open proceedings

GAM Forest Explanation

Claudio Lucchese
claudio.lucchese@unive.com
Ca' Foscari University of Venice
Italy

Salvatore Orlando
orlando@unive.com
Ca' Foscari University of Venice
Italy

Raffaele Perego
raffaele.perego@isti.cnr.it
ISTI-CNR
Italy

Alberto Veneri
alberto.veneri@unive.it
Ca' Foscari University of Venice
ISTI-CNR
Italy

ABSTRACT

Most accurate machine learning models unfortunately produce black-box predictions, for which it is impossible to grasp the internal logic that leads to a specific decision. Unfolding the logic of such black-box models is of increasing importance, especially when they are used in sensitive decision-making processes. In this work we focus on forests of decision trees, which may include hundreds to thousands of decision trees to produce accurate predictions. Such complexity raises the need of developing explanations for the predictions generated by large forests. We propose a post hoc explanation method of large forests, named GAM-based Explanation of Forests (GEF), which builds a Generalized Additive Model (GAM) able to explain, both locally and globally, the impact on the predictions of a limited set of features and feature interactions. We evaluate GEF over both synthetic and real-world datasets and show that GEF can create a GAM model with high fidelity by analyzing the given forest only and without using any further information, not even the initial training dataset.

1 INTRODUCTION

EXplainable AI (XAI) research aims at answering the ineludible need for AI systems of being trustworthy, fair, and understandable [13]. In this work we focus on the understandability of complex AI models. Among the models that are most effective, we limit our interest to forests of decision trees such as Random Forests (RFs) [3] or Gradient Boosted Decision Trees (GBDTs) [8]. These are very accurate in several application scenarios [29], but their large size (up to thousands of decision trees) makes them a black box that is impossible to be interpreted by a human [2]. A common approach to "open the black box" is to conduct a *post hoc explainability* analysis to provide a *surrogate* model, i.e., an understandable model that can be used as an *explainer* of the complex model.

The general scenario for a black-box explanation procedure is the following: a black-box model, trained by some learning algorithm, is given to an explainer algorithm that is able to extract an explanation starting from the characteristics of the model under investigation usually employing also the training dataset. In our case, we assume that the dataset on which the forest was trained is not available anymore, e.g., for privacy concerns between different parties or even different departments of the same company. In other words, we allow the explainer to be a third party, such as a certification authority, which has access to the forest only. We assume that the forest is fully known to this third party, meaning not only the query API but also its structure, its test nodes, and its leaves.

Explanation strategies struggle between *global* explanations that provide a surrogate having a behavior similar to that of the black-box model across a given training dataset, and *local* explanations whose scope is limited to a small subspace around a specific instance. We aim at reconciling these two contrasting efforts by generating a GAM [15] as an explanation of a forest. GAMs are known to be highly explainable models [2]. In fact, a GAM is a piecewise function that can globally approximate the given forest, and where each "piece" can locally explain the behavior of the forest in a limited subspace. We investigate different strategies to train an effective GAM explainer by exploiting the information encompassed within a given forest of decision trees. Since in our scenario we assume to not have access to the original training set, the explanation GAM is trained on a synthetic dataset generated from information elicited from the analysis of the given forest. We propose feature selection and data sampling methods for generating such a dataset to make it possible to efficiently train a GAM explainer that exploits both univariate and bi-variate components. We call this framework GAM-based Explanation of Forests (GEF) and the general schema is illustrated in Fig. 1. We highlight that the GEF framework encompasses two steps: *i)* the generation of the synthetic dataset $\mathcal{D}^*$ on the basis of the forest at hand to be explained, and *ii)* the creation of an explainable GAM-based model Γ. Even though the two steps are normally investigated in different tasks, in this scenario they are strongly intertwined since we do not have access to the original dataset and it is necessary to accurately generate a synthetic version of it to train a GAM as an explanation of the initial forest. For both these two steps we propose and evaluate different strategies.

In summary, the contributions of this work are as follows:

- we propose GAM-based Explanation of Forests (GEF), a post hoc explanation method that provides a GAM as an explanation of a given forest of decision trees;
- GEF provides explanations without using the dataset on which the forest to be explained was trained, thus creating a new way of explaining a model when, for privacy or legal reason, we cannot access data used to train it;
- we propose and evaluate several heuristic strategies for identifying the most important features and feature interactions, as well as for sampling the feature space on the basis of the information available at the test nodes of the given forest.

© 2023 Copyright held by the owner/author(s). Published in Proceedings of the 26th International Conference on Extending Database Technology (EDBT), 28th March-31st March, 2023, ISBN 978-3-89318-088-2 on OpenProceedings.org.
Distribution of this paper is permitted under the terms of the Creative Commons license CC-by-nc-nd 4.0.

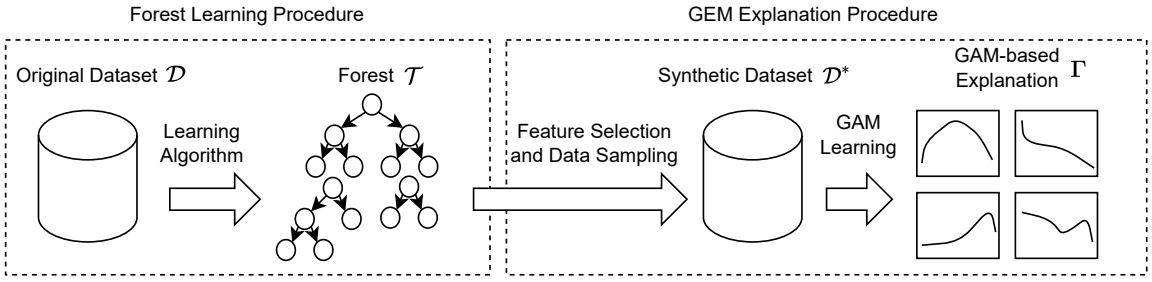


Figure 1: The GAM-based Explanation of Forests (GEF) framework.

- GEF can efficiently provide *global* explanations with a single training step;
- GEF provides *local* explanations that inform the analyst on the impact of the most important features, and also on the expected outcome of the model after slight modifications of such features.

2 RELATED WORKS

We report two different types of works present in the literature that are related to the two main steps of our framework: works encompassing various forms of dataset generation techniques, and works that aim to explain a machine learning model by different means.

Dataset generation techniques. To the best of our knowledge, limited effort has been devoted to the study of dataset generation techniques that can create a dataset from scratch using only the information available within a forest. The approach most relevant to this work is the method proposed by Cohen et al. [5]. The authors used the information encompassed within the splitting values of a forest to generate synthetic data with the aim of training a compact surrogate model. Indeed, the authors propose an intuitive heuristic strategy of considering all the midpoints between the splitting thresholds on the same features. In our analysis (see section 3) we consider this method and we also explore additional methods of exploiting the information enclosed in the given forest with the aim of generating synthetic data.

It is worth citing that there are works in the Adversarial Machine Learning (AdvML) field that use a very small set of the inputs representative of the input domain to build a synthetic dataset, as in the techniques proposed by Papernot et al. [26, 27]. In addition, besides the AdvML field, various techniques are used to generate (partially) data for privacy concerns [7], and, in particular, in some case a RF is used as a synthesizer to perform multiple imputations of data [4]. These methods do not apply directly to the scenario of this work.

Explainable Artificial Intelligence. In XAI we can distinguish between *global* and *local* explainers [12]. Global explainers usually aim at creating a surrogate model that is understandable and has the same behavior as the given black-box model. Tree-prototyping methods fall into this category, where a large forest is summarized by a single and simpler decision tree, or a small number of representative trees [32]. To some extent, Knowledge Distillation [16] can be used to distill an interpretable model out of a complex one. In addition, some standard visualization methods such as the Partial Dependence Plot [8] or the Individual Conditional Expectation [10] are used to get a global understanding of the model behavior, even though in this case the explanation is not an interpretable model but only a representation of the marginal effect of (usually) one or two features.

Local explainers instead train a simple model to provide an explanation of the behavior of the complex model in a *local area* of the feature space. For instance, we might be interested in understanding why a given instance is misclassified. These methods differ in the definition of the *local area* and in the surrogate produced. LIME [28], SHAP [22], and LORE [11], among others, belong to this category.

Other methods try to be both local and global, such as in [24], where a matrix-like visualization technique is used to represent the rules learned by a RF, and in [35] where the authors try to include different visualization techniques to overcome the drawbacks of each approach. Even though the linked visualization techniques present in the literature offer a powerful way to investigate simple ensemble of trees, they usually suffer from scalability problems because they try to fully conserve the structure of each tree that becomes an infeasible way to explain a forest when the number of trees becomes larger. Finally, in this category, it is worth highlighting that also the popular tool SHAP can be used to explain a model globally by aggregating the local explanations on the whole training set [21]. In this work we compare our method mainly with the latter, acknowledging its theoretical solid foundations and its wide usage among practitioners.

Another distinction that we can make is between *model-agnostic* and *model-specific* explanation techniques. As the names suggest, a model-agnostic method can be used with every black-box model since it does not depend on its internal characteristics, and, on the other hand, a model-specific explanation can be used only with specific types of black-box models because it exploits their inner details. During the years a variety of model-specific methods have been developed to explain the prediction made by neural networks, including Saliency Maps [31] and representation of the features learned [25], among others. However, little attention has been paid to explanations specific to forests of decision trees, perhaps considering them quite explainable by themselves while they are de facto black-box models when they encompass hundreds or thousands of trees.

In this work we propose a novel post hoc explanation method, named GEF, that uses GAMs as surrogate models to explain forests of decision trees. A GAM is known to be a highly explainable model since, in its basic form, is a simple sum of a limited set of univariate basis functions that can be used to *explain* how each single feature impacts the final prediction of the given black-box model.

3 GAM-BASED EXPLANATION OF FORESTS

Our goal is to build a GAM explainer Γ for a given forest of decision trees $\mathcal{T}$, where the original training dataset $\mathcal{D}$ is unknown for privacy or other reasons. For training Γ, we need to build a synthetic dataset $\mathcal{D}^*$, generated by exploiting only the information encompassed by $\mathcal{T}$. Furthermore, we want to keep the complexity of Γ as low as possible while maintaining a high level of fidelity with respect to the original model $\mathcal{T}$.

In the following, we briefly introduce GAMs, then we discuss *(i)* how to select a subset of relevant features and feature interactions in order to maintain the complexity of the surrogate model low, *(ii)* how to generate training samples in such feature space, and *(iii)* how to train the explanation GAM.

3.1 GAMs

Before discussing GAMs we introduce some notation. Let $\boldsymbol{x} \in \mathbb{R}^d$ be an instance in a d-dimensional vector space, with x_j the value of its j-th feature. An instance $\boldsymbol{x}$ is associated with a target label y generated by some unknown target function g, i.e., $g(\boldsymbol{x}) = y$. A GAM in its basic form models the target variable y as a combination of p smooth functions, and it can be defined as follows [15]:

$$l(\mu(\boldsymbol{x})) = \alpha + \sum_{j=1}^{p} s_j(x_j), \qquad E[s_j(x_j)] = 0.$$

Where $\mu(\boldsymbol{x}) = E[y|\boldsymbol{x}]$ and $l(\cdot)$ is the so-called link function that is able to describe the relationship between the p functions $s_j(x_j)$ and the expected value of y, while α is a learned constant value that represents the intercept of the model. From the explainability point of view, the analyst is promptly provided with a plot of each basis function s_j that clearly describes the relation between each feature and the target variable y. Each s_j is usually modeled as a cubic spline, which is a piecewise cubic polynomial with continuous derivatives up to order 2. By fitting a GAM on a dataset $\mathcal{D}^* = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$ of N training instances, we can find an approximation of g. Indeed, a GAM with cubic smoothing splines is the minimizer of the following cost function:

$$J(\alpha, s_1, \ldots, s_p) = \sum_{i=1}^{N} \left(y_i - \alpha - \sum_{j=1}^{p} s_j(x_{ij}) \right)^2 + \sum_{j=1}^{p} \lambda_j \int [s_j''(x_j)]^2 dx_j$$

where x_{ij} is the value of the j-th feature of the i-th instance in $\mathcal{D}^*$. The function J formalizes the trade-off between the least squared error of the GAM model and the smoothness of each spline weighted by the λ_j parameters. We highlight that smoothness reduces the overfitting risk and improves explainability by limiting the "wobbly" behavior of the learned splines.

GAMs have been used to build explainable models from data [19]. Interestingly, bi-variate components can be included to improve accuracy [20] thus formalizing a GAM as:

$$l(\mu(\boldsymbol{x})) = \alpha + \sum_{j=1}^{p} s_j(x_j) + \sum_{j=1,k=1}^{j=p,k=p} s_{jk}(x_j, x_k).$$

A judicious use of bi-variate components is however required. The number of bi-variate components is potentially quadratic in the number of features thus increasing the computational cost, overfitting risk, and also reducing the explainability of the final GAM for a human who is overwhelmed by a large number of splines. In this work we aim to build GAM explainers that include a limited number of univariate and bivariate components, i.e., features and features pairs, as described in the following section.

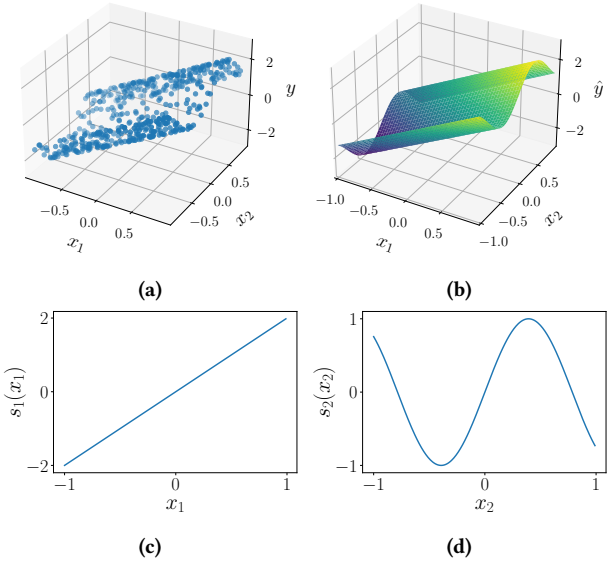


Figure 2: A toy example of a dataset that can be fitted with a GAM.

To conclude the description of GAMs, in Fig. 2 we present a toy example of a set of bi-variate data samples and how they can be easily described by a GAM. Even in this toy example, understanding the behavior of the data plotted in Fig. 2a might not be straightforward. A GAM model, as the one depicted in Fig. 2b, can instead easily show the role of the two variables. In fact, the GAM fitted on such data points is $\hat{y}(\mathbf{x}) = s_1(x_1) + s_2(x_2)$, where $s_1(x_1)$ and $s_2(x_2)$ are illustrated respectively in Fig. 2c and Fig. 2d. It is now clear to the analyst that the data at hand can be easily explained with a linear contribution from feature x_1 and a sinusoidal trend from feature x_2.

From a practical point of view, after the last step of GEF the analyst has the chance to analyze the GAM created and verify how the variation of each feature affects the final prediction. The final result available to the analyst looks similar to the functions of one variable available in Fig. 2c and Fig. 2d, and thus we assume that it is easy for a practitioner familiar with additive models to analyze the contribution of each feature to the final prediction. We recall that each contribution is simply summed up to obtain the final prediction in case we have $l(\mu(\mathbf{x})) = \mu(\mathbf{x})$. Due to their additive form, GAMs are normally considered intrinsically highly interpretable models among researchers [20, 23, 36]. We point out that GAMs are not the only type of models available that can be useful to explain a complex forest, also models that are less general can be used, such as Generalized Linear Model or even a simple linear regression, and using such models instead of GAMs can have pros and cons. For example, a simple linear regression model of the form $\hat{y}(\mathbf{x}) = \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_d x_d$ is fairly easy to interpret because the contribution of each feature x_j corresponds to its corresponding weight d_j, and therefore it is considered more interpretable with respect a GAM. However, the higher interpretability of a simple linear regression model comes

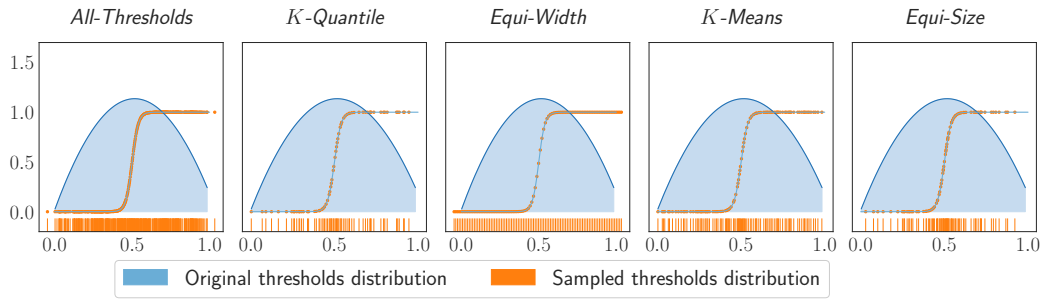


Figure 3: The results of different sampling using the five proposed strategies. The original thresholds distribution is pictorially presented through a kernel density estimation with a gaussian kernel, while the sampled thresholds is showed using a rug plot.

with the cost of having a very limited flexibility in fitting complex functions, e.g., the function in Fig. 2d cannot be approximated reasonably well with a linear function. On the other hand, GAM can comparatively approximate a larger variety of functions.

3.2 Univariate components selection

We focus on the most commonly used type forests made of binary decision trees where each node predicate is in the form $x_i \leq v$, where the i-th features f_i is tested against the threshold v. In order to build our explainer we assume to have full access to the given forest $\mathcal{T}$, and therefore to have full knowledge of its structure, trees, and nodes. We exploit this information to define the feature space of $\mathcal{D}^*$. Given these features, we can then sample a synthetic dataset $\mathcal{D}^*$ from the defined feature space.

A naïve strategy is to select all the features, denoted by F, occurring in $\mathcal{T}$. Unfortunately, this basic approach has the disadvantages of increasing the cost of training the Γ, "sparsifying" the feature space of $\mathcal{D}^*$, and reducing the interpretability of the final Γ due to the large number of splines possibly generated.

We thus perform feature selection to identify a subset of the most relevant features present in the forest $\mathcal{T}$. From a practical point of view, most forest training libraries store the loss reduction contributed by each node in $\mathcal{T}$. Therefore, we resort to selecting the most important features by accumulating their loss reduction across the nodes in the forest where the feature occurs. We denote the resulting set of features as F', $F' \subseteq F$. In our framework, we let the analyst balancing to chose the number of univariate components $|F'|$ so as to balance the accuracy of the generated GAM and its complexity.

3.3 Sampling

Once the feature set F' is given, we can then apply different strategies to generate $\mathcal{D}^*$. We define the *sampling domain* D_i of a feature $f_i \in F'$ as the subset of admissible value in $\mathbb{R}$ for f_i. We then use the following *Random Sampling* strategy. An instance $x \in \mathcal{D}^*$ is generated by sampling uniformly at random in the multidimensional space $D_1 \times \ldots \times D_n$, where D_i is the sampling domain of F_i, $F_i \in F'$. The sampling is repeated N times to build a dataset $\mathcal{D}^*$ of N instances. Larger values of N improve the coverage of the feature space for the GAM learning. Moreover, the larger $|F'|$ is, the larger N should be.

Sampling strategies face the issue of selecting the points that best represent the function to be modeled. By analyzing the given forest $\mathcal{T}$, we can extract the feature thresholds used which can be considered the most relevant points in the features space according to the forest itself. In fact, we aim to produce a more fine-grained sampling for the areas of larger variability in the forest predictions. Also borrowing from feature discretization strategies [34], in the following we investigate different approaches to define the sampling domains. Except for the first method, we denote with K the number of points in each D_i.

All-Thresholds. Given the list of increasing feature thresholds $V_i = (v_1, \ldots, v_t)$ occurring in the predicates $f_i \leq v$ in $\mathcal{T}$, let $W_i = (w_1, \ldots, w_{t-1})$ be the list of midpoints of the corresponding V_i where $w_j = (v_j + v_{j+1})/2$. We define the domain D_i corresponding to the feature f_i as $D_i = \{w : w \in W_i\} \cup \{v_1 - \epsilon, v_t + \epsilon\}$, where v_1 and v_t are respectively the minimum and maximum thresholds, and ϵ allows for extending the sampling domain beyond such thresholds. In our experiments we set $\epsilon = 0.05(v_t - v_1)$. It is worth noticing that we take the midpoints of consecutive thresholds v_j and v_{j+1} to ensure a more representative dataset and to avoid the corner cases where a feature value is equal to a node threshold. This method is equivalent to the approach used by [5].

K-Quantile. The sampling domain of each feature is defined as the set of the K-quantiles of V_i.

Equi-Width. We define the sampling domain of each feature as the set of K points that split the interval $[v_1 - \epsilon, v_t + \epsilon]$ into sub-intervals of equal size.

K-Means. We first cluster known feature thresholds V_i by using the k-Means algorithm with $k = K$, and finally define the sampling domain of the feature as the centroids of the resulting clusters. If the number of different values in V_i is lower than the number of centroids, we use $k = min(|V_i|, K)$.

Equi-Size. The list of increasing thresholds V_i is split into K contiguous sublists of equal size, and then the average value of these sublists defines the sampling domain of a feature. It is a similar strategy to *K-Quantile* but with the main difference that we do not consider the exact values of the quantiles in V_i, but we rather compute the average of the values between two quantiles.

The last four sampling strategies proposed aim at reducing the size of the sampling domain when the number of distinct thresholds is large. *K-Quantile* is designed to reuse the exact thresholds values available, *Equi-Width* takes evenly-spaced points in the feature domain, *K-Means* tries to model the data distribution so as to focus on denser areas, while *Equi-Size* has the goal of properly covering the full domain. For the sake of clarity, Fig. 3 presents a comparison of the samples obtained using the different sampling strategies for a feature F_i that behaves like a sigmoid function of the form $y = exp(50(x - 0.5))/(exp(50(x - 0.5)) + 1)$.

As depicted in the figure, the majority of the thresholds in the original forest $\mathcal{T}$ are accumulated in the sub-domain of the function with the largest variability, which in the illustrative example is around 0.5, and it is worth noticing that the various techniques behave differently in this region. *k-Quantile*, *K-Means*, and *Equi-Size* follow the density of the original distribution, and they emphasize more the difference between regions with low and high density. On the other hand, *Equi-Width* is not affected by the original distribution, and regions with higher variability are treated as the ones with lower variability.

Eventually, a dataset $\mathcal{D}^* = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ of N training instances is generated by sampling $\boldsymbol{x}_i$ as discussed above and feeding the forest $\mathcal{T}$ with such instance $\boldsymbol{x}_i$ to get the corresponding prediction y_i.

3.4 Bi-variate components selection

To improve the accuracy of the explainer Γ, we include also bi-variate splines. To do so we need to identify a subset of feature pairs that are more likely to be relevant, and thus we limit our search to the set of features that are present in F', following the so-called *heredity principle* [6], according to which an interaction between f_i and f_j is considered interesting only if both f_i and f_j are present as main effects, i.e., $f_i, f_j \in F'$. Note that this step is conducted independently of the dataset $\mathcal{D}^*$, and therefore the bi-variate components can be identified before the data sampling.

Since the number of all possible pairs can be extremely large, we propose four different heuristic strategies to identify a suitable number of important pairs. Starting from the one with the smallest computational cost to the one with the highest complexity we have: *Pair-Gain* exploits the features importance, *Count-Path* and *Gain-Path* use the information available in the nodes, and *H-Stat* employs the H-statistic [9]. Therefore, for each pair of features $f_i, f_j \in F'$, we define four different ways to measure the importance $I(f_i, f_j)$ of their interaction as follows.

Pair-Gain. In this case, we simply define the strength of a feature interaction as $I(f_i, f_j) = I(f_i) + I(f_j)$ where $I(f_i)$ is the accumulated loss reduction as in the univariate feature selection. This can be considered a quick baseline, with a very limited computational cost.

Count-Path. Given a tree $T \in \mathcal{T}$, we define the importance $I_T(f_i, f_j)$ of the feature pair (f_i, f_j) limited to the tree T as the number of decision paths in T that include both features. The Count-Path importance can be computed recursively as follows. Given the root r of T, we denote with f_r the feature used in the splitting criterion of r, and then we set the value $I_T(f_r, f_j)$ equal to the number of f_j, $j \neq r$, present in T. Then we recursively repeat the procedure in the left and right subtrees of r accumulating the results for all feature pairs until reaching the leaves. We discard the order of appearance as we consider $I_T(f_i, f_j) = I_T(f_j, f_i)$. Finally, the Count-Path importance for the forest $\mathcal{T}$ is computed by summing up the contributions from its trees: $I(f_i, f_j) = \sum_{T \in \mathcal{T}} I_T(f_i, f_j)$.

Gain-Path. In this strategy we simply apply the same heuristic as in *Count-Path*, but instead of counting the nodes, we accumulate the minimum gain found in each pair of nodes. We recall that each node stores the reduction of loss achieved when the node was added to the tree at training time. In this regard, Gain-Path is a weighted version of Count-Path.

H-Stat. Given the list of all the possible pairs of features (f_i, f_j) we define their importance $I(f_i, f_j)$ by computing the H-statistic. We recall that the H-statistic $H(f_i, f_j)$ between the features f_i and f_j is defined as:

$$H(f_i, f_j) = \frac{\sum_{k=1}^N \left[\hat{F}_{ij}\left(x_{ki}, x_{kj}\right) - \hat{F}_j\left(x_{kj}\right) - \hat{F}_j\left(x_{kj}\right)\right]^2}{\sum_{i=1}^N \hat{F}_{ij}^2\left(x_{ki}, x_{kj}\right)}$$

where $\hat{F}_i$ is the one-dimension estimated partial dependence function for the feature f_i and $\hat{F}_{ij}$ is the bi-dimensional partial dependence function for the features f_i and f_j. In our scenario the partial dependence functions for each feature, and for each pair of features, are computed from a sample of the synthetic dataset created $\mathcal{D}^*$.

Given any of the above strategies, we eventually select only the top feature pairs sorted by their importance $I(f_i, f_j)$. We denote with F'' the set of the most important interactions selected by the analyst. In summary, while F' defines the feature space of $\mathcal{D}^*$ and the set of univariate splines of Γ, the set F'' defines the bi-variate components of Γ.

3.5 Fitting the GAM

In order to create an explanatory model Γ that accurately mimics the behavior of the original forest $\mathcal{T}$ while maintaining a certain user-defined level of explainability, we assume that the end-user is in charge of selecting the number of univariate components $|F'|$ and the number of bi-variate components $|F''|$.

Thus, to sum up the whole procedure, we first employ the feature selection procedures to identify the feature set F', we create the synthetic dataset $\mathcal{D}^*$ using one of the sampling strategies illustrated above, finally we select the set of interactions F''. The last step consists in fitting a GAM on $\mathcal{D}^*$ so as to have an explainable surrogate of the given forest $\mathcal{T}$.

To maintain a low level of complexity and automate the explanation process as much as possible, we propose to use only third-order spline terms with a fixed number of p-spline basis for each continuous feature in F', factor terms for each categorical variable in F', and penalized tensor products for each variable in F''. Since the information about the type of features, categorical or not, is not often available inside the forest, we use, as a heuristic, the number of thresholds present for a feature $\boldsymbol{x}_i$ to identify it as categorical. This means that, if $|V_i| < L$ we assume that $\boldsymbol{x}_i$ is categorical. In our tests we set $L = 10$.

In addition, in a regression problem we set the link function l to a linear function and we assume that y follows a Normal distribution, while in the classification case we set l to a logistic function and we assume that y follows a Binomial distribution.

Finally, to find the best values for the penalization coefficients λ_j of each term, with $1 \leq j \leq p + q$, we used a Generalized Cross Validation (GCV), varying λ_j equally for each term used, meaning that $\lambda_1 = \lambda_2 = \ldots = \lambda_{p+q}$.

4 EXPERIMENTAL EVALUATION OVER SYNTHETIC DATA

GEF is implemented as a Python package and made available in a open source repository[1]. The software used to train the initial forest $\mathcal{T}$ is LightGBM [17] and the software used to fit the explanatory GAM is PyGam [30].

4.1 Dataset and Model Description

In the following we use synthetic data to validate the capability of the explainer Γ generated by GEF to reconstruct the original

[1]Open source implementation available at https://github.com/veneres/gef

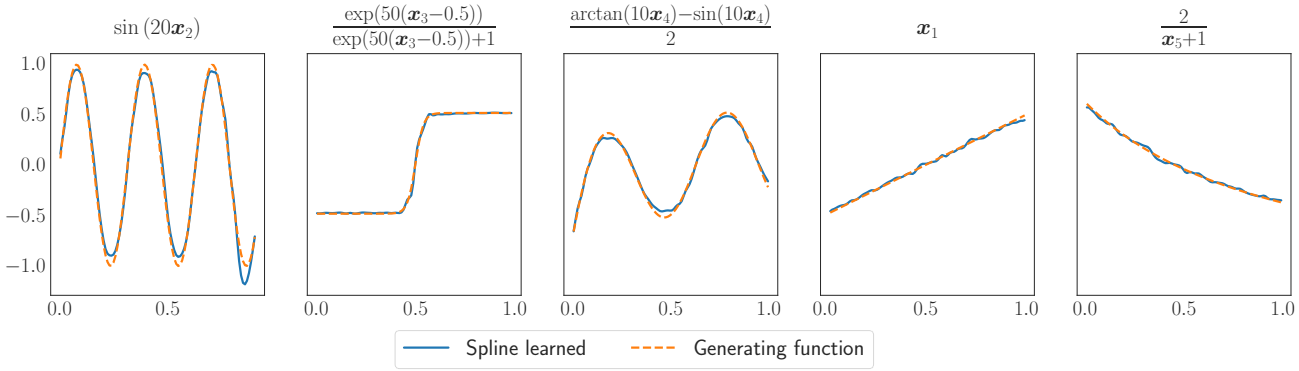


Figure 4: The final components obtained by applying GEF to forest $\mathcal{T}$ trained over $\mathcal{D}'$, where no interactions are present. Each component is centered with respect to its mean, and they are sorted from left to right by their computed importance.

target function that the forest $\mathcal{T}$ approximates, to show the impact of the feature interaction selection strategies and to analyze the impact of the sampling strategies.

In all the experiments, if not specified otherwise, the Root Mean Squared Error (RMSE) reported is computed on a fraction of $\mathcal{D}^*$, used as test set, by comparing the predictions made by the generated explanation Γ and the original forest $\mathcal{T}$, because we assume to not have access to the original data.

To investigate the quality of the approach proposed we use two synthetic datasets $\mathcal{D}'$, $\mathcal{D}''$, for which the target functions g', g'' are known, and defined as follows:

$$g'(\boldsymbol{x}) = \boldsymbol{x}_1 + \sin(20\boldsymbol{x}_2) + \frac{\exp(50(\boldsymbol{x}_3 - 0.5))}{\exp(50(\boldsymbol{x}_3 - 0.5)) + 1} + \frac{\arctan(10\boldsymbol{x}_4) - \sin(10\boldsymbol{x}_4)}{2} + \frac{2}{\boldsymbol{x}_5 + 1}$$

$$h(\boldsymbol{x}_i, \boldsymbol{x}_j) = 2\exp\left(-\frac{1}{\sqrt{2\pi}}\frac{(\boldsymbol{x}_i - 0.5)^2 + (\boldsymbol{x}_j - 0.5)^2}{2}\right)$$

$$g''_{\Pi}(\boldsymbol{x}) = g'(\boldsymbol{x}) + \sum_{(f_i,f_j)\in\Pi} h(x_i, x_j)$$

Function g' maps a 5-dimensional instance $\boldsymbol{x}$ by means of 5 "generator" functions, where each function maps the corresponding feature $\boldsymbol{x}_i$ to a real value y_i. We structured the function g' in such a way that the contribution y_i of each generator function is well bound in the range $[-1, 2]$, thus we do not have a function that dominates all the others. Whereas function g''_{Π} is a modification of g' where Π is a set of three feature pairs which are injected to test the ability of the proposed strategies to identify correctly the interactions present inside the data and, as a consequence, represented by the forest. Specifically, we tested our solution with all the 120 possible triples of interactions pairs. In addition, a noise component $\mathcal{N}(0, 0.1^2)$ was drawn from a Gaussian distribution and added to each generating function. Datasets $\mathcal{D}'$ and $\mathcal{D}''$ are built by *Random Sampling* 10,000 instances in the domain $[0, 1]^5$, with 8,000 instances used as training set and 2,000 as test to evaluate the performance of the model. On each dataset we train a GBDT forest using LightGBM [17] by fine-tuning the number of leaves trees in $\{10, 100, 1000\}$, number of leaves per tree in $\{32, 64, 127, 256\}$ and learning rate in $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ with a 5-fold cross-validation. We used also a 25% of the training dataset as a validation set for the early stopping criterion. The final best configuration found for $\mathcal{T}$, after the cross-validation, is made of 1000 trees with 32 leaves each and with a learning rate of 0.01 for both the datasets under investigation. After previous analysis, we found that the number of instances N of $\mathcal{D}^*$ does not affect significantly the results, and therefore N is fixed to 100,000.

4.2 Forests explanation with GEF

We investigate the results obtained by GEF along three different axes: the impact of the sampling strategy, the effectiveness of the interaction detection methods, and finally the function reconstruction capability. Since the number of features used is limited, to keep the complexity of these synthetic examples bounded on purpose, we skip the investigation over the feature selection used to create F' and F'' and we fix the number of univariate and bi-variate components respectively to $|F'| = 5$ for both datasets, while we set $|F''| = 0$ for the forest learned over $\mathcal{D}'$, and $|F''| = 3$ for the forest learned over $\mathcal{D}''$.

Sampling strategies comparison. Using different sampling strategies to create $\mathcal{D}^*$ can heavily affect the accuracy of the GAM used as an explainer. In Fig. 5 we show how the RMSE changes with respect to the number of sampled points K used, and the sampling strategies employed. We consider values of K up to 20000 which corresponds to the number of thresholds baseline. In particular we focus only on the simplest synthetic function, i.e., $g'(\boldsymbol{x})$, and the corresponding dataset $\mathcal{D}'$. The *Equi-Size* strategy performs best, although only on specific values of K. In general, *K-Quantile* and *Equi-Size* can give a better accuracy with respect to the baseline *All-Thresholds*, while *K-Means* and *Equi-Width* perform worse. The results obtained told us that the sampling strategies employed heavily affect the fidelity of the model distillation. They also confirm the intuition behind the *Equi-size* strategy: it is better to sample more points inside the range where the variability of the unknown generating function is larger, that is also the region where more splits are generated by the forest under investigation.

True function reconstruction. Even though investigating the impact of the different sampling strategies of GEF is important, the main goal of the explanation technique proposed is to reconstruct the function learned by the model. To do that we run GEF setting the sampling strategies to *Equi-size*, and $K = 12{,}000$, using in this way the setting that gives the best accuracy with respect to the RMSE as shown in the previous paragraph. The results are presented in Fig. 4, and they show that the learned components

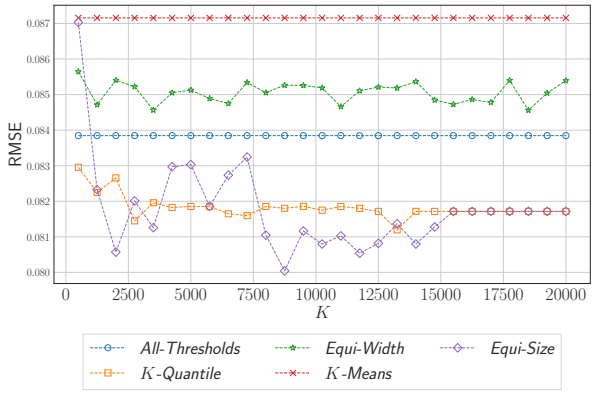


Figure 5: RMSE for different sampling strategies and number of sampled points K.

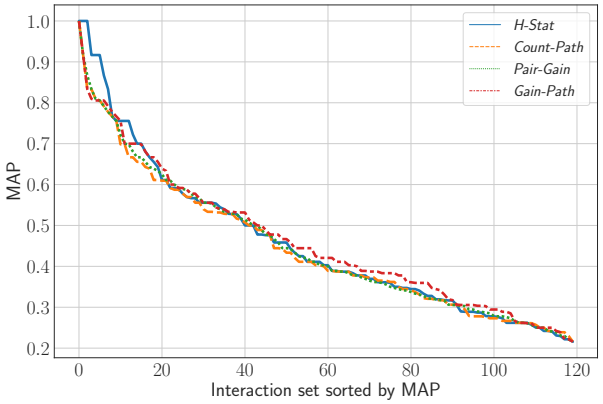


Figure 6: Interaction detection strategies comparison. The horizontal axis indicate the 120 possible sets of interactions.

nicely match the original generator functions with few exceptions at the margins of the feature domains. The unusual behavior of the learned splines near the extremes of the features domain can be seen as a drawback of the specific sampling strategies since it is more accurate in the regions of the feature domains associated with a higher number of splits in $\mathcal{T}$. Let's remark that GEF is able to reconstruct the original generator functions by exploiting only the information encompassed by the forest and without accessing the original training data.

Interaction detection. By focusing on the second dataset $\mathcal{D}''$ where three interactions are added, we analyze the ability of our methods to identify interactions between the data using the four strategies *Pair-Gain*, *Count-Path*, *Gain-Path*, and *H-Stat*. As previously stated, we remark that we evaluate the ability of our method to identify the presence of interactions inside $\mathcal{T}$ with all the possible triples of interactions that could be present. Since in the base function $g'(\boldsymbol{x})$ there are 5 features, we can have $\binom{5}{2} = 10$ possible different interactions, therefore the total number of different triples of interactions is $\binom{10}{3} = 120$. In our analysis we evaluate the accuracy of GEF to identify the interactions between the features on 120 different realizations of g''_{Π} using all the 120 possible sets of interactions.

We borrow the Average Precision (AP) metric from the ranking problem to measure the ability of the four heuristic approaches at ranking the candidate interactions and we show the results obtained in Fig. 6. To allow a simpler comparison between the AP of the different strategies, in Fig. 6, we sorted the 120 possible sets of interactions in the horizontal axis by their AP obtained with each different strategy. This means that on the left of the chart we have the best AP of each strategy and from left to right we can follow the degradation in performance when different interaction sets are added to the generating function g'. To summarize the results, we present the Mean, the Standard Deviation (SD), and the minimum and the maximum of the AP metrics in Table 1. The best results are achieved by *Gain-Path* and *H-Stat*, with a small advantage for the former. Nevertheless, a two tailed Welch's t-tests states that there is no method that is statistically difference w.r.t. *Gain-Path* with a significance level $\alpha = 0.05$. The good performance of *H-Stat* is expected. The H-statistic is an accurate test measuring of the added value of a feature interaction w.r.t. to the single features computed over the whole dataset at hand. Indeed, computing the H-statistic is computationally very expensive because it involves the computation of the partial dependence function of each feature and for each pair of features, which in turn involves making new predictions over the entire dataset (or a sample of it). Thus, the complexity is of the order of $O(N|F'|^2)$. Instead, the proposed heuristic strategy *Gain-Path* is not dataset driven, but rather model driven: we can find the interactions in the forest in linear time with respect to the number of trees, $O(|\mathcal{T}|)$, that is in general much smaller than $O(N|F'|^2)$, and with the same level of accuracy of *H-Stat*.

Fidelity of Γ. Given that Γ is fitted on the synthetic dataset $\mathcal{D}^*$, it is natural to wonder about the "fidelity" of Γ w.r.t. to original forest $\mathcal{T}$ on the original dataset on which $\mathcal{T}$ was trained. In general we assume the original dataset is not available, but we can anyway pretend to have access to it in this synthetic evaluation setting.

We evaluate two different kinds of fidelity. We want to evaluate how close is the prediction of Γ *i)* to the prediction of the forest $\mathcal{T}$, denoted $\mathcal{T}(\boldsymbol{x}_i) \mid \boldsymbol{x}_i$, and *ii)* to the original target label in the training dataset of $\mathcal{T}$, i.e., $y_i \mid \boldsymbol{x}_i$. In Table 2 we report the coefficient of determination (R^2) measured on the test split of the two synthetic datasets $\mathcal{D}'$ and $\mathcal{D}''$. For the dataset $\mathcal{D}''$, we fix the interactions to $F'' = \{(f_1, f_2), (f_1, f_5), (f_2, f_5)\}$ among the 120 configurations available.

Empirical results show that Γ exhibits high R^2 w.r.t. forest prediction $\mathcal{T}(\boldsymbol{x}_i)$ on both $\mathcal{D}'$ and $\mathcal{D}''$, respectively of 0.986 and 0.938. This confirms that the fitted Γ is consistent with the forest we aim at explaining. Interestingly enough, the values of R^2 w.r.t. the original labels y_i are even larger than the original forest for $\mathcal{D}'$. Even without accessing the original dataset, the Γ model is as accurate as the forest $\mathcal{T}$, and it could even replace the original forest. On the one hand, the large accuracy might be expected thanks to the distillation mechanism. On the other hand, this result confirms the effectiveness of the generation process of the dataset $\mathcal{D}^*$ on which Γ is fitted.

5 EXPERIMENTAL EVALUATION OVER REAL-WORLD DATA

We now apply GEF to a forest trained on real-world data, namely *Superconductivity* [14] and *Census* [18] datasets, respectively selected as representative for regression and classification tasks. First of all we give an overview of the dataset involved in our evaluation, then, for sake of brevity, we analyze in detail only the explanation made on the forest trained over *Superconductivity*,

Table 1: Mean value, SD, Minimum and Maximum value of Average Precision for the different interaction detection strategies.

	Pair-Gain	*Count-Path*	*Gain-Path*	*H-Stat*
Mean	0.450	0.445	**0.463**	0.457
SD	0.175	0.172	0.172	0.191
Min	0.216	0.216	0.216	0.216
Max	1.000	1.000	1.000	1.000

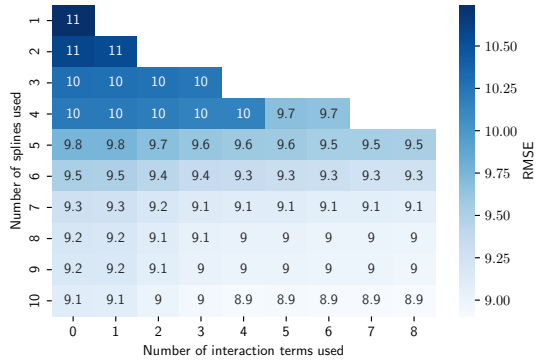


Figure 7: *Superconductivity* dataset: feature selection. Each cell contains the RMSE computed w.r.t. $\mathcal{D}^*$.

while describing only the final results for *Census*, omitting part of the analysis made to chose the parameters needed, such as the sampling strategies or the number of sampled points K.

5.1 Datasets and Models descriptions

The first dataset, *Superconductivity*, contains 81 features extracted from 21,263 superconductors, and the associated goal is to predict their critical temperature. The features present inside *Superconductivity* describe various physical and chemical properties of a sample of superconducting materials downloaded from the Superconducting Material Database offered by Japan's National Institute for Materials Science. The value of each feature is derived by the authors of the original paper aiming to find a data-driven model to overcome the rudimentary empirical rules used in the past to synthesize superconductors. Thus, the main goal of the dataset is a good test case for our explanation method to *discover new knowledge*, which is one of the main motivations in XAI [1]. In addition, the second motivation behind using *Superconductivity* as a real-world dataset in our experiments is that it has a fairly large number of features.

On the other hand, the second dataset, *Census*, has the goal to predict the annual salary of a person given some personal information. It is selected among others because it contains various sensitive features such as race, sex, and relationship information that brings attention to another problem that XAI aims to answer, i.e. *explain to justify* [1]. Thus, while it contains only 14 attributes for 48,842 samples, the nature of the features makes it a perfect example for our purposes.

To train the forest for *Superconductivity* and *Census* datasets, we used a strategy similar to the one used for the synthetic dataset. The software used to train the initial forests is LightGBM [17]

Table 2: R^2 values of the Forest $\mathcal{T}$ and the explainer model Γ produced by GEF over the test splits of the synthetic datasets $\mathcal{D}'$ and $\mathcal{D}''$.

	$\mathcal{D}'$		$\mathcal{D}''$	
	$\mathcal{T}(\boldsymbol{x}_i) \mid \boldsymbol{x}_i$	$y_i \mid \boldsymbol{x}_i$	$\mathcal{T}(\boldsymbol{x}_i) \mid \boldsymbol{x}_i$	$y_i \mid \boldsymbol{x}_i$
Forest ($\mathcal{T}$)	—	0.980	—	0.986
Explainer (GAM)	0.986	0.982	0.938	0.931

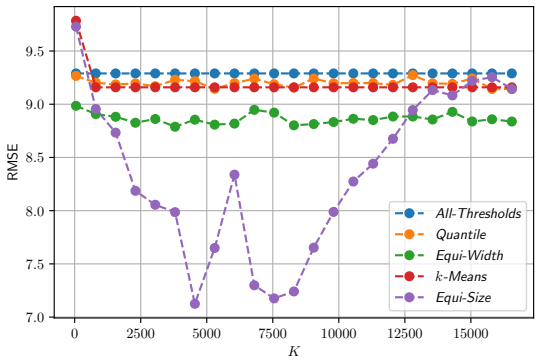


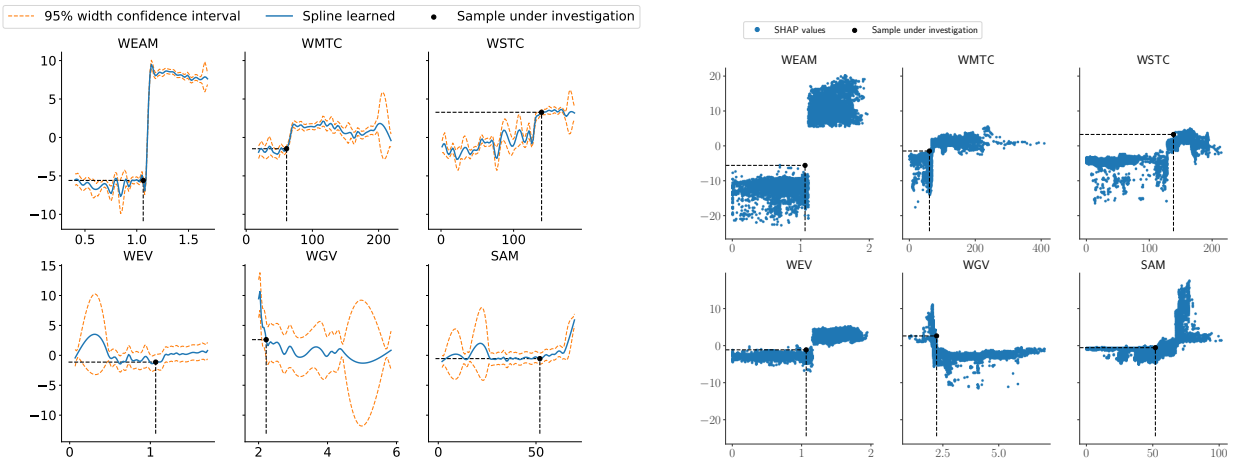
Figure 8: *Superconductivity* dataset: sampling strategies comparison.

by fine-tuning the number of leaves in $\{100, 1000, 1000\}$, number of leaves per tree in $\{32, 64, 127, 256\}$ and learning rate in $\{10^{-3}, 10^{-2}, 10^{-1}\}$ with a 5-fold cross-validation. Even though not relevant for the main purpose of this paper, we highlight that by using the proposed learning strategy we can achieve good results without any particular pre-processing of the data. For example, for the *Superconductivity* dataset the RMSE evaluated on the test set is equal to 11.7 which is not far away from the results presented in [14], where the RMSE is 9.5. Finally, standard pre-processing is applied only to *Census*, where redundant features (*education* and *education-num*) are dropped, and one-hot encoding is applied to the categorical features: *workclass*, *marital-status*, *occupation*, *relationship*, *race*, *sex*, *native-country*.

5.2 Forest explanation with GEF

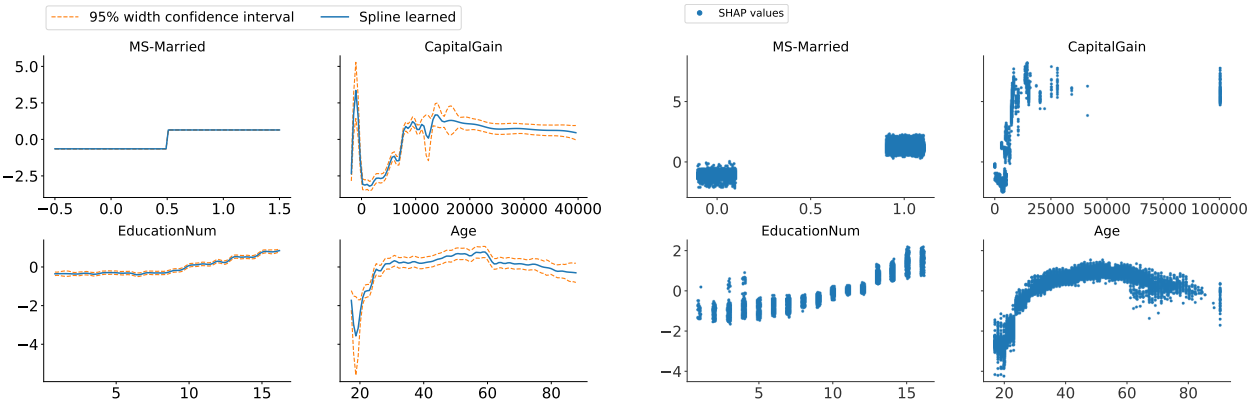
In Fig. 7 we show the RMSE of the explained Γ computed over $\mathcal{D}^*$ using as true labels the predictions made by the original forest. We use *All-Thresholds* as sampling strategy and *Count-Path* as feature interaction heuristic while varying the number of univariate and bi-variate components. As expected the greater the number of components, either univariate or bi-variate, the better the accuracy. However, a limited number of terms is sufficient to achieve a good accuracy, and already with 7 splines we can obtain an accuracy distant only 5% ca. with respect to the maximum achievable in the provided scenario, i.e. 9. In addition, in case 7 splines are used, there is no need to add interaction components since the RMSE improves only of 2% adding 8 interactions. Therefore we decide to use 7 splines without adding any interaction components.

Sampling strategy. After having fixed the number of splines and interaction terms to 7 and 0 respectively, we analyze the results obtained by using the 4 sampling strategies presented in section 3. Fig. 8 reports the behavior of GEF on varying the



(a) (b)

Figure 9: *Superconductivity* dataset: comparison of the univariate splines obtained with GEF (9a) and the partial dependence analysis of the SHAP values (9b).



(a) (b)

Figure 10: *Census* dataset: comparison of the univariate splines obtained with GEF (9a) and the partial dependence analysis of the SHAP values (9b).

sampling methods and the number of sampled points K. We can see that the number of sampled points K for each feature impacts a lot for the *Equi-Size* strategy, while all the other methods are stable with respect to the variation of K. Overall, *Equi-Size* largely outperforms the other sampling strategies after a proper tuning, confirming the previously discussed results on the synthetic dataset.

Analysis of the model. On the basis of the previous results, we fix $K = 4{,}500$ and we use the *Equi-Size* sampling strategy for the *Superconductivity* dataset. Regarding the *Census* dataset, after running the same analysis, we decide to use 5 splines and 1 interaction term, with $K = 800$ using the *K-Quantile* strategy. In Fig. 9a we show the top splines of the resulting explainer model Γ on the *Superconductivity* dataset, highlighting a randomly chosen sample of the dataset. Each plot thus shows the contribution to the final prediction on varying the value of a given feature. The title of each subplot is an acronym that represents the corresponding feature, e.g., WEAM stands for *Weighted Entropy Atomic Mass*. Similarly in Fig. 10a are presented the first 4 splines (sorted by feature importance) used to explain the forest $\mathcal{T}$ trained over the *Census* dataset.

The two plots present a global explanation for the initial forest, and from the point of view of the analyst, it is possible to effectively understand the impact of the different features and their interaction on the final prediction. For instance, taking into account the *Census* dataset, we can observe that the feature *EducationNum*, which represents the level of education a person where a low number represents a lower degree, is positively correlated with the output. This overview can also give us some hints about possible points of discontinuity, such as in *Superconductivity* where there is a big jump near a value of 1.1 for the *Weighted Entropy Atomic Mass* feature. In addition, using a Γ we also obtain the Bayesian confidence intervals of each spline, computed as in [33] that give us a measure of confidence of the model.

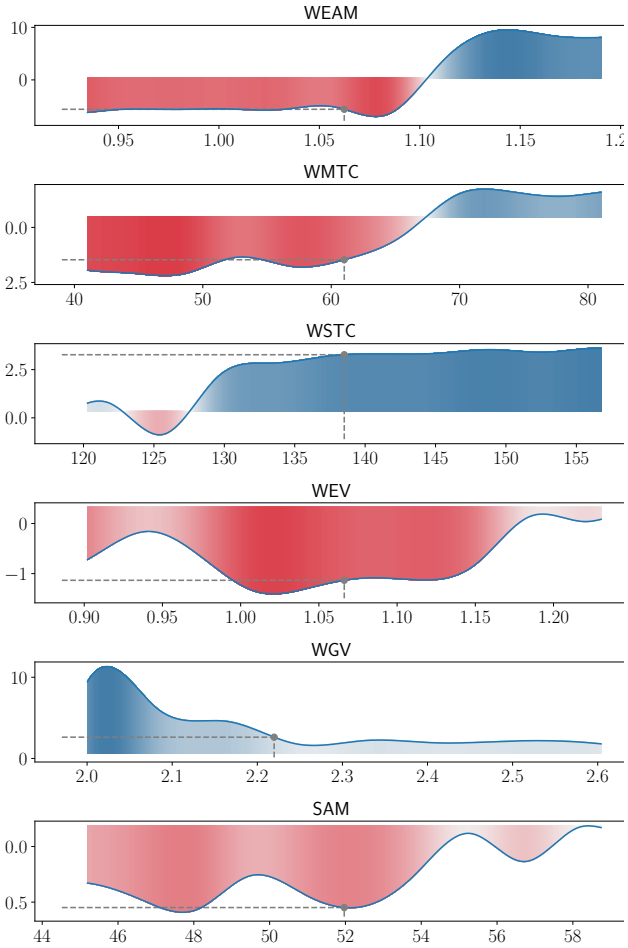


Figure 11: *Superconductivity* dataset: local explanation made by GEF over the specified sample (best seen in colored version).

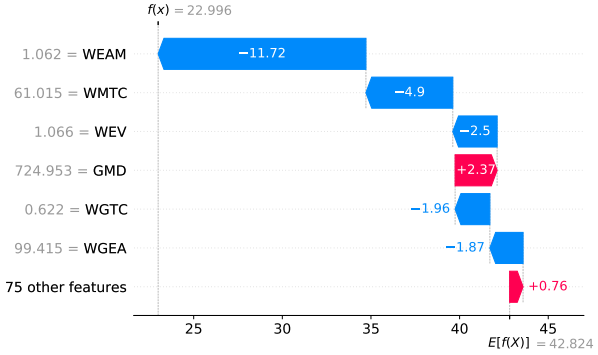


Figure 12: *Superconductivity* dataset: local explanation made by SHAP over the specified sample.

5.3 Comparison with SHAP and LIME

In order to understand the advantages and similarities between the proposed explanation technique and the state of the art, we compare GEF against SHAP and LIME. We recall that SHAP is a local explanation method that can provide a global explanation by aggregating multiple local explanations, while LIME is designed to be used only as a local explanation method.

Global explanation. We analyze the explanations proposed by the two global techniques, GEF and SHAP, at a global level comparing the splines obtained by GEF, and the partial dependence plots generated by SHAP from the forests trained over the two datasets under investigation (Fig. 9, and Fig. 10). We recall that the proposed GEF framework provides an explanation of a given forest without accessing the original training data, while the SHAP partial dependence plots require an expensive process that replicates the local SHAP analysis on every point of the dataset. By looking at the figures representing the two techniques, we first notice that the explanations are consistent with each other, meaning that the impact trend of the features under analysis is the same in GEF and SHAP. In addition, in the plots generated with GEF it is also possible to identify a 95% confidence interval computed using a Bayesian approach, while the computation of a confidence interval for the SHAP values is not so straightforward. This is a very interesting result considering that the explanation generated by the proposed GEF makes no use of the input datasets. From a qualitative point of view, we believe the GEF explanation provides a sharper picture of the role of each feature in the studied phenomenon. Rather than reporting a cloud of training data points for each feature value as in SHAP, the GAM built by GEF provides an expected additive contribution with confidence intervals.

Local explanation. Then, in Figs. 11, 12, 13 are represented the local explanations provided by GEF, SHAP, and LIME respectively, for the same point illustrated in Fig. 9 and highlighted in black.

The local explanation of SHAP provides the contribution of the most important features to the final model prediction for the selected data point with respect to the mean output. In particular, in the figure is indicated the expected value of the forest prediction function with $E(f(X))$, which in SHAP is simply approximate with the sample mean of the datasets used in input. In our case, the sample mean is computed over the training dataset. Starting from the approximation of $E(f(X))$, SHAP estimates positive (depicted in blue) and negative (depicted in red) contributions for each feature that if added to $E(f(X))$ gives the final prediction of the model for the specific sample. In this case, we can see that the feature WEAM, with a value of 1.062 has a strong negative impact over the prediction, and along with other marked negative contributions from other features this results in a final prediction below the sample average.

The explanation given by LIME is related to the values of the coefficients associated with the Ridge Regression Model fitted in the neighborhood of the sample under investigation. Deciding the size of the neighborhood that has to be considered and other parameters for the explanation can be troublesome in LIME [23]. To simplify the analysis we used only the default parameters provided by the open-source implementation proposed by the authors of the original paper [28]. To correctly interpret the figure, we highlight that features that have a negative impact on the prediction are colored in blue and features with a positive impact are colored in orange. From the explanation obtained, it is worth to notice that that the WEAM has a strong negative impact on the prediction as in SHAP, however, the Range Atomic Radius (RAR) seems also to play an important role, while in SHAP is not present in the first 6 most influential features.

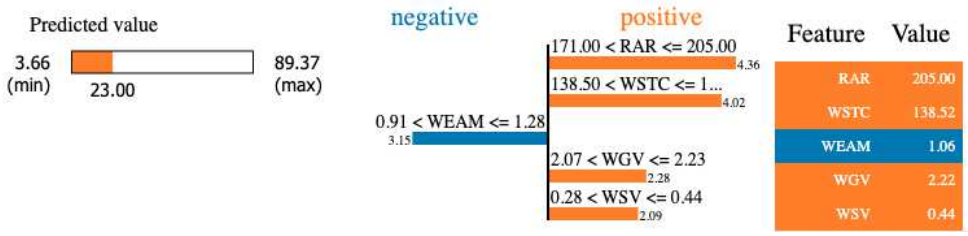


Figure 13: *Superconductivity* dataset: local explanation made by LIME over the specified sample.

More similar information as the one reported by SHAP is delivered by GEF. For the most important features we can observe their contribution to the predicted output. However, given its spline-based nature, in addition GEF shows clearly how the prediction of the model changes under small modifications of the instance under analysis. To create a local explanation with GEF we can simply zoom-in the portions of the splines of interest to obtain a better visualization of the neighborhood of a point under investigation. In particular, Fig. 11 illustrates the different from the average partial contribution of each feature with a more intense blue when the feature contribution is well above the average, and a more intense red when the contribution is well below the average. The information provided by GEF is thus better contextualized in the feature space. This is granted by the fact that GEF is able to build a global explanation model, while SHAP and LIME provide only point-wise information. Such global model, through the different "pieces" of the splines, can provide high-quality local information and understanding to the analyst.

GEF can be seen as a global explanation method, but we can also easily place a given random instance on the GAM plots and investigate how much each feature contributes to the final prediction and in particular we can interpret how the forest prediction is affected when varying a feature value. Considering the sampled data point, we can see how SHAP suggests a strong negative contribution from the feature WEAM. GEF confirms such contribution, but the analyst can clearly see how a small increment may reverse such behavior to a strong positive contribution. This kind of information is not available with the local SHAP explanation. As shown in Fig. 12, even though we can easily understand how each feature contributes to shift the prediction from the expected value, we cannot describe how a little variation of the feature could change the prediction. The same limitation is also present in LIME.

Finally, we highlight that the proposed GEF framework is largely more effective than SHAP where the partial dependent plots require the training of a surrogate model for each point reported, i.e., data instance analyzed. In fact, the computation of the SHAP values for a set of points depends on the size of the set under investigation, while with GEF the training time of the explanation only depends on the number of feature thresholds used by the forest, which is typically much smaller.

6 CONCLUSION AND FUTURE WORK

In this paper we presented a novel post hoc explanation method that uses a GAM as a surrogate model for a forest without the usage of the initial training dataset. We investigated various sampling and feature selection techniques to create a training set to learn the GAM, and showed that the resulting GAM can accurately model a forest's behavior and it can provide a crisp interpretation of the role of a subset of the most important features and feature interactions in the prediction process of the given forest. As a future work, we want to test our post hoc explanation approach to other kinds of forest, such as RF, to further confirm the possibility to apply GEF to all the ensemble of trees given that no strict assumption is made on the forest in input. Finally, we highlight that, even though the presented results are promising, a more accurate evaluation is needed, also involving the end-user to measure the quality of explanation. Testing GEF in a real-case scenario including end-users can be useful also to check the efficacy of an explanation technique that is based only on the model, and not on the training dataset, to pursue one of the main motivations behind XAI, i.e. *explain to control* [1]. Having greater control over the model means, for example, using the information contained in the terms created by GEF to understand possible unexpected behavior with certain inputs and verify the model's robustness against adversarial attacks; everything without the usage of the original training set.

ACKNOWLEDGMENTS

We thank Martina De Zan for her initial investigation on the topic that has been the starting point for our work.

REFERENCES

[1] Amina Adadi and Mohammed Berrada. 2018. Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access* 6 (2018), 52138–52160. https://doi.org/10.1109/ACCESS.2018.2870052

[2] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion* 58 (2020), 82–115. https://doi.org/10.1016/j.inffus.2019.12.012

[3] Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (2001), 5–32. https://doi.org/10.1023/A:1010933404324

[4] Gregory Caiola and Jerome P. Reiter. 2010. Random Forests for Generating Partially Synthetic, Categorical Data. *Transactions on Data Privacy* 3, 1 (2010), 27–42.

[5] Daniel Cohen, John Foley, Hamed Zamani, James Allan, and W. Bruce Croft. 2018. Universal Approximation Functions for Fast Learning to Rank: Replacing Expensive Regression Forests with Simple Feed-Forward Networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 1017–1020. https://doi.org/10.1145/3209978.3210137

[6] D. R. Cox. 1984. Interaction. *International Statistical Review / Revue Internationale de Statistique* 52, 1 (1984), 1–24. https://doi.org/10.2307/1403235 Publisher: [Wiley, International Statistical Institute (ISI)].

[7] Ashish Dandekar, Remmy A. M. Zen, and Stéphane Bressan. 2018. A Comparative Study of Synthetic Dataset Generation Techniques. In *Database and Expert Systems Applications*, Sven Hartmann, Hui Ma, Abdelkader Hameurlain, Günther Pernul, and Roland R. Wagner (Eds.). Vol. 11030. Springer International Publishing, Cham, 387–395. https://doi.org/10.1007/978-3-319-98812-2_35 Series Title: Lecture Notes in Computer Science.

[8] Jerome H. Friedman. 2001. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics* 29, 5 (2001), 1189–1232. https://doi.org/10.1214/aos/1013203451 Publisher: Institute of Mathematical Statistics.

[9] Jerome H. Friedman and Bogdan E. Popescu. 2008. Predictive learning via rule ensembles. *The Annals of Applied Statistics* 2, 3 (2008), 916 – 954.

[10] Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. 2015. Peeking Inside the Black Box: Visualizing Statistical Learning With Plots of Individual Conditional Expectation. *Journal of Computational and Graphical Statistics* 24, 1 (2015), 44–65. https://doi.org/10.1080/10618600.2014.907095

[11] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. 2018. Local Rule-Based Explanations of Black Box Decision Systems. arXiv:1805.10820 [cs.AI]

[12] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. 2018. A Survey of Methods for Explaining Black Box Models. *Comput. Surveys* 51, 5 (2018), 1–42. https://doi.org/10.1145/3236009

[13] David Gunning, Mark Stefik, Jaesik Choi, Timothy Miller, Simone Stumpf, and Guang-Zhong Yang. 2019. XAI—Explainable artificial intelligence. *Science Robotics* 4, 37 (2019), eaay7120. https://doi.org/10.1126/scirobotics.aay7120

[14] Kam Hamidieh. 2018. A data-driven statistical model for predicting the critical temperature of a superconductor. *Computational Materials Science* 154 (2018), 346–354. https://doi.org/10.1016/j.commatsci.2018.07.052

[15] Trevor Hastie and Robert Tibshirani. 1987. Generalized additive models: some applications. *J. Amer. Statist. Assoc.* 82, 398 (1987), 371–386.

[16] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. arXiv:1503.02531 [stat.ML]

[17] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 3149–3157.

[18] Ron Kohavi. 1996. Scaling up the accuracy of Naive-Bayes classifiers: a decision-tree hybrid. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*. AAAI Press, Portland, Oregon, 202–207.

[19] Yin Lou, Rich Caruana, and Johannes Gehrke. 2012. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD - KDD '12*. ACM Press, Beijing, China, 150. https://doi.org/10.1145/2339530.2339556

[20] Yin Lou, Rich Caruana, Johannes Gehrke, and Giles Hooker. 2013. Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD - KDD '13*. ACM Press, Chicago, Illinois, USA, 623. https://doi.org/10.1145/2487575.2487579

[21] Scott M. Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M. Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. 2020. From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence* 2, 1 (2020), 56–67. https://doi.org/10.1038/s42256-019-0138-9 Number: 1 Publisher: Nature Publishing Group.

[22] Scott M Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc., Long Beach, California, USA.

[23] Christoph Molnar. 2022. *Interpretable Machine Learning* (2 ed.). Lulu. com. https://christophm.github.io/interpretable-ml-book

[24] Mário Popolin Neto and Fernando V. Paulovich. 2021. Explainable Matrix - Visualization for Global and Local Interpretability of Random Forest Classification Ensembles. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2021), 1427–1437. https://doi.org/10.1109/TVCG.2020.3030354 Conference Name: IEEE Transactions on Visualization and Computer Graphics.

[25] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom In: An Introduction to Circuits. *Distill* (2020). https://doi.org/10.23915/distill.00024.001 https://distill.pub/2020/circuits/zoom-in.

[26] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. 2016. Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples. https://doi.org/10.48550/ARXIV.1605.07277

[27] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. 2017. Practical Black-Box Attacks against Machine Learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. ACM, Abu Dhabi United Arab Emirates, 506–519. https://doi.org/10.1145/3052973.3053009

[28] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD - KDD 2016*. ACM, San Francisco California USA, 1135–1144. https://doi.org/10.1145/2939672.2939778

[29] Omer Sagi and Lior Rokach. 2018. Ensemble learning: A survey. *WIREs Data Mining and Knowledge Discovery* 8, 4 (2018), e1249. https://doi.org/10.1002/widm.1249

[30] Daniel Servén, Charlie Brummitt, Hassan Abedi, and hlink. 2018. *dswah/pyGAM: v0.8.0.* https://doi.org/10.5281/zenodo.1476122

[31] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. arXiv:1312.6034 [cs.CV]

[32] Sarah Tan, Matvey Soloviev, Giles Hooker, and Martin T. Wells. 2020. Tree Space Prototypes: Another Look at Making Tree Ensembles Interpretable. In *Proceedings of the 2020 ACM-IMS on Foundations of Data Science Conference* (Virtual Event, USA) *(FODS '20)*. Association for Computing Machinery, New York, NY, USA, 23–34. https://doi.org/10.1145/3412815.3416893

[33] Simon N Wood. 2006. *Generalized additive models: an introduction with R*. chapman and hall/CRC.

[34] Ying Yang, Geoffrey I. Webb, and Xindong Wu. 2005. *Discretization Methods*. Springer US, Boston, MA, 113–130. https://doi.org/10.1007/0-387-25465-X_6

[35] Xun Zhao, Yanhong Wu, Dik Lun Lee, and Weiwei Cui. 2019. iForest: Interpreting Random Forests via Visual Analytics. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019), 407–416. https://doi.org/10.1109/TVCG.2018.2864475 46 citations (Crossref) [2022-01-26] Conference Name: IEEE Transactions on Visualization and Computer Graphics.

[36] Honglei Zhuang, Xuanhui Wang, Michael Bendersky, Alexander Grushetsky, Yonghui Wu, Petr Mitrichev, Ethan Sterling, Nathan Bell, Walker Ravina, and Hai Qian. 2021. Interpretable Ranking with Generalized Additive Models. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining (WSDM '21)*. Association for Computing Machinery, New York, NY, USA, 499–507. https://doi.org/10.1145/3437963.3441796