TreeFlow: Going Beyond Tree-Based Parametric Probabilistic Regression

Patryk Wielopolski[a, b;*] **and Maciej Zięba**[a, c]

[a]Wrocław University of Science and Technology
[b]DataWalk
[c]Tooploox

Abstract. The tree-based ensembles are known for their outstanding performance in classification and regression problems characterized by feature vectors represented by mixed-type variables from various ranges and domains. However, considering regression problems, they are primarily designed to provide deterministic responses or model the uncertainty of the output with Gaussian or parametric distribution. In this work, we introduce TreeFlow, the tree-based approach that combines the benefits of using tree ensembles with the capabilities of modeling flexible probability distributions using normalizing flows. The main idea of the solution is to use a tree-based model as a feature extractor and combine it with a conditional variant of normalizing flow. Consequently, our approach is capable of modeling complex distributions for the regression outputs. We evaluate the proposed method on challenging regression benchmarks with varying volume, feature characteristics, and target dimensionality. We obtain the SOTA results for both probabilistic and deterministic metrics on datasets with multi-modal target distributions and competitive results on unimodal ones compared to tree-based regression baselines.

1 Introduction

The modern tree-based models achieve outstanding results for problems where the data representation is tabular, the number of training examples is limited, and the input feature vector is represented by mixed-type variables from various ranges and domains. Most of such algorithms focus on providing deterministic predictions, paying no attention to the probabilistic nature of the provided output. However, for many practical applications, it is impossible to deliver an exact target value based on the given input factors. Consider the regression problem of predicting the future location of the vehicle that is approaching a roundabout [29]. Having past coordinates and other information aggregated in current and past states, we cannot unambiguously predict which of the three remaining exits from the roundabout will be taken by the tracked object. Therefore, it is more beneficial to provide multimodal probability distribution for future locations instead of a single deterministic prediction oscillating around one mode.

Due to the tractable closed-form, the standard approaches assume to model regression uncertainty using Gaussian or parametric distributions [17]. The well-known deterministic gradient boosting machine method adopted those approaches to tree-based structures [6, 13, 25]. Consequently, they can capture the uncertainty of the regression outputs with a standard family of distributions.

* Corresponding Author. Email: patryk.wielopolski@pwr.edu.pl

The major limitation of the current approaches is modeling regression outputs using only Gaussians. Moreover, it is not trivial to extend them to a mixture of Gaussians to capture the multi-modalities of the predictions. Creating multivariate extensions of such models is also ineffective, especially for higher dimensionality, due to the need to estimate the complete covariance matrix.

To reduce the limitations of existing methods, we introduce TreeFlow - a novel tree-based approach for modeling probabilistic regression. The proposed method combines the benefits of using tree-based structures as feature extractors with the normalizing flows [22] capable of modeling flexible data distributions. We introduce the novel concept of combining forest structure with a conditional flow variant to model uncertainty for regression output. Thanks to that approach, we can model complex non-Gaussian or in general non-parametric data distributions even for high-dimensional predictions. We confirm the quality of the proposed model in the experimental part, where we show the superiority of our method over the baselines.

To summarize, our contributions are as follows:

- According to our knowledge, for the first time, tree-based models are used to model non-parametric probabilistic regression for both uni- and multi-variate predictions.
- We propose a novel approach for combining tree-based models with conditional flows via binary representation of the forest structure.
- We obtain the SOTA results for both probabilistic (NLL, CRPS) and deterministic (RMSE) metrics on datasets with multi-modal target distributions and competitive results on unimodal ones compared to tree-based regression baselines.

2 Background

Assume we have a dataset $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1..N}$ where $\mathbf{x}_n = (x_n^1, \ldots, x_n^D)$ is a D-dimensional random vector of features and $\mathbf{y}_n = (y_n^1, \ldots, y_n^P)$ is a P-dimensional vector of targets. We consider regression problems, thus, we assume that $y_n^p \in \mathbb{R}$. Additionally, when $P = 1$ we will refer to that as a univariate regression problem, and when $P \geq 2$ as a multivariate regression problem.

For the probabilistic regression task, we aim at modeling conditional probability distribution $p(\mathbf{y}|\mathbf{x})$. Assuming some parametrization of the regression model $\boldsymbol{\theta}$, the problem of training the probabilistic model can be expressed as minimisation of the conditional negative log likelihood function (NLL) given by $Q(\boldsymbol{\theta}) = -\sum_{n=1}^{N} \log p(\mathbf{y}_n|\mathbf{x}_n, \boldsymbol{\theta})$. During the training procedure we aim at finding $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} Q(\boldsymbol{\theta})$.

Decision Tree Ensembles Decision Tree [2] recursively partition feature space $\mathbb{R}^D$ into K disjoint regions $\mathcal{R}_k$ (tree leaves) and for each region assign value w_k. Formally, the model can be written as $h(\mathbf{x}) = \sum_{k=1}^{K} w_k \mathbf{1}_{\{\mathbf{x} \in \mathcal{R}_k\}}$.

Decision Tree Ensembles are constructed of multiple, usually shallow decision trees, whose results are differently aggregated depending on the training mode. In general, we distinguish two main approaches: independent model training with average or majority voting such as Random Forest [1], and iterative model training with additive aggregation such as Gradient Boosting Machine (GBM) [8].

For the univariate probabilistic regression, GBM optimizes the loss function given by negative log likelihood (NLL). Then it assumes the target variable has a Gaussian distribution, i.e.,

$$p(y|\mathbf{x}, \boldsymbol{\theta}^{(t)}) = \mathcal{N}(y|\mu^{(t)}, \sigma^{(t)}), \quad (1)$$

where $\{\mu^{(t)}, \log \sigma^{(t)}\} = F^{(t)}(\mathbf{x})$ and $F^{(t)}(\mathbf{x})$ is an output of t-th tree from GBM model consisted of T trees. In the multivariate case, it assumes Multivariate Normal distribution and uses the parametrization trick with Cholesky decomposition of the covariance matrix which reduces the number of parameters.

In practice, NGBoost [6] supports both uni- and multi-variate Gaussian distributions and estimates each parameter using one underlying model. CatBoost [13] supports only univariate Gaussians but estimates all distribution parameters using only one model. Moreover, it provides deterministic multivariate regression with the same property, which keeps the total number of trees relatively small. In this case, the loss function is Multioutput Root Mean Squared Error (MultiRMSE).

Normalizing Flows Normalizing flows [22] represent the group of generative models that can be efficiently trained via direct likelihood estimation thanks to the application of a change-of-variable formula. Practically, they utilize a series of (parametric) invertible functions: $\mathbf{y} = \mathbf{f}_n \circ \cdots \circ \mathbf{f}_1(\mathbf{z})$. Assuming given base distribution $p(\mathbf{z})$ for $\mathbf{z}$, the log likelihood for $\mathbf{y}$ is given by $\log p(\mathbf{y}) = \log p(\mathbf{z}) - \sum_{n=1}^{N} \log \left| \det \frac{\partial \mathbf{f}_n}{\partial \mathbf{z}_{n-1}} \right|$. In practical applications $p(\mathbf{y})$ represents the distribution of observable data and $p(\mathbf{z})$ is usually assumed to be Gaussian with independent components.

The sequence of discrete transformations can be replaced by continuous alternative by application of Continuous Normalizing Flows (CNFs) [3, 9] where the aim is to solve the differential equation of the form $\frac{\partial \mathbf{z}}{\partial t} = \mathbf{g}_{\boldsymbol{\beta}}(\mathbf{z}(t), t)$, where $\mathbf{g}_{\boldsymbol{\beta}}(\mathbf{z}(t), t)$ represents the function of dynamics, described by parameters $\boldsymbol{\beta}$. Our goal is to find solution of the equation in t_1, $\mathbf{y} := \mathbf{z}(t_1)$, assuming the given initial state $\mathbf{z} := \mathbf{z}(t_0)$ with a known prior. The transformation function $\mathbf{f}_{\boldsymbol{\beta}}$ is defined as:

$$\mathbf{y} = \mathbf{f}_{\boldsymbol{\beta}}(\mathbf{z}) = \mathbf{z} + \int_{t_0}^{t_1} \mathbf{g}_{\boldsymbol{\beta}}(\mathbf{z}(t), t) dt. \quad (2)$$

The inverted form of the transformation can be easily computed using the formula: $\mathbf{f}_{\boldsymbol{\beta}}^{-1}(\mathbf{y}) = \mathbf{y} - \int_{t_0}^{t_1} \mathbf{g}_{\boldsymbol{\beta}}(\mathbf{z}(t), t) dt$. The log-probability of $\mathbf{y}$ can be computed by:

$$\log p(\mathbf{y}) = \log p(\mathbf{f}_{\boldsymbol{\beta}}^{-1}(\mathbf{y})) - \int_{t_0}^{t_1} \mathrm{Tr}\left(\frac{\partial \mathbf{g}_{\boldsymbol{\beta}}}{\partial \mathbf{z}(t)} \right) dt, \quad (3)$$

where $\mathbf{f}_{\boldsymbol{\beta}}^{-1}(\mathbf{y}) = \mathbf{z}$.

CNFs are rather designed to model complex probability distributions for low-dimensional data, what was confirmed in various applications including point cloud generation [27], future prediction [29] or probabilistic few-shot regression [23]. Compared to models like RealNVP [5] or Glow [11], they can be successfully applied to one-dimensional data and achieve better results for tabular datasets.

3 TreeFlow

Tree-based methods obtain superior results on tabular datasets and have developed multiple techniques to deal with categorical variables, null values, etc. but are limited to distributions with explicitly provided probability distribution functions, e.g., Gaussian. We want to overcome this limitation by introducing TreeFlow - method for uni- and multi-variate tree-based probabilistic regression with non-Gaussian and multi-modal target distributions. The main idea of the solution is to combine the benefits of using tree ensembles with the capabilities of modeling flexible probability distributions using conditional normalizing flows.

The architecture of TreeFlow is provided in fig. 1. The proposed model consists of three components: Tree-based Feature Extractor, Shallow Feature Extractor, and conditional CNF module. The role of the first component is to extract the vector of binary features from the structure of the tree-based ensemble model for a given input observation $\mathbf{x}$. The problem of extracting a unified vector from the tree-based ensemble model is non-trivial due to the complex structure and a large number of base learners. Motivated by the fact that crucial information extracted from input examples is stored in the leaves, we propose a binary occurrence representation that is the most lightweight approach assuming thousands of trees. Formally it could be written as $\mathbf{h}_{\boldsymbol{\psi}}(\mathbf{x}) = [\mathbf{o}_1, \dots, \mathbf{o}_T]$, where $\mathbf{h}_{\boldsymbol{\psi}}(\mathbf{x})$ is Tree-based Feature Extractor with parameters $\boldsymbol{\psi}$ and $\mathbf{o_i} = [\mathbf{1}_{\{\mathbf{x} \in \mathcal{R}_{i,1}\}}, \dots, \mathbf{1}_{\{\mathbf{x} \in \mathcal{R}_{i,K}\}}]$ where $\mathcal{R}_{i,k}$ is a region of kth leaf of ith decision tree in the forest structure.

The size of vector $\mathbf{o}$ is significantly larger than the size of the regression variable $\mathbf{y}$. If we deliver directly the large sparse binary vector as a CNF conditioning component: (i) the number of CNF parameters grows significantly, (ii) the conditioning component dominates training, and (iii) the ordinary differential equation (ODE) solver slows down significantly and behaves in an unstable way. Therefore, we use an additional Shallow Feature Extractor $\mathbf{k}_{\phi}(\cdot)$, that is represented by a neural network responsible for mapping high-dimensional binary vector $\mathbf{o}$ returned by $\mathbf{h}_{\boldsymbol{\psi}}(\mathbf{x})$ to low-dimensional feature representation $\mathbf{w} = \mathbf{k}_{\phi}(\mathbf{o})$. The low-dimensional representation $\mathbf{w}$ of the sparse embedding $\mathbf{o}$ is further passed to the conditional CNF module as a conditioning factor. We postulate to use the variant of the conditional flow-based model provided in [27, 23], where $\mathbf{w}$ is delivered to the function of dynamics, $\mathbf{g}_{\boldsymbol{\beta}}(\mathbf{z}(t), t, \mathbf{w})$. The transformation function is given by eq. (2) is represented as:

$$\mathbf{y} = \mathbf{f}_{\boldsymbol{\beta}}(\mathbf{z}, \mathbf{w}) = \mathbf{z} + \int_{t_0}^{t_1} \mathbf{g}_{\boldsymbol{\beta}}(\mathbf{z}(t), t, \mathbf{w}) dt. \quad (4)$$

The inverse form of the transformation $\mathbf{f}_{\boldsymbol{\beta}}(\cdot)$ given the same $\mathbf{w}$ in both directions is simply: $\mathbf{z} = \mathbf{f}_{\boldsymbol{\beta}}^{-1}(\mathbf{y}, \mathbf{w}) = \mathbf{y} - \int_{t_0}^{t_1} \mathbf{g}_{\boldsymbol{\beta}}(\mathbf{z}(t), t, \mathbf{w}) dt$. For a given model, we can easily calculate the log-probability of regression output $\mathbf{y}$, given the input $\mathbf{x}$ [9]:

$$\log p(\mathbf{y}|\mathbf{w}) = \log p(\mathbf{f}_{\boldsymbol{\beta}}^{-1}(\mathbf{y}, \mathbf{w})) - \int_{t_0}^{t_1} \mathrm{Tr}\left(\frac{\partial \mathbf{g}_{\boldsymbol{\beta}}(\mathbf{z}(t), t, \mathbf{w})}{\partial \mathbf{z}(t)} \right) dt, \quad (5)$$

where $\mathbf{w} = \mathbf{k}_{\phi}(\mathbf{o})$, and $\mathbf{o} = \mathbf{h}_{\boldsymbol{\psi}}(\mathbf{x})$. With the model defined in the following way, we can easily calculate the exact value of log-probability for any possible regression outputs. We can also utilize the generative capabilities of the model by generating samples from a known prior $p(\mathbf{z})$ and transforming them into the space of regression outputs using the function given by eq. (4).

We aim at training the model by optimizing the NLL for a log probability defined by eq. (5) and the set of trainable parameters $\boldsymbol{\theta} = \{\boldsymbol{\psi}, \boldsymbol{\phi}, \boldsymbol{\beta}\}$. In the perfect scenario, we should optimize the entire
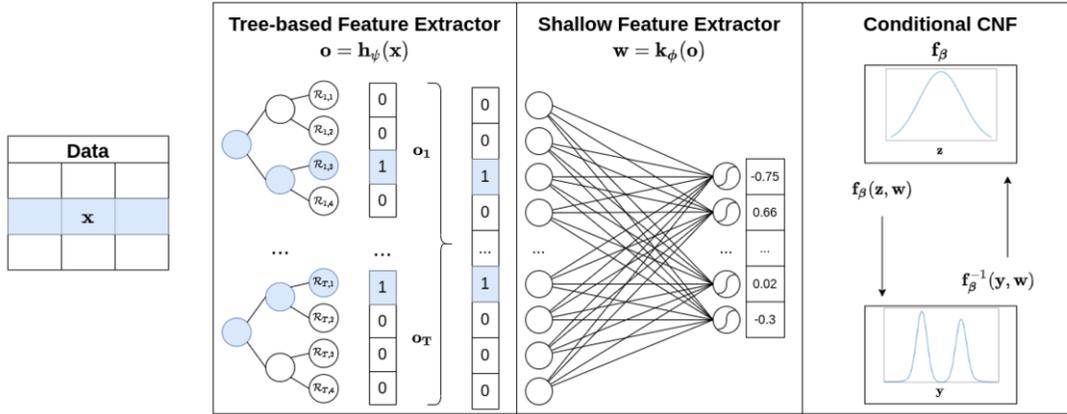
Figure 1. TreeFlow architecture. The proposed model consists of three components: Tree-based Feature Extractor, Shallow Feature Extractor, and Conditional CNF module. The role of the first component is to extract the vector of binary features from the structure of the tree-based ensemble model. The Shallow Feature Extractor is a neural network responsible for mapping high-dimensional binary vectors returned by the Tree-based Feature Extractor to low-dimensional feature space. The resulting vector is further passed to the conditional CNF module as a conditioning factor. The goal of the last component is to model complex probability distribution.

model in an end-to-end fashion, jointly updating the parameters of the Tree-based Feature Extractor ψ, Shallow Feature Extractor ϕ, and conditional CNF β. However, the Shallow Feature Extractor needs to have a constant size input which cannot be easily obtained from our Tree-based Feature Extractor as it learns iteratively. To overcome this limitation, we perform two-staged learning.

In the first stage, only the parameters of Tree-based Feature Extractor ψ^* are trained by optimizing the surrogate criterion specific to the type of tree-based architecture. In our work, we utilize the CatBoost model as it out-of-the-box supports categorical features and null values. Therefore, following [13] we train the Tree-based feature extractor by optimizing NLL loss for a standard Gaussian regression output given by eq. (1). For the multivariate case, we use the protocol from [20] and train the feature extractor by optimizing MultiRMSE.

Given the Tree-based Feature Extractor parameters, we train the remaining components of our model in an end-to-end fashion. Formally, given the estimated parameters ψ^* for $\mathbf{h}_\psi(\mathbf{x})$ we train the model by optimizing NLL with log-probability given by eq. (5) with respect to remaining parameters ϕ and β using the standard gradient-based approach.

The two-stage training has a couple of advantages compared to the end-to-end approach. First, any trained tree-based ensemble can be used as a feature extractor. Second, extracting the forest structure together with optimizing the parameters of the remaining components of the system in an end-to-end fashion is non-trivial and requires handcrafting the training procedure for a particular type of tree-based learner.

4 Related works

One of the best-known examples of gradient boosting methods is XGBoost [4] which iteratively combines weak regression trees to obtain accurate predictions. Further extensions to this method consist of LightGBM [10] and CatBoost [20] which introduce multiple novel techniques to obtain even better point estimates. Recently they have been extended to a probabilistic framework to model the whole probability distributions.

One such approach is NGBoost (Natural Gradient Boosting) [6] algorithm, which can model any probabilistic distribution with a defined probability density function, e.g., Univariate Gaussian, Exponential, or Laplace. It simultaneously estimates the distribution parameters by optimizing a proper scoring rule, e.g., negative log likelihood (NLL) or Continuous Ranked Probability Score (CRPS). The variant of NGBoost that utilizes Multivariate Gaussian to model multidimensional predictions was presented in [19]. RoNGBa [21] is an extension of NGBoost, which improves the performance of NGBoost via a better choice of hyperparameters. This framework has also been adapted to the CatBoost [13] with support to only univariate Gaussian distributions, but contrary to the NGBoost, the model outputs all distribution parameters from one model. There is also a group of approaches that were developed in parallel to NGBoost consisting of XGBoostLSS [15] and CatBoostLSS [16] which make a connection to well-established statistical framework Generalized Additive Models for Shape, Scale, and Location (GAMLSS) [26]. Like NGBoost, these models use one XGBoost or CatBoost model per parameter, but their training consists of two phases: independent model learning for each parameter and iterative parameter correction. One of the most recent approaches is Probabilistic Gradient Boosting Machine (PGBM) [25] which treats the leaf weights in each tree as random variables. This approach is capable to model different sets of posterior distributions but is limited to only distributions parameterized with location and scale parameters.

Besides the tree-based probabilistic models, several works investigate the problem of probabilistic regression. In [24] the authors model conditional density estimators for multivariate data with conditional sum-product networks that combines tree-based structures with deep models. In [7] the authors combine the transformer model with flows for density estimation. The flow models were also applied for future prediction problems in [29]. In [23] and [14] the authors propose to integrate flows with Gaussian Processes for probabilistic regression.

TreeFlow, to our best knowledge, is the first tree-based model for uni-, and multi-variate probabilistic regression, that is capable to model any distribution for regression outputs.

5   Experiments

This section evaluates our method on four different setups - univariate regression on synthetic data, univariate regression on mixed-type data, univariate regression on numerical data, and multivariate regression. Our goal is a quantitative and qualitative analysis of TreeFlow in comparison to the baseline models.

In all experiments, we measure target distribution fit using the negative log likelihood metric in the quantitative part. It is a natural choice as we expect to deal with heavy-tailed and multimodal distributions. Additionally, we calculate the CRPS metric which is defined as the mean squared difference between the forecasted probabilities and the actual outcomes, over all possible thresholds. It is not the best-suited metric for multimodal distributions, although it is often used for probabilistic forecasting and we would like to understand differences between TreeFlow and baselines. Moreover, we investigate point estimates that are usually necessary from the application point of view. For that purpose, we use the standard Root Mean Squared Error (RMSE) metric and introduce Root Mean Squared Error at K (RMSE@K) metric. The latter is a version of the RMSE metric that is adjusted for multimodal distributions and takes into account multiple predictions from the model. More details and justifications are provided in the Appendix in sec. A.1. In the qualitative part, we analyze and discuss the characteristics of obtained probability distributions. Finally, we perform the ablation study whose goal was to justify the design choices. The results of this part are presented in the Appendix in sec. D.

5.1   Univariate regression on synthetic data

This experiment is one of the motivating examples. Here, we want to evaluate the capabilities of TreeFlow to model data when the true probability distribution is known.

Dataset and methodology   We have created a dataset with two conditioning binary variables. For each possible combination of features, we have proposed different continuous distributions: Normal, Exponential, Mixture of Gaussians, and Gamma (see fig. 2 and details in Appendix, in sec. B). After that, we trained TreeFlow and CatBoost models. Finally, we calculated negative log likelihood and visualized the obtained probability distributions.

Results   After five repetitions of the experiment, we obtained negative log likelihood for CatBoost equal 2.52 ± 0.01, and for TreeFlow equal 2.02 ± 0.00. We can observe, that our method effortlessly obtained better results and, contrary to CatBoost, it was able to correctly model all probability distributions (see fig. 2). This is due to its flexibility in modeling probability distributions resulting from the usage of the CNF component.

5.2   Univariate regression on mixed-type data

Our goal is to evaluate and verify our approach to univariate regression problems with mixed-type data. This experiment is the main motivating example of this paper, as tree-based methods cannot model non-gaussian target distributions, and normalizing flows cannot deal with categorical variables without any additional data preparation step.
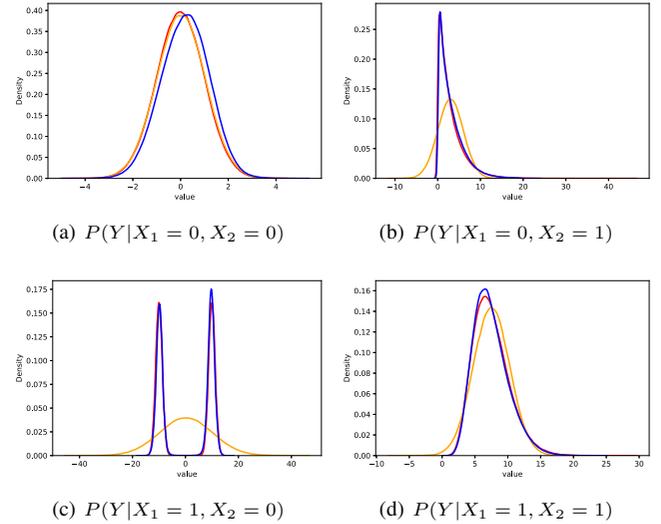
(a) $P(Y|X_1 = 0, X_2 = 0)$	(b) $P(Y|X_1 = 0, X_2 = 1)$
(c) $P(Y|X_1 = 1, X_2 = 0)$	(d) $P(Y|X_1 = 1, X_2 = 1)$

Figure 2.   Comparison of the estimated probability distributions for univariate regression on synthetic data experiment. We can observe that contrary to CatBoost, TreeFlow was able to correctly model all underlying true probability distributions. Legend: Red - True probability distribution; Blue - TreeFlow; Orange - CatBoost.

Datasets and methodology   To the best of our knowledge, there is no established standard benchmark for regression problems with mixed-type datasets. Thus, we propose seven datasets from the well-known data platform - Kaggle. They have various numbers of samples ranging from a few thousand to a hundred thousand, a different number of categorical and numerical variables. All details of the datasets can be found in tab. 6.

In terms of the methodology, we follow the standard 80%/20% training/testing holdout split. We also split the training dataset to train and validation datasets in the same proportion for the best epoch/iteration selection purposes. All experiments are run 5 times and results are averaged.

For obtaining point estimates from TreeFlow we analyze three approaches: (i) Samples averaging (Avg) - the simple average of samples, (ii) RMSE@1 - usage of the most probable sample, (iii) RMSE@2 - usage of the two most probable samples. Finally, we provide ablation studies regarding the design of the Tree-based Feature Extractor and the Shallow Feature Extractor (see Appendix sec. D).

Baselines   Currently, the only approach to work with such problems is a CatBoost which deals out-of-the-box with mixed-type datasets and support modeling target variable with Gaussian distribution. Additionally, we evaluate PGBM with one hot encoding for categorical variables as the representative method for standard tree methods without support for categorical variables. Moreover, this method is also capable of utilizing various parametric distributions. We perform the evaluation on both probabilistic (NLL, CRPS) and deterministic (RMSE / RMSE@K) metrics.

Results   The results of the conducted experiments for probabilistic metrics are provided in tab. 1 and for deterministic metrics in tab. 2. Our method obtains better negative log likelihood scores for most of the datasets and for most of them better CRPS values than reference methods. Furthermore, for the majority of datasets, there

Table 1. Comparison of TreeFlow with existing methods in terms of negative log likelihood (NLL) and Continuous Ranked Probability Score (CRPS) on univariate regression problems with mixed-type data. Our method outperformed both CatBoost and PGBM approaches on most of the datasets thanks to its flexibility in modeling non-gaussian distributions. One Hot Encoding was used for categorical variables for PGBM. Extended information about datasets is provided in tab. 6.

DATASET	NLL			CRPS		
	CATBOOST	PGBM	TREEFLOW	CATBOOST	PGBM	TREEFLOW
AVOCADO	-0.40 ± 0.01	-0.45 ± 0.01	**-0.47 ± 0.03**	0.0992 ± 0.0018	0.0870 ± 0.0013	**0.0854 ± 0.0024**
BIGMART	-0.05 ± 0.02	**-0.10 ± 0.02**	-0.08 ± 0.02	0.1270 ± 0.0021	**0.1259 ± 0.0023**	0.1294 ± 0.0027
DIAMONDS	-1.80 ± 0.02	-1.41 ± 0.76	**-1.94 ± 0.03**	0.0222 ± 0.0002	0.0447 ± 0.0474	**0.0210 ± 0.0005**
DIAMONDS 2	-1.89 ± 0.02	-1.24 ± 0.83	**-2.14 ± 0.05**	0.0217 ± 0.0002	0.0461 ± 0.0504	**0.0197 ± 0.0005**
LAPTOP	-0.89 ± 0.08	**-0.97 ± 0.09**	-0.74 ± 0.13	0.0572 ± 0.0049	**0.0474 ± 0.0034**	0.0563 ± 0.0043
PAK WHEEL	-1.40 ± 0.05	-0.53 ± 0.02	**-1.60 ± 0.03**	0.0362 ± 0.0006	0.0813 ± 0.0009	**0.0327 ± 0.0007**
SYDNEY	-0.54 ± 0.04	0.20 ± 1.02	**-0.66 ± 0.01**	0.0726 ± 0.0011	0.2383 ± 0.2646	**0.0721 ± 0.0008**

Table 2. Comparison of TreeFlow with existing methods in terms of root mean squared error (RMSE) on univariate regression problems with mixed-type data. TreeFlow in approach @2 significantly outperforms other baseline methods by taking advantage of multimodal distribution modeling property.

DATASET	RMSE				
	CATBOOST	PGBM	TREEFLOW(AVG)	TREEFLOW(@1)	TREEFLOW(@2)
AVOCADO	0.1939 ± 0.0043	**0.1624 ± 0.0024**	0.1676 ± 0.0058	0.1769 ± 0.0087	0.1713 ± 0.0066
BIGMART	0.2284 ± 0.0039	**0.2274 ± 0.0040**	0.2335 ± 0.0045	0.2514 ± 0.0087	0.2480 ± 0.0083
DIAMONDS	0.0419 ± 0.0007	0.0403 ± 0.0006	0.0407 ± 0.0009	0.0445 ± 0.0015	**0.0343 ± 0.0017**
DIAMONDS 2	0.0421 ± 0.0006	0.0492 ± 0.0010	0.0398 ± 0.0006	0.0460 ± 0.0014	**0.0364 ± 0.0004**
LAPTOP	0.1028 ± 0.0092	**0.0848 ± 0.0063**	0.1014 ± 0.0082	0.1015 ± 0.0076	0.0958 ± 0.0058
PAK WHEEL	0.0783 ± 0.0009	0.1630 ± 0.0018	0.0729 ± 0.0018	0.0796 ± 0.0021	**0.0654 ± 0.0047**
SYDNEY	0.1528 ± 0.0057	0.1561 ± 0.0047	0.1518 ± 0.0051	0.1721 ± 0.0041	**0.1361 ± 0.0066**

is a substantial improvement in the results. In terms of point estimates, TreeFlow in @2 approach obtains superior results in most of the datasets by the ability to provide multiple predictions for a particular sample that could be modeled by multimodal distributions. The detailed discussion regarding differences between point estimates for TreeFlow is provided in the Appendix in sec. A.1 Moreover, we investigated that the target distributions provided by TreeFlow had more realistic properties such as a heavy tail, multimodality, or does not provide any probability mass for impossible values, e.g., negative values when modeling price as a target variable. The latter example is presented in fig. 3. We analyzed estimated probability density functions for the Wine Reviews datasets for CatBoost and TreeFlow. Both methods predicted similar values for the PDF function; however, only TreeFlow was able to model heavy-tailed distribution and recognize that negative price values are highly unlikely.

5.3 Univariate regression on numerical data

We focus on univariate regression problems with only numerical variables in this setup. We aim to evaluate our method on standard probabilistic regression benchmarks in both probabilistic and deterministic approach. Finally, we investigate the properties of the obtained target distributions.

Datasets and methodology We use established in the reference methods [6, 13] probabilistic regression benchmark with the exclusion of the Boston dataset due to ethical issues. It contains nine varying-size datasets from the UCI Machine Learning Repository. We follow the same protocol as used in the reference papers. We create 20 random folds for all datasets except Protein (5 folds) and Year MSD (1 fold). We keep 10% of samples as a test set for each of these folds, and the remaining 90% of data we split into an 80%/20% train/validation for the best epoch selection purposes.
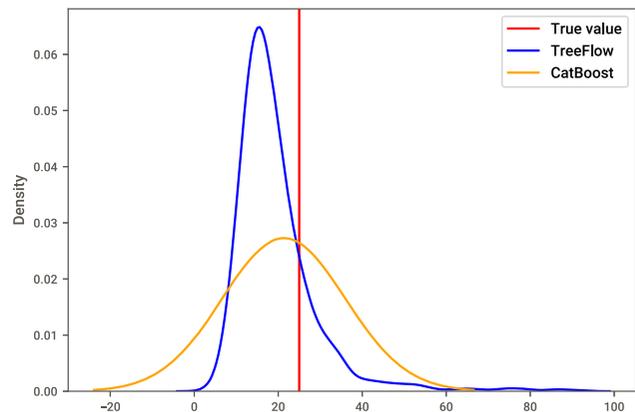

Figure 3. Estimated probability density functions for the Wine Reviews datasets from the univariate regression on mixed-type data experiment. Both methods predicted similar values for the PDF function; however, the properties of the obtained distributions are entirely different. Contrary to the CatBoost approach, TreeFlow was able to model heavy-tailed distribution and recognize that negative price values are highly unlikely.

Baselines We selected four tree-based baseline models: NGBoost [6], RoNGBa [21], CatBoost [13], PGBM [25], and one non tree-based method - Deep Ensemble [12] which was used in [13] as a reference method.

Results The quantitative results for negative log likelihood (NLL) are presented in tab. 3 and for RMSE in tab. 4. In terms of the probabilistic metric, our approach outperforms baseline methods on three

Table 3. Comparison of TreeFlow with existing methods in terms of negative log likelihood (NLL) on univariate regression problems with numerical data. Our method outperformed the other approaches on three datasets and obtained competitive results on others. The superior results were obtained thanks to our method's ability to model multimodal distributions.

DATASET	DEEP. ENS.	CATBOOST	NGBOOST	RONGBA	PGBM	TREEFLOW
CONCRETE	3.06 ± 0.18	3.06 ± 0.13	3.04 ± 0.17	2.94 ± 0.18	**2.75 ± 0.21**	3.02 ± 0.15
ENERGY	1.38 ± 0.22	1.24 ± 1.28	0.60 ± 0.45	**0.37 ± 0.28**	1.74 ± 0.04	0.85 ± 0.35
KIN8NM	**-1.20 ± 0.02**	- 0.63 ± 0.02	-0.49 ± 0.02	-0.60 ± 0.03	-0.54 ± 0.04	-1.03 ± 0.06
NAVAL	**-5.63 ± 0.05**	-5.39 ± 0.04	-5.34 ± 0.04	-5.49 ± 0.04	-3.44 ± 0.04	-5.54 ± 0.16
POWER	2.79 ± 0.04	2.72 ± 0.12	2.79 ± 0.11	2.65 ± 0.08	**2.60 ± 0.02**	2.65 ± 0.06
PROTEIN	2.83 ± 0.02	2.73 ± 0.07	2.81 ± 0.03	2.76 ± 0.03	2.79 ± 0.01	**2.02 ± 0.02**
WINE	0.94 ± 0.12	0.93 ± 0.08	0.91 ± 0.06	0.91 ± 0.08	0.97 ± 0.20	**-0.56 ± 0.62**
YACHT	1.18 ± 0.21	0.41 ± 0.39	0.20 ± 0.26	1.03 ± 0.44	**0.05 ± 0.28**	0.72 ± 0.40
YEAR MSD	3.35 ± NA	3.43 ± NA	3.43 ± NA	3.46 ± NA	3.61 ± NA	**3.27 ± NA**

Table 4. Comparison of TreeFlow with existing methods in terms of Root Mean Squared Error (RMSE) on univariate regression problems with numerical data.

DATASET	DEEP. ENS.	CATBOOST	NGBOOST	RONGBA	PGBM	TREEFLOW (AVG)	TREEFLOW (@1)	TREEFLOW (@2)
CONCRETE	6.03 ± 0.58	5.21 ± 0.53	5.06 ± 0.61	4.71 ± 0.61	**3.97 ± 0.76**	5.33 ± 0.65	5.41 ± 0.72	5.41 ± 0.71
ENERGY	2.09 ± 0.29	0.57 ± 0.06	0.46 ± 0.06	**0.35 ± 0.07**	**0.35 ± 0.06**	0.64 ± 0.11	0.66 ± 0.13	0.65 ± 0.12
KIN8NM	**0.09 ± 0.00**	0.14 ± 0.00	0.16 ± 0.00	0.14 ± 0.00	0.13 ± 0.01	**0.09 ± 0.00**	0.10 ± 0.01	0.10 ± 0.01
NAVAL	**0.00 ± 0.00**	**0.00 ± 0.00**	**0.00 ± 0.00**	**0.00 ± 0.00**	**0.00 ± 0.00**	**0.00 ± 0.00**	**0.00 ± 0.00**	**0.00 ± 0.00**
POWER	4.11 ± 0.17	3.55 ± 0.27	3.70 ± 0.22	3.47 ± 0.19	**3.35 ± 0.15**	3.71 ± 0.26	3.79 ± 0.26	3.79 ± 0.25
PROTEIN	4.71 ± 0.06	3.92 ± 0.08	4.33 ± 0.03	4.21 ± 0.06	3.98 ± 0.06	4.00 ± 0.27	4.79 ± 0.52	**3.01 ± 0.06**
WINE	0.64 ± 0.04	0.63 ± 0.04	0.62 ± 0.04	0.62 ± 0.05	0.60 ± 0.05	0.66 ± 0.05	0.73 ± 0.06	**0.41 ± 0.09**
YACHT	1.58 ± 0.48	0.82 ± 0.40	**0.50 ± 0.20**	0.90 ± 0.35	0.63 ± 0.21	0.75 ± 0.26	0.75 ± 0.25	0.75 ± 0.26
YEAR MSD	8.89 ± NA	8.99 ± NA	8.94 ± NA	9.14 ± NA	9.09 ± NA	9.29 ± NAN	10.97 ± NAN	**8.64 ± NA**

datasets: Protein, Wine, Year MSD, and obtains competitive results on others. For deterministic metrics, we obtain SOTA results for the same three datasets, and for two (kin8nm and naval) we achieve the same results as the current best methods.

To understand the results, we have investigated target distributions. We have compared them with the CatBoost model with default hyperparameters and presented them in fig. 4.

The first subfigure presents results for the Protein dataset. TreeFlow method has discovered that the underlying target distribution has a bimodal character and was able to correctly estimate the high value of the probability density function for the true value. In contrast, the Gaussian-based method did not have such an ability and incorrectly estimated the center of probability mass between two modes. The second subfigure is a representative of naturally occurring integer value datasets: Wine Quality and Year MSD. In this example, our method proposes a multimodal distribution consisting of Gaussian-like and heavy-tailed distributions. Such estimation gives us very rich information for the decision-making process compared to the Gaussian-based approach, which only estimated values around the highest mode and completely ignored information about a minor mode around 5 and a heavy tail for values 8 and 9. The last subfigure is a representative example of the rest of the datasets for which our method obtained similar results to baselines. Both methods proposed Gaussian distribution as a target distribution and assuming that this is a correct target distribution, there is no possibility of obtaining significantly better results.

The above-mentioned analysis also explains the results of the deterministic metrics. We incorporated into the decision-making process additional information about the second modality and it resulted in significant gains in prediction accuracy. To the best of our knowledge, it is the first time when these properties were noticed and exploited.

5.4 Multivariate regression

In the last setup, we focus on multivariate regression problems. Our goal is to quantitatively evaluate our method on datasets with various target dimensionality and examine the properties of obtained distributions.

Datasets and methodology Currently, the only tree-based probabilistic multivariate regression problem was approached by [19] which proposes a task of two-dimensional oceanographic velocities prediction [18]. Moreover, we evaluate our method on five more datasets with a broad range of target and feature dimensionality introduced in [28].

For both groups of datasets, we follow the proposed for these datasets experiment methodology. For the Oceanographic dataset, it is the same protocol as in the univariate regression on numerical data experiment. For the second group, it is a standard training/testing holdout split similar to the univariate regression on the mixed-type data experiment. The exact number of samples is provided in the tab. 5.

Baselines For this setup, we selected two baseline models. The first approach uses NGBoost, which assumes Multivariate Gaussian distribution and models correlation between target variables. The second approach also uses NGBoost, but the separate model models each target dimension; thus, it assumes independence between targets. We do not consider other Independent Gaussian approaches as they similarly model target distribution.

Results The results of the experiments are provided in tab. 5. Our method outperforms baselines by a large margin on three datasets. In contrast to NGBoost-based methods, TreeFlow was able to capture non-gaussianity in the target distributions. It can be evident on Parkinsons and US Flight datasets where differences between Independent NGBoost and Multivariate NGBoost were significant. They were probably caused by the ability to model the correlation between target variables and TreeFlow utilized its flexibility to obtain even better results. The other situation is for the Oceanographic dataset, where all results are close. Here, probably true target distribution is similar to the Independent Gaussian distribution; thus, NGBoost and TreeFlow can not achieve better results. In the last dataset - Energy,
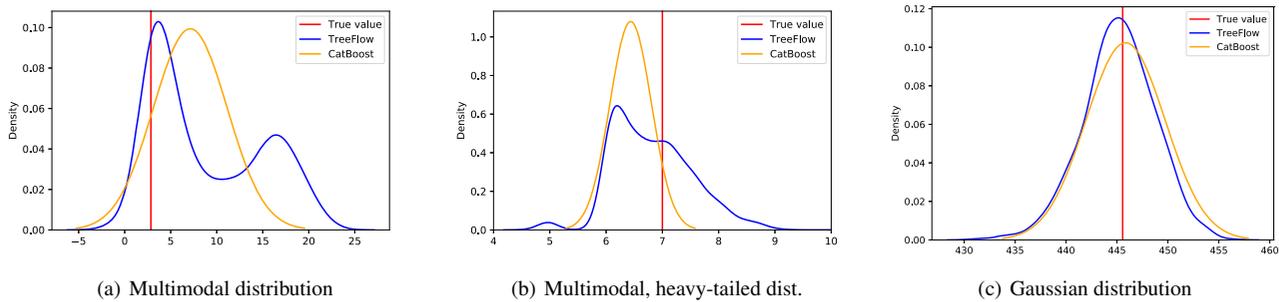
(a) Multimodal distribution   (b) Multimodal, heavy-tailed dist.   (c) Gaussian distribution

Figure 4. Estimated probability density functions for three datasets (Protein, Wine Quality, Power Plant) from the univariate regression on numerical data experiment. Depending on the dataset, TreeFlow is able to model distributions with various properties.

Table 5. Comparison of TreeFlow with existing methods in terms of negative log likelihood on multivariate regression problems. Our method obtains SOTA results on four out of the six datasets thanks to its flexibility in modeling complex distributions. Baseline results for the Oceanographic are taken from the reference paper.

DATASET	IND NGBOOST	NGBOOST	TREEFLOW
PARKINSONS	6.86	5.85	**5.26**
SCM20D	94.40	94.81	**93.41**
WINDTURBINE	-0.65	-0.67	**-2.57**
ENERGY	**166.90**	175.80	180.00
USFLIGHT	9.56	8.57	**7.49**
OCEANOGRAPHIC	7.74±0.02	**7.73±0.02**	7.84±0.01

the best performing model was Independent NGBoost. We suspect that the high dimensionality of the target distribution was too hard to learn for both NGBoost and TreeFlow methods.
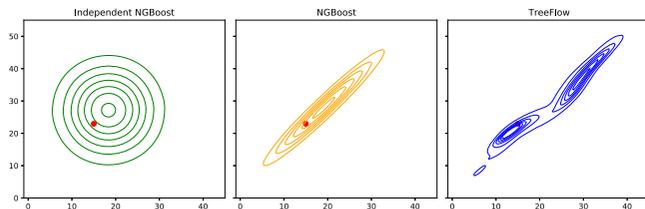

Figure 5. Estimated probability density functions for the Parkinsons datasets from the multivariate regression experiment. TreeFlow is the most flexible method which enables target distribution to correlate between dimensions and has multiple modes.

Moreover, we investigated target distributions for the Parkinsons dataset. The results of all methods are presented in fig. 5. We can easily observe how consecutive methods allow for more flexible distributions. Multivariate NGBoost enables correlation between target variables, while TreeFlow adds multimodality property.

6 Conclusions

In this work, we proposed a novel tree-based approach for probabilistic regression. Our method combines the benefits of ensemble decision trees with the capabilities of flow-based models in modeling complex non-Gaussian, multimodal distributions. We evaluated our approach using four experimental settings for both probabilistic and deterministic metrics and achieve SOTA or comparable results on most of them. We also illustrate some properties of TreeFlow that show the benefits of our approach compared to reference baselines.

Limitations The main trade-off introduced by our method is between computational time and the flexibility of the target distribution. Resource-demanding CNF component limits the scalability of our method, but despite this, we were able to deal with datasets of up to half a million observations. Additionally, our method has multiple hyperparameters, which may be challenging to tune in some cases, but we hope that a broad range of experiments provides good intuitions for end users (see sec. C). Lastly, TreeFlow performs two-staged learning, which might sometimes lead to sub-optimal results. Even so, our method outperforms current baselines, and we hope that TreeFlow will serve as a strong starting point for future end-to-end approaches.

Broader Impact The tree-based models are widely applied in research and industry and often achieve SOTA results. TreeFlow can be seen as an extension of such models, and all ethical considerations, both positive and negative, regarding regression problems apply to our work. However, as we consider target distribution more complex than parametric, our method can better assess uncertainty in the decision-making process or provide realistic probability distributions (see examples in fig. 3, 4, 5). Such properties might be crucial, for example, in medicine or finance applications, and have a largely positive societal impact.

Acknowledgements

Appendix and Code

The code and appendix are available under the GitHub repository: https://github.com/pfilo8/TreeFlow.

References

[1] Leo Breiman, 'Random forests', *Machine Learning*, **45**(1), 5–32, (2001).

[2] Leo Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth, 1984.

[3] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud, 'Neural ordinary differential equations', in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, eds., Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, pp. 6572–6583, (2018).

[4] Tianqi Chen and Carlos Guestrin, '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*, pp. 785–794. ACM, (2016).

[5] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio, 'Density estimation using Real NVP', in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, (2017).

[6] Tony Duan, Avati Anand, Daisy Yi Ding, Khanh K. Thai, Sanjay Basu, Andrew Y. Ng, and Alejandro Schuler, 'NGBoost: Natural Gradient Boosting for Probabilistic Prediction', in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 2690–2700. PMLR, (2020).

[7] Rasool Fakoor, Pratik Chaudhari, Jonas Mueller, and Alexander J Smola, 'Trade: Transformers for density estimation', *arXiv preprint arXiv:2004.02441*, (2020).

[8] Jerome H. Friedman, 'Greedy function approximation: A gradient boosting machine.', *Annals of Statistics*, **29**, 1189–1232, (2001).

[9] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud, 'FFJORD: free-form continuous dynamics for scalable reversible generative models', in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, (2019).

[10] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu, 'LightGBM: A Highly Efficient Gradient Boosting Decision Tree', in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 3146–3154, (2017).

[11] Diederik P. Kingma and Prafulla Dhariwal, 'Glow: Generative flow with invertible 1x1 convolutions', in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 10236–10245, (2018).

[12] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell, 'Simple and scalable predictive uncertainty estimation using deep ensembles', in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 6402–6413, (2017).

[13] Andrey Malinin, Liudmila Prokhorenkova, and Aleksei Ustimenko, 'Uncertainty in Gradient Boosting via Ensembles', in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, (2021).

[14] Juan Maroñas, Oliver Hamelijnck, Jeremias Knoblauch, and Theodoros Damoulas, 'Transforming gaussian processes with normalizing flows', in *International Conference on Artificial Intelligence and Statistics*, pp. 1081–1089. PMLR, (2021).

[15] Alexander März, 'XGBoostLSS - An extension of XGBoost to probabilistic forecasting', *CoRR*, **abs/1907.03178**, (2019).

[16] Alexander März, 'CatBoostLSS - An extension of CatBoost to probabilistic forecasting', *CoRR*, **abs/2001.02121**, (2020).

[17] Kevin P. Murphy, *Machine learning - a probabilistic perspective*, Adaptive computation and machine learning series, MIT Press, 2012.

[18] Michael O'Malley, 'North Atlantic Ocean Drifter Dataset for Multivariate Probabilistic Regression with Natural Gradient Boosting'. Zenodo, (2021).

[19] Michael O'Malley, Adam M. Sykulski, Rick Lumpkin, and Alejandro Schuler, 'Multivariate Probabilistic Regression with Natural Gradient Boosting', *CoRR*, **abs/2106.03823**, (2021).

[20] Liudmila Ostroumova Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin, 'CatBoost: unbiased boosting with categorical features', in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 6639–6649, (2018).

[21] Liliang Ren, Gen Sun, and Jiaman Wu, 'RoNGBa: A Robustly Optimized Natural Gradient Boosting Training Approach with Leaf Number Clipping', *CoRR*, **abs/1912.02338**, (2019).

[22] Danilo Jimenez Rezende and Shakir Mohamed, 'Variational Inference with Normalizing Flows', in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 1530–1538. JMLR.org, (2015).

[23] Marcin Sendera, Jacek Tabor, Aleksandra Nowak, Andrzej Bedychaj, Massimiliano Patacchiola, Tomasz Trzcinski, Przemysław Spurek, and Maciej Zieba, 'Non-Gaussian Gaussian Processes for Few-Shot Regression', in *NeurIPS*, (2021).

[24] Xiaoting Shao, Alejandro Molina, Antonio Vergari, Karl Stelzner, Robert Peharz, Thomas Liebig, and Kristian Kersting, 'Conditional sum-product networks: Imposing structure on deep probabilistic architectures', in *International Conference on Probabilistic Graphical Models*, pp. 401–412. PMLR, (2020).

[25] Olivier Sprangers, Sebastian Schelter, and Maarten de Rijke, 'Probabilistic gradient boosting machines for large-scale probabilistic regression', in *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pp. 1510–1520. ACM, (2021).

[26] D Mikis Stasinopoulos, Robert A Rigby, et al., 'Generalized additive models for location scale and shape (GAMLSS) in R', *Journal of Statistical Software*, (2007).

[27] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge J. Belongie, and Bharath Hariharan, 'PointFlow: 3D Point Cloud Generation With Continuous Normalizing Flows', in *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pp. 4540–4549. IEEE, (2019).

[28] Zhongjie Yu, Mingye Zhu, Martin Trapp, Arseny Skryagin, and Kristian Kersting, 'Leveraging probabilistic circuits for nonparametric multi-output regression', in *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence, UAI 2021, Virtual Event, 27-30 July 2021*, volume 161 of *Proceedings of Machine Learning Research*, pp. 2008–2018. AUAI Press, (2021).

[29] Maciej Zieba, Marcin Przewieźlikowski, Marek Śmieja, Jacek Tabor, Tomasz Trzcinski, and Przemysław Spurek, 'RegFlow: Probabilistic Flow-based Regression for Future Prediction', *CoRR*, **abs/2011.14620**, (2020).