Active Learning for Non-Parametric Regression Using Purely Random Trees

Jack Goetz **Ambuj Tewari** **Paul Zimmerman**
University of Michigan
Ann Arbor, MI 48109
{jrgoetz, tewaria, paulzim}@umich.edu

Abstract

Active learning is the task of using labelled data to select additional points to label, with the goal of fitting the most accurate model with a fixed budget of labelled points. In binary classification active learning is known to produce faster rates than passive learning for a broad range of settings. However in regression restrictive structure and tailored methods were previously needed to obtain theoretically superior performance. In this paper we propose an intuitive tree based active learning algorithm for non-parametric regression with provable improvement over random sampling. When implemented with Mondrian Trees our algorithm is tuning parameter free, consistent and minimax optimal for Lipschitz functions.

1 Introduction

In this paper we study active learning for regression in the pool setting. In our setup we are given a pool of unlabelled data points and want to build the best model with a fixed number of samples, allowing selection of new points to use labels already obtained. Active learning is motivated by scenarios where the experimenter has control over the data labelling process and where unlabelled points are cheap but labels are expensive.

Our primary motivation comes from computational chemistry, where chemical properties of interest can be computed by solving approximations to the Schrödinger equation. One key property to chemists, the rate of chemical reaction, can be quantified via the activation energy, which controls the rate of reaction as a function of temperature [9]. While calculating the activation energy is expensive, there are a small number of readily available features of the reaction that influence the activation energy. This incentivizes building a metamodel for the activation energy to avoid excessive analysis of undesirable (high activation energy) reactions. Since we are restricted in the number of simulations used to build our metamodel, we want to use the most informative data points. Because chemical reactions are discrete entities, we are restricted to a finite (but often large) pool of reactions, thus requiring pool setting active learning even though we are selecting simulations.

Active learning methods are usually built on top of existing prediction algorithms. Decision trees and forests are a popular class of such predictors due to their simplicity, expressiveness, state-of-the-art performance and tuning parameter free nature. In this paper we focus our attention on purely random trees [4], decision trees built independently of any data, due to their amenability to theoretical analysis. We use a recently proposed version called Mondrian Trees [17], which have been shown to produce trees with many attractive properties such as consistency and minimax optimal rate of convergence for Lipschitz functions [19].

As in some previous work [7], our active learning algorithm will be developed in two stages. First we introduce a simple and intuitive *oracle* querying algorithm for purely random trees which is optimal among a natural class of sampling schemes which includes random sampling (Theorem

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

4.4). This algorithm is not active but uses statistics of the true joint distribution which are generally unknown. Second we propose an active learning scheme where we first sample passively to estimate the required statistics, and then use those estimates to approximate the oracle algorithm. We show this algorithm is consistent for the oracle algorithm (Theorem 5.1) and behaves well when our labels are normally distributed (Theorem 5.4). Finally we examine the empirical performance of our active learning algorithm to show that benefits, though sometimes modest, can be significant.

2 Setting and background

We begin by describing the pool based active learning setting, as well as introducing purely random and Mondrian trees. We have a pool of m data points $\{X_i\}_1^m$, with $X_i \in [0,1]^d$ (rescaling our X as needed) and $X_i \sim p_X$, which are always available to the algorithm. For each X_i we have a corresponding label $Y_i \in \mathbb{R}$ with the relationship $Y_i = f(X_i) + \sigma(X_i)\epsilon_i$ with $\epsilon_i \sim p_\epsilon$ iid, $\epsilon_i \perp X_j \; \forall \, j, E(\epsilon_i) = 0, \text{Var}(\epsilon_i) = 1, \sigma(X_i) : [0,1]^d \to \mathbf{R}_+$, meaning our noise is the product of a function of X with an independent random variable. We assume the $(X_i, Y_i) = D_i$ have been drawn iid from a joint distribution $p_{X,Y}$. We will assume that $f(x)$ and $\sigma(x)$ are bounded.

Initially none of these Y_i are known to the algorithm. Instead we have the ability to gain access to any of the Y_i, and the task is to select $n \ll m$ labels with the goal of building a model with the lowest quadratic risk $E\left[(\hat{f}(X) - f(X))^2\right]$, where the expectation is taken over our test point X, the random process which builds our tree and the labelled data we select. Throughout we will assume that our pool is arbitrarily large; in particular we will assume that the marginal density p_X is known, and that there is enough unlabelled data to implement any sampling scheme for selecting n points. We use *active* sampling (or learning) to describe any sampling scheme which samples in multiple batches and uses both $X_i's$ as well as known $Y_i's$ from previous batches when picking points for the next batch. We use passive sampling to denote any sampling scheme which only uses the X_i to pick points, and we use *random* sampling to denote picking the points uniformly at random from our pool (which is the same as sampling from $p_{X,Y}$).

Our active learning method is for purely random trees [4], which are decision trees (or partitions of the space) built using a random process that is independent of the data. We will interchangeably discuss the partition of the space generated by the tree and the leaves of the tree. Let $I_k \in \mathcal{I}$ enumerate the leaves of a tree (partitions of the space), where $k \in \{1...K\}$. We will abuse notation slightly and use the set of partitions $\mathcal{I}$ to denote our tree. These partitions can be used to build regressograms, which make predictions using the average of labelled points within the partition of the test point. With the partitions fixed, the best (in L_2) approximation to f which is piece-wise constant on each partition predicts the conditional mean on that partition [14]. We will denote true values and estimates of this approximation using "tilde" and "hat" notation as shown below.

True best approximation	Estimate of best approximation
$\tilde{f}_{\mathcal{I}}(x) = \sum_{k=1}^{K} \mathbf{1}(x \in I_k)\tilde{\beta}_k$	$\hat{f}_{\mathcal{I}}(x) = \sum_{k=1}^{K} \mathbf{1}(x \in I_k)\hat{\beta}_k$
$\tilde{\beta}_k = E_{p_{X,Y}}[Y \mid X \in I_k]$	$\hat{\beta}_k = \frac{1}{\sum \mathbf{1}(X_i \in I_k)} \sum_{X_i \in I_k} Y_i$

Our experiments and some results will use particular purely random trees built using the Mondrian Process [17]. The Mondrian Process is a stochastic process for partitioning a hypercube in $\mathbb{R}^d$, a single realization of this process gives a Mondrian Tree. The Mondrian Process iteratively splits existing partitions, and the number of partitions is controlled by a parameter λ which, since the Mondrian Process is a generalization of a Poisson Process, is referred to as the *lifetime* parameter. As this parameter increases the number of partitions increases, and the rate at which the number of partitions increase depends on the dimension and size of the hypercube. We will use Mondrian Trees on a fixed domain $[0,1]^d$ with varying lifetime as in [19], which describes how these random partitions are built.

3 Related work on Active Learning

The majority of theoretical work in active learning has taken place in binary classification, and there are many approaches which have been studied (see, e.g. [13], [10], [24], [16], [3], [2]). These algorithms are studied under fairly nonrestrictive assumptions (except occasionally requiring a linear classification boundary). It has been shown that for a variety of realistic noise conditions active learning provides a better minimax learning rate than passive learning ([15]).

In contrast the theory for active learning in regression is less well developed. A negative result [25] showed that for a Lipschitz regression function and constant noise variance, the minimax learning rate for active learning was the same as that for passive (up to a constant). Additional assumptions are required to obtain better rates. Such structure includes assumptions of piece-wise constantness of regression function [25], approximation of a non-linear model by a linear one [22], locally varying smoothness [6], well-specified parametric model [8] or heteroskedasticity [11], [7].

While many of these regression methods are able to provide provably better learning rates in terms of n, d, they are often tailored for their specific assumptions and may perform poorly if the assumptions do not hold. As a recent summary [18] of numerous flexible but guarantee free methods shows, there is great demand for active learning methods without such stringent conditions. Our active learning algorithm will make very mild assumptions, but the improvement will not be in rates in n, d (since it is known this is not always possible). Rather we will adopt the approach [13] of comparing the sampling generated by our algorithm to an optimal sampling scheme, as well as to random sampling.

4 Oracle label querying algorithm

We first describe a simple family of querying algorithms for a fixed purely random tree $\mathcal{I}$ which are not active. In the first two subsections below, we will be implicitly conditioning on the tree $\mathcal{I}$, but will suppress this in the notation.

4.1 Generic algorithm

In our generic algorithm family, the tree is built without using any data. So we build the tree first and query based on the tree's structure. We call it an "oracle" algorithm since it requires $p_{X,Y}$.

Algorithm 1: Generic "oracle" querying algorithm

Input: Leaves of our tree $\mathcal{I}$, pool of data points $\{X_i\}_{i=1}^m$, label budget n and joint distribution $p_{X,Y}$
Output: The set of points to label
foreach $I_k \in \mathcal{I}$ **do**
  Calculate q_k the proportion of points to select from leaf I_k, using $\mathcal{I}, \{X_i\}_{i=1}^m, n, p_{X,Y}$. ;
  Select $n_k = n \cdot q_k$ points uniformly at random from the pool of unlabelled points in that leaf. ;
end

The algorithm is described as picking n_k deterministically for simplification of notation in proofs. However it is clear that if the n_k are random then it is easy (in principle) to discuss the probabilistic properties of the algorithm, and the details of the risk under random versions of Algorithm 1 are discussed in the proof for Corollary 4.6. The pool marginal distribution p_X and the proportion in each leaf q_k from the querying algorithm above induce a marginal distribution p'_X, as well as a joint distribution $p'_{X,Y} = p_{Y|X} p'_X$. The scheme is very general, and it is worth noting that random sampling is a (randomized) version of Algorithm 1. But this is enough structure to produce a somewhat obvious but very important property of our sampling distribution restricted to each leaf.

Proposition 4.1. *Fix a tree structure $\mathcal{I}$, pool marginal density p_X and version of Algorithm 1, giving us an induced marginal density p'_X. Let $p'_X(X|I_k) = p'_X(X|X \in I_k)$ denote the induced marginal density conditioned on $X \in I_k$. Then as long as $q_k \neq 0$, $p'_X(X|I_k) = p_X(X|I_k)$ for any version of Algorithm 1.*

One important property this gives us is that $E_{p'_{X,Y}}[\hat{\beta}_k] = \tilde{\beta}_k$ (as long as I_k has at least 1 labelled point to estimate $\hat{\beta}_k$), meaning our sampling scheme produces unbiased estimates of the optimal regressogram for this tree. It also allows for a bias-variance decomposition of the risk of the tree.

This decomposition was already known [12] under the assumption of independence between tree structure and the data. We relax this assumption slightly as the distribution of the data depends on the structure of the tree, but still permits this decomposition.

Corollary 4.2. *For a fixed tree structure $\mathcal{I}$, under any sampling distribution generated by Algorithm 1 we have the following bias-variance decomposition of our risk:*

$$E\Big[(\hat{f}_{\mathcal{I}}(X) - f(X))^2\Big] = E\Big[(\tilde{f}_{\mathcal{I}}(X) - f(X))^2\Big] + E\Big[(\hat{f}_{\mathcal{I}}(X) - \tilde{f}_{\mathcal{I}}(X))^2\Big].$$

We will refer to these as the *risk bias term* and *risk variance term.* The risk bias term depends only on the structure of the tree, which does not depend our sampling scheme. We thus focus on the risk variance term. Again using Proposition 4.1 we show this term for a single leaf takes a simple form.

Lemma 4.3. *For a fixed tree structure $\mathcal{I}$, under any sampling distribution generated by Algorithm 1 we have that the variance error term on the leaf I_k is:*

$$E\Big[(\hat{f}_{\mathcal{I}}(X) - \tilde{f}_{\mathcal{I}}(X))^2 | X \in I_k\Big] = \frac{1}{n_k}\big(bias_k^2 + \sigma_{\epsilon,k}^2\big) = \frac{1}{n_k} Var(Y|X \in I_k),$$
$$bias_k^2 := E_{p_{X,Y}}\Big[(f(X) - \tilde{\beta}_k)^2 | X \in I_k\Big], \quad \sigma_{\epsilon,k}^2 := E_{p_{X,Y}}\Big[(\sigma(X)\epsilon)^2 | X \in I_k\Big].$$

4.2 Optimal algorithm

In the above lemma we have emphasized that the terms $bias_k^2$ and $\sigma_{\epsilon,k}^2$ have expectations taken with respect to the data generating distribution $p_{X,Y}$ and do not depend on the induced distribution $p'_{X,Y}$. Thus the only way our sampling distribution affects the variance term is through n_k. Averaging out over the contribution of each leaf we get that our overall variance error term is.

$$E\Big[(\hat{f}_{\mathcal{I}}(X) - \tilde{f}_{\mathcal{I}}(X))^2\Big] = \sum_k P(X \in I_k)\frac{1}{n_k}\big(bias_k^2 + \sigma_{\epsilon,k}^2\big). \tag{1}$$

Let $p_k = P(X \in I_k)$ under the pool marginal distribution and $\sigma_{Y,k}^2 = bias_k^2 + \sigma_{\epsilon,k}^2$. Now we are given a budget of n data points, and we want to minimize our variance error term subject to this budget. This gives us the following optimization problem which can be easily solved:

$$\begin{array}{ll} \underset{n_k}{\text{minimize}} & \sum_k \frac{1}{n_k} p_k \sigma_{Y,k}^2 \\ \text{subject to} & \sum_k n_k = n \end{array} \qquad \rightarrow \qquad n_k^* = n\frac{\sqrt{p_k\sigma_{Y,k}^2}}{\sum_{k'}\sqrt{p_{k'}\sigma_{Y,k'}^2}}$$

The proportions are very intuitive; cells with high bias and/or noise, or high (test) marginal density will get more samples. These results are summarized in the following theorem:

Theorem 4.4. *Let $Y_i = f(X_i) + \sigma(X_i)\epsilon_i$ and fix the partitions $\mathcal{I}$ of our tree. The risk minimizing oracle querying algorithm out of the family of algorithms described by Algorithm 1 is the one with the following n_k and error*

$$n_k^* = n\frac{\sqrt{p_k\sigma_{Y,k}^2}}{\sum_{k'}\sqrt{p_{k'}\sigma_{Y,k'}^2}}, \qquad E\Big[(\hat{f}_{\mathcal{I}}(X) - \tilde{f}_{\mathcal{I}}(X))^2\Big] = \frac{1}{n}(\sum_k \sqrt{p_k\sigma_{Y,k}^2})^2.$$

Definition 4.5. The distribution induced by the sampling in Theorem 4.4 will be referred to as p_X^*.

Remark. This has a similar flavour to uncertainty sampling methods from classification in that regions with greater variation will get more samples. However whereas in classification sampling can focus locally near the decision boundary, in regression sampling must remain global.

Random sampling is a randomized version of Algorithm 1, so the risk under random sampling is the bias term plus a weighted average of the variance terms for different $(n_1, ..., n_K)$. The sampling

scheme from Theorem 4.4 has the same bias term, but minimizes the variance term meaning our optimal sampling scheme is better than any randomized version of Algorithm 1 (as long as $m > n$), including random sampling.

Corollary 4.6. *For a fixed tree structure $\mathcal{I}$, the risk from any randomized version of Algorithm 1 is greater than the risk from sampling according to p_X^* unless $P(n_1^*, ..., n_K^*) = 1$. In particular sampling according to p_X^* is strictly better than random sampling.*

We can also calculate the excess error if we use the incorrect values of $\sigma^2_{Y,k}$. Let $\tilde{\sigma}^2_{Y,k} = a_k \sigma^2_{Y,k}$, so a_k is a multiplicative error (we will see that our errors will be multiplicative). Given fixed leaf errors $a_1, ..., a_K$ we can calculate the additional risk generated by using $\tilde{\sigma}^2_{Y,k}$ in our optimal algorithm instead of the true $\sigma^2_{Y,k}$

Lemma 4.7. *For a fixed tree structure $\mathcal{I}$, if $n_k = n\frac{\sqrt{p_k \tilde{\sigma}^2_{Y,k}}}{\sum_{k'} \sqrt{p_{k'} \tilde{\sigma}^2_{Y,k'}}}$ and the variance error term for each leaf is as in Lemma 4.3, then our risk variance is:*

$$E\Big[(\hat{f}_{\mathcal{I}}(X) - \tilde{f}_{\mathcal{I}}(X))^2\Big] = \frac{1}{n}(\sum_k \sqrt{p_k \sigma^2_{Y,k}})^2 + \frac{1}{n}\sum_{k<l}(\frac{\sqrt{a_k}}{\sqrt{a_l}} + \frac{\sqrt{a_l}}{\sqrt{a_k}} - 2)\sqrt{p_k p_l \sigma^2_{Y,k}\sigma^2_{Y,l}}$$
$$:= \text{OPT + EXCESS.}$$

This also lets us get a sense for the suboptimality of random sampling. If we let $a_k = \frac{p_k}{\sigma^2_{Y,k}}$ then we get $n_k = np_k$ which is the expected number of samples per leaf under random sampling, and so for large n the calculated EXCESS term will be close to the excess risk under random sampling. This gives us the following excess error, which can be small (or even zero) as expected since random sampling can be near-optimal. But if there is varying Y variance across the space this can be large:

Corollary 4.8. *For a fixed tree structure $\mathcal{I}$ let $a_k = \frac{p_k}{\sigma^2_{Y,k}}$. Then our excess error is:*

$$\text{EXCESS} = \frac{1}{n}\sum_{k<l}(\sqrt{p_k \sigma^2_{Y,l}} - \sqrt{p_l \sigma^2_{Y,k}})^2 \leq \frac{K}{n}\max_k \sigma^2_{Y,k}.$$

4.3 Additional results using Mondrian Trees

The above results hold for any purely random tree. We will now not assume that $\mathcal{I}$ is fixed, but is randomly built using the Mondrian Process and will take expectation over the tree building process as well. Mondrian Trees trained using random sampling are minimax optimal for Lipschitz regression functions when the sequence of lifetime parameters satisfy $\lambda_n \asymp n^{1/(d+2)}$ and $\text{Var}(Y) < \infty$ [19]. Additionally Mondrian Trees with random sampling are weakly universally consistent under the same lifetime sequence and variance assumption. Since the optimal oracle algorithm has smaller risk we immediately get minimax optimal rates in terms of n, d under the same assumptions lifetime sequence by Proposition 4 in [19] and Theorem 4.4, and weak consistency under Theorem 1 in [20].

Corollary 4.9. *Let our purely random trees be Mondrian Trees with lifetime parameters $\lambda_n \asymp n^{1/(d+2)}$, and let $Y = f(X) + \sigma(X)\epsilon$, Var$(Y) < \infty$. If our training data is sampled according to p_X^* then the resulting regressogram has (as $n, m \to \infty$) minimax optimal rates, in terms of n, d, over Lipschitz functions with $E\Big[(\hat{f}(X) - f(X))^2\Big] = \mathcal{O}(n^{\frac{-2}{2+d}})$ and is weakly consistent.*

5 Active learning algorithm

The oracle querying algorithm has many appealing qualities. However it requires knowledge of the $\sigma^2_{Y,k}$ which are never known in practice. In this section we propose a two stage active "oracle estimating" algorithm to remedy this deficiency. In our first stage we sample $n_{(1)}$ points according to Algorithm 1 and use those samples to calculate estimates $\hat{\sigma}^2_{Y,k}$ of $\sigma^2_{Y,k}$, which in turn produce estimates $\hat{n}_k$ of n_k^*. In the second stage we sample $n_{(2)} = n - n_{(1)}$ points such that the total number of samples in each leaf are $\hat{n}_k$. We analyze the consequences of using these estimates, and show that in the case when Y are normal, our trees are Mondrian Trees, and our Stage 1 samples equally in each leaf, our active algorithm is eventually near optimal with high probability. We also show that

in general this algorithm's estimates $\hat{n}_k$ are consistent for n_k^*. Below we give the active algorithm. By using this algorithm we have introduced two complications: One is the estimates will have errors from using estimates $\hat{\sigma}^2_{Y,k}$. The other comes from reusing the data from Stage 1 in our estimates of $\hat{\beta}_k$. Since active learning is used exactly when data is difficult to label, to make an algorithm which is practically appealing it is important to make the most out of any labelled data. However this introduces dependency between $\hat{\beta}_k$ and $\hat{n}_k$. These issues will each be addressed separately.

Algorithm 2: Active "oracle estimating" algorithm

Input: Leaves of our tree $\mathcal{I}$, pool of data points $\{X_i\}_{i=1}^m$, and label budgets $n_{(1)}, n_{(2)}, n = n_{(1)} + n_{(2)}$.
Output: The set of labelled points.
Stage 1 ;
Query $n_{(1)}$ data points using a version of Algorithm 1. ;
Use those samples (X_i, Y_i) to estimate $\hat{\sigma}^2_{Y,k} = \frac{1}{n_{(1),k}-1} \sum_{X_i \in I_k} (\hat{\beta}_{(1),k} - Y_i)^2$ for each leaf. ;
Stage 2 ;
foreach $I_k \in \mathcal{I}$ **do**
  Calculate $\hat{n}_k = n \frac{\sqrt{p_k \hat{\sigma}^2_{Y,k}}}{\sum_{k'} \sqrt{p_{k'} \hat{\sigma}^2_{Y,k'}}}$ the number of points in the leaf to sample. ;
  Select uniformly at random $n_{(2),k}$ points to query from the leaf so the number of points is $\hat{n}_k$. ;
end

5.1 Using estimates of n_k^*

First we analyze (as n increases) the effect of using the estimates $\hat{\sigma}^2_{Y,k}$. Let us fix a sequence of trees $\mathcal{I}_{(n)}, |\mathcal{I}_{(n)}| = K_n$. Typically our trees will contain more partitions as we get more data. For a given tree we can estimate the required leaf variances unbiasedly using the standard unbiased sample variance on each leaf $\hat{\sigma}^2_{Y,k}$. Therefore as long as our leaf kurtosis $\kappa_{Y,k} = \frac{\sigma^4_{Y,k}}{(\sigma^2_{Y,k})^2}$ (and thus the variance of our sample variance) are all finite, and asymptotically our sample variances on each leaf are consistent for the true variances on each leaf, our estimates $\hat{n}_k \to n_k^*$. We require strong consistency of our variance estimates as a function of both our partitioning method and Stage 1 sampling method, which gives us $\hat{n}_k \to n_k^*$ almost surely. If our trees are grown according to a random process then this strong consistency may be depend on attributes of the tree which my only be true in probability, and in this case we get $\hat{n}_k \to n_k^*$ in probability. Both cases are covered in the below theorem, where generally the b_n denote statistics of the tree and B is either 0 or ∞.

Theorem 5.1. *Assume $\kappa_{Y,k} < \infty \; \forall \; k, n$, and our sequence of trees $\mathcal{I}_{(n)}$ and Stage 1 sampling algorithm is strongly consistent for estimating the conditional variance $E[(Y - f(X))^2 | X = x]$ as some statistic $b_n \to B$. Then if $b_n \to B$ almost surely our estimates $\hat{n}_k \to n_k^*$ almost surely and if $b_n \to B$ in probability our estimates $\hat{n}_k \to n_k^*$ in probability.*

Remark. Note that the condition $\kappa_{Y,k} < \infty \; \forall \; k, n$ is met if $f, \sigma(X)$ are bounded and $\kappa_\epsilon < \infty$.

Now let our sequence of trees be randomly built Mondrian Trees. If we again use $\lambda_n \asymp n^{1/(d+2)}$, as long as $n_{(1)}$ increases linearly with n, these conditions are met when our first round of sampling entails sampling equally in each leaf.

Corollary 5.2. *Let our purely random trees be Mondrian Trees with lifetime parameter sequence $\lambda_n \asymp n^{1/(d+2)}$ and let $n_{(1)} = cn, \; c \in (0,1)$ a constant. Additionally let Stage 1 query by $n_{(1),k} = \frac{n_{(1)}}{K_n} \; \forall \, k$. If $\kappa_{Y,k} < \infty \, \forall \, k, n$ and p_X is bounded away from 0 and ∞ on it's support, so when $p_X > 0$ there exists c, C s.t. $c \leq p_X \leq C$, then our estimates $\hat{n}_k \to n_k^*$ in probability.*

Even with consistency our finite sample estimates will give us some error in $\hat{n}_k$. The variance of our sample variance is $\text{Var}(\hat{\sigma}^2_{Y,k}) = \frac{1}{n_k}(\sigma^4_{Y,k} - (\sigma^2_{Y,k})^2) + \mathcal{O}(\frac{1}{n_k^2}) \approx \frac{1}{n_k}(\kappa_{Y,k} - 1)(\sigma^2_{Y,k})^2$, so our errors will scale multiplicatively with $\sigma^2_{Y,k}$ when our kurtosis $\kappa_{Y,k}$ are bounded. This allows us to use Lemma 4.7 to bound our excess error given bounds on the (multiplicative) error $a_k = \hat{\sigma}^2_{Y,k} / \sigma^2_{Y,k}$.

5.2 Reusing data

Since we are using the data in Stage 1 both to estimate $\hat{n}_k$ as well as in our estimator $\hat{\beta}_k$, we have introduced dependence between the estimated optimal leaf sample size $\hat{n}_k$ and leaf mean estimate contribution from Stage 1. To understand the effects of this dependence we will break up our estimates of the leaf mean as $\hat{\beta}_k = \frac{n_{(1),k}\hat{\beta}_{(1),k}+n_{(2),k}\hat{\beta}_{(2),k}}{n_{(1),k}+n_{(2),k}}$, where $n_{(i),k}, \hat{\beta}_{(i),k}$ are the number and mean estimate during sampling round $i \in \{1, 2\}$. By writing our final mean estimate in terms of our stage-wise mean estimates we can find an expression for this dependence.

Lemma 5.3. *For a fixed tree structure $\mathcal{I}$, under Algorithm 2 the risk variance term becomes:*

$$E[(\hat{\beta}_k - \tilde{\beta}_k)^2] = E_{n_{(2),k}}\Big[\frac{n_{(1),k}^2}{(n_{(1),k}+n_{(2),k})^2} E_{D_{1:n_{(1)}}}\left[(\hat{\beta}_{(1),k} - \tilde{\beta}_k)^2 | n_{(2),k}\right] + \frac{n_{(2),k}\sigma_{Y,k}^2}{(n_{(1),k}+n_{(2),k})^2}\Big].$$

The term $E_{D_{1:n_{(1)}}}\left[(\hat{\beta}_{(1),k} - \tilde{\beta}_k)^2 | n_{(2),k}\right]$ quantifies the dependency introduced by reusing the samples from $n_{(1)}$. The dependency is between the variance of part of our mean estimators $(\hat{\beta}_{(1),1}, ..., \hat{\beta}_{(1),k})$ and $(n_{(2),1}, ..., n_{(2),K}) = g(\hat{\sigma}_{Y,1}^2, ..., \hat{\sigma}_{Y,K}^2)$. When $\hat{\beta}_{(1),k} \perp n_{(2),k}$ we get back our risk variance term from Lemma 4.3. However when there is dependence we no longer have that the n_k^* from Theorem 4.4 are optimal over algorithms with an active stage as in Algorithm 2, since the optimal n_k will depend on the sampling during Stage 1. This dependency can be complex and is generally unknown, though as long as the effect is not too large the n_k^* will still provide a very good solution, and the n_k^* are still better than random sampling. It is worth noting that our active algorithm can take advantage of this dependency in some cases to outperform Algorithm 1, and we informally discuss this in the appendix.

5.3 The Normal case

The complications above depend on the distribution of $a_k = \frac{\hat{\sigma}_{Y,k}^2}{\sigma_{Y,k}^2}$ and the function g, which in general are extremely complicated and hard to analyze for arbitrary f, p_ϵ, p_X. However in the case where Y are normally distributed these become tractable.

Theorem 5.4. *Let $Y \sim N(\mu(X), \sigma^2(X))$ and X queried according to Algorithm 2 for a fixed tree $\mathcal{I}$. Then the risk variance term for a leaf is as in Lemma 4.3 and we have that with probability at least $1 - \sum_{k=1}^{K} e^{-\frac{(n_{(1),k}-1)\alpha^2}{8}}$ the excess error is bounded by:*

$$\text{EXCESS} \leq \frac{1}{n}\sum_{k<l}\Big[\big(\frac{1+\alpha}{1-\alpha}\big)^{1/4} - \big(\frac{1-\alpha}{1+\alpha}\big)^{1/4}\Big]^2 \sqrt{p_k p_l \sigma_{Y,k}^2 \sigma_{Y,l}^2}\,.$$

Additionally if our trees are a sequence of Mondrian Trees with lifetime parameter sequence $\lambda_n \asymp n^{1/(d+2)}$ and our Stage 1 sampling procedure is to sample equally in each leaf with $n_{(1)} = cn$, $c \in (0, 1)$ a constant, then the above bound occurs with probability at least $1 - \delta_1 - \delta_2$ where

$$\delta_1 = \frac{(1+n^{1/(d+2)})^d}{n^{(d+1)/(d+2)}} \quad \delta_2 = n^{(d+1)/(d+2)} \exp\big(\frac{-\alpha^2}{8}((cn)^{1/(d+2)}) - 1\big).$$

First, note that a larger n allows us to choose a smaller α and the bound on excess error goes to 0 as $\alpha \to 0$. Second, even for the normal case, d large requires a very large n before we get any control on the error probability δ_2. This is consistent with the empirical observation that Mondrian Trees struggle with large d.

Finally we also note that there are many reasons why in practice it is impossible to use the exact n_k^*. These include the fact that usually n_k^* will be fractional, a leaf may not have n_k^* points in it, or when using the active algorithm $n_{(1),k} > \hat{n}_k$. These issues will be less significant as $n \to \infty$ and we discuss how each is dealt with in the appendix.

6 Simulations and experiments

We now examine the benefits of active learning on both simulated and real world data. We simulate 2 data sets, one with differing noise variance (our $\sigma^2_{\epsilon,k}$ term), the other with differing function complexity (our $bias^2_k$ term), in different regions of $[0,1]^d$. We also examine performance on the Wine quality data set from UCI and a data set of activation energies of Claisen rearrangement reactions (Cl). We compare the performance of selecting points to label using random sampling, our active algorithm, and a naive uncertainty sampling version of our active algorithm, where each leaf n_k is proportional its variance. In all experiments $n_{(1)} = \frac{n}{2}$ and Mondrian Trees are grown using $\lambda_n = n^{\frac{2}{2+d}} - 1$, which is theoretically motivated, but corrected so when $n = 1, \lambda_n = 0$. We use both Mondrian and Breiman Trees [5] as our final regressor. Details of the data sets are in the appendix, which also contains forest versions of these experiments. Additionally all code and experiments (as well as other experiments) are available at `https://github.com/jackrgoetz/Mondrian_Tree_AL`.

When using Mondrian Trees as the final regressor, the active learning method always provides some improvement, and in the simulations this improvement persists when using Breiman Trees. Additionally the uncertainty sampling method sometimes produces worse sampling than random sampling, which is common for direct translations of classification active learning methods. In the real data our benefits are less pronounced, with active learning even being slightly harmful when used with Breiman Trees (although with forests the active learning is beneficial). We believe this performance drop may be due to the inability of the Mondrian Tree to adapt to differing variable importance. It is also possible that our assumptions that Y has changing variance does not hold, and even here the active algorithm is not harmful, where as the naive uncertainty sampling algorithm can be detrimental.

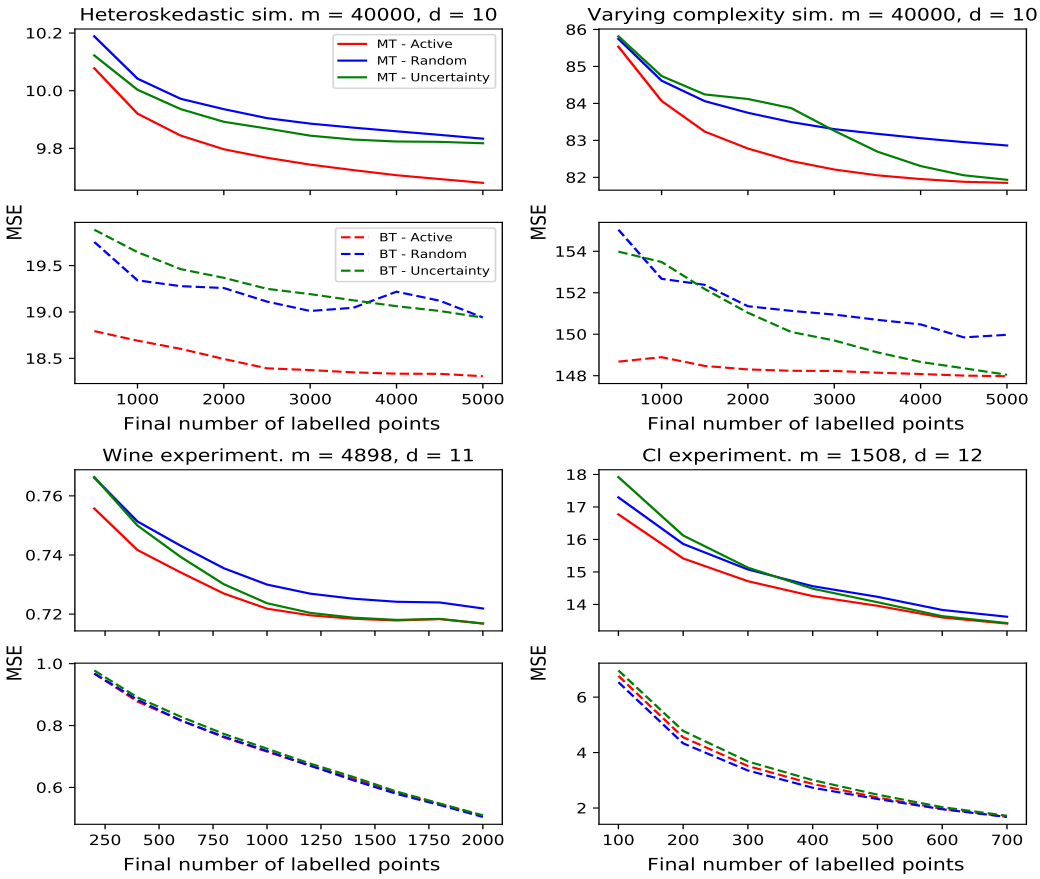

Figure 1: Active learning experiments

7 Conclusion and further directions

In this paper we provide a theoretically justified active learning method for non-parametric regression which can take advantage of beneficial structure when present without being detrimental when such structure is absent. When used with Mondrian Trees the method requires no tuning parameters (which are difficult to tune while actively sampling [1]), is asymptotically minimax optimal for Lipschitz regression functions, and is consistent. Although the improvement for active learning in regression is often restricted to constant factor improvements, these constant improvements are important in real world applications.

Despite technical theoretical arguments needed for the theory, the method itself is simple, leading to many interesting avenues for further exploration. One direction would be extending theory to ensembles of trees, or developing tools to deal with high dimensions. Another possibility is to exploit the online nature of Mondrian Trees to develop a parallel theory for streaming based active learning. Finally it may be possible to extend the ideas here to non tree based active learning for regression.

Acknowledgements

JG acknowledges the support of NSF via grant DMS-1646108. AT acknowledges the support of a Sloan Research Fellowship.

References

[1] Attenberg, J. and Provost, F. (2011). Inactive learning?: difficulties employing active learning in practice. *ACM SIGKDD Explorations Newsletter*, 12(2):36–41.

[2] Awasthi, P., Balcan, M. F., and Long, P. M. (2014). The power of localization for efficiently learning linear separators with noise. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 449–458. ACM.

[3] Balcan, M.-F., Beygelzimer, A., and Langford, J. (2009). Agnostic active learning. *Journal of Computer and System Sciences*, 75(1):78–89.

[4] Breiman, L. (2000). Some infinity theory for predictor ensembles. Technical report, Technical Report 579, Statistics Dept. UCB.

[5] Breiman, L. (2017). *Classification and regression trees*. Routledge.

[6] Bull, A. D. (2013). Spatially-adaptive sensing in nonparametric regression. *The Annals of Statistics*, 41(1):41–62.

[7] Chaudhuri, K., Jain, P., and Natarajan, N. (2017). Active heteroscedastic regression. In *International Conference on Machine Learning*, pages 694–702.

[8] Chaudhuri, K., Kakade, S. M., Netrapalli, P., and Sanghavi, S. (2015). Convergence rates of active learning for maximum likelihood estimation. In *Advances in Neural Information Processing Systems*, pages 1090–1098.

[9] Cramer, C. J. (2013). *Essentials of computational chemistry: theories and models*. John Wiley & Sons.

[10] Dasgupta, S., Hsu, D. J., and Monteleoni, C. (2008). A general agnostic active learning algorithm. In *Advances in neural information processing systems*, pages 353–360.

[11] Efromovich, S. (2008). Optimal sequential design in a controlled non-parametric regression. *Scandinavian Journal of Statistics*, 35(2):266–285.

[12] Genuer, R. (2012). Variance reduction in purely random forests. *Journal of Nonparametric Statistics*, 24(3):543–562.

[13] Golovin, D. and Krause, A. (2011). Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486.

[14] Györfi, L., Kohler, M., Krzyzak, A., and Walk, H. (2006). *A distribution-free theory of nonparametric regression*. Springer Science & Business Media.

[15] Hanneke, S. and Yang, L. (2015). Minimax analysis of active learning. *Journal of Machine Learning Research*, 16(12):3487–3602.

[16] Hoang, T. N., Low, B. K. H., Jaillet, P., and Kankanhalli, M. (2014). Nonmyopic ϵ-bayes-optimal active learning of gaussian processes. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 739–747.

[17] Lakshminarayanan, B., Roy, D. M., and Teh, Y. W. (2014). Mondrian forests: Efficient online random forests. In *Advances in neural information processing systems*, pages 3140–3148.

[18] Liu, H., Ong, Y.-S., and Cai, J. (2017). A survey of adaptive sampling for global metamodeling in support of simulation-based complex engineering design. *Structural and Multidisciplinary Optimization*, pages 1–24.

[19] Mourtada, J., Gaïffas, S., and Scornet, E. (2017). Universal consistency and minimax rates for online mondrian forests. In *Advances in Neural Information Processing Systems*, pages 3761–3770.

[20] Mourtada, J., Gaïffas, S., and Scornet, E. (2018). Minimax optimal rates for mondrian trees and forests. *arXiv preprint arXiv:1803.05784*.

[21] Resnick, S. I. (2013). *A probability path*. Springer Science & Business Media.

[22] Sabato, S. and Munos, R. (2014). Active regression by stratification. In *Advances in Neural Information Processing Systems*, pages 469–477.

[23] Sen, A. (2012). On the interrelation between the sample mean and the sample variance. *The American Statistician*, 66(2):112–117.

[24] Sourati, J., Akcakaya, M., Leen, T. K., Erdogmus, D., and Dy, J. G. (2017). Asymptotic analysis of objectives based on fisher information in active learning. *Journal of Machine Learning Research*, 18(34):1–41.

[25] Willett, R., Nowak, R., and Castro, R. M. (2006). Faster rates in regression via active learning. In *Advances in Neural Information Processing Systems*, pages 179–186.

8 Appendix

8.1 Proof of Proposition 4.1

This results is nothing more than the fact that a random subsample of size $n < m$ from an initial sample of sizer m has the same distribution as a sample of size n from that original distribution. The only issue here is if $q_k = 0$, in which case $p'_X(x) = 0 \; \forall \; x \in I_k$, where as $p_X(x)$ may be non-zero on a set of positive measure.

8.2 Proof of Corollary 4.2

We start by confirming that $E_{p'_{X,Y}}[\hat{\beta}_k] = \tilde{\beta}_k$. Let us fix $\mathcal{I}, k$ with n labelled points and let $n_k = \sum_{i=1}^{n} \mathbf{1}(X_i \in I_k)$. By assumption $n_k > 0$ otherwise $\hat{\beta}_k = \frac{1}{\sum \mathbf{1}(X_i \in I_k)} \sum_{X_i \in I_k} Y_i$ is undefined. Since Algorithm 1 is not active we have that $Y|X \in I_k \perp n_k$.

$$
\begin{aligned}
E_{p'_{X,Y}}[\hat{\beta}_k] &= E_{n_k} E_{p'_{X,Y}}\big[\frac{1}{\sum \mathbf{1}(X_i \in I_k)} \sum_{i=1}^{n} Y_i \mathbf{1}(X_i \in I_k) | n_k\big] \\
&= E_{n_k} \frac{1}{n_k} \sum_{i=1}^{n} E_{p'_{X,Y}}\big[Y_i \mathbf{1}(X_i \in I_k) | n_k\big] \\
&= E_{n_k} \frac{1}{n_k} \sum_{i=1}^{n} P(X_i \in I_k | n_k) E_{p'_{X,Y}}\big[Y_i | n_k, X_i \in I_k\big] \\
&= E_{n_k} \frac{1}{n_k} E_{p_{X,Y}}\big[Y | X \in I_k\big] \sum_{i=1}^{n} P(X_i \in I_k | n_k) \\
&= E_{n_k} E_{p_{X,Y}}\big[Y | X \in I_k\big] = E_{p_{X,Y}}\big[Y | X \in I_k\big].
\end{aligned}
$$

Now we use this to derive the decomposition in the standard way.

$$
\begin{aligned}
E\Big[(\hat{f}_{\mathcal{I}}(X) - f(x))^2\Big] &= E\Big[(\hat{f}_{\mathcal{I}}(X) - \tilde{f}_{\mathcal{I}}(X))^2\Big] + E\Big[(\tilde{f}_{\mathcal{I}}(X) - f(X))^2\Big] \\
&\quad + 2E\Big[(\hat{f}_{\mathcal{I}}(X) - \tilde{f}_{\mathcal{I}}(X))(\tilde{f}_{\mathcal{I}}(X) - f(X))\Big].
\end{aligned}
$$

$$
\begin{aligned}
&E\Big[(\hat{f}_{\mathcal{I}}(X) - \tilde{f}_{\mathcal{I}}(X))(\tilde{f}_{\mathcal{I}}(X) - f(X))\Big] = \\
&\quad E[\hat{f}_{\mathcal{I}}(X)]\tilde{f}_{\mathcal{I}}(X) - E[\hat{f}_{\mathcal{I}}(X)]f(X) - \tilde{f}_{\mathcal{I}}(X)^2 + \tilde{f}_{\mathcal{I}}(X)f(X) = 0.
\end{aligned}
$$

8.3 Proof of Lemma 4.3

We fix n_k. Given $X \in I_k$ we know that $\hat{f}_{\mathcal{I}}(X) = \hat{\beta}_k$ and $\tilde{f}_{\mathcal{I}}(X) = \tilde{\beta}_k$. Let us reorder the data $D_{1:n}$ so that the first n_k are in the leaf k for ease of notation. Then use Proposition 4.1, where the cross term disappears since $\epsilon_i \perp X_i$ under $p_{X,Y}$ by assumption.

$$
\begin{aligned}
&E_{p'_{X,Y}}\Big[(\hat{f}_{\mathcal{I}}(X)-\tilde{f}_{\mathcal{I}}(X))^2|X\in I_k\Big]= \\
&\frac{1}{n_k^2}\Big(\sum_{i=1}^{n_k}E_{p'_{X,Y}}\Big[(f(X_i)-\tilde{\beta}_k)^2|X_i\in I_k\Big]+\sum_{i=1}^{n_k}E_{p'_{X,Y}}\Big[(\sigma(X_i)\epsilon_i)^2|X_i\in I_k\Big] \\
&+2\sum_{i=1}^{n_k}E_{p'_{X,Y}}\Big[(f(X_i)-\tilde{\beta}_k)\sigma(X_i)\epsilon_i|X_i\in I_k\Big]\Big) \\
&=\frac{1}{n_k^2}\Big(\sum_{i=1}^{n_k}E_{p_{X,Y}}\Big[(f(X_i)-\tilde{\beta}_k)^2|X_i\in I_k\Big]+\sum_{i=1}^{n_k}E_{p_{X,Y}}\Big[(\sigma(X_i)\epsilon_i)^2|X_i\in I_k\Big] \\
&=\frac{1}{n_k}\big(bias_k^2+\sigma_{\epsilon,k}^2\big).
\end{aligned}
$$

8.4 Proof of Corollary 4.6

The proof involves looking at the expected risk under a random version of Algorithm 1. Formally allow Algorithm 1 to generate the q_i in a randomized fashion (with the randomness independent from all other sources of randomness), potentially using the other inputs to Algorithm 1 $(\mathcal{I},\{X_i\}_{i=1}^m,n,p_{X,Y})$ as parameters. Thus $(q_1,...,q_K)$ are drawn from a distribution, which in turn for all $(n_1,...,n_K)\in\mathbb{N}^K\ s.t.\ \sum n_k=n$ generates $P(n_1,...,n_K)$ the probability of the algorithm sampling $(n_1,...,n_K)$ points from each of the tree leaves. Let $Risk(n_1,...,n_K)$ denote the risk when our by leaf samples sizes are $n_1,...,n_k$, with $RiskBias$ and $RiskVar(n_1,...,n_K)$ being the bias and variance terms of the decomposition. The $RiskBias$ does not depend on $n_1,...,n_K$ since the risk bias term does not depend on how we sample. Then the risk of the randomized version of Algorithm 1 is

$$
\begin{aligned}
Risk&=\sum_{(n_1,...,n_K)}P(n_1,...,n_K)Risk(n_1,...,n_K) \\
&=RiskBias+\sum_{(n_1,...,n_K)}P(n_1,...,n_K)RiskVar(n_1,...,n_K).
\end{aligned}
$$

If $n_1^*,...,n_K^*$ is our optimal solution then by Theorem 4.4 $RiskVar(n_1^*,...,n_K^*)\leq RiskVar(n_1,...,n_K)\ \forall\ (n_1,...,n_K)\in\mathbb{N}^K\ s.t.\ \sum n_k=n$. For random sampling, unless $P(n_1^*,...,n_K^*)=1$ the Risk will clearly be greater than (or equal to) that of the optimal since the probability weighted average is greater than (or equal to) the min term of the sum.

8.5 Proof of Lemma 4.7

This is all algebra. By Equation 1

$$
\begin{aligned}
E\Big[(\hat{f}_{\mathcal{I}}(X)-\tilde{f}_{\mathcal{I}}(X))^2\Big]&=\frac{1}{n}\sum_{k=1}^K\sqrt{a_k}\sqrt{p_k\sigma_{Y,k}^2}\times\sum_{l=1}^K\frac{1}{\sqrt{a_l}}\sqrt{p_l\sigma_{Y,l}^2} \\
&=\frac{1}{n}\Big(\sum_k p_k\sigma_{Y,k}^2+\sum_{k\neq l}\frac{\sqrt{a_k}}{\sqrt{a_l}}\sqrt{p_kp_l\sigma_{Y,k}^2\sigma_{Y,l}^2}\Big) \\
&=\frac{1}{n}\Big(\sum_k p_k\sigma_{Y,k}^2+\sum_{k<l}\Big(\frac{\sqrt{a_k}}{\sqrt{a_l}}+\frac{\sqrt{a_k}}{\sqrt{a_l}}\Big)\sqrt{p_kp_l\sigma_{Y,k}^2\sigma_{Y,l}^2}\Big) \\
&=\frac{1}{n}\Big(\sum_k\sqrt{p_k\sigma_{Y,k}^2}\Big)^2+\frac{1}{n}\sum_{k<l}\Big(\frac{\sqrt{a_k}}{\sqrt{a_l}}+\frac{\sqrt{a_l}}{\sqrt{a_k}}-2\Big)\sqrt{p_kp_l\sigma_{Y,k}^2\sigma_{Y,l}^2} \\
&=\text{OPT + ERROR}.
\end{aligned}
$$

8.6 Proof of Corollary 4.8

Again, this is just algebra.

$$\frac{1}{n}\sum_{k<l}(\frac{\sqrt{p_k\sigma^2_{Y,l}}}{\sqrt{p_l\sigma^2_{Y,k}}}+\frac{\sqrt{p_l\sigma^2_{Y,k}}}{\sqrt{p_k\sigma^2_{Y,l}}}-2)\sqrt{p_kp_l\sigma^2_{Y,k}\sigma^2_{Y,l}}=\frac{1}{n}\sum_{k<l}(\sqrt{p_k\sigma^2_{Y,l}}-\sqrt{p_l\sigma^2_{Y,k}})^2$$
$$\leq\frac{1}{n}\sum_{k<l}(2p_k\sigma^2_{Y,l}+2p_l\sigma^2_{Y,k})\leq\frac{1}{n}\max_k\sigma^2_{Y,k}\sum_{k\neq l}(p_k+p_l)\leq\frac{K}{n}\max_k\sigma^2_{Y,k}.$$

8.7 Proof of Theorem 5.1

By the assumption that our sequence of trees $\mathcal{I}_{(n)}$ and Stage 1 sampling algorithm is strongly consistent for estimating the conditional variance $E[(Y-f(X))^2|X=x]$ as some statistic $b_n\to B$ we have that $\hat{\sigma}^2_{1,k}\to\sigma^2_k$ a.s. as $b_n\to B$. To see this let $\hat{\sigma}^2_{1,k}(x)=\hat{\sigma}^2_{1,k}$ for $x\in I_k$, $\sigma^2_k(x)=\sigma^2_k$ for $x\in I_k$ and let $\sigma^2(x)=E[(Y-f(X))^2|X=x]$. Then $|\hat{\sigma}^2_{1,k}(x)-\sigma^2_k(x)|\leq|\hat{\sigma}^2_{1,k}(x)-\sigma^2(x)|+|\sigma^2_k(x)-\sigma^2(x)|\to 0$, where the first term disappears due to the strong consistency, and the second term disappears due to the size of the partitions shrinking.

If $\hat{\sigma}^2_{1,k}\to\sigma^2_k$ a.s. then $\sum_{k=1}^{K_n}\sqrt{p_k\hat{\sigma}^2_{1,k}}\to\sum_{k=1}^{K_n}\sqrt{p_k\sigma^2_k}$ a.s. as $b_n\to B$. So if $b_n\to B$ a.s. then $\hat{n}_k\to n^*_k$ almost surely.

Now assume $b_n\to B$ in probability as $n\to\infty$ and want to show that these implies $\sum_{k=1}^{K_n}\sqrt{p_k\hat{\sigma}^2_{1,k}}\to\sum_{k=1}^{K_n}\sqrt{p_k\sigma^2_k}$ in probability $n\to\infty$. We will use Lemma 6.3.1.b from [21] which states:

Lemma (6.3.1.b in [21]). *$X_n\to X$ in probability iff for each subsequence $\{X_{n_k}\}, n_k\to\infty$ there exists a further subsubsequence $\{X_{n_{k_t}}\}, n_{k_t}\to\infty$ which converges a.s. to X.*

(The n_k here are unrelated to the n_k in our trees).

Let $Y_n=|\sum_{k=1}^{K_n}\sqrt{p_k\hat{\sigma}^2_{1,k}}-\sum_{k=1}^{K_n}\sqrt{p_k\sigma^2_k}|$, so $Y_n\to 0$ a.s. if $b_n\to B$. Thus we have a subset of the overall probability space Ω which is

$$\Omega\supset\Omega^*=\{\omega\in\Omega:\lim b_n(\omega)\neq B\text{ or }Y_n(\omega)\to 0\}$$

where $P(\Omega^*)=1$. Now take a subsequence $n_k\to\infty$ of n. By $b_n\to B$ in probability $\exists n_{k_t}\to\infty$ such that $b_{n_{k_t}}\to B$ a.s. as $n_{k_t}\to\infty$. This gives us a second subset of Ω

$$\Omega\supset\Omega'=\{\omega\in\Omega:b_{n_{k_t}}(\omega)\to B\}$$

where again $P(\Omega')=1$. On the intersection of these we get

$$\Omega^*\cap\Omega'\subset\{\omega\in\Omega:Y_{n_{k_t}(\omega)}(\omega)\to 0\}$$

where $P(\Omega^*\cap\Omega')=1$. n_k was an arbitrary subsequence of n and so by using Lemma 6.3.1.b in the reverse direction we get that $Y_n\to 0$ in probability.

8.8 Proof of Corollary 5.2

Here our $b_n=\frac{K_n}{n^{\frac{d+1}{d+2}}}$ and $B=0$. Since $E[K_n]=(1+n^{\frac{1}{d+2}})^d$ by Markov $\frac{K_n}{n^{\frac{d+1}{d+2}}}\to 0$ in probability.

Now we need to show that if we assume $\frac{K_n}{n^{\frac{d+1}{d+2}}} \to 0$ we get strong consistency of our conditional variance function estimation. By Theorem 23.3 in [14] we get that our tree is strongly consistent for estimating the mean function, since $\frac{K_n \log(n)}{n} \to 0$ so eventually every partition will have more than $\log(n)$ samples in the leaf, and the augmented estimator in Theorem 23.3 is the same as the usual estimator. (The augmented estimator in Theorem 23.3 is the usual decision tree estimator if there are more than $\log(n)$ data points in the partition and 0 otherwise). Finally we need the p_X bounded since Theorem 23.3 assumes that our test X density is the same as our training one, but since p_X is bounded the Radon Nikodym derivative is bounded and so we get strong consistency even with the different test density.

So our tree and Stage 1 sampling scheme are strongly consistent for estimating the mean function $f(x) = E[Y|X = x]$. Now assume we had access to a new set of random variables $Z_i = (Y_i = f(X_i))^2$. Because of the bounded kurtosis our tree would also be strongly consistent for estimating the mean function of the Z_i which we will call $f_Z(x) = E[(Y - f(X))^2|X = x]$. So if we had access to the Z_i we could use them to estimate our Y conditional variance function using $\hat{f}_Z(x) = \frac{\sum Z_i \mathbf{1}_{X_i \in I(x)}}{\sum \mathbf{1}_{X_i \in I(x)}}$.

We don't have these Z_i but we do have $\tilde{Z}_i = (Y_i - \hat{f}(X_i))^2$, and it's easy to show that $\frac{\sum \tilde{Z}_i \mathbf{1}_{X_i \in I(x)}}{\sum \mathbf{1}_{X_i \in I(x)} - 1} \to \frac{\sum Z_i \mathbf{1}_{X_i \in I(x)}}{\sum \mathbf{1}_{X_i \in I(x)}}$ by adding and subtracting $f(x)$ inside the square. This gives us a strongly consistent estimator of our conditional variance as required.

8.9 Proof of Lemma 5.3

Since the Stage 1 sampling uses Algorithm 1 our $n_{(1),k}$ are fixed (though this could be extended to randomized version of Algorithm 1). The proof is mostly algebra, using the fact that $\hat{\beta}_{(1),k}$ is conditionally independent of $\hat{\beta}_{(2),k}$ given $n_{(2),k}$.

$$
\begin{aligned}
E[(\hat{\beta}_k - \tilde{\beta}_k)^2] &= E\Big[\big(\frac{n_{(1)}(\hat{\beta}_{(1),k} - \tilde{\beta}_k)}{n_{(1),k} + n_{(2),k}} + \frac{n_{(2)}(\hat{\beta}_{(2),k} - \tilde{\beta}_k)}{n_{(1),k} + n_{(2),k}}\big)^2\Big] \\
&= E_{n_{(2),k}}\Big[\frac{n_{(1),k}^2}{(n_{(1),k} + n_{(2),k})^2} E_{D_{1:n_{(1)}}}\big((\hat{\beta}_{(1),k} - \tilde{\beta}_k)^2 | n_{(2),k}\big) \\
&+ 2\frac{n_{(1),k} n_{(2),k}}{(n_{(1),k} + n_{(2),k})^2} E_{D_{1:n_{(1)}}}\big((\hat{\beta}_{(1),k} - \tilde{\beta}_k)|n_{(2),k}\big) E_{D_{n_{(1)}+1:n}}\big((\hat{\beta}_{(2),k} - \tilde{\beta}_k)|n_{(2),k}\big) \\
&+ \frac{n_{(2),k}^2}{(n_{(1),k} + n_{(2),k})^2} E_{D_{n_{(1)}+1:n}}\big((\hat{\beta}_{(2),k} - \tilde{\beta}_k)^2 | n_{(2),k}\big)\Big].
\end{aligned}
$$

We have that

$$
E_{D_{n_{(1)}+1:n}}\big((\hat{\beta}_{(2),k} - \tilde{\beta}_k)|n_{(2),k}\big) = 0,\ E_{D_{n_{(1)}+1:n}}\big((\hat{\beta}_{(2),k} - \tilde{\beta}_k)^2|n_{(2),k}\big) = \frac{\sigma_{Y,k}^2}{n_{(2),k}}
$$

which gives us the desired result.

8.10 Proof of Theorem 5.4

By assumption we have that Y_i's are Normally distributed. We first deal with the dependence $E_{D_{1:n_1}}\big((\hat{\beta}_{(1),k} - \tilde{\beta}_{(1),k})^2|n_{(2),k}\big)$. A well known property of the Normal distribution [23] is that the estimate of the mean $\hat{\beta}_{(1),k}$ and the estimate of the variance $\hat{\sigma}_{Y,k}^2$ are independent. This immediately gives that $E_{D_{1:n_1}}\big((\hat{\beta}_{(1),k} - \tilde{\beta}_{(1),k})^2|n_{(2),k}\big) = E_{D_{1:n_1}}\big((\hat{\beta}_{(1),k} - \tilde{\beta}_{(1),k})^2\big) = \frac{\sigma_{Y,k}^2}{n_{(1),k}}$ as there is no dependence between $\hat{\beta}_{(1),k}$ and $n_{(2),k}$. Thus we get that the risk variance for that leaf is just as from Lemma 4.3.

Now we want to bound the probability $\hat{n}_k$ above is far away from n_k^*. We will do this by bounding the a_k. Another well known property of the normal distribution is $\frac{(n_k-1)S_{Y,k}^2}{\sigma_{Y,k}^2} = (n_k-1)a_k \sim \chi^2_{(n_k-1)}$. By characterization of sub-exponential random variables:

$$P((n_k-1)|a_k-(n-1)| > \sqrt{2(n_k-1)t}+2t) \leq e^{-t}$$

$$P(|a_k-1| > \frac{\sqrt{2t}}{\sqrt{(n_k-1)}} + \frac{2t}{(n_k-1)}) \leq e^{-t}$$

$$\frac{\sqrt{2t}}{\sqrt{(n_k-1)}} + \frac{2t}{(n_k-1)} \in (0,1) \implies \frac{2t}{(n_k-1)} \leq 1 \implies \frac{2t}{(n_k-1)} < \frac{\sqrt{2t}}{\sqrt{(n_k-1)}}$$

$$\implies P(|a_k-1| > \frac{2\sqrt{2t}}{\sqrt{(n_k-1)}}) \leq P(|a_k-1| > \frac{\sqrt{2t}}{\sqrt{(n_k-1)}} + \frac{2t}{(n_k-1)}) \leq e^{-t}$$

$$\forall\, \alpha \in (0,1)$$

$$P(|a_k-1| > \alpha) \leq e^{-\frac{(n_k-1)\alpha^2}{8}}$$

$$P(\exists k \text{ s.t. } |a_k-1| > \alpha) \leq \sum_{k=1}^{K} e^{-\frac{(n_k-1)\alpha^2}{8}}.$$

And now we apply Lemma 4.7 to bound the excess. Now we assume our purely random tree is a Mondrian Tree with the above assumptions, so $n_k = \frac{cn}{K}$. By Markov inequality and Proposition 2 in [20] we have that:

$$P(K_n - 1 > n^{\frac{d+\epsilon}{d+2}}) \leq \frac{E[K_n]}{n^{\frac{d+\epsilon}{d+2}}} = \frac{(1+n^{\frac{1}{2+d}})^d}{n^{\frac{d+\epsilon}{d+2}}} = \delta_1$$

$$P(\exists k \text{ s.t. } |a_k-1| > \alpha | K_n \leq n^{\frac{d+\epsilon}{d+2}}) \leq n^{\frac{d+\epsilon}{d+2}} e^{-\frac{\alpha^2}{8}((cn)^{\frac{2-\epsilon}{d+2}}-1)} = \delta_2$$

By setting $\epsilon = 1$ and using the union bound we get the result.

Remark. It is worth noting that in the above proof we have only used the property that χ^2 are subexponetial. A slightly stronger (in terms of n, α) inequality is possible using Chernoff Bounds and exploiting the structure of χ^2 random variables.

8.11 Dependence in non-normal case

We are interested in the question of when is $E_{D_{1:n_{(1)}}}\left[(\hat{\beta}_{(1),k} - \tilde{\beta}_k)^2 | n_{(2),k} < n^*_{(2),k}\right] < E_{D_{1:n_{(1)}}}\left[(\hat{\beta}_{(1),k} - \tilde{\beta}_k)^2\right]$. Unfortunately $n_{(2),k}$ is a function not only of $\hat{\sigma}^2_{(1),k}$ but of all other $\hat{\sigma}^2_{(1),l}$. Let us start with a more simple and general question of when $E\left[(\hat{\mu}-\mu)^2 | \hat{\sigma}^2 < \sigma^2\right] < E\left[(\hat{\mu}-\mu)^2\right]$. We present no formal arguments here but rather share our findings and conjectures which we consider both interesting in their own right as well as excellent candidates for further study. The first observation is that far from this being an unusual property this seems to be a fairly common property. In fact for symmetric distributions the relationship appears to be well behaved. From [23] the sample mean and sample variance are asymptotically MVN (multivariate normal) with cross correlation equal to the skew, so when our distribution is symmetric the sample mean and sample variance are independent in the limit. For the finite sample case the relationship between $\hat{\sigma}^2_{1,k} - \sigma^2_{1,k}$ and $E\left[(\hat{\mu}-\mu)^2 | \hat{\sigma}^2 - \sigma^2\right] - E\left[(\hat{\mu}-\mu)^2\right]$ appear to be monotonic and to go through the origin (so when the sample variance is the true variance, the conditional variance of the sample mean is the unconditioned variance, which is what we would hope is the case). In fact it appears both the magnitude and parity of this relationship depends on the *excess kurtosis* $\kappa - 3$. If $\kappa - 3 < 0$ this relationship is negative and if $\kappa - 3 > 0$ this relationship is positive, with the magnitude increasing as you move further away from zero.

If these observations are true for all symmetric distributions it would be quite fortuitous, since large values of κ imply that the estimates of our variances will be more noisy, but those are exactly

the cases where actively fitting to the sample variance of our first stage is beneficial: If our sample variance is larger than the population variance, then the variance of our $\hat{\beta}_{(1),k}$ is larger than expected, so it is beneficial to use more points in the second stage than the optimal passive sampling would have assigned. Meanwhile when a smaller sample variance implies the variance of our $\hat{\beta}_{(1),k}$ is larger than expected, κ is small and so our sample variance will itself have small variance. We have not yet been able to prove this relationship, and things become much more complicated in the more realistic case where our distribution is skewed. However these results give us confidence that things are unlikely to go too badly wrong when our labels are not normally distributed.

8.12 Experimental data set info

For both simulations our marginal X distribution was uniform over the space $[0,1]^{10}$. Heteroskedastic simulation had constant regression function and Gaussian noise, with space split into high variance region (25) and low variance region (1). Varying complexity had sinusodial regression function $f(x) = C\sin(\frac{2\pi}{d*F} * \sum x_i)$ and Gaussian noise with constant (1) variance. It was split into high variation region ($C = 20, F = 0.05$) and low variation region ($C = 5, F = 0.1$). For both sets $[0.1,1]^{10}$ were the high areas, with everything else a low area.

8.13 Practitioners guide

Here we compile information related to actually using this active learning method in practice.

8.13.1 Heuristics to deal with difference between theory n_k^* and possible values

There are many reason why you may not actually be able to sample according to your estimates of the optimal n_k^*. For a start our n_k^* will almost always be fractional. Additionally there may be less than n_k^* points in a leaf. These issues are fairly minor and become less influential as sample sizes increase. However a more consistent issue that occurs when using the approximating algorithm is when a leaf is oversampled during stage 1, so that $n_{(1),k} > n_k^*$. This means that some other leaf will get fewer than it is optimal number of samples. Although this again can be dealt with asymptotically by making our stage 1 a small fraction of the total number of samples, in practice this is a problem which often occurs when our sample size is not large.

In our code we implemented heuristics to deal with these mismatches. We emphasize that these heuristics are subjective and one could easily use or argue for others. After calculating our $\hat{n}_k$ we immediately floor them all. We then set $\hat{n}_k = \max(\min(\hat{n}_k, \eta_k), n_{1,k})$ (where η_k is the total number of points in leaf k). It is possible that $\sum \hat{n}_k \neq n$ after these adjustments. If we have too many points, we reduce the largest $\hat{n}_k$ until we achieve the correct total. If we have too few points we increase the $\hat{n}_k$ by 1 each, starting with the smallest, and starting over once we have increased them all by 1. This asymmetry is because increasing small values can have a large reduction on the variance of the estimate, but decreasing large ones leads to a small increase in variance.

8.13.2 Lifetime parameter sequence

We have found that the best general form for the lifetime parameter sequence is $\lambda_n = \frac{1}{\gamma}(n^{\frac{2}{2+d}} - 1)$. The γ can be fairly freely chosen with $\gamma = 1$ a reasonable default (and is what is used in all simulations and experiments in this paper), but the -1 is very important; it ensures that we do not start with a lifetime $= 1$ for $n = 1$, $\forall\, d$ as when d is large this can result in a very large number of leaves early on.

8.13.3 Sampling method during stage 1

During stage one our theory assumed that $n_1 = cn$ and then each leaf received the same fraction of points, as this gives important asymptotic properties. In practice if c is too large this can result in putting too many samples in certain small leaves during stage 1, so that $n_{1,k} > n_k^*$, meaning that we have oversampled this leaf and will have to reduce other sampling elsewhere. One way of avoiding this is by making c small, but this risks getting bad leaf estimates and suboptimal stage 2 sampling unless n is large, where the n required increases as d increases. Another is to sample passively. We have found that generally if $c = 0.5$ then sampling passively tends to produce pretty good results

unless your function has massive amounts of variation. Another option is to use a hybrid sampling scheme in stage 1, where each leaf is given a small number of samples, and then the rest of the samples are distributed randomly, but empirically this seems to be worse than random sampling for small values of n.

8.13.4 Final regression model

As shown in our experiments, although most the theory assumes that you are using the same tree for your active learning as you are for your final predictions, you also get good results doing active learning with Mondrian Forests, and then taking that data and fitting your final model with a more data adaptive model, although not always.

8.13.5 Forests

Just as with Breiman decision trees you can ensemble purely random trees into forests. These forests show improved performance at the cost of increased computational cost since they average out the random process used to build the trees. We also have an intuitive (though theory free) extension of our active learning method to utilize the power of multiple Mondrian Trees. The idea is each tree determines the optimal number of samples per leaf in the usual way, and then gives data points weights such that the expected number of points sampled from each leaf is the optimal number. These probabilities are then averaged out over all the trees in the forest and the new points are sampled using these averaged probabilities. The formal algorithm is given below:

Algorithm 3: Forest version of oracle approximation algorithm

Input: Leaves of our T trees $\mathcal{I}_1 ... \mathcal{I}_T$, pool of data points $\{X_i\}_{i=1}^m$, and label budgets $n_{(1)}, n_{(2)}, n = n_{(1)} + n_{(2)}$.
Output: The set of labelled points.
Stage 1: ;
Sample $n_{(1)}$ data points (possibly according to the structure of the trees $\mathcal{I}_t$) using a version of algorithm 1. ;
foreach *t* **do**
  Use those samples (X_i, Y_i) to estimate $\hat{\sigma}^2_{Y,k,t}$ for each leaf. ;
end
Stage 2: ;
foreach *t* **do**
  foreach $I_{k,t} \in \mathcal{I}_t$ **do**
    Calculate $\hat{n}_{k,t} = n \frac{\sqrt{p_{k,t} \hat{\sigma}^2_{Y,k,t}}}{\sum_{k'} \sqrt{p_{k',t} \hat{\sigma}^2_{Y,k',t}}}$ the number of points in the leaf to sample. ;
    Count $m_{k,t}$ the number of unlabelled points in leaf $I_{k,t}$;
    foreach *Unlabelled* $X_i \in I_{k,t}$ **do**
      Assign weight $W_{i,t} = \frac{\hat{n}_{k,t} - n_{(1),k,t}}{n_2 * m_{k,t}}$. ;
    end
  end
end
foreach *Unlabelled* X_i **do**
  Final weight $W_i = \frac{1}{T} \sum W_{i,t}$. ;
end
Sample $n_{(2)}$ points with weights W_i.

Below we show the results of using Mondrian Forests for our active learning, and both Mondrian Forests and Random Forests as our final regression model. Here we see some benefit using Mondrian Forest for active learning and then Random Forests for our final regressor (although in fact the naive uncertainty sampling method outperforms ours). Although the benefit on the real data appears to be a small constant factor, the actively learned models provide similar accuracy with 10s of fewer data points, which can be significant.

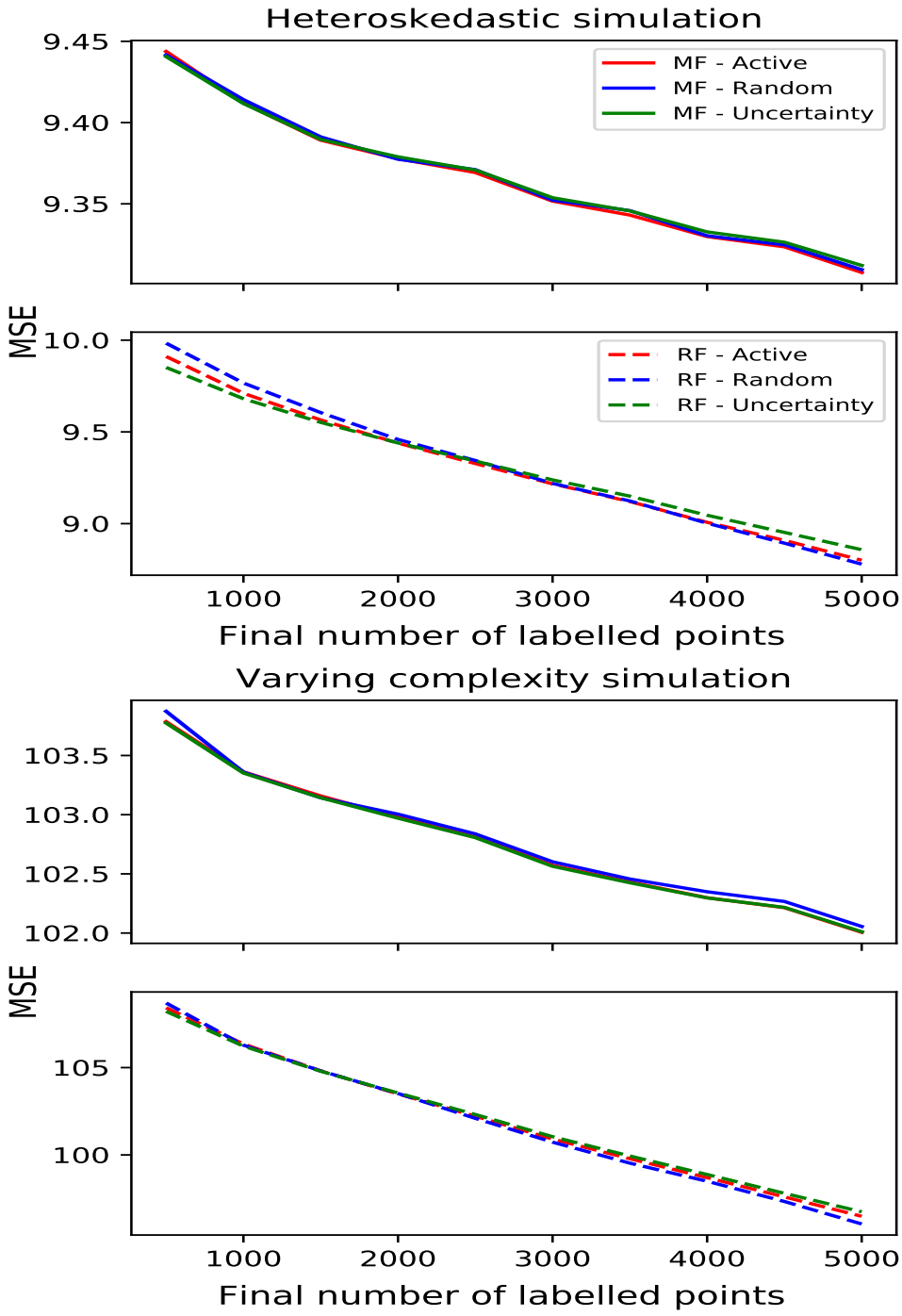

Figure 2: Mondrian Forest active learning simulations

8.13.6 Using more than 2 stages

It is of course possible to do more than 2 stages, updating your estimates of the leaf variances during each stage to guide sampling during the next stage. We found that in practice the benefits of doing this are generally fairly small. Of course the first stage should still be sufficiently large that you get decent initial estimates for the leaf variances. Much of the theory could be extended to increasing number of stages as long that the number is not increasing with n without much work. Increasing the number of stages as n increases may require additional care and effort.

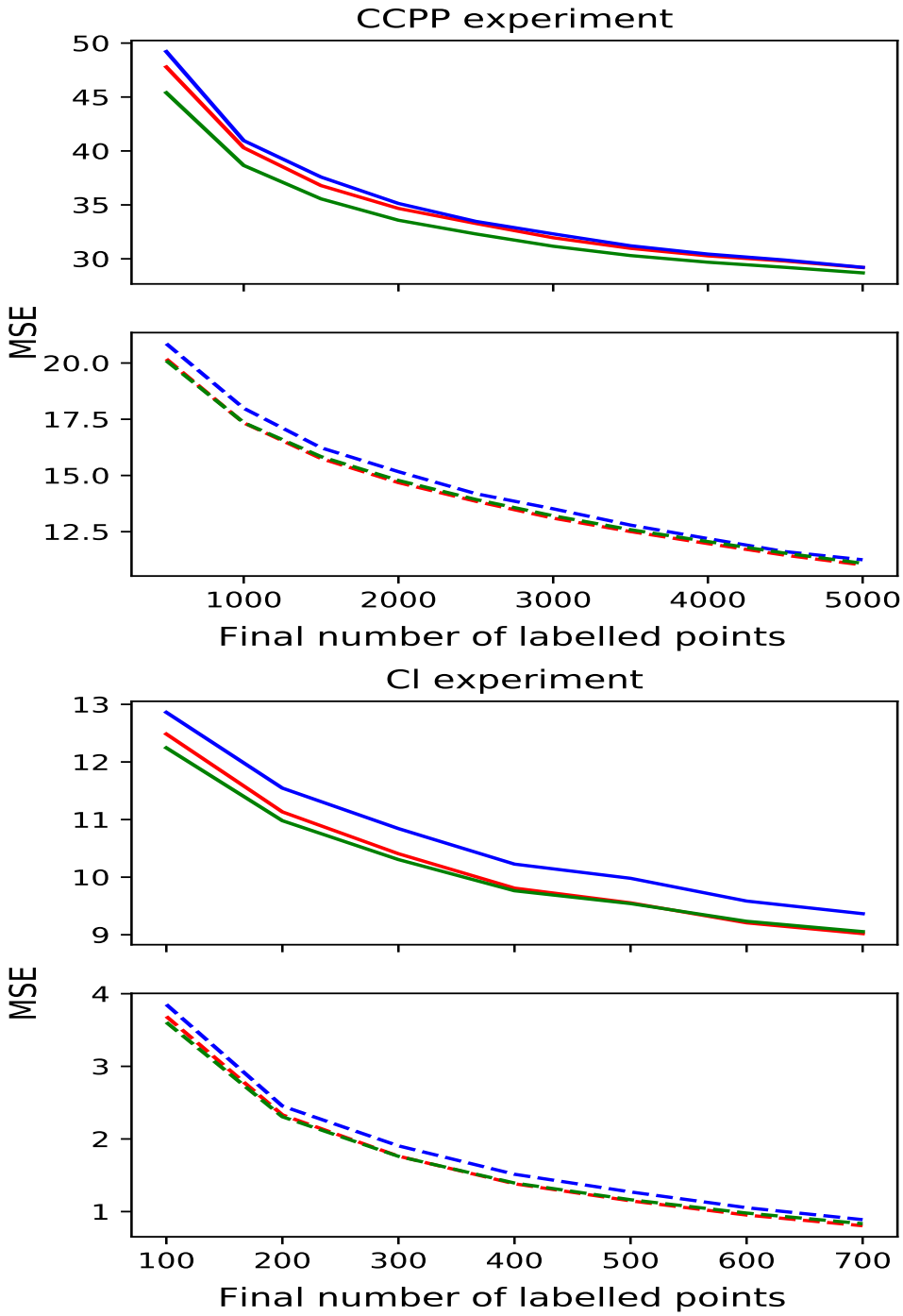

Figure 3: Mondrian Forest active learning experiments

8.13.7 Additional experimental results

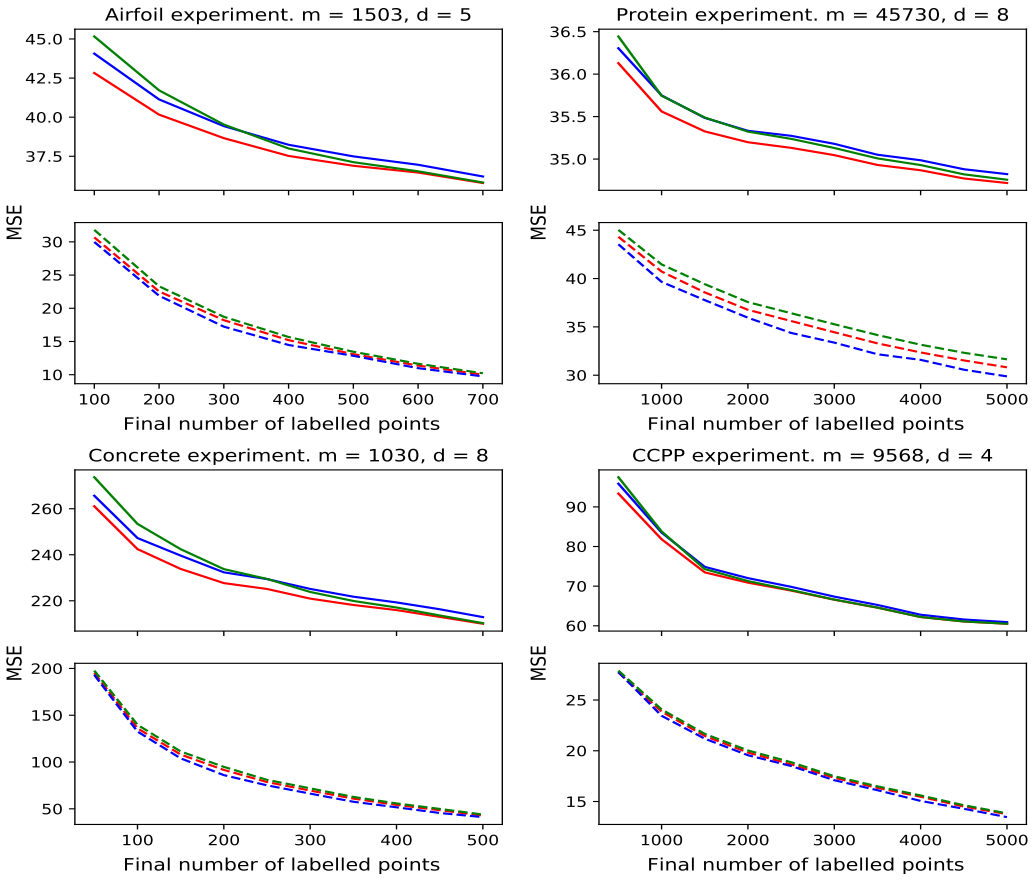

Figure 4: Additional active learning experiments on UCI data with Mondrian Trees