Combining SHAP-driven Co-clustering and Shallow Decision Trees to Explain XGBoost

Ruggero G. Pensa[1][0000−0001−5145−3438], Anton Crombach[2,3][0000−0002−2889−5120], Sergio Peignier[4][0000−0002−9004−3033], and Christophe Rigotti[2,3][0000−0003−1265−1479]

[1] University of Turin, Italy
ruggero.pensa@unito.it
[2] Inria Centre de Lyon, France
anton.crombach@inria.fr
[3] INSA Lyon, CNRS, Universite Claude Bernard Lyon 1, LIRIS, UMR5205, F-69621 Villeurbanne, France
christophe.rigotti@insa-lyon.fr
[4] INSA Lyon, INRAE, BF2I, UMR0203, F-69621, Villeurbanne, France
sergio.peignier@insa-lyon.fr

Abstract. Transparency is a non-functional requirement of machine learning that promotes interpretable or easily explainable outcomes. Unfortunately, interpretable classification models, such as linear, rule-based, and decision tree models, are superseded by more accurate but complex learning paradigms, such as deep neural networks and ensemble methods. For tabular data classification, more specifically, models based on gradient-boosted tree ensembles, such as XGBoost, are still competitive compared to deep learning ones, so they are often preferred to the latter. However, they share the same interpretability issues, due to the complexity of the learnt model and, consequently, of the predictions. While the problem of computing local explanations is largely addressed, the problem of extracting global explanations is scarcely investigated. Existing solutions consist of computing some feature importance score, or extracting approximate surrogate trees from the learnt forest, or even using a black-box explainability method. However, those methods either have poor fidelity or their comprehensibility is questionable. In this paper, we propose to fill this gap by leveraging the strong theoretical basis of the SHAP framework in the context of co-clustering and feature selection. As a result, we are able to extract shallow decision trees that explain XGBoost with competitive fidelity and higher comprehensibility compared to two recent state-of-the-art competitors.

Keywords: Explainable AI · SHAP values · Co-clustering.

1 Introduction

Thanks to the advances of deep learning methods, most recent contributions of the machine learning community target complex and multimedia data, where transformer-based architectures, such as large language models (LLMs) and vision transformer models (ViTs) show their potential. Yet, tabular data are crucial in many important and sensitive applications, such as credit scoring, loan granting, medical diagnosis, insurance premium definition, and so on. In such scenarios, "traditional" machine learning models, such as XGBoost [5], based on gradient-boosted decision trees [10], still outperform more sophisticated deep neural network architectures [23], although attention-based methods show improved performance on tabular data [1]. Unfortunately, the outcomes of XGBoost, in terms of both the classification model learnt and the predictions made, lack of transparency, as they are poorly interpretable and explainable. Indeed, the outstanding performances on tabular data are strongly linked to the ability of such method to capture non-linear relationships among the features, which are intrinsically unintelligible for humans. This is a major problem in all those sensitive application areas where the prediction made by a machine has a direct and important impact on the life of human beings, as wrong or faulty decisions could harm their health, freedom and dignity. Additionally, recent regulatory frameworks, such as the GDPR and AI Act adopted in the European Union, explicitly enforce transparency in AI-based decision support systems.

To address the challenge of transparent AI, the involved organizations and companies usually take two types of measures: either they abandon complex models in favor of more interpretable but less accurate ones (such as logistic regression or decision trees), or they resort to explainability methods such as global surrogate models or local explanations. The second option in particular provides a good trade-off between accuracy and transparency and is also encouraged by recent advances in the prosperous field of so-called Explainable AI (XAI) [8]. Indeed, many recent research works have addressed the problems of providing global interpretations of opaque or even black-box models [13,30], and explaining every decision made on individual input instances [19]. Global interpretations are typically provided by measuring and visualizing the importance of features (or groups of them) in the overall model [9,12,15] or by computing an interpretable surrogate model (in general, a linear one or a decision tree) that tries to imitate its behavior and to approximate its predictions [4]. Local explanations, instead, are supplied on individual predictions by computing local surrogates [26], anchors [27], counterfactual explanations [16], or by measuring the contribution of each feature in the specific prediction [32]. Among the latter, the SHAP framework [21] in particular has gained attention thanks to its strong theoretical foundation [25]. In fact, it is derived from a famous contribution to game theory made by Lloyd S. Shapley in 1953 [31]. This makes the SHAP framework particularly adapted to applications that must adhere to strong legal requirements, such as financial and medical ones.

Although SHAP values are very flexible and can be aggregated to provide global explanations, they are not free of criticism [14]. For instance, they are less expressive than decision tree paths in explaining individual predictions. Decision paths, i.e., the sequence of decision tree nodes that are triggered during the prediction of the target variable for an input instance, can be expressed in the form of *if-then-else* rules on every feature participating in the prediction. Thus, compared to SHAP values, they provide more insight on the specific range of feature values that leads to one particular outcome. Consequently, they are well understood by humans and perfectly comply with legal transparency requirements. Of course, decision paths come with their own potential drawbacks. The surrogate decision trees computed on non-linear models can be extremely complex, leading to long decision paths that are less human-readable. Two recent papers [4,29] address this issue using two alternative but effective approaches. The first one [4], is a model-agnostic method that uses microaggregation to compute a collection of micro-clusters on the training data. Then, for each cluster a shallow decision tree is learnt. To provide a local interpretation, the unseen instance is mapped to the closest cluster and processed by the corresponding decision tree. The second approach [29] is tailored on XGBoost and is aimed at inferring an approximate decision tree from the learnt forest. To do that, a pruning strategy is initially performed on the pre-trained ensemble. Successively, a representative set of conjunctions is extracted from the pruned ensemble. Finally, a decision tree of a specified maximum depth is computed leveraging the conjunction set. Compared to [4], the second method results in explanation models with higher fidelity, but at the cost of deeper (thus less comprehensible) decision trees.

To cope with the issues surrounding transparent AI, in this paper we propose a new approach to XGBoost explanation for tabular data, which computes what we call a "companion model" whose purpose is two-fold: i) it provides a global fine-grained interpretation of the original model in terms of a collection of decision trees, each learnt on a subset of features selected according to their SHAP values; ii) it provides explanations in terms of compact decision paths. To achieve the first goal, we first compute the SHAP values for the training data instances. Then we compute a co-clustering of the matrix consisting of the training instances as its rows, of the features as its columns, and containing the corresponding SHAP values. This co-clustering optimizes a statistical associative measure, the Goodman-Kruskal's τ [11], that expresses the strength of the link between a partition of the features and a partition of the data instances over the matrix. Then, for each cluster of instances, we learn a decision tree built on the subset consisting of the most important features for this set of instances. As a result, we obtain a set of shallow decision trees, one per cluster of instances, each characterized by the distribution (over the training set) of SHAP values for each feature. For the second objective (i.e., providing explanations), we compute the SHAP values of the features for a single input instance of interest, and we map the instance to the closest cluster of training instances according to a similarity measure based on τ [2]. Then, for the instance, we compute the decision path using the tree learnt on the cluster this instance has been mapped to.

Our framework can be adopted in applications requiring robust but simple and human-readable explanations. Although it could be potentially applied to any black box model, it is particularly efficient on gradient-boosted decision trees, thanks to the availability of a polynomial-time algorithm for computing SHAP values on tree-based models [20]. We show this through an extensive experimental validation, from which we can conclude that our method achieves high fidelity scores and provides simpler explanations in term of decision path length, compared to the two recent state-of-the-art approaches for explaining XGBoost [29] and black box models [4] mentioned beforehand. The source code of our framework and all scripts allowing for reproducibility, are available online at `https://github.com/rupensa/xccshap_ds2024`.

2 Background

In this section, we introduce the necessary preliminaries and notions required for the full understanding of our framework. We first introduce the SHAP framework [21], based on the computation of Shapley values for explaining individual classification outcomes [33]. Then, we introduce the de-normalized Goodman-Kruskal's τ association measure that we use for computing the strength of the association of groups of features to groups of data instances.

In the remainder of the paper, we will consider a dataset $D = \{d_1, \dots, d_n\}$ of n data instances represented in an m-dimensional feature space. The features are represented by their indices, $F = \{1, \dots, m\}$ being the set of all indices. We consider a classification task over a set of classes $\mathcal{C}$, performed by a model f associating a class $z = f(d)$ to an instance d. We suppose that, for each class h of $\mathcal{C}$, the model can also output $f^h(d)$, i.e., the membership probability of d to class h.

2.1 The SHAP framework

The goal of the SHAP (SHapley Additive exPlanations) framework [21] is to assign a local importance value (hereafter called SHAP value) to each feature of F in the prediction $z = f(d)$ for an instance d. According to [21,33], and when adapted to a classification task, the SHAP value $\phi_v(f^h, d)$ measures the contribution of the feature v to the computation of probability $f^h(d)$.

Given E, a subset of the feature set F, the value of $f^h(d)$ when only the values of features in E are known is denoted by $f^h_E(d)$. As a special case, $f^h_\varnothing(d)$ is the probabiity $f^h(d)$ when no knowledge about the features is available. The *local accuracy* property, of the SHAP values, ensures that

$$f^h(d) = f^h_\varnothing(d) + \sum_{v=1}^{m} \phi_v(f^h, d), \tag{1}$$

Note that each $\phi_v(f^h, d)$ can be either positive or negative, and according to this additive framework, the most important features are those with the largest $|\phi_v(f^h, d)|$.

The way SHAP values $\phi_v(f^h, d)$ are specified follows a seminal result in cooperative game theory [31], which proposes to assign a fair reward to players by measuring their individual contributions (known as Shapley values) to a grand coalition F they participate in. If the players in coalition F are considered to be features, then, according to [21,33], the SHAP values can be obtained as

$$\phi_v(f^h, d) = \sum_{E \subseteq F \setminus \{v\}} w(E) \left(f^h_{E \cup \{v\}}(d) - f^h_E(d) \right) \tag{2}$$

where the function w only depends on m (the overall number of features) and on $|E|$:

$$w(E) = \frac{|E|!(m - |E| - 1)!}{m!} \tag{3}$$

2.2 Co-clustering based on Goodman-Kruskal's τ

According to [11], given two discrete random variables X and Y, the Goodman-Kruskal's τ is defined as the proportional reduction of the error in predicting X when Y is known:

$$\tau_{X|Y} = \frac{e_X - \mathbb{E}[e_{X|Y}]}{e_X} \tag{4}$$

where e_X is the error in predicting X, i.e., the probability that two independent realizations of the random variable differ, and $\mathbb{E}[e_{X|Y}]$ is the expected value of the error in predicting X when Y is known. In more detail, if we define $p(i,j)$ as the probability of observing both $X = x_i$ and $Y = y_j$, $p(i)$ as the probability of $X = x_i$ and $p(j)$ the probability of $Y = y_j$, then Goodman-Kruskal's $\tau_{X|Y}$ can be defined as [11]:

$$\tau_{X|Y} = \frac{\sum_i \sum_j \frac{p(i,j)^2}{p(j)} - \sum_i p(i)^2}{1 - \sum_i p(i)^2} \tag{5}$$

Given a non-negative matrix A, the Goodman-Kruskal's τ can been used to measure the strength of the link between a partition of the rows of A and a partition of the columns of A. Considering the two sets of clusters formed by these two partitions, X is the random variable representing the event that a row u is assigned to cluster i, while Y is the random variable that represents the event that a column v belongs to cluster j. To estimate the probability distributions of X and Y, a contingency table T can be associated to the double partitioning (X, Y) where an element t_{ij} of T is the sum of the elements of A for all rows in cluster i and all columns in cluster j. Consequently, the joint probability that a row is assigned to cluster i and a column is assigned to cluster j is $p(i,j) = t_{ij} / \sum_{i'} \sum_{j'} t_{i'j'}$. The marginal probabilities $p(i)$ and $p(j)$ are then equal to $\sum_{j'} t_{ij'} / \sum_{i'} \sum_{j'} t_{i'j'}$ and $\sum_{i'} t_{i'j} / \sum_{i'} \sum_{j'} t_{i'j'}$, respectively. According to [28], an optimal partitioning of the rows and columns of A is the one that maximizes both $\tau_{X|Y}$ and $\tau_{Y|X}$, as τ is not symmetric. However, as shown by Battaglia *et al.* [2], the direct optimization of $\tau_{X|Y}$ has several disadvantages, including its computational cost. Hence, they propose an alternative measure consisting of a non-normalized version of $\tau_{X|Y}$ and $\tau_{Y|X}$ that they call $\hat{\tau}_{X|Y}$ and

$\hat{\tau}_{Y|X}$, and defined as:

$$\hat{\tau}_{X|Y} = \sum_i \sum_j \frac{p(i,j)^2}{p(j)} - \sum_i p(i)^2 \tag{6}$$

$$\hat{\tau}_{Y|X} = \sum_i \sum_j \frac{p(i,j)^2}{p(i)} - \sum_j p(j)^2 \tag{7}$$

In [2], a fast algorithm is also proposed to find the optimal partitioning of the rows and columns of A, with an *a priori* unspecified number of row and column clusters. This algorithm, PB-τCC, can be used to perform a co-clustering [7,3] of the rows and columns of any non-negative matrix after scaling its elements so that the new values sum to one (the matrix is rescaled by PB-τCC itself).

3 The XCCSHAP framework

In this section, we introduce our framework that provides compact and accurate companion explanations for classification models based on XGBoost trained on tabular data. Before entering into the details of the framework, we first present a use case in which it could be deployed.

We consider a car insurance company that operates through the Web to propose its products to potential customers. To provide a quote of the insurance premium, the online application requires multiple pieces of information that feed one or multiple algorithms contributing to determining the applicant's risk category. A classification model is used to determine the risk class of the applicant, thus leading to the consequent premium. The algorithm has been previously trained on part of the historical tabular data owned by the company, which records both personal information (e.g., age, gender, job position) and risk factors (such as the number of previous accidents, yearly mileage).

Suppose now that a group of potential customers makes a complaint against the company, because of suspected discrimination based on supposed disparate treatment or impact for some protected category of people. If the machine learning model adopted is not explainable (as it is the case for gradient-boosted decision trees), to verify that the actual model is not biased and to be able to show its compliance with all laws and regulations in force, the company could resort to some global explanation model and look for some critical decision paths. Such a "companion" can be even computed before a new learning model is deployed in order to check its fairness, to search for other "strange" patterns, and even to verify the conditions triggering inaccurate or wrong predictions.

As we argued in Section 1, feature importance alone could not provide sufficient insights into the predictions, as it does not consider the range of values at the origin of the prediction. Decision paths, instead, are more effective because not only do they take into account the feature values at each decision node, but they are also more human-readable, as they can be represented as sequences of "if-then-else" rules. Unfortunately, decision trees tend to become rather large and hence less interpretable, when they reproduce complex decision patterns.

To address this issue, we propose to leverage SHAP values to extract a collection of shallow decision trees, each learnt on a subset of samples and a subset of features. In the following, we present the theoretical details of our framework, called XCCSHAP (eXplanations through Co-Clustering of SHAP values).

3.1 Cluster-based companion surrogate model

Let f be a classification model, over a set of classes $\mathcal{C}$, trained using XGBoost [5] on a set of training data instances $D = \{d_1, \ldots, d_n\}$, with feature indices $F = \{1, \ldots, m\}$. Let $d_u \in D$ be the u-th data instance, we consider the SHAP values $\phi_v(f^h, d_u)$ representing the importance of the contribution of feature v to the probability of class h for the instance d_u. To capture the "global" importance of v in computing all class probabilities for d_u, we choose to retain it's highest contribution, that is it's maximum over all classes. So, this importance score is the Maximum Absolute SHAP value (MASHAP) of v for d_u, and is defined as:

$$\text{MASHAP}(v, d_u) = \max_{h \in \mathcal{C}} |\phi_v(f^h, d_u)| \tag{8}$$

Then, a Normalized MASHAP (NMASHAP) is obtained by simply normalizing over the features:

$$\text{NMASHAP}(v, d_u) = \frac{\text{MASHAP}(v, d_u)}{\max_{v' \in F} \text{MASHAP}(v', d_u)}. \tag{9}$$

Let $S = (s_{uv})$ be the non-negative matrix containing the NMASHAP values for every data instance d_u and feature v (i.e., $s_{uv} = \text{NMASHAP}(v, d_u)$). On matrix S we can compute a partition R of the rows as a set of clusters $\{R_1, \ldots, R_k\}$ and a partition $C = \{C_1, \ldots, C_l\}$ of the columns of S. This partitioning process will be detailed later.

Then, we compute the mean NMASHAP value of each matrix block consisting of a cluster of rows and a cluster of columns of S. More formally, we build a $k \times l$ matrix $M = (m_{ij})$, where each element m_{ij} is defined as:

$$m_{ij} = \frac{\sum_{u \in R_i} \sum_{v \in C_j} s_{uv}}{|R_i| \cdot |C_j|} \tag{10}$$

Then, we learn a collection $\mathcal{DT} = \{DT_1, \ldots DT_k\}$ of k decision trees, one tree DT_i for each row cluster $R_i \in R$, as follows. For a cluster R_i, we select a dataset $D^i \subset D$:

$$D^i = \{d_u \in D \text{ s.t. } u \in R_i\} \tag{11}$$

and build a vector Z^i containing the target values of the instances in D^i. For each $d \in D^i$, this target value is simply $f(d)$, i.e., the value predicted by the model we want to explain[1].

[1] Alternatively, the true labels of the training set (if available) could be used as well.

Then, we select a subset F^i of the features F as follows:

$$F^i = \left\{ v \in F \text{ s.t. } \frac{\sum_{u \in R_i} s_{uv}}{|R_i|} >= m_{ij^\star} \wedge \sum_{u \in R_i} s_{uv} > 0 \right\} \tag{12}$$

where $j^\star$ is the index of the cluster containing feature v. The first condition means that the average NMASHAP of v over R_i is not less than this average for all features of the same cluster (i.e., $m_{ij^\star}$). The second condition is needed to avoid retaining feature v when NMASHAP values of v for all instances in R_i are equal to zero. The tree DT_i is learnt using instances in D^i, target values Z^i and features F^i.

In other words, DT_i is trained on a dataset D^i consisting of the data points d_u whose corresponding NMASHAP vectors in S are assigned to cluster R_i, and by a subset F^i of F, where each feature v is such that its average absolute NMASHAP value in cluster R_i is at least equal to the average NMASHAP values in the block of M formed by cluster R_i and the feature cluster v belongs to.

Intuitively, each decision tree is trained only on a subset of data points represented by the most important features w.r.t. the original model f for this subset. So far, we have not given any characterization of the bi-partition (R, C), but it is clear that it influences the overall process and, consequently, it should be chosen accurately. A good bi-partition should provide a strong link between the clusters of rows and the clusters of columns of S. In principle, one could use a clustering algorithm on S to compute a partition R, and on $S^\top$ to obtain a partition C. However, this solution suffers from different drawbacks. First, the two partitions would not be linked by any association as they are computed independently by leveraging all the columns of S or $S^\top$. Second, in most application scenarios, tabular data are such that $m \ll n$, where m is the number of features and n is the number of data points. Consequently, when computing the clustering on $S^\top$, the results could be affected by the curse of dimensionality and be meaningless.

So, a co-clustering algorithm is better suited to obtain an appropriate bi-partition. Co-clustering is a machine learning task that, given an input matrix A, computes a partition on the rows and a partition on the columns of A simultaneously [7]. This is done by optimizing an objective function that takes into account both partitions. The way the overall optimization algorithm searches for the optimal co-clustering can be roughly viewed as iteratively and alternately executing clustering on rows while taking advantage of dimensionality reduction on columns and vice-versa. There exist many co-clustering algorithms based on different approaches and optimization criteria [3], however, for many of them it is required to supply the number of desired clusters on rows and columns as input parameter. Additionally, some categories of approaches search for block diagonal co-clustering solutions, but this is too strong a constraint for our purposes, since the number of row and column clusters might be different.

For this reason, we adopt the co-clustering algorithm PB-τCC proposed in [2] (see Section 2.2), which optimizes the non-normalized version of the Goodman-Kruskal's τ and, additionally, only requires an arbitrary large upper bound of the final number of clusters as input parameter. Thus, k (resp. l) the number of clusters in partition R (resp. C) do not need to be provided, but are determined by the algorithm itself.

Algorithm 1: XCCSHAP(D, F, f)

```
Input: A dataset $D$ with set of feature indices $F$, a classification model $f$
       trained with XGBoost
Result: A matrix $S$ of NMASHAP values, a partitioning $R$ (resp. $C$) of the
        rows (resp. columns) of $S$, a collection $\mathcal{DT}$ of decision trees
1  // Compute the NMASHAP matrix
2  foreach $d_u \in D$ do
3      foreach $v \in F$ do
4          MASHAP$(v, d_u) \leftarrow \max_{h \in \mathcal{C}} |\phi_v(f^h, d_u)|$;
5      end foreach
6      foreach $v \in F$ do
7          $s_{uv} \leftarrow \frac{\text{MASHAP}(v,d_u)}{\max_{v' \in F} \text{MASHAP}(v',d_u)}$;
8      end foreach
9  end foreach
10 $(R, C) \leftarrow$ PB-$\tau$CC$(S)$;            // Compute partitions $R$ and $C$ using [2]
11 $\mathcal{DT} \leftarrow \{\varnothing\}$;
12 // Build datasets $D_i$ using $(R, C)$
13 foreach $R_i \in R$ do
14     $D^i = \{d_u \in D \text{ s.t. } u \in R_i\}$;
15     $F^i = \left\{v \in F \text{ s.t. } \frac{\sum_{u \in R_i} s_{uv}}{|R_i|} >= m_{ij^\star} \wedge \sum_{u \in R_i} s_{uv} > 0\right\}$; // see Eq. (12)
16     $Z^i \leftarrow f(D^i)$;    // Vector of classes predicted for instances in $D^i$
17     $DT_i \leftarrow$ decision tree trained on instances $D^i$, features $F^i$, target values $Z^i$;
18     $\mathcal{DT} \leftarrow \mathcal{DT} \cup \{DT_i\}$;
19 end foreach
20 return $S$, $R$, $C$, $\mathcal{DT}$
```

The whole procedure described above is formally presented in Algorithm 1. The algorithm has two critical parts. The first one is the rather high complexity of computing the SHAP values in the general case. Fortunately, for tree-based ensemble models (such as the one computed by XGBoost), many enhanced and parallel algorithms to improve the computational speed exists, mainly based on TreeSHAP [20]. Among the others, GPUTreeSHAP [24] uses GPU computational facilities, FastTreeSHAP [34] adopts caching mechanisms, and Linear TreeSHAP [35] exploits the properties of polynomials. We use FastTreeSHAP in our experiments.

The second critical part is the computation of the partition (R, C). In this case, the computational cost is not an issue, as the algorithm adopted has good scalability capabilities [2]. However, as most clustering and co-clustering algorithms, the results depends on the initialization. To address this issue, a practical solution is to execute the co-clustering algorithm multiple times and select the best solution, i.e., the one maximizing $\hat{\tau}_{X|Y}$. This is exactly the solution adopted in our experiments, where the number of executions is set to 30.

3.2 Explaining individual predictions using XCCSHAP

Although our framework provides a global interpretation of the model as a set of decision trees, it can be also used to provide a human-readable explanation of individual predictions (i.e., local explanation). In fact, once the companion model has been computed, one can exploit it to explain individual predictions made by model f for a instance d_t. This is done simply by associating d_t to a cluster R_i. Then, statistics about the NMASHAP values over cluster R_i provide a first level of explanation (e.g., boxplot SHAP values of each feature). Of course, XCCSHAP goes beyond such feature scoring by providing the decision path computed for d_t in the related decision tree DT_i. Associating R_i to d_t is trivial when d_t belongs to the set of instances D used to compute R, but for any unseen test data instance $d_t \notin D$, its cluster assignment must first be determined. To this purpose, we adapt the strategy used in [2] for measuring the similarity between a data instance and the cluster prototypes build during the co-clustering process.

In our case, we need to assign the data instance d_t to the closest row cluster in R by also taking into account the partitioning C of the columns of S. To do that, we first compute the vector s^t representing the NMASHAP values of the features of d_t (see Eq. 9). Each component s^t_v is computed as follows:

$$s^t_v \leftarrow \text{NMASHAP}(v, d_t). \tag{13}$$

Then, for each cluster R_i we measure the proximity of s^t to the cluster prototype r^i, where this prototype is obtained as follows, accordingly to [2]. Each j-th component of the l-sized prototype vector r^i is computed as:

$$r^i_j = \frac{\sum_{u \in R_i} \sum_{v \in C_j} s_{uv}}{\sum_{u=1}^{n} \sum_{v \in F} s_{uv}} \tag{14}$$

We can exploit the similarity function defined also in [2] to compute the proximity between s^t and each prototype r^i:

$$\sigma\left(s^t, r^i\right) = \sum_{j=1}^{l} \frac{\sum_{v \in C_j} s^t_v}{\sum_{w=1}^{k} r^w_j} r^i_j - \left(\sum_{v \in F} s^t_v\right) \cdot \left(\sum_{j=1}^{l} r^i_j\right) \tag{15}$$

Now, d_t can be assigned the cluster index $i^t \in \{1, \ldots, k\}$ that satisfies the following condition:

$$i^t = \underset{i=1,\ldots,k}{\arg\max} \big(\sigma\left(s^t, r^i\right)\big) \tag{16}$$

Finally, to explain the prediction made by f on d_t, one can use the decision tree $DT_{i^t} \in \mathcal{DT}$ to compute the decision path as the sequence of split nodes traversed by d_t. However, one must be aware that the outcome of the prediction of $f(d_t)$ may be different from the one of $DT_{i^t}(d_t)$. Hence, our local explanation only makes sense when $DT_{i^t}(d_t) = f(d_t)$. If it is not the case, the method returns by default the raw SHAP values themselves as local explanation.

4 Experiments

In this section we present and discuss the results of our experiments aimed at showing the effectiveness of XCCSHAP in providing good explainability performances in terms of both fidelity w.r.t. the original machine learning model and compactness of the explanations provided. We assess the behavior of the framework in a classification task on a large number of datasets. We use XGBoost [5] as learning model, and its implementation available online[2].

Table 1. Pre-processed dataset statistics.

Dataset	**Target variable**	**#classes**	**#rows**	**#cols**	**Accuracy**
Adult	income	4	47621	108	0.5949
AIDS	cid	2	2139	23	0.8863
Bank Marketing	y	2	30907	44	0.8847
Breast Cancer	Class	2	277	39	0.7143
Breast Cancer W	Diagnosis	2	569	30	0.9649
Car Evaluation	class	4	1728	21	0.9750
Voting Records	Class	2	232	32	0.9714
Credit Approval	A16	2	653	46	0.8418
Glass Identification	Type_of_glass	6	214	9	0.6923
Glioma	Grade	2	839	26	0.8651
HCV	Category	5	589	13	0.9209
Heart Disease	num	5	297	13	0.5667
Heart Failure	death_event	2	299	12	0.7889
Image Segmentation	class	7	210	19	0.8571
Ionosphere	Class	2	351	34	0.9528
Students' Dropout	Target	3	4424	36	0.7681
Spambase	Class	2	4601	57	0.9566
SUPPORT2	hospdead	2	753	62	0.8673
Wine	class	3	178	13	0.9630
Yeast	localization_site	10	1484	8	0.6009

The data are downloaded from the UCI Machine Learning Repository [17][3]. Columns with more than 50% of missing values are removed, and after this step, the rows with at least one missing value are filtered out. Categorical attributes are converted into numerical one by applying one-hot encoding. This is partly due to the poor support for non-numeric attributes in most machine learning models, partly to the non-trivial processing they would require within our framework. However, we plan to investigate methods to handle them directly as future work. The statistics of the pre-processed datasets are given in Table 1.

After splitting the data into training (70% of the data) and test set (the remaining 30%), when the number of training data instances is less than 1 000, we execute a grid search on the space of the hyper-parameters, otherwise we use

[2] `https://xgboost.ai/`
[3] `https://archive.ics.uci.edu/datasets`

the approach described in [18]. In both cases, to reduce biases, we use a five-fold cross validation. Then we retrain the classifier on the whole training set using the hyper-parameters found. The accuracy on the test set is reported in Table 1

Once the model is trained, we apply XCCSHAP on the training set and measure two performance indicators: the **fidelity** of the learnt explanations and their **comprehensibility** [25]. To measure the fidelity in the classification task, we compute the accuracy of the target variables predicted by XCCSHAP on the test set w.r.t. those predicted by XGBoost. The other indicator, the comprehensibility, is measured by computing the average decision path length of the predictions on the test set. Additionally, we also compute the number of clusters of the explanation model (i.e., the number of shallow surrogate decision trees).

As competitors, we consider MaSDT (Microaggregation-based Shallow Decision Trees), an approach computing shallow decision trees on micro-aggregated data [4], and XGBTA (XGBoost Tree Approximator), a method that approximates a forest of trees with a single decision tree built by combining pruning and conjunction set extraction [29]. The former is a black-box explanation method that, as such, is model-agnostic. The latter is specifically designed to explain ensemble models trained with XGBoost and, consequently, should outperform the former in terms of fidelity. Our expectation is that XCCSHAP fidelity lies in-between the two competitors' ones, but with much shorter decision paths, thanks to the feature selection step performed by the algorithm. Since the maximum depth of the surrogate shallow decision trees is a hyperparameter for all methods, it is determined by performing a grid search with 5-fold cross-validation in the set $\{2, 3, 4, 5, 10, 100\}$ using the accuracy as scoring function. Notice that, since XGBTA approximates the forest learnt by XGBoost on the true labels, the grid search has been performed by considering the true labels of the training set for all competitors, coherently with the experimental setting used in [29].

MaSDT also requires an extra parameter called representativeness, i.e., the minimum number of data instances per cluster. Similarly as done in [4], we consider values of representativeness varying between 0.1% and 30% of the training set and retain the setting leading to the highest fidelity. The data and the source code required to reproduce all the experiments are available online[4].

In Table 2, we report the results, including fidelity, decision path length, and also, for MaSDT and XCCSHAP, the number of shallow trees (XGBTA builds a single tree). The average rank of all competing algorithms for the fidelity and the average decision path length is also reported. Lower values of the average rank for an algorithm means that, on average, it ranks better than the other algorithms for the given performance indicator. With the exception of dataset Credit Approval, XCCSHAP always obtains the highest or the second highest fidelity scores. In more detail, XCCSHAP achieves the highest fidelity score 11 times, XGBTA wins 8 times, and MaSDT 5 times. However, XGBTA and MaSDT obtain the lowest score on 6 and 11 datasets, respectively. This observation is confirmed by the average ranks (last row of Table 2). As further analysis, we conduct a Friedman statistical test followed by a Nemenyi post-hoc

[4] `https://github.com/rupensa/xccshap_ds2024`

Table 2. Number of surrogate trees, fidelity and average decision path length. The best results are highlighted in gray. The second best results are in lighter gray.

	# trees		**Fidelity**			**Avg. Dec. path length**		
Dataset	XCCSHAP	MaSDT	XCCSHAP	XGBTA	MaSDT	XCCSHAP	XGBTA	MaSDT
Adult	10	3	0.9387	0.9603	0.9373	3.3±0.7	10.0±0.1	5.0±0.1
AIDS	2	6	0.9315	0.9299	0.9393	2.4±0.5	10.0±0.3	2.5±0.8
Bank Marketing	5	3	0.9214	0.8919	0.9232	5.9±2.9	10.0±0.1	3.7±1.4
Breast Cancer	3	5	0.9048	0.7738	0.881	2.4±0.5	5.0±0.0	2.7±1.1
Breast Cancer W	3	3	0.9708	0.9825	0.9532	1.9±0.3	9.3±1.1	2.2±0.8
Car Evaluation	5	3	0.9634	0.9788	0.9403	2.0±2.5	9.9±0.3	3.9±3.3
Voting Records	2	10	1	1	1	1.0±0.0	2.0±0.0	0.7±0.7
Credit Approval	3	4	0.9133	0.9439	0.9235	2.0±0.0	5.0±0.0	2.7±0.5
Glass Identification	3	6	0.7538	0.7692	0.7231	4.2±1.5	8.3±1.6	2.1±0.8
Glioma	5	6	0.9683	0.9563	0.9683	2.0±0.6	5.0±0.0	2.3±0.6
HCV	3	20	0.9774	0.9379	0.9379	2.4±0.8	4.0±0.0	0.5±0.9
Heart Disease	3	20	0.6222	0.7111	0.6	4.4±1.8	5.0±0.1	1.7±0.9
Heart Failure	3	10	0.8667	0.8556	0.8667	2.0±1.2	5.0±0.0	2.1±0.6
Image Segmentation	7	4	0.9524	0.873	0.7619	0.9±0.9	7.9±1.6	2.3±0.9
Ionosphere	4	20	0.9906	0.8868	0.9717	1.5±0.6	5.0±0.0	1.0±0.1
Students' Dropout	5	3	0.9021	0.8938	0.872	2.6±1.0	9.9±0.3	3.2±0.4
Spambase	5	3	0.9544	0.9471	0.9276	3.0±1.2	10.0±0.3	6.6±2.7
SUPPORT2	2	3	0.9292	0.9159	0.885	2.0±0.0	5.0±0.1	2.4±0.8
Wine	3	10	0.963	0.9815	0.9074	1.0±0.1	5.0±0.0	0.8±0.7
Yeast	6	3	0.8049	0.7668	0.7646	2.5±0.5	9.8±0.7	4.9±1.9
Avg. rank (the lower the better)			1.50	1.90	2.30	1.35	3.00	1.65

test [6] to assess whether the differences among the three method are statistically significant. The null hypothesis of the Friedman test is that the Friedman statistics is similar to the critical value of the χ^2 distribution with $k-1$ degrees of freedom (k being the number of methods compared). When this is true, it means that all the methods obtain similar performances. For the fidelity scores, the Friedman statistics is $Q = 7.479$, while the critical value of the χ^2 distribution at significance level $\alpha = 0.05$ is 5.991. Hence, the null hypothesis that the differences among the three fidelity scores are not statistically significant can be rejected. We can then proceed with the Nemenyi post-hoc test, to verify whether the differences among every pairs of competitors are significant. The test is passed when the difference of the average ranks of two competitors is above the critical difference at some significance level α. The critical difference at $\alpha = 0.05$ is $CD = 0.74$, consequently, although the average rank is in favor of XCCSHAP, the difference in performance is only statistically significant when compared to MaSDT. Incidentally, the differences between XGBTA and MaSDT are not significant.

When we look at the average path length computed by the three algorithms, we observe a clear predominance of XCCSHAP and MaSDT, confirmed by the average ranks and statistical tests. In this case, even when considering a lower significance level ($\alpha = 0.005$), the null hypothesis of the Friedman test can be rejected (the critical value of the χ^2 distribution is 10.597, while the Friedman statistics is $Q = 30.90$). Moreover, according to the Nemenyi test, both XCCSHAP and MaSDT perform significantly better than XGBTA (the critical difference is $CD = 0.92$ at significance level $\alpha = 0.01$). The difference between XCCSHAP and MaSDT, instead, does not pass the Nemenyi test at any significance level. However, in most cases, the number of trees generated by XCCSHAP is smaller than that of MaSDT: the average number of trees is, respectively, 4.1 for XCCSHAP and 7.25 for MaSDT. In this case we use the Wilcoxon rank-sum test (also known as the Mann–Whitney U test [22]), a non-parametric statistical test to verify the null hypothesis that two samples come from the same population. We use it because it does not assume that the populations are drawn from a normal distribution. In our case, the null hypothesis can be rejected with observed significance level $p < 0.1$.

For the sake of completeness, we add a few more words on the computational time of the three methods on a representative dataset (Adult). On a server with 32 Intel Xeon Skylake cores running at 2.1 GHz and 256 GB RAM, MaSDT is the fastest approach, as it takes from 2 seconds to 3 minutes to compute the model, depending on the representativeness. XGBTA is by far the slowest competitor, with more than 53 hours of running time. Finally, XCCSHAP takes 17 minutes, almost all devoted to the computation of the SHAP values and the co-clustering.

In conclusion, the experiments confirm that our method is competitive in terms of fidelity and better in terms of comprehensibility, when compared to XGBTA and MaSDT. More precisely, the decision trees produced by XGBTA must be more complex to reproduce XGBoost predictions with high fidelity. MaSTD, as a method for explaining black-box models, has good comprehensibility performances, but at the cost of a diminished fidelity.

5 Conclusion

We have presented XCCSHAP, a global explanation method that provides compact and accurate interpretations of XGBoost predictions. Given the trained model and the training data, it exploits SHAP values to extract a co-clustering of data instances and features. Then, it computes one shallow decision tree per cluster of instances using an associated subset of features. Each cluster is thus characterized by its SHAP value distribution and corresponding decision tree. XCCSHAP is applicable beyond the training data, by mapping new unseen data instances to the closest SHAP-based cluster, predictions on these new data can then be explained by interrogating the corresponding shallow decision tree. Through extensive experiments on 20 real-world datasets, we have showed that our approach is competitive with two state-of-the-art methods in terms of fidelity. Moreover, it outperforms them in terms of comprehensibility, measured as the average decision path of the explanations. We plan to further investigate the limitation of the approach, related to the slow computation of the SHAP values for very large datasets. To address this issue, we will try several options, such as using alternative optimized versions of TreeSHAP (e.g., [24,35]) or adopting a sampling strategy on the training set. Other directions of research include the generalization of the method to tackle regression tasks and to be model-agnostic.

Acknowledgments. For this work, R.G. Pensa has received a grant from Inria, under its visiting scholar program. This work was also partially funded by BQR INSA Lyon 2023 Neurinfo, ANR project C2R-IA ANR-22-CE56-0005, PEPR Santé Numérique project 22-PESN-0002, Fondation ARC grant ARCPJA22020060002212, Institut National du Cancer grant PLBIO22-071, and the National Institutes of Health (NIH) award R01DC020478. R.G. Pensa is member of Gruppo Nazionale Calcolo Scientifico – Istituto Nazionale di Alta Matematica (GNCS-INdAM). C. Rigotti is a member of the LabEx IMU (ANR-10-LABX-0088) of Université de Lyon.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Arik, S.Ö., Pfister, T.: Tabnet: Attentive interpretable tabular learning. In: Proceedings of AAAI/IAAI/EAAI 2021. pp. 6679–6687 (2021)
2. Battaglia, E., Peiretti, F., Pensa, R.G.: Fast parameterless prototype-based co-clustering. Mach. Learn. **113**(4), 2153–2181 (2024)
3. Biernacki, C., Jacques, J., Keribin, C.: A survey on model-based co-clustering: High dimension and estimation challenges. J. Classif. **40**(2), 332–381 (2023)
4. Blanco-Justicia, A., Domingo-Ferrer, J., Martínez, S., Sánchez, D.: Machine learning explainability via microaggregation and shallow decision trees. Knowl. Based Syst. **194**, 105532 (2020)
5. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of ACM SIGKDD 2016. pp. 785–794 (2016)
6. Demsar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1–30 (2006)
7. Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: Proceedings of ACM SIGKDD 2001. pp. 269–274 (2001)
8. Dwivedi, R., Dave, D., Naik, H., Singhal, S., Rana, O.F., Patel, P., Qian, B., Wen, Z., Shah, T., Morgan, G., Ranjan, R.: Explainable AI (XAI): core ideas, techniques, and solutions. ACM Comput. Surv. **55**(9), 194:1–194:33 (2023)
9. Fisher, A., Rudin, C., Dominici, F.: All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. J. Mach. Learn. Res. **20**, 177:1–177:81 (2019)
10. Friedman, J.H.: Greedy function approximation: A gradient boosting machine. The Annals of Statistics **29**(5), 1189 – 1232 (2001)
11. Goodman, L.A., Kruskal, W.H.: Measures of association for cross classification. J. Am. Stat. Assoc. **49**, 732–764 (1954)
12. Greenwell, B.M., Boehmke, B.C.: Variable importance plots - an introduction to the vip package. R J. **12**(1), 343 (2020)
13. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM Comput. Surv. **51**(5), 93:1–93:42 (2019)
14. Huang, X., Marques-Silva, J.: On the failings of shapley values for explainability. Int. J. Approx. Reason. p. 109112 (2024)
15. Inglis, A., Parnell, A., Hurley, C.B.: Visualizing variable importance and variable interaction effects in machine learning models. J. Comput. Graph. Stat. **31**(3), 766–778 (2022)

16. Keane, M.T., Kenny, E.M., Delaney, E., Smyth, B.: If only we had better counterfactual explanations: Five key deficits to rectify in the evaluation of counterfactual XAI techniques. In: Proceedings of IJCAI 2021. pp. 4466–4474 (2021)
17. Kelly, M., Longjohn, R., Nottingham, K.: The UCI Machine Learning Repository, `https://archive.ics.uci.edu`
18. Li, L., Jamieson, K.G., DeSalvo, G., Rostamizadeh, A., Talwalkar, A.: Hyperband: A novel bandit-based approach to hyperparameter optimization. J. Mach. Learn. Res. **18**, 185:1–185:52 (2017)
19. Liang, Y., Li, S., Yan, C., Li, M., Jiang, C.: Explaining the black-box model: A survey of local interpretation methods for deep neural networks. Neurocomputing **419**, 168–182 (2021)
20. Lundberg, S.M., Erion, G.G., Chen, H., DeGrave, A.J., Prutkin, J.M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., Lee, S.: From local explanations to global understanding with explainable AI for trees. Nat. Mach. Intell. **2**(1), 56–67 (2020)
21. Lundberg, S.M., Lee, S.: A unified approach to interpreting model predictions. In: Proceedings of NIPS 2017. pp. 4765–4774 (2017)
22. Mann, H.B., Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other. Ann. Math. Stat. **18**(1), 50–60 (1947)
23. McElfresh, D.C., Khandagale, S., Valverde, J., C., V.P., Ramakrishnan, G., Goldblum, M., White, C.: When do neural nets outperform boosted trees on tabular data? In: Proceedings of NeurIPS 2023 (2023)
24. Mitchell, R., Frank, E., Holmes, G.: Gputreeshap: massively parallel exact calculation of SHAP scores for tree ensembles. PeerJ Comput. Sci. **8**, e880 (2022)
25. Molnar, C.: Interpretable Machine Learning. Self-published, 2 edn. (2022), `https://christophm.github.io/interpretable-ml-book`
26. Ribeiro, M.T., Singh, S., Guestrin, C.: "why should I trust you?": Explaining the predictions of any classifier. In: Proceedings of ACM SIGKDD 2016. pp. 1135–1144 (2016)
27. Ribeiro, M.T., Singh, S., Guestrin, C.: Anchors: High-precision model-agnostic explanations. In: Proceedings of AAAI/IAAI/EAAI 2018. pp. 1527–1535 (2018)
28. Robardet, C., Feschet, F.: Comparison of three objective functions for conceptual clustering. In: Proceedings of PKDD 2001. pp. 399–410 (2001)
29. Sagi, O., Rokach, L.: Approximating xgboost with an interpretable decision tree. Inf. Sci. **572**, 522–542 (2021)
30. Saleem, R., Yuan, B., Kurugollu, F., Anjum, A., Liu, L.: Explaining deep neural networks: A survey on the global interpretation methods. Neurocomputing **513**, 165–180 (2022)
31. Shapley, L.S.: A value for n-person games. In: Contributions to the Theory of Games II, pp. 307–317. Princeton University Press, Princeton (1953)
32. Strumbelj, E., Kononenko, I.: An efficient explanation of individual classifications using game theory. J. Mach. Learn. Res. **11**, 1–18 (2010)
33. Strumbelj, E., Kononenko, I.: Explaining prediction models and individual predictions with feature contributions. Knowl. Inf. Syst. **41**(3), 647–665 (2014)
34. Yang, J.: Fast TreeSHAP: Accelerating SHAP value computation for trees. In: Proceedings of NeurIPS Workshop on XAI approaches for debugging and diagnosis (2021)
35. Yu, P., Bifet, A., Read, J., Xu, C.: Linear tree shap. In: Proceedings of NeurIPS (2022)