Regression tree-based active learning

Ashna Jose, João Paulo Almeida de Mendonça, Emilie Devijver, Noël Jakse, Valérie Monbet, Roberta Poloni

Regression Tree-based Active Learning

Ashna Jose, Joao Paulo Almeida de Mendonça, Emilie Devijver,
Noel Jakse, Valérie Monbet, Roberta Poloni

October 18, 2023

Abstract

Machine learning algorithms often require large training sets to perform well, but labeling such large amounts of data is not always feasible, as in many applications, substantial human effort and material cost is needed. Finding effective ways to reduce the size of training sets while maintaining the same performance is then crucial: one wants to choose the best sample of fixed size to be labeled among a given population, aiming at an accurate prediction of the response. This challenge has been studied in detail in classification, but not deeply enough in regression, which is known to be a more difficult task for active learning despite its need in practice. Few model-free active learning methods have been proposed that detect the new samples to be labeled using unlabeled data, but they lack the information of the conditional distribution between the response and the features. In this paper, we propose a standard regression tree-based active learning method for regression that improves significantly upon existing active learning approaches. It provides impressive results for small and large training sets and an appreciably low variance within several runs. We also exploit model-free approaches, and adapt them to our algorithm to utilize maximum information. Through experiments on numerous benchmark datasets, we demonstrate that our framework improves existing methods and is effective in learning a regression model from a very limited labeled dataset, reducing the sample size for a fixed level of performance, even with many features.

1 Introduction

In many applications, for example in physics and chemistry, machine learning algorithms are used to predict properties, typically labels, given available information (referred to as input features). Machine learning approaches usually give good results to prediction problems when large training data sets are available. However, when some properties of interest are expensive to compute, the labeling process required for a supervised machine learning approach becomes difficult. It is then necessary to limit the number of samples in the training set. For example, superconductors are materials that conduct current with zero resistance below the super-conducting critical temperature, T_c, thus allowing very efficient energy transport. Yet most superconductors exhibit a T_c well below room temperature, which limits their range of applicability (**?**). The development of novel materials with high T_c is an extremely active field of research, however, predicting T_c is still computationally challenging. For such problems, where samples are difficult to label, it is of utmost importance to build the training set that brings the most information about the relation between the features and the label, at a constant labeling cost.

Passive learning, that involves constructing the training set via Random Sampling (RS), is a common and simple approach. However, it does not exploit any knowledge of the available dataset. Thus, if the target sample size is small, there is a significant risk that samples selected by passive learning do not provide meaningful information. In such cases, active learning (AL) techniques are more appropriate: they add samples step by step taking advantage of the available knowledge of the structure of the dataset. For example, if at each step new samples have been labeled, they can use these labels to drive the sampling process. In the sequel, we distinguish between model-free AL, where only the features are used, and model-based AL, when the response is also used to construct the training set.

Many approaches have been proposed to construct optimal training sets for classification problems **???**, but efficient methods that significantly outperform RS are scarce for regression. Here optimality means that we select the sample (of given size) that leads to the lowest prediction error for the regression function learned on the sample. Moreover, most of the methods that exist for regression tasks focus on model-free AL i.e. using only the knowledge of the features to choose the samples to be labeled. Some promising results have been shown for methods that exploit diversity (**?**) and representativeness (**?**) in the input feature space. However, such approaches may not have enough diversity in the response space, thus resulting in potentially inefficient predictions. It is intuitive that adding some knowledge of the response should help in understanding the conditional distribution between the features and the response, resulting in the selection of better samples to be labeled in a prediction objective. Model-based AL schemes accomplish this task by defining a sampling criterion based on the knowledge of the samples already labeled. Good rates of convergence for the regression error have previously been established under strong modeling assumptions when the size of the labeled sample increases, but to our knowledge, no general, assumption-free method has been put forward for efficient model-based AL in regression yet.

In this paper, we propose a model-based AL approach for non-linear regression based on regression trees, with a focus on constructing an optimal training set of small size, especially suitable for practical applications where labeling data is expensive. Decision trees and their ensemble extensions form an excellent group of predictors as they are simple yet robust, fast, interpretable and easy to use. A regression tree splits the descriptor space into regions homogeneous for the response and the partitions are used to select the set of samples to be labeled. We demonstrate how a regression tree in the joint feature-response space can substantially detect interesting samples to enhance the performance of the learned regression model, as it simultaneously takes the response into account. Moreover, most of the existing model-based methods are based on a first sample that is drawn randomly, thus not initializing well, and we prove that smartly mixing the advantages of model-free and model-based methods by adapting model-free methods in our regression tree outperforms other methods. The main contributions of our work are summarised as follows:

- We propose a model-based AL approach for non-linear regression based on regression trees.
- We demonstrate, with experiments on various benchmark datasets, how a regression tree in the joint feature-response space can substantially detect interesting samples to enhance the performance of the learned regression model, as it simultaneously takes the response into account.
- We show that smartly mixing the advantages of model-free and model-based methods by adapting model-free methods in our regression tree outperforms other existing methods.

The above claims are supported via numerous experiments and analysis for several benchmark datasets in Section 4. We succeed to perform the best consistently over datasets of various sizes and dimensions and show that our model has the lowest variance, a factor very important in the low data regime.

The remainder of the paper is organized as follows. In Section 2, we describe the related work. We introduce our method in Section 3, and illustrate it in Section 4. We first evaluate our method on several benchmark data sets and show that it outperforms the state-of-the-art methods. We then focus on the superconductivity dataset to illustrate the performance of our method in detail. Finally, Section 5 concludes the paper.

2 Related Work

Active learning has been studied in several areas of research, that we introduce in detail in this section. Broadly speaking, AL methods can be categorised based on their query criteria into model-free and model-based methods (**?**). Model-free methods exploit only the feature space information, to construct the most informative training set, while model-based methods use response information through regression functions trained on previously labeled samples. This has been detailed as a flowchart in Fig. 1.

The model-free approaches lack the information of the response and one can hope to obtain a sample which allows a better description of the joint feature-response distribution. This is where model-based AL

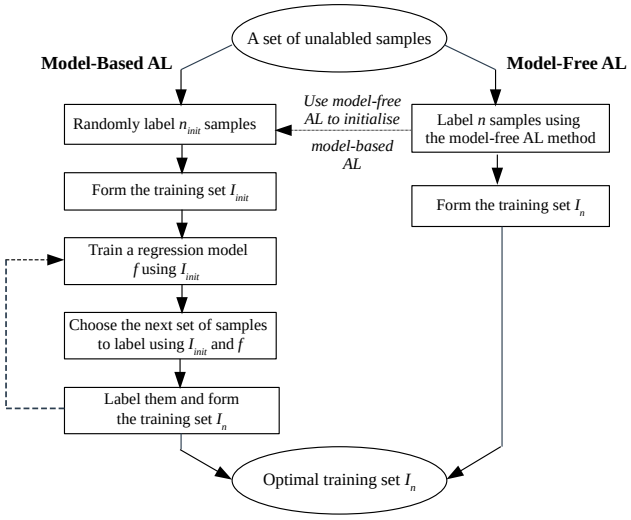

Figure 1: Illustration of model-free vs model-based AL methods in regression.

has an upper hand. Ref. (**?**) provides a theoretical understanding of when model-based AL achieves better rates of convergence than passive/model-free methods. Their result is rather pessimistic, concluding that most of the time, passive learning/model-free AL is the best that one can do. This is why an efficient model-based AL mechanism is still a challenge for regression tasks in practice. However, we argue that model-based AL methods are crucial for optimal sampling, especially when labeling is expensive, as these methods take into account the information of the response as well, and are thus more complete.

For sake of completeness and comparison, we list seminal works in both model free and model-based AL for regression below.

2.1 Model-free active learning

In the recent past, many theoretical studies have focused on linear models and on the necessary assumptions to improve rates of convergence of the learned prediction model when increasing the size of the labeled sample set (**???**). In the design of experiments field, optimal design (**?**) aims at constructing a training set by minimizing a statistical criterion related to the information matrix in a linear model, e.g. its trace (**?**) or its determinant (**?**). It has further been extended to generalized linear models (**?**) and also to more general parametric models (**?**).

Space filling approaches like Greedy Sampling over the feature space (GSx, (**?**), inspired by (**?**)) also come under model-free methods, as they target a training set diverse in features. It selects the sample closest to the centroid of the feature space as the first sample in the training set, followed by the one farthest from it, according to L_2 distance. The samples to be labeled consequently are the ones farthest from all the samples that have been previously selected to ensure diversity. However, such an approach samples well only if the feature space is uniformly distributed, which rarely occurs.

Iterative Representativeness Diversity Maximization (iRDM, (**?**)) successfully outperformed GSx by taking into account both diversity and representativeness of the feature space. It uses k-means clustering to partition the feature space into a number of clusters corresponding to the number of samples to be labeled. It then selects the samples closest to the centroids of these clusters as the starting points, and over the course of the algorithm, combines it with the basic idea of feature space diversity from GSx to update the centroids to the samples which are representative and diverse. Even though they achieve modest performances, the method relies on the assumption that the data can be well clustered, which is not always the case, especially

not for large number of clusters.

In the field of survey methodology, the cube method **?** has been used for balanced sampling from finite populations. It constructs a sample of fixed size with the same characteristics as those of the features of the full dataset, assuming that this sample will lead to a good approximation of the distribution of the response, or its conditional distribution, given the features.

In Ref. (**?**), the authors propose another model-free AL scheme where they aim at reducing an uncertainty measure at each iteration of the algorithm. More precisely, they argue that missing labels in the unlabeled data set leads to uncertainty in the prediction model and they select the points to be labeled as the ones which are most distant from the already labeled points and close to many points of the unlabeled set, according to the L_1 distance. In practice, their algorithm, referred to as graph-based in the sequel, may be represented as a graph with edges between labeled and unlabeled samples, and the uncertainty to be minimized is computed from the edges.

2.2 Model-based active learning

A typical model-based AL algorithm consists of the passive part (or model-free active part) that is used to label a few samples to construct an initial model, which is then used to pick the next samples (or set of samples in the batch-mode) to be labeled for improving the regression model. One of the pioneer works in this context came from Ref. (**?**), where a method to select samples is proposed by reducing the variance under weak parametric modeling assumptions (mixture of Gaussian regressions and locally weighted regressions). This theoretical study however is not competitive on real datasets.

Greedy Sampling over the response (GSy, (**?**)) follows the AL architecture by using GSx for the first part to get an initial regression function based on linear Ridge regression, followed by a prediction of the unlabeled samples. The next sample to be labeled is picked based on diversity of the response, similar to what is done for GSx. The L_2 distance computations are now performed in the response space, with labels of samples in the labeled set being their true labels, and those of samples in the unlabeled set being the predictions made by the previously trained model. This repeated update improves the regression model and was shown to perform better than passive learning and a few AL methods. However, their results hold true only for linear models, thus restricting their use for non-linear problems.

Variance-based Query By Committee (QBC) for regression (**?**) is another model-based AL strategy that selects the samples with the highest variance among the predictions from a committee of models. The committee is constructed by bootstrapping on an initial set of passively labeled samples. However, training many models in the committee makes the approach computationally complex. Moreover, the choice for the class of models is not always clear.

Expected Model Change Maximization (EMCM, (**?**)) uses a set of passively labeled samples to get an initial regression model, followed by construction of a set of regression models via bootstrapping. It then computes the expected change in these models with respect to the initial regressor, inspired by stochastic gradient descent update rules, and selects the sample that leads to the largest model change. They showed improvement over passive sampling and a few model-free AL methods. They also developed a batch version of their method, B-EMCM (**?**), which although is computationally cheaper, does not succeed to outperform EMCM. However, (**?**) have recently shown that their method iRDM works better on many datasets, even though it is model-free.

In (**?**), the author proposes a set of Representation and Diversity (RD) based extensions of a few AL methods, namely RD-QBC, RD-EMCM and RD-GS. After performing a k-means clustering on the dataset (with the number of clusters equal to the number of features of the dataset), it labels the samples closest to the centroid of these clusters. This is followed by clustering the dataset with one more cluster. The largest cluster without a labeled sample is identified, and a new sample is chosen to be labeled from that cluster, either the one closest to the centroid, or via QBC, EMCM or Greedy Sampling (GS). They showed that these algorithms give the same results statistically, which are not very far from Greedy sampling. Moreover, iRDM (**?**) tends to work better than these as well.

Another recent advancement in the field came from (**?**), in which AL was proposed via purely random Mondrian trees (**?**). Mondrian trees differ from classical regression trees in their branching and prediction:

the branching does not depend on the features and is random, contrary to regression trees that branch to minimize the impurity in the features; and their prediction is a weighted average over the previous branchings. After labeling an initial set of samples randomly, they construct a purely random Mondrian tree on [0,1]. They then theoretically derive the number of samples to be selected in each leaf of their random trees to achieve the best possible loss. Although they show modest improvement over random sampling, it is important to note that the purely random nature of the trees leads to a very high variance over repeated experiments. Moreover, they scale the entire dataset to [0,1] as the purely random Mondrian tree is constructed on [0,1], which may not be the ideal approach for all datasets.

Recently, there is a growing interest in new approaches to AL based on Bayesian models and Gaussian Processes (GP) on one hand and with deep learning techniques on the other. In Ref. (**?**), the authors propose two algorithms, a Bayesian version of QBC and a second one using a GP as a mixture of Gaussian models for AL and is inspired by Ref. (**?**). However, they focus on datasets of (very) low dimension. GP active learning has also been extended to ensemble GP AL (**?**), again considering datasets of low dimension only. Ref. (**?**) propose a novel Batch Model Deep Active Learning (BMDAL) method to improve the sampling-efficiency of neural network regression using network-dependent kernels and kernel transformations. Although they show a good performance, the method is broadly dedicated to large datasets and training a large number of samples, thus not beneficial for cases where labeling is expensive, which is what we focus on.

3 Method

In this section, we formally introduce the context of AL, and more specifically model-based AL. Then, we introduce our main method, named Regression tree-based active learning (RT-AL), and how it can be tuned by adapting ideas from model-free methods.

3.1 Model-based active learning

Let $\mathbf{X}$ be a D-dimensional random vector and let Y be a random variable linked to $\mathbf{X}$ through a regression model. In pool-based AL methods, a dataset of size N with all observations $\{\mathbf{x}_i\}_{i=1}^N$ unlabeled is given, and let $(y_i)_{i=1}^N$ denote the (unknown) associated labels.

Formally, if $\hat{f}_{I_n}$ denotes the estimator among a class $\mathcal{F}$ of prediction models learnt on a training set of size n indexed by $I_n \subset \{1, \dots, N\}$ with respect to the risk R,

$$\hat{f}_{I_n} = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{i \in I_n} R(\hat{f}_{I_n}(\mathbf{x}_i), y_i) \right\}, \tag{1}$$

we look for the set of observations indexed by $\mathcal{I}_n \subseteq \{1, \dots, N\}$ of size n such that the transductive risk is minimized:

$$\mathcal{I}_n = \underset{I_n \subseteq \{1,\dots,N\}}{\operatorname{argmin}} \left\{ \frac{1}{N-n} \sum_{i \notin I_n} R(\hat{f}_{I_n}(\mathbf{x}_i), y_i) \right\}. \tag{2}$$

However, in practice, one does not have access to the transductive risk because the labels are not observed.

Model-based AL methods focus on the knowledge of the joint distribution of $\mathbf{X}$ and Y to mimic the minimization problems given by Eqs. (1) and (2). To do so, a first set I_{init} of n_{init} samples is detected by a model-free method (either RS or something smarter). Then, $\hat{f}_{I_{\text{init}}}$ is considered as a first estimate of the chosen ML model. This prediction function is now used to select the next $n_{\text{act}} = n - n_{\text{init}}$ samples. The selection criteria is a fundamental step in model-based AL approaches and we develop the proposed method in the next sub-section. Note that the n_{act} samples can be detected sequentially (re-training the model after adding each sample), be split into batches of moderate dimension or be detected in one step.

Before moving to the method, we want to highlight the need to select the first samples smartly.

Although it is common to use RS, we argue here that this may not always be the best approach. In model-based AL methods, as the samples that are selected subsequently for labeling are based on a first estimate given by the initial samples, the first set should also be the most diverse, representative and informative. Exploiting model-free AL methods like GSx **?** and iRDM **?** could be really profitable as this would lead to a better initialisation for most datasets. Moreover, a good training set should be diverse and well represent both the feature and the response space. For example, if there is a linear relationship between the features and the response, it is intuitive that a model-free AL method like GSx (or iRDM) would lead to a better sampling as compared to sampling randomly, because a set of samples diverse in the feature space will ensure diversity in the response space as well. The choice of which model-free AL method to use can be inferred from the structure of the dataset: for datasets that are uniformly distributed in the feature space, a diversity based method like GSx would be a good initialiser; a dataset that appears to show prominent clustering in the feature space would be better initialised by iRDM. However if neither of them applies, then RS tends to perform better. Note that this information from the structure of the dataset alone is not sufficient in general, as the details of the response are still hidden. But in many applications, a minor intuition of the relationship between the response and the features is sufficient to conclude the best initialisation sampling scheme for the concerned dataset. A multimodal dataset however, is more complex: a set of samples diverse in the feature space may not always correspond to diversity in the response. In this case using RS may be more apt than using GSx and iRDM as the algorithm prefers no information (of $\mathbf{X}$ and Y) rather than information related only to the features, which is what GSx and iRDM do. As this underlying distribution of the response is not available in AL scenarios, model-based AL methods become all the more relevant. In cases when a good number of labels can be afforded, the best method (out of RS and model-free AL schemes) can also be chosen based on their performances on a small test set.

3.2 Regression tree-based active learning (RT-AL)

The method we propose is based on standard regression trees. Regression trees partition the feature space into a set of K hyper-rectangles, referred to as regions and denoted $\mathcal{R}_k = \prod_{\ell=1}^{p} [a_{k,\ell}, b_{k,\ell}]$ for $1 \leq k \leq K$, and assign a common weight $\gamma_k \in \mathbb{R}$ to each region k:

$$f(\mathbf{x}; \Theta) = \sum_{k=1}^{K} \gamma_k \mathbf{1}_{\{\mathbf{x} \in \mathcal{R}_k\}},$$

where the set of parameters $\Theta = ((\mathcal{R}_k, \gamma_k)_{1 \leq k \leq K})$ corresponds to the set of regions and the associated weights. These parameters can be easily estimated using a labeled set: the weights minimize the quadratic loss for fixed regions, leading to the empirical mean of the observations in each region. The best regions are constructed recursively by finding the best feature and the best splitting point to divide a current region into two sub-regions that makes the prediction less variable. The class of prediction models $\mathcal{F}$ that we consider in Eq. (1) is then the set of piece-wise constant functions. As the splitting process is dyadic, it can be represented as a tree, where each node determines the features to split and its corresponding value, and the final partition is given by the leaves of the tree.

Let I_{init} be the indices of the first samples detected by a model-free method. We construct a standard regression tree with K leaves using the corresponding labeled set $(\mathbf{x}_i, y_i)_{i \in I_{\text{init}}}$, and use it to predict every unlabeled sample: $(\hat{Y}_i^{I_{\text{init}}})_{i \notin I_{\text{init}}}$. Conditionally to the first labeled set, we use the results derived in (**?**) [1] that state that the best performance for a sample of size n with the tree structure learnt on $(\mathbf{x}_i, y_i)_{i \in I_{\text{init}}}$ will be achieved if n_k^* samples are picked for labeling from leaf k, where

$$n_k^* = n_{\text{act}} \frac{\sqrt{\pi_k \hat{\sigma}_k^2}}{\sum_{\ell=1}^{K} \sqrt{\pi_\ell \hat{\sigma}_\ell^2}}; \tag{3}$$

[1] These results were given for purely random Mondrian trees. Here, we use a regression tree that contains the knowledge of the training set, thus improving the structure of the tree.

Algorithm 1 Regression Tree-based Active Learning (RT-AL)

Input: Labeled set $(\mathbf{x}_i, y_i)_{i \in I_{\text{init}}}$ and unlabeled set $(\mathbf{x}_i)_{i \notin I_{\text{init}}}$;
n_{act} the maximum number of new samples to be labeled

1: Construct a standard regression tree with K leaves using $(\mathbf{x}_i, y_i)_{i \in I_{\text{init}}}$
2: **for** $k = 1, \ldots, K$ **do**
3: Compute π_k and $\hat{\sigma}_k^2$ using Eqs. (4) and (5)
4: Calculate the number of samples n_k^* to be labeled from leaf k using Eq. (3)
5: Detect I_{act}^k, the set of n_k^* observations from leaf k
6: **end for**
7: **Output**: The set $\cup_{k=1}^K (\mathbf{x}_i)_{i \in I_{\text{act}}^k}$ of observations to be labeled

where $n_{\text{act}} = n - n_{\text{init}}$, $\hat{\sigma}_k^2$ denotes the variance computed on the true labels in leaf k, and π_k the probability that an unlabeled sample $\mathbf{x}_i$ belongs to leaf k, defined formally as follows: for $1 \leq k \leq K$,

$$\hat{\sigma}_k^2 = \frac{\sum_{i \in I_{\text{init}}: \mathbf{x}_i \in \mathcal{R}_k} (\hat{Y}_i^{I_{\text{init}}} - Y_i)^2}{|i \in I_{\text{init}} : \mathbf{x}_i \in \mathcal{R}_k| - 1}, \tag{4}$$

$$\pi_k = \frac{|i \notin I_{\text{init}} : \mathbf{x}_i \in \mathcal{R}_k|}{N}. \tag{5}$$

This can be seen as a trade-off to select samples diverse in the response but representative of the entire $\mathbf{X}$ space, thus taking into account maximum possible information. Using n_k^*, the n_{act} samples to be labeled are thus divided into the sets I_{act}^k for each leaf k, and the final labeled set proposed to approximate $\mathcal{I}_n$ from Eq. (2) is

$$\hat{\mathcal{I}}_n = I_{\text{init}} \cup (\cup_{k=1}^K I_{\text{act}}^k).$$

If the selection of these samples I_{act}^k from the leaf k is done passively as proposed by (**?**), it would contribute to an additional randomness in the samples selected and we argue that this can be improved. We describe ways to detect these samples efficiently in the following section.

The method is summarized in the form of a pseudo-code in Algorithm 1 and as a flowchart in Appendix A. This AL design works well as it is but can be improved further by splitting the active part into various steps, i.e. by adding few samples at each step, followed by retraining the regression tree at each iteration. This process is then repeated till the desired size of the training set is reached.

3.3 Query method to select the n_k^* samples

Once we know the number of samples to be labeled from each leaf, the obvious choice of selecting them is by passive learning i.e. by random sampling. In Fig. 2 we show how using RT-AL with random sampling in leaves, improves RS, by sampling from different regions in the feature-response space. However, there are two concerns with this choice: firstly, it leads to an additional randomness, which can be detrimental when there is a large amount of unlabeled data and may lead to insignificant improvement of model-based active learning over passive sampling, we observe in Fig. 2 that similar (and redundant) points are labeled. Secondly, even though the tree is constructed on the labeled part of the data, thus is quite informative, a passive sampling of the next set of points ignores the samples that have already been labeled. For instance, in a leaf with a few samples already labeled, selecting more samples passively would not lead to the most diverse and representative set of samples from that leaf.

Thus, we propose to adapt ideas from model-free AL algorithms for the selection of the n_k^* samples. We describe two methods that achieve this goal below. We highlight here that this is an important contribution and difference of our work when compared to (**?**) as the algorithm based on Mondrian trees does not take any feature or response information into account when building the next model.

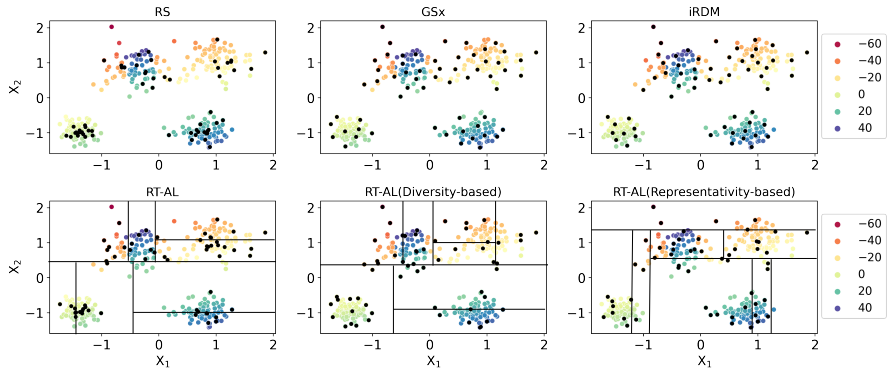

Figure 2: Comparison of the samples selected for labeling (shown as black dots) by our method from a generated dataset with 2 features and 500 samples, using different query criteria (labeled as RT-AL, RT-AL(Diversity-based) and RT-AL(Representativity-based), with passive sampling and model-free AL methods GSx and iRDM. The black lines correspond to the different regions the regression tree splits the feature-response space into. The colors represent the true values of response in the data.

Algorithm 2 Diversity-based query in leaves

Input: Labeled set $(\mathbf{x}_i, y_i)_{i \in I_{\text{init}}}$ and unlabeled set $(\mathbf{x}_i)_{i \notin I_{\text{init}}}$; n_{act} the maximum number of new samples to be labeled

Construct a regression tree with K leaves using $(\mathbf{x}_i, y_i)_{i \in I_{\text{init}}}$
for $k = 1, \ldots, K$ **do**
    Compute π_k and $\hat{\sigma}_k^2$ using Eqs. (4) and (5)
    Calculate the number n_k^* of samples to be labeled from leaf k using Eq. (3)
    for $\ell = 1, \ldots, n_k^*$ **do**
        Use the diversity-based query criteria given by Eq. (6)
        Detect the next sample to be labeled using Eq.(7)
    end for
    Construct I_{act}^k, the set of n_k^* observations from the leaf k
end for
Output: The set $\cup_{k=1}^K (\mathbf{x}_i)_{i \in I_{\text{act}}^k}$ of observations to be labeled

3.3.1 Diversity-based query in leaves

For datasets that would be sampled well by feature-space diversity based methods, we propose to sample points from the leaves keeping this diversity in mind. For leaf $k \in \{1, \ldots, K\}$ from which n_k^* samples are to be chosen for labeling, we label the samples using the following routine.

First, we calculate the L_2 distance of each unlabeled sample in leaf k to all the labeled samples: for $j \in \{\ell \notin I_{\text{init}} : \mathbf{x}_\ell \in \mathcal{R}_k\}$ and $i \in I_{\text{init}}$,

$$d_{ji} = \|\mathbf{x}_j - \mathbf{x}_i\|. \tag{6}$$

Then, we compute d_j, the shortest distance from $\mathbf{x}_j$ to all labeled samples, and we select the unlabeled sample corresponding to the largest distance for labeling:

$$j^* = \underset{j \in \{\ell \notin I_{\text{init}} : \mathbf{x}_\ell \in \mathcal{R}_k\}}{\operatorname{argmax}} \min_{i \in I_{\text{init}}} d_{ji}. \tag{7}$$

Thereby, the samples that are selected for labeling in the leaves are the ones most distant from the samples that have already been labeled in the initialisation step, and also among themselves, such that it gives a more diverse set of samples.

Algorithm 3 Representativity-based query in leaves

Input: Labeled set $(\mathbf{x}_i, y_i)_{i\in I_{\text{init}}}$ and unlabeled set $(\mathbf{x}_i)_{i\notin I_{\text{init}}}$; n_{act} the maximum number of new samples to be labeled

Construct a regression tree with K leaves using $(\mathbf{x}_i, y_i)_{i\in I_{\text{init}}}$
for $k = 1, \ldots, K$ **do**
    Compute π_k and$\hat{\sigma}_k^2$ using Eqs. (4) and (5)
    Calculate the number n_k^* of samples to be labeled from leaf k using Eq. (3)
    Perform a k-means clustering in leaf k with n_k^* clusters
    Calculate R using Eq. (9)
end for
while it has not converged **do**
    for $\ell = 1, \ldots, n_{\text{act}}$ **do**
        Update Δ using the current centroids $(\mathbf{x}_{j_l^*})_{l\neq\ell}$
        Use Eq. (8) to find the next sample to be labeled $\mathbf{x}_{j_\ell^*}$
    end for
end while
Output: the set $\cup_{\ell=1}^{n_{\text{act}}}\mathbf{x}_{j_\ell^*}$ of observations to be labeled

Note that the straightforward use of GSx in each leaf would ignore the samples labeled before training the initial model. In Fig. 2, we illustrate how sampling using our regression tree with the diversity-based criteria (denoted by RT-AL(Diversity-based)) differs from the feature-space diversity-based method GSx **?**. The model-free method GSx only takes into account the feature space information, thus the samples selected by it are spread all over the dataset uniformly. However, our method takes into account the response through the regression tree, thus resulting in a set of samples that are diverse in response as well, as the samples are now spread in the regions defined by homogeneous values of the response. We also illustrate the efficiency of this query criterion in Section 4. The pseudo-code of the algorithm is provided in Algorithm 2.

3.3.2 Representativity-based query in leaves

For datasets with prominent clustering, we propose a representativity-based criteria to select the samples from the leaves. We denote $J_1 = \{1, \ldots, n_1^*\}$ and $J_k = \{\sum_{u=1}^{k-1} n_u^* + 1, \ldots, \sum_{u=1}^{k} n_u^*\}$ for $k \in \{2, \ldots, K\}$. We first perform a clustering in each leaf k only on the unlabeled samples, with n_k^* clusters, denoted by $(c_\ell)_{\ell\in J_k}$. The sample closest to the centroid of these clusters, denoted $(\mathbf{x}_{j_\ell^*})_{\ell\in J_k}$, is a good representative of these clusters. However, it is possible that all the n_k^* centroids do not form a diverse enough set of samples in leaf k, and even more generally, that the set of all $n_{\text{act}} = \sum_{k=1}^{K} n_k^*$ centroids do not form a diverse enough set of samples in the feature space, amongst themselves and also with respect to the samples that have already been labeled. For each cluster $\ell = 1, \ldots, n_{\text{act}}$, we thus update the centroids by optimising the following criteria,

$$j_\ell^* = \underset{j\in c_\ell}{\text{argmax}}[\Delta(\mathbf{x}_j) - R(\mathbf{x}_j)], \tag{8}$$

that considers the diversity in the measure $\Delta(\mathbf{x}_j)$ and maintains the representativeness in the measure $R(\mathbf{x}_j)$, where for each unlabeled sample $\mathbf{x}_j \in c_\ell$, for $\ell = 1, \ldots, n_{\text{act}}$,

$$R(\mathbf{x}_j) = \frac{1}{|c_\ell| - 1} \sum_{\substack{\mathbf{x}_m \in c_\ell \\ m\neq j}} \|\mathbf{x}_j - \mathbf{x}_m\|; \tag{9}$$

$$\Delta(\mathbf{x}_j) = \min_{m\in I_{\text{init}}\cup\{j_l^*\}_{l\neq\ell}} \|\mathbf{x}_j - \mathbf{x}_m\| = \min_{m\in I_{\text{init}}\cup\{j_l^*\}_{l\neq\ell}} d_{jm}. \tag{10}$$

Note that at this point, $\Delta(\mathbf{x}_j)$ includes information of not just the cluster centroids of the cluster in leaf k, but also of the cluster centroids in all other leaves, and all the samples that were labeled beforehand.

The above routine is then repeated for all other clusters in all the leaves and is iterated over till the n_{act} samples to be selected are optimised (or till a threshold is reached). In Fig. 2, we illustrate the benefits of this representativity-based criteria (denoted by RT-AL(Representativity-based)) by comparing it with the model-free AL method iRDM **?**. As it is only a feature-space based method, we see that the samples selected by iRDM are well representative of the clusters, however they do not represent the response enough. Thus, many samples containing similar information of the response are chosen. However, as our method takes the response into account, we choose a set of samples that are representative of the response as well. We also show in Section 4 how this algorithm works efficiently. The pseudo-code of the algorithm is provided in Algorithm 3.

4 Experimental validation

We illustrate our method on several datasets, and compare its performance with many state-of-the-art methods. First, we describe the settings we used. This is followed by an evaluation of the performance of our method over several benchmark datasets in Section 4.2. Finally, we illustrate our results in detail on the Superconductivity dataset in Section 4.3.

4.1 Settings

We compare the performance of our method with several others, which we list here along with their settings and hyperparameters used. Random Sampling from a uniform distribution (RS) is used as the simplest case of sampling. The model-free AL methods we consider are:

- GSx: as in Ref. (**?**), the first sample chosen is the centroid of the training set, and the rest of the samples are chosen based on diversity measured by the L_2 distance.
- iRDM: the samples selected for labeling from the clusters were optimised at most 5 times, or lesser if it stopped evolving before, as proposed in Ref. (**?**). The Matlab code for iRDM is available at https://github.com/drwuHUST/iRDM.
- Cube: the cube sampling method proposed in Ref. (**?**), using uniform inclusion probabilities. The code is available in the R package `sampling`.
- Graph-Based: as proposed in Ref. (**?**) where the closeness is measured using the L_1 distance.

We also draw a comparison with known model-based AL methods in regression:

- GSy: also proposed in Ref. (**?**), GSx is used as the model-free method for constructing the first regressor. The responses for the unlabeled samples are predicted by Ridge regression at every step, exactly as in (**?**), and samples to be labeled are detected using the L_2 distance in the response space using these predictions.
- QBC: Query By Committee (**?**) using regression trees as the models in a committee of 5 models. This non-linear variant of QBC was chosen and implemented by us as our method is based on trees.
- EMCM: the model-variance based approach by (**?**). We use the non-linear version of EMCM, that uses Gradient Boosting Trees (GBT) in the committee of models, with 100 trees in the GBT (except for the superconductivity dataset where we use 25 trees, due to the large size of the dataset). The number of models in the committee are 4 as in Ref. (**?**). As Batch-EMCM is worse than EMCM in performance, we do not use it for comparison in this paper.
- Mondrian trees: denoted by MT, the approach in Ref. (**?**). The python code for Mondrian trees method (MT) is available at https://github.com/jackr-goetz/Mondrian_Tree_AL.

Batch Mode Deep Active Learning (BMDAL): denoted by LCMD (where LCMD stands for the selection method - Greedy distance maximisation in largest cluster), the method proposed in Ref. (**?**). Their code is available at , and we use the same hyperparameters as the ones in the code.

For all the methods mentioned above, the hyperparameters from the original papers were used unless stated above. For our method RT-AL, the hyperparameters are as follows: the minimum samples in the leaf of the decision tree was set to 5 (i.e. the trees do not branch if the number of samples in a leaf is smaller than 5), for meaningful variance calculations, while simultaneously optimising the tree as much as possible. The sampling from the leaves for our method was done with respect to the results of passive learning and model-free AL methods.

For the methods that require an initial set of labeled samples (i.e. Graph-Based, QBC, EMCM, MT, LCMD and RT-AL), the set is sampled using RS, GSx or iRDM depending on the best performer out of these for each dataset. In appendix B we show how this is indeed the smarter choice for initialisation for all these methods. Note that for GSy, GSx is used to label the initial set as proposed by (**?**).

For each dataset and each method, we consider a test set of size 50% of the whole dataset, uniformly drawn, on which the final regressor inferred on the constructed training set is tested, unless specified otherwise. Considering a test set particularly large is a choice made to keep the test set as diverse as possible. When considering the model-based AL methods, we select $n_{\text{init}} = 15$. The Root Mean Squared Error (RMSE) of the target (response) was used as the performance measure for all the methods,

$$\text{RMSE} = \sqrt{\frac{1}{T}\sum_{i=1}^{T}(f(\mathbf{x}_i) - y_i)^2} \tag{11}$$

where T is the size of the test set, y_i are the true labels of the test set, and $f(\mathbf{x}_i)$ is the prediction for a test sample $\mathbf{x}_i$. The final regressor that we use to evaluate every method at each step is a Random Forest built on the labeled set selected by the method. The number of trees in the forest were set to 100, which is a good estimate of the optimum hyperparameter throughout the prediction phase as shown in Appendix C. The depth of the trees in the forest was defined by minimum samples in the leaf, set to 3, so as to avoid overfitting. Note, as we compare different methods throughout the prediction phase, we keep the same hyperparameters for all the methods for consistency. Moreover, random forest was chosen as the final predictor because it is versatile and robust to outliers. It is also particularly suited for our method as we rely on its basic architecture - regression trees, to construct our training set. The experiments were run 200 times each for all the methods to compute statistics, and we discuss the relevance of differences based on a t-test at level 5%. To see the evolution of the performance of model-based AL methods with respect to the number of labeled points, we do the following: the first 15 samples to be labeled are selected using a passive/model-free AL method. This number is kept small so as to take maximum benefit of the model-based part of the method. Then, the active step is performed after every 20 samples have been labeled (except for the first round where only 5 new samples are labeled, for uniformity in the plots). Thereafter, the new samples are added to the training set and the model is refit. Note that methods like RS, GSx, GSy, Graph-based, QBC and EMCM are sequential, so we label after each point, but measure the performance only after every 20 points.

Scaling the data, for example to a normal distribution with mean equal to 0 and standard deviation equal to 1, or to a scale of desired range might be detrimental in some cases as it can hide certain technicalities of the dataset leading to a loss of information. Thus, we do not scale the data in this study, unless there is a specific distance calculation in the method (for example in GSx) that calls for it.

The python code used in this work can be found here https://github.com/AshnaJose/Regression-Tree-based-Active-Learning, along with the datasets that are used in the next section.

4.2 Comparison with state-of-the-art

We illustrate the behavior of our AL method on several benchmark datasets in this section, and compare it with state-of-the-art methods. We consider the passive method RS, model-free AL methods GSx, iRDM,

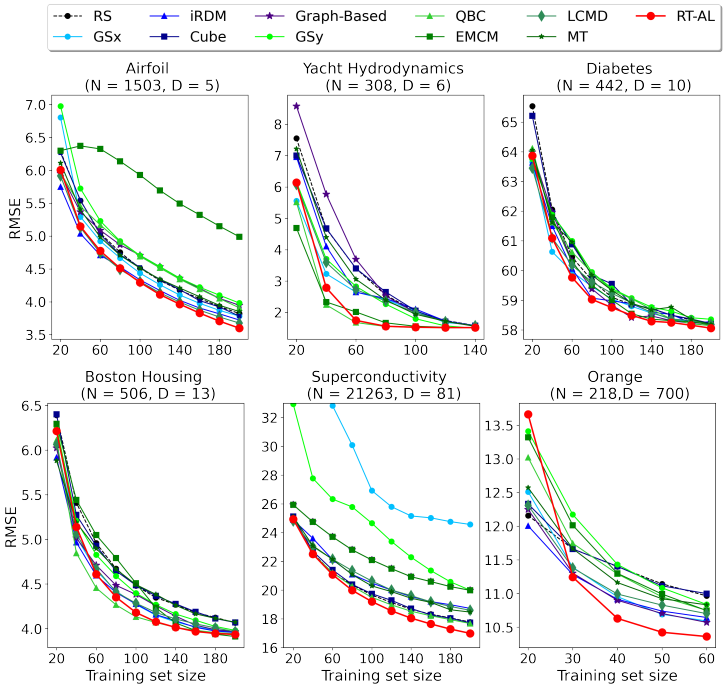

Figure 3: Performance in prediction using RMSE averaged over 200 runs when the training set is constructed using passive and active learning methods for 6 real datasets. The training set size varies from 20 to 200 (except for orange dataset where it varies from 20 to 60, and for Yacht dataset, where it varies from 20 to 140).

the cube sampling method, the graph-based approach, and model-based AL methods GSy, QBC, EMCM, MT and LCMD, all as described before. The method (depending on the datasets) to sample in the leaf is selected depending on the initialiser (e.g. RT (Diversity-based) where GSx is used to label the first set of samples).

We use 6 datasets of various sizes, and of different number and types of attributes (continuous and discrete), namely Airfoil, Boston Housing, Diabetes, Orange Juice, Superconductivity (denoted by SP) and Yacht Hydrodynamics, all commonly used in regression tasks and available at UCI repository[2], except Orange Juice that is available in R package `cggd` and Diabetes that is available in R package `lars`. The details of these datasets can be found in Table 1. The 2-component PCA plots and the histograms of the response are provided in Appendix D to understand the distributions of the features and the response better, and the extent to which they are linked. Reminder however, that the post analysis on the response cannot be done in practice on a new dataset as no response would be available for the same.

The mean and the variance of RMSE over 200 runs for a sample size of 100 (except for Orange Juice dataset that is quite small so we consider 60 labeled samples) are given in Table 1. Our method has been represented as RT-AL. The initialiser, for methods that require and initial labeled set, corresponds to the best method out of passive sampling, GSx and iRDM, which is dataset dependent. The RMSE values of the initialiser have been italicised in the table for each dataset. Overall, it is clear from the table that our approach performs the best for all datasets. We describe the results in detail below.

Firstly, we draw a comparison between passive sampling and model-free AL methods GSx, iRDM, Cube and graph-based:

- We can conclude from the table that the performance of the passive/model-free methods depends

[2]https://archive.ics.uci.edu/

Table 1: Performance for 100 labeled samples (except for orange dataset where we consider only 60 labeled samples due to the smaller size of the dataset), depicted using RMSE (average and variance over 200 repetitions). Best performance highlighted with bold and star (with the best passive/model-free active learner shown in italics, that corresponds to the method used to initialize Graph-based,QBC, EMCM, MT, LCMD and RT-AL for each dataset, and accordingly in the leaves for RT-AL), their statistical equivalent result in bold (t-test at level 0.05)

	Airfoil	Yacht	Diabetes	Boston	SP	Orange
N	1503	308	442	506	21263	218
D	5	6	10	13	81	700
RS	4.51	*2.04*	**59.10**	4.48	*19.61*	10.97
	(0.07)	(0.46)	(5.09)	(0.21)	(0.82)	(1.06)
GSx	4.44	2.04	***59.04***	*4.27*	26.89	*10.59*
	(0.05)	(0.54)	(5.35)	(0.20)	(8.25)	(1.01)
iRDM	***4.33***	2.11	**58.75***	4.28	20.49	10.64
	(0.04)	(0.63)	(6.02)	(0.21)	(1.42)	(1.10)
Cube	4.52	2.00	59.55	4.49	19.73	11.00
	(0.06)	(0.50)	(7.89)	(0.27)	(1.00)	(1.08)
Graph-Based	4.70	2.09	**58.83**	4.38	19.45	10.57
	(0.08)	(0.71)	(6.37)	(0.29)	(0.43)	(0.96)
GSy	4.71	1.80	59.40	4.40	24.53	10.84
	(0.10)	(0.30)	(6.26)	(0.24)	(7.70)	(1.51)
QBC	4.69	**1.52***	59.35	**4.13**	19.50	10.75
	(0.09)	(0.11)	(6.44)	(0.20)	(1.53)	(1.00)
EMCM	5.93	**1.56**	**59.01**	4.51	22.1	10.75
	(0.41)	(0.11)	(7.15)	(0.17)	(3.18)	(1.02)
MT	4.51	1.94	59.34	4.49	20.26	10.83
	(0.06)	(0.46)	(6.40)	(0.26)	(1.53)	(1.06)
LCMD	**4.31**	2.00	59.24	4.28	20.63	10.70
	(0.04)	(0.49)	5.86	0.22	1.43	0.99
RT-AL	**4.30***	**1.53**	**58.98**	**4.18***	**19.18***	**10.36***
	(0.04)	(0.12)	(5.68)	(0.19)	(0.76)	(1.01)

.

strongly on the dataset. This is in line with our understanding of the structure of the dataset using the principle component analysis plots.

- iRDM and GSx, the supposedly smart model-free AL methods in comparison with RS, are the best for 4 datasets, while random sampling performs better than them for SP and Yacht datasets. Thus for these datasets, we use RS to sample from the leaves.
- For datasets like Diabetes, the geometry of the feature space is uniformly distributed, and so is the distribution of the response. This explains why almost all the methods perform similar, however, using our diversity-based criterion in the trees makes the most sense due to the uniformity.
- For the Superconductivity dataset on the other hand, as it is not uniformly distributed, or well clustered, passive sampling turns out to be better than model-free AL approaches.
- In general, the cube sampling method performs the worst out of these schemes, and does not show a consistency with respect to all the datasets either.
- The graph-based approach has the highest variance among these methods, thus suggesting that it is not a reliable scheme.

On comparing all the model-based AL approaches i.e. GSy, QBC, EMCM and MT, we conclude the following:

- It is clear from the table that not all model-based AL methods perform better than the model-free ones.
- For datasets where GSx was the best initialiser, GSy does not tend to improve GSx. We guess that this is because the improvement by GSy is guaranteed only for linear regression problems ((**?**) use ridge regression), and although we use ridge regression for selecting the samples, we predict using a random forest to be consistent, and because RFs are more robust and achieve better performance on all datasets.
- QBC, where we use regression trees in the committee of models, performs modestly in terms of both RMSE and variance. This again highlights the benefits of the tree, and its ability to generate better models in general. This improvement is also a result of the fact that regression trees deal with outliers better, thus select a better set of samples.
- In general, out of the model-based AL methods, EMCM and MT have the largest values of variance, and are therefore of limited practical use.
- Both EMCM and MT give modest performances for some datasets, but are really bad sampling schemes for some others, and thus lack consistency.
- LCMD, the deep active learning method, is also not consistent over the different datasets. It has a high variance, and in some cases higher RMSE than even random sampling. Moreover, as neural networks require a lot of data to train well, we argue here that AL using deep learning is not a good choice for a small set of samples.

Now, coming to a detailed evaluation of our method:

- Our method (labeled RT-AL in Table 1) is the best performer for all the datasets in consideration.
- Our method is the only one that is consistent for all the datasets, which is very important for any ML model.
- Of all the methods in Table 1, our approach has the lowest value of variance for all datasets, thus showing how robust our sampling scheme is. We point again that the variance of a sampling method is very important, because practitioners will construct the training dataset only once.

Table 2: Pairwise ranking over 200 runs. The numbers denote the number of times our method, RT-AL, performs better than the respective methods by row, for 100 labeled samples (except for orange dataset where it is 60), among a series of 200 runs, for each dataset by column.

	Airfoil	Yacht	Diabetes	Boston	SP	Orange
RS	151	193	115	163	128	167
GSx	138	186	112	125	200	152
iRDM	102	187	107	131	169	154
Cube	157	189	124	172	131	172
Graph-Based	190	192	111	149	123	137
GSy	171	170	118	151	197	141
QBC	182	93	120	80	115	165
EMCM	200	139	108	160	191	164
MT	154	179	116	171	154	180
LCMD	109	190	122	134	167	160

- This means using a regression tree in the query strategy, along with utilising model-free AL methods for initialising and using diversity and representativity-based criteria inside the leaves, is in fact a successful sampling strategy, that produces the most consistent and the least varying results.

Fig. 3 depicts the plots with evolution of RMSE averaged over 200 runs, with respect to the sample size, and the boxplots for the 200 runs are available in Appendix D. The figure further confirms the conclusions from the table, and we see that our method succeeds to perform very well for all the datasets, which are of very different sizes and dimensions, thus highlighting the versatility of our approach.

In Table 2, we show the number of times our method RT-AL outperformed the state-of-the-art with the help of a pairwise ranking among 200 runs. For example, in a series of 200 runs, our method gives a lower RMSE than RS 193 times for the yacht dataset. The numbers in this table show that our approach is very consistent, as we rank first a very large number of times, as compared to other methods. It also highlights the subtle nature of the diabetes dataset, as almost all the methods seem to give similar results due to uniformity in its feature space. It is however important to note that even though the table is highly rewarding for our method, we can derive more information from the values of the RMSE. For example, even though it seems like RT-AL is better than RS 128 times for the superconductivity dataset, the 72 times that RS is better may not all be statistically significant. We show using histograms of the difference in RMSE between our method and all others in Appendix E, that when another method is better than ours, the difference between the RMSEs is very low. The main message from the table is that our method is the most consistent one, regardless of the dataset.

An important point to mention here is that even though the final model we train is a random forest, it does not restrict the use of RT-AL to it. In appendix F we show that RT-AL is the best sampling scheme for two other classes of final predictors, linear models and neural networks. This shows the robustness of our approach and its ability to adapt to different machine learning tasks.

Finally, we would also like to highlight the computational complexity of the methods discussed in this work. We argue that compared to other model-based AL methods, our method is computationally cheaper. For example, QBC is a computationally complex method if non-linear models are used in its committee because to label a single new sample, QBC needs to train 5 models (as we set the number of models in the committee to 5), while our method is able to label a set of samples by training a single regression tree. Similarly, EMCM is computationally complex as it requires training a GBT with many trees in order to label new samples. We provide a table of computation times for all the different methods in Appendix G for reference.

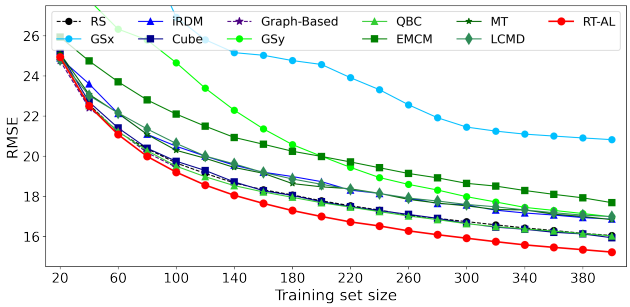

Figure 4: Performance in prediction using RMSE averaged over 200 runs when the training set is constructed using passive and active learning methods on the dataset Superconductivity (N = 21263, D = 81). The training set size varies from 20 to 400.

4.3 Illustration on a real dataset

In this section, we illustrate our method on the superconductivity dataset[3], taken from the UCI repository[4]. As this dataset is very large, with about 21k samples and 81 attributes, we use 75% of the dataset in the test set.

Like before, for our method, the number of samples labeled by the passive/model-free AL method are 15, after which new samples are labeled using the tree, with the tree being fit regularly as mentioned in Section 4.1. The results are depicted in Figure 4 using mean RMSE over 200 runs as the performance measure, for all the methods described in the previous sections. Our method has been represented as RT-AL, that uses the regression tree with passive sampling (RS), as for this dataset, we find that RS is better than the model-free AL approaches. This is clear once we look at the structure of the datatset using the Principal Component Analysis (PCA) provided in the supplementary material, where we see that this dataset is not uniformly distributed and may not be well clustered. Moreover, as a post analysis, we also see that this dataset is multimodal, therefore any conclusions drawn on it based on features alone will not be enough. Thus, we use RS as the initialiser as well as the method to pick the samples from the leaves. The experiments were repeated till 400 samples were labeled. In Fig. 5, we present the results in the form of box plots, that gives a more clear insight of the variance and the outliers within the 200 runs. We can draw the following conclusions from the learning curves shown in Fig. 4 and 5:

- For all the methods, the RMSE on the test set decreases with increasing training set size, that shows that the samples added by all these methods contribute positively towards a better ML model in general.
- The best performance is guaranteed by our method RT-AL, as is evident from the figure, not only for a training set size as small as 20, but also for large training set sizes.
- Comparing RS with all the model-free AL methods (i.e. GSx, iRDM, Cube and graph-based) we conclude that indeed RS performs better for this dataset, not only for a few labeled samples, but also for a training set size as large as 400. This also shows that our conclusion from the structure of the dataset was right. Thus, choosing RS as the initialiser and inside the leaves is the right choice for this dataset.
- The performance of GSx is particularly bad, as expected, as the dataset is not uniformly distributed.
- iRDM performs satisfactorily for a very small number of labeled samples, however as the training set size increases, it fails as the data is now separated into far too many clusters even though there is no inherent clustering present.

[3]For a deeper understanding, we illustrate our method on a generated multimodal dataset in Appendix H as well.
[4]https://archive.ics.uci.edu/ml/datasets/superconduct- ivty+data/

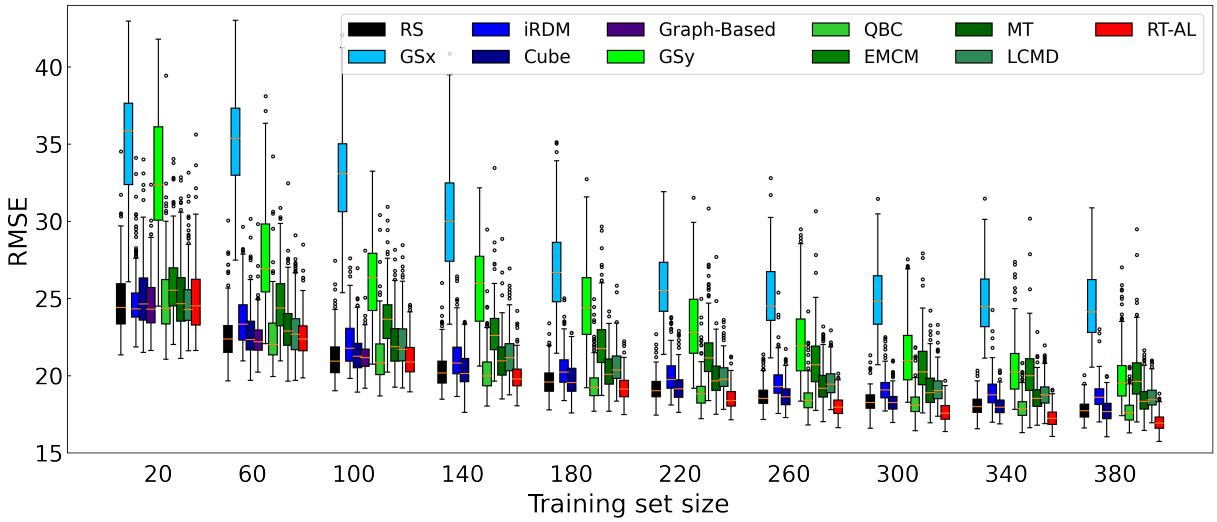

Figure 5: Performance in prediction using boxplots over 200 runs when the training set is constructed using passive and active learning methods on the dataset Superconductivity. The training set size varies from 20 to 400.

- Coming to the model-based AL methods, GSy succeeds to improve GSx, however it still performs very poorly as it is not suited for non-linear problems (while we use random forests as the final predictor, samples are selected to be good for Ridge regression).
- MT does not succeed to give good results, and gives large values of RMSE with a very high variance, that is seen in Fig. 5. This can be understood from the fact that there is a lot of randomness in the algorithm, be it in the way of initialisation, the method to select samples, or the fundamental construction of the Mondrian tree itself.
- EMCM also does not give a very good performance, close to that of GSy, even though it deploys GBT in their algorithm, thus being suitable for the non-linear case.
- LCMD also does not succeed to sample well and builds a model high in variance, with a large number of outliers.
- QBC gives a modest performance, however its high variance and large number of outliers in Fig. 5 compared to our method does not make it an apt choice for AL in general.

Therefore, it is clear that adding information of the response space via our regression trees tremendously improves the sampling by lowering the RMSE. Our method also has the lowest variance, and the least number of outliers for the entire learning curve. Thus, our approach is simple, robust, and overall the most apt choice for AL in regression. Being low in variance also implies that it is very reliable, as in practice, the method would be used only once.

Another important factor affecting the performance of the model is the number of samples labeled during the initialisation process, either by passive sampling, or by a model-free AL method. The optimum number of samples to be labeled together by the two parts would of course depend on the maximum labeling capacity, but the ratio n_{init}:n_{act} can vary. In Figure 6, we compare two cases: one is our standard routine where n_{init} is 15 and n_{act} is $n - n_{\text{init}}$, while the tree is refit after every 20 samples have been labeled. The other case we consider is when $n_{\text{init}} = n_{\text{act}}$, considering independent experiments at each training set size, thus giving equal weight to the passive/model-free AL and the model-based AL part. We see that for training sets of small size, giving equal weight to both parts is more or less similar to giving a higher weight to the model-based

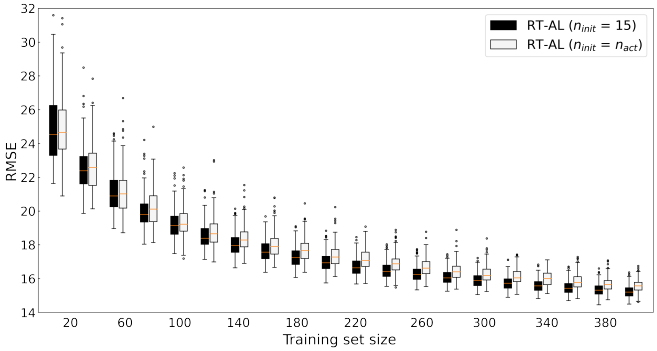

Figure 6: Performance in prediction using boxplots over 200 runs when the training set is constructed using RT-AL, while changing the ratio n_{init}:n_{act}, on the dataset Superconductivity. The training set size varies from 20 to 400.

AL part, but for larger training sets, the AL part is more beneficial and gives better results. This is intuitive, as a regressor (a regression tree in our case), learnt on a very small training set, may not be very accurate. This is particularly the case for our trees, as we fix the depth of the tree (via the hyperparameter minimum number of samples in a leaf set to 5), so the tree is not deep enough for a small training set. However, with substantial training of the regressor, it indeed performs significantly better. On observing the evolution of the tree structure in our AL scheme, we see that the first two or three trees vary slightly in their structure, however subsequently, the basic structure of the tree remains the same, with only further branching. It confirms that in fact model-based AL is really important for training sets of large sizes, whereas model-free AL has usually been illustrated and motivated for small sample sizes.

Note that although we have shown this result only for RT-AL with RS as the initialiser and in the leaves, this behaviour also holds true for other model-free AL methods as initialisers and in the leaves (i.e. GSx + RT (Diversity-based) and iRDM + RT(Representativity-based)).

5 Conclusion and discussion

In this paper we propose a model-based AL method for non-parametric regression that smartly uses the knowledge in the available data through the regression tree structure to improve the performance of the model. It selects new samples from each leaf of the regression tree, that is constructed on the labeled part of the dataset, thus the new samples contain information of both the features and the response. We also take advantage of already established model-free AL methods by utilising them to improve the algorithm. We get good and consistent results for several benchmark datasets, and succeed to show that our approach works better than the existing schemes in active learning in regression. We especially highlight the low variance of the results on all the experiments made in this study, which is vital when doing active learning.

As the proposed RT-AL method is based on regression trees, it is possible that the samples selected by it are more suitable for tree-based prediction models like Random Forests, Gradient Boosting Trees etc. This is due to the fact that RT-AL tends to sample points where the gradient of targets is large, thus helping the tree based algorithms to find good boundaries between regions homogeneous in the target. As this is not the foundation of model-training for other machine learning models, it may not be the best sampling strategy for other models. Having said that, as Random Forests and other ensemble tree methods are known to generalize well, and show good performance especially when the dataset is small in size, we believe that our method of sampling is robust in general.

The use of regression trees opens the door to many extensions. For example, every region of the tree is supposed to be homogeneous in the output value (low variance), so one application of interest is to construct a set of samples tuned for specific range of target values, e.g. one may be interested in constructing a training

set that achieves good performance to detect materials with high values of target. Moreover, although RT-AL is based on trees, it is also interesting to consider extensions of the method to other models like neural networks in the future. As a perspective, there is a need of theoretical guarantees of active learning methods. One can hope for a better rate of convergence than the one achieved by passive learning, given by the central limit theorem, but this study is out of the scope of this paper.

6 Statements and Declarations

Conflict of interest: The authors declare that they have no conflict of interest.

Code availability: The code is available at https://github.com/AshnaJose/Regression-Tree-based-Active-Learning along with the datasets.

Funding: This work has been partially supported by MIAI@Grenoble Alpes (ANR-19-P31A-0003).

A RT-AL

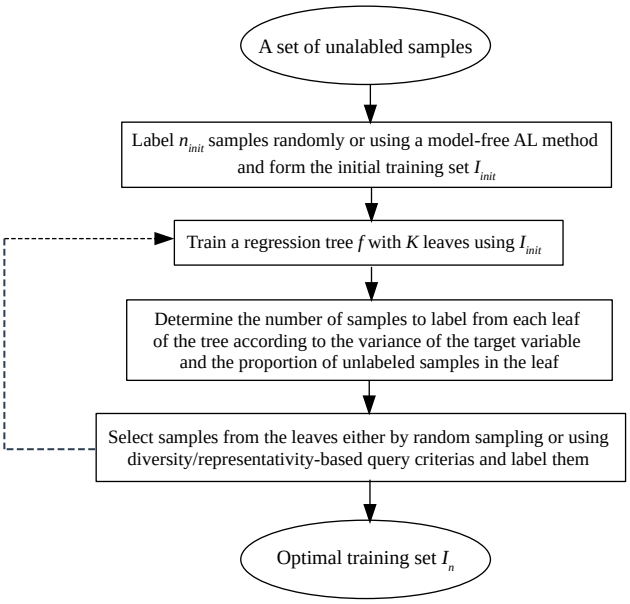

Figure 7: Flowchart giving an overview of Regression Tree-based Active Learning (RT-AL)

B Initialisation

In Table 3, we compare using RMSE for 100 labeled samples (except for orange dataset where we consider only 60 labeled samples) the performance of AL methods requiring an initial set of labeled samples, when the initial samples are selected using random sampling (shown with (RS) in front of the method name), and when the initial samples are selected smarty (shown in bold) using iRDM for Airfoil dataset and GSx for Diabetes, Boston and Orange datasets. The datasets in the main paper for which RS was the best initialiser have not been shown here. We see from the table that indeed initialising the first model smartly helps in building a better model as the RMSEs for this case are lower than for the cases initialised by RS. Thus, the first set of samples should indeed be chosen smartly.

Table 3: Performance for 100 labeled samples (except for orange dataset where we consider only 60 labeled samples due to the smaller size of the dataset), depicted using RMSE (average over 200 repetitions) with and without smart initialisation.

AL method	Airfoil	Diabetes	Boston	Orange
Graph-Based (RS)	5.16	58.77	4.40	10.77
Graph-Based	**4.70**	**58.83**	**4.38**	**10.57**
QBC (RS)	4.75	59.28	4.17	10.94
QBC	**4.69**	**59.35**	**4.13**	**10.75**
EMCM (RS)	6.00	58.82	4.47	10.94
EMCM	**5.93**	**59.01**	**5.51**	**10.75**
MT (RS)	4.51	59.55	4.52	10.96
MT	**4.51**	**59.34**	**4.49**	**10.83**
RT-AL (RS)	4.48	59.41	4.40	10.87
RT-AL	**4.30**	**58.98**	**4.18**	**10.36**

C Hyperparameters for the final model: Random Forest

The two parameters of interest in the final regressor, Random Forest, are the depth of the tree (defined by the minimum number of samples required to form a leaf), and the number of trees in the forest. We set the minimum samples in the leaf to 3 (while the default is 1) to avoid over-fitting. Further, we take the number of trees in the forest to be 100 as it is a good estimate of the point where the performance of different methods that we test converge, throughout the prediction phase (as shown in Figure 8 and 9). Although hyperparameter optimisation for the final regressor is important in general, it depends on the samples selected, thus the method of sampling used, therefore it could be unfair to compare different methods with different hyperparameter optimisations. So we chose 100 trees, which is a safe hyperparameter for all the methods.

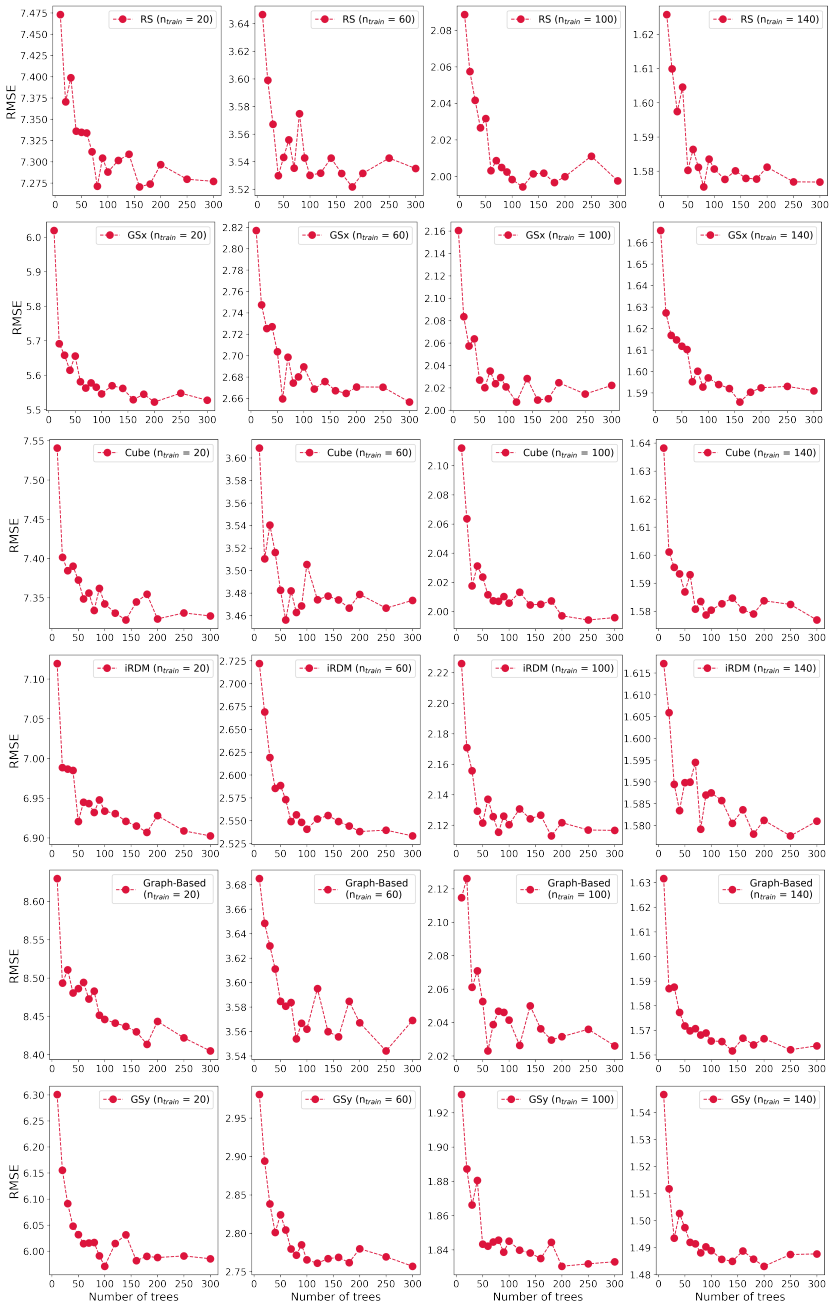

Figure 8: The performance in prediction using RMSE averaged over 200 runs for different stages in the training using passive and active learning methods for Yacht dataset, as a function of the number of trees in the Random Forest.

D Plots for Section 4.2

In this section, we present the two-component PCA plots and response histograms for all the real datasets studied in Section 4.2 of the main paper. We also show the performance in prediction using our method and compare it with the state-of-the-art with the help of boxplots, that highlight the difference in the RMSE,

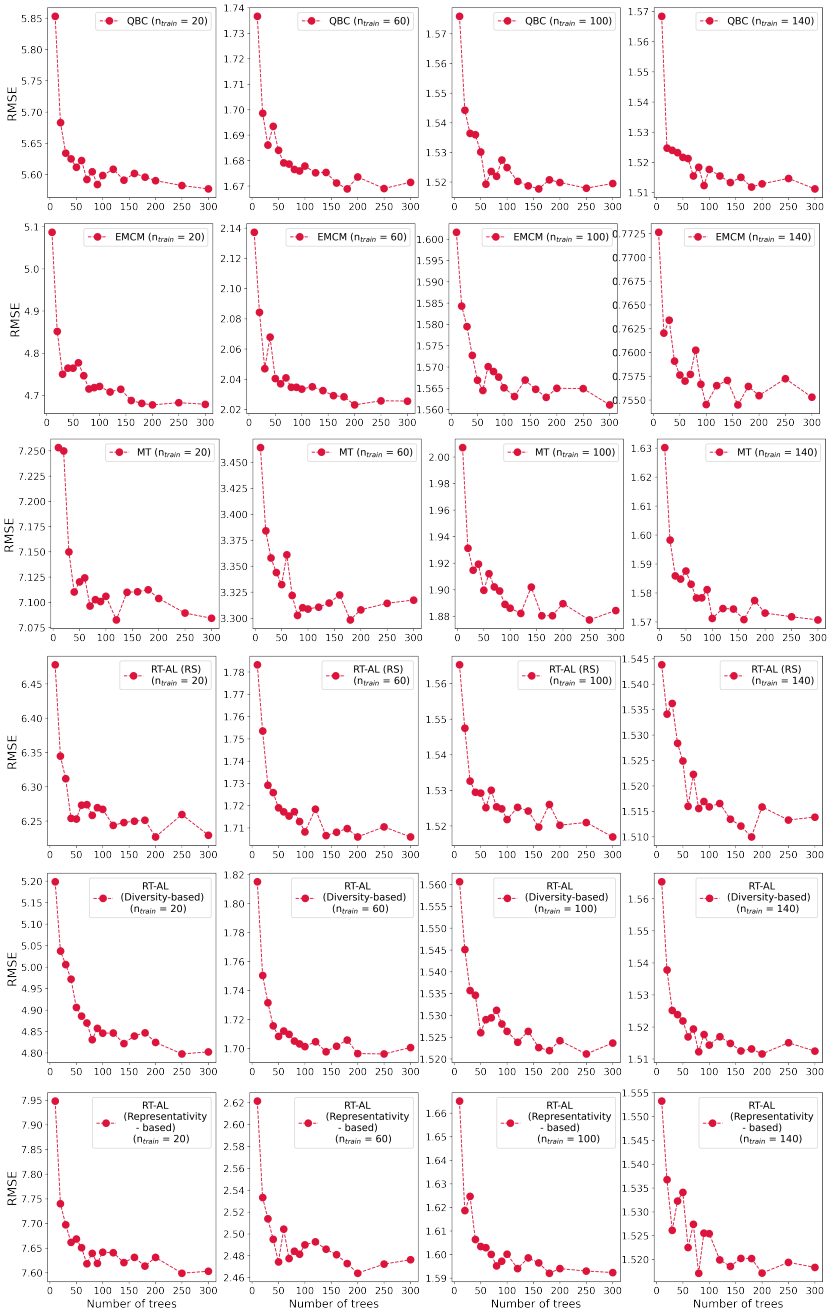

Figure 9: The performance in prediction using RMSE averaged over 200 runs for different stages in the training using passive and active learning methods for Yacht dataset, as a function of the number of trees in the Random Forest.

and also in the variance and the number of outliers.

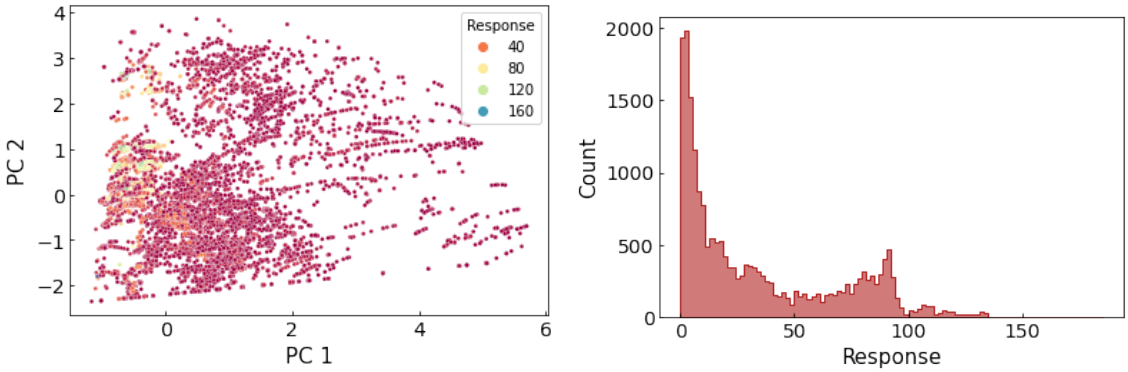

Figure 10: Two-component PCA and histogram of the response of the Superconductivity dataset. Performance in prediction using boxplots over 200 runs when the training set is constructed using passive and AL approaches. The training set size varies from 20 to 200.

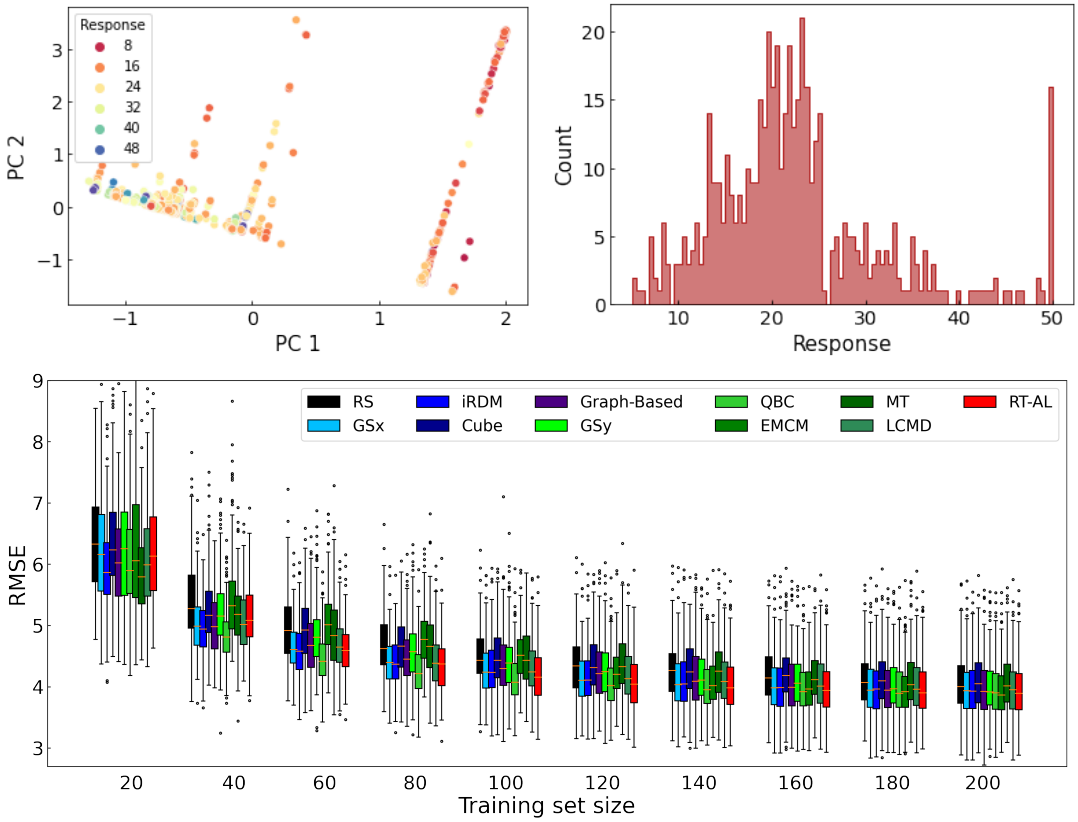

Figure 12: Two-component PCA and histogram of the response of the Boston Housing dataset. Performance in prediction using boxplots over 200 runs when the training set is constructed using passive and AL approaches. The training set size varies from 20 to 200.

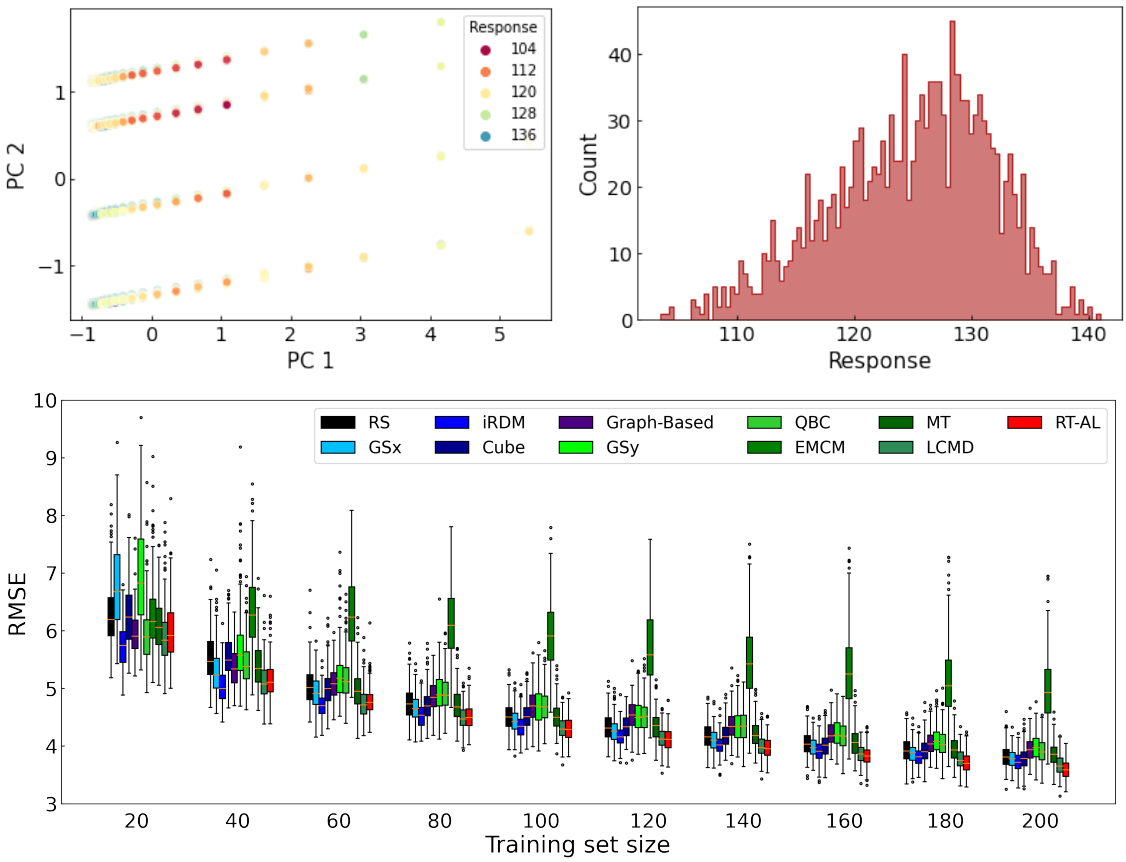

Figure 11: Two-component PCA and histogram of the response of the Airfoil dataset. Performance in prediction using boxplots over 200 runs when the training set is constructed using passive and AL approaches. The training set size varies from 20 to 200.

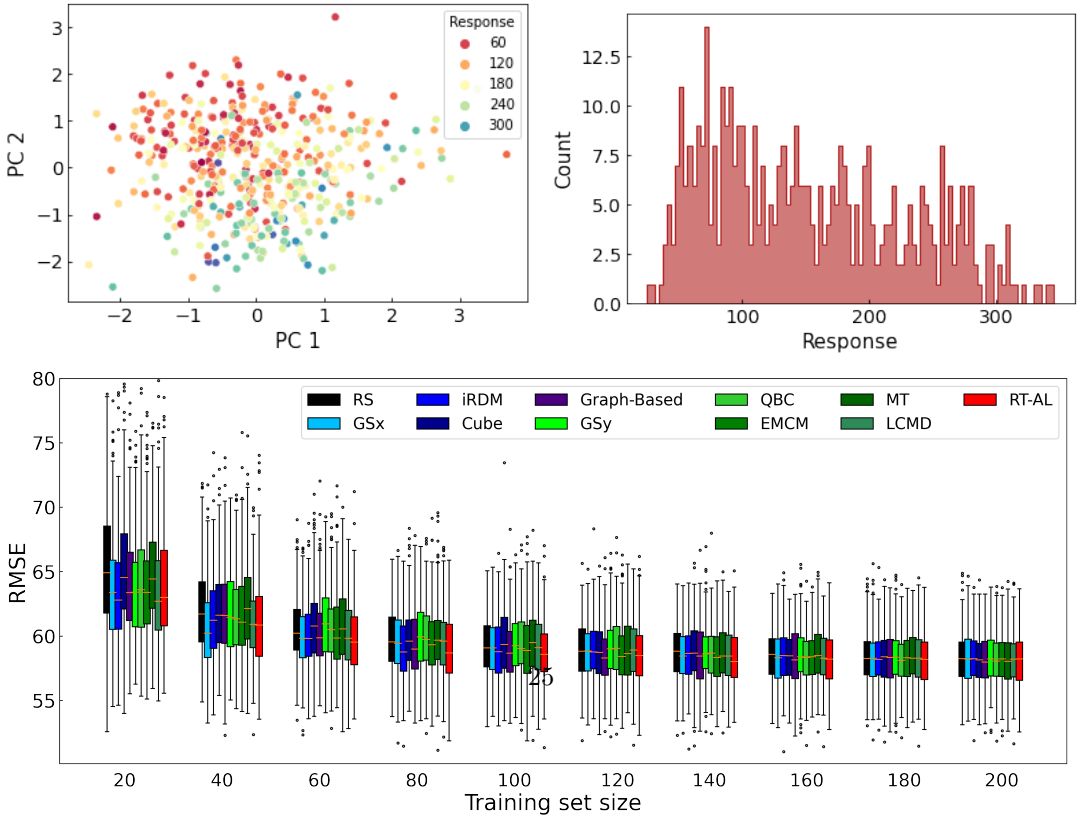

Figure 13: Two-component PCA and histogram of the response of the Diabetes dataset. Performance in

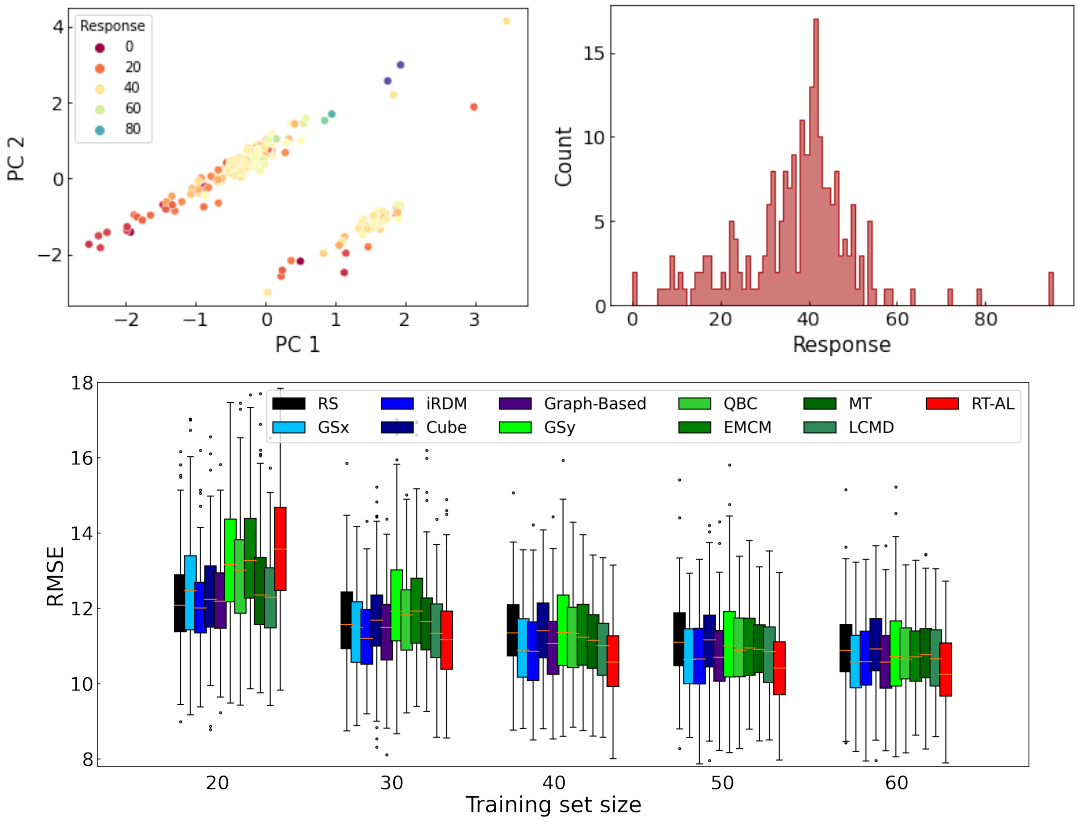

Figure 14: Two-component PCA and histogram of the response of the Orange juice dataset. Performance in prediction using boxplots over 200 runs when the training set is constructed using passive and AL approaches. The training set size varies from 20 to 60.

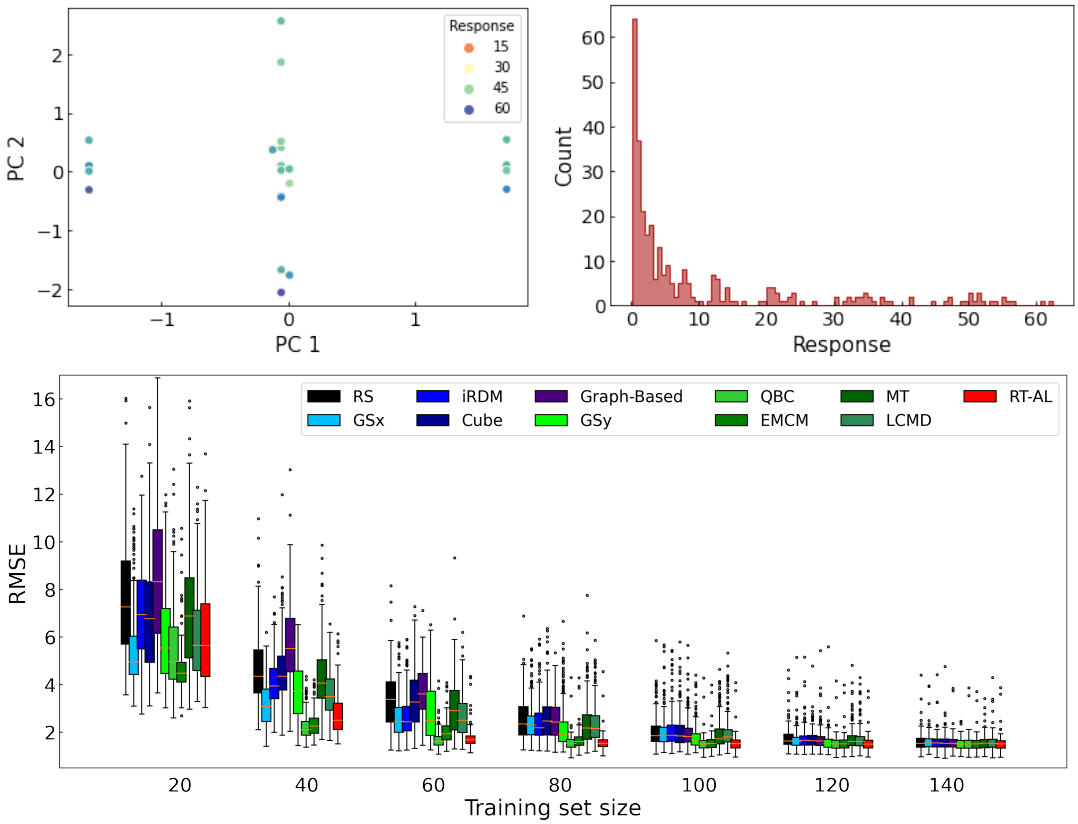

Figure 15: Two-component PCA and histogram of the response of the Yacht hydrodynamics dataset. Performance in prediction using boxplots over 200 runs when the training set is constructed using passive and AL approaches. The training set size varies from 20 to 140.

E Histograms of difference in RMSEs

In this section, we show the histograms of the error differences between our method and the state-of-the-art, to have a visual understanding of Table 2 of the main paper. With these histograms, we see that indeed our method is consistently a good performer for all the datasets. We also highlight that for cases where other approaches may give equivalent results compared to ours, the vales of the difference in errors is very low, thus not statistically significant. This can be seen from the peak at 0 on the X axis, suggesting that there are many such cases where other methods may be better than us but not statistically.

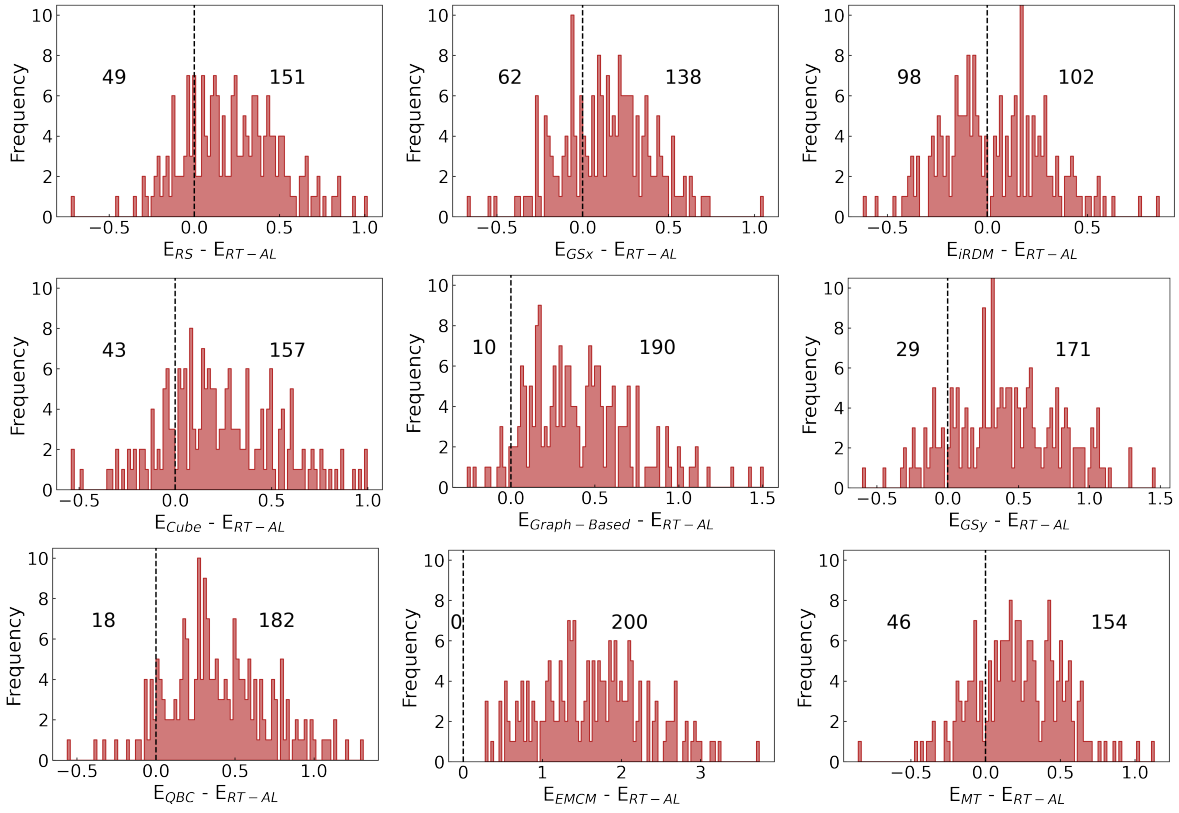

Figure 16: Histograms depicting difference in RMSE (E) of our approach compared to the state-of-the-art over a series of 200 experiments for the airfoil dataset, for 100 labeled samples.

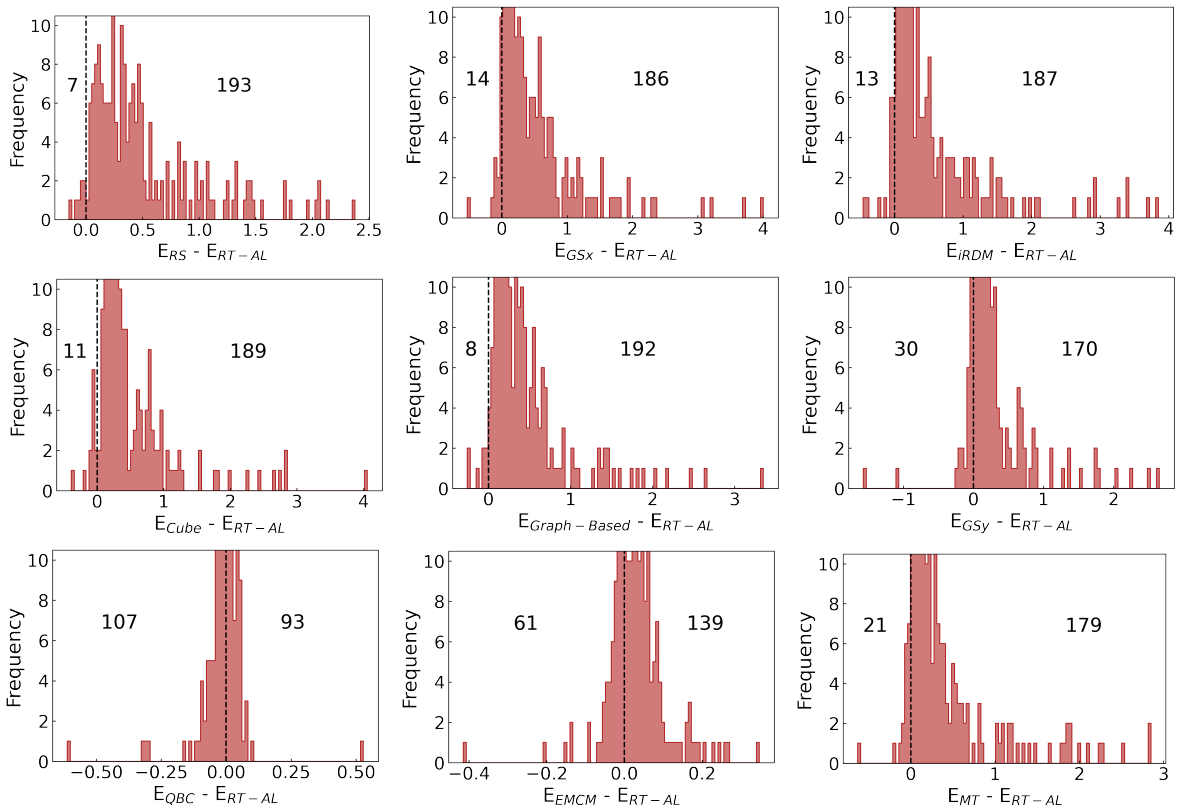

Figure 17: Histograms depicting difference in RMSE (E) of our approach compared to the state-of-the-art over a series of 200 experiments for the yacht dataset, for 100 labeled samples.

F RT-AL with other classes of machine learning models

We use three different models for the final predictor to show how our method can be applied to other models. We show in Fig. 5 the performance in prediction using RMSE averaged over 200 runs for different stages in training using the model-based AL methods GSy, QBC, EMCM, MT, LCMD and our method RT-AL, for the superconductivity dataset (with N = 21263, D = 81). RS was used to initialise all the methods (except for GSy where GSx is used as proposed by the authors). The different models we use for the final predictor with hyperparameter optimisation for each run are described below:

- Linear model Lasso: We use lasso model as implemented in Scikit-learn, while optimising alpha, and keeping the other hyperparameters as default.
- Random Forest: We used the random forest method (RF) as implemented in Scikit-learn, while optimising the number of estimators in the forest, maximum number of features and minimum samples in the leaf, and keeping the other hyperparameters as default.
- Multi Layer Perceptron Regressor: We use MLPRegressor (MLP) as implemented in Scikit-learn, while optimising hidden layer sizes, keeping the learning rate adaptive, maximum iterations 1000, setting early stopping to true and the other hyperparameters as default.

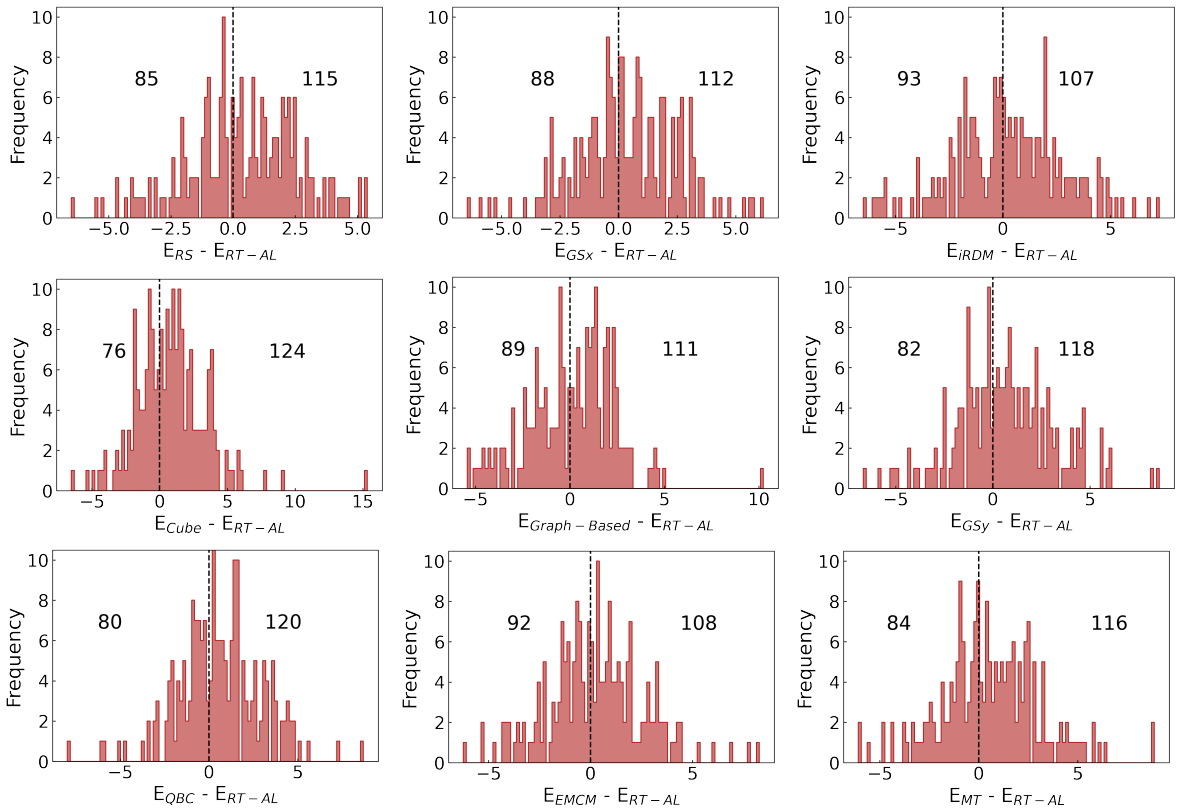

Figure 18: Histograms depicting difference in RMSE (E) of our approach compared to the state-of-the-art over a series of 200 experiments for the diabetes dataset, for 100 labeled samples.

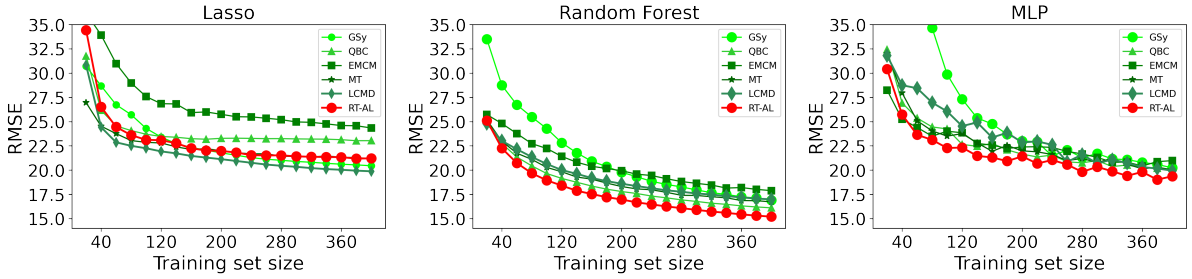

Figure 22: The performance in prediction using RMSE averaged over 200 runs for different stages in the training using model-based active learning methods for the superconductivity dataset (N = 21263, D = 81).

From the plots in Fig. 5 we can conclude that the performance of each model-based AL method varies based on the prediction model. However, our method with regression trees, RT-AL, is well positioned compared to other model-based AL algorithms irrespective of the prediction model, thus implying that RT-AL can indeed be applied to other models. Moreover, using random forest as the prediction model leads to the best performance among these three classes of models according to the RMSE score, thus justifying our choice for comparing all the methods. Thus, our method using regression trees is indeed very efficient and robust.

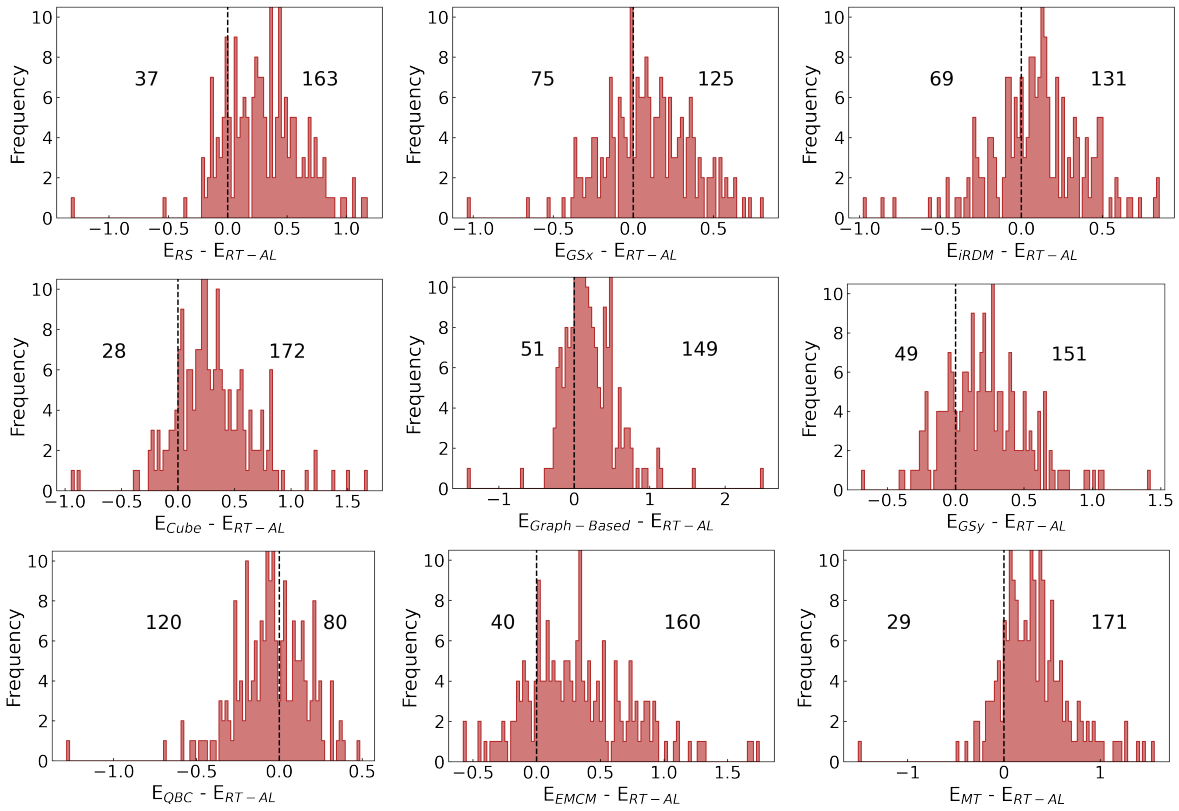

Figure 19: Histograms depicting difference in RMSE (E) of our approach compared to the state-of-the-art over a series of 200 experiments for the boston housing dataset, for 100 labeled samples.

G Computation time

We show in Table 4 the time (in seconds) it takes to label 100 samples for all the datasets (except for orange dataset, where we show the times for 60 labeled samples). Note that for our method, the methods corresponding to the italicised times have been used as initialiser for the respective datasets, while for other model-based methods, RS is the initialiser. We see that our method is indeed very competitive, and most often takes the lowest times, especially among the model-based AL methods. RT-AL is far more efficient than QBC (with trees as models in the committee) which gave a performance close to ours for some datasets. We also would like to mention here that for cases where labeling is expensive, therefore not many samples can be used to construct the training set, it is more crucial to have an accurate model and model complexity may not play a big role in general.

H Illustration on a simulated dataset

It has been shown in (**?**) that AL is not always interesting in regression, we try to answer why, and where it is indeed extremely necessary. We argue here that datasets with different distributions of the response and the features are the ones where AL is in fact most beneficial, and rather imperative when labeling is expensive. Keeping this idea in mind, we depict the importance of our method on a simulation that was generated particularly to have different structures for the features and the response. As most methods focus on diversity in the features, we start our illustration with a simulation so as to control the link between the response and the features. We construct a D-dimensional feature space of N samples, such that it consists

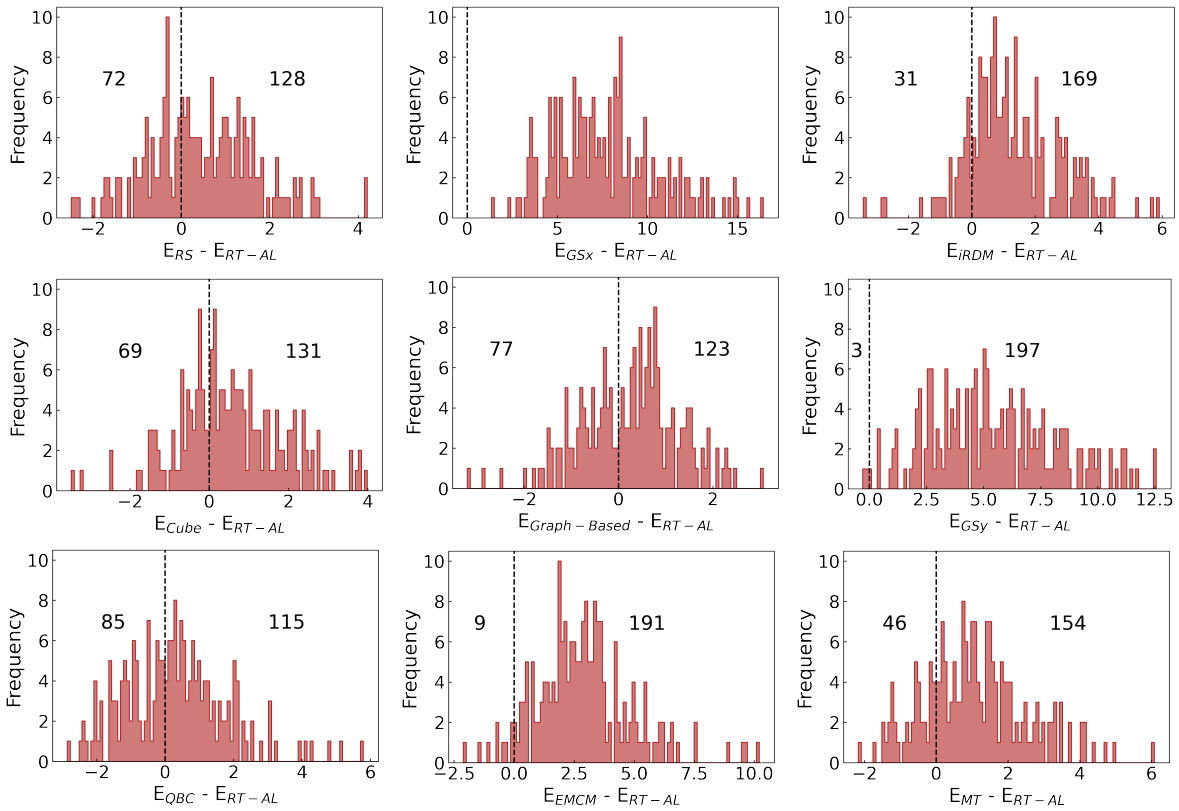

Figure 20: Histograms depicting difference in RMSE (E) of our approach compared to the state-of-the-art over a series of 200 experiments for the Superconductivity dataset, for 100 labeled samples.

of c clusters. The response is then defined as a non-linear function of the first principal component of the features, so as to keep a weak relation between the two distributions.

In Figure 24, we compare our approach to the RS, GSx and iRDM, for a simulation with $N = 3000$, $D = 15$ and $c = 10$ the number of clusters. As expected, GSx and iRDM both underperform compared to RS because for both these methods, we provide additional information about the features at every step, while missing critical details of the structure of the response. The comparison thus shows that for data with weak correlation between the features and the response, picking points to be labeled without using any prior information at all (using RS) is better than adding details about the features, because it focuses on the wrong subset of points.

However, we see that adding information about the response using our regression trees is indeed helpful. It is clear from the figure that for all the methods in question, we observe a significant improvement by picking the set of points to be labeled using a regression tree that is built on the samples that are already labeled. Further, we also note that GSx + RT (RS) or iRDM + RT (RS) is better for such datasets, than GSx + RT (Diversity-based) or iRDM + RT (Representativity-based). This is because both GSx and iRDM do not work well for such datasets to begin with, so the knowledge they add in each leaf is also damageable. Thus, for datasets such as these, where AL schemes are hoped to be most interesting, we succeed to show that learning with simple regression trees alone reduces the size of the training set to be labeled by a good margin, for a desired level of accuracy. For example, from Fig. 24 we see that on an average, the performance with 120 samples well selected by our trees is the same as the performance with 140 samples uniformly selected, 180 samples selected by iRDM and much more when selecting using GSx.

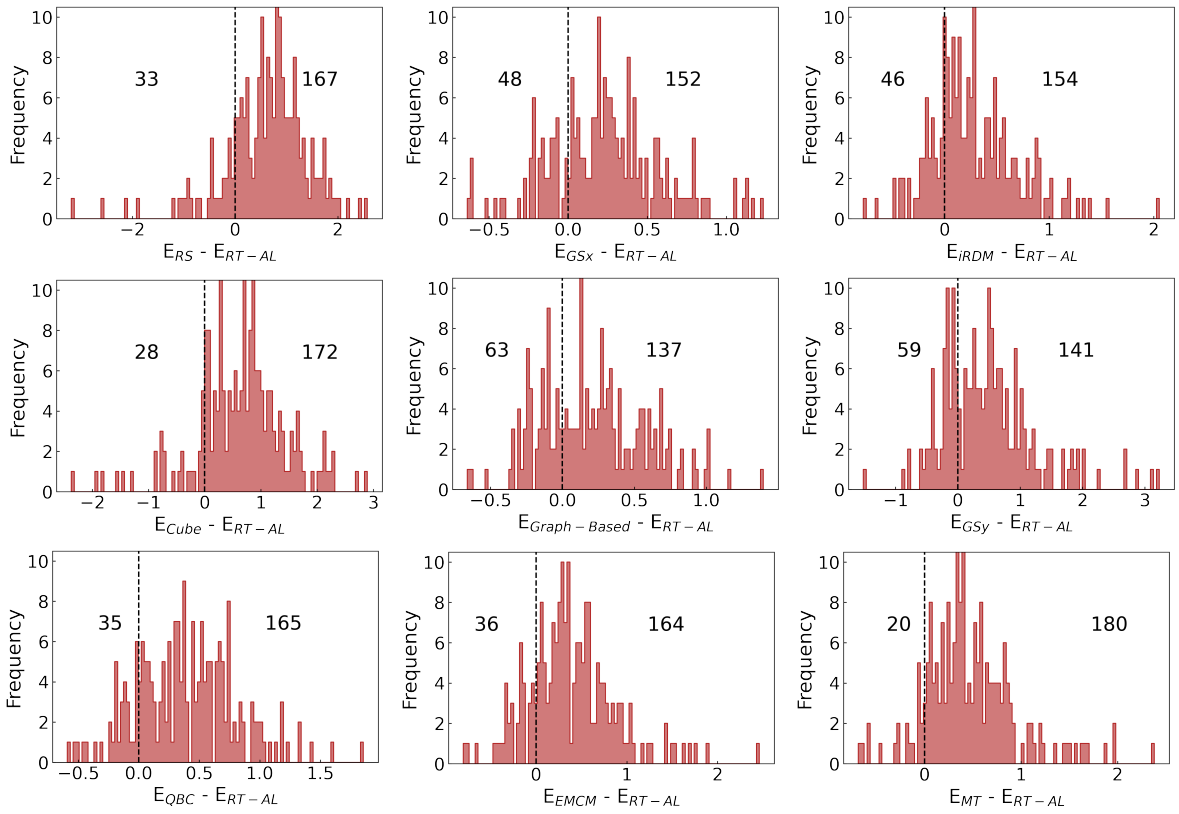

Figure 21: Histograms depicting difference in RMSE (E) of our approach compared to the state-of-the-art over a series of 200 experiments for the orange dataset, for 100 labeled samples.

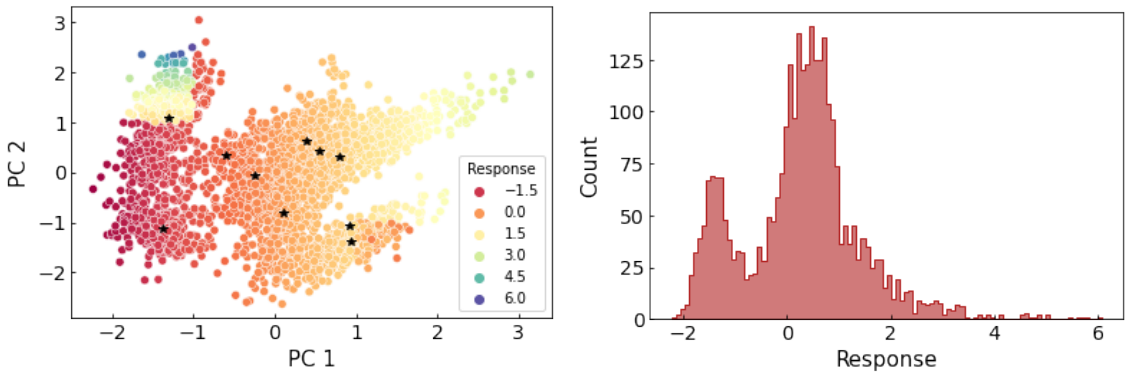

Figure 23: Two-component PCA and histogram of the response of the generated dataset.

Table 4: Mean computational time (in seconds) to label 100 samples (except for orange dataset where it is 60), for a series of 50 runs, for each dataset by column (N and D represent the total number of samples in the dataset and its dimension respectively). Their respective variance reported in brackets below.

	Airfoil	Yacht	Diabetes	Boston	SP	Orange
N	1503	308	442	506	21263	218
D	5	6	10	13	81	700
RS	0.433	*0.271*	0.314	0.404	*2.696*	2.325
	(0.001)	(0.011)	(0.004)	(0.003)	(0.543)	(0.360)
GSx	2.127	0.442	*0.849*	*1.020*	37.316	*2.485*
	(0.173)	(0.006)	(0.015)	(0.013)	(16.559)	(0.148)
iRDM	*16.148*	4.544	6.598	7.519	75.755	9.320
	(7.962)	(0.430)	(0.609)	(2.029)	(0.302)	(2.176)
Cube	0.335	0.321	0.406	0.268	30.591	2.400
	(0.083)	(0.038)	(0.082)	(0.002)	(58.525)	(0.376)
Graph-Based	1182.630	24.445	67.298	101.090	66171.045	18.003
	(29.068)	(0.005)	(0.130)	(0.301)	(364904.528)	(8.275)
GSy	4.410	0.935	1.513	1.781	167.867	2.530
	(0.593)	(0.061)	(0.004)	(0.031)	(1296.503)	(0.185)
QBC	47.370	7.920	12.335	16.211	148.480	7.851
	(35.921)	(1.980)	(2.690)	(2.931)	(0.845)	(0.546)
EMCM	2681.392	386.255	648.438	778.196	4074.811	224.986
	(129.618)	(8.058)	(6.226)	(29.610)	(522.883)	(4.926)
MT	0.385	0.273	0.283	0.376	3.133	1.714
	(0.011)	(0.007)	(0.003)	(0.002)	(0.089)	(0.044)
LCMD	2.019	1.340	1.703	2.193	79.920	3.346
	(0.213)	(0.249)	(0.146)	(0.182)	(296.485)	(0.510)
RT-AL	45.342	0.258	0.534	0.742	4.531	2.924
	(19.726)	(0.001)	(0.016)	(0.019)	(0.892)	(0.080)

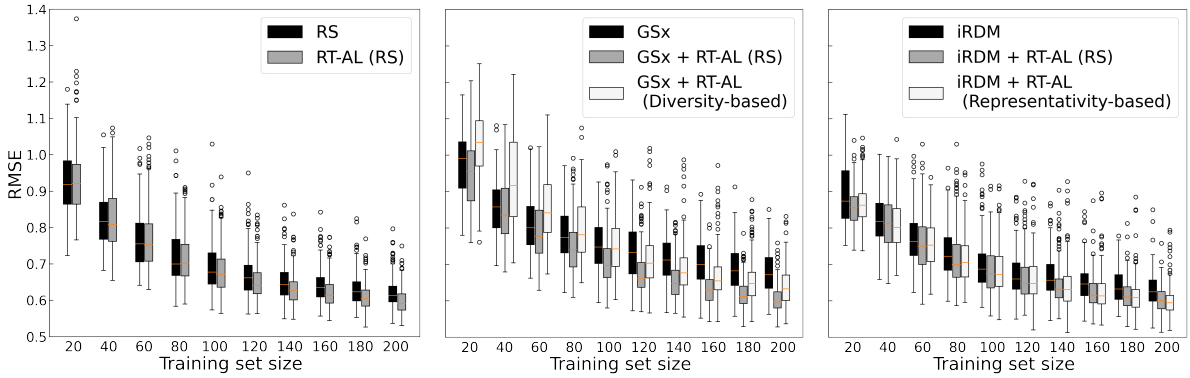

Figure 24: Performance in prediction using boxplots over 200 runs when the training set is constructed using passive/model-free AL or RT-AL (grey and white) on a generated dataset, using the query criteria mentioned in parenthesis for RT-AL. The training set size varies from 20 to 200.