wedge (s_d \leq x_d \leq t_d).$$

Now, recall that $\mathcal{P}$ is a mixture of axis-aligned Gaussians, so it has probability density function

$$p_{\mathcal{P}}(x) = \sum_{j=1}^{K} \phi_j \cdot p_{\mathcal{N}(\mu_j, \Sigma_j)}(x) = \sum_{j=1}^{K} \phi_j \prod_{i=1}^{d} p_{\mathcal{N}(\mu_{ji}, \sigma_{ji})}(x_i),$$

where $\sigma_{ji} = (\Sigma_j)_{ii}$. The conditional distribution is

$$p_{\mathcal{P}|C}(x) \propto \sum_{j=1}^{K} \phi_j \prod_{i=1}^{d} p_{\mathcal{N}(\mu_{ji}, \sigma_{ji})|C}(x_i) = \sum_{j=1}^{K} \phi_j \prod_{i=1}^{d} p_{\mathcal{N}(\mu_{ji}, \sigma_{ji})|(s_i \leq x_i \leq t_i)}(x_i).$$

Since the Gaussians are axis-aligned, the unnormalized probability of each component is

$$\tilde{\phi}'_j = \int \phi_j \prod_{i=1}^{d} p_{\mathcal{N}(\mu_{ji}, \sigma_{ji})|(s_i \leq x_i \leq t_i)}(x_i) dx = \phi_j \prod_{i=1}^{d} \left(\Phi\left(\frac{t_i - \mu_{ji}}{\sigma_{ji}} \right) - \Phi\left(\frac{s_i - \mu_{ji}}{\sigma_{ji}} \right) \right),$$

where Φ is the cumulative density function of the standard Gaussian distribution $\mathcal{N}(0, 1)$. Then, the normalization constant is $Z = \sum_{j=1}^{K} \tilde{\phi}'_j$, and the component probabilities are $\phi = Z^{-1}\phi'$.

Therefore, to sample $x \sim \mathcal{P} \mid C$, we sample $j \sim \text{Categorical}(\tilde{\phi})$, and

$$x_i \sim \mathcal{N}(\mu_{ji}, \sigma_{ji}) \mid (s_i \leq x_i \leq t_i) \quad (\text{for each } i \in [d]).$$

We use standard algorithms for sampling truncated Gaussian distributions to sample each x_i.

4 Theoretical Guarantees

We show that our decision tree extraction produces a decision tree that is close to the exact greedy tree for sufficiently large n. A related result is [20], but their analysis is limited to discrete features, for which convergence is much easier to analyze. We give proofs in Appendix A.

We begin by describing our assumptions. First, we make mild assumptions about the distribution $\mathcal{P}$.

Assumption 1. The probability density function $p(x)$ of the distribution $\mathcal{P}$ over $\mathcal{X}$ is continuous, bounded (i.e., $p(x) \leq p_{\max}$), and has bounded domain (i.e., $p(x) = 0$ for $|x| > x_{\max}$).

To satisfy this assumption, we can truncate the Gaussian mixture models used by our algorithm to $\mathcal{X} = \{x \in \mathbb{R}^d \mid \|x\|_\infty \leq x_{\max}\}$, for some $x_{\max} \in \mathbb{R}$. Intuitively, for reasonably large $x_{\max}$, this modification should not affect either the exact greedy tree or the approximate greedy tree by very much, since Gaussian distributions have exponential tails.

Our next assumption says that the exact greedy tree is well defined:

Assumption 2. We assume that the maximizers (i^*, t^*) of (1) and y_L, y_R of (3) are unique.

Otherwise, multiple exact greedy trees exist; our extracted tree would converge to an arbitrary one.

We now define the notion in which the extracted tree converges to the exact tree. For simplicity, we additionally assume that we are learning *complete* decision trees of depth D.

Definition 1. Let T, T' be complete decision trees of depth D. For $\epsilon > 0$, we say T is an ϵ approximation of T' if $\Pr_{x\sim\mathcal{P}}[T(x) = T'(x)] \leq \epsilon$. Let T^* denote the exact greedy decision tree that is complete of depth D. For any $\epsilon, \delta > 0$, we say an extracted decision tree T is (ϵ, δ) exact if

$$\Pr[T \text{ is an } \epsilon \text{ approximation of } T^*] \geq 1 - \delta,$$

where the randomness is taken over the training samples $x \sim \mathcal{P}$ obtained by our algorithm.

Theorem 1. For any $\epsilon, \delta > 0$, there exists $n \in \mathbb{N}$ such that the decision tree extracted by our algorithm using n samples per node is (ϵ, δ) exact.

The key challenge to proving Theorem 1 is accounting for the fact that small errors made early on in the tree introduce small errors into the constraints C_N, which can recursively introduce small errors into the probabilities $\Pr_{x\sim\mathcal{P}_\mathcal{X}}[\,\cdot \mid C_N]$ in the gain (1).

The dependence of n in Theorem 1 on ϵ and δ depends on the *gap* of the exact greedy decision tree.

Definition 2. Let $\mathcal{P}$ be a distribution over $\mathbb{R}$, and let $F(t)$ be an unnormalized cumulative distribution function for $\mathcal{P}$. We say $g : \mathbb{R} \to \mathbb{R}$ is (ϵ', δ') *gapped* according to $\mathcal{P}$ if it has a unique maximizer $t^* = \arg\max_{t\in\mathbb{R}} g(t)$, and for every $t \in \mathbb{R}$ such that $|F(t) - F(t^*)| > \epsilon'$, we have $g(t^*) > g(t) + \delta'$.

In particular, our result depends on the relationship between ϵ' and δ'. For the special case where δ' depends linearly on ϵ', we can obtain sample complexity bounds:

Corollary 1. Let $f : \mathcal{X} \to \mathcal{Y}$. Suppose that for every $i \in [d]$, there exists $\kappa > 0$ such that for every $\epsilon' > 0$, the gain function $g(t) = G(i, t)$ in (1) is $(\epsilon', \kappa \cdot \epsilon')$ gapped. Then, Theorem 1 holds for

$$n \geq \max\left\{\left(1 + \frac{16m}{(\epsilon/2k)^{3/2} \cdot \kappa}\right)^{3D}, \frac{2^{D+1}k}{\delta}, 4\right\}.$$

The exponential dependence on depth is unavoidable since errors increase multiplicatively from a parent node to its children; in practice, depth is small (e.g., $D \leq 5$), so this is not an issue. Besides the gap in the gain g, the value n must also be lower bounded by a function of the gap between the maxima of the gain functions across different dimensions i, as well as the gap in the objective (3) for choosing leaf node labels; we omit this additional term for clarity.

As we discuss in Appendix A.4, we intuitively expect axis-aligned random forests f to satisfy the premise of Corollary 1.

5 Evaluation

We implement our algorithm on top of scikit-learn [14], and evaluate it in two ways. First, we show that it outperforms a baseline that uses CART [11], a popular decision tree learning algorithm. Second, we show how extracted decision trees can be used to understand and debug models.

Description of Problem Instance					Absolute			Relative	
Dataset	Task	Samples	Features	Model	f	T	T_{base}	T	T_{base}
breast cancer [47]	classify	569	32	random forest	0.966	**0.942**	0.934	**0.957**	0.945
prostate cancer [40]	classify	97	9	random forest	0.749	**0.778**	0.689	**0.900**	0.818
dermatology [25]	classify	366	34	random forest	0.994	**0.969**	0.964	**0.970**	0.967
wine origin [23]	classify	178	13	random forest	0.981	**0.925**	0.890	**0.938**	0.890
auto mpg [36]	regress	398	8	random forest	8.62	**11.13**	11.84	**2.29**	2.51
student grade [17]	regress	382	33	random forest	4.47	**4.70**	5.10	**0.40**	0.64
breast cancer [47]	classify	569	32	neural net	0.951	**0.934**	0.933	**0.956**	0.949
prostate cancer [40]	classify	97	9	neural net	0.723	**0.715**	0.659	**0.842**	0.820
dermatology [25]	classify	366	34	neural net	0.988	**0.983**	0.976	**0.997**	0.964
wine origin [23]	classify	178	13	neural net	0.795	**0.755**	0.751	**0.913**	0.905
auto mpg [36]	regress	398	8	neural net	13.77	**15.56**	15.86	**2.37**	2.59
student grade [17]	regress	382	33	neural net	6.60	**5.07**	5.72	**4.27**	5.10
cartpole [10]	reinforce	100	4	control policy	200.0	**190.0**	35.6	**86.8%**	83.8%
mountain car [33]	reinforce	100	2	control policy	-140.0	**-166.6**	-178.6	**81.3%**	78.6%
pendulum [5]	reinforce	100	3	control policy	-638.2	**-1227.1**	-1431.9	**0.56**	1.66

Table 1: Comparison of the decision tree T extracted by our algorithm to the one T_{base} extracted by the baseline. We show absolute performance on ground truth and performance relative to the model f. For classification (resp., regression), performance is F_1 score (resp., MSE) on the test set. For reinforcement learning, it is accuracy (discrete actions) or MSE (continuous actions) on the test set for relative performance, and estimated reward using the decision tree as the policy for absolute performance. We bold the higher score between T and T_{base}.

5.1 Comparison to Baseline Decision Tree Extraction Algorithm

First, we compare our algorithm to a baseline.

Datasets. We study two applications. First, we interpret supervised learning models. We train both a random forest and a neural net on a number of classification and regression tasks, primarily from the UCI Machine Learning Repository [7]. We randomly split each dataset into 70% training set and 30% test set; all results are averaged over 10 splits.

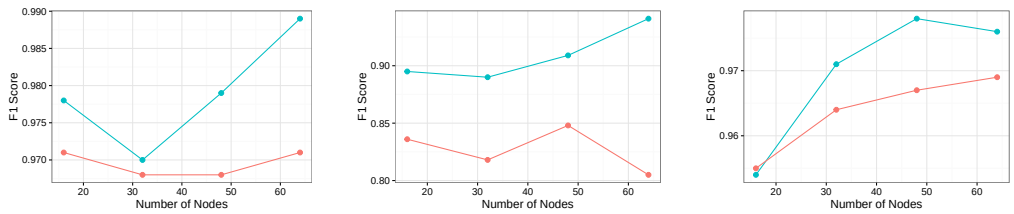
We estimate a Gaussian mixture model $\mathcal{P}$ using the training set, with $K = 50$ components for smaller datasets (< 200 samples), and $K = 100$ components for larger datasets. Then, we extract a decision tree of size at most $k = 31$ (i.e., same size as a depth 5 tree; we show how our results vary with k later) from the model f according to $\mathcal{P}$, using $n = 2000$ samples to estimate (1) and (3). The random forests, trained using scikit-learn, consist of 1000 CART trees, using Gini impurity for classification and MSE for regression. The neural nets, also trained using scikit-learn, are trained with 500 hidden units, ReLU activations, L-BFGS optimization, and regularization parameter $\alpha = 10^{-5}$.

Second, we interpret Markov decision process (MDP) control policies $\pi^* : \mathcal{S} \to \mathcal{A}$, learned using reinforcement learning, for several classical control problems from OpenAI Gym [12]. Here, $\mathcal{S} \subseteq \mathbb{R}^d$ is the state space and $\mathcal{A}$ is the action space, which can be discrete or continuous. To obtain a distribution $\mathcal{P}$ over $\mathcal{S}$, we sample $n = 100$ states from the distribution over $\mathcal{S}$ induced by executing the learned policy π^*, and then split it into training and test sets as before. We then fit a Gaussian mixture model to the training set with $K = 10$ components. Finally, we extract a decision tree approximating π^* according to $\mathcal{P}$ using $n = 2000$ samples to estimate (1) and (3).

The cartpole and mountain car control policies are learned using Q-learning [41]. For the cartpole (discrete actions) [2, 10], the learner uses a discretized state space (7 bins per dimension) [1], and for the mountain car (discrete actions) [4, 33], it approximates the Q-function with a linear model over RBF features [3]. For the pendulum (continuous actions) [5], the learner represents the policy using a 3-layer neural net, which it trains using the cross-entropy method [6].

Baseline. Our baseline is to use CART to train a decision tree approximating f on the training set $\{(x, f(x)) \mid x \in X_{\text{train}}\}$. As before, we use scikit-learn to train these trees, using Gini impurity for classification and MSE for regression. Additionally, we bound the size of the tree to $k = 31$ nodes.

Results. We show results in Table 1. We show the test set performance of the extracted tree compared to ground truth (or for MDPs, estimated the reward when it is used as a policy), as well as the relative performance compared to the model f on the same test set. Note that our goal is to obtain high relative performance, since a better approximation of f is a better interpretation of f, even if f has poor performance. Our algorithm outperforms the baseline on every problem instance.



(a) (b) (c)

Figure 1: Relative performance of our algorithm (blue) and the baseline (red), for (a) random forests trained on the breast cancer dataset, (b) random forests trained on the prostate cancer dataset, and (c) neural nets trained on the breast cancer dataset.

Avoiding overfitting. Our algorithm outperforms the baseline by using active learning to avoid overfitting, especially for high dimensional input spaces with few samples. Even on the relatively large breast cancer dataset, the baseline tree T_{base} obtains an F_1 score of 0.999 on the training set, whereas the extracted tree T obtains only 0.961, i.e., T_{base} has overfit. This discrepancy holds across datasets, and is more pronounced for smaller datasets—on the prostate cancer dataset, T_{base} obtains an F_1 score of 0.981 on the training set, whereas T obtains only 0.860.

Performance vs. size. In Figure 1, we show performance as a function of the number of nodes k on three problem instances. Our algorithm outperforms the baseline even as the tree sizes vary, achieving larger gains for larger k. By using active learning, we can learn arbitrarily large decision trees, whereas the baseline would perfectly fit the data if k becomes sufficiently large.

Consistency of top branch. We examine the consistency of the top branch of the decision trees extracted using our algorithm across the 10 random splits of the datasets, focusing on the case where f is a random forest. In particular, we count the number of features r that occur in the top branch across the splits. We have $r \leq 2$ except for the breast cancer dataset ($r = 3$) and the prostate cancer dataset ($m = 4$), showing that the top feature remains fairly consistent. For the prostate cancer dataset, r is relatively high since the number of samples is small so the models f vary more.

5.2 Examples of Use Cases

Next, we show how the extracted decision trees can be used to interpret and debug models.

Use of invalid features. Using an invalid feature is a common problem when training models. In particular, some datasets contain multiple response variables; then, one response should not be used as a feature when predicting another. For example, the breast cancer dataset contains two response variables indicating cancer recurrence—the length of time before recurrence and whether recurrence occurs within 24 months. This issue can be thought of as a special case of using a non-causal feature, an important problem in healthcare settings. We train a random forest f to predict whether recurrence occurs within 24 months, where time to recurrence is incorrectly included as a feature. Then, we extract a decision tree approximating f of size $k = 7$ nodes, using 10 random dataset splits. The invalid feature occured in every extracted tree, and as the top branch in 6 of the 10 trees.

Use of prejudiced features. We can use our algorithm to evaluate how a model f depends on prejudiced features. For example, gender is a feature in the student grade dataset, and may be considered sensitive when estimating student performance. However, if we simply omit gender, then f may reconstruct it from the remaining features. For a model f trained with gender available, we show how a decision tree extracted from f can be used to understand how f depends on gender. Our approach does not guarantee fairness, but it can be useful for evaluating the fairness of f.

We extract decision trees T from the random forests f trained on 10 random splits of the student grades dataset. The top features are consistently grades in other classes or number of failing grades received in the past. Gender occurs below these features (at the fourth or fifth level) in 7 of 10 of the trees. We can estimate the overall effect of changing the gender label:

$$\Delta = \mathbb{E}_{x \sim \mathcal{P}}[f(x) \mid \text{gender} = \text{M}] - \mathbb{E}_{x \sim \mathcal{P}}[f(x) \mid \text{gender} = \text{F}].$$

When gender occurs, Δ is between 0.31 and 0.70 grade points (average 0.49) out of 20 total grade points. For the remaining models, Δ is between 0.11 and 0.32 (average 0.25). Therefore, the extracted tree includes gender when the model has a relatively large dependence on gender.

More interestingly, because T approximates f, we can use it to identify a subgroup of students where f has particularly strong dependence on gender. Letting $N = (N_L, N_R, C)$ be the internal node branching on gender, $\mathcal{F}(C_N)$ are a subset of inputs whose label $f(x) \in \mathcal{Y}$ is determined in part by gender in T. Also, we can use T to measure the dependence on gender within this subset:

$$\Delta_N = \mathbb{E}_{x\sim\mathcal{P}}[f(x) \mid C_{N_L}] - \mathbb{E}_{x\sim\mathcal{P}}[f(x) \mid C_{N_R}].$$

Furthermore, we can estimate the fraction of students in this subset using the test set X_{test}, i.e., $P = \sum_{x\in X_{\text{test}}} \mathbb{I}[x \in \mathcal{F}(C_N)]$. Finally, $P \cdot \Delta_N/\Delta$ measures the fraction of the overall dependence of f on gender that is accounted for by the subtree rooted at N. For models where gender occurs in the extracted tree, the subgroup effect size Δ_N ranged from 0.44 to 0.77 grade points, and the estimated fraction of students in this subroup ranged from 18.3% to 39.1%. The two trees that had the largest effect size had Δ_N of 0.77 and 0.43, respectively, and P of 39.1% and 35.7%, respectively; the identified subgroup accounted for 67.3% and 65.6% of the total effect of gender, respectively.

Having identified a subgroup of students likely to be adversely affected, the user might be able to train a better model specifically for this subgroup. In 5 of the 7 extracted trees where gender occurs, the affected students were students with low grades, in particular, the 27% of students who scored fewer than 8.5 points in another class. This fine-grained understanding of f relies on the extracted model, and cannot be obtained using feature importance metrics alone.

Comparing models. We can use the extracted decision trees to compare different models trained on the same dataset, and gain insight into why some models perform better than others. For example, random forests trained on the wine origin dataset performed very well, all achieving an F_1 score of at least 0.961. In contrast, the performance of the neural nets was bimodal—5 had F_1 score of at least 0.955, and the remaining had an F_1 score of at most 0.741.

We examined the top 3 layers of the extracted decision trees T, and made two observations. First, occurrence of the feature "chlorides" in T was almost perfectly correlated with poor performance of the neural nets. This feature occured in only one of the 10 trees extracted from random forests, and in none of the trees extracted from high performing neural nets. A weaker observation was the branching of T on the feature "alcohol", which is a very important feature—it is the top branch for all but one of the 20 extracted decision trees. For the high performing models, the branch threshold t tended to be higher (749.8 to 999.6) than those for the poorly performing models (574.4 to 837.3). The latter observation relies on having an extracted model—feature influence metrics alone are insufficient.

Understanding control policies. We can use the extracted decision tree to understand a learned control policy. For example, we describe a decision tree of size $k = 7$ extracted from the cartpole control policy. While its estimated reward of 152.3 is lower than for larger trees, it captures a significant fraction of the policy behavior. The tree says to move the cart to the right exactly when

$$(\text{pole velocity} \geq -0.286) \wedge (\text{pole angle} \geq -0.071),$$

where the pole velocity is in $[-2.0, 2.0]$ and the pole angle is in $[-0.5, 0.5]$. In other words, move the cart to the right exactly when the pole is already on the right relative to the cart, and the pole is also moving toward the left (or more precisely, not moving fast enough toward the right). This policy is asymmetric, focusing on states where the cart is moving to the left. Examining an animation of simulation, this bias occurs because the cart initially moves toward the left, so the portion of the state space where the cart is moving toward the right is relatively unexplored.

6 Conclusions

We have proposed an approach for interpreting blackbox models based on decision tree extraction, and shown how it can be used to interpret random forests, neural nets, and control policies. Important directions for future work include devising algorithms for model extraction using more expressive input distributions, and developing new ways to gain insight from the extracted decision trees.

References

[1] Cartpole algorithm. `https://github.com/YuriyGuts/cartpole-q-learning/blob/master/cartpole.py`. Accessed: 2017-05-18.

[2] Openai cartpole-v0 environment. `https://gym.openai.com/envs/CartPole-v0`. Accessed: 2017-05-18.

[3] Openai mountaincar-v0 algorithm. `https://github.com/nydevyura/deep_rl_experiments/blob/master/rl/experiments/mountain_car.py`. Accessed: 2017-05-18.

[4] Openai mountaincar-v0 environment. `https://gym.openai.com/envs/MountainCar-v0`. Accessed: 2017-05-18.

[5] Openai pendulum-v0 environment. `https://gym.openai.com/envs/Pendulum-v0`. Accessed: 2017-05-18.

[6] Pendulum algorithm. `https://gym.openai.com/evaluations/eval_c0KRCYlQqaUczlcGTwQUA`. Accessed: 2017-05-18.

[7] Uci machine learning repository. `http://archive.ics.uci.edu/ml`. Accessed: 2017-05-18.

[8] Elaine Angelino, Nicholas Larus-Stone, Daniel Alabi, Margo Seltzer, and Cynthia Rudin. Learning certifiably optimal rule lists for categorical data. *KDD*, 2017.

[9] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *Advances in neural information processing systems*, pages 2654–2662, 2014.

[10] Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846, 1983.

[11] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[12] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[13] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541. ACM, 2006.

[14] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[15] Rich Caruana, Yin Lou, and Johannes Gehrke. Intelligible models for classification and regression. In *Proceedings of the 23rd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Citeseer, 2012.

[16] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1721–1730. ACM, 2015.

[17] Paulo Cortez and Alice Maria Gonçalves Silva. Using data mining to predict secondary school student performance. 2008.

[18] Houtao Deng. Interpreting tree ensembles with intrees. *arXiv preprint arXiv:1408.5456*, 2014.

[19] Berkeley J Dietvorst, Joseph P Simmons, and Cade Massey. Algorithm aversion: People erroneously avoid algorithms after seeing them err. *Journal of Experimental Psychology: General*, 144(1):114, 2015.

[20] Pedro Domingos and Geoff Hulten. Mining high-speed data streams. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 71–80. ACM, 2000.

[21] Finale Doshi-Velez and Been Kim. A roadmap for a rigorous science of interpretability. *arXiv preprint arXiv:1702.08608*, 2017.

[22] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 214–226. ACM, 2012.

[23] M Forina et al. An extendible package for data exploration, classification and correlation. *Institute of Pharmaceutical and Food Analisys and Technologies, Via Brigata Salerno*, 16147, 1991.

[24] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[25] H Altay Güvenir, Gülşen Demiröz, and Nilsel Ilter. Learning differential diagnosis of erythemato-squamous diseases using voting feature intervals. *Artificial intelligence in medicine*, 13(3):147–165, 1998.

[26] Moritz Hardt, Eric Price, Nati Srebro, et al. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems*, pages 3315–3323, 2016.

[27] Jongbin Jung, Connor Concannon, Ravi Shroff, Sharad Goel, and Daniel G Goldstein. Simple rules for complex decisions. *arXiv preprint arXiv:1702.04690*, 2017.

[28] Jon Kleinberg, Himabindu Lakkaraju, Jure Leskovec, Jens Ludwig, and Sendhil Mullainathan. Human decisions and machine predictions. Technical report, National Bureau of Economic Research, 2017.

[29] P. W. Koh and P. Liang. Understanding black-box predictions via influence functions. *arXiv preprint arXiv:1703.04730*, 2017.

[30] Igor Kononenko. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in medicine*, 23(1):89–109, 2001.

[31] Benjamin Letham, Cynthia Rudin, Tyler H McCormick, David Madigan, et al. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, 9(3):1350–1371, 2015.

[32] Scott Lundberg and Su-In Lee. An unexpected unity among methods for interpreting model predictions. *arXiv preprint arXiv:1611.07478*, 2016.

[33] Andrew William Moore. *Efficient Memory-based Learning for Robot Control*. PhD thesis, University of Cambridge, 1990.

[34] Judea Pearl. *Causality*. Cambridge university press, 2009.

[35] Dino Pedreshi, Salvatore Ruggieri, and Franco Turini. Discrimination-aware data mining. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 560–568. ACM, 2008.

[36] J Ross Quinlan. Combining instance-based and model-based learning. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 236–243, 1993.

[37] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM, 2016.

[38] Cynthia Rudin. Algorithms for interpretable machine learning. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1519–1519. ACM, 2014.

[39] Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*, 2016.

[40] Thomas A Stamey, John N Kabalin, John E McNeal, Iain M Johnstone, Fuad Freiha, Elise A Redwine, and Norman Yang. Prostate specific antigen in the diagnosis and treatment of adenocarcinoma of the prostate. ii. radical prostatectomy treated patients. *The Journal of urology*, 141(5):1076–1083, 1989.

[41] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[42] Selim Temizer, Mykel Kochenderfer, Leslie Kaelbling, Tomas Lozano-Pérez, and James Kuchar. Collision avoidance for unmanned aircraft using markov decision processes. In *AIAA guidance, navigation, and control conference*, page 8040, 2010.

[43] Berk Ustun and Cynthia Rudin. Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*, 102(3):349–391, 2016.

[44] Gilmer Valdes, José Marcio Luna, Eric Eaton, Charles B Simone, et al. Mediboost: a patient stratification tool for interpretable decision making in the era of precision medicine. *Scientific Reports*, 6, 2016.

[45] Anneleen Van Assche and Hendrik Blockeel. Seeing the forest through the trees. In *International Conference on Inductive Logic Programming*, pages 269–279. Springer, 2007.

[46] Gilles Vandewiele, Olivier Janssens, Femke Ongenae, Filip De Turck, and Sofie Van Hoecke. Genesim: genetic extraction of a single, interpretable model. *arXiv preprint arXiv:1611.05722*, 2016.

[47] William H Wolberg and Olvi L Mangasarian. Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proceedings of the national academy of sciences*, 87(23):9193–9196, 1990.

A Proofs of Main Results

In this section, we give proofs of Theorem 1 and Corollary 1.

A.1 Proof of Main Lemmas

In this section, we state and prove the main lemmas required to prove Theorem 1 and Corollary 1. Essentially, these results are the inductive steps needed to prove our main results.

Lemma 1. Let $\mathcal{X} \subseteq \mathbb{R}^d$, and let $\mathcal{P}$ and $\tilde{\mathcal{P}}$ be distributions over $\mathcal{X}$, and let $p(x)$ and $\tilde{p}(x)$ be unnormalized probability density functions for $\mathcal{P}$ and $\tilde{\mathcal{P}}$, respectively. The function $\tilde{p}$ depends on a parameter $n \in \mathbb{N}$; in applications of this lemma, $\tilde{p}$ is an estimate of p computed by our algorithm on n samples. Assume that for any $\epsilon, \delta > 0$, there exists $n \in \mathbb{N}$ such that $\|p - \tilde{p}\|_1 \leq \epsilon$ with probability at least $1 - \delta$.

Furthermore, let $i \in [d]$, and define

$$G(t) = -\sqrt{\sum_{s \in \{\pm 1\}} \sum_{y \in \mathcal{Y}} \frac{g_{s,y}(t)^2}{h_s(t)}}$$

$$\tilde{G}(t) = -\sqrt{\sum_{s \in \{\pm 1\}} \sum_{y \in \mathcal{Y}} \frac{\tilde{g}_{s,y}(t)^2}{\tilde{h}_s(t)}}$$

$$\hat{\tilde{G}}(t) = -\sqrt{\sum_{s \in \{\pm 1\}} \sum_{y \in \mathcal{Y}} \frac{\hat{\tilde{g}}_{s,y}(t)^2}{\tilde{h}_s(t)}},$$

where

$$h_s(t) = \int \mathbb{I}[s \cdot x_i \leq t] \cdot p(x) dx$$

$$\tilde{h}_s(t) = \int \mathbb{I}[s \cdot x_i \leq t] \cdot \tilde{p}(x) dx$$

$$g_{s,y}(t) = \int \mathbb{I}[f(x) = y] \cdot \mathbb{I}[s \cdot x_i \leq t] \cdot p(x) dx$$

$$\tilde{g}_{s,y}(t) = \int \mathbb{I}[f(x) = y] \cdot \mathbb{I}[s \cdot x_i \leq t]] \cdot \tilde{p}(x) dx$$

$$\hat{\tilde{g}}_{s,y}(t) = \frac{Z}{n} \sum_{i=1}^{n} \mathbb{I}[f(x^{(k)}) = y] \cdot \mathbb{I}[s \cdot x_i^{(k)} \leq t],$$

where $x^{(1)}, ..., x^{(n)}$ are i.i.d. samples from $\tilde{\mathcal{P}}$, and where

$$Z = \int \tilde{p}(x) dx$$

is a normalization constant. Now, let

$$t^* = \arg\max_{t \in \mathbb{R}} G(t)$$

$$\tilde{t}^* = \arg\max_{t \in \mathbb{R}} \hat{\tilde{G}}(t),$$

and let

$$p'(x) = p(x) \cdot \mathbb{I}[x_i \leq t^*]$$

$$\tilde{p}'(x) = \tilde{p}(x) \cdot \mathbb{I}[x_i \leq \tilde{t}^*]$$

be unnormalized probability density functions for $\mathcal{P} \mid (x_i \leq t^*)$ and $\tilde{\mathcal{P}} \mid (x_i \leq \tilde{t})$, respectively.

Assume that $\|h_s\|_\infty \geq \gamma > 0$ for each $s \in \{\pm 1\}$. For any $\epsilon', \delta' > 0$, there exists $n \in \mathbb{N}$ such that

$$\|p' - \tilde{p}'\|_1 \leq \epsilon'$$

with probability at least $1 - \delta'$.

Proof. Let $\epsilon', \delta' > 0$ be arbitrary. First, we show that the L_∞ norm of each $g = g_{s,y}$ is bounded with high probability. By Lemma 4, we have

$$\|g - \hat{\tilde{g}}\|_\infty \le \|g - \tilde{g}\|_\infty + \|\tilde{g} - \hat{\tilde{g}}\|_\infty \le \|g - \tilde{g}\|_\infty + \frac{4 \log n}{\sqrt{n}}$$

with probability at least $1 - 2n^{-3/2}$. Furthermore, by Lemma 3 and by our assumption, for any $\epsilon, \delta > 0$, there exists $n \in \mathbb{N}$ such that

$$\|g - \hat{\tilde{g}}\|_\infty \le \epsilon + \frac{4 \log n}{\sqrt{n}}$$

with probability at least $1 - 2n^{-3/2} - \delta$. Similarly, for $h = h_s$, by Lemma 3 and by our assumption, using the same (ϵ, δ), we have $\|h - \tilde{h}\|_\infty \le \epsilon$. In particular, it follows that $\|\tilde{h}\|_\infty \ge \gamma - \epsilon$. Next, let

$$g(t) = \operatorname*{arg\,max}_{s \in \{\pm 1\}, y \in \mathcal{Y}} \frac{g_{s,y}(t)}{h_s(t)}$$

$$\hat{\tilde{g}}(t) = \operatorname*{arg\,max}_{s \in \{\pm 1\}, y \in \mathcal{Y}} \frac{\hat{\tilde{g}}_{s,y}(t)}{\tilde{h}_s(t)}.$$

Then, note that

$$\begin{aligned}
\left\| \frac{1}{\sqrt{h_s}} - \frac{1}{\sqrt{\tilde{h}_s}} \right\|_\infty = \left\| \frac{\sqrt{\tilde{h}_s} - \sqrt{h_s}}{\sqrt{h_s \cdot \tilde{h}_s}} \right\|_\infty &= \left\| \frac{\tilde{h}_s - h_s}{\sqrt{h_s \cdot \tilde{h}_s}} \right\|_\infty \cdot \left\| \sqrt{\tilde{h}_s} + \sqrt{h_s} \right\|_\infty^{-1} \\
&\le \frac{\epsilon}{\sqrt{\gamma \cdot (\gamma - \epsilon)}} \cdot \left(\frac{1}{\sqrt{\gamma - \epsilon}} + \frac{1}{\sqrt{\gamma}} \right) \\
&\le \frac{\epsilon}{2(\gamma - \epsilon)^{3/2}},
\end{aligned}$$

from which it follows that

$$\begin{aligned}
\left\| \frac{g_{s,y}}{\sqrt{h_s}} - \frac{\hat{\tilde{g}}_{s,y}}{\sqrt{\tilde{h}_s}} \right\|_\infty &= \left\| \frac{g_{s,y}}{\sqrt{h_s}} - \left(\frac{1}{\sqrt{\tilde{h}_s}} + \frac{1}{\sqrt{h_s}} - \frac{1}{\sqrt{h_s}} \right) \cdot \hat{\tilde{g}}_{s,y} \right\|_\infty \\
&\le \left\| \frac{g_{s,y} - \hat{\tilde{g}}_{s,y}}{\sqrt{h_s}} \right\|_\infty + \left\| \left(\frac{1}{\sqrt{\tilde{h}_s}} - \frac{1}{\sqrt{h_s}} \right) \cdot \hat{\tilde{g}}_{s,y} \right\|_\infty \\
&\le \frac{\epsilon}{\sqrt{\gamma}} + \frac{4 \log n}{\sqrt{\gamma} \cdot \sqrt{n}} + \frac{\epsilon}{2(\gamma - \epsilon)^{3/2}} \\
&\le \frac{2\epsilon}{(\gamma - \epsilon)^{3/2}} + \frac{4 \log n}{\sqrt{\gamma} \cdot \sqrt{n}}.
\end{aligned}$$

Therefore, letting $m = |\mathcal{Y}|$, we have

$$\begin{aligned}
&\|G - \hat{\tilde{G}}\|_\infty \\
&\le \left\| \sqrt{\sum_{s \in \{\pm 1\}} \sum_{y \in \mathcal{Y}} \frac{g_{s,y}^2}{h_s}} - \sqrt{\sum_{s \in \{\pm 1\}} \sum_{y \in \mathcal{Y}} \frac{\hat{\tilde{g}}_{s,y}^2}{\tilde{h}_s}} \right\|_\infty \\
&= \left\| \sum_{s \in \{\pm 1\}} \sum_{y \in \mathcal{Y}} \frac{g_{s,y}^2}{h_s} - \sum_{s \in \{\pm 1\}} \sum_{y \in \mathcal{Y}} \frac{\hat{\tilde{g}}_{s,y}^2}{\tilde{h}_s} \right\|_\infty \cdot \left\| \sqrt{\sum_{s \in \{\pm 1\}} \sum_{y \in \mathcal{Y}} \frac{g_{s,y}^2}{h_s}} + \sqrt{\sum_{s \in \{\pm 1\}} \sum_{y \in \mathcal{Y}} \frac{\hat{\tilde{g}}_{s,y}^2}{\tilde{h}_s}} \right\|_\infty^{-1} \\
&\le \sum_{s \in \{\pm 1\}} \sum_{y \in \mathcal{Y}} \left\| \frac{g_{s,y}^2}{h_s} - \frac{\hat{\tilde{g}}_{s,y}^2}{\tilde{h}_s} \right\|_\infty \cdot \|g + \hat{\tilde{g}}\|_\infty^{-1} \\
&\le \sum_{s \in \{\pm 1\}} \sum_{y \in \mathcal{Y}} \left\| \frac{g_{s,y}}{\sqrt{h_s}} - \frac{\hat{\tilde{g}}_{s,y}}{\sqrt{\tilde{h}_s}} \right\|_\infty \\
&\le \frac{4m\epsilon}{(\gamma - \epsilon)^{3/2}} + \frac{8m \log n}{\sqrt{\gamma} \cdot \sqrt{n}}.
\end{aligned}$$

Now, let $\mathcal{P}_i$ be the marginal probability distribution of $\mathcal{P}$ along dimension i, let

$$p_i(t) \propto \int p(x_1, ..., x_{i-1}, t, x_{i+1}, ..., x_d) dx_1 ... dx_{i-1} dx_{i+1} ... dx_d$$

be an unnormalized marginal probability density function for $\mathcal{P}_i$, and let

$$F_i(t) = \int_{-\infty}^{t} p_i(t') dt'$$

be the corresponding unnormalized cumulative distribution function.

Next, let $\epsilon'' > 0$ be arbitrary; we choose its value later. By Lemma 2, for any $\epsilon'' > 0$, there exists $\delta_G(\epsilon'') > 0$ such that G is $(\epsilon'', \delta_G(\epsilon''))$ gapped according to the function p_i.

Recall that we have assumed that ϵ and δ can be made arbitrarily close to zero by taking n to be sufficiently large. In other words, there exists $n \in \mathbb{N}$ such that both of the following hold:

$$\frac{4m}{(\gamma - \epsilon)^{3/2}} \cdot \epsilon + \frac{8m \log n}{\sqrt{\gamma} \cdot \sqrt{n}} \leq \frac{\delta_G(\epsilon'')}{2} \tag{5}$$

$$\delta + \frac{2}{n^{3/2}} \leq \delta', \tag{6}$$

in which case

$$\|G - \hat{\tilde{G}}\|_\infty \leq \frac{\delta_G(\epsilon'')}{2}$$

with probability at least $1 - \delta'$. Then, by Lemma 5, we have

$$|F_i(\tilde{t}^*) - F_i(t^*)| \leq \epsilon'',$$

so by Lemma 6, we have

$$\|p' - \tilde{p}'\|_1 \leq \epsilon + \epsilon''.$$

Thus, the claim follows taking $\epsilon' = \epsilon + \epsilon''$. □

Corollary 2. Assume the same setup as in Lemma 1. Assume that $\gamma \geq 2\epsilon$, and suppose that

$$\delta_g(\epsilon'') \geq \kappa \epsilon''$$

for all $\epsilon'' > 0$ and some constant $\kappa > 0$. For any $\epsilon', \delta' > 0$, we have $\|p' - \tilde{p}\|_1 \leq \epsilon'$ with probability at least $1 - \delta'$ as long as $\|p - \tilde{p}\|_1 \leq \epsilon$ with probability at least δ, where

$$\epsilon \leq \left(1 + \frac{4m}{(\gamma/2)^{3/2}} \cdot \frac{4}{\kappa}\right)^{-1} \cdot \epsilon'$$

$$\delta \leq \frac{1}{2} \cdot \delta',$$

and $n \geq 3$ satisfies

$$\epsilon \geq \frac{2(\gamma/2)^{3/2}}{\sqrt{\gamma}} \cdot \frac{\log n}{\sqrt{n}}$$

$$\delta \geq \frac{2}{n^{3/2}}.$$

Proof. The claim follows from the proof of Lemma 1 by choosing

$$\epsilon'' = \frac{4m}{(\gamma/2)^{3/2}} \cdot \frac{4}{\kappa} \cdot \epsilon$$

in (5). □

A.2 Proof of Theorem 1

Proof. Let $\epsilon, \delta > 0$ be arbitrary; we need to show that there exists $n \in \mathbb{N}$ such that T is an ϵ approximation of T^* with probability at least $1 - \delta''$.

First, for each leaf node $N \in \text{leaves}(T^*)$ of the exact tree, let $\phi(N)$ be the corresponding leaf in the learned tree T. We assume that sufficiently many samples are taken so that the estimates of the following information are correct with probability at least $1 - \frac{\delta}{4k(d+m)}$ (where $m = |\mathcal{Y}|$):

- For each internal node, the dimension $i \in [d]$ along which to branch.
- For each leaf node, the label y assigned to that leaf node.

First, we describe how to ensure that the optimal dimension $i \in [d]$ along which to branch is chosen for each of the $\frac{k}{2}$ internal nodes in T. In particular, by Assumption 2, there exists $\Delta > 0$ such that for every $i \neq i^*$ and for every t, we have

$$G(i^*, t^*) > G(i, t) + \Delta.$$

Let $G_i(t) = G(i, t)$. As long as $\|G - \hat{\hat{G}}\|_\infty \leq \frac{\Delta}{2}$ for each $i \in [d]$, then the optimal dimension is selected. Lemma 1 already shows that $\|G - \hat{\hat{G}}\|_\infty$ is arbitrarily small for $n \in \mathbb{N}$ sufficiently large.

Next, we describe how to ensure that the optimal label y is selected for each leaf node. In particular, labels are selected according to (3). To simplify notation, let

$$p_y = \int \mathbb{I}[f(x) = y] \cdot \mathbb{I}[C_N] \cdot p(x) dx$$

be the unnormalized version of (3), which can equivalently be used to select the optimal label (since the normalization factor is a constant across all $y \in \mathcal{Y}$). Our goal is to choose the optimal label

$$y^* = \arg\max_{y \in \mathcal{Y}} p_y$$

for a leaf node $N = (y^*)$ in T^*. By Assumption 2, there exists a gap $\Delta > 0$ such that for every $y \neq y^*$, we have $p_{y^*} \geq p_y + \Delta$, so it suffices to show that $|p_y - \hat{\hat{p}}_y| \leq \frac{\Delta}{2}$, where $\hat{\hat{p}}_y$ is our estimate of p_y, as before, with two kinds of error. In particular, for each $y \in \mathcal{Y}$, we can break down the error of our estimate $\hat{\hat{p}}_y$ into

$$|p_y - \hat{\hat{p}}_y| \leq |p_y - \tilde{p}_y| + |\tilde{p}_y - \hat{\hat{p}}_y|,$$

where

$$\tilde{p} = \int \mathbb{I}[f(x) = y] \cdot \mathbb{I}[C_{\phi(N)}] \cdot p(x) dx.$$

We assume that N satisfies $\Pr_{x \sim \mathcal{P}}[C_N] \geq \gamma$, so we have $\|p_N - \tilde{p}_N\|_1 \leq \epsilon'$. Then, we have

$$|p_y - \tilde{p}_y| = \left| \int \mathbb{I}[f(x) = y] \cdot (\mathbb{I}[C_N] - \mathbb{I}[C_{\phi(N)}]) \cdot p(x) dx \right| \leq \|p_N - p_{\phi(N)}\|_1 \leq \epsilon'.$$

For the estimation error, by Hoeffding's inequality, we have

$$\Pr_{x^{(1)}, \ldots, x^{(n)} \sim \mathcal{P}} \left[|\hat{\hat{p}}_y - \tilde{p}_y| \geq \frac{\Delta}{4} \right] \leq 2 \exp\left(-\frac{2n\Delta^2}{16} \right).$$

Taking $\epsilon' \leq \frac{\Delta}{4}$, the total error $|p_y - \hat{\hat{p}}_y| \leq \frac{\Delta}{2}$. Taking a union bound over $y \in \mathcal{Y}$, this inequality holds with probability $2m \exp\left(-\frac{2n\Delta^2}{16}\right)$, where $m = |\mathcal{Y}|$. Since this probability is exponentially decreasing in n, we can easily bound it by $\frac{\delta}{2}$.

Next, let N be a node in the exact tree T^*, and let $\phi(N)$ be its corresponding node in the learned tree T. Let $\mathcal{P} \mid C_N$ be the distribution over points that flow to N in T^*, let $\mathcal{P} \mid C_{\phi(N)}$ be the distribution over points that flow to $\phi(N)$ in T, and let

$$p_N(x) = p(x) \cdot \mathbb{I}[C_N]$$
$$p_{\phi(N)}(x) = p(x) \cdot \mathbb{I}[C_{\phi(N)}]$$

be their respective unnormalized probability density functions.

We prove by induction that for each node $N \in \text{leaves}(T^*)$, one of the following holds:

- The premise of Lemma 1 holds, i.e., for any $\epsilon', \delta' > 0$, there exists $n \in \mathbb{N}$ such that $\|p_N - p_{\phi(N)}\|_1 \leq \epsilon'$ with probability at least $1 - \delta'$.
- A negligible amount of probability mass flows to N, i.e., $\Pr[C_N] \leq \gamma = \frac{\epsilon}{k}$.

For the root node N_{T^*} of T^*, the hypothesis of Lemma 1 holds for $\mathcal{P} = \tilde{\mathcal{P}}$, since then $\|p - \tilde{p}\|_1 = 0$ with probability 1.

For any node in N', suppose that the inductive hypothesis holds for its parent N. If a negligible amount of probability mass flows to N, then the same is true of N'. Otherwise, the premise of Lemma 1 holds for N. Consider the gain (1) used to construct the branch $C_N = (x_{i^*} \leq t^*$ for N. First, it is easy to check that the maximum t^* of the function $G(t)$ in Lemma 1 equals the maximizer of the gain—the only difference between $G(t)$ is the square root (which is monotone) and a normalization constant $\int p(x)dx$ (since the probability density function $p(x)$ in Lemma 1 may be unnormalized).

Therefore, either the conclusion of Lemma 1 holds, in which case we have shown the inductive hypothesis, or $\|h_s(t)\|_\infty \leq \gamma$ for some $s \in \{\pm 1\}$. But

$$\gamma \geq \|h_s(t)\|_\infty = \int \mathbb{I}[C_N] \cdot p(x)dx = \Pr[C_N],$$

so again, the inductive hypothesis holds.

Now, note that

$$\begin{aligned}
|\mathbb{I}[C_N] - \mathbb{I}[C_{\phi(N)}]| &= (\mathbb{I}[C_N] - \mathbb{I}[C_{\phi(N)}])^2 \\
&= \mathbb{I}[C_N] + \mathbb{I}[C_{\phi(N)}] - 2 \cdot \mathbb{I}[C_N] \cdot \mathbb{I}[C_{\phi(N)}] \\
&= (1 - \mathbb{I}[C_{\phi(N)}]) \cdot \mathbb{I}[C_N] + \mathbb{I}[C_{\phi(N)}] - \mathbb{I}[C_N] \cdot \mathbb{I}[C_{\phi(N)}] \\
&\geq (1 - \mathbb{I}[C_{\phi(N)}]) \cdot \mathbb{I}[C_N].
\end{aligned}$$

Let L be the set of leaves N in T^* for which $\Pr_{x \sim \mathcal{P}}[C_N] \geq \gamma = \frac{\epsilon}{k}$, i.e., at least γ fraction of the probability mass flows to N, and let L' be the remaining leaves. Then, the total error of the decision tree is

$$\begin{aligned}
\Pr_{x \sim \mathcal{P}}[T(x) \neq T^*(x)] &= \sum_{N \in \text{leaves}(T^*)} \Pr_{x \sim \mathcal{P}}[T(x) \neq T^*(x) \mid C_N] \cdot \Pr_{x \sim \mathcal{P}}[C_N] \\
&\leq \sum_{N \in L} \Pr_{x \sim \mathcal{P}}[\neg C_{\phi(N)} \mid C_N] \cdot \Pr_{x \sim \mathcal{P}}[C_N] + \sum_{N \in L'} \Pr_{x \sim \mathcal{P}}[C_N] \\
&\leq \sum_{N \in L} \Pr_{x \sim \mathcal{P}}[\neg C_{\phi(N)} \mid C_N] \cdot \Pr_{x \sim \mathcal{P}}[C_N] + |L'| \cdot \gamma \\
&= \sum_{N \in L} \int (1 - \mathbb{I}[C_{\phi(N)}]) \cdot \mathbb{I}[C_N] \cdot p(x)dx + |L'| \cdot \gamma \\
&\leq \sum_{N \in L} \int |\mathbb{I}[C_N] - \mathbb{I}[C_{\phi(N)}]| \cdot p(x)dx + |L'| \cdot \gamma \\
&\leq \sum_{N \in L} \int |\mathbb{I}[C_N] - \mathbb{I}[C_{\phi(N)}]| \cdot p(x)dx + |L'| \cdot \gamma \\
&= \sum_{N \in L} \|p_N - \tilde{p}_N\|_1 + |L'| \cdot \gamma \\
&\leq |L| \cdot \epsilon' + |L'| \cdot \gamma \\
&\leq \epsilon,
\end{aligned}$$

where in the last step, we choose $\epsilon' = \frac{\epsilon}{k}$. This inequality holds with probability at least $1 - \frac{\delta}{2} - k \cdot \delta'$, by taking a union bound, including over the probability $1 - \frac{\delta}{2}$ that the label choices described above hold. Therefore, the result follows taking $\epsilon' = \frac{\epsilon}{2k}$ and $\delta' = \frac{\delta}{4k}$. □

A.3 Proof of Corollary 1

Proof. Let $\epsilon, \delta > 0$ be arbitrary. We show that for the claimed value of n, the decision tree T extracted by our algorithm is (ϵ, δ) exact. Following the proof of Thereom 1, it suffices to show that for each node N in T^*, either

$$\|p_N - p_{\phi(N)}\|_1 \leq \frac{\epsilon}{2k}$$

with probability at least $\delta' = \frac{\delta}{4k}$, or

$$\mathrm{Pr}_{x\sim\mathcal{P}}[C_N] \leq \gamma = \frac{\epsilon}{k}.$$

Consider the inequality

$$\mathrm{Pr}_{x^{(1)},\ldots,x^{(n)}}[\|p_N - \tilde{p}_N\|_1 \geq \epsilon''] \leq \delta''.$$

Recall that at the root $N = N_T$, this inequality holds for all $\epsilon'', \delta'' > 0$. By Corollary 2, as we descend the tree from N to a child node N', either this relation holds where the parameters ϵ'' picks up a factor of

$$1 + \frac{4m}{(\epsilon/2k)^{3/2}} \cdot \frac{4}{\kappa}$$

(where $m = |\mathcal{Y}|$) and δ picks up a factor of 2, or the probability mass flowing to that node becomes small, i.e.,

$$\mathrm{Pr}_{x\sim\mathcal{P}}[C_{N'}] \leq \frac{\epsilon''}{k}.$$

Therefore, to obtain the desired bound for the leaf nodes (which are at depth $D = \log k$, where k is the number of nodes in T and T^*), we need

$$\epsilon'' = \left(1 + \frac{4m}{(\epsilon/2k)^{3/2}} \cdot \frac{4}{\kappa}\right)^{-D} \cdot \frac{\epsilon}{k}$$
$$\delta'' = 2^{-D} \cdot \frac{\delta}{4k}$$

at the root node. By Corollary 2, to obtain these parameters, we need $n \in \mathbb{N}$ samples, where $n \geq 3$ and

$$\frac{2(\epsilon/2k)^{3/2}}{\sqrt{\epsilon/k}} \cdot \frac{\log n}{\sqrt{n}} \leq \epsilon'' = \left(1 + \frac{4m}{(\epsilon/2k)^{3/2}} \cdot \frac{4}{\kappa}\right)^{-D} \cdot \frac{\epsilon}{k}$$
$$\frac{2}{n^{3/2}} \leq \delta'' = 2^{-D} \cdot \frac{\delta}{4k}.$$

It is easy to check that for $n \geq 4$, taking

$$n \geq \max\left\{\left(1 + \frac{16m}{(\epsilon/2k)^{3/2} \cdot \kappa}\right)^{3D}, \frac{2^{D+1}k}{\delta}\right\}.$$

suffices to satisfy these inequalities. □

A.4 Gap for Random Forests

We briefly discuss why intuitively, axis-aligned random forests should satisfy the premise of Corollary 1. In particular, note that axis-aligned random forests are piecewise constant, and furthermore the pieces are axis aligned. Therefore, the label frequencies $\mathrm{Pr}_{x\sim\mathcal{P}}[f(x) = y \mid C]$ are piecewise linear, so the gain in (1) is quadratic. As we showed in Lemma 1, we can extract a square root from the gain without affecting our theoretical results, in which case the gain becomes approximately linear again. Another issue is that we are integrating against $p(x)$, and the presence of the weighting term $\mathrm{Pr}_{x\sim\mathcal{P}}[C]$. However, as long as $p(x)$ is fairly close to uniform, then its effect on the gain is fairly negligible, so we expect that the premises are satisfied.

B Proofs of Technical Lemmas

In this section, we prove the technical lemmas required for our proofs of Lemma 2, Theorem 1, and Corollary 1.

B.1 Proof of Existence of a Gap

We prove that a gap exists for reasonably behaved functions.

Lemma 2. Let $g : \mathbb{R} \to [0, 1]$ be a continuous function with bounded support, and assume that its global maximizer

$$t^* = \arg\max_{t \in \mathbb{R}} g(t)$$

is unique. Then, for any $\epsilon' > 0$, there exists $\delta' > 0$ such that g is (ϵ', δ') gapped.

Proof. Let t_{max} be a bound on the support of g, i.e., $g(t) = 0$ if $|t| > t_{\text{max}}$. Let $\epsilon' > 0$ be arbitrary, and let

$$A_{\epsilon'} = \{t \in \mathbb{R} \mid |t| \leq t_{\text{max}} \text{ and } |F(t) - F(t^*)| \geq \epsilon'\}.$$

Note that $A_{\epsilon'}$ is a compact set, so g achieves its maximum on $A_{\epsilon'}$, i.e.,

$$t^*_{\epsilon'} = \arg\max_{t \in A_{\epsilon'}} g(t).$$

Then, the result follows for

$$\delta' = \frac{g(t^*) - g(t^*_{\epsilon'})}{2} > 0.$$

Note that we divide by 2 since the inequality in Definition 2 is strict. □

B.2 Bound on Intrinsic Error

The following lemma enables us to bound the intrinsic error $\|g - \tilde{g}\|_\infty$ given that the distributions have probability density functions bounded in L_1 norm.

Lemma 3. Let $\mathcal{X} \subseteq \mathbb{R}^d$, and let $\mathcal{P}$ and $\tilde{\mathcal{P}}$ be distributions over $\mathcal{X}$, and let $p(x)$ and $\tilde{p}(x)$ be unnormalized probability density functions for $\mathcal{P}$ and $\tilde{\mathcal{P}}$, respectively, that satisfy $\|p - \tilde{p}\|_1 \leq \epsilon$. Let $\beta : \mathcal{X} \times \mathbb{R} \to [0, 1]$ be any function, and let

$$g(t) = \int \beta(x, t) \cdot p(x) dx$$
$$\tilde{g}(t) = \int \beta(x, t) \cdot \tilde{p}(x) dx.$$

Then, we have

$$\|g - \tilde{g}\|_\infty \leq \epsilon.$$

Proof. Note that

$$\begin{aligned}
\|g - \tilde{g}\|_\infty = \sup_{t \in \mathbb{R}} |g - \tilde{g}| &\leq \sup_{t \in \mathbb{R}} \int \beta(x, t) |p(x) - \tilde{p}(x)| dx \\
&\leq \sup_{t \in \mathbb{R}} \int |p(x) - \tilde{p}(x)| dx \\
&\leq \epsilon,
\end{aligned}$$

as claimed. □

B.3 Bound on Estimation Error

The following lemma enables us to bound the estimation error $\|\tilde{g} - \hat{\tilde{g}}\|_\infty$.

Lemma 4. Let $\mathcal{X} \subseteq \mathbb{R}^d$, let $\mathcal{P}$ be a distribution over $\mathcal{X}$, and let $p(x)$ be an unnormalized probability density function for $\mathcal{P}$. Let $\alpha : \mathcal{X} \to [0, 1]$ be an arbitrary function, let $i \in [d]$, and let

$$g(t) = \int \alpha(x) \cdot \mathbb{I}[x_i \leq t] \cdot p(x)dx.$$

Let $x^{(1)}, ..., x^{(n)}$ be i.i.d. samples from $\mathcal{P}$, and let

$$\hat{g}(t) = \frac{Z}{n} \sum_{i=1}^{n} \alpha(x^{(k)}) \cdot \mathbb{I}[x_i^{(k)} \leq t]$$

be the empirical estimate of g on these points, where

$$Z = \int p(x)dx$$

is a normalization constant. Then, we have

$$\Pr_{x^{(1)},...,x^{(n)} \sim \mathcal{P}} \left[\|g - \hat{g}\|_\infty \geq \frac{4 \log n}{\sqrt{n}} \right] \leq \frac{2}{n^{3/2}},$$

for any $n \geq 3$.

Proof. First, we define points $t_0, t_1, ..., t_{\sqrt{n}} \in \mathbb{R}$ that divide $\mathbb{R}$ into $\sqrt{n}$ intervals according to the marginal probability density function $p_i(t)$ along dimension i (for convenience, we assume n is a perfect square), which has corresponding cumulative distribution function $F_i(t)$. Then, we choose t_j to satisfy

$$t_j \in F_i^{-1} \left(\frac{j \cdot Z}{\sqrt{n}} \right).$$

For convenience, we choose $t_0 = -\infty$ and $t_{\sqrt{n}} = \infty$, which satisfy the condition. Now, for each $j \in [\sqrt{n}]$, let $I_j = (t_{j-1}, t_j]$. Note that these intervals cover $\mathbb{R}$, i.e., $\mathbb{R} = I_1 \cup ... \cup I_{\sqrt{n}}$.

Then, we can decompose the quantity $\|g - \hat{g}\|_\infty$ into three parts:

$$\begin{aligned}
\|g - \hat{g}\|_\infty &= \sup_{t \in \mathbb{R}} |g(t) - \hat{g}(t)| \\
&= \sup_{j \in [\sqrt{n}]} \sup_{t \in I_j} |g(t) - \hat{g}(t)| \\
&\leq \sup_{j \in [\sqrt{n}]} \sup_{t \in I_j} \{|g(t) - g(t_j)| + |g(t_j) - \hat{g}(t_j)| + |\hat{g}(t_j) - \hat{g}(t)|\} \\
&\leq \sup_{j \in [\sqrt{n}]} \sup_{t \in I_j} |g(t) - g(t_j)| + \sup_{j \in [\sqrt{n}]} |g(t_j) - \hat{g}(t_j)| + \sup_{j \in [\sqrt{n}]} \sup_{t \in I_j} |\hat{g}(t_j) - \hat{g}(t)|.
\end{aligned}$$

We show that each of these three parts can be made arbitrarily small with high probability by taking n sufficiently large.

First, consider the term $\sup_{j \in [\sqrt{n}]} \sup_{t \in I_j} |g(t) - g(t_j)|$. Since $t \leq t_j$, we have $\mathbb{I}[x_i \leq t] \leq \mathbb{I}[x_i \leq t_j]$. Thus, for all $t \in I_j$, we have

$$\begin{aligned}
|g(t) - g(t_j)| &= \left| \int \alpha(x) \cdot (\mathbb{I}[x_i \leq t] - \mathbb{I}[x_i \leq t_j]) \cdot p(x)dx \right| \\
&\leq \int (\mathbb{I}[x_i \leq t_j] - \mathbb{I}[x_i \leq t]) \cdot p(x)dx \\
&= F_i(t_j) - F_i(t) \\
&\leq n^{-1/2},
\end{aligned}$$

where the last inequality follows from the definition of t_j and the fact that $t \in I_j$.

Second, consider the term $\sup_{j\in[\sqrt{n}]} |g(t_j) - \hat{g}(t_j)|$. Note that each $Z \cdot \alpha(x^{(k)}) \cdot \mathbb{I}[x_i^{(k)} \geq t]$ is a random variable in $[0, 1]$. Therefore, by the Hoeffding inequality, we have

$$\text{Pr}_{x^{(1)},\ldots,x^{(n)}\sim\mathcal{P}}\left[|g(t_j) - \hat{g}(t_j)| \geq \frac{\log n}{n}\right] \leq e^{-2(\log n)^2} \leq \frac{1}{n^2}$$

for $n \geq 3$. By a union bound, this inequality holds for every $j \in [\sqrt{n}]$ with probability $n^{-3/2}$.

Third, consider the term $\sup_{j\in[\sqrt{n}]} \sup_{t\in I_j} |\hat{g}(t_j) - \hat{g}(t)|$. We first show that for every $j \in [\sqrt{n}]$, the interval I_j contains at most $n^{1/2} \log n$ of the points $x^{(1)}, ..., x^{(n)}$ with high probability. By the definition of the points t_j, the probability that a single randomly selected point $x^{(k)}$ falls in I_j is $n^{-1/2}$ (since the points t_j were constructed according to the cumulative distribution function F_i):

$$M = \mathbb{E}_{x\sim\mathcal{P}}\left[\mathbb{I}[x \in I_j]\right] = \text{Pr}_{x\sim\mathcal{P}}[x \in I_j] = \frac{1}{\sqrt{n}}.$$

Then, the fraction of the n points $x^{(k)}$ that fall in the interval I_j is

$$\hat{M} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}[x^{(k)} \in I_j].$$

Note that each $\mathbb{I}[x^{(k)} \in I_j]$ is an random variable in $[0, 1]$, so by Hoeffding's inequality, we have

$$\text{Pr}_{x^{(1)},\ldots,x^{(n)}\sim\mathcal{P}}\left[|\hat{M} - M| \geq \frac{\log n}{\sqrt{n}}\right] \leq e^{-2(\log n)^2} \leq \frac{1}{n^2}.$$

Now, note that each point $x^{(k)}$ in I_j can increase the value of $|\hat{g}(t_j) - \hat{g}(t)|$ by at most n^{-1}. Since there are $n \cdot \hat{M}$ points $x^{(k)}$ in I_j, the total increase is bounded by $\hat{M}$, i.e.,

$$\text{Pr}_{x^{(1)},\ldots,x^{(n)}\sim\mathcal{P}}\left[\sup_{t\in I_j} |\hat{g}(t_j) - \hat{g}(t)| \geq \frac{2\log n}{\sqrt{n}}\right] \leq \frac{1}{n^2}.$$

As before, by a union bound, this inequality holds for every $j \in [\sqrt{n}]$ with probability $n^{-3/2}$.

Putting these three results together, we can conclude that for $n \geq 3$, we have

$$\text{Pr}_{x^{(1)},\ldots,x^{(n)}\sim\mathcal{P}}\left[\|g - \hat{g}\|_\infty \geq \frac{4\log n}{\sqrt{n}}\right] \leq \frac{2}{n^{3/2}},$$

as claimed. □

B.4 Bound on Error of Maximizers

The following lemma enables us to bound the maximizer of two functions that are close in L_∞ norm.

Lemma 5. Let $\mathcal{P}$ be a probability distribution over $\mathbb{R}$, and let $F(t)$ be an cumulative distribution function for $\mathcal{P}$. Suppose that $g : \mathbb{R} \to \mathbb{R}$ is (ϵ, δ) gapped according to $\mathcal{P}$, and $h : \mathbb{R} \to \mathbb{R}$ satisfies $\|g - h\|_\infty \leq \frac{\delta}{2}$. Let

$$t_g^* = \arg\max_{t\in\mathbb{R}} g(t)$$
$$t_h^* = \arg\max_{t\in\mathbb{R}} h(t).$$

Then, we have $|F(t_g^*) - F(t_h^*)| \leq \epsilon$.

Proof. By definition of the L_∞ norm, we have

$$g(t_g^*) - h(t_g^*) \leq \frac{\delta}{2}$$
$$h(t_h^*) - g(t_h^*) \leq \frac{\delta}{2}.$$

Combining these gives

$$g(t_g^*) - g(t_h^*) \leq \delta + h(t_g)^* - h(t_h^*) \leq \delta,$$

where the second inequality follows because t_h^* is a maximizer of h. Since g is (ϵ, δ) gapped, and since t_g^* is the maximizer of g, this inequality implies the claim. □

B.5 Bound on Error of Probabiliy Density Functions

The following lemma enables us to bound the L_1 error of the estimated probability density function.

Lemma 6. Let $\mathcal{X} \subseteq \mathbb{R}^d$, let $\mathcal{P}$ and $\tilde{\mathcal{P}}$ be distributions over $\mathcal{X}$, and let $p(x)$ and $\tilde{p}(x)$ be unnormalized probability density functions for $\mathcal{P}$ and $\tilde{\mathcal{P}}$, respectively, satisfying $\|p - \tilde{p}\| \leq \epsilon$. Let $i \in [d]$, let $\mathcal{P}_i$ be the marginal distribution of $\mathcal{P}$ along dimension i, let F_i be an unnormalized cumulative distribution function for $\mathcal{P}_i$, and let $t, \tilde{t} \in \mathbb{R}$ satisfying $|F_i(t) - F_i(\tilde{t})| \leq \epsilon'$. Let $i \in [d]$, and let

$$p'(x) = p(x) \cdot \mathbb{I}[x_i \leq t]$$
$$\tilde{p}'(x) = \tilde{p}(x) \cdot \mathbb{I}[x_i \leq \tilde{t}]$$

be the unnormalized probability density functions for $\mathcal{P} \mid (x_i \leq t)$ and $\tilde{\mathcal{P}} \mid (x_i \leq \tilde{t})$, respectively. Then, we have

$$\|p' - \tilde{p}'\|_1 \leq \epsilon + \epsilon'.$$

Proof. Assume without loss of generality that $t \leq \tilde{t}$. Then, we have

$$\begin{aligned}
\|p' - \tilde{p}'\|_1 &= \int |p'(x) - \tilde{p}'(x)| dx \\
&= \int |p(x) \cdot \mathbb{I}[x_i \leq t] - \tilde{p}(x) \cdot \mathbb{I}[x_i \leq \tilde{t}]| dx \\
&= \int |p(x) \cdot \mathbb{I}[x_i \leq t] - (p(x) + \tilde{p}(x) - p(x)) \cdot \mathbb{I}[x_i \leq \tilde{t}]| dx \\
&\leq \int p(x) \cdot |\mathbb{I}[x_i \leq t] - \mathbb{I}[x_i \leq \tilde{t}]| dx + \int |\tilde{p}(x) - p(x)| \cdot \mathbb{I}[x_i \leq \tilde{t}] dx \\
&\leq \int p(x) \cdot |\mathbb{I}[x_i \leq t] - \mathbb{I}[x_i \leq \tilde{t}]| dx + \epsilon \\
&= \int p(x) \cdot \mathbb{I}[t \leq x_i \leq \tilde{t}] dx + \epsilon \\
&= |F_i(\tilde{t}) - F_i(t)| + \epsilon \\
&\leq \epsilon' + \epsilon,
\end{aligned}$$

as claimed. □