

# HDI-Forest: Highest Density Interval Regression Forest

Lin Zhu\*, Jiaxing Lu and Yihong Chen

Ctrip Travel Network Technology Co., Limited.

{zhulb, lujx, yihongchen}@Ctrip.com

## Abstract

By seeking the narrowest prediction intervals (PIs) that satisfy the specified coverage probability requirements, the recently proposed quality-based PI learning principle can extract high-quality PIs that better summarize the predictive certainty in regression tasks, and has been widely applied to solve many practical problems. Currently, the state-of-the-art quality-based PI estimation methods are based on deep neural networks or linear models. In this paper, we propose Highest Density Interval Regression Forest (HDI-Forest), a novel quality-based PI estimation method that is instead based on Random Forest. HDI-Forest does not require additional model training, and directly reuses the trees learned in a standard Random Forest model. By utilizing the special properties of Random Forest, HDI-Forest could efficiently and more directly optimize the PI quality metrics. Extensive experiments on benchmark datasets show that HDI-Forest significantly outperforms previous approaches, reducing the average PI width by over 20% while achieving the same or better coverage probability.

## 1 Introduction

Let  $D_{XY}$  be an unknown joint distribution over instances  $x \in \mathcal{X}$  and responses  $y \in \mathbb{R}$ , where  $X, Y$  denote random variables, and  $x, y$  are their instantiations. A common goal shared by many predictive tasks is to infer certain properties of the conditional distribution  $D_{Y|X}$ . For example, in a standard regression task, we are given a training set of  $\{(x_i, y_i)\}_{i=1}^n$  sampled i.i.d. from  $D_{XY}$  and a new test instance  $x$  sampled from  $D_X$ , the goal is to predict  $E(Y|X=x)$ , namely the conditional mean of  $Y$  at  $x$ . Although estimation of the mean is highly useful in practice, it conveys no information about the predictive uncertainty, which can be very important for improving the reliability and robustness of the predictions [Khosravi *et al.*, 2011b; Pearce *et al.*, 2018].

Prediction intervals (PIs) is one of the most widely used tools for quantifying and representing the uncertainty of pre-

dictions. For a specified confidence level  $\alpha$ , the goal of PI construction is to estimate the  $100(1-\alpha)\%$  interval  $[l, u] \in \mathbb{R}^2$  that will cover no less than  $1-\alpha$  of the probability mass of  $D_{Y|X=x}$ , namely:

$$P(l \leq Y \leq u | X = x) \geq 1 - \alpha. \quad (1)$$

PIs directly express uncertainty by providing lower and upper bounds for each prediction with specified coverage probability, and are more informative and useful for decision making than conditional means alone [Pearce *et al.*, 2018; Stine, 1985].

A plethora of techniques have been proposed in the literature for construction of PIs [Rosenfeld *et al.*, 2018]. However, the majority of existing works only consider the coverage probability criteria (1), and yet ignore other crucial aspects of the elicited PIs [Khosravi *et al.*, 2011b]. In particular, there exists a fundamental trade-off between PI width and coverage probability, and (1) can always be trivially fulfilled by a large enough yet useless interval [Rosenfeld *et al.*, 2018]. This motivates the development of *quality-based* PI elicitation principle, which seeks the shortest PI that contains the required amount of probability [Pearce *et al.*, 2018]. So far, quality-based PI estimation has been applied to solve many practical problems, such as the predictions of electronic price [Shrivastava *et al.*, 2015], wind speed [Lian *et al.*, 2016], and solar energy [Galván *et al.*, 2017], etc.

Although some promising results have been shown, existing approaches for quality-based PI construction still have some limitations. Firstly, most of these methods are built upon deep neural networks (DNNs) [Khosravi *et al.*, 2011b; Pearce *et al.*, 2018], and yet currently the quality-based PI learning principle has primarily been applied to handle tabular data. Despite the tremendous success of DNNs for various domains such as image and text processing, it is known that for tabular data, tree-based ensembles such as Random Forest [Breiman, 2001] and Gradient Boosting Decision Tree (GBDT) [Friedman, 2001] often perform better, and are more widely used in practice [Klambauer *et al.*, 2017; Huang *et al.*, 2015; Feng *et al.*, 2018]. Secondly, quality-based PI learning objectives are generally non-convex, non-differentiable, and even discontinuous, and are thus difficult to optimize. Although existing methods partially solved this problem by optimizing continuous and differentiable surrogate functions instead [Pearce *et al.*, 2018], the overall predic-

\*Contact Author

tive performance may be improved by resolving such a mismatch between the objective functions used in training and the final evaluation metrics used in testing.

Motivated by the above considerations, in this paper we propose Highest Density Interval Regression Forest (HDI-Forest), a novel quality-based PI estimation method based on Random Forest. HDI-Forest does not require additional model training, and directly reuses the trees learned in a standard Random Forest model. By utilizing the special properties of Random Forest introduced in previous works [Lin and Jeon, 2006; Meinshausen, 2006], HDI-Forest could efficiently and more directly optimize the PI quality evaluation metrics. Experiments on benchmark datasets show that HDI-Forest significantly outperforms previous approaches in terms of PI quality metrics.

The rest of the paper is organized as follows. In Section 2 we review related work on prediction intervals. In Section 3, the mechanism of Random Forest is introduced, along with its interpretation as an approximate nearest neighbor method [Lin and Jeon, 2006]. Using this interpretation, HDI-Forest is introduced in Section 4 as a generalization of Random Forest. Encouraging numerical results for benchmark data sets are presented in Section 5.

## 2 Related Work

### 2.1 Quantile-based PI Estimation

For the pair of random variables  $(X, Y)$ , the *conditional quantile*  $Q_\tau(x)$  is the cut point that satisfies:

$$P(Y \leq Q_\tau(x) | X = x) = \tau. \quad (2)$$

It is easy to verify that the equal-tailed interval  $[l, u] = [Q_{\alpha/2}(x), Q_{1-\alpha/2}(x)]$  is a valid solution to (1). Based on this insight, the classic approach for constructing PIs would first estimate  $D_{Y|X=x}$ , and then estimate its quantiles as solutions [Rosenfeld *et al.*, 2018]. If  $D_{Y|X=x}$  is assumed to have some parametric form (e.g., Gaussian), it is often possible to compute the desired quantiles in closed-form. However, explicit assumptions about the conditional distribution may be too restrictive for real-world data modeling [Sharpe, 1970], and considerable research has been devoted to tackle this limitation. For example, Quantile Regression [Koenker and Hallock, 2001] avoid explicit specification of the conditional distribution, and directly infer the conditional quantiles from data by minimizing an asymmetric variant of the absolute loss. On the other hand, re-sampling-based approaches would train a number of models on different re-sampled versions of the training dataset, where commonly used re-sampling techniques include leave-one-out [Steinberger and Leeb, 2016] and bootstrap [Stine, 1985], and then use the point forecasts of trained models to estimate the conditional quantiles. Due to the need for training multiple models, a major disadvantage of re-sampling-based methods is the high computational cost for large-scale datasets [Rivals and Personnaz, 2000; Khosravi *et al.*, 2011b]. A notable exception is quantile regression forest [Meinshausen, 2006], which estimates quantiles by using the sets of local weights generated by Random Forest. Quantile regression forest is based on the alternative interpretation of Random Forest as an approximate

nearest neighbor method [Lin and Jeon, 2006], the details of which will be reviewed in Section 3.1.

In recent years, motivated by the impressive successes of Deep Neural Network (DNN) models in miscellaneous machine learning tasks, there has been growing interest in enhancing DNN algorithms with uncertainty estimation capabilities. For instance, Mean Variance Estimation (MVE) [Khosravi and Nahavandi, 2014; Lakshminarayanan *et al.*, 2017] assumes the conditional  $D_{Y|X=x}$  to be Gaussian, and jointly predicts its mean and variance via maximum likelihood estimation (MLE), while [Gal and Ghahramani, 2016] propose using Monte Carlo dropout [Srivastava *et al.*, 2014] to estimate predictive uncertainty. We refer the interested reader to [Pearce *et al.*, 2018] for a up-to-date overview of related techniques. Despite the differences in learning principles, most existing DNN-based approaches still adopt the traditional strategy of extracting quantile-based equal-tailed intervals.

### 2.2 Quality-based PI Estimation

As demonstrated in the previous section, existing studies related to PI construction have tended to focus on the intermediate problems of estimating conditional distributions and quantiles, and yet very little attention has been paid to quantitatively examine the quality of elicited PIs [Khosravi *et al.*, 2011b]. The coverage criteria (1) could be adopted to evaluate the constructed intervals, but (1) alone is not sufficient for determining a meaningful PI. For example, if the intervals are set to be wide enough (e.g.,  $[-\infty, +\infty]$ ), the true response values could always be contained therein. This phenomenon emphasizes a fundamental trade-off between coverage probability and width of the PI [Khosravi *et al.*, 2011a], and motivates the following quality-based criteria for determining optimal PIs [Pearce *et al.*, 2018]:

$$\begin{aligned} & \min_{l, u} u - l \\ & \text{s.t. } P(l \leq Y \leq u | X = x) \geq 1 - \alpha. \end{aligned} \quad (3)$$

In other words, given any  $0 \leq \alpha \leq 1$ , we would like to find the shortest interval that covers the required probability mass. Such intervals are known as the highest density intervals in the statistical literature [Box and Tiao, 1973]. An illustrative example that compares highest density and traditional quantile-based equal-tailed intervals is provided in Fig. 1.

So far, a number of methods have been proposed to predict quality-based PIs. The key idea shared by these approaches is to infer model parameters by minimizing a loss function based on (3). The loss function typically consists of two parts, which respectively measures the mean PI width (MPIW) and PI coverage probability (PICP) on the training set [Pearce *et al.*, 2018]. For example, Lower Upper Bound Estimation (LUBE) [Khosravi *et al.*, 2011b] trains neural networks by optimizing

$$\frac{\text{MPIW}}{r} (1 + \exp(\lambda \max(0, (1 - \alpha) - \text{PICP}))), \quad (4)$$

where  $r$  is the numerical range of the response variable,  $\lambda$  controls the trade-off between PICP and MPIW. A limitation of the LUBE loss (4) is that it is non-differentiable and hard

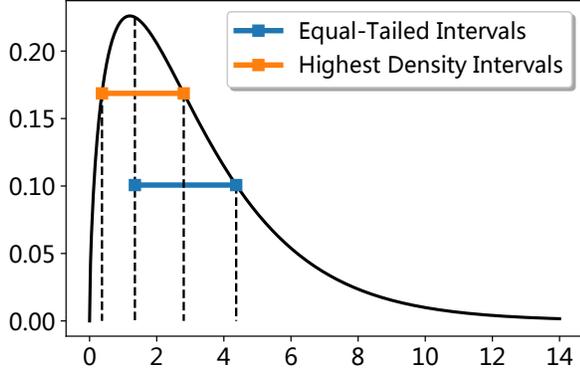


Figure 1: Comparison of 50% highest density and equal-tailed intervals for a Gamma distribution  $\Gamma(0, 2)$ .

to optimize. Quality-driven Neural Network Ensemble (QD-Ens) [Pearce *et al.*, 2018] instead minimizes

$$\text{MPIW}_{\text{capt}} + \lambda \frac{n}{\alpha(1-\alpha)} \max(0, (1-\alpha) - \text{PICP})^2, \quad (5)$$

where  $n$  is the training set size,  $\text{MPIW}_{\text{capt}}$  denotes MPIW of points that fall into the predicted intervals. Experiments show that QD-Ens significantly outperforms previous neural-network-based methods for eliciting quality-based PIs. On the other hand, [Rosenfeld *et al.*, 2018] propose IntPred for quality-based PI construction in the batch learning setting, where PIs for a set of test points are constructed simultaneously. To deal with the non-differentiability of PI quality metrics, both QD-Ens and IntPred instead adopt proxy losses that can be minimized efficiently using standard optimization techniques.

The proposed HDI-Forest model is different from existing quality-based PI estimation works in multiple aspects. Firstly, existing methods mainly consider linear or DNN-based predictive functions, while HDI-Forest is built upon tree ensembles; secondly, HDI-Forest does not require local search heuristics or smooth/convex loss relaxation techniques, and could efficiently obtain global optimal solution of the non-differentiable objective function; finally, by exploiting the special property of Random Forest that will be discussed in the next section, HDI-Forest does not require model re-training for different trade-offs between PICP and MPIW.

### 3 Random Forest

A Random Forest is a predictor consisting of a collection of  $m$  randomized regression trees. Following the notations in [Breiman, 2001; Meinshausen, 2006], each tree  $T(\theta)$  in this collection is constructed based on a random parameter vector  $\theta$ . In practice,  $\theta$  could control various aspects of the tree growing process, such as the re-sampling of the input training set and the successive selections of variables for tree splitting. Once learned, the  $L$  leaves of  $T(\theta)$  partition the input feature space  $\mathcal{X}$  into  $L$  non-overlapping axis-parallel subspaces  $\{\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_L\}$ , then for any  $x \in \mathcal{X}$ , there exists one and only one leaf  $l(x, \theta)$  such that  $x \in \mathcal{X}_{l(x, \theta)}$ . Meanwhile, the prediction of  $T(\theta)$  for  $x$  is given by averaging

over the observations that fall into  $\mathcal{X}_{l(x, \theta)}$ . Concretely, let  $w(x_i, x, \theta)$ ,  $1 \leq i \leq m$  be defined as:

$$w(x_i, x, \theta) = \frac{\mathbb{I}(x_i \in \mathcal{X}_{l(x, \theta)})}{|\{j : x_j \in \mathcal{X}_{l(x, \theta)}\}|}, \quad (6)$$

where  $\mathbb{I}(\cdot)$  is the indicator function and  $|\cdot|$  denotes the cardinality of a set, then

$$\hat{y}_{\text{single-tree}}(x, \theta) = \sum_{i=1}^n w(x_i, x, \theta) y_i. \quad (7)$$

In Random Forest regression, the conditional mean of  $Y$  given  $X = x$  is predicted as the average of predictions of  $m$  trees constructed with i.i.d. parameters  $\theta_i$ ,  $1 \leq i \leq m$ :

$$\hat{y}(x) = \frac{1}{m} \sum_{i=1}^m \hat{y}_{\text{single-tree}}(x, \theta_i). \quad (8)$$

#### 3.1 Random Forest for Conditional Distribution Estimation

Although the original formulation of Random Forest only predicts the conditional mean, the learned trees could also be exploited to predict other interesting quantities [Meinshausen, 2006; Li and Martin, 2017; Feng and Zhou, 2018]. For example, note that (8) can be rearranged as

$$\hat{y}(x) = \sum_{i=1}^n w(x_i, x) y_i, \quad (9)$$

where

$$w(x_i, x) = \frac{1}{m} \sum_{i=1}^m w(x_i, x, \theta_i). \quad (10)$$

Therefore, (8) could be alternatively interpreted as the weighted average of the response values of all training instances, and the weight for a specific instances  $x_i$  measures the frequency that  $x_i$  and  $x$  are partitioned into the same leaf in all grown trees, which offers an intuitive measure of similarity between them. Theoretically, it can also be shown that  $x_i$  tends to be weighted higher if the conditional distributions  $D_{Y|X=x}$  and  $D_{Y|X=x_i}$  are similar [Lin and Jeon, 2006]. Furthermore, note that the conditional cumulative distribution function of  $Y$  given  $X = x$  can be written as:

$$\begin{aligned} F(y|X=x) &= P(Y \leq y|X=x) \\ &= E(\mathbb{I}(Y \leq y)|X=x). \end{aligned} \quad (11)$$

It could be proven that under certain conditions, (11) can be estimated using the weights from (10) as [Meinshausen, 2006]:

$$\hat{F}(y|X=x) = \sum_{i=1}^n w(x_i, x) \mathbb{I}(y_i \leq y). \quad (12)$$

It has been demonstrated in [Meinshausen, 2006] that (12) can be exploited to accurately estimate conditional quantiles. In this work, we instead utilize it to perform quality-based PI estimation, as detailed in the next section.

#### 4 Highest Density Interval Regression Forest

In the section, we describe the proposed HDI-Forest algorithm. Concretely, we first use the standard Random Forest algorithm to infer a number of trees from the data, then based on (12), for any observation  $x$ , the probability that the associated response value would fall into interval  $[l, u]$  can be estimated as:

$$\hat{P}(l \leq Y \leq u | X = x) = \sum_{i=1}^n w(x_i, x) \mathbb{I}(l \leq y_i \leq u). \quad (13)$$

Using (13), the quality-based criteria (3) can be approximated as the following optimization problem:

$$\begin{aligned} \min_{l, u} \quad & u - l \\ \text{s.t.} \quad & \hat{P}(l \leq Y \leq u | X = x) \geq 1 - \alpha. \end{aligned} \quad (14)$$

Note that the optimization problem in (14) is non-convex since its constraint function is piece-wise constant and discontinuous. However, its global optimal solution can still be efficiently obtained by exploiting the problem structure, as detailed below.

Firstly, we present Theorem 1, which shows that the optimal solution of (14) must exist in a pre-defined finite set:

**Theorem 1.** *The optimal solution of (14) satisfies the following conditions:*

$$u, l \in \{y_i\}_{i=1}^n. \quad (15)$$

*Proof.* Assume by contradiction that a pair of  $[l, u]$  optimizes (14) and does not satisfy (15), then

$$\hat{P}(l \leq Y \leq u | X = x) \geq 1 - \alpha. \quad (16)$$

Let  $l_{\text{alt}}$  and  $u_{\text{alt}}$  be defined as

$$l_{\text{alt}} = \min_i \{y_i | y_i \geq l, 1 \leq i \leq n\}, \quad (17)$$

$$u_{\text{alt}} = \max_i \{y_i | y_i < u, 1 \leq i \leq n\}. \quad (18)$$

Recall that  $[l, u]$  does not satisfy (15), thus either  $u_{\text{alt}} \neq u$  or  $l_{\text{alt}} \neq l$ , and

$$u_{\text{alt}} - l_{\text{alt}} < u - l. \quad (19)$$

Meanwhile, by combining (16), (17), and (18), we have

$$\begin{aligned} & \hat{P}(l_{\text{alt}} \leq Y \leq u_{\text{alt}} | X = x) \\ &= \sum_{i=1}^n w(x_i, x) \mathbb{I}(l_{\text{alt}} \leq y_i \leq u_{\text{alt}}) \\ &= \sum_{i=1}^n w(x_i, x) \mathbb{I}(l \leq y_i \leq u) \\ &= \hat{P}(l \leq Y \leq u | X = x) \\ &\geq 1 - \alpha. \end{aligned} \quad (20)$$

Equations (19) and (20) mean that  $[l_{\text{alt}}, u_{\text{alt}}]$  is a feasible and better solution than  $[l, u]$ , which contradicts the assumption that  $[l, u]$  optimizes (14).  $\square$

Let the unique elements of  $\{y_i\}_{i=1}^n$  be arranged in increasing order as  $\tilde{y}_1 < \tilde{y}_2 < \dots < \tilde{y}_{\tilde{n}}$ . Then based on Theorem 1, (14) can be equivalently reformulated as

$$\begin{aligned} \min_{i, j} \quad & \tilde{y}_j - \tilde{y}_i \\ \text{s.t.} \quad & \sum_{k=i}^j w_k \geq 1 - \alpha, \end{aligned} \quad (21)$$

where

$$w_k = \sum_{i=1}^n \mathbb{I}(y_i = \tilde{y}_k) w(x_i, x). \quad (22)$$

Problem (21) can then be solved simply by enumerating and evaluating all pairs of elements from  $\{\tilde{y}_i\}_{i=1}^{\tilde{n}}$ , which nevertheless is still costly and takes  $O(\tilde{n}^2)$  time per prediction. Fortunately, we can reduce the time complexity by rearranging the computations, so that the time is only linear to  $O(\tilde{n})$ . The method is described below.

Firstly, (21) can be optimized using a two-stage approach instead: we start by solving the following optimization problem for each  $1 \leq i \leq \tilde{n}$ :

$$\begin{aligned} \min_j \quad & \tilde{y}_j - \tilde{y}_i \\ \text{s.t.} \quad & \sum_{k=i}^j w_k \geq 1 - \alpha. \end{aligned} \quad (23)$$

Then, let  $\mathcal{I} \subseteq \{i | i = 1, 2, \dots, \tilde{n}\}$  be the set of indices for which (23) has a feasible solution, and the optimal solution of (23) for  $i \in \mathcal{I}$  be denoted as  $j_{\text{opt}}(i)$ , it is easy to verify that (21) is optimized by the pair of  $(i, j_{\text{opt}}(i))$  that attains the smallest  $\tilde{y}_{j_{\text{opt}}(i)} - \tilde{y}_i$ .

To compute  $j_{\text{opt}}(i)$  for  $i \in \mathcal{I}$ , we exploit the strict monotonicity of  $\{\tilde{y}_i\}_{i=1}^{\tilde{n}}$ , and equivalently reformulate (23) as

$$\begin{aligned} \min_j \quad & j \\ \text{s.t.} \quad & \sum_{k=i}^j w_k \geq 1 - \alpha. \end{aligned} \quad (24)$$

In other words,  $j_{\text{opt}}(i)$  is simply the smallest index for which the constraint in (23) holds. Moreover,  $j_{\text{opt}}(i)$  is monotonously increasing with respect to  $i$ :

**Theorem 2.** *For any  $1 \leq i_2 < i_1 \leq \tilde{n}$ , we have*

$$j_{\text{opt}}(i_2) \leq j_{\text{opt}}(i_1). \quad (25)$$

*Proof.* Assume by contradiction that there exist  $i_1$  and  $i_2$  such that  $j_{\text{opt}}(i_2) > j_{\text{opt}}(i_1)$  and  $i_1 > i_2$ , recall from the definitions in (6), (10) and (22) that  $w_k \geq 0, 1 \leq k \leq \tilde{n}$ , therefore

$$\begin{aligned} \sum_{k=i_2}^{j_{\text{opt}}(i_1)} w_k &= \sum_{k=i_1}^{j_{\text{opt}}(i_1)} w_k + \sum_{k=i_2}^{i_1-1} w_k \\ &\geq \sum_{k=i_1}^{j_{\text{opt}}(i_1)} w_k \\ &\geq 1 - \alpha. \end{aligned} \quad (26)$$

**Algorithm 1** Solve (23) for all  $1 \leq i \leq \tilde{n}$

---

**Require:**  $\{\tilde{y}_i\}_{i=1}^{\tilde{n}}$  sorted in increasing order, the associated weights  $\{w_i\}_{i=1}^{\tilde{n}}$ , threshold  $\alpha$

```

1:  $i \leftarrow 1, j \leftarrow 0, w \leftarrow 0$ 
2: for  $1 \leq i \leq n$  do                                ▷ Outer loop
3:   while  $j \leq n$  AND  $w < \alpha$  do                ▷ Inner loop
4:      $j \leftarrow j + 1, w \leftarrow w + w_j$ 
5:   end while
6:   if  $w \geq \alpha$  then                                ▷ (23) has a feasible solution
7:      $j_{\text{opt}}(i) = j$ 
8:   end if
9:    $w \leftarrow w - w_i$ 
10: end for
    
```

---

On the other hand, based on the strict monotonicity of  $\{\tilde{y}_i\}_{i=1}^{\tilde{n}}$  we have

$$\tilde{y}_{j_{\text{opt}}(i_1)} - \tilde{y}_{i_2} < y_{j_{\text{opt}}(i_2)} - \tilde{y}_{i_2}. \quad (27)$$

Equations (26) and (27) contradict the optimality of  $j_{\text{opt}}(i_2)$  and thereby we complete the proof.  $\square$

Based on the above analysis, in order to solve (23) for all  $1 \leq i \leq \tilde{n}$ , we only need to walk down the sorted list of  $\{\tilde{y}_i\}_{i=1}^{\tilde{n}}$  once to identify for each  $i$  the first index  $j$  such that  $\sum_{k=i}^j w_k \geq 1 - \alpha$ . The whole algorithm is presented in Algorithm 1.

## 5 Experiments

### 5.1 Experimental Settings

#### Baseline Methods

Based on the survey of related works in Section 2, we adopted two types of baseline methods for comparison, including quantile-based methods and quality-based methods. Quantile-based methods include Quantile Regression Forest (QRF) [Meinshausen, 2006]<sup>1</sup>, Quantile Regression (QR) [Koenker and Hallock, 2001]<sup>2</sup>, and Gradient Boosting Decision Tree with Quantile Loss (QR<sub>GBDT</sub>) implemented in the Scikit-learn package [Pedregosa *et al.*, 2011]. On the other hand, quality-based PI methods include IntPred [Rosenfeld *et al.*, 2018] and Quality-Driven Ensemble (QD-Ens) [Pearce *et al.*, 2018]<sup>3</sup>, the state-of-the-art approach for neural-network-based PI elicitation.

#### Benchmark Datasets

We compare various methods on 11 datasets from the UCI repository<sup>4</sup>. Statistics of these datasets are presented in Table 1. Each dataset is split in train and test sets according to a 80%-20% scheme, and we report the average performance over 10 random data splits. The hyper-parameters of all tested methods were tuned via 5-fold cross-validation on the training set.

<sup>1</sup><https://cran.r-project.org/web/packages/quantregForest/index.html>

<sup>2</sup><https://cran.r-project.org/web/packages/quantreg/index.html>

<sup>3</sup><https://github.com/TeaPearce/>

<sup>4</sup><http://archive.ics.uci.edu/ml/index.php>

Dataset	Size	Dimensionality
Boston Housing	506	13
Parkinsons	5875	26
Wine Quality	1599	11
Forest Fires	517	13
Concrete Compression Strength	1030	9
Energy Efficiency	768	8
Naval Propulsion	11934	16
Combined Cycle Power Plant	9568	4
Protein Structure	45730	9
Communities	1994	128
Online News Popularity	39797	61

Table 1: Characteristics of the datasets used in the experiments.

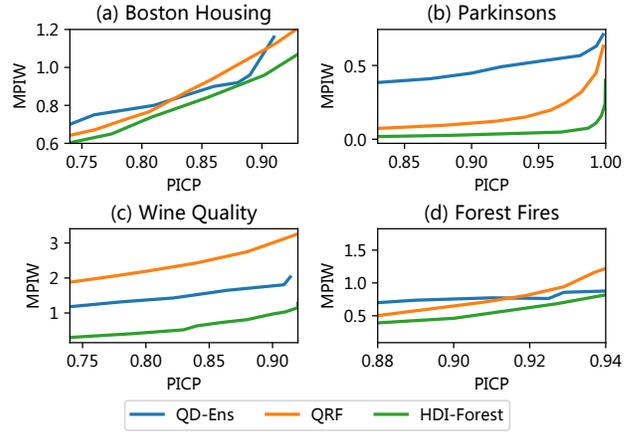


Figure 2: Comparison between three methods by examining the MPIW score as a function of PICP.

#### Evaluation Metrics

Following previous works [Khosravi *et al.*, 2011b; Pearce *et al.*, 2018; Rosenfeld *et al.*, 2018], PICP and MPIW mentioned in Section 2 were adopted as the evaluation metrics.

### 5.2 Performance Comparison

As mentioned earlier, there exists a trade-off between the coverage probability (measured by the PICP metric) and width (measured by the MPIW metric) of extracted PIs. To facilitate the comparison of various methods, we first evaluate their performance when they achieve roughly the same level of PICP. The results are presented in Table 2. As can be seen, HDI-Forest significantly outperforms all other baselines for all but one dataset. Compared with the best-performing baseline method (QD-Ens), HDI-Forest could substantially reduce the average interval width by 34%, while achieving slightly better coverage probability.

To further compare the performance of HDI-Forest against QD-Ens and QRF, the two top-performing baselines, we examine the MPIW scores of three methods for a range of PICP values. As shown in Fig.2, HDI-Forest still achieves the best performance among all models.

Dataset	Metrics	HDI-Forest	QRF	QR	QR <sub>GBDT</sub>	IntPred	QD-Ens
Boston Housing	PICP	<b>0.93</b>	0.92	0.92	0.92	0.92	0.91
	MPIW	<b>1.02</b>	1.19	2.28	1.70	1.79	1.16
Parkinsons	PICP	<b>0.99</b>	0.99	0.95	0.99	0.96	0.98
	MPIW	<b>0.11</b>	0.45	1.18	0.88	0.98	0.62
Wine Quality	PICP	<b>0.92</b>	0.92	0.91	0.92	0.92	0.92
	MPIW	<b>1.11</b>	3.33	2.64	2.47	2.54	2.33
Forest Fires	PICP	<b>0.95</b>	0.94	0.95	0.94	0.94	0.94
	MPIW	<b>0.81</b>	1.24	1.09	1.04	1.03	0.96
Concrete Compression Strength	PICP	0.94	0.94	0.94	0.94	0.94	<b>0.94</b>
	MPIW	1.18	1.22	2.22	2.23	1.87	<b>1.09</b>
Energy Efficiency	PICP	<b>0.97</b>	0.96	0.95	0.97	0.95	0.97
	MPIW	<b>0.39</b>	0.50	1.73	0.79	1.56	0.47
Naval Propulsion	PICP	<b>0.99</b>	0.96	0.98	0.98	0.98	0.98
	MPIW	<b>0.24</b>	0.68	0.89	1.34	0.73	0.28
Combined Cycle Power Plant	PICP	<b>0.95</b>	0.95	0.95	0.95	0.95	0.95
	MPIW	<b>0.75</b>	0.78	0.97	0.90	0.84	0.86
Protein Structure	PICP	<b>0.95</b>	0.94	0.95	0.95	0.95	0.95
	MPIW	<b>1.77</b>	1.82	2.76	2.36	2.15	2.27
Communities	PICP	<b>0.92</b>	0.92	0.89	0.92	0.92	0.87
	MPIW	<b>1.50</b>	1.73	2.03	1.69	1.94	1.74
Online News Popularity	PICP	<b>0.96</b>	0.96	0.95	0.96	0.95	0.96
	MPIW	<b>1.18</b>	1.98	1.27	1.72	1.38	1.60
Average Performance	PICP	<b>0.95</b>	0.95	0.94	0.95	0.94	0.94
	MPIW	<b>0.91</b>	1.35	1.73	1.56	1.53	1.22

Table 2: Performance comparison of various methods. The results are averaged over 20 random runs, with best results in bold. Here, the best was chosen according to the strict rule that its performance should be equal to or better than all other methods measured by both PICP and MPIW. The last two rows show the average performance of all tested methods.

## 6 Conclusion

In this paper, we propose HDI-Forest, a novel algorithm for quality-based PI estimation, extensive experiments on benchmark datasets show that HDI-Forest significantly outperforms previous approaches.

For future work, we plan to extend HDI-Forest to the batch learning setting, where the overall performance on a group of test instances can be further improved by adjusting the per-instance coverage probability constraints [Rosenfeld *et al.*, 2018]. On the other hand, HDI-Forest is based on the original Random Forest model that is mainly suitable for standard regression/classification tasks, however, a large number of Random-Forest-based approaches have been proposed in the literature to handle other types of problems [Sathe and Aggarwal, 2017; Barbieri *et al.*, 2016]. It would also be interesting to study quality-based PI estimation for these models.

## References

- [Barbieri *et al.*, 2016] Nicola Barbieri, Fabrizio Silvestri, and Mounia Lalmas. Improving post-click user engagement on native ads via survival analysis. In *WWW*, pages 761–770, 2016.
- [Box and Tiao, 1973] George EP Box and George C Tiao. *Bayesian inference in statistical analysis*. Wiley, 1973.
- [Breiman, 2001] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
- [Feng and Zhou, 2018] Ji Feng and Zhi-Hua Zhou. Autoencoder by forest. In *AAAI*, 2018.
- [Feng *et al.*, 2018] Ji Feng, Yang Yu, and Zhi-Hua Zhou. Multi-layered gradient boosting decision trees. In *NIPS*, pages 3555–3565, 2018.
- [Friedman, 2001] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232, 2001.
- [Gal and Ghahramani, 2016] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, pages 1050–1059, 2016.
- [Galván *et al.*, 2017] Inés M Galván, José M Valls, Alejandro Cervantes, and Ricardo Aler. Multi-objective evolutionary optimization of prediction intervals for solar energy forecasting with neural networks. *Information Sciences*, 418:363–382, 2017.
- [Huang *et al.*, 2015] Yiqing Huang, Fangzhou Zhu, Mingxuan Yuan, Ke Deng, Yanhua Li, Bing Ni, Wenyuan Dai, Qiang Yang, and Jia Zeng. Telco churn prediction with big data. In *SIGMOD*, pages 607–618, 2015.
- [Khosravi and Nahavandi, 2014] Abbas Khosravi and Saeid Nahavandi. An optimized mean variance estimation method for uncertainty quantification of wind power forecasts. *International Journal of Electrical Power & Energy Systems*, 61:446–454, 2014.
- [Khosravi *et al.*, 2011a] Abbas Khosravi, Saeid Nahavandi, Doug Creighton, and Amir F Atiya. Comprehensive review of neural network-based prediction intervals and

- new advances. *IEEE Transactions on neural networks*, 22(9):1341–1356, 2011.
- [Khosravi *et al.*, 2011b] Abbas Khosravi, Saeid Nahavandi, Doug Creighton, and Amir F Atiya. Lower upper bound estimation method for construction of neural network-based prediction intervals. *IEEE Trans. Neural Networks*, 22(3):337–346, 2011.
- [Klambauer *et al.*, 2017] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. In *NIPS*, pages 971–980, 2017.
- [Koenker and Hallock, 2001] Roger Koenker and Kevin F Hallock. Quantile regression. *Journal of economic perspectives*, 15(4):143–156, 2001.
- [Lakshminarayanan *et al.*, 2017] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NIPS*, pages 6402–6413, 2017.
- [Li and Martin, 2017] Alexander Hanbo Li and Andrew Martin. Forest-type regression with general losses and robust forest. In *ICML*, pages 2091–2100, 2017.
- [Lian *et al.*, 2016] Cheng Lian, Zhigang Zeng, Wei Yao, Huiming Tang, and Chun Lung Philip Chen. Landslide displacement prediction with uncertainty based on neural networks with random hidden weights. *IEEE Trans. Neural Networks and Learning systems*, 27(12):2683–2695, 2016.
- [Lin and Jeon, 2006] Yi Lin and Yongho Jeon. Random forests and adaptive nearest neighbors. *Journal of the American Statistical Association*, 101(474):578–590, 2006.
- [Meinshausen, 2006] Nicolai Meinshausen. Quantile regression forests. *JMLR*, 7(Jun):983–999, 2006.
- [Pearce *et al.*, 2018] Tim Pearce, Alexandra Brintrup, Mohamed Zaki, and Andy Neely. High-quality prediction intervals for deep learning: A distribution-free, ensembled approach. In *ICML*, pages 4075–4084, 2018.
- [Pedregosa *et al.*, 2011] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *JMLR*, 12(Oct):2825–2830, 2011.
- [Rivals and Personnaz, 2000] Isabelle Rivals and Léon Personnaz. Construction of confidence intervals for neural networks based on least squares estimation. *Neural Networks*, 13(4-5):463–484, 2000.
- [Rosenfeld *et al.*, 2018] Nir Rosenfeld, Yishay Mansour, and Elad Yom-Tov. Discriminative learning of prediction intervals. In *AISTATS*, pages 347–355, 2018.
- [Sathe and Aggarwal, 2017] Saket Sathe and Charu C Aggarwal. Similarity forests. In *SIGKDD*, pages 395–403, 2017.
- [Sharpe, 1970] Ken Sharpe. Robustness of normal tolerance intervals. *Biometrika*, pages 71–78, 1970.
- [Shrivastava *et al.*, 2015] Nitin Anand Shrivastava, Abbas Khosravi, and Bijaya Ketan Panigrahi. Prediction interval estimation of electricity prices using pso-tuned support vector machines. *IEEE Transactions on Industrial Informatics*, 11(2):322–331, 2015.
- [Srivastava *et al.*, 2014] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014.
- [Steinberger and Leeb, 2016] Lukas Steinberger and Hannes Leeb. Leave-one-out prediction intervals in linear regression models with many variables. *arXiv preprint arXiv:1602.05801*, 2016.
- [Stine, 1985] Robert A Stine. Bootstrap prediction intervals for regression. *Journal of the American Statistical Association*, 80(392):1026–1031, 1985.