

---

# Interpreting Deep Forest through Feature Contribution and MDI Feature Importance

---

Yi-Xiao He, Shen-Huan Lyu, Yuan Jiang  
 National Key Laboratory for Novel Software Technology,  
 Nanjing University, Nanjing, 210023, China.  
 {heyx, lvsh, jiangy}@lamda.nju.edu.cn

## Abstract

Deep forest is a non-differentiable deep model which has achieved impressive empirical success across a wide variety of applications, especially on categorical/symbolic or mixed modeling tasks. Many of the application fields prefer explainable models, such as random forests with feature contributions that can provide local explanation for each prediction, and Mean Decrease Impurity (MDI) that can provide global feature importance. However, deep forest, as a cascade of random forests, possesses interpretability only at the first layer. From the second layer on, many of the tree splits occur on the new features generated by the previous layer, which makes existing explanatory tools for random forests inapplicable. To disclose the impact of the original features in the deep layers, we design a calculation method with an estimation step followed by a calibration step for each layer, and propose our feature contribution and MDI feature importance calculation tools for deep forest. Experimental results on both simulated data and real world data verify the effectiveness of our methods.

**Keywords:** ensemble methods, deep forest, feature importance, interpretability

## 1 Introduction

By suggesting that the key to deep learning may lie in the *layer-by-layer processing, in-model feature transformation and sufficient model complexity*, Zhou and Feng [ZF17] propose the first deep forest model and the gcForest algorithm, which are realized by non-differentiable modules without backward-propagation in training. It consists of a cascade forest structure, each layer contains multiple random forests, and the predictive probability vectors output by the forests are concatenated with the original features, then serve as the input to the next layer [ZF19]. Benefiting from the feature transformation in cascade structure, deep forests (DFs) outperform various classical tree-based algorithms, e.g., classification and regression tree [Bre+84, CART], AdaBoost [SF12], random forest [Bre01, RF], gradient boost decision tree [Fri01, GBDT], extremely random forest [GEW06, ERF] and XGBoost [CG16] in empirical studies. In recent years, deep forests have been widely extended to various real-world applications [Su+19; BFF19; Zha+19] and learning tasks [UR18; UR19; Yan+20; WYL20]. There are also variants aiming at improving performance and reducing computational and memory cost [ZZC19; Pan+22; CLJ21; Ma+22].

Compared with deep neural networks (DNNs), the decision tree model [Loh11], which is the basic component of deep forest, is easier to interpret while improving the prediction performance. In order to make a prediction, trees ask each observation a series of questions, each one being the form:

$$\text{IF: } x^{(j)} \underset{\leq}{\underset{\geq}} s, \text{ THEN: } response, \quad (1)$$

where  $(j, s)$  are to be determined by the splitting algorithm and  $\underset{\leq}{\underset{\geq}}$  represents  $\geq$  or  $<$ . Thus, the prediction for any instance only depends on the sequence of decision rules. However, forests prediction cannot be explained through decision rules

because their prediction is the average of a large number of randomized decision trees [Bre01; GEW06]. But researchers develop approach to compute *feature contribution* for random forest regression and classification models [Kuz+11; Pal+13]. It enables us to explain every single prediction of a random forest through the contribution of each feature. While feature contribution focus on the local interpretation, Mean Decrease Impurity (MDI) [CCS12] offers a global *feature importance* measure over the whole data set. It sums up the total reduction of the splitting criterion brought by that feature as the intrinsic feature importance measure.

Since deep forest is more powerful than random forest, while we enjoy its good predictive performance, we naturally wants to know why it makes certain predictions. However, because of the feature transformation in the cascade forest structure, as Lyu, He, and Zhou [LHZ22] have proved that the new features dominate the forests in the second and following layers, we lose the idea of how the original features function in deep layers. The interpretation tools for random forests are no longer applicable for deep forests, and knowing that feature transformations are important does not help the end user understand why the model makes specific predictions.

To overcome this problem, we develop a two-step calculation process and extend feature contribution and feature importance to deep forest. The estimation step associates the contributions of the new features with the contributions of the original features at the previous layer through the specific training samples in the splitting nodes. The calibration step ensures that our proposed method yields proper feature contribution and feature importance.

Our main contributions are summarized as follows.

- We propose a feature contribution calculation method for deep forest, enabling us to explain its predictions for each instance, i.e., whether the considered feature enlarges or reduces the predicted value for the considered class, and by how much.
- We propose an MDI feature importance calculation method for deep forest, enabling us to tell the overall impact of each feature in building the whole model.
- We analyze the properties that feature contribution and feature importance should have to ensure that our designed methods are proper feature contribution and feature importance.
- Experimental results show that our proposed tools faithfully reflect the influence of each feature on deep forest prediction for both regression and classification problems. Furthermore, since deep forest is more powerful than random forest, our MDI feature importance also exhibits better estimation quality than that of random forest.

The rest of this paper is organized as follows. Section 2 introduces related work. Section 3 briefly describes existing methods for calculating feature contribution and feature importance for random forests and discuss why they are not directly applicable to deep forest. Section 4 analyzes the properties of feature contribution and feature importance to provide guidance for developing our own methods for interpreting deep forests. Section 5 and Section 6 present our proposed explaining tools. Section 7 reports the experimental results and Section 8 concludes the paper.

## 2 Related work

**Deep forest** Since Zhou and Feng [ZF17] propose the first deep forest model, there are mainly three lines of work to study it. A line of work establishes a screening framework to reduce computational cost and memory requirement for deep forest, including confidence screening [Pan+18], feature screening [Pan+22], hash-based method [ZZC19; Ma+22] and stability-based method [CLJ21]. The second line of work extends deep forest algorithms to different learning tasks and real applications. Zhang et al. [Zha+19] use deep forest to achieve outstanding identification performance for financial anti-fraud system. Yang et al. [Yan+20] employ the measure-aware mechanisms to help deep forest solve multi-label problems. Wang, Yang, and Li [WYL20] address weak-label learning by using a label complement procedure in the feature transformation process of deep forest. For metric learning tasks, Utkin and Ryabinin [UR18; UR19] propose a Siamese deep forest as an alternative to the Siamese neural network. Empirical successes have attracted a line of work to the theoretical analysis of deep forest. To start theoretical analysis of deep forests, Lyu, Yang, and Zhou [LYZ19] prove a margin-based generalization bound for the additive cascade forest. For the consistency of deep forests, Arnould, Boyer,

and Scornet [ABS21] prove a tight bound on the excess risk of two-layer centered random trees, which regards the second layer as a variance reducer. Assuming that the raw and new features are separated and independently used in two stages, Lyu, Chen, and Zhou [LCZ22] prove that new features are easy to cause overfitting risk.

Although deep forest has been widely used and studied, very little work has been done to help users understand the specific predictions made by deep forest models. Kim, Jeong, and Ko [KJK20] tried to improve the explainability by simplifying a trained deep random forest model. Although they select the most important paths in the component forests, going through all the selected paths layer by layer is still a big workload, hence the interpretability is limited. In this paper, we aim to explain the predictions made by the original trained deep forest, and the explanation is by directly telling how much the considered feature enlarges or reduces the predicted value for a given class.

**Explaining random forests** Random forest has long been a successful machine learning method, especially for tabular data [GOV22]. A wealth of work has focused on explaining how each input feature functions in the model. To characterize the overall importance of each feature, there are two widely used measures: the Mean Decrease Impurity [CCS12, p. MDI] and the Mean Decrease Accuracy [Bre01, MDA]. MDI sums up the total reduction of splitting criterion caused by each feature respectively. It is known to favor features with many categories [Str+07; Nic11] and may lead to systematic bias in feature selection by incorrectly assigning high importance to irrelevant features [SZ08; NM09; Li+19; ZH21]. On the other hand, MDA measures the importance of a feature by the reduction in the accuracy after randomly permuting the sample values of a given feature. Different permuting choices have been studied by Strobl et al. [Str+08] and Janitza, Celik, and Boulesteix [JCB18]. In empirical studies, Wright, Ziegler, and König [WZK16] show that MDI and MDA can capture interactions between features but are unable to differentiate from the marginal effects. From a theoretical perspective, there are only a few results on the feature importance of tree-based models. Louppe et al. [Lou+13] investigate the theoretical MDI measure (infinite sample regime) when all features are categorical. After that, the result is extended by Sutera et al. [Sut+16] to the context-dependent features case. Some additional feature importance measures, such as split count [Str+07; Bas+18] can also be used. Recently, there are studies focusing on interpreting every single prediction of random forest through feature contribution [Pal+13; Saa14]. Meanwhile, using this tool, a debiased MDI estimate can be obtained [Li+19]. It functions well even when there are a large number of irrelevant features and severe noise that MDA fails because of poor accuracy. Since MDA is applicable to any black box models, it is also applicable to deep forest. However, to our knowledge, there is no existing work to extend feature contribution and MDI feature importance measure to deep forest.

### 3 Preliminaries

In this section, we briefly introduce feature contribution and feature importance, and discuss why they are not directly applicable to deep forest. The key symbols and notations used in this paper are listed in Table 1.

#### 3.1 Feature contribution

Feature contribution is applicable for every single prediction. It enables us to decompose the prediction into the sum of contributions from each feature [Pal+13].

**For decision trees** The prediction of an instance  $\mathbf{x}$  is the average of the training instances in the leaf node it falls in. Let  $l$  denote the depth of the leaf node that  $\mathbf{x}$  falls in. Along the decision path to the leaf node, as  $\mathbf{x}$  goes through node  $t_0, t_1, \dots, t_l$ , the average response of the training instances changes from node to node. As demonstrated in Figure 1, the final prediction can be represented as

$$f_T(\mathbf{x}) = \mu(t_0) + \sum_{0 < i \leq l} \Delta\mu(t_i), \tag{2}$$

where  $\Delta\mu(t_i) = \mu(t_i) - \mu(t_{i-1})$ .  $\Delta\mu(t_i)$  is the change in average response caused by splitting node  $t_i$  from  $t_{i-1}$ . Note that this calculation is applicable for both regression trees and classification trees. For regression,  $f_T(\mathbf{x})$ ,  $\mu(t_0)$  and  $\Delta\mu(t_i)$

Subject	Sign	Description
Setting	$\mathcal{D}$	The training data.
	$n$	The total number of training instances.
	$K$	The number of features.
	$C$	The number of classes.
Tree & Forest	$t$	A tree node.
	$l$	The depth of a leaf node.
	$I(T)$	All the internal nodes in tree $T$ .
	$n(t)$	The number of training instances in node $t$ .
	$s(t)$	The splitting feature of an internal node $t$ .
	$\Delta_{\mathcal{I}}(t)$	The decrease of impurity by splitting node $t$ .
	$\mu_0$	The average response of training data.
	$\mu(t)$	The average response of the training data in node $t$ .
	$\Delta\mu(t_i)$	The change in average response of node $t_i$ and its parent node $t_{i-1}$ .
	$f_F(\mathbf{x})$	The prediction of forest $F$ on $\mathbf{x}$ .
	$f_{F,k}(\mathbf{x})$	The amount that feature $k$ contributes to $F$ 's prediction value on $\mathbf{x}$ .
$\text{MDI}(k, F)$	The MDI feature importance of feature $k$ in forest $F$ .	
Deep Forest	$\hat{y}_i$	The prediction of previous layer forest on $\mathbf{x}_i$ .
	$\hat{\mu}(t)$	The average predictive value by previous layer of the training data in node $t$ , which can be viewed as average response estimated by previous layer's prediction.
	$\Delta\hat{\mu}(t_i)$	The change in average response of node $t_i$ and its parent node $t_{i-1}$ estimated by previous layer's prediction.
	$\Delta\hat{\mu}(t_i, k)$	Estimated average response change caused by feature $k$ .
	$\Delta\tilde{\mu}(t_i, k)$	The calibrated average predictive response change caused by feature $k$ .
	$k'$	A new feature generated by previous layer's prediction.
	$K'$	The number of new features in the second and above layers.
	$g$	The calibration function that maps the estimated feature contribution $(\Delta\hat{\mu}(t_i, k))_{k=1}^K$ to the calibrated feature contribution $(\Delta\tilde{\mu}(t_i, k))_{k=1}^K$ .
	$\tilde{f}_{F',k}(\mathbf{x})$	The calculated contribution of the original feature $k$ for the second layer forest $F'$ .
	$\tilde{f}_L(\mathbf{x})$	The prediction of $\mathbf{x}$ from the last layer of deep forest.
	$\tilde{f}_{L,k}(\mathbf{x})$	The calculated contribution of feature $k$ in the last layer of deep forest.
$\widehat{\text{MDI}}(k, DF)$	The estimated MDI feature importance of feature $k$ in DF.	

Table 1: Key symbols and notations.

are scalars. For classification, they are  $C$ -dimensional vectors, where  $C$  denotes the number of classes. Let  $s(t)$  denote the splitting feature of node  $t$ . Let  $K$  denote the number of input features. If we sum up the split contributions made by the same feature, the prediction of  $\mathbf{x}$  can be written as

$$f_T(\mathbf{x}) = \mu(t_0) + \sum_{k=1}^K f_{T,k}(\mathbf{x}), \quad (3)$$

where

$$f_{T,k}(\mathbf{x}) = \sum_{0 < i \leq l: s(t_{i-1})=k} \Delta\mu(t_i) \quad (4)$$

is the contribution of feature  $k$  in the prediction of tree  $T$  on  $\mathbf{x}$ .

**For random forests** To extend feature contribution from a tree to a forest, we only need to take the average of the trees in the forest. Assuming each tree has the same distribution of training instances, we use  $\mu_0$  to substitute  $\mu(t_0)$ . Therefore,

$$f_F(\mathbf{x}) = \mu_0 + \sum_{k=1}^K f_{F,k}(\mathbf{x}), \quad (5)$$

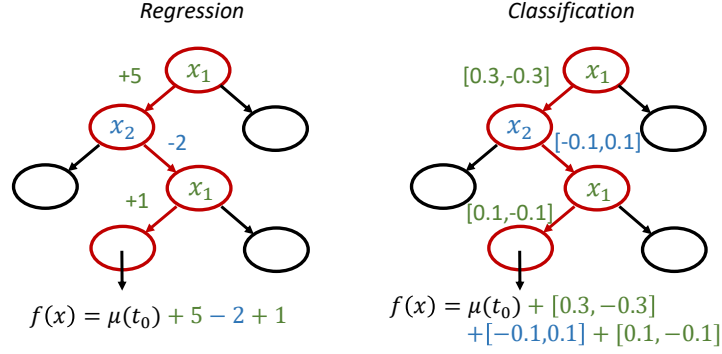


Figure 1: The contribution of each split to the prediction of an instance  $\mathbf{x}$ . Color blue and green correspond to the change of average response by splitting on  $x_1$  or  $x_2$ . Positive and negative values indicate the split enlarges or decreases the predictive value or the probability for the corresponding class.

where

$$f_{F,k}(\mathbf{x}) = \frac{1}{|F|} \sum_{T \in F} f_{T,k}(\mathbf{x}) \quad (6)$$

represents the contribution of feature  $k$  in forest  $F$ 's prediction.

### 3.2 Feature importance

Mean Decrease Impurity (MDI) [CCS12] is an intrinsic feature importance measure for random forests. It measures the overall importance of each feature in building the whole model. For a tree  $T$ , its MDI is calculated by summing the impurity decrease in each internal node weighted by the fraction of training data in the node. Let  $n$  denote the number of training instances,  $I(T)$  denote all the internal nodes in tree  $T$ ,  $n(t)$  denote the number of training instances falling in node  $t$ . The feature importance of feature  $k$  in tree  $T$  is calculated as

$$\text{MDI}(k, T) = \sum_{t \in I(T), s(t)=k} \frac{n(t)}{n} \Delta_{\mathcal{I}}(t), \quad (7)$$

where the decrease impurity of any node  $t$  is defined as

$$\Delta_{\mathcal{I}}(t) \triangleq \text{Impurity}(t) - \frac{n(t^{\text{left}})}{n(t)} \text{Impurity}(t^{\text{left}}) - \frac{n(t^{\text{right}})}{n(t)} \text{Impurity}(t^{\text{right}}), \quad (8)$$

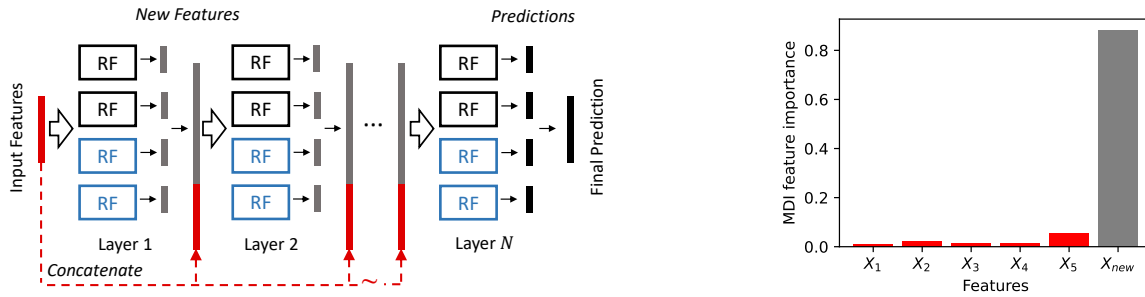
and the impurity of any node  $t$  is defined as

$$\text{Impurity}(t) \triangleq \frac{1}{n(t)} \sum_{i: \mathbf{x}_i \in t} (y_i - \mu(t))^2. \quad (9)$$

Here  $t^{\text{left}}$  and  $t^{\text{right}}$  are two child nodes of node  $t$ ,  $\mu(t)$  is the average response of training instances in node  $t$ . Note that the above expressions assume using the sample variance of labels as the impurity measure for regression problems. For classification problems, Li et al. [Li+19] disclose that using one-hot encoding of the categorical responses, i.e., substituting  $y$  and  $\mu(t)$  with vectors, the impurity expression in Eq. (9) is equivalent to Gini index, a popular impurity measure for classification.

Since the forest is an average of all the individual trees, the feature importance of feature  $k$  in forest  $F$  is also the average

$$\text{MDI}(k, F) = \frac{1}{|F|} \sum_{T \in F} \text{MDI}(k, T). \quad (10)$$



(a) The cascade forest structure of deep forests. Color black and blue indicate different kinds of random forests. The vectors in gray color are the new features generated by individual forests, being input to the next layer.

(b) A typical MDI feature importance result of the second and above layers.  $X_{new}$  is the sum of MDI of new features. The new features tend to have dominating importance.

Figure 2: The new features play an important role in deep forests but are hard to interpret.

### 3.3 Deep forest and the difficulty of interpretation

Figure 2(a) illustrates the cascade forest structure of deep forests. Each layer consists of multiple random forests, with different colors indicating different kinds of random forests. The whole model is built of multiple layers of forests, and the number of layers is automatically determined by the validation accuracy. Note that each layer outputs its prediction vectors (in gray color) to serve as new features for the next layer. That is to say, for the second and deeper layers, the input features are the concatenation of the original features (in red color) and new features. With the layer-by-layer processing, the predictive performance can be improved accordingly.

Existing empirical results and theoretical analyses [ABS21; LCZ22] show that in the second and following layers, the individual trees in random forests will primarily split on the new features, especially in the first several nodes of decision trees. For an arbitrary data set, the feature importance in the second layer forest will very likely behave like Figure 2(b). Apart from the fact that the new features are very important in prediction, it does not help us to understand the prediction made by deep forest because the new feature is the interaction of all the old features.

## 4 Properties of feature importance and feature contribution

To interpret deep forests, we start with analyzing the properties that feature contribution and feature importance should have and their relationship. These serve as guidance for developing interpretation tools for deep forests, ensuring that our designed methods are proper feature importance and feature contribution.

### 4.1 Properties of feature contribution

The term *feature contribution* means decomposing the prediction on a single instance into a sum of contributions from each feature. A proper feature contribution should satisfy the following properties,

- Feature contribution is defined for each single prediction.
- In regression problems, feature contribution is a  $K$ -dimensional vector, whose element  $f_k(\mathbf{x})$  denotes the contribution of feature  $k$  in the final prediction. In classification problems, feature contribution is a matrix of shape  $(K, C)$ , its element  $f_{k,c}(\mathbf{x})$  denoting the contribution of feature  $k$  in the predictive probability of class  $c$ .
- The elements of feature contribution can be positive or negative or zero.

- Given an instance  $\mathbf{x}$ , the sum of bias and feature contribution equals the prediction. That is, for regression,

$$f(\mathbf{x}) = \mu_0 + \sum_{k=1}^K f_k(\mathbf{x}). \quad (11)$$

For classification,

$$f(\mathbf{x})_c = \mu_{0,c} + \sum_{k=1}^K f_{k,c}(\mathbf{x}), 1 \leq c \leq C. \quad (12)$$

## 4.2 Properties of feature importance

The term *feature importance* means the overall impact of each feature in building the whole model. A proper feature importance measure should have the following properties,

- Feature importance is defined over a whole data set, with labels available.
- Feature importance is a  $K$ -dimensional vector, for both regression and classification.
- The elements of feature importance are non-negative.
- The sum of feature importance is no larger than the total response variance of the data set. For regression, the total response variance is  $\frac{1}{n} \sum_i (y_i - \mu_0)^2$ . For classification, the total response variance is defined to be  $\frac{1}{n} \sum_i \sum_{1 \leq c \leq C} (y_{i,c} - \mu_{0,c})^2$ .

Note that here we use variance-based definitions for feature importance. We can easily scale the feature importance to sum to 1 by normalizing with the total response variance.

## 4.3 Relationship between feature contribution and feature importance

Recently, Li et al. [Li+19] disclose that the original expression of MDI as Eq. (7) can be written as

$$\text{MDI}(k, T) = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} f_{T,k}(\mathbf{x}_i) \cdot y_i = \text{Cov}(f_{T,k}(\mathbf{x}_i), y_i). \quad (13)$$

Here  $\mathcal{D}$  is the data set we use for computing MDI feature importance. If  $\mathcal{D}$  is the same as the training set, then Eq. (13) yields the same result as the original MDI expression. Li et al. [Li+19] claim that using out-of-bag data to calculate Eq. (13) can achieve an unbiased result.

Equation (13) offers a new way of computing MDI feature importance in a decision tree  $T$ , i.e., the covariance of feature contribution  $f_{T,k}(\mathbf{x})$  and label  $y$ . Before, we could only compute MDI by summing the decrease of impurity at each node. Equation (13) enables us to compute feature importance by summing over instances, decoupling the computation of feature contribution from the tree training process.

For a forest  $F$ , suppose we use the same data set  $\mathcal{D}$  to calculate MDI for every tree in it, its MDI can be calculated as

$$\text{MDI}(k, F) = \frac{1}{|F|} \sum_{T \in F} \text{MDI}(k, T) = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} y_i f_{F,k}(\mathbf{x}_i). \quad (14)$$

## 5 Feature contribution for deep forest

To compute feature contribution for deep forest, the main difficulty lies in the splits on the new features. We propose to use the training samples in the corresponding node to link the influence of splits back to the last layer forests. However,

the estimated contribution of the original features through last layer's prediction are probably imprecise, and the error will accumulate layer by layer. Therefore, we further propose a calibration step to ensure that the calculation result after each layer maintains a proper feature contribution measure.

## 5.1 The estimation step

Suppose the node  $t_i$  in a second-layer tree  $T'$  uses a new feature  $k'$  to split. We can estimate the average response using the predictions from the previous layer

$$\begin{aligned}\hat{\mu}(t_i) &= \frac{1}{n(t_i)} \sum_{j:\mathbf{x}_j \in t_i} \hat{y}_j \\ &= \frac{1}{n(t_i)} \sum_{j:\mathbf{x}_j \in t_i} \left( \mu_0 + \sum_{k=1}^K f_{F,k}(\mathbf{x}_j) \right) \\ &= \mu_0 + \frac{1}{n(t_i)} \sum_{j:\mathbf{x}_j \in t_i} \left( \sum_{k=1}^K f_{F,k}(\mathbf{x}_j) \right),\end{aligned}$$

where  $F$  is the forest in the previous layer that generates  $k'$  (i.e.,  $\hat{y}$ ), and  $f_{F,k}(\mathbf{x})$  is the amount that feature  $k$  contributes to  $F$ 's prediction value on  $\mathbf{x}$ .

Thus the estimated response change from splitting cell  $t_{i-1}$  into  $t_i$  is

$$\Delta\hat{\mu}(t_i) = \frac{1}{n(t_i)} \sum_{j:\mathbf{x}_j \in t_i} \left( \sum_{k=1}^K f_{F,k}(\mathbf{x}_j) \right) - \frac{1}{n(t_{i-1})} \sum_{j:\mathbf{x}_j \in t_{i-1}} \left( \sum_{k=1}^K f_{F,k}(\mathbf{x}_j) \right). \quad (15)$$

Let

$$\Delta\hat{\mu}(t_i, k) = \frac{1}{n(t_i)} \sum_{j:\mathbf{x}_j \in t_i} f_{F,k}(\mathbf{x}_j) - \frac{1}{n(t_{i-1})} \sum_{j:\mathbf{x}_j \in t_{i-1}} f_{F,k}(\mathbf{x}_j) \quad (16)$$

denote the contribution of feature  $k$  in the estimated change of average response, then

$$\Delta\hat{\mu}(t_i) = \sum_{k=1}^K \Delta\hat{\mu}(t_i, k). \quad (17)$$

Therefore we attribute the estimated response change caused by splitting node  $t_i$  using feature  $k'$  to the contribution of original features.

## 5.2 The calibration step

Meanwhile, using the labels of data, the actual change in average response from node  $t_i$  to  $t_{i+1}$  is calculated as

$$\Delta\mu(t_i) = \mu(t_i) - \mu(t_{i-1}) = \frac{1}{n(t_i)} \sum_{j:\mathbf{x}_j \in t_i} y_j - \frac{1}{n(t_{i-1})} \sum_{j:\mathbf{x}_j \in t_{i-1}} y_j. \quad (18)$$

However,  $\Delta\hat{\mu}(t_i)$  and  $\Delta\mu(t_i)$  are usually unequal, because the predicted response of  $\mathbf{x}_j$  is usually not exactly equal to its true label. And the error might propagate layer by layer. To avoid this, we propose a calibration step after estimation to ensure the computed feature contribution satisfies the property that the sum of bias and feature contribution equals the prediction as stated in Section 4.



The calibration operation can be expressed as a function

$$g : (\Delta\hat{\mu}(t_i, 1), \Delta\hat{\mu}(t_i, 2), \dots, \Delta\hat{\mu}(t_i, K)) \rightarrow (\Delta\tilde{\mu}(t_i, 1), \Delta\tilde{\mu}(t_i, 2), \dots, \Delta\tilde{\mu}(t_i, K)) , \quad (19)$$

such that

$$\sum_{k=1}^K \Delta\tilde{\mu}(t_i, k) = \Delta\mu(t_i) . \quad (20)$$

Thus the prediction of  $f_{T'}(\mathbf{x})$  can be interpreted as the following decomposition,

$$f_{T'}(\mathbf{x}) = \mu_0 + \sum_{k=1}^K \tilde{f}_{T',k}(\mathbf{x}) , \quad (21)$$

where

$$\tilde{f}_{T',k}(\mathbf{x}) = \sum_{s(t_{i-1})=k} \Delta\mu(t_i) + \sum_{s(t_{i-1})=k'} \Delta\tilde{\mu}(t_i, k) \quad (22)$$

is the calibrated contribution of the  $k$ -th original feature.

**Naive multiplicative calibration** A naive instantiation of Eq. (19) is to re-scale the decomposition using a common scale factor,

$$\Delta\tilde{\mu}(t_i, k) = \Delta\hat{\mu}(t_i, k) \frac{\Delta\mu(t_i)}{\Delta\hat{\mu}(t_i)} . \quad (23)$$

Obviously, Eq. (20) holds. However, since the feature contributions can be both positive and negative, the naive re-scaling might lead to numerical problems.

**Naive additive calibration** Using additive calibration can better avoid numerical problems than multiplicative scaling. The naive modification is proportional to the absolute value of its estimated contribution.

$$\Delta\tilde{\mu}(t_i, k) = \Delta\hat{\mu}(t_i, k) + \frac{|\Delta\hat{\mu}(t_i, k)|}{\sum_k |\Delta\hat{\mu}(t_i, k)|} (\Delta\mu(t_i) - \Delta\hat{\mu}(t_i)) . \quad (24)$$

However, if the value of  $(\Delta\mu(t_i) - \Delta\hat{\mu}(t_i))$  is large, features whose estimated contribution is negative will be modified to be positive, and the more negative it was, the more positive it will be, which is obviously not sensible.

**Partial additive calibration** Here we advocate a partial calibration method. We choose a subset of features according to their signs

$$S(t_i) = \{k : \text{sign}(\Delta\hat{\mu}(t_i, k)) = \text{sign}(\Delta\mu(t_i))\} , \quad (25)$$

and we only calibrate this subset of features

$$\Delta\tilde{\mu}(t_i, k) = \Delta\hat{\mu}(t_i, k) \left( 1 + \frac{\Delta\mu(t_i) - \Delta\hat{\mu}(t_i)}{\sum_{k \in S(t_i)} \Delta\hat{\mu}(t_i, k)} \right) , \quad k \in S(t_i) . \quad (26)$$

It is easy to check that Eq. (20) holds. Therefore, the response change brought by splitting on  $k'$  has been decomposed into contributions of the original  $K$  features.

---

**Algorithm 1** Calculate feature contribution for a tree

---

**Require:** Tree node  $t$ **Ensure:** CalculateContribution( $t$ )

```
1: if  $t$  is a root node then
2:    $f_{t,k} \leftarrow 0, k = 1, \dots, K$ 
3: end if
4: for each child node  $t^*$  of  $t$  do
5:   if  $s(t) \in \{1, \dots, K\}$  then
6:     Calculate  $\Delta\mu(t^*, s(t))$  as in Eq. (18)
7:      $f_{t^*,s(t)} \leftarrow f_{t,s(t)} + \Delta\mu(t^*, s(t))$ 
8:   else
9:     Calculate  $(\Delta\tilde{\mu}(t^*, k))_{k=1}^K$  using Eqs. (16) and (19)
10:     $f_{t^*,k} \leftarrow f_{t,k} + \Delta\tilde{\mu}(t^*, k), k = 1, \dots, K$ 
11:   end if
12:   if node  $t^*$  is a leaf node then
13:     return  $(f_{t^*,k})_{k=1}^K$ 
14:   else
15:     CalculateContribution( $t^*$ )
16:   end if
17: end for
```

---

### 5.3 The calculation procedure

Algorithm 1 illustrates the procedure of calculating feature contribution for a tree. When traversing the tree, if the node chooses an original feature to split, we simply add the contribution to this feature; if the node uses a new feature to split, we trace back the contribution to each of the corresponding original features as in Eq. (16) and calibrate as in Eq. (19). The calculation is done by recursively call *CalculateContribution* and get the contributions at all the leaf nodes. To compute feature contribution on a given instance, we simply find the leaf node  $t$  that  $\mathbf{x}$  falls in and get  $\tilde{f}_{T',k}(\mathbf{x}) = f_{t,k}$ .

Based on the analysis of a tree, it is easy to interpret the prediction of a forest

$$f_{F'}(\mathbf{x}) = \mu_0 + \sum_{k=1}^K \tilde{f}_{F',k}(\mathbf{x}), \quad (27)$$

where

$$\tilde{f}_{F',k}(\mathbf{x}) = \frac{1}{|F'|} \sum_{T' \in F'} \tilde{f}_{T',k}(\mathbf{x}). \quad (28)$$

For the above analysis, we assume  $T'$  is the individual tree in the second-layer forest  $F'$ . But it is easy to see that, after the above calculation, we will get the contributions of the  $K$  original features  $f_{F',k}(\mathbf{x})$ . Therefore, from the third layer and beyond, we can repeat the same calculation strategy.

Usually, each layer in a deep forest contains several forests. The final prediction is made by taking the average of all the outputs of the last layer of forests. Let  $L$  denote the set containing forests in the last layer, the final prediction  $f(\mathbf{x})$  made by deep forest can be interpreted as

$$f_L(\mathbf{x}) = \mu_0 + \sum_{k=1}^K \tilde{f}_{L,k}(\mathbf{x}), \quad (29)$$

where

$$\tilde{f}_{L,k}(\mathbf{x}) = \frac{1}{|L|} \sum_{F \in L} \tilde{f}_{F,k}(\mathbf{x}) \quad (30)$$

is contribution of feature  $k$  in the last layer of forests, which is also the feature contribution for the whole deep forest.

**Time complexity.** As shown in Algorithm 1, we only need to traverse each tree once and do simple computation concerning the feature contributions of the training instances falling in each node. The depth of a tree is approximately  $O(\log n)$ . For nodes that have the same depth, all the training instances are processed once. Therefore, the time complexity of the computation for a tree is  $O(n \log n)$ . Let  $M_1$  denote the number of trees in each layer,  $M_2$  denote the maximum number of layers, the time complexity for computing feature contribution for deep forest is  $O(M_1 M_2 n \log n)$ .

**Space complexity.** When computing feature contribution for a tree, we need to maintain  $[f_{t,k}]_{k=1}^K$  through the traversal of a tree and store  $[f_{t,k}]_{k=1}^K$  for every leaf node.  $f_{t,k}$  is a  $C$ -dimensional vector for classification problems (set  $C = 1$  for regression problems), and the number of leaf node is at most  $n$ . Therefore, the space complexity is  $O(nKC)$ .

While computing feature contribution for a forest, we only need to keep the average of tree contributions on the training instances instead of the nodes of all the trees. Therefore the space complexity is still  $O(nKC)$ .

While computing feature contribution layer by layer through the cascade forest structure, we only need to keep the feature contribution of the previous layer. Let  $|L|$  denote the number of forests in each layer, the space complexity is  $O(nKC|L|)$ .

## 6 Feature importance for deep forests

After obtaining feature contribution for deep forest, according to Eq. (14), we can compute MDI feature contribution for deep forest (DF) as

$$\widehat{\text{MDI}}(k, DF) = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \tilde{f}_{L,k}(\mathbf{x}_i) \cdot y_i, \quad (31)$$

where  $\tilde{f}_{L,k}$  is defined in Eq. (30). We then show that it is a proper feature importance measure.

**Proposition 1.** *The sum of the estimated MDI feature importance as Eq. (31) over all the original features equals the sum of the last layer MDI directly over the  $K + K'$  features ( $K'$  denoting the number of new features), i.e.,*

$$\sum_{k=1}^K \widehat{\text{MDI}}(k, DF) = \frac{1}{|L|} \sum_{F \in L} \sum_{k=1}^{K+K'} \text{MDI}(k, F). \quad (32)$$

*Proof.* For a last-layer forest  $F$ ,

$$\begin{aligned} \sum_{k=1}^K \widehat{\text{MDI}}(k, F) &= \sum_{k=1}^K \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \tilde{f}_{F,k}(\mathbf{x}_i) \cdot y_i \\ &= \sum_{k=1}^K \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \left( \frac{1}{|F|} \sum_{T \in F} \tilde{f}_{T,k}(\mathbf{x}_i) \right) \cdot y_i \\ &= \sum_{k=1}^K \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \left( \frac{1}{|F|} \sum_{T \in F} \left( \sum_{j \in T: s(t_{j-1})=k} \Delta\mu(t_j) + \sum_{k'=K+1}^{K+K'} \sum_{j \in T: s(t_{j-1})=k'} \Delta\tilde{\mu}(t_j, k) \right) \right) \cdot y_i \\ &= \sum_{k=1}^K \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \left( \frac{1}{|F|} \sum_{T \in F} \sum_{j \in T: s(t_{j-1})=k} \Delta\mu(t_j) \right) \cdot y_i + \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \left( \frac{1}{|F|} \sum_{T \in F} \sum_{k'=K+1}^{K+K'} \sum_{j \in T: s(t_{j-1})=k'} \Delta\mu(t_j) \right) \cdot y_i \\ &= \sum_{k=1}^K \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \left( \frac{1}{|F|} \sum_{T \in F} f_{T,k}(\mathbf{x}_i) \right) \cdot y_i + \sum_{k'=K+1}^{K+K'} \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \left( \frac{1}{|F|} \sum_{T \in F} f_{T,k'}(\mathbf{x}_i) \right) \cdot y_i \\ &= \sum_{k=1}^K \text{MDI}(k, F) + \sum_{k'=K+1}^{K+K'} \text{MDI}(k', F) \end{aligned}$$

$$= \sum_{k=1}^{K+K'} \text{MDI}(k, F). \quad (33)$$

Taking the averaging over all forests in the last layer,

$$\sum_{k=1}^K \widehat{\text{MDI}}(k, DF) = \frac{1}{|L|} \sum_{F \in L} \sum_{k=1}^K \widehat{\text{MDI}}(k, F) = \frac{1}{|L|} \sum_{F \in L} \sum_{k=1}^{K+K'} \text{MDI}(k, F).$$

□

Proposition 1 shows that the proposed method is a way of disassembling the importance of the new feature to the original features, thus meeting the properties suggested in Section 4 that a proper MDI feature importance should have.

## 7 Experiments

We first show that our computation methods yield reasonable results for feature contribution and feature importance of deep forest in Section 7.1 and Section 7.2. We generate simulated data sets with clear mechanisms behind them, and we conduct experiments on both regression and classification cases. Then in Section 7.3, we compare the quality of our deep forest MDI to other feature importance measures based on its ability of identifying relevant features, showing that as deep forest is more powerful than random forest, it also has better estimation of feature importance. In Section 7.4, we further compare the quality of our deep forest MDI method equipped with different calibration methods, showing that partial additive calibration is the most suitable choice. In order to facilitate the understanding of possible applications in real-world tasks, in Section 7.5, a real-world bike sharing data set is taken as an example to show the calculation results of feature contribution and feature importance.

### 7.1 Illustration of feature contribution for deep forest

In this section, we illustrate our computation of feature contribution for deep forest. We generate synthetic data sets for regression and classification respectively. With the synthetic data sets, we are able to check whether the calculated feature contributions match the underlying data generating process. We first report the change in deep forest’s test error layer by layer. Then visualize the feature contributions in the first layer and last layer respectively.

In this experiment, each layer of deep forest has 4 forests, and each forest contains 50 trees, with their maximum depth fixed to 5. The number of cascade layers of deep forest is determined automatically by the performance on the validation set.

**Regression** 2000 training samples and 2500 test samples are generated using function

$$f(\mathbf{x}) = \sin 2\pi x_1 + \cos 2\pi x_2 \quad (34)$$

and Figure 3(a) is a heatmap of it. The decreasing validation error in Figure 3(b) shows improvement of performance layer by layer. Figure 3(c) and 3(d) decompose the predictions in the first layer and the last layer to the contribution plots of two input features respectively. It is easy to observe that the first layer’s prediction lacks details in the peaks and valley while last layer’s prediction is much closer to the data generating function. We can also observe corresponding refinements of feature contributions. Note that we can see feature interaction in deep layers. In Figure 3(c) the change of feature contribution only along the considered feature. But in Figure 3(d), we can observe the influence of the other feature. Though this may not honestly recovered the data generating process, it helps deep forest boost performance.

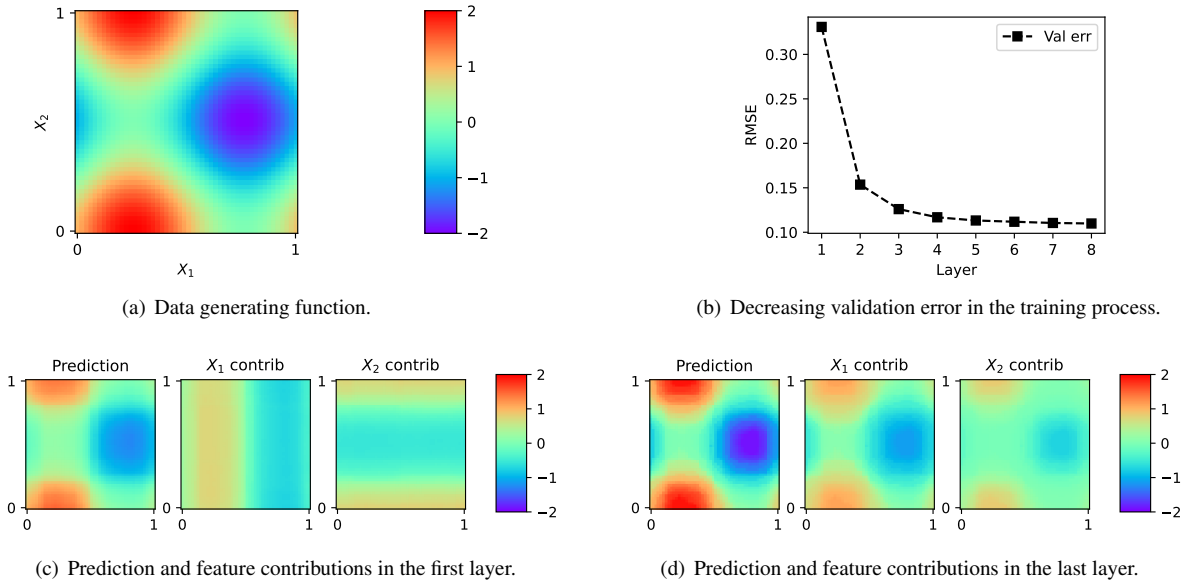


Figure 3: Feature contributions for the regression problem. As shown in (b) and the prediction figures in (c) (d), the predictive performance improves from the first layer to the last layer. The feature contributions show that the model captures more details in both input features.

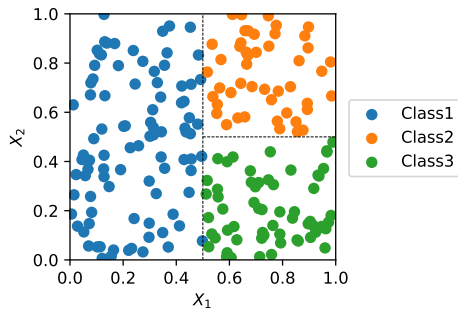
$\mathbf{x}$	$\mu_0$	$\tilde{f}_{L,1}(\mathbf{x})$	$\tilde{f}_{L,2}(\mathbf{x})$	$f_L(\mathbf{x})$
(0, 1)	[ <b>0.46</b> , 0.24, 0.30]	[ <b>0.37</b> , -0.14, -0.16]	[ <b>0.00</b> , 0.00, -0.06]	[ <b>0.88</b> , 0.08, 0.04]
(1, 1)	[0.46, <b>0.24</b> , 0.30]	[-0.36, <b>0.24</b> , 0.03]	[0.00, <b>0.28</b> , -0.21]	[0.05, <b>0.88</b> , 0.07]

Table 2: Example of feature contributions of two sample points for the 3-class classification problem. Only values related to the two relevant dimensions are presented. The values associated with the predicted class are shown in bold.

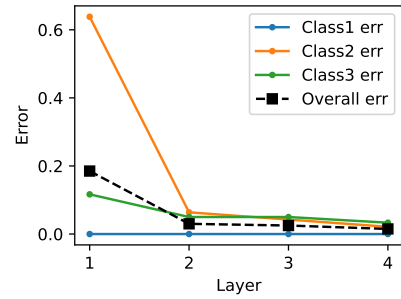
**Classification** We generate a 3-class classification problem. Figure 4(a) shows the training data in the first two dimensions. We add another 100 dimensions of irrelevant features uniformly valued between 0 and 1 to simulate background noise. There are 200 training samples and 2500 test samples. With Figure 4(b) confirming us that the cascade forest structure improves performance layer by layer, Figure 4(c) and 4(d) visualize the feature contribution in the first layer and the last layer to unveil what extra information the model has learned in the cascade layers. We plot for three classes separately, using the corresponding color with different darkness to indicate predictive probabilities. Then there are two contribution plots for  $X_1$  and  $X_2$  respectively. The colored areas indicate the corresponding feature enlarges the probabilities of data in these areas belonging to the corresponding class, and the gray areas versa.

We can see that the first layer’s feature contribution is vague and of light color, while the last layer feature contribution is of darker color, leading to clearer peak probabilities in the prediction. The refinement of feature contribution matches the improvement of performance, and also better captures the underlying data generating scheme. We can also observe that deep forest learns feature interaction in deep layers. Different from the simulated regression data, there actually is feature interaction in generating the classification data. However, in the first layer, the change of feature contribution is mainly along the considered feature. But in the last layer, the influence of the other feature emerges in the plots, showing that the cascade structure enables deep forest to learn details on the two relevant features.

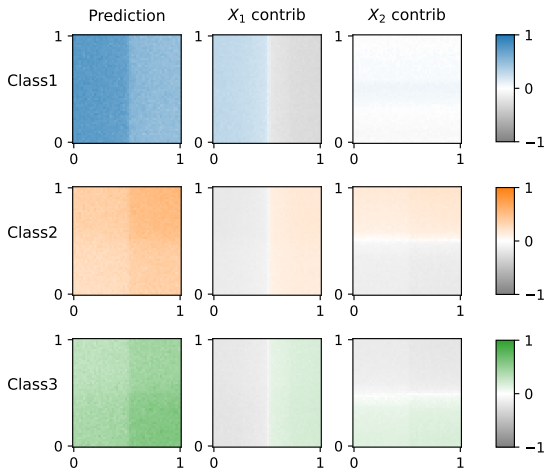
Given the last layer’s feature contribution, we can easily tell how does deep forest make prediction. For example, the point (0, 1) has the highest predicted probability of belonging to Class 1, and the contribution to the Class 1’s probability is mainly due to  $X_1$ . The point (1, 1) has the highest predicted probability of belonging to Class 2, and the contribution



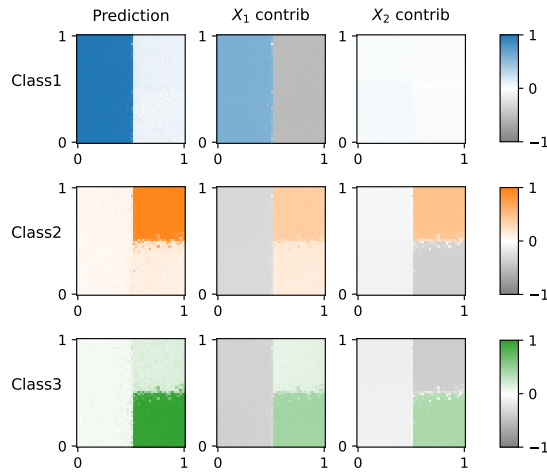
(a) Training data shown in first two dimensions.



(b) Decreasing validation error in the training process.



(c) Prediction and feature contributions in the first layer.



(d) Prediction and feature contributions in the last layer.

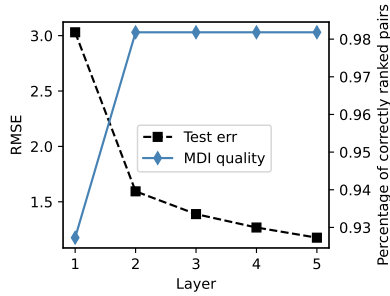
Figure 4: Feature contributions for the 3-class classification problem. As shown in (b) and the prediction figures in (c) (d), the predictive performance improves from the first layer to the last layer. The learned feature contributions are coarse and ambiguous in (c), while clear and precise in (d). The colored areas in the contribution plots indicate an increase in the probabilities of data in these areas belonging to the corresponding class in the areas, and the gray areas indicate a decrease in probability.

comes both from  $X_1$  and  $X_2$ . Table 2 shows the detailed values of decomposing the predicted probabilities of these two points into feature contributions. Only the contributions of the first two features are reported, and the other 100 irrelevant features also contribute to the prediction, but their contributions are small.

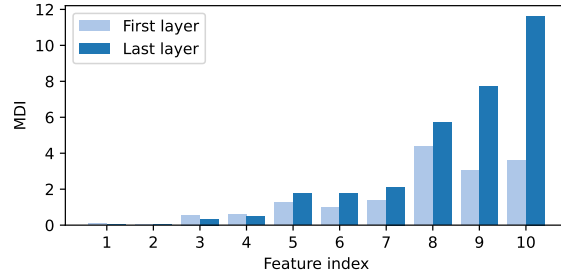
## 7.2 Illustration of feature importance for deep forest

In this section, we show the computation of feature importance for deep forest. We design synthetic regression and classification data sets so that we can tell the relative importance of each feature. The hyper-parameter settings remain the same as in the previous section.

**Regression** we generate a regression problem with different coefficients for each feature and no interaction between features so that we can easily check whether the feature importance is reasonable. We generate 1000 training and test

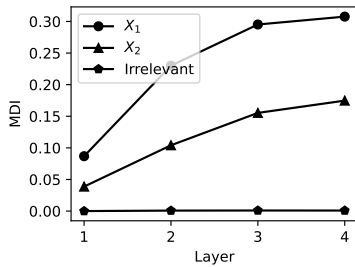


(a) Decreasing test error and increasing MDI quality measured by the percentage of correctly ranked pairs.

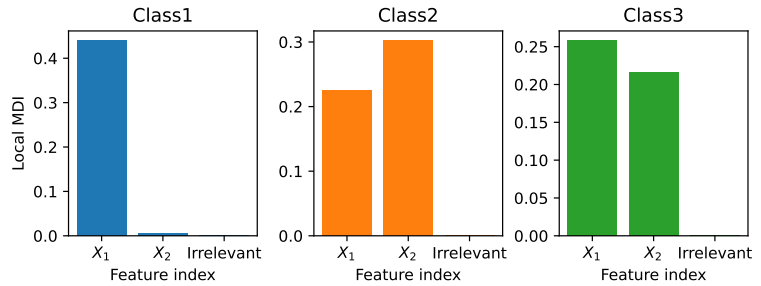


(b) Comparison of MDI of the first layer and the last layer in deep forest. The MDI in the last layer has an obvious refinement over the result in the first layer.

Figure 5: With the growing of layers in deep forest, the estimated MDI catches the underlying importance of features more accurately.



(a) MDI of the two relevant features and the average MDI of the irrelevant features with increasing layers.



(b) Local MDI in deep forest model with respect to each class.  $X_1$  and  $X_2$  play different roles in different classes. The average MDI for irrelevant features is always zero.

Figure 6: Overall MDI and local MDI for the 3-class problem.

samples each according to the data generation function

$$f(\mathbf{x}) = \sum_{k=1}^K kx_k. \quad (35)$$

We measure the MDI quality using the percentage of correctly ranked pairs by relative estimated importance. Figure 5(a) shows that as the test error decreases layer by layer, the ranking quality of relative MDI feature importance increases. As we would expect, deep forest learns more and improves performance through the cascade structure, so MDI would be more accurate in the deep layers. Figure 5(b) directly compares the estimated MDI in the first layer and the last layer. We can see a clear refinement of the estimated MDI towards the underlying data generating mechanism.

**Classification** We turn back to the 3-class classification problem as in Figure 4. Figure 6(a) shows the change of global MDI with respect to layers. We can see that

- The global MDI of  $X_1$  and  $X_2$  both increase layer by layer, indicating deep forest gradually mines more information from the relevant features.
- $X_1$  has a higher MDI than  $X_2$ , which matches the data generating process, since using  $X_1$  alone can separate all the instances from Class 1 which amounts to half of the data distribution.
- The average MDI of the 100 irrelevant features is always close to zero, indicating that the model has a good ability to distinguish relevant features from irrelevant features.

	MDI(RF)	MDI-oob(RF)	MDA(RF)	MDA(DF)	MDI(DF)
<i>sim</i>	0.14	0.80	0.70	0.72	<b>0.82</b>
<i>vehicle</i>	0.95	0.92	0.68	0.64	<b>0.99</b>
<i>segment</i>	0.86	0.93	0.65	0.69	<b>0.95</b>
<i>phishing</i>	0.58	0.96	0.70	0.68	<b>0.97</b>
<i>pendigits</i>	<b>1.0</b>	<b>1.0</b>	0.98	0.99	<b>1.0</b>
<i>satimage</i>	<b>1.0</b>	<b>1.0</b>	0.60	0.59	<b>1.0</b>

Table 3: Average AUC scores of 20 runs for relevant feature identification. The best result of each data set is shown in bold.

Figure 6(b) shows the local MDI calculated for each class. Since equation (13) enables us to compute feature importance by summing over instances, we can calculate local feature importance for each class as

$$\widehat{\text{MDI}}_c(k, DF) = \frac{1}{|\{i : y_i = c\}|} \sum_{y_i=c} \tilde{f}_{L,k}(\mathbf{x}_i) \cdot [\mathbb{I}(y_i = 1), \mathbb{I}(y_i = 2), \mathbb{I}(y_i = 3)], \quad c = 1, 2, 3,$$

From Figure 6(b) we can observe that, for Class 1, only feature  $X_1$  matters, and feature  $X_2$  has very low feature importance, while for Class 2 and Class 3, both  $X_1$  and  $X_2$  play very important roles. The above observation clearly matches the data generating mechanism.

### 7.3 Comparison of feature importance ranking quality

In this section, we compare our MDI feature importance for deep forest, named **MDI(DF)**, to the following methods.

- **MDI(RF)**: the mean decrease of impurity by splitting on the given feature during the training process of random forest [Bre+84]. But it tends to overestimate the feature importance of irrelevant features [Li+19].
- **MDI-oob(RF)**: a debiased version of MDI using out-of-bag samples for random forest [Li+19].
- **MDA(RF)**: the mean decrease in accuracy of a trained random forest caused by randomly permuting the values of the given feature.
- **MDA(DF)**: the mean decrease in accuracy of a trained deep forest caused by randomly permuting the values of the given feature.

In each layer of deep forest, there are still 4 forests, each with 50 trees in it. Accordingly, we grow 200 trees in random forest. Since we are dealing with more complicated data sets, the depth of trees in deep forest is fixed to 8, and the trees in random forests are fully grown.

The simulation data set *sim* is generated according to [Li+19]. It has 50 features, with the  $j^{\text{th}}$  feature taking random value from  $0, 1, \dots, j$  with equal probability. We randomly choose a set  $S$  of 5 features from the first ten features as relevant features. The labels are generated according to  $P(Y = 1 | X) = \text{Logistic} \left( \frac{2}{5} \sum_{j \in S} X_j / j - 1 \right)$ . We generate 1000 i.i.d. training samples and 1000 i.i.d. validation samples. The validation samples are only used for MDA based methods.

The five benchmark data sets are processed with two steps. First, we copy a data set’s feature matrix but randomly permute values of each feature, and then concatenate them to the original feature matrix. By this way, the feature set is now half relevant and half irrelevant. Second, since identifying irrelevant features is a relatively easier task compared with prediction, we reduce the number of training samples to avoid the situation where all the methods are equally perfect. We reduced the number of training samples of data set *vehicle*, *segment* and *phishing* to 20%, for data set *phishing* and *pendigits* we reduce to 10%. We also keep a separate validation set for MDA based methods, whose number of samples is the same as the training set.



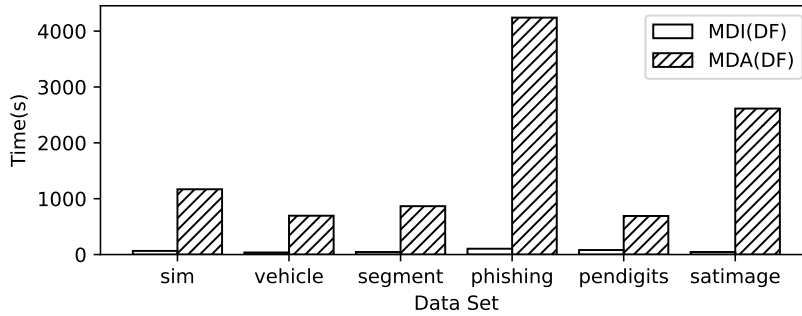


Figure 7: Running time comparison of our MDI method to that of MDA for deep forest. Each experiment is repeated 20 times and the average running time is reported.

	MDI(DF)*	MDI(DF) <sup>+</sup>	MDI(DF)
<i>sim</i>	0.80	<b>0.82</b>	<b>0.82</b>
<i>vehicle</i>	0.84	0.98	<b>0.99</b>
<i>segment</i>	0.93	0.94	<b>0.95</b>
<i>phishing</i>	0.96	0.96	<b>0.97</b>
<i>pendigits</i>	0.98	<b>1.0</b>	<b>1.0</b>
<i>satimage</i>	0.93	<b>1.0</b>	<b>1.0</b>

Table 4: Comparison of different calibration methods based on average AUC scores for relevant feature identification. The best result of each data set is shown in bold.

Table 3 reports the average AUC of identifying relevant features. We can find that MDI(DF) is better than all the RF based methods, namely MDI(RF), MDI-oob(RF), and MDA(RF). Meanwhile, it is also much better than MDA(DF). This is because these generated and manipulated data sets have low accuracy, which hinders the performance of accuracy based method. Furthermore, MDI(DF) is not only better than MDA(DF) in the AUC value of identifying related features, but also requires less running time as shown in Figure 7. The reported running time includes model training and feature importance computation. MDI(DF) can compute feature contribution and feature importance during training process, which lead to only mild increasing of training time. In contrast, MDA(DF) needs to call the test process repeatedly for each feature, and usually for multiple times in order to get a reliable estimation of the accuracy decrease.

#### 7.4 Comparison of different calibration methods

To evaluate the effectiveness of using partial additive calibration as Eq. (26) in the step of calculating feature contribution, we compare MDI(DF) to that using naive multiplicative calibration as Eq. (23), denoted by **MDI(DF)**<sup>\*</sup>, and that using naive multiplicative calibration as Eq. (24), denoted by **MDI(DF)**<sup>+</sup>. The data sets and other settings remain the same as the previous section. And we still use feature importance ranking quality as the evaluation criterion.

Table 4 shows that MDI(DF), which uses partial additive calibration, achieves best AUC in all the data sets. The performance of MDI(DF)<sup>+</sup>, which uses naive additive calibration, follows closely behind. The comparison results justifies the our design of calibration method.

#### 7.5 Application on bike sharing task

This section does not aim to compare, instead it provides an example with a real-world bike sharing task [FG13], showing how our deep forest interpretation tools can be used in an application.

date		Year	isWorkingDay	isClearDay	Temperature	Humidity	WindSpeed
2011/12/7	feature value	2011	1	0	17°C	97%	18 km/h
	feature contribution	-714	24	-104	-1479	-244	-136
2012/9/21	feature value	2012	1	1	25°C	67%	10 km/h
	feature contribution	1031	44	252	1117	145	74

Table 5: Example of feature contribution of deep forest in the bike sharing task. Positive and negative contributions are based on the training label mean.

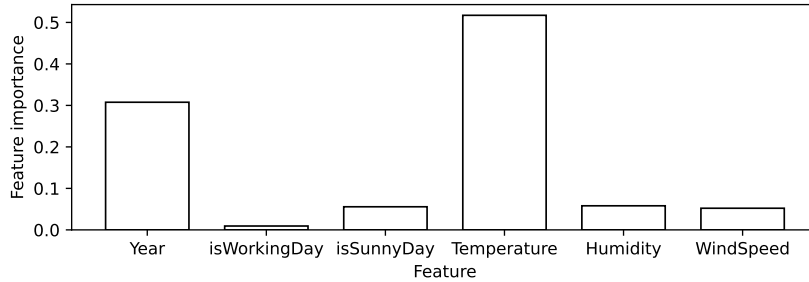


Figure 8: MDI feature importance of deep forest in the bike sharing task. The sum of feature importance is normalized.

The bike sharing data set contains 731 days of records. For clearer demonstration, we process the data set and only keep six features. This is a regression task with a single prediction value, so the feature contributions are scalars. The data are randomly split into training and test data sets in a 2:1 ratio.

The explanatory results of each prediction help the end user understand how the deep forest model works, enabling more reliable deployment of model and helping to find out why a prediction is unexpected. For example, table 5 shows the feature contribution of a trained deep forest on two days in the test set. We can see the platform doing better in 2012 than it did in 2011. In addition, the low temperature and rainy and windy weather on 2011/12/7 both contributed to a decrease in bike rental volumes. Mild temperature and nice weather have led to an increase in bike rental volumes on 2012/9/21. Figure 8 shows the MDI feature importance of deep forest. We can see that temperature is the most important feature. In addition, the year is also an important feature, which shows that the platform has improved a lot in two years. Other weather conditions also play a role, but whether it is a working day has little impact on bike rental volumes.

## 8 Conclusion

In this paper, we propose two interpretation tools for deep forests, namely feature contribution and feature importance. The former decomposes every single prediction of deep forest into the contributions of each original feature. The latter characterizes the overall impact of a feature in the whole model. Experimental results show that they can not only faithfully reflect the influence of each feature on deep forest prediction, but also have a better characterization of feature importance than random forest. We believe that the interpreting tools we provide can further promote the application of deep forests, as well as deepen our understanding of the data and model.

## References

- [ABS21] Ludovic Arnould, Claire Boyer, and Erwan Scornet. “Analyzing the tree-layer structure of deep forests”. In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. 2021, pp. 342–350 (Cited on pages 2, 6).
- [Bas+18] Sumanta Basu et al. “Iterative random forests to discover predictive and stable high-order interactions”. In: *Proceedings of the National Academy of Sciences* 115.8 (2018), pp. 1943–1948 (Cited on page 3).
- [BFF19] Yaakoub Boualleg, Mohamed Farah, and Imed Riadh Farah. “Remote sensing scene classification using convolutional features and deep forest classifier”. In: *IEEE Geoscience and Remote Sensing Letters* 16.12 (2019), pp. 1944–1948 (Cited on page 1).
- [Bre+84] Leo Breiman et al. *Classification and Regression Trees*. Boca Raton, FL: Chapman and Hall/CRC, 1984 (Cited on pages 1, 16).
- [Bre01] Leo Breiman. “Random forests”. In: *Machine Learning* 45.1 (2001), pp. 5–32 (Cited on pages 1–3).
- [CCS12] Adele Cutler, D. Richard Cutler, and John R. Stevens. “Random forests”. In: *Ensemble Machine Learning: Methods and Applications*. 2012, pp. 157–175 (Cited on pages 2, 3, 5).
- [CG16] Tianqi Chen and Carlos Guestrin. “XGBoost: A scalable tree boosting system”. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2016, pp. 785–794 (Cited on page 1).
- [CLJ21] Yi-He Chen, Shen-Huan Lyu, and Yuan Jiang. “Improving deep forest by exploiting high-order interactions”. In: *Proceedings of the 21st IEEE International Conference on Data Mining*. 2021, pp. 1036–1041 (Cited on pages 1, 2).
- [FG13] Hadi Fanaee-T and Joao Gama. “Event labeling combining ensemble detectors and background knowledge”. In: *Progress in Artificial Intelligence* (2013), pp. 1–15 (Cited on page 17).
- [Fri01] Jerome H. Friedman. “Greedy function approximation: A gradient boosting machine”. In: *The Annals of Statistics* 29.5 (2001), pp. 1189–1232 (Cited on page 1).
- [GEW06] Pierre Geurts, Damien Ernst, and Louis Wehenkel. “Extremely randomized trees”. In: *Machine Learning* 63.1 (2006), pp. 3–42 (Cited on pages 1, 2).
- [GOV22] Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. “Why do tree-based models still outperform deep learning on tabular data?” In: *arXiv preprint arXiv:2207.08815* (2022) (Cited on page 3).
- [JCB18] Silke Janitzka, Ender Celik, and Anne-Laure Boulesteix. “A computationally fast variable importance test for random forests for high-dimensional data”. In: *Advances in Data Analysis and Classification* 12.4 (2018), pp. 885–915 (Cited on page 3).
- [KJK20] Sangwon Kim, Mira Jeong, and Byoung Chul Ko. “Interpretation and Simplification of Deep Forest”. In: *arXiv preprint arXiv:2001.04721* (2020) (Cited on page 3).
- [Kuz+11] Victor E Kuz'min et al. “Interpretation of QSAR models based on random forest methods”. In: *Molecular informatics* 30.6-7 (2011), pp. 593–603 (Cited on page 2).
- [LCZ22] Shen-Huan Lyu, Yi-He Chen, and Zhi-Hua Zhou. “A region-based analysis for the feature concatenation in deep forests”. In: *Chinese Journal of Electronics* 31.6 (2022), pp. 1072–1080 (Cited on pages 3, 6).
- [LHZ22] Shen-Huan Lyu, Yi-Xiao He, and Zhi-Hua Zhou. “Depth is more powerful than width with prediction concatenation in deep forests”. In: *Advances in Neural Information Processing Systems* 35. 2022, pp. 29719–29732 (Cited on page 2).
- [Li+19] Xiao Li et al. “A debiased MDI feature importance measure for random forests”. In: *Advances in Neural Information Processing Systems* 32 (2019) (Cited on pages 3, 5, 7, 16).
- [Loh11] Wei-Yin Loh. “Classification and regression trees”. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 1.1 (2011), pp. 14–23 (Cited on page 1).
- [Lou+13] Gilles Louppe et al. “Understanding variable importances in forests of randomized trees”. In: *Advances in Neural Information Processing Systems* 26. 2013, pp. 431–439 (Cited on page 3).

- [LYZ19] Shen-Huan Lyu, Liang Yang, and Zhi-Hua Zhou. “A refined margin distribution analysis for forest representation learning”. In: *Advances in Neural Information Processing Systems* 32. 2019, pp. 5530–5540 (Cited on page 2).
- [Ma+22] Pengfei Ma et al. “HW-Forest: Deep forest with hashing screening and window screening”. In: *ACM Transactions on Knowledge Discovery from Data* (2022) (Cited on pages 1, 2).
- [Nic11] Kristin K Nicodemus. “On the stability and ranking of predictors from random forest variable importance measures”. In: *Briefings in Bioinformatics* 12.4 (2011), pp. 369–373 (Cited on page 3).
- [NM09] Kristin K Nicodemus and James D Malley. “Predictor correlation impacts machine learning algorithms: Implications for genomic studies”. In: *Bioinformatics* 25.15 (2009), pp. 1884–1890 (Cited on page 3).
- [Pal+13] Anna Palczewska et al. “Interpreting random forest classification models using a feature contribution method”. In: *Integration of Reusable Systems*. 2013, pp. 193–218 (Cited on pages 2, 3).
- [Pan+18] Ming Pang et al. “Improving deep forest by confidence screening”. In: *Proceeding of the 18th IEEE International Conference on Data Mining*. 2018, pp. 1194–1199 (Cited on page 2).
- [Pan+22] Ming Pang et al. “Improving deep forest by screening”. In: *IEEE Transactions on Knowledge and Data Engineering* 34.9 (2022), pp. 4298–4312 (Cited on pages 1, 2).
- [Saa14] Ando Saabas. *Interpreting random forests*. 2014. URL: <https://blog.datadive.net/interpreting-random-forests/> (Cited on page 3).
- [SF12] Robert E Schapire and Yoav Freund. *Boosting: Foundations and Algorithms*. London: MIT Press, 2012 (Cited on page 1).
- [Str+07] Carolin Strobl et al. “Bias in random forest variable importance measures: Illustrations, sources and a solution”. In: *BMC Bioinformatics* 8.1 (2007), pp. 1–21 (Cited on page 3).
- [Str+08] Carolin Strobl et al. “Conditional variable importance for random forests”. In: *BMC Bioinformatics* 9.1 (2008), pp. 1–11 (Cited on page 3).
- [Su+19] Ran Su et al. “Deep-Resp-Forest: A deep forest model to predict anti-cancer drug response”. In: *Methods* 166 (2019), pp. 91–102 (Cited on page 1).
- [Sut+16] Antonio Sutera et al. “Context-dependent feature analysis with random forests”. In: *Proceedings of the 32nd Conference on Uncertainty in Artificial Intelligence*. 2016, pp. 716–725 (Cited on page 3).
- [SZ08] Carolin Strobl and Achim Zeileis. *Danger: High power exploring the statistical properties of a test for random forest variable importance*. 2008 (Cited on page 3).
- [UR18] Lev V. Utkin and Mikhail A. Ryabinin. “A Siamese deep forest”. In: *Knowledge-Based Systems* 139 (2018), pp. 13–22 (Cited on pages 1, 2).
- [UR19] Lev V. Utkin and Mikhail A. Ryabinin. “Discriminative metric learning with deep forest”. In: *International Journal on Artificial Intelligence Tools* 28.2 (2019), 1950007:1–1950007:19 (Cited on pages 1, 2).
- [WYL20] Qian-Wei Wang, Liang Yang, and Yu-Feng Li. “Learning from weak-label data: A deep forest expedition”. In: *Proceedings of the 34th AAAI Conference on Artificial Intelligence*. Vol. 34. 2020, pp. 6251–6258 (Cited on pages 1, 2).
- [WZK16] Marvin N Wright, Andreas Ziegler, and Inke R König. “Do little interactions get lost in dark random forests?” In: *BMC Bioinformatics* 17.1 (2016), pp. 1–10 (Cited on page 3).
- [Yan+20] Liang Yang et al. “Multi-label learning with deep forest”. In: *Proceedings of the 24th European Conference on Artificial Intelligence*. Vol. 325. 2020, pp. 1634–1641 (Cited on pages 1, 2).
- [ZF17] Zhi-Hua Zhou and Ji Feng. “Deep forest: Towards an alternative to deep neural networks”. In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 2017, pp. 3553–3559 (Cited on pages 1, 2).
- [ZF19] Zhi-Hua Zhou and Ji Feng. “Deep forest”. In: *National Science Review* 6.1 (2019), pp. 74–86 (Cited on page 1).
- [ZH21] Zhengze Zhou and Giles Hooker. “Unbiased measurement of feature importance in tree-based methods”. In: *ACM Transactions on Knowledge Discovery from Data* 15.2 (2021), pp. 1–21 (Cited on page 3).

- [Zha+19] Ya-Lin Zhang et al. “Distributed deep forest and its application to automatic detection of cash-out fraud”. In: *ACM Transactions on Intelligent Systems and Technology* 10.5 (2019), pp. 1–19 (Cited on pages 1, 2).
- [ZZC19] Meng Zhou, Xianhua Zeng, and Aozhu Chen. “Deep forest hashing for image retrieval”. In: *Pattern Recognition* 95 (2019), pp. 114–127 (Cited on pages 1, 2).