
Variational Boosted Soft Trees

Tristan Cinquin

University of Tuebingen¹

tristan.cinquin@uni-tuebingen.de

Tammo Rukat

Deekard¹

tammorukat@gmail.com

Philipp Schmidt

Amazon

phschmid@amazon.com

Martin Wistuba

Amazon

marwistu@amazon.com

Artur Bekasov

Amazon

abksv@amazon.com

Abstract

Gradient boosting machines (GBMs) based on decision trees consistently demonstrate state-of-the-art results on regression and classification tasks with tabular data, often outperforming deep neural networks. However, these models do not provide well-calibrated predictive uncertainties, which prevents their use for decision making in high-risk applications. The Bayesian treatment is known to improve predictive uncertainty calibration, but previously proposed Bayesian GBM methods are either computationally expensive, or resort to crude approximations. Variational inference is often used to implement Bayesian neural networks, but is difficult to apply to GBMs, because the decision trees used as weak learners are non-differentiable. In this paper, we propose to implement Bayesian GBMs using variational inference with *soft* decision trees, a fully differentiable alternative to standard decision trees introduced by Irsoy et al. Our experiments demonstrate that variational soft trees and variational soft GBMs provide useful uncertainty estimates, while retaining good predictive performance. The proposed models show higher test likelihoods when compared to the state-of-the-art Bayesian GBMs in 7/10 tabular regression datasets and improved out-of-distribution detection in 5/10 datasets.

¹Work done while at Amazon.

1 INTRODUCTION

Tabular data, often combining both real valued and categorical variables, is common in machine learning applications (Borisov et al., 2021; Hoang and Wiegatz, 2021; Clements et al., 2020; Ebbehøj et al., 2021; Mustafa and Rahimi Azghadi, 2021). A number of these applications require uncertainty estimates, in addition to the model’s predictions. Predictive uncertainty is particularly important when using the model for making decisions associated with risk, as common in healthcare (Abdullah et al., 2022; Kompa et al., 2021) or finance (Bew et al., 2019; Gonzalez et al., 2019). In order to make optimal decisions, we require the uncertainty estimates to be *well-calibrated*: the predictive distribution returned by the model should capture the true likelihood of the targets (Guo et al., 2017).

A *decision tree* is a predictive model that is fit by recursively partitioning the feature space. These models are fast, interpretable and support categorical data without additional pre-processing. To increase their expressivity without exacerbating overfitting, multiple decision trees can be combined into an ensemble using *gradient boosting* (Freund and Schapire, 1997; Friedman, 2001). *Gradient boosting machines* (GBM) are an expressive class of machine learning models trained by sequentially fitting an ensemble of weak learners (e.g. decision trees). At each iteration, the gradient boosting procedure fits an additional weak learner that would minimize the training error when added to the ensemble. These models demonstrate state-of-the-art results on tabular data (Bojer and Meldgaard, 2021; Liu et al., 2021), often out-performing deep neural networks (Borisov et al., 2021; Shwartz-Ziv and Armon, 2022).

While GBM models demonstrate good predictive performance, they do not provide well-calibrated predictive uncertainties (Niculescu-Mizil and Caruana, 2012). Bayesian methods are known to improve predictive uncertainties by explicitly capturing the *epistemic* uncertainty, a type of uncertainty that results from learning with finite data (Kristiadi

et al., 2020; Mitros and Namee, 2019). Bayesian GBMs have been proposed, and indeed improve upon the predictive uncertainties of standard GBMs (Chipman et al., 1998; Linero and Yang, 2017; Ustimenko and Prokhorenkova, 2020). Current methods rely on *Markov chain Monte Carlo* (MCMC) to sample from the posterior, however, and do not scale well to larger models and datasets. Alternative approximate sampling methods have been proposed by He and Hahn (2020) and Malinin et al. (2020). These methods make the sampling procedure more efficient, but degrade the quality of the resulting predictive uncertainties. More scalable approximate inference methods such as the Laplace approximation or variational inference are difficult to apply to GBMs, because the decision trees used as weak learners are non-differentiable.

Unlike standard decision trees that route the input to a unique leaf based on a series of binary conditions, *soft decision trees* (Irsoy et al., 2012) compute a convex combination of *all* leaves, weighing their values by a series of learned gating functions. As a result, soft decision trees are fully differentiable, and therefore amenable to scalable Bayesian inference methods that are common in Bayesian deep learning (Blundell et al., 2015; Cobb and Jalaian, 2020; Immer et al., 2020).

In this work we propose to implement Bayesian GBMs by using soft trees as weak learners in a GBM, and performing variational inference on the resulting model. In particular, we make the following contributions:

1. We propose a method for performing variational inference in soft decision trees. The chosen variational distribution allows trading-off memory and computation for a richer posterior approximation.
2. We increase the expressivity of the model by using variational soft trees as weak learners in a GBM.
3. We run experiments to demonstrate that the proposed models perform well on tabular data, yielding useful predictive uncertainties in regression, out-of-distribution detection and contextual bandits.

2 BACKGROUND

2.1 Bayesian inference

Bayesian inference provides a theoretical framework for reasoning about model uncertainty, and has been shown to improve calibration of machine learning models in practice (Mitros and Namee, 2019; Kristiadi et al., 2020). In Bayesian inference we place a prior on the model parameters θ , and use the Bayes rule to define the posterior distribution having observed the data:

$$p(\theta | \mathcal{D}) \propto p(\mathcal{D} | \theta)p(\theta) \quad (1)$$

where $\mathcal{D} = \{x_i, y_i\}_{i=1}^n$ is the training data, $p(\theta)$ is the prior distribution, and $p(\mathcal{D} | \theta)$ the likelihood. The predictive distribution of a new target y^* given a feature vector x^* is then computed as

$$p(y^* | x^*) = \mathbb{E}_{\theta \sim p(\theta | \mathcal{D})} [p(y^* | x^*, \theta)] \quad (2)$$

The posterior distribution in the Bayesian paradigm explicitly captures *epistemic* or *model* uncertainty that results from learning with finite data. Epistemic uncertainty is one of the two fundamental types of uncertainty (alongside the *aleatoric* or *data* uncertainty), and capturing it was shown to be especially important in risk-aware decision making (Riquelme et al., 2018; Abdullah et al., 2022; Deisenroth and Rasmussen, 2011), active learning (Kirsch et al., 2019) and Bayesian optimisation (Gonzalez et al., 2019; Krause et al., 2006; Springenberg et al., 2016). Unfortunately, the posterior distribution is typically intractable and requires numerical approximation.

Variational inference Variational inference (VI) is an approximate inference method which selects a distribution q^* within a variational family \mathcal{Q} that best approximates the posterior p . More formally, we select the distribution q^* such that:

$$\begin{aligned} q^* &= \operatorname{argmin}_{q \in \mathcal{Q}} D_{\text{KL}}(q || p) \\ &= \operatorname{argmin}_{q \in \mathcal{Q}} \mathbb{E}_{\theta \sim q} \left[\log \frac{q(\theta)}{p(\theta | \mathcal{D})} \right]. \end{aligned} \quad (3)$$

In practice, we define a parametric variational distribution q_ϕ and use standard optimization methods like stochastic gradient descent to minimize the KL loss w.r.t. ϕ . We then use $q^*(\theta)$ instead of $p(\theta | \mathcal{D})$ to approximate the predictive distribution in Equation (2). In this paper, we focus on VI because it can capture arbitrarily complex, high-dimensional posteriors, assuming the chosen variational family is sufficiently expressive. At the same time, VI can scale to large datasets when combined with stochastic optimisation (Hoffman et al., 2013; Swiatkowski et al., 2020; Tomczak et al., 2020; Tran et al., 2015).

Other commonly used approximate inference methods are Markov chain Monte Carlo (MCMC) (Welling and Teh, 2011) and the Laplace approximation (Immer et al., 2020). MCMC samples from the posterior directly, but does not scale well to large models and datasets. The Laplace approximation is more scalable, but assumes that the posterior can be approximated well with a Gaussian distribution.

2.2 Decision trees

Decision trees are predictive models trained by recursively partitioning the input space and fitting a simple model (often constant) in each partition (leaf). At each non-terminal node of the tree, the model forwards the input to a child

based on a binary condition. For example, at a particular node the model might route the input x to the left sub-tree if $x[i] \leq \tau$, and to the right sub-tree otherwise, where i indexes a particular feature and τ is a threshold. The splits are determined by greedily optimizing a chosen criterion at each node. For instance, in the regression setting the feature index i and threshold τ could be selected to minimize the *sum of square errors* of the induced partition. The splitting hyperplanes are always parallel to the axes of the feature space. Predictions are made by forwarding x to a unique leaf and outputting the class (for classification) or the real value (for regression) associated with this particular leaf.

Soft decision trees First introduced by Irsoy et al. (2012), soft decision trees use sigmoid gating functions to “soften” the binary routing at each node. More formally, the output $f(x)$ of a soft decision tree is computed as the sum of the leaf outputs $f_l(x)$ weighted by the probability $p_l(x)$ that the input x is routed to the leaf l :

$$f(x) = \sum_{l \in L} p_l(x) f_l(x), \quad (4)$$

where L is the set of leaf nodes. The probability $p_l(x)$ is defined as the product of probabilities returned by the *soft gating functions* on the path from the root to the leaf l . Details on the exact form of p_l and f_l are provided in Section 4.1. The sigmoid gating functions allow learning non-axis-aligned decision boundaries and smooth functions. This results in a differentiable model that can be trained by gradient descent, and yet performs well on tabular data (Feng et al., 2020; Irsoy et al., 2012; Linero and Yang, 2017; Luo et al., 2021). Computing the prediction of a soft tree requires evaluating *all* the leaves, and is therefore more expensive than doing so for the standard decision tree. In practice, however, a shallow soft decision tree can match the performance of a deep hard decision tree, which alleviates some of the computational overhead (Irsoy et al., 2012).

Multiple variations of a soft decision tree exist. The tree depth can be fixed (Feng et al., 2020; Kotschieder et al., 2015; Luo et al., 2021) or adjusted dynamically by greedily adding new nodes while the validation loss decreases (Irsoy et al., 2012). Furthermore, the sigmoid gating functions can be based on linear models (Feng et al., 2020; Frosst and Hinton, 2017; Irsoy et al., 2012; Luo et al., 2021), multi-layer perceptrons (Feng et al., 2020; Kotschieder et al., 2015), or convolutional neural networks (Ahmetoğlu et al., 2018). Likewise, the leaves of the tree can output constant values (Feng et al., 2020; Frosst and Hinton, 2017; Kotschieder et al., 2015; Luo et al., 2021), or be defined using complex models such as neural networks (Ahmetoğlu et al., 2018). To address the exponential increase in prediction cost as the soft tree grows, Frosst and Hinton (2017) propose to only consider the *most probable* path from the root to a leaf. This method requires special regularization encouraging nodes to make equal use of each sub-tree to work and performs

worse than averaging over all leaves, but is more efficient.

In addition to state-of-the-art performance on tabular data, soft decision trees can be preferred to neural networks due to their greater interpretability (Frosst and Hinton, 2017). As soft trees rely on hierarchical decisions, one can examine the weights of the soft gating functions and gain insight into the learned function. In prior work, neural networks have been distilled into soft trees to make their decisions explainable, both in the context of image classification (Frosst and Hinton, 2017) and reinforcement learning (Coppens et al., 2019).

Soft decision trees are similar to *hierarchical mixtures of experts* (HME, Jordan and Jacobs, 1993). While HMEs and soft decision trees share the same architecture, HMEs explicitly model the assignment of each input to a leaf using latent variables. Therefore, HMEs are trained using the expectation maximisation algorithm (EM, Dempster et al., 1977), rather than by maximising the log-likelihood. Jordan and Jacobs find that, when compared to the standard maximum likelihood with backpropagation, fitting an HME using EM requires fewer passes over the data, but the resulting models demonstrate higher test error.

2.3 Gradient boosting machines

A *gradient boosting machine* (GBM) (Friedman, 2001) sequentially combines a set of weak learners (e.g. decision trees) into an ensemble to obtain a stronger learner. At each iteration, a weak learner is trained to minimize the error of the existing ensemble, and is subsequently added to the ensemble. More formally, at iteration t the GBM combines the existing ensemble $f_t(x) = \sum_{j=1}^t \gamma_j h_j(x)$ with an additional learner $h_{t+1} \in \mathcal{H}$ from some function class \mathcal{H} s.t. $f_{t+1}(x) = f_t(x) + \gamma_{t+1} h_{t+1}(x)$ minimizes the training loss. Finding the optimal weak learner h_{t+1} is intractable in general, so the optimization problem is simplified by taking a step in the direction of the steepest decrease of the loss \mathcal{L} in function space:

$$f_{t+1}(x) = f_t(x) - \gamma_{t+1} \sum_{i=1}^n \nabla_{f_t} \mathcal{L}(f_t(x_i), y_i), \quad (5)$$

where $\mathcal{D} = \{x_i, y_i\}_{i=1}^n$ is the training data. In other words, each weak learner h_{t+1} is fit to match $-\gamma_{t+1} \sum_{i=1}^n \nabla_{f_t} \mathcal{L}(f_t(x_i), y_i)$. Taking multiple gradient boosting steps minimizes the training loss and yields an accurate model. GBMs demonstrate state-of-the-art performance on tabular data for both classification and regression tasks: models that win Kaggle¹ competitions are commonly based on GBMs (Bojer and Meldgaard, 2021; Liu et al., 2021).

¹<https://www.kaggle.com>

3 RELATED WORK

Bayesian hierarchical mixture of experts Prior work performed Bayesian inference on the parameters of HMEs using variational inference. These methods consider an approximate posterior which factorizes the parameters into groups (leaf function, node gating function, latent variables, prior parameters) and places isotropic Gaussian priors on the leaf and node model parameters, as well as Gamma hyper-priors. Waterhouse et al. (1995) perform inference by iteratively optimizing the VI objective with respect to each group of parameters separately, keeping the others fixed. However, for tractability they use the Laplace approximation on the node gating function parameters and hence do not get a lower bound on the evidence. Ueda and Ghahramani (2002) propose to model the joint distribution over both input and output variables, which allows to derive the variational posterior without approximations for a tree of height 1. However, capturing the input distribution is both expensive and unnecessary for regression and classification. Also this method does not apply to deeper trees. Finally, Bishop and Svensen (2012) derive a tractable lower bound on the variational inference objective, which is then optimised by iteratively updating the parameters of each factor of the variational posterior, keeping the others fixed.

In contrast, we consider a simpler model which does not explicitly capture the assignment of data samples to leaves with latent variables. Furthermore, our proposed variational posterior is more expressive and can capture correlations between all the parameters of the model. Finally, we fit our model to optimise the variational objective using the reparameterization trick from Kingma and Welling (2013) and stochastic gradient descent.

Bayesian gradient boosted trees Previous work on Bayesian GBMs uses MCMC to sample from the Bayesian posterior. Chipman et al. (1998) suggest performing inference in a GBM using a Gibbs sampler which iteratively samples each tree conditioned on the other trees in the ensemble. This method, *BART*, requires a long burn-in period (≈ 1000 iterations). Pratola et al. (2017) improve *BART* with a heteroscedastic noise model at the cost of making inference slower, as it is performed on both the mean and noise models. In order to speed up the convergence of the *BART* sampler, He and Hahn (2020) propose *Accelerated BART* (*XBART*). Rather than making small changes to a given tree at each iteration, *XBART* grows an entirely new tree. This speeds up mixing considerably requiring only 15 burn-in iterations. As a mechanism for variable selection, the authors also propose placing a Dirichlet prior on the input features, improving performance on high-dimensional data.

In parallel, Ustimenko and Prokhorenkova (2020) draw analogies between gradient boosting and MCMC, and prove that a set of modifications is enough to cast gradient boosting

with Gaussian noise as stochastic gradient Langevin dynamics in function space. The resulting algorithm, *stochastic gradient Langevin boosting* (*SGLB*), allows to draw multiple samples from the posterior by training an ensemble of GBMs with targets perturbed by Gaussian noise. To reduce the memory and time complexity of *SGLB*, Malinin et al. (2020) introduce *virtual ensembles*. Taking advantage of the additive structure of GBMs, virtual ensembles collect multiple sub-ensembles by running a single *SGLB* model. However, faster training comes at the cost of higher correlation among models, thus reducing the effective sample size.

Soft gradient boosting machine Just like decision trees, soft decision trees can be combined into ensembles to improve their predictive performance. Gradient boosting is a particularly successful method for doing so, hence Luo et al. (2021) experiment with combining soft trees using gradient boosting, and find the resulting method to be highly effective on tabular data. Similarly, Feng et al. (2020) suggest to fit soft GBMs by directly minimizing the gradient boosting loss using stochastic gradient descent. As each weak learner is updated in parallel, this method diverges significantly from the gradient boosting framework.

An alternative line of research aims to arrive at a Bayesian soft GBM by making *BART* “smoother”. Linero and Yang (2017) propose *soft BART* (*SBART*), a model which replaces the standard decision trees in *BART* with soft trees. Empirically, fitting the resulting method is significantly slower than *BART*. Maia et al. (2022) suggest putting Gaussian process priors on each leaf in a *BART* model, demonstrating that this method performs better than *BART* and *SBART*, but is also slower.

4 METHODS

4.1 Variational soft trees

In this section we propose a *variational soft tree* (*VST*), a model that has the same architecture as a fixed depth soft decision tree (Figure 1), but where we perform variational inference to approximate the full posterior distribution. We then sample from the posterior approximation to compute the predictive distribution in Equation (2) by Monte Carlo integration. We consider two variations of the variational soft tree: one with *constant leaves* as in (Feng et al., 2020; Frosst and Hinton, 2017; Kotschieder et al., 2015; Luo et al., 2021), and another with *linear leaves*, where each leaf outputs an affine function of the input. Linear leaves allow the model to represent piecewise linear functions, which are strictly more expressive than piecewise constant functions. In fact, a piecewise linear function can represent *any* differentiable scalar function in the limit of infinite pieces by setting each piece to the derivative of the function. Linear leaves use two additional parameter vectors per leaf, and in

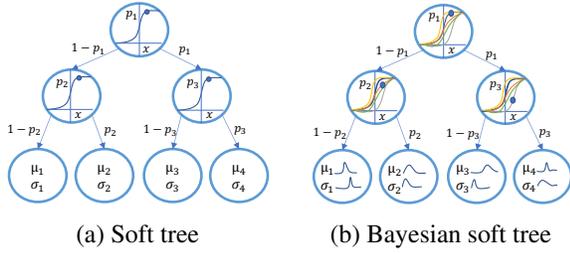


Figure 1: Standard vs. Bayesian soft decision tree. In a standard soft tree each weight is assigned a fixed value, while in a Bayesian soft tree each weight is a distribution.

our experiments had a marginal effect on the computation cost. To our knowledge, we are the first to consider soft decision trees with linear leaves.

Likelihood We define the likelihood of the soft tree as

$$p(y|x, \theta) = \sum_{l \in L} P(l|x, \phi) p(y|x, l, \psi), \quad (6)$$

where L is the set of tree leaves (terminal nodes) and $\theta = \{\phi, \psi\}$ are the parameters of the soft tree.

$P(l|x, \phi)$, the first term inside the sum in Equation (6), is the probability that the input x is routed to the leaf l , defined as:

$$P(l|x, \phi) = \prod_{n \in \text{Path}(l)} \left[\frac{P(r|n, x, \phi)^{n \searrow l}}{P(r|n, x, \phi)^{l \swarrow n} + P(r|n, x, \phi)^{n \searrow l}} \right], \quad (7)$$

where $\text{Path}(l)$ the set of tree nodes on the path from the root to the leaf l , $n \searrow l = 1$ if the leaf l is in the *right* subtree of node n , $l \swarrow n = 1$ if it is in the *left* subtree, and $P(r|n, x, \phi)$ is the probability that x is routed to the right subtree at the node n , with $P(\neg r|n, x, \phi) = 1 - P(r|n, x, \phi)$. We parametrize $P(r|n, x, \phi)$ using a *soft gating function* f_ϕ :

$$P(r|n, x, \phi) = f_\phi(x, n), \quad (8)$$

restricting $f_\phi(x, n) \in [0, 1]$ for all x and n . In this work, we choose a simple linear gating function (Feng et al., 2020; Frosst and Hinton, 2017; Irsoy et al., 2012; Luo et al., 2021):

$$f_\phi(x, n) = \sigma(\beta(w_n^T x + b_n)), \quad (9)$$

where σ is the standard sigmoid function, β is an inverse temperature parameter used to regularize the function during training (Frosst and Hinton, 2017), and $\phi = \{(w_n, b_n)\}_{n \in N}$ are the parameters of the function for the set of tree nodes N .

The second term inside the sum in Equation (6), $p(y|x, l, \psi)$, is the predictive probability when the input x is routed to the leaf l . In this work, we consider two variations of this term: a *constant leaf* model, and a *linear leaf*

model. In the constant leaf model, the output at each leaf is independent of the input:

$$p(y|x, l, \psi) = p(y|l, \psi) \quad (10)$$

$$= \mathcal{N}(y|\mu_l, \text{softplus}(\alpha_l)^2), \quad (11)$$

where $\psi = \{(\mu_l, \alpha_l)\}_{l \in L}$. In the linear leaf model, on the other hand, the parameters of the Gaussian are linear in the input:

$$p(y|x, l, \psi) = \mathcal{N}(y|\mu_l(x), \sigma_l^2(x)) \quad (12)$$

$$\mu_l(x) = w_l^T x + b_l \quad (13)$$

$$\sigma_l(x) = \text{softplus}(\hat{w}_l^T x + \hat{b}_l) \quad (14)$$

where $\psi = \{(w_l, b_l, \hat{w}_l, \hat{b}_l)\}_{l \in L}$ and $\text{softplus}(x) = \log(1 + \exp(x))$. We use softplus functions to constrain the standard deviation parameters to remain strictly positive, while ensuring numerical stability (Blundell et al., 2015).

Prior We use a spherical, zero centered Gaussian prior for the model parameters:

$$p(\theta) = \mathcal{N}(\theta|0, \tau^2 \mathbb{I}). \quad (15)$$

Note that for a constant leaf model we put this prior on the inverse softplus of the standard deviation at each leaf, which restricts the variance to stay strictly positive.

Variational posterior We use the variational distribution proposed by Tomczak et al. (2020), which is a Normal distribution with a low-rank covariance matrix:

$$q(\theta) = \mathcal{N}(\theta|\mu, \text{diag}[\sigma^2] + VV^T) \quad (16)$$

where $\sigma \in \mathbb{R}^p$ and V is a matrix of rank k . The hyperparameter k determines the rank of the covariance matrix, and hence the expressivity of the posterior. If $\theta \in \mathbb{R}^p$, choosing $k = p$ allows learning arbitrary covariance matrices, but increases the memory requirements and computational cost. As a result, in our experiments we typically choose $k < p$. This choice of the variational posterior and the prior admits an analytical KL-divergence which is detailed in Section 1.1 in the supplementary material.

4.2 Gradient boosted variational soft trees

In order to increase the expressivity of the variational soft tree, we propose to use these models as weak learners in a gradient boosting machine that we refer to as a *variational soft GBM* (VSGBM). More formally, the model is defined as

$$F_\theta(x) = \sum_{t=1}^T f_{\theta_t}(x) + \epsilon \quad \text{where} \quad \epsilon \sim \mathcal{N}(\epsilon|0, \sigma^2) \quad (17)$$

Algorithm 1: Gradient boosted variational soft trees

Input: Prior parameters a_σ and b_σ
 T the number of soft trees in the GBM

Data: $\{X, y\}$

 Fit variational soft tree f_{θ_1} to $\{X, y\}$
for $t \in [2, \dots, T]$ **do**

 Sample parameters for all models $f_{\theta_j} : \theta_j \sim q_j$ for

 $j = 1 \dots t - 1$

Compute the residual vector:

$$r_t = y - \sum_{j=1}^{t-1} f_{\theta_j}(X)$$

 Fit the new model f_{θ_t} to $\{X, r_t\}$
end

 Sample parameters $\theta = \{\theta_j\}_{j=1}^T$ where $\theta_j \sim q_j$

 Compute final residuals $r = y - F_\theta(X)$

 Sample $\sigma^2 \sim \text{inverse-Gamma}(a_\sigma + n, b_\sigma + r^T r)$

where f_{θ_t} is a variational soft tree with parameters θ_t , T is the number of trees in the ensemble and σ is a scale parameter. We consider variational soft trees f_{θ_t} that output the estimated mean value of the target and no standard deviation. Hence the variational soft GBM has a homoskedastic (input-independent) variance model. Heteroskedastic (input-dependent) uncertainty could in principle be modeled by returning the predictive variances from the weak learners alongside the predictive means, and combining them upstream. In our early experiments this configuration proved to be costly to train (it doubles the number of model parameters), and made the training unstable (the gradient boosting loss had high variance). In prior work Pratola et al. (2017) train a separate ensemble to capture aleatoric uncertainty, likely to avoid this instability.

Inspired by He and Hahn (2020), we place an inverse-Gamma(a_σ, b_σ) prior on σ^2 , yielding an inverse-Gamma($a_\sigma + n, b_\sigma + r^T r$) posterior, where n is the number of data samples and $r = y - F_\theta(X)$ is the residual vector. The variational soft GBM is fit following the procedure described in Algorithm 1. Once trained, samples from the approximate posterior of the variational soft GBM are obtained by drawing parameters from the variational distribution of each variational soft tree model in the ensemble. The predictive distribution is then approximated by Monte Carlo integration of the likelihood as per Equation (2), i.e. by aggregating the predictions of the variational soft trees as defined by Equation (17) for different posterior samples.

5 EXPERIMENTS

In this section we evaluate the variational soft tree and variational soft GBM on regression, out-of-distribution detection and multi-armed bandits. Although we are not measuring model calibration directly, like prior work we assume that model calibration and performance on these tasks are corre-

lated. When reporting results, we bold the highest score, as well as any score if its error bar and the highest score’s error bar overlap, considering the difference to not be statistically significant.

5.1 Regression

We consider a suite of regression datasets used by Malinin et al. (2020) described in Table 4 in supplementary material. We perform 10-fold cross validation on the training data and report the mean and standard deviation of the test log-likelihood and root mean square error (RMSE) of our models across the folds. We compare our variational soft decision trees (VST) and variational soft GBM (VSGBM) with two Bayesian GBMs (XBART (He and Hahn, 2020) and SGLB (Ustimenko and Prokhorenkova, 2020)), an ensemble of neural networks, and XGBoost (Chen and Guestrin, 2016). When comparing models to VSTs, we consider baselines with an ensemble size of one, and otherwise multiple ensemble members when comparing with the VSGBM. The only exception is the ensemble of deep neural networks for which we always use an ensemble of ten models to measure epistemic uncertainty.

Variational soft decision tree We find that our model performs well in terms of test log-likelihood and RMSE across the considered datasets (see Tables 1 and 6), which confirms that soft decision trees are well-suited to tabular data (Luo et al., 2021). VSTs outperform the baselines on 7/10 datasets in terms of log-likelihood, and 3/10 datasets in terms of RMSE, also matching the log-likelihood and RMSE of best performing baselines on 1/10 and 3/10 datasets respectively. In addition, soft decision trees with linear leaves perform better than equivalent constant leaf models on 8/10 datasets in terms of log-likelihood and 5/10 in terms of RMSE, bringing evidence that linear leaf models are more expressive than constant leaf models.

VSTs are less competitive when trained on small datasets like *boston* (506 training examples) and *yacht* (308 training examples). We hypothesize that in these cases the model underfits the data due to the choice of the prior. The parameters of the prior were chosen by hyper-parameter tuning with the validation ELBO objective, and the underfitting behavior could be an artifact of this tuning procedure. An alternative hypothesis is that a normal prior is simply ill-suited to this model. Choosing priors for complex, deep models is difficult, and is an active research direction in Bayesian deep learning (Flam-Shepherd, 2017; Fortuin et al., 2021; Krishnan et al., 2020; Vladimirova et al., 2018).

Variational soft GBM VSGBMs perform competitively in terms of log-likelihood compared to the baselines (see Tables 1 and 6), yielding the best log-likelihood on 7/10 datasets. The VSGBMs perform on par with baselines on *power* and *boston*, worse on *yacht* but better on the other

Table 1: Test log-likelihood of evaluated methods on regression datasets. We find that variational soft decision trees (VST) and variational soft GBMs (VSGBM) perform strongly compared to baselines in terms of test log-likelihood, both methods out-performing them on 7/10 datasets.

MODEL	POWER	PROTEIN	BOSTON	NAVAL	YACHT	SONG	CONCRETE	ENERGY	KIN8NM	WINE
DEEP ENSEMBLE	-1.404 ± 0.001	-1.422 ± 0.000	-1.259 ± 0.007	-1.412 ± 0.000	-1.112 ± 0.009	-1.415 ± 0.001	-1.390 ± 0.001	-1.677 ± 0.073	-1.416 ± 0.000	-1.725 ± 0.075
XBART (1 TREE)	-0.096 ± 0.003	-1.136 ± 0.003	-0.508 ± 0.042	-0.485 ± 0.419	1.110 ± 0.021	-1.151 ± 0.006	-0.793 ± 0.079	-0.848 ± 0.567	-1.153 ± 0.014	-1.250 ± 0.008
SGLB (1 TREE)	-1.425 ± 0.027	-1.499 ± 0.002	-1.361 ± 0.044	-1.410 ± 0.009	-0.891 ± 0.047	-1.305 ± 0.007	-1.748 ± 0.050	-1.078 ± 0.006	-1.659 ± 0.015	-1.559 ± 0.033
VST (CONST.)	-0.032 ± 0.010	-0.719 ± 0.005	-0.920 ± 0.002	-0.555 ± 0.088	0.193 ± 0.024	-0.733 ± 0.010	-0.999 ± 0.006	-0.710 ± 0.280	-0.877 ± 0.014	1.458 ± 0.615
VST (LINEAR)	0.024 ± 0.002	-0.631 ± 0.011	-0.682 ± 0.004	-0.335 ± 0.026	-0.754 ± 0.005	-0.579 ± 0.002	-0.461 ± 0.070	0.377 ± 0.123	-0.707 ± 0.012	3.067 ± 1.168
XBART	-0.001 ± 0.006	-1.136 ± 0.003	-0.462 ± 0.054	0.008 ± 0.130	1.116 ± 0.023	-1.018 ± 0.002	-0.581 ± 0.024	-0.462 ± 0.054	-1.057 ± 0.007	-1.251 ± 0.010
SGLB	-0.500 ± 0.001	-1.204 ± 0.002	-0.504 ± 0.013	-0.877 ± 0.004	-0.018 ± 0.009	-0.907 ± 0.001	-0.931 ± 0.010	-0.232 ± 0.009	-1.125 ± 0.002	-1.220 ± 0.006
VSGBM (CONST.)	-0.002 ± 0.011	-1.039 ± 0.005	-0.567 ± 0.062	0.573 ± 0.311	0.045 ± 0.912	-0.933 ± 0.008	-0.465 ± 0.040	-0.227 ± 0.038	-0.457 ± 0.021	-1.192 ± 0.007
VSGBM (LINEAR)	-0.015 ± 0.011	-1.051 ± 0.012	-0.462 ± 0.054	0.542 ± 0.295	-0.033 ± 0.049	-0.729 ± 0.019	-0.591 ± 0.079	0.057 ± 0.134	-0.363 ± 0.031	-1.167 ± 0.004

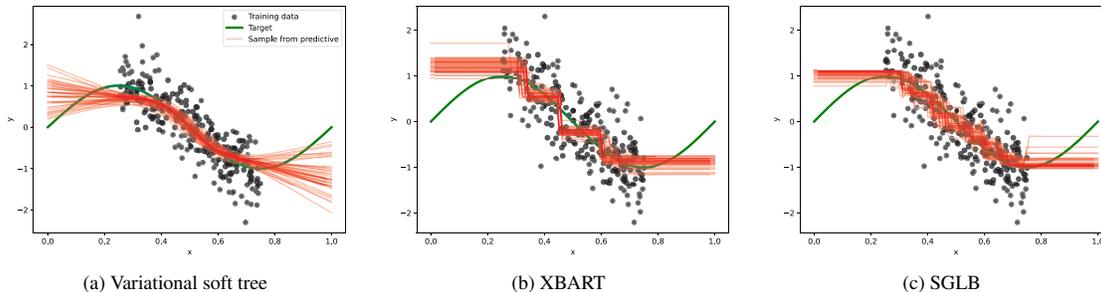


Figure 2: Variational soft trees (VST) show high uncertainty in the tails outside of the data support while closely agreeing with the mean inside the support. Sampled functions show more diversity in tails with VSTs than with XBART and SGLB. The VST uses linear leaves, while XBART and SGLB use constant leaves.

datasets. However, VSGBMs are less competitive in terms of RMSE when compared to XGBoost that yields smaller RMSE on all but three datasets. Compared to other baselines, VSGBM consistently outperforms the deep ensemble, and performs favorably when compared to SGLB and XBART. In addition, we find that VSGBMs with linear leaves perform slightly better than equivalent constant leaf models, outperforming them on 4/10 datasets in terms of log-likelihood and 3/10 in terms of RMSE, while also matching the log-likelihood and RMSE of constant leaves on 5/10 and 6/10 datasets respectively.

Compared to a VST, a VSGBM performs better in terms of RMSE on all datasets but two where both models score equally well. However, VSTs obtain higher log-likelihoods than VSGBMs on 5/10 datasets, equivalent scores on 2/10 and performs worse on 3/10. We explain this by the simpler variance model used in the VSGBM, which does not allow for heteroskedastic (input dependent) uncertainty.

Uncertainty estimation visualization An important property of a reliable predictive model is its increased predictive uncertainty for inputs in regions with little training data. To understand if VSTs demonstrate this behavior, we plot functions sampled from the variational posterior for simple synthetic datasets, and compare them to functions sampled from XBART and SGLB posteriors. We plot the sampled functions in Figures 2 and 5.

We find that VSTs indeed capture the uncertainty outside of

the support of the data better than both XBART and SGLB: functions agree with the true mean within the support of the data, but diverge outside of it.

5.2 OOD detection

Out-of-distribution (OOD) data is data that is sampled from a distribution that is sufficiently different from the training data distribution. It has been observed in prior work (Ma and Hernández-Lobato, 2021; Tomczak et al., 2020) that an effective Bayesian model will show high epistemic uncertainty when presented with OOD data, so we further evaluate our models by testing if their epistemic uncertainty is predictive of OOD data.

Following the setup of Malinin et al. (2020), we define OOD data by selecting a subset of data from a different dataset. To perform OOD detection, we compute the epistemic uncertainty of each sample under our models and evaluate how well a single splitting threshold separates the epistemic uncertainty of in-distribution and OOD samples, as proposed by Mukhoti et al. (2021). Epistemic uncertainty is measured as the variance of the mean prediction with respect to samples from the posterior or the ensemble. The detection performance is quantified using the area under the receiver operating characteristic curve (AUROC). We report the mean and standard deviation of the AUROC of each model across 10 folds of cross-validation. As before, we compare our VSTs and VSGBMs with XBART, SGLB

Table 2: OOD detection AUROC scores with variational soft trees (VST) and variational soft GBMs (VSGBM) against baselines (higher is better). Variational soft decision trees and variational soft GBMs perform well on out of distribution detection, yielding the best score on 6/10 and 5/10 datasets respectively.

MODEL	POWER	PROTEIN	BOSTON	NAVAL	YACHT	SONG	CONCRETE	ENERGY	KIN8NM	WINE
DEEP ENSEMBLE	0.540 ± 0.015	0.504 ± 0.003	0.545 ± 0.023	1.000 ± 0.000	0.691 ± 0.088	0.510 ± 0.007	0.767 ± 0.042	0.554 ± 0.036	0.508 ± 0.005	0.547 ± 0.019
XBART (1 TREE)	0.562 ± 0.018	0.684 ± 0.015	0.656 ± 0.042	0.578 ± 0.050	0.603 ± 0.021	0.608 ± 0.033	0.671 ± 0.028	0.852 ± 0.141	0.525 ± 0.010	0.548 ± 0.016
SGLB (1 TREE)	0.593 ± 0.025	0.760 ± 0.018	0.732 ± 0.039	0.770 ± 0.021	0.607 ± 0.035	0.589 ± 0.035	0.723 ± 0.018	0.969 ± 0.003	0.519 ± 0.005	0.571 ± 0.021
VST (CONST.)	0.591 ± 0.025	0.866 ± 0.018	0.712 ± 0.020	0.884 ± 0.024	0.718 ± 0.048	0.560 ± 0.025	0.677 ± 0.049	0.863 ± 0.045	0.529 ± 0.016	0.796 ± 0.010
VST (LINEAR)	0.618 ± 0.024	0.920 ± 0.019	0.698 ± 0.020	1.000 ± 0.000	0.655 ± 0.015	0.601 ± 0.015	0.840 ± 0.013	0.996 ± 0.003	0.586 ± 0.011	0.742 ± 0.036
XBART	0.572 ± 0.011	0.678 ± 0.016	0.741 ± 0.041	0.912 ± 0.033	0.607 ± 0.026	0.565 ± 0.013	0.818 ± 0.020	0.837 ± 0.146	0.541 ± 0.011	0.549 ± 0.021
SGLB	0.600 ± 0.011	0.874 ± 0.006	0.738 ± 0.027	0.907 ± 0.011	0.662 ± 0.036	0.697 ± 0.029	0.763 ± 0.024	0.973 ± 0.007	0.508 ± 0.004	0.613 ± 0.019
VSGBM (CONST.)	0.569 ± 0.018	0.890 ± 0.011	0.775 ± 0.101	1.000 ± 0.000	0.663 ± 0.055	0.586 ± 0.014	0.666 ± 0.033	0.901 ± 0.018	0.562 ± 0.015	0.625 ± 0.019
VSGBM (LINEAR)	0.571 ± 0.010	0.950 ± 0.016	0.795 ± 0.025	1.000 ± 0.000	0.618 ± 0.020	0.602 ± 0.006	0.763 ± 0.023	0.993 ± 0.008	0.608 ± 0.012	0.688 ± 0.027

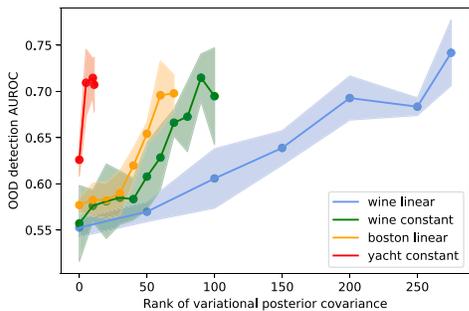


Figure 3: OOD detection AUROC score vs. the rank of the variational posterior covariance matrix in variational soft trees (VST). The quality of the OOD detection generally increases with the rank of the approximate posterior covariance. As we consider a large number of different model and dataset configurations, we only present the ones for which the increase in AUROC is most significant. The rightmost point of each plot corresponds to a full rank covariance approximation.

and a deep ensemble.

We find that variational soft decision trees perform well on OOD detection, outperforming baselines on 6/10 datasets and tying on others (see Table 2). VSGBMs also perform well, yielding the best scores on 5/10 datasets and additionally matching best baselines on 2/10 (see Table 2). However, the improvements over the top baselines are less significant than with variational soft decision trees. As expected, we find that increasing the rank of the approximate variational posterior increases the OOD performance, although the significance of the improvement depends on the dataset (see Figure 3). This effect is most prominent with variational soft decision trees, and is less so in VSGBMs.

5.3 Bandits

We further evaluate the VST by using it in Thompson sampling (Thompson, 1933) on four different multi-armed bandit problems (see Section 2.3 in the supplementary material for more detail on the datasets), comparing to XBART and

Table 3: Cumulative regret of the bandits using variational soft trees (VST) vs. baselines. We find that VST performs competitively compared to baselines, achieving the lowest cumulative regret on 2/4 bandits.

BANDIT	FINANCIAL	EXPLORATION	MUSHROOM	JESTER
RANDOM POLICY	1682.05 ± 20.23	18904.95 ± 24.37	1791.28 ± 29.61	7845.28 ± 26.99
VST	357.47 ± 34.32	7054.60 ± 748.79	1412.55 ± 5.64	6048.99 ± 42.26
SGLB	613.12 ± 45.96	9469.07 ± 1197.60	DOESN'T WORK	5762.89 ± 82.10
XBART	984.86 ± 47.59	5039.93 ± 322.63	1577.70 ± 61.56	6305.43 ± 31.97

SGLB as before. We do not experiment with VSGBMs in the bandit setting due to computational cost. In this setting, the model has to be re-trained after every update, and training a VSGBM is expensive: boosting is sequential by nature, and VST weak learners require a number of gradient descent iterations to converge. Each bandit simulation is run five times with different random seeds, and we report the mean and the standard deviation of the cumulative regret across the simulations.

The predictive uncertainty of VST is competitive in this setting as well, as shown in Table 3. The VST obtains the lowest cumulative regret on the *mushroom* and *financial* datasets, converging to near zero instantaneous regret on the latter (see Figure 4). On the *jester* dataset, the VST demonstrates second best cumulative regret, although the performance of all three models is comparable to the random policy. We explain this by the difficulty of the regression problem in this particular dataset. Finally, just like on *jester*, the VST shows second best cumulative regret on the *exploration* dataset.

6 DISCUSSION

In this work we proposed to perform variational inference to approximate the Bayesian posterior of a soft decision tree. We use a variational distribution that allows trading off memory and computation for a richer posterior approximation. Using soft decision trees models as weak learners, we build a variational soft GBM that is more expressive than the individual trees, but preserves their well-calibrated predictive uncertainty. We find that both variational soft trees and variational soft GBMs perform well on tabular data compared to strong baselines, and provide useful uncertainty

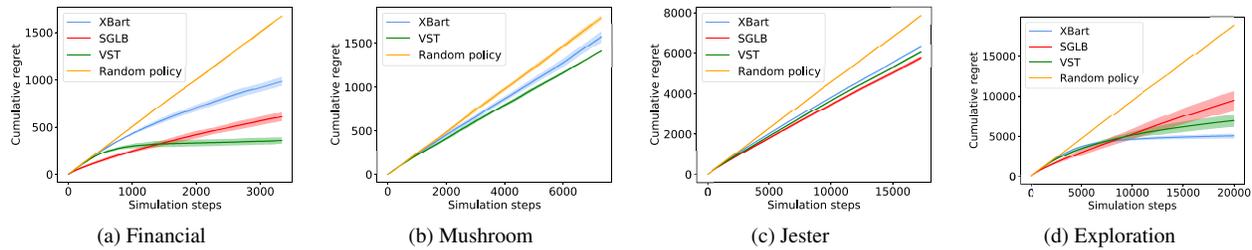


Figure 4: Bandit results. We find that the variational soft tree (VST) converges to near zero instantaneous regret on the financial bandit and performs competitively compared to baselines in the bandit setting.

estimates for OOD detection and multi-armed bandits.

An important direction for future work is to speed up the training of variational soft trees and GBMs. This is especially important when working in the low-data regime ($<10^4$ samples), where our models are slower than the baseline based on standard decision trees, and in the context of multi-armed bandits, where the model needs to be updated repeatedly. For the latter, an alternative direction is to adapt our models for continual learning by implementing an efficient Bayesian update, which would accelerate sequential learning tasks like multi-armed bandits, Bayesian optimization and active learning. Finally, adapting VST and VSGBM to the classification setting is an interesting direction for future work. Classification with tabular data is important in many applications, many of which might also benefit from well-calibrated predictive uncertainties (Ebbehøj et al., 2021; Mustafa and Rahimi Azghadi, 2021).

References

- Abdullah, A., Hassan, M., and Mustafa, Y. (2022). A review on Bayesian deep learning in healthcare: Applications and challenges. *IEEE Access*, 10:1–1.
- Ahmetoğlu, A., İrsoy, O., and Alpaydın, E. (2018). Convolutional soft decision trees. In Kůrková, V., Manolopoulos, Y., Hammer, B., Iliadis, L., and Maglogiannis, I., editors, *Artificial Neural Networks and Machine Learning – ICANN 2018*, pages 134–141, Cham. Springer International Publishing.
- Bew, D., Harvey, C. R., Ledford, A., Radnor, S., and Sinclair, A. (2019). Modeling analysts’ recommendations via Bayesian machine learning. *The Journal of Financial Data Science*, 1(1):75–98.
- Bishop, C. M. and Svensen, M. (2012). Bayesian hierarchical mixtures of experts.
- Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight uncertainty in neural networks.
- Bojer, C. S. and Meldgaard, J. P. (2021). Kaggle forecasting competitions: An overlooked learning opportunity. *International Journal of Forecasting*, 37(2):587–603.
- Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., and Kasneci, G. (2021). Deep neural networks and tabular data: A survey.
- Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. (2018). JAX: Composable transformations of Python+NumPy programs.
- Chen, T. and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 785–794.
- Chipman, H. A., George, E. I., and McCulloch, R. E. (1998). Bayesian cart model search. *Journal of the American Statistical Association*, 93(443):935–948.
- Clements, J. M., Xu, D., Yousefi, N., and Efimov, D. (2020). Sequential deep learning for credit risk monitoring with tabular financial data.
- Cobb, A. D. and Jalaian, B. (2020). Scaling Hamiltonian Monte Carlo inference for Bayesian neural networks with symmetric splitting.
- Coppens, Y., Efthymiadis, K., Lenaerts, T., and Nowé, A. (2019). Distilling deep reinforcement learning policies in soft decision trees. In *IJCAI 2019*.
- Deisenroth, M. and Rasmussen, C. (2011). Pilco: A model-based and data-efficient approach to policy search. pages 465–472.
- Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38.
- Dua, D. and Graff, C. (2017). UCI machine learning repository.
- Ebbehøj, A., Thunbo, M., Andersen, O. E., Glindtvad, M. V., and Hulman, A. (2021). Transfer learning for non-image data in clinical research: a scoping review. *medRxiv*.
- Feng, J., Xu, Y.-X., Jiang, Y., and Zhou, Z.-H. (2020). Soft gradient boosting machine.

- Flam-Shepherd, D. (2017). Mapping Gaussian process priors to Bayesian neural networks.
- Fortuin, V., Garriga-Alonso, A., Ober, S. W., Wenzel, F., Rätsch, G., Turner, R. E., van der Wilk, M., and Aitchison, L. (2021). Bayesian neural network priors revisited.
- Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139.
- Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189 – 1232.
- Frosst, N. and Hinton, G. (2017). Distilling a neural network into a soft decision tree.
- Goldberg, K., Roeder, T., Gupta, D., and Perkins, C. (2001). Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4:133–151.
- Gonzalvez, J., Lezmi, E., Roncalli, T., and Xu, J. (2019). Financial applications of Gaussian processes and Bayesian optimization.
- Graf, F., Kriegel, H.-P., Schubert, M., Pölsterl, S., and Cavallaro, A. (2011). 2D image registration in CT images using radial image descriptors. *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, 14 Pt 2:607–14.
- Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR.
- He, J. and Hahn, P. R. (2020). Stochastic tree ensembles for regularized nonlinear regression.
- Hennigan, T., Cai, T., Norman, T., and Babuschkin, I. (2020). Haiku: Sonnet for JAX.
- Hoang, D. and Wiegatz, K. (2021). Machine learning methods in finance: Recent applications and prospects.
- Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *Journal of Machine Learning Research*, 14(40):1303–1347.
- Immer, A., Korzepa, M., and Bauer, M. (2020). Improving predictions of Bayesian neural nets via local linearization.
- Irsoy, O., Yildiz, O., and Alpaydin, E. (2012). Soft decision trees. pages 1819–1822.
- Jordan, M. and Jacobs, R. (1993). Hierarchical mixtures of experts and the EM algorithm. In *Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan)*, volume 2, pages 1339–1344 vol.2.
- Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization.
- Kingma, D. P. and Welling, M. (2013). Auto-encoding variational Bayes.
- Kirsch, A., van Amersfoort, J., and Gal, Y. (2019). Batch-bald: Efficient and diverse batch acquisition for deep Bayesian active learning.
- Kompa, B., Snoek, J., and Beam, A. (2021). Second opinion needed: communicating uncertainty in medical machine learning. *npj Digital Medicine*, 4.
- Kontschieder, P., Fiterau, M., Criminisi, A., and Rota Bulò, S. (2015). Deep neural decision forests. pages 1467–1475.
- Krause, A., Guestrin, C., Gupta, A., and Kleinberg, J. (2006). Near-optimal sensor placements: Maximizing information while minimizing communication cost. In *2006 5th International Conference on Information Processing in Sensor Networks*, pages 2–10.
- Krishnan, R., Subedar, M., and Tickoo, O. (2020). Specifying weight priors in Bayesian deep neural networks with empirical Bayes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):4477–4484.
- Kristiadi, A., Hein, M., and Hennig, P. (2020). Being Bayesian, even just a bit, fixes overconfidence in ReLU networks.
- Linero, A. R. and Yang, Y. (2017). Bayesian regression tree ensembles that adapt to smoothness and sparsity.
- Liu, Z., Pavao, A., Xu, Z., Escalera, S., Ferreira, F., Guyon, I., Hong, S., Hutter, F., Ji, R., Jacques Junior, J. C. S., Li, G., Lindauer, M., Luo, Z., Madadi, M., Nierhoff, T., Niu, K., Pan, C., Stoll, D., Treguer, S., Wang, J., Wang, P., Wu, C., Xiong, Y., Zela, A., and Zhang, Y. (2021). Winning solutions and post-challenge analyses of the ChaLearn AutoDL challenge 2019. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Luo, H., Cheng, F., Yu, H., and Yi, Y. (2021). SDTR: Soft decision tree regressor for tabular data. *IEEE Access*, 9:55999–56011.
- Ma, C. and Hernández-Lobato, J. M. (2021). Functional variational inference based on stochastic process generators. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 21795–21807. Curran Associates, Inc.
- Maia, M., Murphy, K., and Parnell, A. C. (2022). GP-BART: A novel Bayesian additive regression trees approach using Gaussian processes. *arXiv e-prints*, page arXiv:2204.02112.
- Malinin, A., Prokhorenkova, L., and Ustimenko, A. (2020). Uncertainty in gradient boosting via ensembles.
- Mitros, J. and Namee, B. M. (2019). On the validity of Bayesian neural networks for uncertainty estimation.
- Mukhoti, J., Kirsch, A., van Amersfoort, J., Torr, P. H. S., and Gal, Y. (2021). Deep deterministic uncertainty: A simple baseline.

- Mustafa, A. and Rahimi Azghadi, M. (2021). Automated machine learning for healthcare and clinical notes analysis. *Computers*, 10(2).
- Niculescu-Mizil, A. and Caruana, R. A. (2012). Obtaining calibrated probabilities from boosting.
- Pratola, M., Chipman, H., George, E., and McCulloch, R. (2017). Heteroscedastic BART using multiplicative regression trees.
- Riquelme, C., Tucker, G., and Snoek, J. (2018). Deep Bayesian bandits showdown: An empirical comparison of Bayesian deep networks for Thompson sampling.
- Shwartz-Ziv, R. and Armon, A. (2022). Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90.
- Springenberg, J. T., Klein, A., Falkner, S., and Hutter, F. (2016). Bayesian optimization with robust Bayesian neural networks. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.
- Swiatkowski, J., Roth, K., Veeling, B. S., Tran, L., Dillon, J. V., Snoek, J., Mandt, S., Salimans, T., Jenatton, R., and Nowozin, S. (2020). The k-tied normal distribution: A compact parameterization of Gaussian mean field posteriors in Bayesian neural networks.
- Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294.
- Tomczak, M., Swaroop, S., and Turner, R. (2020). Efficient low rank Gaussian variational inference for neural networks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4610–4622. Curran Associates, Inc.
- Tran, D., Ranganath, R., and Blei, D. M. (2015). The variational Gaussian process.
- Ueda, N. and Ghahramani, Z. (2002). Bayesian model search for mixture models based on optimizing variational bounds. *Neural Networks*, 15(10):1223–1241.
- Ustimenko, A. and Prokhorenkova, L. (2020). SGLB: Stochastic gradient Langevin boosting.
- Vladimirova, M., Verbeek, J., Mesejo, P., and Arbel, J. (2018). Understanding priors in Bayesian neural networks at the unit level.
- Waterhouse, S., MacKay, D., and Robinson, A. (1995). Bayesian methods for mixtures of experts. In Touretzky, D., Mozer, M., and Hasselmo, M., editors, *Advances in Neural Information Processing Systems*, volume 8. MIT Press.
- Welling, M. and Teh, Y. W. (2011). Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning, ICML'11*, page 681–688, Madison, WI, USA. Omnipress.

1 Additional method details

1.1 Variational inference

We use the variational distribution proposed by Tomczak et al. (2020), which is a normal distribution with a low-rank covariance matrix:

$$q(\theta) = \mathcal{N}(\theta | \mu, \text{diag}[\sigma^2] + VV^T) \tag{18}$$

where $\mu \in \mathbb{R}^p$ is the mean of the normal distribution over parameters, $\text{diag}[\sigma^2] \in \mathbb{R}^{p \times p}$ is the diagonal component of the covariance with $\sigma \in \mathbb{R}^p$, and $V \in \mathbb{R}^{p \times k}$ is the rank k component of the covariance. The hyperparameter k determines the rank of the covariance matrix, and hence the expressivity of the posterior. Choosing $k = p$ allows learning arbitrary covariance matrices, but increases the memory requirements and computational cost. As a result, in our experiments we typically choose $k < p$. We fit the variational soft tree by maximizing the evidence lower bound (ELBO):

$$\text{ELBO} = \mathbb{E}_{\theta \sim q(\theta)}[\log p(y | x, \theta)] - D_{\text{KL}}(q(\theta) \| p(\theta)). \tag{19}$$

We compute a Monte Carlo estimate of the expectation:

$$\mathbb{E}_{\theta \sim q}[\log p(y | x, \theta)] \approx \frac{1}{m} \sum_{i=1}^m \log p(y | x, \theta^{(i)}), \tag{20}$$

where $\theta^{(i)} \sim q(\theta)$. For the chosen posterior in Equation (18) and an isotropic Gaussian prior $p(\theta) = \mathcal{N}(\theta | 0, \gamma \mathbb{I})$, the KL divergence term is available in a closed form:

$$D_{\text{KL}}(q(\theta) \| p(\theta)) = \frac{1}{2} \left[\sum_{d=1}^p \left(\frac{\sigma_d^2}{\gamma} - \log \sigma_d^2 \right) + \frac{1}{\gamma} \sum_{i=1}^k \|v_i\|_2^2 - \Delta + \frac{1}{\gamma} \|\mu\|_2^2 + p(\log \gamma - 1) \right], \tag{21}$$

where $V = [v_1, \dots, v_k] \in \mathbb{R}^{p \times k}$, $\sigma \in \mathbb{R}^p$ and $\Delta = \log \det(\mathbb{I}_k + V^T \text{diag}[\sigma^2]^{-1} V)$.

2 Experimental details

2.1 Regression

We consider a suite of regression datasets from Malinin et al. (2020) to measure the expressivity of our model. Table 4 summarizes the number of examples and features in each dataset. We preprocess the datasets by encoding categorical variables as one-hot vectors and normalizing each feature to have zero mean and unit variance. We further split the datasets into train (80%) and test (20%) data, and perform 10-fold cross validation on the training data. We report the mean and the standard deviation of the test log-likelihood and root mean square error (RMSE) across all folds.

Table 4: Regression dataset description

DATASET	BOSTON	NAVAL	POWER	PROTEIN	YACHT	SONG	CONCRETE	ENERGY	KIN8NM	WINE
NUMBER SAMPLES	506	11 934	9 568	45 730	308	514 345	1 030	768	8 192	1 599
NUMBER FEATURES	13	16	4	9	6	90	8	8	8	11

2.2 OOD detection

Following the setup from Malinin et al. (2020), we consider the test data to be the in-distribution (ID) data, and create a dataset of OOD data by taking a subset of the *song* dataset of the same size. We select a subset of features from the *song* dataset in order to match the feature dimension of the ID dataset. The only exception is the *song* dataset itself, for which we use the *Relative location of CT slices on axial axis* dataset from Graf et al. (2011) as OOD data. In both settings, categorical features are encoded as one-hot vectors, and all features are normalized to have zero mean and unit variance. We use a one-node decision tree to find the splitting threshold for the epistemic uncertainty that would separate ID and ODD data.

2.3 Bandit

We consider the following datasets for our bandit experiments:

Financial The financial bandit (Riquelme et al., 2018) was created by pulling the stock prices of $d = 21$ publicly traded companies in NYSE and Nasdaq, for the last 14 years for a total of $n = 3713$ samples. For each day, the context was the price difference between the beginning and end of the session for each stock. The arms are synthetically created to be a linear combination of the contexts, representing $k = 8$ different portfolios.

Jester This bandit problem is taken from Riquelme et al. (2018). The Jester dataset (Goldberg et al., 2001) provides continuous ratings in $[-10, 10]$ for 100 jokes from 73 421 users. The authors find a complete subset of $n = 19 181$ users rating all 40 jokes. We take $d = 32$ of the ratings as the context of the user, and $k = 8$ as the arms. The agent recommends one joke, and obtains the reward corresponding to the rating of the user for the selected joke.

Exploration The exploration bandit tests the extent to which the models explore the different arms. The reward function of each arm is sparse and requires maintaining high uncertainty to discover the context values for which the reward is high. The reward given input x and selected arm a is given by:

$$r(x, a) = \sigma(\beta(x + \alpha - \text{offset}[a])) - \sigma(\beta(x - \alpha - \text{offset}[a])) + \epsilon \tag{22}$$

where $\sigma(x) = 1 / (1 + e^{-x})$ is the sigmoid function, $\alpha, \beta \in \mathbb{R}$ are constants, $\epsilon \sim N(0, \delta^2)$, and $\text{offset}[a]$ is a function mapping each arm a to a particular offset. Intuitively, this reward is composed of a smooth unit valued “bump” per arm and is otherwise 0. The positions of the “bumps” across arms are disjoint and barely overlap. Furthermore, the context is sampled uniformly at random in the interval $[-1, 1]$. We run this bandit for 20 000 steps.

Mushroom This bandit problem is taken from Riquelme et al. (2018). The Mushroom Dataset (Dua and Graff, 2017) contains $n = 7310$ samples, 22 attributes per mushroom, and two classes: poisonous and safe. The agent must decide whether to eat a given mushroom or not. Eating a safe mushroom provides reward of 5. Eating a poisonous mushroom delivers reward 5 with probability 1/2 and reward -35 otherwise. If the agent does not eat a mushroom, then the reward is 0.

2.4 Implementation

We implement the variational soft decision tree and variational soft GBM using the Jax (Bradbury et al., 2018) and DM-Haiku (Hennigan et al., 2020) Python libraries. We fit our models using the Adam (Kingma and Ba, 2014) optimizer. Hyperparameter optimization was conducted using Amazon SageMaker’s² automatic model tuning that uses Bayesian optimisation to explore the hyperparameter space. We define the set of hyperparameters that the Bayesian optimisation needs to explore, as well as the optimization objective. The service then runs a series of model tuning jobs with different hyperparameters to find the set of values which minimize the objective. We use the same hyperparameter ranges as (He and Hahn, 2020) and (Ustimenko and Prokhorenkova, 2020), summarized in Table 5.

Table 5: Hyperparameter tuning ranges.

VSGBM		XBART		XGBOOST		DEEP ENSEMBLE		VST		SGLB	
LEARNING RATE	{0.0001, 0.001}	S	{1, 300}	LEARNING RATE	{0.001, 0.1}	LEARNING RATE	{0.0001, 0.001}	LEARNING RATE	{0.0001, 0.001}	LEARNING RATE	{0.001, 0.01, 0.1}
PRIOR SCALE	{0.01, 2.5}	TAU	{1, 300}	TREE DEPTH	{1, ..., 6}	NUM. LAYERS	{2, ..., 5}	PRIOR SCALE	{0.01, 2.5}	TREE DEPTH	{3, ..., 6}
MAX. DEPTH	{1, ..., 5}	BETA	{0.75, 2}	LAMBDA	{0.001, 1}	HIDDEN DIM.	{10, ..., 50}	MAX. DEPTH	{1, ..., 5}	NUM. LEARNERS	{1, ..., 1000}
BETA	{1, 50}	NUM. LEARNERS	{1, 100}	ALPHA	{0.001, 1}	DROPOUT RATE	{0.1, 0.4}	BETA	{1, 50}		
NUM. LEARNERS	{1, ..., 10}	BURN-IN	{15}	NUM. LEARNERS	{1, ..., 1000}	PRIOR SCALE	{0.001, 2.5}				
PRIOR SIGMA ALPHA	{2.5, 5}	ALPHA	{0.95}								
PRIOR SIGMA BETA	{0.1, 5}	KAPPA	{3}								

3 Additional experimental results

3.1 Regression

In this section we present additional regression results reporting the root mean square error (RMSE) of evaluated methods across the considered baselines, see Table 6. We find that the VST performs well in terms of test RMSE across datasets:

²<https://aws.amazon.com/sagemaker/>

Table 6: Test RMSE of evaluated methods on regression datasets. We find that variational soft decision trees (VST) compares well with baselines in terms of test RMSE, out-performing them on 3/10 datasets and matching the RMSE of the best baselines on three. Variational soft GBM (VSGBM) performs competitively in terms of RMSE compared to SGLB, deep ensemble and XBART, but is generally weaker than XGBoost. This is unsurprising as XGBoost is the state-of-the-art for fitting tabular data.

MODEL	POWER	PROTEIN	BOSTON	NAVAL	YACHT	SONG	CONCRETE	ENERGY	KIN8NM	WINE
DEEP ENSEMBLE	0.986 ± 0.001	1.003 ± 0.001	0.870 ± 0.004	0.993 ± 0.001	0.685 ± 0.002	0.999 ± 0.001	0.972 ± 0.001	1.007 ± 0.009	0.998 ± 0.001	1.013 ± 0.011
XBART (1 TREE)	0.243 ± 0.001	0.693 ± 0.003	0.356 ± 0.026	0.339 ± 0.125	0.052 ± 0.003	0.680 ± 0.007	0.474 ± 0.046	0.406 ± 0.267	0.703 ± 0.013	0.815 ± 0.006
SGLB (1 TREE)	0.280 ± 0.006	0.869 ± 0.002	0.483 ± 0.025	0.830 ± 0.007	0.405 ± 0.007	0.958 ± 0.007	0.541 ± 0.010	0.193 ± 0.003	0.740 ± 0.002	0.827 ± 0.004
XGBOOST (1 TREE)	0.242 ± 0.003	0.828 ± 0.001	0.397 ± 0.077	0.703 ± 0.001	0.056 ± 0.006	0.795 ± 0.003	0.454 ± 0.021	0.059 ± 0.005	0.776 ± 0.010	0.854 ± 0.020
VST (CONST.)	0.239 ± 0.001	0.827 ± 0.008	0.538 ± 0.004	0.359 ± 0.033	0.477 ± 0.027	0.710 ± 0.010	0.592 ± 0.005	0.483 ± 0.099	0.550 ± 0.010	0.788 ± 0.009
VST (LINEAR)	0.242 ± 0.001	0.798 ± 0.011	0.469 ± 0.003	0.376 ± 0.002	0.500 ± 0.004	0.633 ± 0.006	0.423 ± 0.066	0.281 ± 0.044	0.530 ± 0.018	0.781 ± 0.008
XBART	0.233 ± 0.001	0.692 ± 0.003	0.303 ± 0.022	0.152 ± 0.045	0.051 ± 0.003	0.576 ± 0.003	0.355 ± 0.012	0.292 ± 0.295	0.590 ± 0.007	0.815 ± 0.008
SGLB	0.453 ± 0.001	0.822 ± 0.001	0.583 ± 0.004	0.649 ± 0.002	0.300 ± 0.004	0.834 ± 0.001	0.593 ± 0.002	0.318 ± 0.002	0.778 ± 0.001	0.857 ± 0.002
XGBOOST	0.173 ± 0.003	0.614 ± 0.004	0.261 ± 0.012	0.057 ± 0.001	0.041 ± 0.006	0.450 ± 0.001	0.285 ± 0.014	0.050 ± 0.001	0.479 ± 0.019	0.735 ± 0.014
VSGBM (CONST.)	0.238 ± 0.001	0.652 ± 0.006	0.363 ± 0.025	0.111 ± 0.060	0.142 ± 0.107	0.563 ± 0.004	0.347 ± 0.016	0.227 ± 0.015	0.348 ± 0.009	0.781 ± 0.008
VSGBM (LINEAR)	0.242 ± 0.001	0.663 ± 0.012	0.297 ± 0.009	0.074 ± 0.018	0.134 ± 0.005	0.483 ± 0.007	0.341 ± 0.014	0.183 ± 0.037	0.316 ± 0.010	0.769 ± 0.006

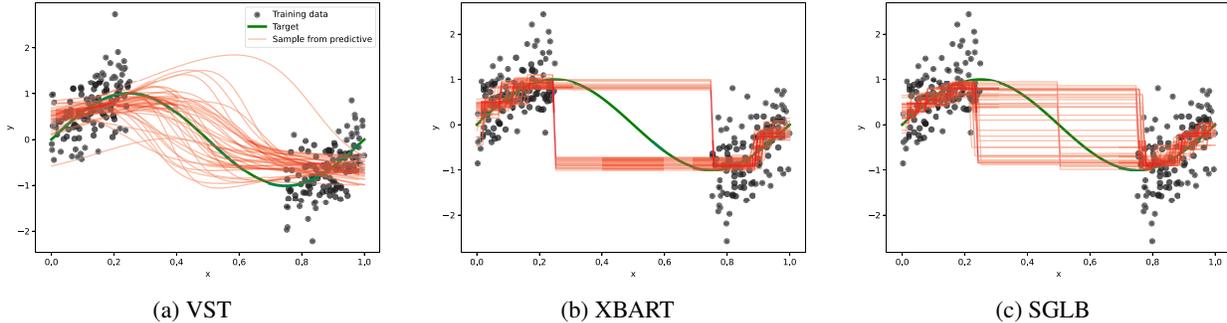


Figure 5: In-between uncertainty. Variational soft trees (VST) show high uncertainty between the two blobs of data while closely agreeing with the true mean inside of the data support. The plot for the VST was generated with the linear leaf model.

our model outperforms the baselines on 3/10 datasets and matching the RMSE of the top baselines on 3/10. This further confirms that soft decision trees perform well on tabular data (Luo et al., 2021).

We observe that VSGBMs are generally weaker in terms of RMSE compared to XGBoost. XGBoost yields a smaller RMSE on all but three datasets, where VSGBM performs better on one and ties on two. This result is not surprising as XGBoost is the gold standard for fitting tabular data. Compared to other baselines, VSGBM systematically outperforms deep ensembles, and performs favorably compared to SGLB and XBART. Compared to a single variational soft tree, VSGBMs perform better in terms of RMSE on all datasets but two where they tie.

3.2 Uncertainty visualization

In order to demonstrate the quality of the predictive uncertainty of our model, we present additional plots of functions sampled from the VST, comparing these to samples from XBART and SGLB. See Figure 2 and Figure 5. We find that variational soft decision trees model the uncertainty outside of the support of the data better than XBART and SGLB: functions agree with the true mean within the support of the data, but diverge outside of it.

3.3 Regression visualization

To test the flexibility of the VST in the regression setting, we fit our model to a step function (Figure 6) and a Daubechies wavelet (Figure 7) as done in (Linero and Yang, 2017), and plot functions sampled from our model. We see that the VST captures the non-smooth step function as well as the hard decision trees used in XBART and SGLB. On the Daubechies wavelet, however, we find that a VST of a reasonable depth (5) underfits. VSGBM visually improves the fit over a VST, but still underfits the data compared to SGLB and XBART. We hypothesize that this effect is due to the choice of zero mean Gaussian prior on the node weights w_n in VSTs and VSGBMs, which encourages smooth functions. This phenomenon is particularly visible when fitting the Daubechies wavelet, a complex function that fluctuates at different frequencies and amplitudes. As expected, increasing the amount of training data reduces underfitting.

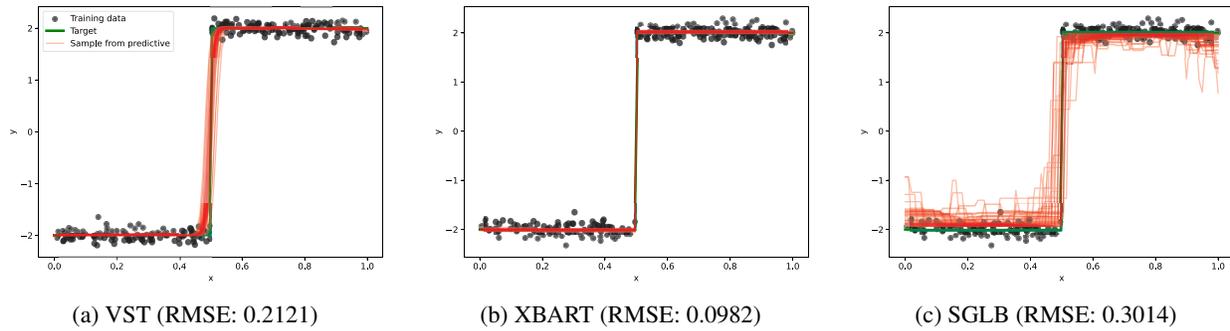


Figure 6: Step function. The variational soft decision tree (VST) can fit this non-smooth function. The plot for the VST was generated with the linear leaf model.

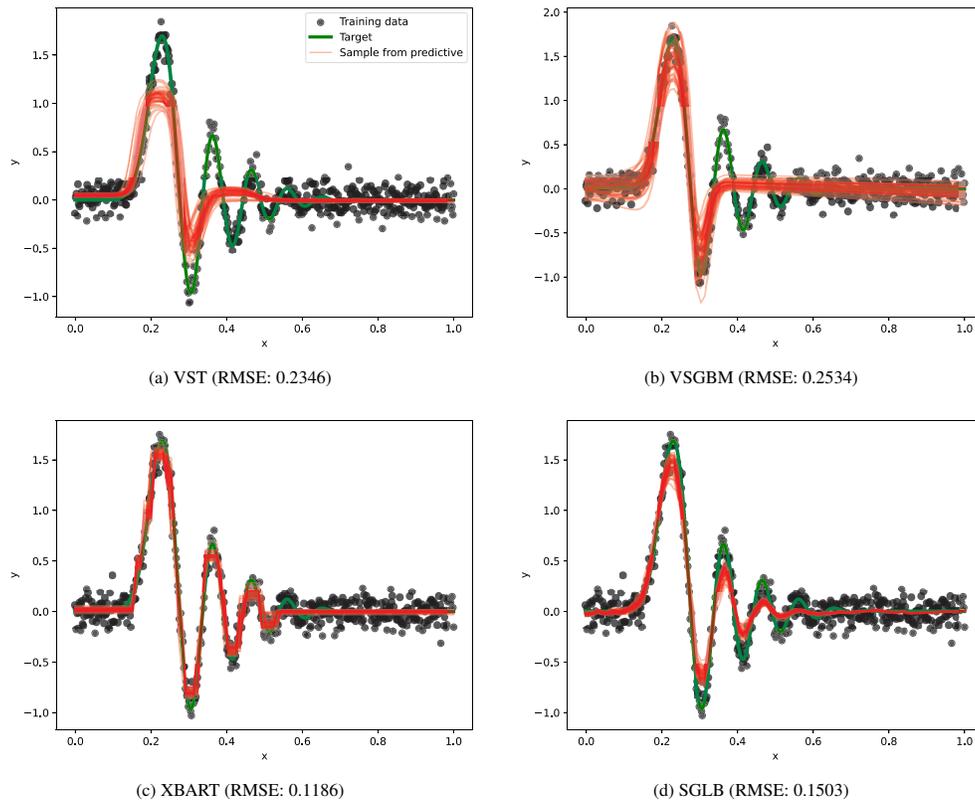


Figure 7: Daubechies wavelet. The variational soft tree (VST) and variational soft GBM (VSGBM) underfit this complex function.