Information Sciences 686 (2025) 121369

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Regression trees for fast and adaptive prediction intervals

Luben M.C. Cabezas [a,b,*], Mateus P. Otto [a,b], Rafael Izbicki [a], Rafael B. Stern [c]

[a] Department of Statistics, Federal University of São Carlos, São Carlos, 13566-590, São Paulo, Brazil
[b] Institute of Mathematical Sciences and Computation, University of São Paulo, São Carlos, 13565-905, São Paulo, Brazil
[c] Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 05508-090, São Paulo, Brazil

ARTICLE INFO

MSC:
62H30
1111

Keywords:
Supervised learning
Prediction intervals
Conformal prediction
Local calibration

ABSTRACT

In predictive modeling, quantifying prediction uncertainty is crucial for reliable decision-making. Traditional conformal inference methods provide marginally valid predictive regions but often produce non-adaptive intervals when naively applied to regression, potentially biasing applications. Recent advances using quantile regressors or conditional density estimators improve adaptability but are typically tied to specific prediction models, limiting their ability to quantify uncertainty around arbitrary models. Similarly, methods based on partitioning the feature space adopt sub-optimal strategies, failing to consistently measure predictive uncertainty across the feature space, especially in adversarial examples. This paper introduces a model-agnostic family of methods to calibrate prediction intervals for regression with local coverage guarantees. By leveraging regression trees and Random Forests, our approach constructs data-adaptive partitions of the feature space to approximate conditional coverage, enhancing the accuracy and scalability of prediction intervals. Our methods outperform established benchmarks on simulated and real-world datasets. They are implemented in the Python package `clover`, which integrates seamlessly with the *scikit-learn* interface for practical application.

1. Introduction

Predictive models often make mistakes, making it crucial to quantify the uncertainty associated with their predictions. Conformal inference has emerged as a powerful tool for creating statistically valid prediction regions around point predictions, providing marginal coverage guarantees even in a distribution-free setting [2,46,29]. However, the naive application of conformal methods to regression problems typically results in non-adaptive prediction regions, which may not provide accurate coverage for specific subgroups within the data [4]. This lack of calibration within specific potentially leads to biased decision-making [48].

Recently, conformal scores have been developed to address the lack of adaptation of prediction regions. However, they often rely on predictive models based on quantile regressors or conditional density estimators [41,23,12], and thus do not resolve the fundamental challenge of quantifying uncertainty around arbitrary predictive models. Other approaches such as Localized Conformal Prediction [20,1] and methods based on partitioning the feature space [6,7] have also been explored but fail to provide a satisfactory solution to this challenge. Partition-based methods, for instance, employ sub-optimal partitioning strategies that may result in poor adaptation. Local conformal prediction (LCP) requires adjusting the miscoverage level for each testing point to ensure coverage, which

* Corresponding author.
E-mail address: lucruz45.cab@gmail.com (L.M.C. Cabezas).

https://doi.org/10.1016/j.ins.2024.121369
Received 13 February 2024; Received in revised form 16 August 2024; Accepted 16 August 2024

Available online 22 August 2024

is sensitive to hyperparameter choice and lacks scalability. In Section 3, we further discuss these methods and illuminate where they fall short.

This paper introduces a new model-agnostic family of methods that calibrate prediction intervals for regression problems with local coverage guarantees. Our approach is innovative in its use of regression trees and Random Forests to partition the feature space, thereby approximating conditional coverage.

By training these tree-based models on conformity scores, we create partitions that adapt to the data's local structure, ensuring that prediction intervals are both accurate and scalable. This method stands out due to its versatility, as it can be applied to various conformity scores and prediction settings, and its superior performance when compared to established baselines in both simulated and real-world datasets [34,43].

Furthermore, we provide a practical implementation of our methods in a Python package called `clover`, which uses the standard *scikit-learn* interface. This allows for easy integration and application of our techniques in real-world scenarios, making our approach both theoretically robust and practically valuable.

To summarize, our main contributions are:

- We develop a family of methods to calibrate predictive regions with marginal and local coverage guarantees.
- We effectively use regression trees and Random Forests to partition the feature space optimally.
- We provide users with an efficient and easy-to-use implementation that can be promptly integrated into predictive modeling pipelines within the *scikit-learn* framework.
- We validate our methods through extensive experimentation and comparisons, demonstrating their effectiveness across a diverse set of real and simulated datasets.

The remainder of the paper is organized as follows. The mathematical goals of the paper are stated in Section 2. Section 3 reviews related work. Section 4 introduces the notation that will be used throughout the paper. We give an overview of Conformal Inference and Regression Trees/Forests in Section 5. In Section 6 we introduce our methods, discussing their properties and theoretical guarantees. In Section 7, we validate our approach in simulated and real-world datasets and compare it to baselines. We conclude with a brief discussion in Section 8. All proofs are deferred to Appendix B.

2. Problem setup

Let Y denote the label and $\mathbf{x}$ the features. We propose methods to construct local prediction intervals around point estimates of a regression function $\hat{\mu}(\mathbf{x})$. Specifically, our confidence sets take the form:

$$C(\mathbf{x}) = \left(\hat{\mu}(\mathbf{x}) - \hat{t}_{1-\alpha}(\mathbf{x}), \hat{\mu}(\mathbf{x}) + \hat{t}_{1-\alpha}(\mathbf{x})\right),$$

where $\hat{t}_{1-\alpha}$ are cutoffs learned from data that determine the coverage and width of the sets. Under minimal assumptions, our prediction sets $C(\mathbf{X})$ ensure that for a fixed $\alpha \in (0,1)$ set by the user, $\mathbb{P}[Y \in C(\mathbf{X})] = 1-\alpha$, even if the model used to construct C is misspecified. However, achieving this marginal coverage alone is often insufficient. For instance, the *conditional coverage* of conformal regions, given by $\mathbb{P}[Y \in C(\mathbf{X}) \mid \mathbf{X} = \mathbf{x}]$, can deviate significantly from $1-\alpha$ at various locations $\mathbf{x}$, which can create biases in downstream decisions that use these intervals. Unfortunately, in the distribution-free setting, it is impossible to construct predictive intervals $C(\mathbf{X})$ based on a finite number of samples that attain conditional coverage and have reasonable lengths [30,16].

Thus, we propose a hierarchy of methods to construct local prediction intervals around point estimates of a regression function $\hat{\mu}(\mathbf{x})$ that *approximately* achieve conditional coverage. Our methods interpolate between conformal and non-conformal procedures, yield calibrated intervals with theoretical guarantees, and exhibit improved scalability compared to literature baselines.

Our first method, which serves as a starting point in the hierarchy, is `Locart`. `Locart` is based on creating a partition $\mathcal{A}$ of the feature space and defining the cutoffs $\hat{t}_{1-\alpha}(\mathbf{x})$ by separately applying conformal prediction to each element of $\mathcal{A}$. Unlike conventional methodologies, the selection of $\mathcal{A}$ is guided by a data-driven optimization process designed to yield the most accurate estimates of the cutoffs. Specifically, the partition $\mathcal{A}$ is such that the output set of `Locart`, $C_{\texttt{Locart}}(\mathbf{X})$, satisfies

$$\mathbb{P}[Y \in C_{\texttt{Locart}}(\mathbf{X}) \mid \mathbf{X} \in A] \approx \mathbb{P}[Y \in C_{\texttt{Locart}}(\mathbf{X}) \mid \mathbf{X} = \mathbf{x}]$$

and guarantees local coverage,

$$\mathbb{P}[Y \in C_{\texttt{Locart}}(\mathbf{X}) \mid \mathbf{X} \in A] \geq 1-\alpha,$$

for every $A \in \mathcal{A}$. In other words, we expect `Locart` to have good conditional coverage. To achieve this property, we leverage regression trees, which are known to approximate conditional distributions [34].

In the next hierarchy level, we have `Loforest`. `Loforest` builds on `Locart` by using multiple regression trees on conformity scores to define its prediction interval. That is, `Loforest` is a Random Forest of trees on conformal scores. Although `Loforest` is not a conformal method, it shows excellent conditional coverage in practice (see Section 7).

The key innovation of our methods lies in utilizing regression trees to partition the feature space. As demonstrated through theoretical analysis and experimental results, this approach yields superior performance compared to other methods. This is because the partitioning closely approximates the optimal (but unknown) partitioning of the feature space. Moreover, it is faster than competing approaches that attempt to control conditional coverage by using regression trees.

3. Relation to other work

There are many non-conformal approaches for uncertainty quantification in regression. These include Gaussian Processes, quantile regression [28,34], the bootstrap, the jackknife, cross-validation, out-of-bag uncertainty estimates [10], among others (see Dey et al. [14] and Dewolf et al. [13]). However, these methods generally fail to provide even marginally valid predictive intervals in the finite-sample setting (see, for instance, Barber et al. [3] for the failure of jackknife and cross-validation).

In contrast, conformal inference methods always yield marginally valid prediction intervals [36,45,30,44]. Moreover, beyond marginal coverage, several conformal methods construct conditionally valid prediction intervals in the asymptotic regime. These methods often leverage consistent quantile regression estimators or rely on the design of sophisticated conformal scores whose conditional distribution (on $\mathbf{X}$) is independent of $\mathbf{X}$. Examples include conformalized quantile regression [41], distributional conformal prediction [12], Dist-split [22], and HPD-split [23]. Nevertheless, different from ours, these methods are not based on estimates of the regression function $\hat{\mu}(\mathbf{x})$, which makes them unsuitable for building prediction intervals centered at $\hat{\mu}(\mathbf{x})$.

A common goal of the aforementioned methods is to make prediction intervals adapt to the local structure of the data. In the regression setting, this is often sought by normalizing conformal scores with point-wise model uncertainty estimates. This procedure, known as normalization [36,38,24] or locally weighted conformal prediction [29], is similar to ours only in the sense that we also accomplish data-adaptivity. Our approach, however, seeks adaptation with an optimal level of "granularity", which, as discussed in Section 6, is not on the level of instances or point-wise. We give an example in Fig. 1 where the locally weighted regression split method [29] with a perfectly estimated "normalizer" fails to provide prediction intervals that are asymptotically conditionally valid. See Appendix A for details.

In fact, seeking adaptivity of prediction intervals on the level of instances is unfeasible, as constructing non-trivial conditionally valid prediction intervals is inherently hard in the finite-sample setting [30,16]. As a consequence, many approaches have instead focused on defining sensible collections of subsets (e.g., partitions) of the feature space. This is the case of [45], [30], [6], [7], and [16]. For instance, [30] partitions the feature space by creating a hyper-rectangular mesh, while Mondrian-based approaches [6,7] define a partition with a predefined Mondrian taxonomy [46]. For both these methods, the split conformal approach is applied to each partition element to control local coverage similarly to ours. However, our approach differs significantly in that the partition is induced by training a regression tree of conformity scores, allowing it to adapt to the data. We remark that, although [29,35] train regression models for the conformal scores, this is done within the normalization framework and thus inherits its limitations. Meanwhile, Mondrian taxonomies induced by binning certain uncertainty estimates [6], such as the conditional variance of the response [8], are suboptimal and fail to produce prediction intervals with asymptotic conditional coverage (see the example in Fig. 1 and details on Appendix A). Lastly, we are motivated by a goal similar to [15], where the authors sought to cluster instances with a similar distribution of conformal scores. Nonetheless, their clustering scheme is class-conditional (i.e., Y-conditional), while ours is strictly $\mathbf{X}$-conditional.

Localized conformal prediction (LCP) [20] is another framework for conformal inference in regression. LCP is based on weighting calibration samples according to their similarity to the test sample, as measured by a localizer. These similarity measures are then used to compute an estimate of the conditional (on $\mathbf{X}$) distribution of scores. In practice, because it requires storing the similarity matrix and recomputing the miscoverage level to rectify the induced non-exchangeability, LCP falls short in terms of computational scalability. Within the LCP framework, LCP-RF [1] is especially akin to our approach. LCP-RF leverages the adaptive nearest neighbor [31] characterization of Quantile Regression Forests [34] to estimate the conditional distribution of scores. Two significant distinctions set us apart. First, we do not include the test sample in the estimate of the conditional distribution of scores, and thus we do not have to adjust the miscoverage level (a trademark of LCP-based methods). Second, we do not construct the prediction intervals based on the estimated quantile of the Quantile Regression Forest. Instead, we estimate quantiles for every tree with a Quantile Regression Forest fitted on that tree and average these quantiles across all trees in the forest.

It is worth noticing that regression trees have been used in conformal inference within the context of model selection/aggregation [32] and predictive uncertainty quantification [35,25]. In the first, bootstrap or subsampling is used to reduce data splits [27,21] or to decrease the width of prediction intervals [18]. In the second, out-of-bag samples are used to estimate the prediction uncertainty of the ensemble and plugged in as normalizers [36,38,24] of conformal scores. Unlike our approach, these methods do not use regression trees to partition the feature space. As we will show, our method leverages this partition to create prediction intervals that are adaptive in an optimal sense, whereas the aforementioned methods produce sub-optimal intervals that, even if adaptive, lack good conditional coverage.

Finally, recent literature in conformal prediction seeks to extend the framework to handle various challenges, such as distribution shifts [26], time-series forecasting [47], and longitudinal data [5]. Overall, these papers are unrelated to our objective of constructing prediction intervals with good conditional coverage.

4. Notation

For an integer $n \geq 1$, we write $[n]$ to denote the set $\{1, \ldots, n\}$. We will denote by $\sqcup$ the union of disjoint sets. We use $\mathbb{1}\{\cdot\}$ for the indicator function. In addition, we denote by $\hat{q}_\phi(A)$ the empirical ϕ-quantile of the set $A = \{a_1, \ldots, a_n\}$; that is,

$$\hat{q}_\phi(A) = \inf \left\{ t : \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{a_i \leq t\} \geq \phi \right\}.$$

Finally, Table 1 summarizes the additional notation used throughout the manuscript.

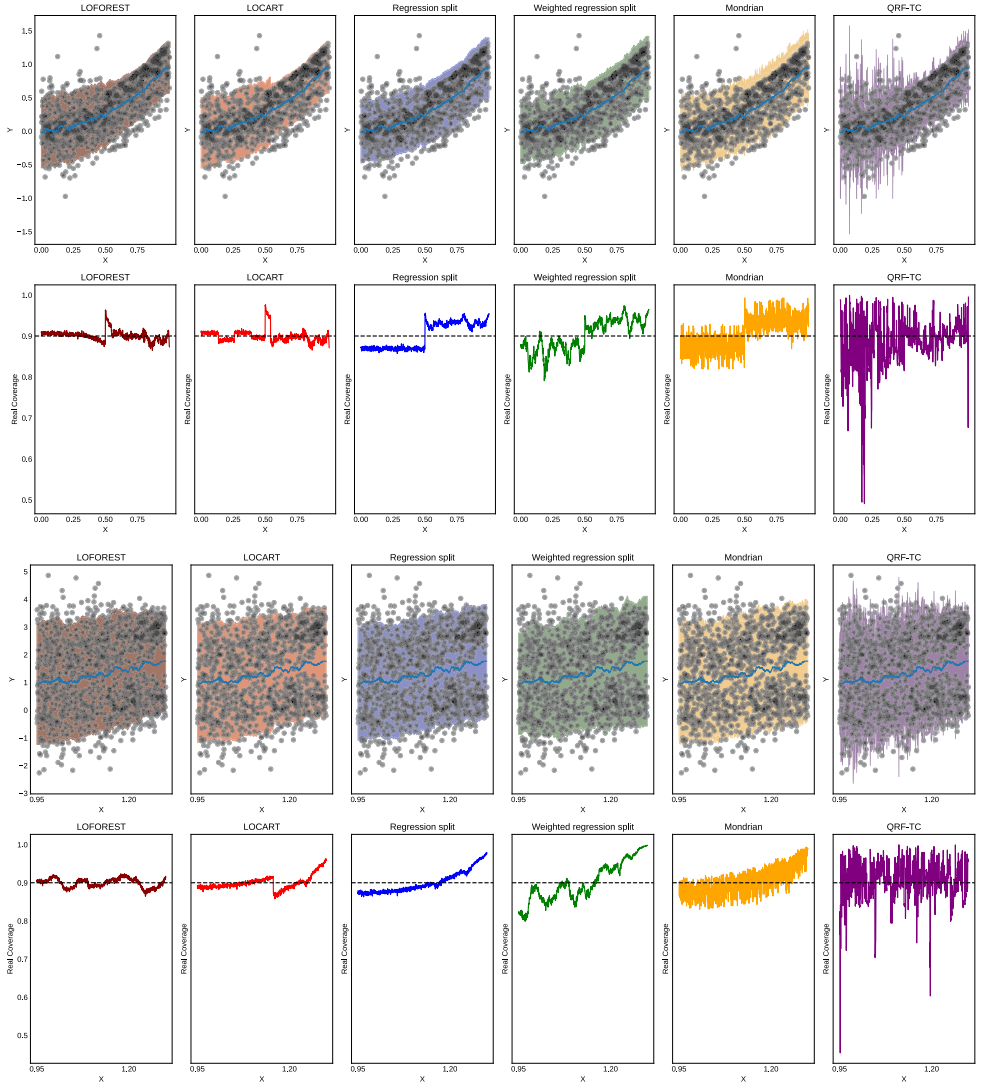

Fig. 1. Top: failure of locally weighted regression split: the conditional mean absolute deviation of $[Y - \hat{\mu}(X)]$ is constant, but the conditional quantiles of $|Y - \hat{\mu}(X)|$ are not. Bottom: failure of using "difficulty" as the conditional variance of Y: the conditional variance $\mathbb{V}(Y|X = x)$ is constant, but the conditional quantiles of $|Y - \hat{\mu}(X)|$ are not. In both cases, the intervals do not adapt to the data's local features and would fail to guarantee asymptotic conditional coverage. Our methods (`Locart` and `Loforest`), however, have good performance.

Table 1
Summary of notation.

$(\mathbf{X}_i, Y_i)_{i=1}^{n+1}$	Full dataset plus test instance
I_{train}	Indices of training split
I_{cal}	Indices of calibration split
I_{part}	Indices of split to create partition
I_{cut}	Indices of split to estimate cutoffs
$\hat{s} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$	Conformity score based on a estimator trained with I_{train}
$\hat{s}_i$	$\hat{s}_i = \hat{s}(\mathbf{X}_i, Y_i)$ for $i \in I$ with I a set of indices
C^*	The oracle prediction interval
$C_{\texttt{Locart}}$	The prediction interval of `Locart`
$C_{\texttt{Loforest}}$	The prediction interval of `Loforest`
$\mathcal{T} : \mathcal{X} \to \mathcal{A}$	Map from $\mathbf{x} \in \mathcal{X}$ to the partition element it belongs
$U \sim V$	U and V have the same probability distribution
$t^*_{1-\alpha}(\mathbf{x})$	Oracle cutoff on $\mathbf{x}$
$\hat{t}_{1-\alpha}(\mathbf{x})$	Estimated cutoff on $\mathbf{x}$
$\hat{\mu}(\mathbf{x})$	Regression estimator
$\mathcal{A}$	Partition of the feature space
$X_n = o_{\mathbb{P}}(1)$	$\lim_{n\to\infty} \mathbb{P}(X_n \geq \epsilon) = 0$, for all $\epsilon > 0$.

5. Preliminaries

In this section, we provide background on conformal inference, focusing on the split conformal prediction framework. Next, we revisit the construction of regression trees using the CART methodology and discuss their application as conditional distribution estimators.

5.1. Conformal inference

The main ingredient of the conformal inference framework is the conformity score $\hat{s} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, which measures how well the outputs of a regression or classification model accurately predict new labels $Y \in \mathcal{Y}$. In the regression setting, given an estimator $\hat{\mu}(\cdot)$ of the regression function $\mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$, a standard choice is the regression residual $\hat{s}(\mathbf{x}, y) = |y - \hat{\mu}(\mathbf{x})|$. Similarly, for the classification setting, given an estimator $\hat{\pi}_Y(\mathbf{x})$ of the true class probabilities $\pi_Y(\mathbf{x})$, one can use the generalized inverse quantiles from adaptive prediction sets [42] or its regularized version [2]. In what follows, to avoid unnecessary restriction to either regression or classification, we let the underlying predictive model implicit in the choice of the conformity score. In this sense, "training" a conformal score refers to training this underlying predictive model and plugging it into the definition of the chosen conformity score.[1]

A general method for obtaining prediction intervals in conformal inference with distribution-free guarantees is the split conformal prediction (split CP) framework [37,46]. Given exchangeable data $\{(\mathbf{X}_i, Y_i)\}_{i=1}^{n+1}$, split CP works as follows:

1. Split the data indices into two non-empty disjoint sets I_{train} and I_{cal}, such that $I_{\text{train}} \sqcup I_{\text{cal}} = [n]$.
2. Train a conformity score $\hat{s} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ on $\{(\mathbf{X}_i, Y_i)\}_{i \in I_{\text{train}}}$.
3. Evaluate $\hat{s}(\mathbf{X}_i, Y_i)$ for all $i \in I_{\text{cal}}$.
4. Calculate s^* as the empirical $\left(1 + 1/n_{\text{cal}}\right)(1 - \alpha)$-quantile of $\hat{s}_i = \hat{s}(\mathbf{X}_i, Y_i)$ for $i \in I_{\text{cal}}$, where n_{cal} is the size of I_{cal}.
5. For the new instance $(\mathbf{X}_{n+1}, Y_{n+1})$, define the split conformal prediction set as

$$C_{\text{split}}(\mathbf{X}_{n+1}) = \{y \in \mathcal{Y} : \hat{s}(\mathbf{X}_{n+1}, y) \leq s^*\}. \tag{1}$$

That is, the prediction set is taken to be the set of all y's that have a small conformity score.

Because of the exchangeability of the data, the splitting step guarantees that the scores $\hat{s}_i$ for $i \in I_{\text{cal}} \sqcup \{n+1\}$ are also exchangeable. As a consequence, the rank of $\hat{s}(\mathbf{X}_{n+1}, Y_{n+1})$ among these $n+1$ conformity scores is drawn uniformly from the $n+1$ possible indices. By choosing a cutoff s^* as the empirical $\left(1 + 1/n_{\text{cal}}\right)(1 - \alpha)$-quantile of the scores, we ensure that $\mathbb{P}[Y_{n+1} \in C_{\text{split}}(\mathbf{X}_{n+1})] = \mathbb{P}[\hat{s}(\mathbf{X}_{n+1}, Y_{n+1}) \leq s^*] \geq 1 - \alpha$. The following theorem formalizes this explanation and asserts that, conditional on the training data (used to construct $\hat{s}$) and on average over all possible calibration datasets with n_{cal} elements, the prediction band C_{split} is not too wide provided n_{cal} is sufficiently large. (See [45] for guarantees that incorporate the randomness of both the training and calibration datasets).

Theorem 1 *(Lei et al. [29]). Given an exchangeable sequence $(\mathbf{X}_i, Y_i)_{i=1}^{n+1}$ and a miscoverage level $\alpha \in (0, 1)$,*

$$\mathbb{P}[Y_{n+1} \in C_{split}(\mathbf{X}_{n+1})] \geq 1 - \alpha,$$

where C_{split} is the prediction band from the split conformal framework (1). In addition, if the conformal scores are almost surely distinct, then

$$\mathbb{P}[Y_{n+1} \in C_{split}(\mathbf{X}_{n+1})] \leq 1 - \alpha + \frac{1}{n_{cal} + 1}.$$

The split conformal framework can be naturally extended to produce prediction bands with local validity [46,30]. Let $\mathcal{A} = \{A_1, \ldots, A_K\}$ be a finite partition of the feature space, and consider $\mathcal{T} : \mathcal{X} \to \mathcal{A}$ as the function that maps $\mathbf{X}$ to the partition element it belongs to. The extension works as follows: given the initial split I_{train} and I_{cal}, we further split I_{cal} into the sets $I_j = \{i \in I_{\text{cal}} : \mathcal{T}(\mathbf{X}_i) = A_j\}$ for $j = 1, \ldots, K$. As before, the splitting step guarantees that the scores $\hat{s}_i$ for $i \in I_j \sqcup \{n+1\}$ are exchangeable. In complete analogy to split CP, for each split I_j, we calculate s_j^* as the empirical $(1 + 1/n_j)(1 - \alpha)$-quantile of the scores $\hat{s}_i$ for $i \in I_j$, where n_j is the size of I_j. Thus, we obtain that $\mathbb{P}[Y_{n+1} \in C_{\text{local}}(\mathbf{X}_{n+1}) | \mathbf{X}_{n+1} \in A_j] \geq 1 - \alpha$, where

$$C_{\text{local}}(\mathbf{X}_{n+1}) = \{y \in \mathcal{Y} : \hat{s}(\mathbf{X}_{n+1}, y) \leq s_j^*\} \tag{2}$$

is the local split conformal prediction set. If I_j is an empty set, we take the convention that $s^* = +\infty$. Theorem 2 formalizes this discussion by stating the local validity of the prediction intervals (2).

Theorem 2. *Let $\mathcal{A} = \{A_1, \ldots, A_K\}$ be a finite partition of the feature space. Given an exchangeable sequence $(\mathbf{X}_i, Y_i)_{i=1}^{n+1}$ and a miscoverage level $\alpha \in (0, 1)$,*

[1] We write $\hat{s}$ (with hat) to reinforce this dependence on an underlying trained model.

$$\mathbb{P}[Y_{n+1} \in C_{local}(\mathbf{X}_{n+1}) | \mathbf{X}_{n+1} \in A_j] \geq 1 - \alpha,$$

for all $j = 1, \ldots, K$*, where* C_{local} *is the prediction band in* (2).

Applying the split CP framework to each partition element incurs a cost. If partition elements are not well-populated, prediction intervals grow wide. To see this, suppose we are given a realization of the dataset. If $\alpha = 0.05$, then any partition element containing less than $\bar{n} = 19$ instances will have trivial local bands, with $C_{\text{local}}(\cdot) = \mathbb{R}$. Therefore, the extension of split CP with local validity must be combined with a partition of the feature space whose elements are guaranteed to contain a sufficient number of instances from the calibration split.

5.2. Regression trees

Suppose we want to create an estimator $\hat{\mu}(\mathbf{x})$ of the regression function $\mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$ given access to i.i.d. samples $\{(\mathbf{X}_i, Y_i)\}_{i=1}^n$. Regression trees build $\hat{\mu}$ by partitioning the feature space $\mathcal{X}$ into sets of the form $\{X_j \leq \ell\}$ and $\{X_j > \ell\}$, where X_j is a feature and ℓ is termed a level. These features and levels are chosen to minimize the sample variance of Y within each partition element. However, a "global" partitioning strategy is unfeasible due to the combinatorial hardness of testing all possible combinations of splitting features and levels. Instead, the partition is chosen by optimizing a greedy or local objective.

A standard way to greedily grow the tree is through the CART methodology [9]. Given a node t, starting from $\mathrm{t} = \mathrm{t}_1 = \mathcal{X}$ in the first step, the CART algorithm operates as follows:

1. Consider the left and right splits of t,

$$\mathrm{t}_L(j, \ell) = \{\mathbf{X} \in \mathrm{t} : X_j \leq \ell\} \quad \text{and} \quad \mathrm{t}_R(s, \ell) = \{\mathbf{X} \in \mathrm{t} : X_j > \ell\}.$$

2. Determine the split $(\hat{\jmath}, \hat{\ell})$ such that

$$\frac{1}{N(\mathrm{t})}\left(\sum_{\mathbf{X}_i \in \mathrm{t}_L} (Y_i - \overline{Y}_{\mathrm{t}_L})^2 + \sum_{\mathbf{X}_i \in \mathrm{t}_R} (Y_i - \overline{Y}_{\mathrm{t}_R})^2\right)$$

   is minimized, where $N(\mathrm{t}) = |\{\mathbf{X} \in \mathrm{t}\}|$ and $\overline{Y}_{\mathrm{t}} = \frac{1}{N(\mathrm{t})} \sum_{\mathbf{X}_i \in \mathrm{t}} Y_i$.
3. Recurse the procedure on t_L and t_R until the maximum depth K is reached or some other stopping criterion is satisfied.

If the maximum depth K of the tree is set to a high value, trees may overfit. This occurs after successive splittings of the training data since each terminal node of a tree may contain only a single data instance. To avoid this, the tree may be pruned as the training unrolls or afterward. In the former case, called pre-pruning, the growth of the tree is controlled by setting the minimal number of samples required for splitting internal nodes and for forming terminal nodes (via the `min_samples_split` and `min_samples_leaf` hyperparameters, respectively, in the *scikit-learn* [39] implementation). In the latter case, called post-pruning, subtrees or nodes are removed until a balance between model complexity and predictive performance is reached.

After training and pruning, the tree has a set of terminal nodes (or leaves) $\mathcal{A}$ that can be naturally identified as a partition of the feature space. With this, define $\mathcal{T} : \mathcal{X} \to \mathcal{A}$ as the function that maps a new instance $\mathbf{x}_{n+1} \in \mathcal{X}$ to the partition element it belongs. Then, the regression function estimate at $\mathbf{x}_{n+1}$ is $\hat{\mu}(\mathbf{x}_{n+1}) = \overline{Y}_{\mathcal{T}(\mathbf{x}_{n+1})}$. It is also possible to view this estimate as a weighted nearest-neighbor method [31]; defining the weighting function

$$w_i(\mathbf{x}) = \frac{\mathbb{1}\{\mathbf{X}_i \in \mathcal{T}(\mathbf{x})\}}{N(\mathcal{T}(\mathbf{x}))},$$

the regression function estimate can be rewritten as

$$\hat{\mu}(\mathbf{x}_{n+1}) = \sum_{i=1}^{n} w_i(\mathbf{x}_{n+1}) Y_i,$$

which highlights the data-adaptive nature of regression trees because the weight function assigns higher weights to training points closer to the new instance $\mathbf{x}_{n+1}$. This property allows one to construct tree-based estimators of quantities beyond the conditional mean [11]. Indeed, Meinshausen [34] shows that, under some mild assumptions, the function

$$\hat{F}(y|\mathbf{X} = \mathbf{x}_{n+1}) = \sum_{i=1}^{n} w_i(\mathbf{x}_{n+1}) \mathbb{1}\{Y_i \leq y\} \tag{3}$$

is a consistent estimator of the conditional distribution $Y|\mathbf{X} = \mathbf{x}$ for all $\mathbf{x} \in \mathcal{X}$. Although the rate of convergence of $\hat{F}$ to the true distribution function might be possibly improved by bagging multiple trees, as done in Quantile Regression Forests [34], the consistency holds even for a single tree. We will explore this fact to show the asymptotic conditional coverage property of `Locart`.

6. Methodology

In this section, we introduce our procedures for calibrating predictive intervals. First, building on the construction of the oracle prediction interval, we present an optimal partitioning scheme for local conformal. Then, we propose Local Coverage Regression Trees (`Locart`) as a realization of this scheme. Next, we introduce Local Coverage Regression Forests (`Loforest`), which is a non-conformal alternative to `Locart` based on averaging cutoff estimates that exhibit improved empirical performance. Then, we introduce the augmented versions of both algorithms, `A-Locart` and `A-Loforest`, that can further refine the partitions by augmenting the feature space. Finally, we present a weighted version of `Loforest` called `W-Loforest` that can be used to improve locally weighted prediction intervals [29].

6.1. Motivation

We consider prediction sets of the form:

$$C(\mathbf{x}) = (\hat{\mu}(\mathbf{x}) - \hat{t}_{1-\alpha}(\mathbf{x}), \hat{\mu}(\mathbf{x}) + \hat{t}_{1-\alpha}(\mathbf{x})).$$

where $\hat{\mu}(\mathbf{x})$ is the regression estimator and $\hat{t}_{1-\alpha}(\mathbf{x})$ is a cutoff parameter that needs to be determined. Our objective is to choose $\hat{t}_{1-\alpha}(\mathbf{x})$ such that, for a given miscoverage level $\alpha \in (0,1)$, the prediction set $C(\mathbf{x})$ controls the conditional coverage. The prediction set that uses this optimal cutoff is termed the *oracle prediction interval*:

Definition 1 *(Oracle prediction interval).* Given a regression estimator $\hat{\mu}$, the oracle prediction interval $C^*(\mathbf{x})$ is defined as

$$C^*(\mathbf{x}) := (\hat{\mu}(\mathbf{x}) - t^*_{1-\alpha}(\mathbf{x}), \hat{\mu}(\mathbf{x}) + t^*_{1-\alpha}(\mathbf{x})),$$

where $t^*_{1-\alpha}(\mathbf{x})$ is the $(1-\alpha)$-quantile of the regression residual $\hat{s}(\mathbf{X}, Y) = |Y - \hat{\mu}(\mathbf{X})|$ conditional on $\mathbf{X} = \mathbf{x}$. This is the only symmetric interval around $\hat{\mu}(\cdot)$ such that

$$\mathbb{P}[Y \in C^*(\mathbf{X}) | \mathbf{X} = \mathbf{x}, \hat{\mu}] = 1 - \alpha.$$

However, in practice, obtaining $\hat{t}^*_{1-\alpha}(\mathbf{x})$ is infeasible due to the unavailability of the true regression residual distribution. Thus, to approximate $\hat{t}^*_{1-\alpha}(\mathbf{x})$, we employ a local conformal approach, which involves calibrating the cutoff $\hat{t}_{1-\alpha}(\mathbf{x})$ locally within partitions of the feature space. The effectiveness of this approximation hinges on designing partitions $\mathcal{A}$ such that all x values within a given partition share a similar optimal cutoff. This ensures that:

$$\mathbb{P}[Y \in C(\mathbf{X}) | \mathbf{X} = \mathbf{x}, \hat{\mu}] \approx \mathbb{P}[Y \in C(\mathbf{X}) | \mathbf{X} \in A, \hat{\mu}] = 1 - \alpha,$$

thereby achieving local and near-conditional coverage.

One straightforward approach to partitioning the feature space is using Euclidean partitions, where the feature space is divided into regions that shrink as the dimensionality increases, eventually resulting in single-point partitions [30]. However, Euclidean partitions may not be optimal because they do not account for the possibility that two distant points in the feature space may share the same optimal cutoff, as the following examples show:

Example 1 *(Location family).* Let $h(y)$ be a density, $\mu(\mathbf{x})$ a function, and $Y|\mathbf{x} \sim h(y - \mu(\mathbf{x}))$. In this case, $t^*_{1-\alpha}(\mathbf{x}_a) = t^*_{1-\alpha}(\mathbf{x}_b)$, for every $\mathbf{x}_a, \mathbf{x}_b \in \mathbb{R}^d$. For instance, if $Y|\mathbf{x} \sim N(\beta^t \mathbf{x}, \sigma^2)$, then all instances would have the same optimal cutoff. Thus, in this specific scenario, a trivial partition $\mathcal{A} = \mathcal{X}$ would already lead to conditional validity as the calibration sample increases.

Example 2 *(Irrelevant features).* If $\mathbf{x}_S$ is a subset of the features such that the conditional densities satisfy $f(y|\mathbf{x}) = f(y|\mathbf{x}_S)$, then $t^*_{1-\alpha}(\mathbf{x}) = t^*_{1-\alpha}(\mathbf{x}_S)$, that is, the optimal cutoff does not depend on the irrelevant features, S^c. These irrelevant features, however, affect the Euclidean distance between $\mathbf{x}$'s.

Therefore, even if two $\mathbf{x}$ values are distant, they can still be included in the same A if they share the same $t^*_{1-\alpha}(\mathbf{x})$ value. This is equivalent to saying that the optimal partition should group instances with the same conditional distribution of the residuals $\hat{s}(\mathbf{X}, Y)|\mathbf{X} = \mathbf{x}$.

Theorem 3 formalizes the construction and optimality of such partition.

Theorem 3 *(The coarsest partition with same oracle cutoffs). Let $\mathcal{A}$ be the partition of $\mathcal{X}$ such that, for any $A \in \mathcal{A}$, $\mathbf{x}_1, \mathbf{x}_2 \in A$ if and only if $\hat{s}(\mathbf{X}, Y)|\mathbf{X} = \mathbf{x}_1 \sim \hat{s}(\mathbf{X}, Y)|\mathbf{X} = \mathbf{x}_2$, where $\hat{s}(\mathbf{X}, Y) = |Y - \hat{\mu}(\mathbf{X})|$. Let $t^*_{1-\alpha}(\mathbf{x})$ be the $(1-\alpha)$-quantile of the regression residual, as in Definition 1. Then,*

1. *If $\mathbf{x}_1, \mathbf{x}_2 \in A$, then $t^*_{1-\alpha}(\mathbf{x}_1) = t^*_{1-\alpha}(\mathbf{x}_2)$ for every $\alpha \in (0,1)$, that is, $\mathbf{x}_1$ and $\mathbf{x}_2$ have the same optimal cutoff.*

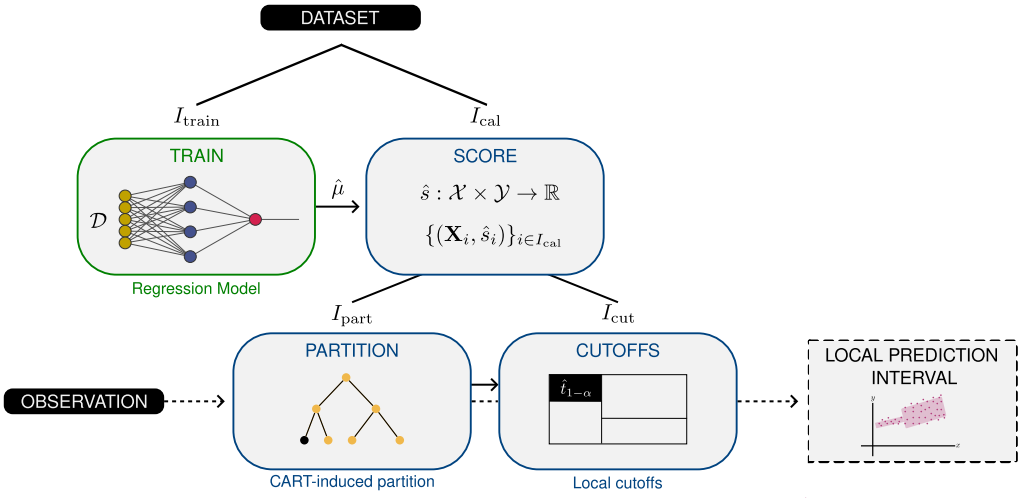

Fig. 2. Schematic diagram of `Locart`. Train: a regression model (e.g., neural network) is fitted on the I_{train} split, producing the estimator $\hat{\mu}$. Score: the regression residuals are calculated on I_{cal} and a new dataset of features and residuals produced. This dataset is split into I_{part}, used to create a partition of the feature space by fitting a regression tree of residuals, and I_{cut}, used to populate this partition and calculate cutoffs that calibrate prediction intervals locally. In each partition element, the residuals are (approximately) uniformly distributed. As a result, the local thresholds computed in each leaf will converge to the oracle threshold in the asymptotic regime. The path of a new observation (**- - ·**): the partition element it belongs to is identified; then, the local cutoff is retrieved; next, this cutoff is used to create an adaptive interval centered at the prediction of the regression model for this observation.

2. *If $\mathcal{J}$ is another partition of $\mathcal{X}$ such that, for any $J \in \mathcal{J}$, $\mathbf{x}_1, \mathbf{x}_2 \in J$ implies that $t^*_{1-\alpha}(\mathbf{x}_1) = t^*_{1-\alpha}(\mathbf{x}_2)$ for every $\alpha \in (0,1)$, then*

$$\mathbf{x}_1, \mathbf{x}_2 \in J \implies \mathbf{x}_1, \mathbf{x}_2 \in A,$$

that is, if $\mathbf{x}_1$ and $\mathbf{x}_2$ share the same optimal cutoff, they must belong to the same element of $\mathcal{A}$.

`Locart` attempts to recover the partition described in this theorem by using regression trees to form partitions based on the residuals, ensuring that all instances within a partition element share a similar distribution of residuals. `Loforest` extends this idea by averaging cutoffs across multiple trees, leading to more robust and empirically effective prediction intervals. We detail these approaches in the next section.

6.2. `Locart`: *local coverage regression trees*

A regression tree naturally induces a partition of the feature space. Moreover, as discussed in Section 5.2, regression trees are consistent estimators of conditional distributions. Thus, a regression tree that predicts the residual $\hat{s}(\mathbf{X}, Y)$ using $\mathbf{x}$ as an input attempts to recover the partition described by Theorem 3, that is, the partition that groups features according to the conditional distribution of the residuals. This is, therefore, how `Locart` builds the partition $\mathcal{A}$. We detail `Locart`'s algorithm and practical considerations in Section 6.2.1 and formalize its theoretical properties in Section 6.2.2.

6.2.1. *The algorithm*

Given a regression estimator $\hat{\mu}(\cdot)$ trained on $\{(\mathbf{X}_i, Y_i)\}_{i \in I_{\text{train}}}$ and the calibration split with indices I_{cal}, `Locart` consists of four steps, as outlined in Fig. 2:

1. Obtain the regression residuals $\hat{s}_i$ of $\hat{\mu}$ on I_{cal}, that is, compute $\hat{s}_i = \hat{s}(\mathbf{X}_i, Y_i) = |\hat{\mu}(\mathbf{X}_i) - Y_i|$, for every $i \in I_{\text{cal}}$.
2. Create the dataset $\{(\mathbf{X}_i, \hat{s}_i)\}_{i \in I_{\text{cal}}}$ of pairs of features and regression residuals computed on the calibration dataset, and split it into two disjoint sets I_{part} and I_{cut}.
3. Fit a regression tree on $\{(\mathbf{X}_i, \hat{s}_i)\}_{i \in I_{\text{part}}}$ that predicts the residuals $\hat{s}$ based on the features $\mathbf{X}$. This tree induces a partition $\mathcal{A}$ of the feature space. Let $\mathcal{T} : \mathcal{X} \to \mathcal{A}$ be the function mapping an element of $\mathcal{X}$ to the partition element it belongs.
4. Estimate $\hat{t}_{1-\alpha}(\mathbf{x}_{n+1})$ for a new instance $\mathbf{x}_{n+1}$ by applying the conformal approach to all instances in $\{(\mathbf{X}_i, s_i)\}_{i \in I_{\text{cut}}}$ that fall into the same element of $\mathcal{A}$ as $\mathbf{x}_{n+1}$ does. That is, we estimate $\hat{t}_{1-\alpha}(\mathbf{x}_{n+1})$ as

$$\hat{t}_{1-\alpha}(\mathbf{x}_{n+1}) = \hat{q}_{1-\alpha}(A(\mathbf{x}_{n+1})) \tag{4}$$

where

$$A(\mathbf{x}_{n+1}) = \left\{ \hat{s}_i : \mathcal{T}(\mathbf{X}_i) = \mathcal{T}(\mathbf{x}_{n+1}), (\mathbf{X}_i, \hat{s}_i) \in \{(\mathbf{X}_i, \hat{s}_i)\}_{i \in I_{\text{cut}}} \right\}, \tag{5}$$

is the set of all conformal scores of all calibration instances that fall into the same element of the partition as $\mathbf{x}_{n+1}$ does.

5. Using $\hat{t}_{1-\alpha}(\mathbf{x}_{n+1})$ from Equation (4), define the `Locart` prediction interval as

$$C_{\texttt{Locart}}(\mathbf{x}) = (\hat{\mu}(\mathbf{x}) - \hat{t}_{1-\alpha}(\mathbf{x}), \hat{\mu}(\mathbf{x}) + \hat{t}_{1-\alpha}(\mathbf{x}))$$

When using the regression tree algorithm to partition, we may need to control tree growth to avoid empty or redundant partitions. To do that, we perform pre and post-pruning. The pre-pruning is done by fixing the `min_samples_split` hyperparameter to a value, such as 100 samples, which enables us to obtain well-populated leaves resulting in accurate cutoff estimations. For the post-pruning step, we remove extra leaves and nodes via cost-complexity pruning. This process aims to balance partition complexity and predictive performance by reducing the amount of less informative partition elements.

The remaining regression tree hyperparameters are fixed at *scikit-learn*'s defaults. As a result, we can construct flexible and adaptive partitions without the need for hyperparameter tuning, which requires multiple refits and could significantly increase the algorithm's running time. Plus, as we make use of the *scikit-learn*'s efficient CART implementation, `Locart` often outperforms benchmarks in the running time comparison (see Section 7). It is also worth noticing that, to increase `Locart`'s power and circumvent sample size reduction in the calibration step, the splitting of the calibration set presented in Step 2 can be skipped, as it is only necessary to derive the theoretical properties explored in Section 6.2.2. The empirical results in Section 7 reinforce this observation, as `Locart` presents good performances without splitting the calibration set.

6.2.2. Theoretical properties

By construction of the algorithm, if we fix the partition index set I_{part}, `Locart` produces a fixed partition $\mathcal{A}$ of the feature space, and thus, satisfies Theorem 2 (i.e., it is locally valid). Consequently, it also satisfies marginal validity. Corollaries 1 and 2 formalize both statements:

Corollary 1 *(Local coverage property of `Locart`). Given an i.i.d. sequence $(\mathbf{X}_i, Y_i)_{i=1}^{n+1}$ and a fixed partition $\mathcal{A}$ of the feature space $\mathcal{X}$ induced by `Locart` using I_{part}, the predictive intervals of `Locart` satisfy:*

$$\mathbb{P}[Y_{n+1} \in C_{\texttt{Locart}}(\mathbf{X}_{n+1}) | \mathbf{X}_{n+1} \in A, I_{part}, I_{train}] \geq 1 - \alpha,$$

for all $A \in \mathcal{A}$.

Corollary 2 *(Marginal coverage property of `Locart`). Given an exchangeable sequence $(\mathbf{X}_i, Y_i)_{i=1}^{n+1}$, the predictive intervals of `Locart` satisfy:*

$$\mathbb{P}[Y_{n+1} \in C_{\texttt{Locart}}(\mathbf{X}_{n+1}) | I_{train}] \geq 1 - \alpha.$$

Also, notice that the `Locart` cutoff $\hat{t}_{1-\alpha}(\mathbf{x}_{n+1})$ can be written as the inverse of the conditional distribution estimator from Equation (3) at level $1-\alpha$, that is:

$$\hat{t}_{1-\alpha}(\mathbf{X}_{n+1}) = \hat{F}^{-1}(1 - \alpha | \mathbf{X}_{n+1}) \,. \tag{6}$$

Thus, it is intuitive to think that, by the consistency of regression trees as conditional distribution estimators [34], the `Locart` cutoff will be close to the oracle cutoff as sample size increases. Indeed, under additional assumptions about the partition's scheme and conditional cumulative distribution, `Locart` converges to the oracle prediction interval, as defined next.

Definition 2 *(Convergence to the oracle prediction interval).* A prediction interval method $C(\cdot)$ converges to the oracle prediction interval $C^*(\cdot)$ if:

$$\mathbb{P}[Y_{n+1} \in C^*(\mathbf{X}_{n+1}) \Delta C(\mathbf{X}_{n+1})] = o(1), \text{ where } A \Delta B := (A \cap B^c) \cup (B \cap A^c) \,.$$

To show that `Locart` prediction intervals have this property, we assume that asymptotically: (i) the partition elements' are well-populated and (ii) the partition is sufficiently thin:

Assumption 1. Let $A_n(\mathbf{x})$, denote the `Locart` partition element assigned to $\mathbf{x}$, as in Equation (5), when I_{cut} has size n. Then,

$$\lim_{n \to \infty} \inf_{\mathbf{x} \in \mathcal{X}} n \cdot \mathbb{P}[A_n(\mathbf{x})] > 0.$$

Also, we assume that the partition-based $1-\alpha$ quantile of the conditional distribution of Y is close to the $1-\alpha$ quantile of the true conditional distribution of Y given $\mathbf{X}$:

Assumption 2. Let $F_{\mathbf{x}}(y|\mathbf{X}_i) := \mathbb{P}[Y_i \leq y | \mathbf{X}_i \in A_n(\mathbf{x})]$. For every $\mathbf{x}$, $F_{\mathbf{x}}$ is continuous and increasing in a neighborhood around $F^{-1}(1-\alpha|\mathbf{x})$ and $\sup_{\mathbf{x} \in \mathcal{X}} |F_{\mathbf{x}}^{-1}(1-\alpha|\mathbf{X}_i) - F^{-1}(1-\alpha|\mathbf{x})| = o_{\mathbb{P}}(1)$.

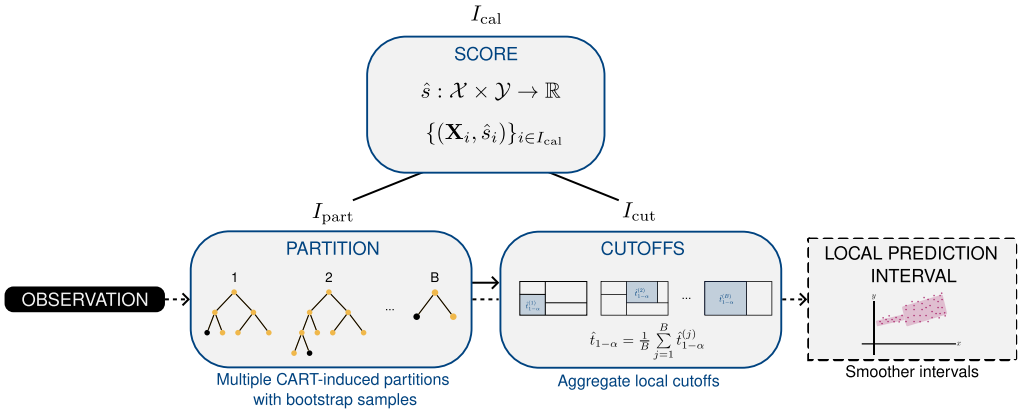

Fig. 3. Schematic diagram of `Loforest`, as a modification of the scoring procedure of `Locart`. The regression residuals are calculated on I_{cal}, and a new dataset of features and residuals is produced. This dataset is split into I_{part}, used to create multiple partitions (each one associated with a full bootstrap resample of I_{part}) of the feature space by fitting a Random Forest of residuals, and I_{cut}, used to populate these partitions and calculate the cutoffs for each tree. The path of a new observation (**- - -**): all the partition elements it belongs to are identified; then, the local cutoffs for each partition are retrieved; next, an average of all the cutoffs is used to create an adaptive interval centered at the prediction of the regression model for this observation. As a result of the averaging step, these intervals are smoother than their `Locart` counterparts.

Under both assumptions, `Locart` converges to the oracle prediction interval in Theorem 4:

Theorem 4 *(Consistency of* `Locart`*). Let* $(\mathbf{X}_i, Y_i)_{i=1}^{n+1}$ *be an i.i.d. sequence. Consider* $\hat{t}_{1-\alpha}(\mathbf{X}_{n+1})$ *and* $t^*_{1-\alpha}(\mathbf{X}_{n+1})$ *the* `Locart` *cutoff estimate and the oracle cutoff for the test point* $\mathbf{X}_{n+1}$*. Then, under Assumptions 1 and 2,* $|t^*(\mathbf{X}_{n+1}) - \hat{t}_{1-\alpha}(\mathbf{X}_{n+1})| = o_{\mathbb{P}}(1)$.

Finally, by using Theorem 4 in the asymptotic regime, we show that `Locart` satisfies asymptotic conditional validity.

Definition 3 *(Asymptotic conditional validity [29]).* A prediction interval method $C(\cdot)$ satisfies asymptotic conditional validity if there exist random sets Λ_n, such that $\mathbb{P}[X_{n+1} \in \Lambda_n | \Lambda_n] = 1 - o_{\mathbb{P}}(1)$ and:

$$\sup_{\mathbf{x}_{n+1} \in \Lambda_n} \left| \mathbb{P}[Y_{n+1} \in C(\mathbf{X}_{n+1}) | \mathbf{X}_{n+1} = \mathbf{x}_{n+1}] - (1-\alpha) \right| = o(1) \,.$$

This is formalized in the next theorem.

Theorem 5 *(Asymptotic conditional validity of* `Locart`*). Let* $(\mathbf{X}_i, Y_i)_{i=1}^{n+1}$ *be an i.i.d. sequence. Under Assumptions 1 and 2,* `Locart` *satisfies asymptotic conditional validity.*

6.3. `Loforest`: local coverage regression forest

As a result, `Locart` may yield sub-optimal results in datasets where regression residuals exhibit this behavior. Moreover, due to the non-smooth partitioning structure of regression trees, `Locart` may also provide poor cutoff estimates for instances at the boundaries of its partition elements. `Loforest` improves `Locart` by leveraging Random Forests. It seeks to increase the expressivity of `Locart` and provides smoother cutoff estimates. Specifically, `Loforest` builds several partitions $\mathcal{A}_k$ by fitting regression trees to bootstrap samples of the original residuals and features in the calibration set. Let B be the number of created regression trees. `Loforest`, as depicted in Fig. 3, consists of five steps:

1. Obtain the regression residuals $\hat{s}_i$ of $\hat{\mu}$ on I_{cal}, that is, compute $\hat{s}_i = \hat{s}(\mathbf{X}_i, Y_i) = |\hat{\mu}(\mathbf{X}_i) - Y_i|$, for every $i \in I_{\text{cal}}$.
2. Create the dataset $\{(\mathbf{X}_i, s_i)\}_{i \in I_{\text{cal}}}$ of pairs of features and regression residuals computed on the calibration dataset. Randomly split this dataset into two disjoint sets I_{part} and I_{cut}.
3. Fit a Random Forest on $\{(X_i, s_i)\}_{i \in I_{\text{part}}}$, which predicts the residuals $\hat{s}$ based on the features $\mathbf{X}$. This algorithm induces several partitions of the feature space $\mathcal{A}_j$, $j = 1, \ldots, B$, using the regression tree algorithm applied to each of the B bootstrap resamples.
4. For a new instance $\mathbf{x}_{n+1}$, estimate the local cutoff $\hat{t}^{(k)}_{1-\alpha}(\mathbf{x}_{n+1})$ in each decision following the same approach as in the `Locart` algorithm, using the dataset $\{(\mathbf{X}_i, s_i)\}_{i \in I_{\text{cut}}}$.
5. Compute the final cutoff $\hat{t}_{1-\alpha}(\mathbf{x}_{n+1})$ by averaging the cutoffs from all decision-tree partitions $\mathcal{A}_k$:

$$\widehat{t}_{1-\alpha}(\mathbf{x}_{n+1}) := \frac{1}{n} \sum_{k=1}^{B} \widehat{t}^{(k)}_{1-\alpha}(\mathbf{x}_{n+1}). \tag{7}$$

6. Using $t_{1-\alpha}(\mathbf{x}_{n+1})$ from Equation (7), define the `Loforest` prediction interval as

$$C_{\texttt{Loforest}}(\mathbf{x}) = (\hat{\mu} - \widehat{t}_{1-\alpha}(\mathbf{x}), \hat{\mu} + \widehat{t}_{1-\alpha}(\mathbf{x})).$$

As in `Locart`, to obtain meaningful local cutoffs in each regression tree, it is essential to control the growth of each regression tree by either pre- or post-pruning. However, post-pruning can become computationally expensive if applied to all trees. Most importantly, it can diminish the variability between the regression trees and undermine the benefit of ensembling different partitions induced by the Random Forest ensemble. Thus, for `Loforest`, we only apply a pre-pruning step, similar to that in Section 6.2.1, fixing the `min_samples_split` hyperparameter at values such as 100 samples. For the remaining Random Forest hyperparameters, we set the number of trees to 100.

Our implementation of `Loforest` uses the *scikit-learn* Random Forest implementation and, as a consequence, is scalable, as shown in the benchmarking studies in Section 7. Similar to `Locart`, empirical results attest that we can skip splitting the calibration set presented in step 2 without harming the method's performance. The results presented in Section 7 show that `Loforest` outperforms several baselines with respect to the conditional and marginal coverage of its intervals in benchmark datasets.

6.4. `A-Locart` *and* `A-Loforest`*: augmenting the feature space*

In Sections 6.2 and 6.3, we discuss how `Locart` and `Loforest` obtain prediction intervals by partitioning the feature space. These strategies aim to recover the optimal partition from Theorem 3, which groups each sample according to the conditional distribution of residuals. In both methods, we use only the original features to create one or several partitions of the feature space.

The idea of the augmented versions of these methods, `A-Locart` and `A-Loforest`, is to improve the partitioning by providing additional statistics (i.e., other functions of the features) as inputs to the trees. One possible way of doing this is by giving first-order proxies to the conditional distribution of residuals. This can be done by using an estimate of $\mathbb{V}[Y|\mathbf{X}]$, for example. Notice that using $\widehat{\mathbb{V}}[Y|\mathbf{X}]$ as a feature is closely connected to the approach adopted in Mondrian Conformal Regressors [46,6]. In fact, by using conditional variance estimates as new covariates, we generalize the Mondrian taxonomy by expanding the binning based on $\mathbb{V}[Y|\mathbf{X}]$ to a general partitioning of both $\mathbb{V}[Y|\mathbf{X}]$ and $\mathcal{X}$. This avoids the need to tune or fix the number of bins, as we use `Locart` and `Loforest` engines to obtain the partitions. Moreover, it also avoids the adversarial setting for the vanilla Mondrian approach, where the variance is constant but the conditional quantiles of residuals are not (as discussed in Section 1 and expanded in Appendix A).

In general, we may also add as features of our model other proxies for the distribution of the residuals (e.g., their conditional mean absolute deviation, as used by Lei et al. [29]) or new feature representations (e.g., random Fourier features [40] and random projections [17]). Section 7 illustrates how the augmentation procedure can improve `Locart` and `Loforest` partitioning.

6.5. *Extensions to other conformal scores and* `W-Loforest`

In the previous sections, we focused on partitioning $\mathcal{X}$ based on the regression residuals. However, as introduced in Section 5.1, this is a particular choice of conformity score. In some cases, it is preferable to build locally valid prediction intervals based on other conformity scores, such as the quantile score and the weighted regression score, mainly when performing inferential tasks other than regression (e.g., quantile regression, conditional distribution estimation).

The `Locart` and `Loforest` frameworks can be straightforwardly generalized to accommodate any of these situations since these methods are completely agnostic to the choice of conformity score. Even the theoretical guarantees for `Locart` are not directly tied to the regression residual and naturally extend to other conformity scores. In Section 7, we showcase the benefits of using the weighted regression score of Lei et al. [29] within `Loforest`, an approach we call `W-Loforest`.

7. Experiments

In this section, we compare the conditional coverage performance of `Locart`, `A-Locart`, `Loforest`, `A-Loforest`, and `W-Loforest` to other state-of-the-art methods for conformal regression and regression intervals on simulated and real-world datasets.

We use Random Forests as the base model for estimating $\hat{\mu}(\cdot)$ across all methods.[2] We set `n_estimators` to 100, i.e., predictions are averaged over 100 trees. We set a common miscoverage level for predictive intervals at $\alpha = 0.1$. We compare our methods against the following baselines:

- Regression split [46,29]: implemented in the *nonconformist* library [33]. The only hyperparameter is the miscoverage level.
- Weighted regression split [29]: implemented in our library, `clover`. We use a Random Forest regressor to estimate the mean absolute deviation of the residuals. The only hyperparameter is the miscoverage level.
- Mondrian split [6]: implemented in our library, `clover`. We use the default difficulty estimator (variance of predictions across trees in the Random Forest). Difficulty estimates were binned in $k = 30$ groups according to the quantiles $(1/k, 2/k, \ldots, \frac{k-1}{k}, 1)$.
- QRF-TC [1,20]: implemented in the *acpi* library [1]. We set the calibration model as a Random Forest Regressor with 100 trees, a minimal node size of 10, a maximum tree depth of 15, and squared error as the split criterion.

[2] Observe that this choice has no relation whatsoever to how `Locart` or `Loforest` is constructed; any other predictive model could have been chosen (e.g., ridge regression, Lasso, neural network).

To evaluate the conditional validity of intervals output by all methods, we employed the SMIS metric [19] for real-world datasets and the conditional coverage absolute deviation for synthetic datasets. We also evaluated marginal validity by computing the average marginal coverage for all datasets. These metrics are briefly described in Section 7.1. We replicate the experiments 100 times and compute the average and standard error at the end to estimate the various measures of interest. For datasets with over 100,000 instances, we conduct 30 experimental replications. To formally compare each method, we perform paired t-tests on each pair of methods and calculate p-values to assess the significance of each comparison. At a significance level of 0.05, we apply the Bonferroni correction for multiple comparisons to decide whether to reject the null hypothesis of no difference between each pair of average metrics. Furthermore, we identify the top methods for each dataset or simulation configuration as the method with the best average metric, as well as all other methods that do not significantly differ from the best one. All computed p-values and test results are shown in Appendix C (Figs. C.8–C.15). To give a comprehensive analysis, we divide our evaluation into three parts: conformal methods only, non-conformal methods only, and a combined assessment of all methods. Additionally, we assess the running time of each method to compare their scalability.

7.1. Metrics for assessing marginal and conditional validity

In this section, we introduce the metrics used in the experiments. Here, we denote by $C(\cdot)$ any prediction interval and by $\{(\mathbf{x}_i, y_i)\}_{i \in I_{\text{test}}}$ a testing set used to compute coverage diagnostics for all compared methods.

7.1.1. Conditional coverage absolute deviation (CCAD)

In the simulation study, we know the ground truth distribution of $Y|\mathbf{X}$. Thus, we can directly measure the conditional coverage. We do this by evaluating the conditional coverage absolute deviation (CCAD): for each $i \in I_{\text{test}}$, we obtain a sample $\mathcal{D} = \{(\mathbf{x}_i, Y_1), \ldots, (\mathbf{x}_i, Y_{B_y})\}$ of size B_y (fixed at $1,000$) from $Y|\mathbf{X} = \mathbf{x}_i$ and compute

$$\delta_\alpha(\mathbf{x}_i) = \frac{1}{B_y} \sum_{y \in \mathcal{D}} \mathbb{1}\{y \in C(\mathbf{x}_i)\}.$$

Notice that $\delta_\alpha(\mathbf{x}_i)$ is an estimate of the conditional coverage $\mathbb{P}[Y \in C(\mathbf{X})|\mathbf{X} = \mathbf{x}_i]$. We derive the conditional coverage absolute deviation (CCAD) by averaging the absolute difference between $\delta_\alpha(\mathbf{x}_i)$ and the nominal level $1 - \alpha$ for all $i \in I_{\text{test}}$,

$$\text{CCAD} = \frac{1}{|I_{\text{test}}|} \sum_{i \in I_{\text{test}}} \left|\delta_\alpha(\mathbf{x}_i) - (1 - \alpha)\right|. \tag{8}$$

7.1.2. Standard mean interval score (SMIS)

In real-world datasets, we cannot compute CCAD because the data-generating process is unknown. Instead, we use the standard mean interval score (SMIS) from Gneiting and Raftery [19]. The interval score associated with y and the miscoverage level α is calculated as

$$\begin{aligned} \text{IS}_\alpha(C(\mathbf{x}), y) = &(\max C(\mathbf{x}) - \min C(\mathbf{x})) \\ &+ \frac{2}{\alpha}(\min C(\mathbf{x}) - y)\mathbb{1}\{y < \min C(\mathbf{x})\} \\ &+ \frac{2}{\alpha}(y - \max C(\mathbf{x}))\mathbb{1}\{y > \max C(\mathbf{x})\}, \end{aligned} \tag{9}$$

where $\min C(\mathbf{x})$ and $\max C(\mathbf{x})$ are the minimum and maximum of the (closed) prediction interval $C(\mathbf{x})$. Intuitively, the interval score penalizes large prediction intervals (first term) that do not contain y (second and third terms) in proportion to how far y is from the limits of the interval and the stricter the miscoverage level. In other words, this score prioritizes the narrowest valid (i.e., containing y) prediction interval. The standard mean interval score is then the average over the whole testing set $\{(\mathbf{x}_i, y_i)\}_{i \in I_{\text{test}}}$:

$$\text{SMIS}_\alpha(C, \mathcal{D}) = \frac{1}{|I_{\text{test}}|} \sum_{i \in I_{\text{test}}} \text{IS}_\alpha(C(\mathbf{x}_i), y_i), \tag{10}$$

which gives us a general balance of interval length and coverage of y for the prediction interval method C. Thus, it can be interpreted as an approximate measure of conditional coverage.

7.1.3. Average marginal coverage

We also estimate the average marginal coverage $\mathbb{P}[Y \in C(\mathbf{X})]$ using the average observed coverage on the test set,

$$\text{AMC} = \frac{1}{|I_{\text{test}}|} \sum_{i \in I_{\text{test}}} \mathbb{1}\{y_i \in C(\mathbf{x}_i)\}. \tag{11}$$

7.2. Simulated data performances

We adopt the simulated settings employed in Izbicki et al. [23], incorporating some modifications to the number of irrelevant variables, and introduce four additional scenarios. Let $\mathbf{X} = (X_1, \ldots, X_d)$, where $X_i \overset{\text{i.i.d.}}{\sim} \text{Unif}(-1.5, 1.5)$, and d and p denotes the total number of features and relevant features respectively. The selected simulated settings are as follows:

- **Homoscedastic**:

$$Y|\mathbf{x} \sim N\left(\frac{2}{p}\sum_{i=1}^{p} x_i, 1\right)$$

- **Heteroscedastic**:

$$Y|\mathbf{x} \sim N\left(\frac{2}{p}\sum_{i=1}^{p} x_i, 0.25 + 2\left|\frac{1}{p}\sum_{i=1}^{p} x_i\right|\right)$$

- **Asymmetric**:

$$Y|\mathbf{x} = \frac{2}{p}\sum_{i=1}^{p} x_i + \varepsilon,$$

where $\varepsilon \sim \text{Gamma}\left(1 + \gamma\left|\frac{\sum_{i=1}^{p} x_i}{p}\right|, 1 + \gamma\left|\frac{\sum_{i=1}^{p} x_i}{p}\right|\right)$, with $\gamma = 0.6$ and 1.5

- **t-residuals**:

$$Y|\mathbf{x} = \frac{2}{p}\sum_{i=1}^{p} x_i + \varepsilon,$$

where $\varepsilon \sim \text{t}_4$.

- **Non-Correlated Heteroscedastic**:

$$Y|\mathbf{x} \sim N\left(1, 0.25 + \left|\frac{2}{p}\sum_{i=1}^{p} x_i\right|\right)$$

- **Normal and exponential residuals:**

$$Y|\mathbf{x} = \frac{1}{p}\sum_{i=1}^{p} x_i^2 + \mathbb{1}(\mathbf{x} \leq 0)\varepsilon_1 + \mathbb{1}(\mathbf{x} > 0)\left(\varepsilon_2 - \frac{1}{1.5}\right),$$

where:

$$\varepsilon_1 \sim N\left(0, 0.5 + \left|\frac{1}{p}\sum_{i=1}^{p} x_i\right|\right) \text{ and } \varepsilon_2 \sim \text{Exponential}(1.5)$$

- **Correlated heteroscedastic**:
We use the same conditional distribution from the **heteroscedastic** scenario, but now consider that $\mathbf{X} \sim N(\mathbf{0}, \Sigma)$ where Σ is the $d \times d$ covariance matrix with $\Sigma_{ij} = 0.7^{|i-j|}$

To enhance the diversity of simulated scenarios, we consider three values for the number of significant features p: 1, 3, and 5 and fix d as 20 for all possible scenarios. For each value of p, we generate a total of 10,000 training and calibration samples and 5,000 testing samples.

In each run, we evaluate the performance of various methods by estimating the mean conditional coverage absolute deviation (CCAD) and the marginal coverage (AMC) associated with the predictive intervals. To determine these metrics, we employ a holdout testing set and measure the running time required for the calibration step for each method.

In general, all methods achieve marginal coverage levels that are very close to the nominal value of 90%, as shown in Table C.5 (see Appendix C for details). However, when it comes to conditional coverage, Table 2 and Fig. 4 indicate variations in performance across different methods:

- In both homoscedastic settings (homoscedastic and t-residuals), except $p = 1$ for t-residuals, we observe in Table 2 that the regression split method demonstrates the best performance, followed by `Locart` and `A-Locart`. This is because the cutoff $\hat{t}_{1-\alpha}(\mathbf{x})$ remains constant for $\mathbf{x}$. `Locart` and `A-Locart` regression trees capture this behavior and reduce to a trivial tree with only one leaf, effectively replicating the regression split.
- In the t-residuals scenario with $p = 1$ `A-Locart` exhibited superior and distinct performance compared to both regression split and `Locart`. This is likely due to the presence of outliers in some of the replications, which `A-Locart` effectively detected using the estimated conditional variance in the partitioning process.

Table 2
Mean conditional coverage absolute deviation (CCAD) values for each method and simulation setting. The average across 100 runs is reported with two times the standard error in parentheses. Values in bold indicate the methods with better performance according to the formal multiple comparison analysis. In general, our methods achieve the best performance in almost all datasets.

Dataset	p	Type of method								
		Conformal					Non-conformal			
		`Locart`	`A-Locart`	RegSplit	W-RegSplit	Mondrian	`Loforest`	`A-Loforest`	`W-Loforest`	QRF-TC
Asym.	1	0.033	0.036	0.06	0.042	0.039	0.028	**0.026**	0.039	0.041
		(0.0004)	(0.0004)	(0.0003)	(0.0002)	(0.0002)	(0.0002)	**(0.0002)**	(0.0002)	(0.0003)
Asym.	3	0.046	0.04	0.051	0.045	0.041	**0.036**	**0.036**	0.041	0.049
		(0.0007)	(0.0003)	(0.0003)	(0.0003)	(0.0003)	**(0.0003)**	**(0.0002)**	(0.0003)	(0.0005)
Asym.	5	0.045	0.04	0.045	0.046	0.041	0.039	**0.037**	0.043	0.048
		(0.0003)	(0.0003)	(0.0003)	(0.0003)	(0.0002)	(0.0002)	**(0.0002)**	(0.0003)	(0.0006)
Asym. 2	1	0.04	0.043	0.086	0.045	0.041	**0.027**	0.028	0.042	0.049
		(0.0005)	(0.0005)	(0.0003)	(0.0003)	(0.0002)	**(0.0002)**	(0.0002)	(0.0002)	(0.0004)
Asym. 2	3	0.064	0.055	0.084	0.055	0.053	**0.047**	**0.049**	0.051	0.065
		(0.0007)	(0.0003)	(0.0003)	(0.0003)	(0.0003)	**(0.0003)**	**(0.0002)**	(0.0003)	(0.0005)
Asym. 2	5	0.072	0.059	0.073	0.063	0.058	0.06	**0.054**	0.057	0.068
		(0.0004)	(0.0003)	(0.0003)	(0.0003)	(0.0002)	(0.0003)	**(0.0002)**	(0.0003)	(0.0005)
Heterosc.	1	0.029	0.032	0.066	0.041	0.031	**0.017**	0.019	0.04	0.044
		(0.0005)	(0.0004)	(0.0002)	(0.0001)	(0.0002)	**(0.0002)**	(0.0002)	(0.0001)	(0.0004)
Heterosc.	3	0.054	0.046	0.069	0.051	0.043	**0.041**	**0.041**	0.049	0.062
		(0.0006)	(0.0003)	(0.0002)	(0.0002)	(0.0002)	**(0.0003)**	**(0.0001)**	(0.0002)	(0.0004)
Heterosc.	5	0.064	0.052	0.066	0.057	0.051	0.056	**0.049**	0.054	0.064
		(0.0004)	(0.0002)	(0.0002)	(0.0002)	(0.0002)	(0.0003)	**(0.0002)**	(0.0002)	(0.0004)
Homosc.	1	**0.01**	**0.01**	**0.01**	0.034	0.017	0.012	0.012	0.033	0.028
		(0.0001)	**(0.0001)**	**(0.0001)**	(0.0002)	(0.0003)	(0.0001)	(0.0001)	(0.0001)	(0.0005)
Homosc.	3	**0.012**	**0.012**	**0.012**	0.035	0.018	0.013	0.013	0.033	0.03
		(0.0001)	**(0.0001)**	**(0.0001)**	(0.0001)	(0.0003)	(0.0001)	(0.0001)	(0.0001)	(0.0004)
Homosc.	5	**0.012**	**0.012**	**0.012**	0.035	0.018	0.014	0.014	0.033	0.03
		(0.0001)	**(0.0001)**	**(0.0001)**	(0.0002)	(0.0002)	(0.0001)	(0.0001)	(0.0001)	(0.0004)
Non-corr. heterosc.	1	0.029	0.031	0.066	0.041	0.05	**0.016**	0.017	0.04	0.044
		(0.0006)	(0.0006)	(0.0002)	(0.0002)	(0.0003)	**(0.0002)**	(0.0002)	(0.0001)	(0.0004)
Non-corr. heterosc.	3	0.054	0.063	0.068	0.05	0.065	**0.041**	0.054	0.048	0.061
		(0.0006)	(0.0006)	(0.0002)	(0.0002)	(0.0002)	**(0.0003)**	(0.0004)	(0.0002)	(0.0005)
Non-corr. heterosc.	5	0.064	0.064	0.065	0.057	0.063	0.056	0.059	**0.054**	0.064
		(0.0004)	(0.0002)	(0.0002)	(0.0002)	(0.0002)	(0.0003)	(0.0002)	**(0.0002)**	(0.0004)
t-residuals	1	0.012	**0.011**	0.012	0.035	0.017	0.014	0.013	0.033	0.03
		(0.0002)	**(0.0001)**	(0.0001)	(0.0002)	(0.0003)	(0.0002)	(0.0001)	(0.0001)	(0.0006)
t-residuals	3	**0.012**	**0.012**	**0.012**	0.035	0.018	0.014	0.013	0.033	0.03
		(0.0001)	**(0.0001)**	**(0.0001)**	(0.0002)	(0.0003)	(0.0001)	(0.0001)	(0.0001)	(0.0005)
t-residuals	5	**0.011**	**0.011**	**0.011**	0.035	0.017	0.013	0.013	0.032	0.03
		(0.0001)	**(0.0001)**	**(0.0001)**	(0.0002)	(0.0003)	(0.0001)	(0.0001)	(0.0001)	(0.0005)
N-Exp. residuals	1	0.031	0.032	0.098	0.041	0.036	0.024	**0.023**	0.039	0.039
		(0.0004)	(0.0006)	(0.0002)	(0.0002)	(0.0004)	(0.0002)	**(0.0002)**	(0.0002)	(0.0004)
N-Exp. residuals	3	0.044	0.043	0.074	0.045	0.045	**0.033**	0.035	0.042	0.045
		(0.0006)	(0.0004)	(0.0002)	(0.0002)	(0.0002)	**(0.0002)**	(0.0002)	(0.0002)	(0.0002)
N-Exp. residuals	5	0.048	0.047	0.062	0.044	0.048	**0.038**	**0.038**	0.041	0.045
		(0.0004)	(0.0004)	(0.0002)	(0.0002)	(0.0002)	**(0.0002)**	**(0.0002)**	(0.0002)	(0.0002)
Corr. heterosc.	1	0.033	0.036	0.077	0.043	0.035	**0.019**	0.022	0.042	0.047
		(0.0008)	(0.0006)	(0.0002)	(0.0002)	(0.0002)	**(0.0002)**	(0.0002)	(0.0002)	(0.0004)
Corr. heterosc.	3	0.05	0.044	0.076	0.048	0.041	**0.034**	0.036	0.046	0.049
		(0.0004)	(0.0002)	(0.0002)	(0.0002)	(0.0002)	**(0.0002)**	(0.0002)	(0.0002)	(0.0004)
Corr. heterosc.	5	0.055	0.047	0.075	0.049	0.044	**0.039**	0.040	0.047	0.05
		(0.0004)	(0.0002)	(0.0002)	(0.0002)	(0.0002)	**(0.0002)**	(0.0002)	(0.0002)	(0.0002)

- In all other settings, particularly the asymmetric, heteroscedastic and correlated heteroscedastic settings, our non-conformal methods (`Loforest`, `A-Loforest`) exhibit superior performance. Additionally, among the conformal methods, the first panel in Fig. 4 and Table 2 demonstrate that our two conformal procedures also excel in these settings.
- QRF-TC does not perform as well as either the other non-conformal or conformal methods, as evident from Table 2 and the second and third panels in Fig. 4.
- Considering a comprehensive analysis of all methods and settings, the third panel of Fig. 4 indicates that our methods outperform all other regression interval methods.

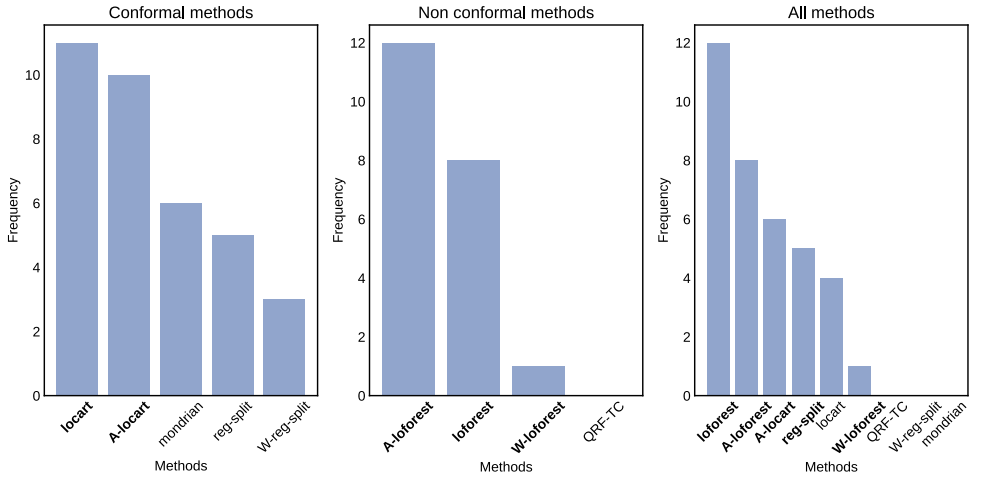

Fig. 4. Frequency of times each method performs better in simulated settings according to the formal p-value analysis of the average CCAD in three different comparisons: between only conformal methods (first panel), between only non-conformal methods (second panel), and between all methods altogether (third panel). The plots indicate that our methods have competitive performance and are flexible to different data scenarios.

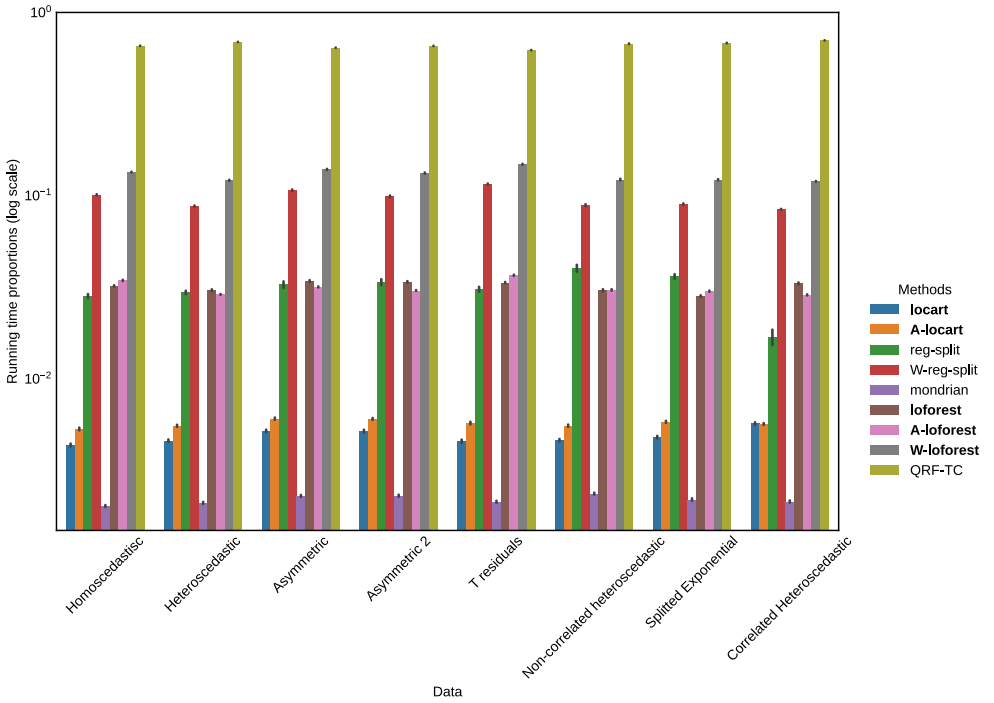

Fig. 5. Mean proportion of running time spent in each method for each simulation setting. Our methods are highlighted in bold. Our methods are scalable compared to competing approaches, and considerably cheaper than LCP-based methods.

In terms of computational costs, Fig. 5 illustrates that `Locart`, `A-Locart`, and Mondrian are the most affordable methods, accounting for less than 1% of the total running time. Except for QRF-TC, the remaining methods exhibit moderate to low computational costs, utilizing between 1% and 10% of the total simulation running time. Conversely, QRF-TC stands out as the most resource-intensive method, requiring considerably more than 10% of the running time to produce predictive intervals. To provide context, when executed on a computer with an Intel Core i7-8700 CPU with 3.20 GHz, 6 cores, 12 threads, and 54 GB of RAM, QRF-TC takes an average of approximately 11 minutes to process a dataset comprising 10,000 training and calibration samples with 20 dimensions. In contrast, `W-Loforest`, the second most computationally demanding method, halts in approximately 1 minute and 15 seconds.

Therefore, based on these comparisons, we conclude that our framework, despite including non-conformal methods, consistently displays excellent conditional validity performance and superior scalability. These results also highlight the flexibility of our methods in adapting to different data-generating processes, such as homoscedastic, heteroscedastic, and correlated designs.

Table 3
Real-world data information. For each dataset, n is the sample size, and p is the number of features.

Dataset	n	p	Source	Dataset	n	p	Source
Concrete	1030	8	Concrete (UCI)	Meps19	15781	141	Meps19 (clover repository)
Airfoil	1503	5	Airfoil (UCI)	Superconductivity	21263	81	Superconductivity (UCI)
Wine white	1599	11	Wine white (UCI)	News	39644	59	News (UCI)
Star	2161	48	Star (ACPI repository)	Protein	45730	8	Protein (UCI)
Wine red	4898	11	Wine red (UCI)	Wave energy converter	54000	49	WEC (UCI)
Cycle	9568	4	Cycle (UCI)	Amazon	130000[a]	500	Amazon (Kaggle)
Electric	10000	12	Electric (UCI)	SGMM	241600	14	SGEMM (UCI)
Bike	10886	12	Bike (Kaggle)	Year Prediction	463715	90	Year Prediction (UCI)

[a] In each iteration, we subsampled from a large text dataset with 393,930 instances.

7.3. Real data performances

Next, we comprehensively compared various regression predictive interval methods using real-world datasets listed in Table 3. Each dataset was split into three distinct sets: training (40%), calibration (40%), and testing (20%). We evaluated the performance of each method by computing the SMIS score, the marginal coverage (AMC) on the testing set, and the running time during the calibration step.

Regarding the marginal coverage of each method, Figs. C.12 and C.13 in Appendix C reveals two main observations:

- Except for the Mondrian method, all conformal methods demonstrate marginal coverage that is close to the nominal level across all datasets. However, the Mondrian method exhibits over-coverage in small datasets such as Wine white, Wine red, Concrete, and Airfoil and in large datasets such as WEC. This can be attributed to the large number of fixed bins ($k = 30$) relative to the small sample sizes, resulting in bins with only a few instances grouped. For large datasets, it can be due to the small number of fixed bins relative to the large sample size, resulting in bins mixing several outliers with commonly observed samples. Consequently, larger cutoffs are generated for each partition element in both cases.
- Among the non-conformal methods, some show slight over-coverage in specific datasets such as Superconductivity, News, Meps19, Bike, and WEC. Notably, QRF-TC demonstrates occasional instances of under-coverage, as observed in the airfoil and star datasets. Thus, even though some of our methods are non-conformal, they generally exhibit marginal coverage at least greater than the nominal level.

Based on Table 4 and Fig. 6, we also conclude the following:

- Table 4 demonstrates that our conformal and non-conformal methods, particularly `A-Loforest` and `A-Locart`, exhibit strong performance across most datasets, except for the Airfoil and Concrete datasets. Particularly, `A-Loforest` shows remarkable performance in larger datasets such as Protein, Superconductivity, Bike-sharing, News, WEC, SGEMM, Amazon, and Year Prediction data. This indicates that our adaptive partitioning of the feature space effectively works for large datasets and achieves favorable conditional coverage in this setting.
- Analyzing the first panel of Fig. 6, we observe that our conformal proposal ranks as a top performer among all conformal approaches in twelve datasets. Moreover, by referring to Table 4 and the multiples comparisons presented in Figs. C.10 and C.11, we find that `A-Locart` generally performs equally or better than Mondrian, suggesting that our approach enhances the conditional variance binning of Mondrian by incorporating other features and increasing their flexibility.
- Further examination of Table 4 and Fig. C.10 reveals that QRF-TC achieves superior performance in small datasets, notably the Airfoil and Concrete datasets, but lags behind other non-conformal and conformal methods in all other scenarios.
- Notably, as revealed by Table 4 and Fig. C.11, `A-Loforest` significantly outperforms all competing non-conformal and conformal approaches in large-scale datasets like WEC, Amazon, SGEMM, and Year Prediction. This underscores the consistency and scalability of the `A-Loforest` partitioning scheme for handling large and possibly high-dimensional data.
- Overall, Fig. 6 demonstrates that our class of methods achieves strong conditional coverage performance across a great variety of real-world datasets.

Regarding the computational cost analysis of each method, Fig. 7 exhibits a behavior similar to what was observed in the simulated experiments, but with some variations. In addition to `Locart`, `A-Locart`, and Mondrian, regression split is also identified as one of the most computationally efficient methods, with a running time spanning from 0.01% to close to 1% of the total analysis running time. Although the remaining methods, except for QRF-TC, demonstrate moderate to low computational costs (time consumption between 1% and 10%) for the majority of datasets, W-reg-split and `W-Loforest` exceed 10% consumption in the superconductivity, news, WEC, Amazon and Year Prediction datasets, even surpassing the runtime proportion of QRF-TC in the Amazon and Year Prediction datasets. This may be attributed to the high computational cost of training the MAD model in these high dimensional contexts, which is avoided by both `Loforest` and `A-Loforest`. Additionally, for all the large-scale datasets, both top performers, `A-Locart` and `A-Loforest`, required only 1% to 10% of the total running time to fit, highlighting the excellent scalability of our approaches in high-dimensional and large-sample contexts.

Table 4
SMIS values for each method and real-world dataset. The average across 100 runs is reported with two times the standard deviation in parentheses. Values in bold indicate the method with better performance according to the formal multiple comparison analysis. Both `A-Locart` and `A-Loforest` have good performance.

Dataset	Type of method								
	Conformal					Non-conformal			
	`Locart`	`A-Locart`	RegSplit	W-RegSplit	Mondrian	`Loforest`	`A-Loforest`	`W-Loforest`	QRF-TC
Concrete	27.301	26.196	27.729	28.290	28.490	27.729	27.729	28.290	**25.497**
	(0.526)	(0.497)	(0.542)	(0.442)	(0.399)	(0.542)	(0.542)	(0.442)	**(0.487)**
Airfoil	10.395	10.232	10.915	10.140	10.587	10.341	10.201	10.085	**9.533**
	(0.170)	(0.168)	(0.194)	(0.176)	(0.157)	(0.173)	(0.167)	(0.176)	**(0.153)**
Wine white	2.950	2.884	2.977	3.012	2.864	2.905	**2.853**	2.996	**2.843**
	(0.0217)	(0.023)	(0.022)	(0.021)	(0.021)	(0.022)	**(0.021)**	(0.021)	**(0.021)**
Star	973.786	973.614	972.470	1042.652	1007.822	**970.550**	**970.513**	1039.369	997.357
	(6.959)	(7.094)	(7.053)	(7.450)	(7.278)	**(6.959)**	**(6.981)**	(7.387)	(8.134)
Wine red	2.780	**2.750**	2.798	2.874	2.843	2.758	**2.725**	2.869	**2.725**
	(0.033)	**(0.0307)**	(0.033)	(0.031)	(0.031)	(0.032)	**(0.031)**	(0.030)	**(0.034)**
Cycle	15.850	15.211	15.930	15.480	15.191	15.623	**15.087**	15.386	15.420
	(0.118)	(0.120)	(0.121)	(0.110)	(0.115)	(0.116)	**(0.118)**	(0.109)	(0.109)
Electric $\times (10^{-2})$	5.360	5.320	5.580	5.050	5.340	5.220	5.190	**4.990**	5.170
	(0.03)	(0.03)	(0.03)	(0.02)	(0.03)	(0.02)	(0.02)	**(0.02)**	(0.02)
Bike	179.017	162.536	221.847	165.679	163.245	178.467	**160.772**	164.572	171.750
	(1.549)	(1.112)	(1.832)	(1.098)	(1.126)	(1.397)	**(1.105)**	(1.074)	(1.242)
Meps19	73.316	67.089	105.539	**66.575**	**66.697**	71.681	**66.585**	**66.574**	69.372
	(0.977)	(1.068)	(1.191)	**(1.033)**	**(1.105)**	(0.957)	**(1.068)**	**(1.005)**	(0.966)
Superconductivity	41.248	37.590	53.095	41.005	39.165	39.953	**36.571**	40.147	37.352
	(0.280)	(0.223)	(0.283)	(0.230)	(0.223)	(0.237)	**(0.212)**	(0.225)	(0.214)
News $\times (10^4)$	3.304	2.988	3.571	2.953	2.961	3.163	**2.911**	2.966	3.124
	(0.043)	(0.040)	(0.043)	(0.040)	(0.042)	(0.041)	**(0.040)**	(0.040)	(0.040)
Protein	16.607	14.039	17.583	14.693	14.045	16.303	**13.812**	14.612	15.388
	(0.047)	(0.048)	(0.036)	(0.044)	(0.043)	(0.037)	**(0.046)**	(0.043)	(0.038)
WEC $\times (10^3)$	94.8	64.392	151.207	77.456	68.834	91.172	**61.799**	71.092	70.219
	(0.544)	(0.486)	(0.73)	(0.514)	(0.376)	(0.512)	**(0.34)**	(0.56)	(0.386)
SGEMM	19.694	16.181	30.131	16.85	17.089	19.773	**15.957**	16.742	17.932
	(0.061)	(0.057)	(0.094)	(0.049)	(0.07)	(0.058)	**(0.062)**	(0.047)	(0.052)
Amazon	4.815	4.520	5.147	4.628	4.519	4.809	**4.502**	4.599	4.888
	(0.008)	(0.01)	(0.007)	(0.01)	(0.01)	(0.008)	**(0.01)**	(0.01)	(0.014)
Year Prediction	39.454	37.031	43.644	37.641	37.129	39.202	**36.902**	37.001	38.603
	(0.035)	(0.032)	(0.039)	(0.04)	(0.038)	(0.034)	**(0.038)**	(0.036)	(0.036)

Finally, we reinforce that QRF-TC is considerably more computationally costly than all the other methods in real data settings. For comparison, QRF-TC takes on average 14.4 seconds and 5 minutes to run concrete and protein datasets, respectively, while W-Loforest (the second most computationally heavy algorithm) takes 0.4 and 23 seconds to run each.

8. Final remarks

We propose two methods to calibrate prediction intervals: `Locart` and `Loforest`. These methods produce adaptive prediction intervals based on a sensible partition of the feature space created by training regression trees on conformity scores. A central aspect of `Locart` and `Loforest` is their applicability to high-dimensional settings, as our partition is not based on distance metrics on the feature space. Instead, this partition is obtained by grouping instances according to the conditional distribution of the conformal scores. Furthermore, these methods can be extended to calibrate prediction intervals based on any conformal score.

From a theoretical perspective, we show that `Locart` produces prediction intervals with marginal and local coverage guarantees. In addition, we prove that, under additional assumptions, `Locart` has asymptotic conditional coverage. Since `Loforest` is essentially an ensemble of `Locart` instances, we expect that `Loforest` inherits the properties of `Locart` while stabilizing the cutoff estimates that define the prediction intervals.

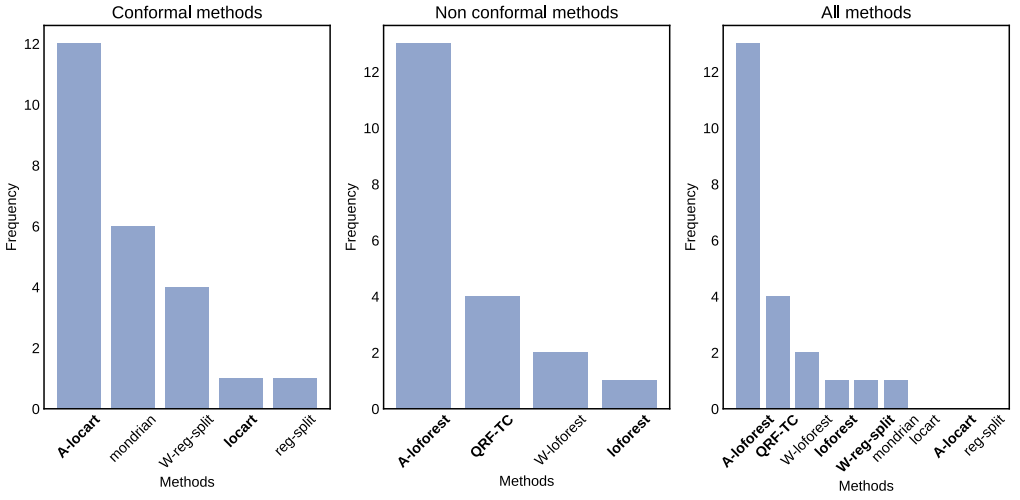

Fig. 6. Frequency of times each method performs better in real datasets according to the formal p-value analysis of the average SMIS in three different comparisons: between only conformal methods (first panel), between only non-conformal methods (second panel), and between all methods altogether (third panel). All panels suggest that both `A-Locart` and `A-Loforest` stand out as the most frequent methods. The third panel also suggests that, among all methods, `A-Loforest` has superior performance across several real datasets.

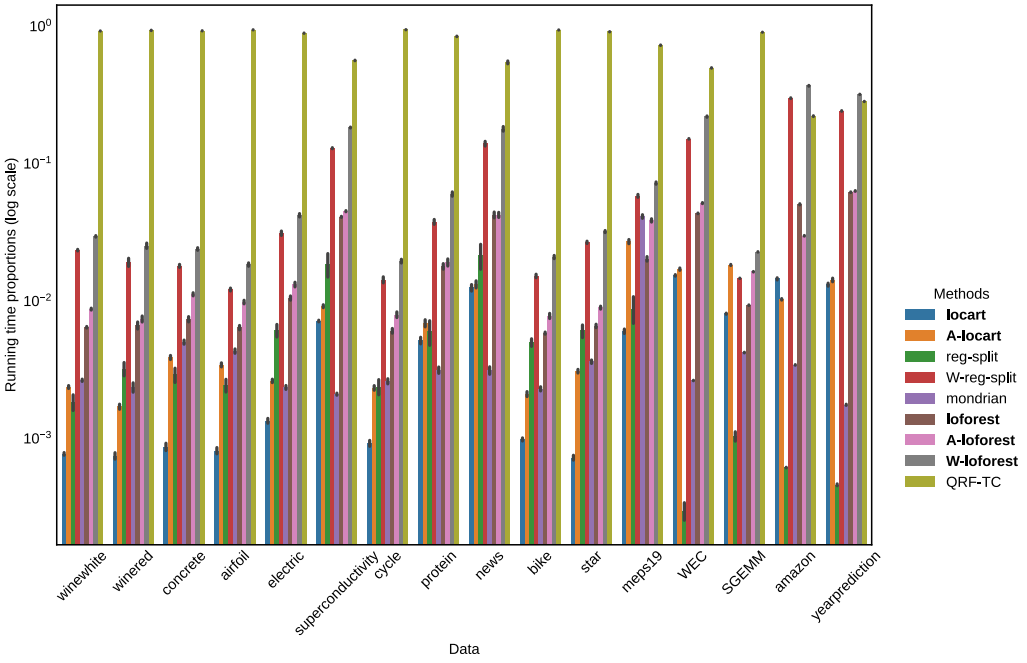

Fig. 7. Mean proportion of running time spent in each method for each real dataset. Our methods are highlighted in bold. Even for medium to big data, our methods are still scalable compared to the remaining approaches.

We validated our methods on simulated and real datasets, where they exhibited superior performance compared to baselines. In the simulated settings, our methods and their extensions, such as `A-Locart` and `A-Loforest`, present good conditional coverage and control the marginal coverage at the nominal level. The same holds when applying these methods to real data for both small and large samples. In all scenarios, our methods outperform established state-of-the-art algorithms, such as LCP and QRF-TC, including in running time comparisons.

We provide a Python package, named `clover`, that implements our methods using the *scikit-learn* interface and replicates all experiments done in this work. For future work, we plan to extend our methods to incorporate a variety of conformal scores, aiming to enhance their local coverage properties. Additionally, we will explore the use of boosting techniques as an alternative to forests for creating the partitions, which may offer improved accuracy and flexibility. Finally, we intend to investigate the application of our approach to time series data and other non-i.i.d. data structures, broadening the scope and applicability of our methods to a wider range of real-world problems.

CRediT authorship contribution statement

Luben M.C. Cabezas: Writing – review & editing, Writing – original draft, Supervision, Software, Project administration, Methodology, Investigation, Data curation, Conceptualization. **Mateus P. Otto:** Writing – review & editing, Writing – original draft, Supervision, Software, Methodology, Investigation, Data curation. **Rafael Izbicki:** Writing – review & editing, Writing – original draft, Supervision, Methodology, Investigation, Conceptualization. **Rafael B. Stern:** Writing – review & editing, Supervision, Methodology, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

I have shared the link to my github repository in the article.

Acknowledgements

L. M. C. C. is grateful for the fellowship provided by São Paulo Research Foundation (FAPESP), grant 2022/08579-7. R. I. is grateful for the financial support of Fundação de Amparo à Pesquisa do Estado de São Paulo (grants 2019/11321-9 and 2023/07068-1) and Conselho Nacional de Desenvolvimento Científico e Tecnológico (grant 305065/2023-8). M. P. O. was supported through grant 2021/02178-8, São Paulo Research Foundation (FAPESP). R. B. S. produced this work as part of the activities of Fundação de Amparo à Pesquisa do Estado de São Paulo Research, Innovation and Dissemination Center for Neuromathematics (grant 2013/07699-0). The authors are also grateful for the suggestions given by Rodrigo F. L. Lassance.

Appendix A. Failures of baselines

We recall from the discussion of Section 6.1 that the oracle prediction interval depends on the conditional quantiles of the regression residual (calculated with the true regression function). Unless the data is homoscedastic, the residual is non-constant across the feature space, and so is its quantile as a function of $\mathbf{x} \in \mathcal{X}$. Therefore, any method that seeks to approximate the oracle prediction interval – and inherit its optimal properties – should be based on conformity scores that adapt to the data's local features. In the following, we present some data-generating processes where purportedly adaptive methodologies fail.

A.1. Locally weighted regression split

Let $\hat{\mu}$ be an estimator of the regression function. Lei et al. [29] uses the conditional mean absolute deviation (MAD) of $[Y - \hat{\mu}(X)]|X = x$ to normalize prediction intervals. To see why this might fail, consider

$$Y|X = x \sim \begin{cases} \text{Laplace}(0, M), & x \in [0, 1), \\ \text{Beta}(\alpha, \beta), & x \in [1, 2], \end{cases}$$

where $\alpha = 2, \beta = 2$, and $M = \frac{2\alpha^\alpha \beta^\beta}{B(\alpha,\beta)(\alpha+\beta)^{\alpha+\beta+1}}$, with $B(\alpha, \beta)$ the Beta function. The conditional mean absolute deviation of $Y|X = x$ is constant and equal to M. The conditional quantiles of $|Y - \mu(X)|$, however, depend on whether $x \in [0, 1)$ or $x \in [1, 2]$. Therefore, normalizing with the MAD does not render the locally weighted regression residual adaptive in this situation.

Difficulty as conditional variance of Y. Boström et al. [8] uses the empirical variance of predictions made by the trees of a Random Forest to normalize prediction regions. Even in the hypothetical scenario where we replace the empirical variance with the true conditional variance of the labels, $\mathbb{V}(Y|X = x)$, the variance-normalized prediction intervals cannot yield asymptotically valid prediction intervals. To see this, consider the mixture model

$$Y|X = x \sim 0.5\, N(-x, s^2 - x^2) + 0.5\, N(x, s^2 - x^2),$$

where we take $x \in [0, 1]$ for simplicity and s sufficiently large so as to $Y|X$ be unimodal. It is easy to show that $\mathbb{V}(Y|X = x) = s^2$, thus independent of x. On the other hand, the conditional quantiles of $|Y - \mu(X)|$ or of $|Y||X = x$ (since $\mu(x) = 0$) depend on x: as x increases, the density is concentrated at larger values of x, forcing the quantiles to drift away from zero. It follows that even if the samples have the same conditional variance – and thus, would be binned together in the approach of Boström et al. [8] – they do not have the same distribution of regression residuals. Therefore, normalization with conditional variance cannot be used to construct prediction intervals that adapt to the heteroscedasticity of the data

Appendix B. Proofs

The proofs are organized into four subsections: B.1 proves that `Locart` has local and marginal validity (Theorem 2 and Corollaries 1 and 2), B.2 proves Theorem 3, B.3 proves the consistency of `Locart` (Theorem 4), and B.4 proves `Locart` asymptotic conditional validity (Theorem 5).

B.1. Theorem 2

Proof of Theorem 2. Let $A_j \in \mathcal{A}$ be an arbitrary partition element. Consider $I_j = \{i \in I_{\text{cal}} : \mathbf{X}_i \in A_j\}$. Since the full data $\{(\mathbf{X}_i, Y_i)\}_{i \in I_{\text{cal}} \sqcup \{n+1\}}$ is exchangeable, the scores $\hat{s}_i$ for $i \in I_j \sqcup \{n+1\}$ are exchangeable conditional on I_j and $\mathbf{X}_{n+1} \in A_j$. Therefore, by the definition of s_j^*,

$$\begin{aligned}\mathbb{P}[Y_{n+1} \in C_{\text{local}}(\mathbf{X}_{n+1}) | \mathbf{X}_{n+1} \in A_j, I_j] &= \mathbb{P}[\hat{s}(\mathbf{X}_{n+1}, Y_{n+1}) \leq s_j^* | \mathbf{X}_{n+1} \in A_j, I_j] \\ &\geq 1 - \alpha.\end{aligned}$$

Because this holds for all I_j, the bound is true after marginalizing over I_j. Thus,

$$\mathbb{P}[Y_{n+1} \in C_{\text{local}}(\mathbf{X}_{n+1}) | \mathbf{X}_{n+1} \in A_j] \geq 1 - \alpha. \tag{B.1}$$

As this holds for an arbitrary element A_j, (B.1) is valid for all $j = 1, \ldots, K$. Corollary 1 is proved by applying Theorem 2 in the `Locart` partition and Corollary 2 follows from Lei and Wasserman [30]. □

B.2. Theorem 3

Proof of Theorem 3. To prove item 1 notice that if $\mathbf{x}_1, \mathbf{x}_2 \in A$, then $\hat{s}(\mathbf{X}, Y) | \mathbf{X} = \mathbf{x}_1 \sim \hat{s}(\mathbf{X}, Y) | \mathbf{X} = \mathbf{x}_2$, which by Definition 1 implies directly that $t^*_{1-\alpha}(\mathbf{x}_1) = t^*_{1-\alpha}(\mathbf{x}_2)$ for any $\alpha \in (0, 1)$. To show item 2, assume that $\mathbf{x}_1, \mathbf{x}_2 \in J$, and then $t^*_{1-\alpha}(\mathbf{x}_1) = t^*_{1-\alpha}(\mathbf{x}_2)$ for every $\alpha \in (0, 1)$. Conclude that $F^{-1}_{\hat{s}(\mathbf{X},Y)|\mathbf{X}=\mathbf{x}_1}(1-\alpha|\mathbf{x}_1) = F^{-1}_{\hat{s}(\mathbf{X},Y)|\mathbf{X}=\mathbf{x}_2}(1-\alpha|\mathbf{x}_2)$ for every $\alpha \in (0, 1)$. Thus, from the monotonicity and continuity of $F_{\hat{s}(\mathbf{X},Y)|\mathbf{X}=\mathbf{x}}$, we obtain that $F_{\hat{s}(\mathbf{X},Y)|\mathbf{X}=\mathbf{x}_1}(r|\mathbf{x}_1) = F_{\hat{s}(\mathbf{X},Y)|\mathbf{X}=\mathbf{x}_2}(r|\mathbf{x}_2)$ for all $r \in \mathbb{R}$. It follows that $\hat{s}(\mathbf{X}, Y) | \mathbf{X} = \mathbf{x}_1 \sim \hat{s}(\mathbf{X}, Y) | \mathbf{X} = \mathbf{x}_2$, and therefore $\mathbf{x}_1, \mathbf{x}_2 \in A$. □

B.3. Theorem 4

Proof of Theorem 4. Under Assumptions 1 and 2, the theorem follows from Lemma 34 in Izbicki et al. [23]. □

B.4. Theorem 5

To prove Theorem 5 we use the following Lemmas 1 and 2 proved below:

Lemma 1. *If a prediction interval method $C(\cdot)$ converges to the oracle prediction interval $C^*(\cdot)$, then it satisfies asymptotic conditional coverage.*

Proof of Lemma 1. Since $\mathbb{P}[Y_{n+1} \in C^*(\mathbf{X}_{n+1}) \Delta C(\mathbf{X}_{n+1})] = o(1)$, it follows from Markov's inequality and the dominated convergence theorem that

$$\mathbb{P}[Y_{n+1} \in C^*(\mathbf{X}_{n+1}) \Delta C(\mathbf{X}_{n+1}) | \mathbf{X}_{n+1}] = o_{\mathbb{P}}(1)\,.$$

Therefore, there exists $\phi_n = o(1)$ such that, for

$$\Lambda^c_{n+1} = \{\mathbf{x}_{n+1} \in \mathcal{X} : \mathbb{P}[Y_{n+1} \in C^*(\mathbf{X}_{n+1}) \Delta C(\mathbf{X}_{n+1}) | \mathbf{X}_{n+1} = \mathbf{x}_{n+1}] > \phi_n\}\,,$$

one obtains $\mathbb{P}[\mathbf{X}_{n+1} \in \Lambda^c_{n+1}] = o(1)$. Conclude that $\mathbb{P}[\mathbf{X}_{n+1} \in \Lambda_{n+1}] = 1 - o(1)$ and that:

$$\begin{aligned}&\sup_{\mathbf{x}_{n+1} \in \Lambda_n} \left| \mathbb{P}[Y_{n+1} \in C(\mathbf{X}_{n+1}) | \mathbf{X}_{n+1} = \mathbf{x}_{n+1}] - (1 - \alpha) \right| \\ &\leq \sup_{\mathbf{x}_{n+1} \in \Lambda_n} \left| \mathbb{P}[Y_{n+1} \in C^*(\mathbf{X}_{n+1}) \Delta C(\mathbf{X}_{n+1}) | \mathbf{X}_{n+1} = \mathbf{x}_{n+1}] \right| \leq \phi_n = o(1).\end{aligned}$$ □

Lemma 2. *Let $C^* = (\hat{\mu}(\mathbf{X}_{n+1}) - t^*_{1-\alpha}(\mathbf{X}_{n+1}), \hat{\mu}(\mathbf{X}_{n+1}) + t^*_{1-\alpha}(\mathbf{X}_{n+1}))$ and $C_{\texttt{Locart}} = (\hat{\mu}(\mathbf{X}_{n+1}) - \hat{t}_{1-\alpha}(\mathbf{X}_{n+1}), \hat{\mu}(\mathbf{X}_{n+1}) + \hat{t}_{1-\alpha}(\mathbf{X}_{n+1}))$, where $\hat{t}(.)$ is the `Locart` estimated cutoff for a fixed induced partition. Under Assumption 1:*

$$\mathbb{P}[Y_{n+1} \in C^* \Delta C_{\texttt{Locart}} | \mathbf{X}_{n+1}] = o_{\mathbb{P}}(1)\,.$$

Proof of Lemma 2. Under Assumption 1, $|t^*_{1-\alpha}(\mathbf{X}_{n+1}) - \hat{t}_{1-\alpha}(\mathbf{X}_{n+1})| = o_{\mathbb{P}}(1)$ by Theorem 4. Thus, there exists $\lambda_n = o(1)$ such that $\mathbb{P}[|t^*_{1-\alpha}(\mathbf{X}_{n+1}) - \hat{t}_{1-\alpha}(\mathbf{X}_{n+1})| > \lambda_n] = o(1)$. Note that, $\{Y_{n+1} \in C^* \Delta C_{\texttt{Locart}}\} \subseteq \{|t^*_{1-\alpha}(\mathbf{X}_{n+1}) - \hat{t}_{1-\alpha}(\mathbf{X}_{n+1})| > \lambda_n\}$. Thus:

$$\begin{aligned}\mathbb{P}[Y_{n+1} \in C^* \Delta C_{\texttt{Locart}}] &\leq \mathbb{P}[|t^*_{1-\alpha}(\mathbf{X}_{n+1}) - \hat{t}_{1-\alpha}(\mathbf{X}_{n+1})| > \lambda_n] \\ &= o(1)\,.\end{aligned}$$

Since $\mathbb{P}[Y_{n+1} \in C^* \Delta C_{\texttt{Locart}}] = o(1)$ if follows from Markov's inequality that $\mathbb{P}[Y_{n+1} \in C^* \Delta C_{\texttt{Locart}} | \mathbf{X}_{n+1}] = o_{\mathbb{P}}(1)$. □

Now, we prove Theorem 5.

Proof of Theorem 5. Follows directly from Lemmas 1 and 2. □

Appendix C. Experiments additional figures and tables

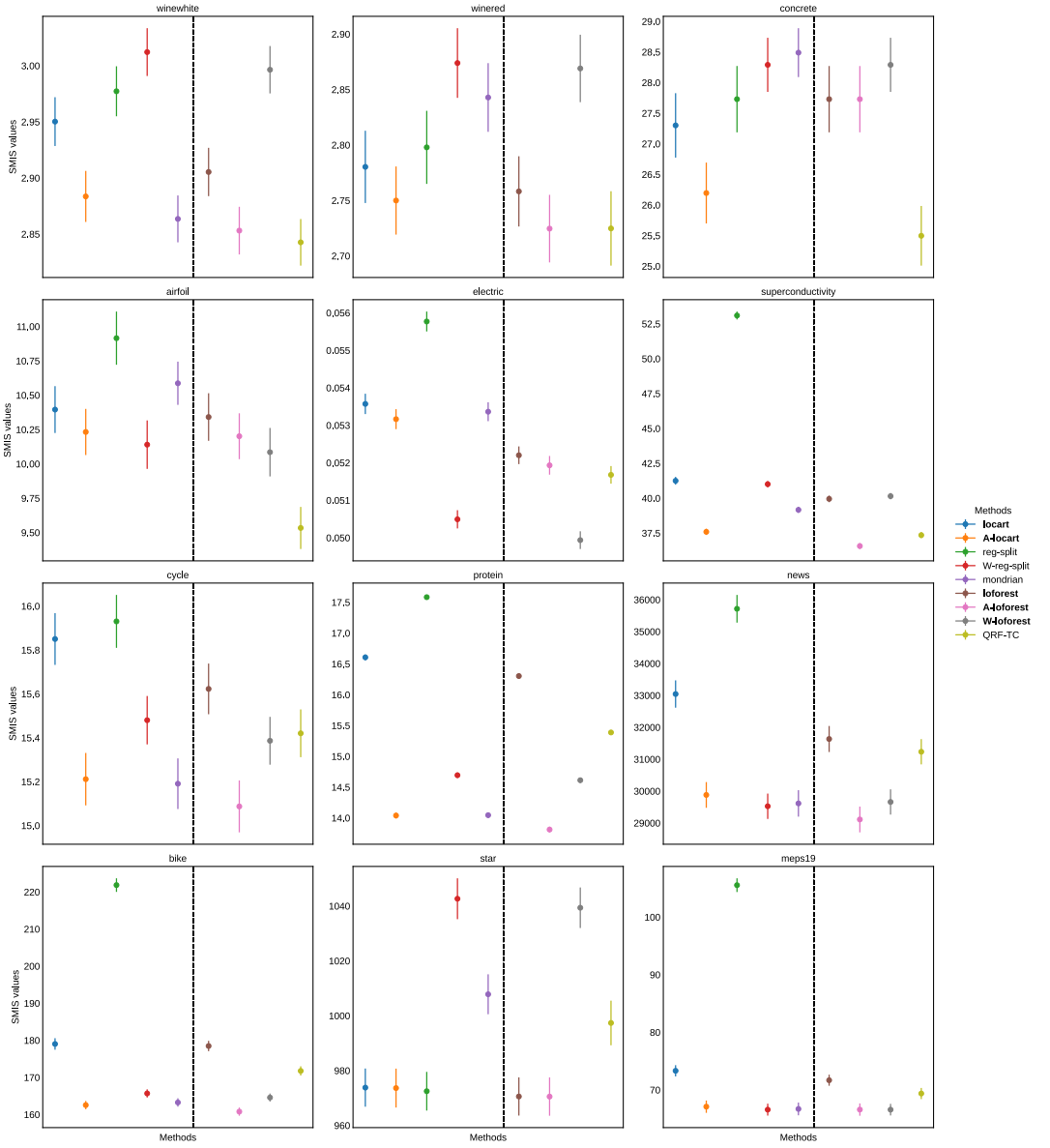

Fig. C.8. SMIS 95% confidence interval of each method for each small/medium sized real dataset. The black dotted line separates the conformal from the non-conformal methods. We observe that `A-Locart` and `A-Loforest` stand out in several datasets and perform well in general.

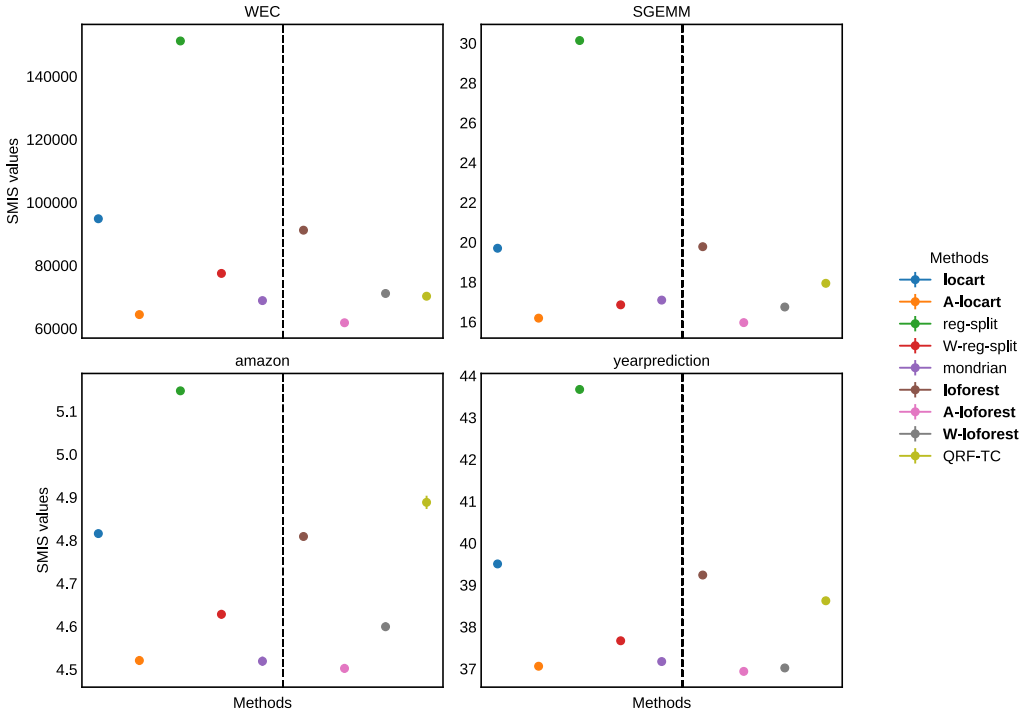

Fig. C.9. SMIS 95% confidence interval of each method for each large real dataset. The black dotted line separates the conformal from the non-conformal methods. We observe that `A-Locart` and `A-Loforest` stand out in practically all big datasets and have superior performance in general.

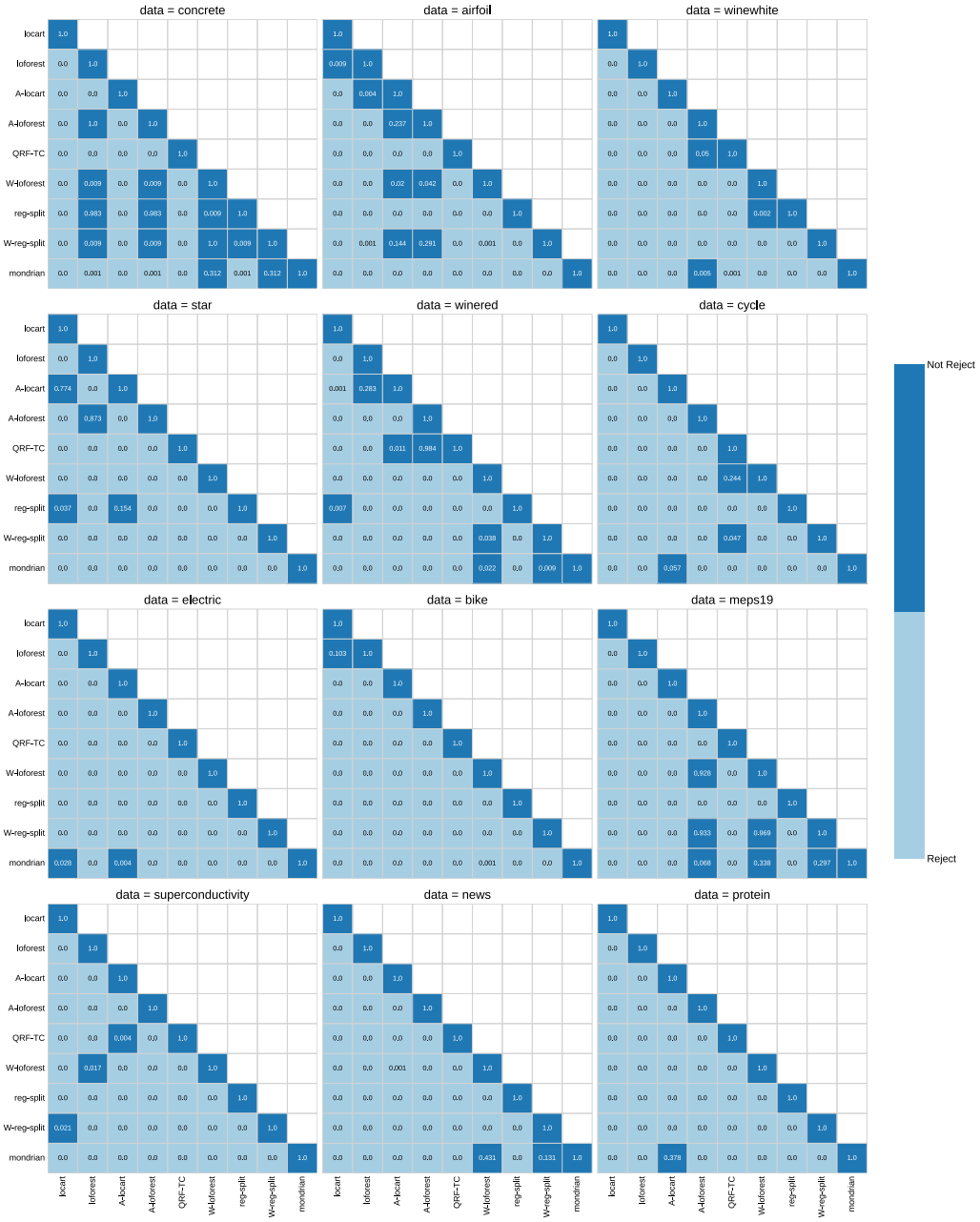

Fig. C.10. Heatmap of *p*-values for pairwise comparisons in real data settings for all small/medium-sized datasets. Dark blue entries indicate comparisons with no significant differences in the average SMIS, while light blue entries indicate comparisons with significant differences. *p*-values less than 0.001 are considered zero.

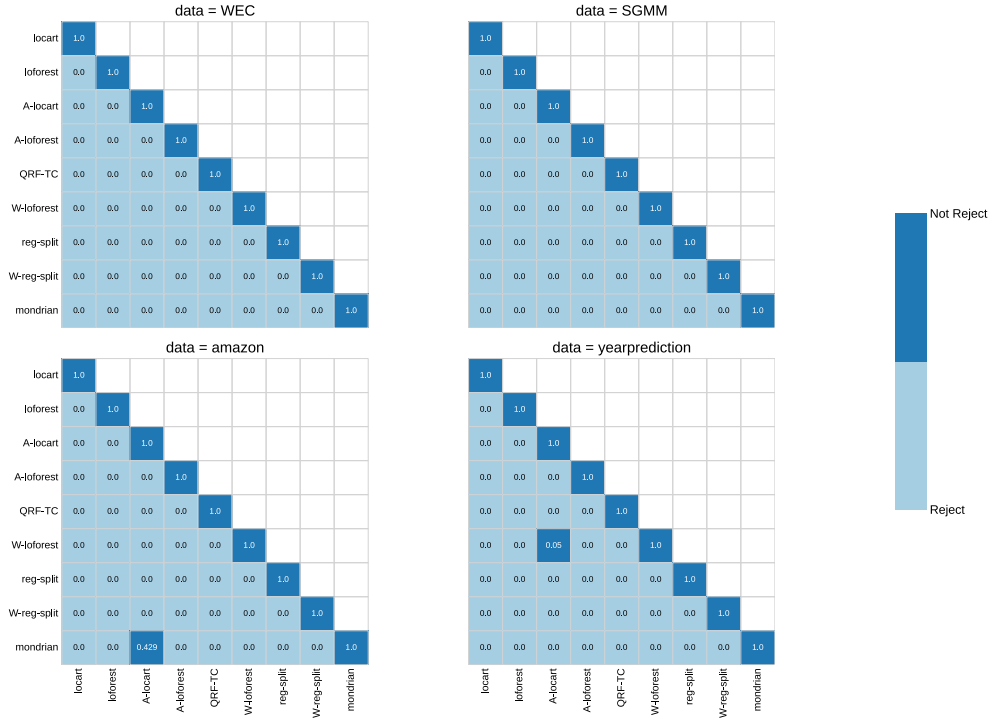

Fig. C.11. Heatmap of p-values for pairwise comparisons in real data settings for all large datasets. Dark blue entries indicate comparisons with no significant differences in the average SMIS, while light blue entries indicate comparisons with significant differences. p-values less than 0.001 are considered zero.

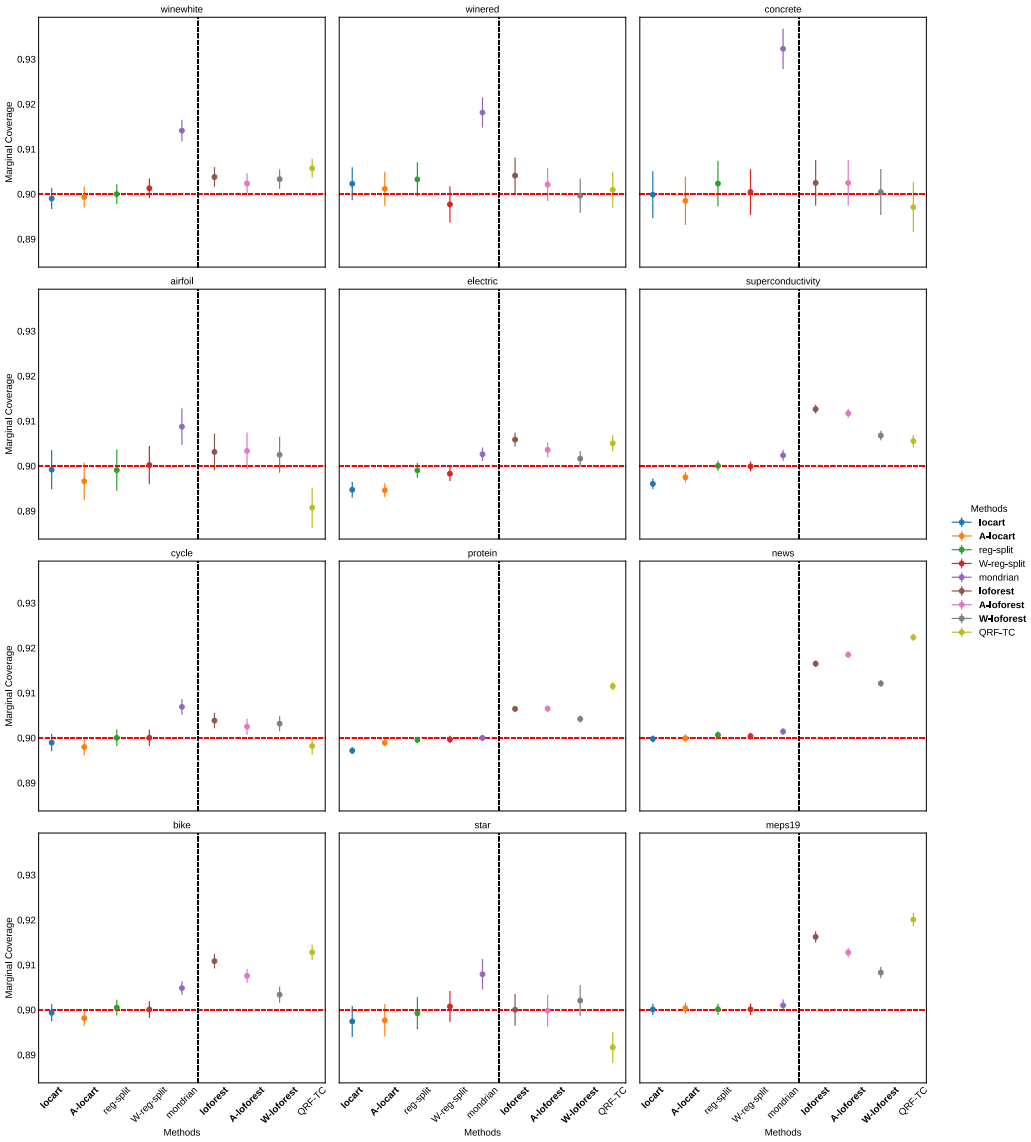

Fig. C.12. Marginal coverage 95% confidence interval of each method for each small/medium sized real dataset. The red dotted line indicates the nominal level ($1 - \alpha = 0.9$) and the black dotted line separates the conformal from the non-conformal methods. In general, all conformal methods are close to the nominal level for each real dataset, while the non-conformal ones present a slight over-coverage in some of the datasets.

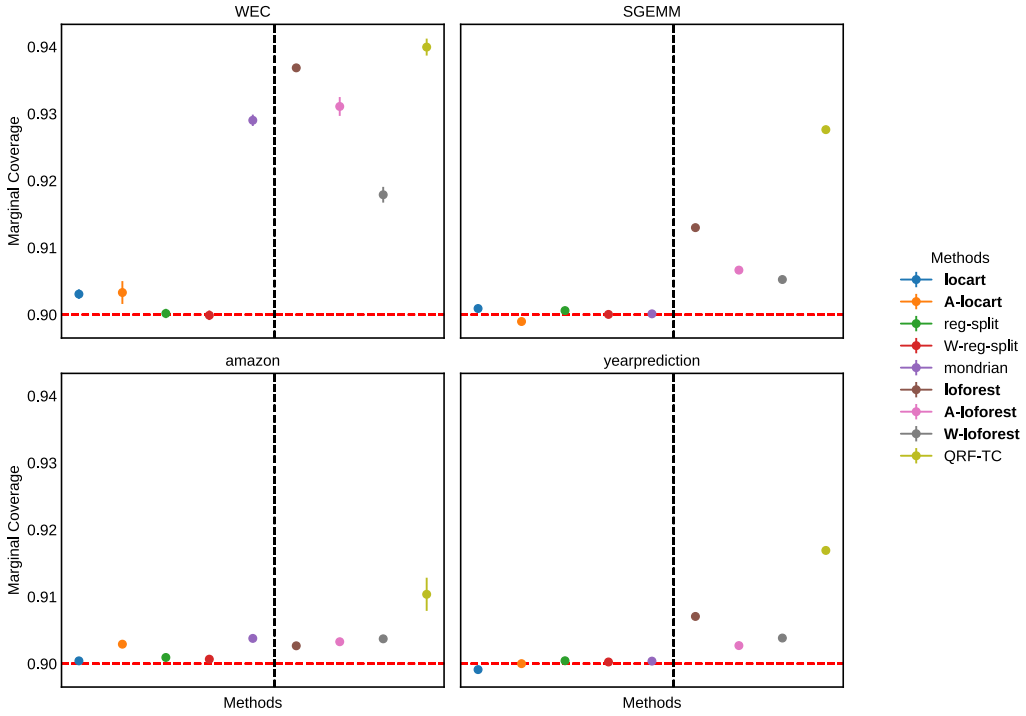

Fig. C.13. Marginal coverage 95% confidence interval of each method for each large real dataset. The red dotted line indicates the nominal level ($1-\alpha=0.9$) and the black dotted line separates the conformal from the non-conformal methods. In general, except Mondrian for the WEC dataset, all conformal methods are close to the nominal level for each real dataset, while the non-conformal ones present a slight over-coverage in some of the datasets.

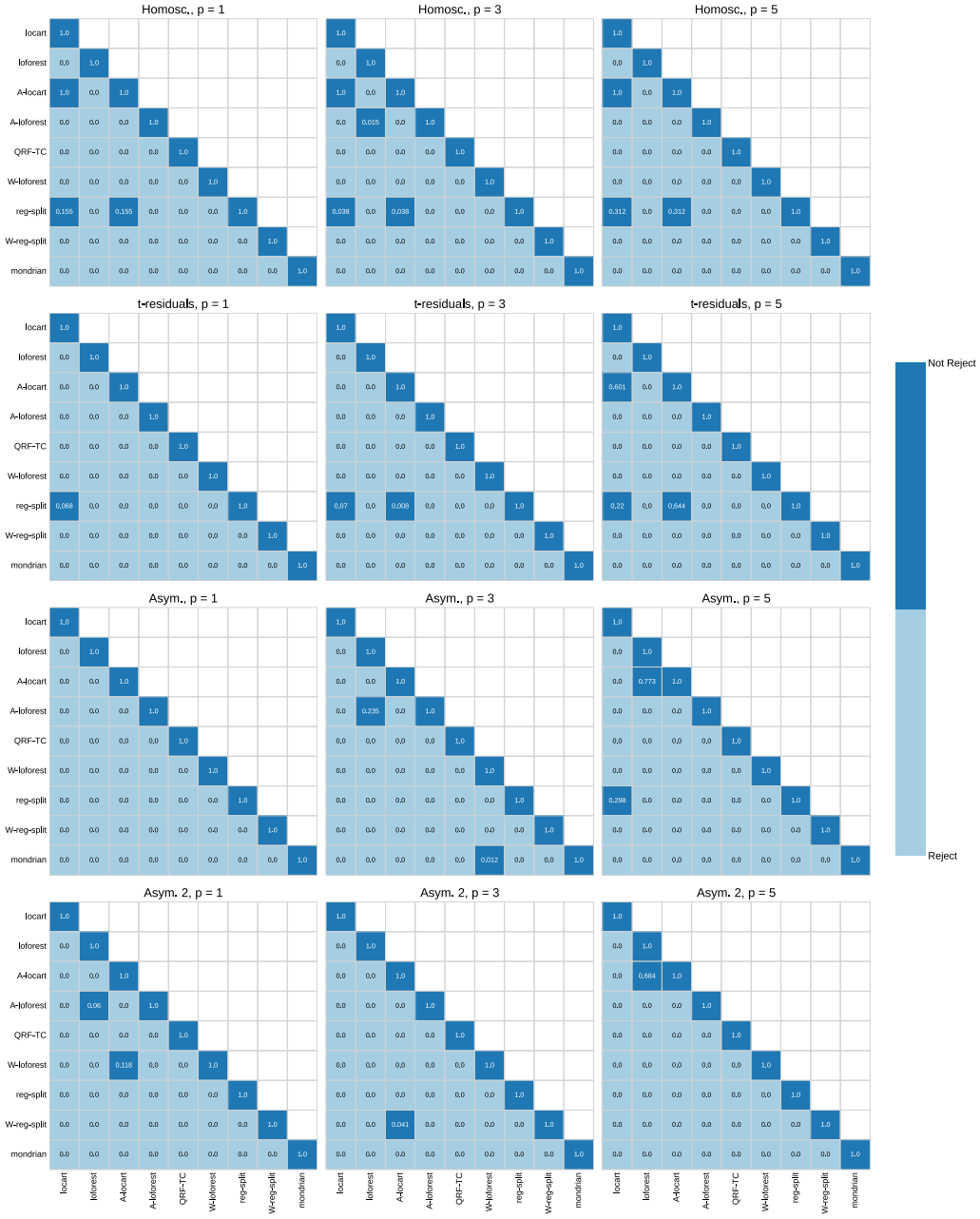

Fig. C.14. Heatmap of *p*-values for pairwise comparisons in simulated settings for the following scenarios: homoscedastic, t-residuals, asymmetric, and asymmetric 2. Dark blue entries indicate comparisons with no significant differences in the average CCAD, while light blue entries indicate comparisons with significant differences. *p*-values less than 0.001 are considered zero.

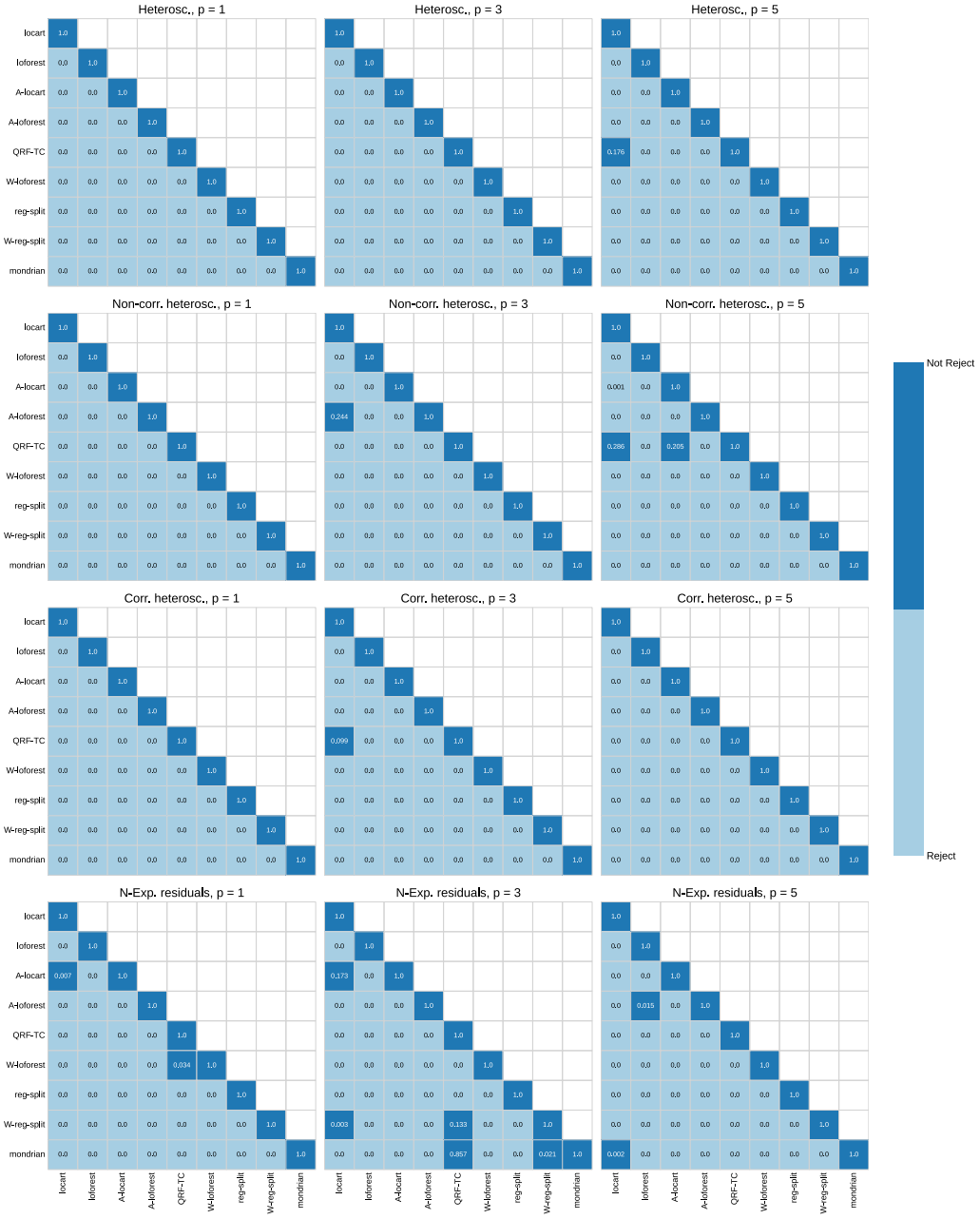

Fig. C.15. Heatmap of *p*-values for pairwise comparisons in simulated settings for the following scenarios: heteroscedastic, non-correlated heteroscedastic, correlated heteroscedastic, Normal-Exponential residuals. Dark blue entries indicate comparisons with no significant differences in the average CCAD, while light blue entries indicate comparisons with significant differences. *p*-values less than 0.001 are considered zero.

Table C.5
Mean marginal coverage values of each method for each kind of simulation. The average across 100 simulations is reported with two times the standard error in parentheses. Values in bold indicate methods with the most distant marginal coverage from the nominal ($1-\alpha=0.9$).

kind	p	Locart	A-Locart	RegSplit	WRegSplit	Mondrian	Loforest	A-Loforest	W-Loforest	QRF-TC
Asym.	1	0.897	0.898	0.898	0.898	0.9	0.903	**0.906**	0.905	**0.906**
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.001)	**(0.001)**
Asym.	3	0.897	0.898	0.898	0.898	0.9	**0.906**	0.905	0.906	0.9
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.001)	(0.001)	(0.002)
Asym.	5	0.898	0.899	0.899	0.898	0.9	0.905	0.905	**0.906**	0.898
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.002)
Asym. 2	1	0.897	0.898	0.9	0.899	0.9	0.904	0.906	0.905	**0.913**
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**
Asym. 2	3	0.893	0.897	0.896	0.897	0.898	**0.911**	0.904	0.904	0.906
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.001)	(0.001)	(0.001)
Asym. 2	5	0.899	0.899	0.9	0.9	0.901	**0.911**	0.908	0.908	0.904
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.001)	(0.001)	(0.001)
Heterosc.	1	0.898	0.899	0.898	0.901	0.902	0.902	**0.903**	0.902	0.902
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.001)	(0.001)
Heterosc.	3	0.896	0.9	0.899	0.9	0.902	**0.906**	0.902	0.902	0.9
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.001)	(0.001)	(0.001)
Heterosc.	5	0.899	0.9	0.9	0.9	0.902	**0.904**	0.902	0.902	0.899
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.001)	(0.001)	(0.002)
Homosc.	1	0.9	0.9	0.9	0.901	0.902	0.901	0.901	0.902	**0.897**
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.002)**
Homosc.	3	0.901	0.901	0.901	0.901	0.903	0.902	0.902	0.902	**0.896**
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.002)**
Homosc.	5	0.901	0.901	0.901	0.901	0.902	0.902	0.901	0.903	**0.895**
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**
Non-corr. heterosc.	1	0.898	0.898	0.898	0.9	0.9	**0.902**	**0.902**	**0.902**	**0.902**
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	**(0.001)**	**(0.001)**	**(0.001)**
Non-corr. heterosc.	3	0.896	0.897	0.899	0.9	0.901	**0.906**	0.904	0.902	0.9
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.001)	(0.001)	(0.001)
Non-corr. heterosc.	5	0.898	0.899	0.9	0.9	0.901	**0.904**	0.902	0.902	0.898
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.001)	(0.001)	(0.002)
t-residuals	1	0.9	0.9	0.9	0.901	0.902	0.903	**0.904**	**0.904**	0.897
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	**(0.001)**	(0.002)
t-residuals	3	0.901	0.901	0.901	0.901	0.903	0.904	0.904	**0.905**	0.898
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.002)
t-residuals	5	0.901	0.901	0.901	0.902	0.903	0.904	0.904	**0.906**	0.898
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.002)
N-Exp. residuals	1	0.898	0.899	0.9	0.9	0.902	0.902	0.902	0.902	**0.903**
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.002)**
N-Exp. residuals	3	0.895	0.899	0.901	0.9	0.902	**0.904**	0.902	0.902	0.902
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.001)	(0.001)	(0.001)
N-Exp. residuals	5	0.894	0.898	0.899	0.899	0.901	**0.904**	0.902	0.901	0.901
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.001)	(0.001)	(0.001)
Corr. heterosc.	1	0.898	0.899	0.9	0.9	0.902	0.902	0.902	0.902	**0.903**
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**
Corr. heterosc.	3	0.895	0.898	0.901	0.901	0.902	**0.904**	0.902	0.902	0.902
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.001)	(0.001)	(0.001)
Corr. heterosc.	5	0.894	0.898	0.899	0.899	0.902	**0.904**	0.902	0.901	0.9
		(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	**(0.001)**	(0.001)	(0.001)	(0.001)

References

[1] S.I. Amoukou, N.J.B. Brunel, Adaptive conformal prediction by reweighting nonconformity score, arXiv:2303.12695, 2023.

[2] A.N. Angelopoulos, S. Bates, M. Jordan, J. Malik, Uncertainty sets for image classifiers using conformal prediction, in: International Conference on Learning Representations, 2021, https://openreview.net/forum?id=eNdiU_DbM9.

[3] R.F. Barber, E.J. Candès, A. Ramdas, R.J. Tibshirani, Predictive inference with the jackknife+, Ann. Stat. 49 (2021) 486–507, https://doi.org/10.1214/20-AOS1965.

[4] S. Barocas, M. Hardt, A. Narayanan, Fairness and Machine Learning: Limitations and Opportunities, MIT Press, 2023, Chapter 3.

[5] D. Batra, S. Mercuri, R. Khraishi, Conformal predictions for longitudinal data, https://doi.org/10.48550/ARXIV.2310.02863, arXiv:2310.02863, 2023.

[6] H. Boström, U. Johansson, Mondrian conformal regressors, in: A. Gammerman, V. Vovk, Z. Luo, E. Smirnov, G. Cherubin (Eds.), Proceedings of the Ninth Symposium on Conformal and Probabilistic Prediction and Applications, PMLR, 2020, pp. 114–133, https://proceedings.mlr.press/v128/bostrom20a.html.

[7] H. Boström, U. Johansson, T. Löfström, Mondrian conformal predictive distributions, in: L. Carlsson, Z. Luo, G. Cherubin, K. An Nguyen (Eds.), Proceedings of the Tenth Symposium on Conformal and Probabilistic Prediction and Applications, PMLR, 2021, pp. 24–38, https://proceedings.mlr.press/v152/bostrom21a.html.

[8] H. Boström, H. Linusson, T. Löfström, U. Johansson, Accelerating difficulty estimation for conformal regression forests, Ann. Math. Artif. Intell. 81 (2017) 125–144, https://api.semanticscholar.org/CorpusID:26467763.

[9] L. Breiman, Classification and Regression Trees, Chapman & Hall, 1993.

[10] L. Breiman, Out-of-bag estimation, Technical Report, Statistics Department, University of California, 1996, https://www.stat.berkeley.edu/users/breiman/OOBestimation.pdf.

[11] D. Cevid, L. Michel, J. Näf, P. Bühlmann, N. Meinshausen, Distributional random forests: heterogeneity adjustment and multivariate distributional regression, J. Mach. Learn. Res. 23 (2022) 1–79, http://jmlr.org/papers/v23/21-0585.html.

[12] V. Chernozhukov, K. Wüthrich, Y. Zhu, Distributional conformal prediction, Proc. Natl. Acad. Sci. 118 (2021).

[13] N. Dewolf, B.D. Baets, W. Waegeman, Valid prediction intervals for regression problems, Artif. Intell. Rev. 56 (2023) 577–613, https://doi.org/10.1007/s10462-022-10178-5.

[14] B. Dey, D. Zhao, J.A. Newman, B.H. Andrews, R. Izbicki, A.B. Lee, Conditionally calibrated predictive distributions by probability-probability map: application to galaxy redshift estimation and probabilistic forecasting, https://doi.org/10.48550/ARXIV.2205.14568, arXiv:2205.14568, 2022.

[15] T. Ding, A.N. Angelopoulos, S. Bates, M.I. Jordan, R.J. Tibshirani, Class-conditional conformal prediction with many classes, https://doi.org/10.48550/ARXIV.2306.09335, arXiv:2306.09335, 2023.

[16] R. Foygel Barber, E.J. Candès, A. Ramdas, R.J. Tibshirani, The limits of distribution-free conditional predictive inference, Inf. Inference, J. IMA 10 (2020) 455–482, https://doi.org/10.1093/imaiai/iaaa017.

[17] D. Fradkin, D. Madigan, Experiments with random projections for machine learning, in: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, New York, NY, USA, 2003, pp. 517–522, https://doi.org/10.1145/956750.956812.

[18] M. Gasparin, A. Ramdas, Conformal online model aggregation, https://doi.org/10.48550/ARXIV.2403.15527, arXiv:2403.15527, 2024.

[19] T. Gneiting, A.E. Raftery, Strictly proper scoring rules, prediction, and estimation, J. Am. Stat. Assoc. 102 (2007) 359–378, https://doi.org/10.1198/016214506000001437.

[20] L. Guan, Localized conformal prediction: a generalized inference framework for conformal prediction, Biometrika 110 (2023) 33–50, https://arxiv.org/abs/1908.08558.

[21] C. Gupta, A. Podkopaev, A. Ramdas, Distribution-free binary classification: prediction sets, confidence intervals and calibration, Adv. Neural Inf. Process. Syst. 33 (2020) 3711–3723.

[22] R. Izbicki, G. Shimizu, R. Stern, Flexible distribution-free conditional predictive bands using density estimators, in: S. Chiappa, R. Calandra (Eds.), Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, PMLR, 2020, pp. 3068–3077, https://proceedings.mlr.press/v108/izbicki20a.html.

[23] R. Izbicki, G. Shimizu, R.B. Stern, CD-split and HPD-split: efficient conformal regions in high dimensions, J. Mach. Learn. Res. 23 (2022) 1–32, http://jmlr.org/papers/v23/20-797.html.

[24] U. Johansson, H. Boström, T. Löfström, H. Linusson, Regression conformal prediction with random forests, Mach. Learn. 97 (2014) 155–176, https://doi.org/10.1007/s10994-014-5453-0.

[25] U. Johansson, C. Sönströd, H. Linusson, H. Boström, Regression trees for streaming data with local performance guarantees, in: 2014 IEEE International Conference on Big Data (Big Data), IEEE, 2014, pp. 461–470.

[26] K. Kasa, Z. Zhang, H. Yang, G.W. Taylor, Adapting conformal prediction to distribution shifts without labels, https://doi.org/10.48550/ARXIV.2406.01416, arXiv:2406.01416, 2024.

[27] B. Kim, C. Xu, R. Barber, Predictive inference is free with the jackknife+-after-bootstrap, in: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc., 2020, pp. 4138–4149, https://proceedings.neurips.cc/paper_files/paper/2020/file/2b346a0aa375a07f5a90a344a61416c4-Paper.pdf.

[28] R. Koenker, Quantile Regression, Econometric Society Monographs, Cambridge University Press, 2005.

[29] J. Lei, M. G'Sell, A. Rinaldo, R.J. Tibshirani, L. Wasserman, Distribution-free predictive inference for regression, J. Am. Stat. Assoc. 113 (2018) 1094–1111.

[30] J. Lei, L. Wasserman, Distribution-free prediction bands for non-parametric regression, J. R. Stat. Soc., Ser. B, Stat. Methodol. 76 (2013) 71–96, https://doi.org/10.1111/rssb.12021.

[31] Y. Lin, Y. Jeon, Random forests and adaptive nearest neighbors, J. Am. Stat. Assoc. 101 (2006) 578–590, https://doi.org/10.1198/016214505000001230.

[32] H. Linusson, U. Johansson, H. Boström, Efficient conformal predictor ensembles, Neurocomputing 397 (2020) 266–278, https://doi.org/10.1016/j.neucom.2019.07.113, https://www.sciencedirect.com/science/article/pii/S0925231219316108.

[33] H. Linusson, I. Samsten, Z. Zając, M. Villanueva, nonconformist: Python implementation of the conformal prediction framework, https://github.com/donlnz/nonconformist, 2021.

[34] N. Meinshausen, Quantile regression forests, J. Mach. Learn. Res. 7 (2006) 983–999, http://jmlr.org/papers/v7/meinshausen06a.html.

[35] D. Nolte, S. Ghosh, R. Pal, Efficient normalized conformal prediction and uncertainty quantification for anti-cancer drug sensitivity prediction with deep regression forests, https://doi.org/10.48550/ARXIV.2402.14080, arXiv:2402.14080, 2024.

[36] H. Papadopoulos, A. Gammerman, V. Vovk, Normalized nonconformity measures for regression conformal prediction, in: Proceedings of the IASTED International Conference on Artificial Intelligence and Applications (AIA 2008), 2008, pp. 64–69.

[37] H. Papadopoulos, K. Proedrou, V. Vovk, A. Gammerman, Inductive confidence machines for regression, in: European Conference on Machine Learning, Springer, 2002, pp. 345–356.

[38] H. Papadopoulos, V. Vovk, A. Gammerman, Regression conformal prediction with nearest neighbours, J. Artif. Intell. Res. 40 (2011) 815–840.

[39] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, Édouard Duchesnay, Scikit-learn: machine learning in python, J. Mach. Learn. Res. 12 (2011) 2825–2830, http://jmlr.org/papers/v12/pedregosa11a.html.

[40] A. Rahimi, B. Recht, Random features for large-scale kernel machines, in: J. Platt, D. Koller, Y. Singer, S. Roweis (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc., 2007, https://proceedings.neurips.cc/paper_files/paper/2007/file/013a006f03dbc5392effeb8f18fda755-Paper.pdf.

[41] Y. Romano, E. Patterson, E. Candes, Conformalized quantile regression, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc., 2019, https://proceedings.neurips.cc/paper_files/paper/2019/file/5103c3584b063c431bd1268e9b5e76fb-Paper.pdf.

[42] Y. Romano, M. Sesia, E.J. Candès, Classification with valid and adaptive coverage, https://doi.org/10.48550/ARXIV.2006.02544, arXiv:2006.02544, 2020.

[43] R.J. Tibshirani, R.F. Barber, E.J. Candès, A. Ramdas, Conformal prediction under covariate shift, in: Proceedings of the 33rd International Conference on Neural Information Processing Systems, 2019, pp. 2530–2540.

[44] D. Valle, R. Izbicki, R.V. Leite, Quantifying uncertainty in land-use land-cover classification using conformal statistics, Remote Sens. Environ. 295 (2023) 113682, https://doi.org/10.1016/j.rse.2023.113682, https://www.sciencedirect.com/science/article/pii/S003442572300233X.

[45] V. Vovk, Conditional validity of inductive conformal predictors, in: S.C.H. Hoi, W. Buntine (Eds.), Proceedings of the Asian Conference on Machine Learning, PMLR, Singapore Management University, Singapore, 2012, pp. 475–490, https://proceedings.mlr.press/v25/vovk12.html.

[46] V. Vovk, et al., Algorithmic Learning in a Random World, Springer Science & Business Media, 2005.

[47] Z. Yang, E. Candès, L. Lei, Bellman conformal inference: calibrating prediction intervals for time series, https://doi.org/10.48550/ARXIV.2402.05203, arXiv:2402.05203, 2024.

[48] S. Zhao, T. Ma, S. Ermon, Individual calibration with randomized forecasting, in: H.D. III, A. Singh (Eds.), Proceedings of the 37th International Conference on Machine Learning, PMLR, 2020, pp. 11387–11397, https://proceedings.mlr.press/v119/zhao20e.html.