Data Mining and Knowledge Discovery
https://doi.org/10.1007/s10618-022-00901-9

Model-agnostic feature importance and effects with dependent features: a conditional subgroup approach

Christoph Molnar[1,4] · **Gunnar König**[1,2,3] · **Bernd Bischl**[1,3] · **Giuseppe Casalicchio**[1,3]

Received: 14 June 2021 / Accepted: 19 November 2022

Abstract
The interpretation of feature importance in machine learning models is challenging when features are dependent. Permutation feature importance (PFI) ignores such dependencies, which can cause misleading interpretations due to extrapolation. A possible remedy is more advanced conditional PFI approaches that enable the assessment of feature importance conditional on all other features. Due to this shift in perspective and in order to enable correct interpretations, it is beneficial if the conditioning is transparent and comprehensible. In this paper, we propose a new sampling mechanism for the conditional distribution based on permutations in conditional subgroups. As these subgroups are constructed using tree-based methods such as transformation trees, the conditioning becomes inherently interpretable. This not only provides a simple and effective estimator of conditional PFI, but also local PFI estimates within the subgroups. In addition, we apply the conditional subgroups approach to partial dependence plots, a popular method for describing feature effects that can also suffer from extrapolation when features are dependent and interactions are present in the model. In simulations and a real-world application, we demonstrate the advantages of the conditional subgroup approach over existing methods: It allows to compute conditional PFI that is more true to the data than existing proposals and enables a fine-grained interpretation of feature effects and importance within the conditional subgroups.

Responsible editor: Martin Atzmueller, Johannes Fürnkranz, Tomáš Kliegr and Ute Schmid

✉ Giuseppe Casalicchio
giuseppe.casalicchio@stat.uni-muenchen.de

1 Department of Statistics, Ludwig-Maximilians-University Munich, Munich, Germany

2 Research Group Neuroinformatics, University of Vienna, Vienna, Austria

3 Munich Center for Machine Learning (MCML), Munich, Germany

4 Leibniz Institute for Prevention Research and Epidemiology - BIPS GmbH, Bremen, Germany

Published online: 10 January 2023

Keywords Interpretable machine learning · Explainable AI · Permutation feature importance · Partial dependence plot

1 Introduction

A promising avenue of research suggests to make inference about the data generating process by analyzing machine learning models using Interpretable Machine Learning (IML). The Partial Dependence Plot (PDP) (Friedman et al. 1991) and Permutation Feature Importance (PFI) (Breiman 2001) are model-agnostic tools (working for all kinds of machine learning models) that have been used for scientific discoveries. Applications range from medicine (Boulesteix et al. 2020; Stiglic et al. 2020; Pintelas et al. 2020) and the social sciences (Stachl et al. 2020; Zhao et al. 2020) to ecology (Bair et al. 2013; Esselman et al. 2015; Obringer and Nateghi 2018). PDP and PFI are used to study effect and importance of features: The PDP visualizes how a change in a feature, on average, changes the predicted outcome; the PFI ranks the features based on how much they contribute to the model performance.

Both PDP and PFI rely on marginal sampling of feature values. A range of work argues that marginal-sampling based interpretation techniques, including PDP and PFI, are not suitable for learning about the data generating process (Hooker and Mentch 2019; Frye et al. 2020; Chen et al. 2020; Freiesleben et al. 2022). The reason is that marginal-sampling based techniques ignore dependencies between the features and as a consequence may explain the model's behaviour in unlikely or even unrealistic regions of the feature space.

As a solution, conditional-sampling based techniques, such as conditional permutation feature importance (cPFI) and conditional partial dependence plots (cPDP) were proposed which only evaluate the model within the joint distribution (Strobl et al. 2008; Apley and Zhu 2016; Hooker and Mentch 2019). Given loss-optimal models, they allow insights into the data generating process. More specifically, cPFI allows to quantify whether knowing a feature is required to achieve the same predictive performance, such that nonzero cPFI can be linked with conditional dependence in the data (König et al. 2020). cPDPs visualize the relationships in the data (through the model's perspective), i.e. they describe how the conditional expectation of the outcome varies with the feature of interest (Freiesleben et al. 2022).

Although theoretically appealing, conditional-sampling based methods are more difficult to apply than marginal-sampling based methods. Existing proposals for cPFI require sampling from the conditional distribution of the feature of interest given the remaining features, which is challenging. The estimation of cPDP is especially challenging, since sampling from the multivariate conditional of the remaining features given the feature of interest is required.

Contributions: Instead of modeling the conditional distribution, we suggest to learn a tree-based partitionioning of the feature space into blocks within which the feature of interest is not (or at least less) correlated with the remaining features. This partitioning can be leveraged in several ways to derive interpretations that allow interesting insight. First of all, we can compute the well-established global cPFI by computing the PFI for each subgroup and aggregating the result. Leveraging the flexibility of tree-based

learners, this approach allows the computation of cPFI for mixed continuous and categorical data. Secondly, in situations where the partitioning requires only a few splits, the partitioning itself is interpretable. We can then leverage the partitioning to (a) get insight into the dependence structure in the data and (b) derive subgroup specific versions of PFI and PDP, to also understand under which circumstances variables are relevant or have a certain effect. For instance, by applying PFI in each subgroup, we find that temperature is not predictive of bike rentals given that we know it's summer, but highly predictive if we know that it's winter. Furthermore, by looking at the PDP within each subgroup we can understand how the conditional expectation varies with temperature given that we know that it's winter.

The paper is structured as follows: We introduce our notation in Sect. 2 and discuss related work in Sect. 3. We motivate and formally introduce the conditional subgroup approach in Sect. 4. We demonstrate the usefulness of the method on benchmarks with synthetic and real data (Sect. 5) and illustrate its interpretation in a real-world application (Sect. 6).

2 Notation and background

We consider ML prediction functions $\hat{f} : \mathbb{R}^p \mapsto \mathbb{R}$, where $\hat{f}(\boldsymbol{x})$ is a model prediction and $\boldsymbol{x} \in \mathbb{R}^p$ is a p-dimensional feature vector. We use $\mathbf{x}_j \in \mathbb{R}^n$ to refer to an observed feature (vector) and X_j to refer to the j-th feature as a random variable. With $\mathbf{x}_{-j}$ we refer to the complementary feature values $\mathbf{x}_{\{1,\ldots,p\}\setminus\{j\}} \in \mathbb{R}^{n\times(p-1)}$ and with X_{-j} to the corresponding random variables. We refer to the value of the j-th feature from the i-th instance as $x_j^{(i)}$ and to the tuples $\mathcal{D} = \{\left(\mathbf{x}^{(i)}, y^{(i)}\right)\}_{i=1}^n$ as data.

The *Permutation feature importance (PFI)* (Breiman 2001; Fisher et al. 2019) is defined as the increase in loss when feature X_j is permuted:

$$PFI_j = \mathbb{E}[L(Y, \hat{f}(\tilde{X}_j, X_{-j}))] - \mathbb{E}[L(Y, \hat{f}(X_j, X_{-j}))] \tag{1}$$

The theoretical PFI for a feature X_j is the difference between the expected loss when the feature is permuted and the original loss. If the random variable $\tilde{X}_j$ has the same marginal distribution as X_j (e.g., permutation), the estimate yields the marginal PFI. If $\tilde{X}_j$ follows the conditional distribution $\tilde{X}_j \sim X_j|X_{-j}$, we speak of the conditional PFI. The PFI is estimated with the following formula:

$$\widehat{PFI}_j = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{M}\sum_{m=1}^{M}\left(\tilde{L}_m^{(i)} - L^{(i)}\right)\right) \tag{2}$$

where $L^{(i)} = L(y^{(i)}, \hat{f}(\boldsymbol{x}^{(i)}))$ is the loss for the i-th observation and $\tilde{L}_m^{(i)} = L(y^{(i)}, \hat{f}(\tilde{x}_j^{(i)}, \boldsymbol{x}_{-j}^{(i)}))$ is the loss where $x_j^{(i)}$ was replaced by the m-th sample of $\tilde{x}_j^{(i)}$. The latter refers to the i-th feature value obtained by a sample of $\mathbf{x}_j$. The sample can be repeated M-times for a more stable estimation of $\tilde{L}^{(i)}$. Numerous variations of this formulation exist. Breiman (2001) proposed the PFI for random forests, which is

computed from the out-of-bag samples of individual trees. Subsequently, Fisher et al. (2019) introduced a model-agnostic PFI version.

The marginal *partial dependence plot (PDP)* (Friedman et al. 1991) describes the average effect of the j-th feature on the prediction.

$$PDP_j(x) = \mathbb{E}[\hat{f}(x, X_{-j})] \tag{3}$$

The theoretical PDP is a marginalized version of the prediction function. All features with the exception of X_j are integrated out, and the p-dimensional prediction function becomes a 1-dimensional function, the PDP. There are two options: Integrate with respect to the marginal distribution $\P_{X_{-j}}$ or the conditional distribution $\P_{X_{-j}|X_j}$. If the expectation is conditional on X_j, $\mathbb{E}[\hat{f}(x, X_{-j})|X_j = x]$, we speak of the conditional PDP. The marginal PDP evaluated at feature value x is estimated using Monte Carlo integration.

$$\widehat{PDP}_j(x) = \frac{1}{n}\sum_{i=1}^{n} \hat{f}\left(x, \boldsymbol{x}_{-j}^{(i)}\right) \tag{4}$$

In other words, at any given position x along the range of X_j, the PDP can be estimated by taking the data, setting $X_j = x$ for all observations and averaging the results.

3 Related work

In this section, we review conditional variants of PDP and PFI and other approaches that try to avoid extrapolation.

3.1 Related work on conditional PDP

The conditional PDP (M-Plot) (Apley and Zhu 2016) averages the predictions locally on the feature grid and mixes effects of dependent features. Apley and Zhu (2016) also address the interpretation problem that conditional PDP is influenced by feature effects of correlated features. The authors proposed accumulated local effect (ALE) plots, which reduce extrapolation by accumulating the finite differences computed within intervals of the feature of interest. By definition, interpretations of ALE plots are thus only valid locally within the intervals. Furthermore, there is no straightforward approach to derive ALE plots for categorical features, since ALE requires ordered feature values. Our proposed approach can handle categorical features.

Hooker (2007) proposed a functional ANOVA decomposition with hierarchically orthogonal components, based on integration using the joint distribution of the data, which in practice is difficult to estimate.

Another PDP variant based on stratification was proposed by Parr and Wilson (2019). However, this stratified PDP describes only the data and is independent of the model.

Individual conditional expectation (ICE) curves by Goldstein et al. (2015) can be used to visualize the interactions underlying a PDP, but they also suffer from the extrapolation problem. The "conditional" in ICE refers to conditioning on individual observations and not on certain features. As a solution, Hooker and Mentch (2019) suggested to visually highlight the areas of the ICE curves in which the feature combinations are more likely.

3.2 Related work on conditional PFI

We review approaches that modify the PFI (Breiman 2001; Fisher et al. 2019) in presence of dependent features by using a conditional sampling strategy.

Strobl et al. (2008) proposed the conditional variable importance for random forests (CVIRF), which is a conditional PFI variant of Breiman (2001). CVIRF was further analyzed and extended by Debeer and Strobl (2020). Both CVIRF and our approach rely on permutations based on partitions of decision trees. However, there are fundamental differences. CVIRF is specifically developed for random forests and relies on the splits of the underlying individual trees of the random forest for the conditional sampling. In contrast, our cs-PFI approach trains decision trees for each feature using X_{-j} as features and X_j as the target. Therefore, the subgroups for each feature are constructed from their conditional distributions (conditional on the other features) in a separate step, which is decoupled from the machine learning model to be interpreted. Our cs-PFI approach is model-agnostic, independent of the target to predict and not specific to random forests.

Hooker and Mentch (2019) made a general suggestion to replace feature values by estimates of $\mathbb{E}[X_j|X_{-j}]$.

Fisher et al. (2019) suggested to use matching and imputation techniques to generate samples from the conditional distribution. If X_{-j} has few unique combinations, they suggested to group $x_j^{(i)}$ by unique $\boldsymbol{x}_{-j}^{(i)}$ combinations and permute them for these fixed groups. For discrete and low-dimensional feature spaces, they suggest non-parametric matching and weighting methods to replace X_j values. For continuous or high-dimensional data, they suggest imputing X_j with $\mathbb{E}[X_j|X_{-j}]$ and adding residuals (under the assumption of homogeneous residuals). Our approach using permutation in subgroups can be seen as a model-driven, binary weighting approach extended to continuous features.

Knockoffs (Candes et al. 2018) are random variables which are "copies" of the original features that preserve the joint distribution but are independent of the prediction target conditional on the remaining features. Knockoffs can be used to replace feature values for conditional feature importance computation. Watson and Wright (2021) developed a testing framework for PFI based on knockoff samplers such as Model-X knockoffs (Candes et al. 2018). Our approach is complementary since Watson and Wright (2021) is agnostic to the sampling strategy that is used. Others have proposed to use generative adversarial networks for generating knockoffs (Romano et al. 2019). Knockoffs are not transparent with respect to how they condition on the features, while our approach creates interpretable subgroups.

Conditional importance approaches based on model retraining have been proposed (Hooker and Mentch 2019; Lei et al. 2018; Gregorutti et al. 2017). However, retraining the model can be expensive, and answers a fundamentally different question, often related to feature selection and not based on a fixed set of features. Hence, we focus on approaches that compute conditional PFI for a fixed model without retraining.

None of the existing approaches makes the dependence structures between the features explicit. It is unclear which of the features in X_{-j} influenced the replacement of X_j the most and how. Furthermore, little attention has been paid on evaluating how well different sampling strategies address the extrapolation problem. We address this gap with an extensive data fidelity experiment on the OpenML-CC18 benchmarking suite. To the best of our knowledge, our paper is also the first to conduct experiments using ground truth for the conditional PFI. Our approach works with any type of feature, be it categorical, numerical, ordinal and so on, since we rely on decision trees to find the subgroups used for conditioning. Further we are the first to discuss the trade-off between conditional and marginal PFI and PDP in depth. The differences between the different (conditional) PDP and PFI approaches ultimately boil down to how they sample from the conditional distribution. Table 1 lists different sampling strategies of model-agnostic interpretation methods and summarizes their assumptions to preserve the joint distribution.

4 Conditional subgroups

In this section, we propose a subgroup-based approach that allows us to (1) estimate the cPFI and to (2) introduce novel subgroup-specific versions of PDP and PFI that allow novel insight into model and data.

More specifically, we suggest to leverage tree-based learners to partition the feature space into groups G_j within which X_j is independent of the remaining features X_{-j} (Sect. 4.1). Permuting observations within such groups does not lead to extrapolation, because in each group the marginal and the conditional distribution coincide. We illustrate the idea in Fig. 1.

As a consequence, we can compute the cPFI by applying the PFI in each subgroup and aggregating the results (Sect. 4.2). Furthermore, if the data allow for a human-intelligible partitioning, we can also interpret the subgroup-wise PFI and PDP to gain novel insight about the circumstances given which variables are relevant or have a certain effect on the prediction (Sects. 4.2 and 4.3).

4.1 Learning conditional subgroups

In order to learn the grouping G_j, any algorithm can be used that splits the data in X_{-j} so that the distribution of X_j becomes more homogeneous within a group and more heterogeneous between groups. We consider decision tree algorithms for this task, which predict X_j based on splits in X_{-j}. Decision tree algorithms directly or indirectly optimize splits for heterogeneity of some aspects of the distribution of X_j in the splits. The partitions in a decision tree can be described by decision rules that lead to

Table 1 Sampling strategies for model-agnostic interpretation techniques

Sampling strategy	Used/suggested by	Assumptions
No intervention on X_j	Drop-and-Refit, LOCO (Lei et al. 2018)	
Permute X_j	Marginal PFI (Breiman 2001; Fisher et al. 2019), PDP (Friedman et al. 1991)	$X_j \perp\!\!\!\perp X_{-j}$
Replace X_j by knockoff Z_j with $(Z_j, X_{-j}) \sim (X_j, X_{-j})$ and $Z_j \perp\!\!\!\perp Y$	Knockoffs (Candes et al. 2018), CPI (Watson and Wright 2021)	$(X_j, X_{-j}) \sim N$
Move each $x_j^{(i)}$ to left and right interval bounds	ALE (Apley and Zhu 2016)	$X_j \perp\!\!\!\perp X_{-j}$ in intervals
Permute X_j in subgroups	cs-PFI, cs-PDP	$X_j \perp\!\!\!\perp X_{-j}$ in subgroups
Permute X_j in random forest tree nodes	CVIRF (Strobl et al. 2008; Debeer and Strobl 2020)	$X_j \perp\!\!\!\perp X_{-j}$ cond. on tree splits in X_{-j} to predict Y
Impute X_j from X_{-j}	(Fisher et al. 2019)	Homogeneous residuals

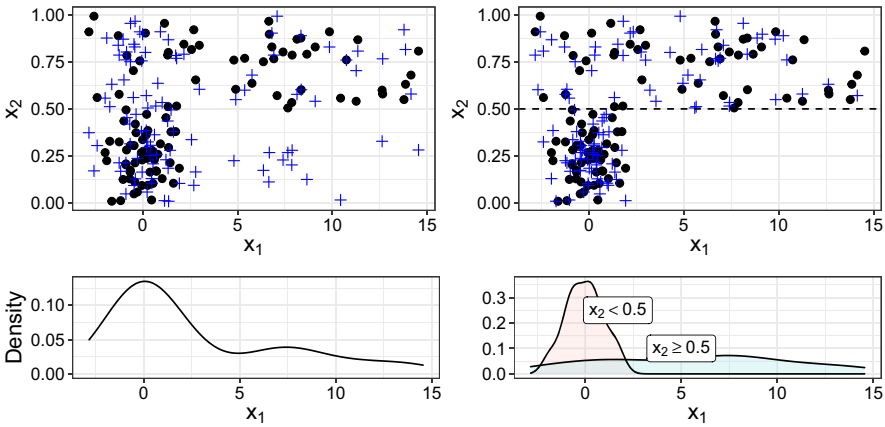

Fig. 1 Features $X_2 \sim U(0, 1)$ and $X_1 \sim N(0, 1)$, if $X_2 < 0.5$, else $X_1 \sim N(4, 4)$ (black dots). Top left: The crosses are permutations of X_1. For $X_2 < 0.5$, the permutation extrapolates. Bottom left: Marginal density of X_1. Top right: Permuting X_1 within subgroups based on X_2 ($X_2 < 0.5$ and $X_2 \geq 0.5$) reduces extrapolation. Bottom right: Densities of X_1 conditional on the subgroups

that terminal leaf. We leverage this partitioning to construct groups $\mathcal{G}_j^1, \ldots, \mathcal{G}_j^K$ based on random variable G_j for a specific feature X_j. The new variable can be calculated by assigning every observation the indicator of the partition that it lies in (meaning for observation i with $x_{-j}^{(i)} \in \mathcal{G}_j^k$ the group variable's value is defined as $g_j^{(i)} := k$).

Transformation trees (trtr) (Hothorn and Zeileis 2017) are able to model the conditional distribution of a variable. This approach partitions the feature space so that the distribution of the target (here X_j) within the resulting subgroups $\mathcal{G}_j^k$ is homogeneous, which means that the group-wise parameterization of the modeled distribution is independent of X_{-j}. Transformation trees directly model the target's distribution $¶(X_j \leq x) = F_Z(h(x))$, where F_Z is the chosen (cumulative) distribution function and h a monotone increasing transformation function (hence the name transformation trees). The transformation function is defined as $\mathbf{a}(y)^T\boldsymbol{\theta}$ where $\mathbf{a} : \mathbb{R} \mapsto \mathbb{R}^k$ is a basis function of polynomials or splines. The task of estimating the distribution is reduced to estimating $\boldsymbol{\theta}$, and the trees are split based on hypothesis tests for differences in $\boldsymbol{\theta}$ given X_{-j}, and therefore differences in the distribution of X_j. For more detailed explanations of transformation trees please refer to Hothorn and Zeileis (2017).

In contrast, a simpler approach would be to use *classification and regression trees (CART)* (Breiman et al. 1984), which, for regression, minimizes the variance within nodes, effectively finding partitions with different means in the distribution of X_j. However, CART's split criterion only considers differences in the expectation of the distribution of X_j given X_{-j}: $\mathbb{E}[X_j|X_{-j}]$. This means CART could only make X_j and X_{-j} independent if the distribution of X_j only depends in its expectation on X_{-j} (and if the dependence can be modeled by partitioning the data). Any differences in higher moments of the distribution of X_j such as the variance of $X_j|X_{-j}$ cannot be detected.

We evaluated both trtr, which are theoretically well equipped for splitting distributions and CART, which are established and well-studied. For the remainder of this

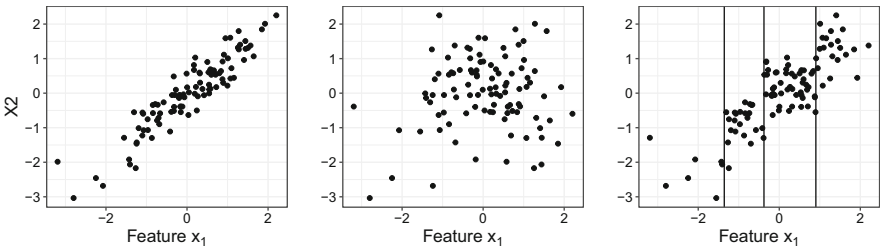

Fig. 2 Left: Simulation of features $X_1 \sim N(0, 1)$ and $X_2 \sim N(0, 1)$ with a covariance of 0.9. Middle: Unconditional permutation extrapolates strongly. Right: Permuting on partitions found by CART (predicting X_2 from X_1) has greatly reduced extrapolation, but cannot get rid of it completely. x_1 and x_2 remain correlated in the partitions

paper, we have set the default minimum number of observations in a node to 30 for both approaches. For the transformation trees, we used the Normal distribution as target distribution and we used Bernstein polynomials of degree five for the transformation function. Higher-order polynomials do not seem to increase model fit further (Hothorn 2018).

We denote the subgroups by $\mathcal{G}_j^k \subset \mathbb{R}^{p-1}$, where $k \in \{1, \ldots, K_j\}$ is the k-th subgroup for feature j, with K_j groups in total for the j-th feature. The subgroups per feature are disjoint: $\mathcal{G}_j^l \cap \mathcal{G}_j^k = \emptyset, \forall l \neq k$ and $\bigcup_{k=1}^{K} \mathcal{G}_j^k = \mathbb{R}^{p-1}$. Let $(\mathbf{y}_j^k, \mathbf{x}_j^k)$ be a subset of $(\mathbf{y}, \mathbf{x})$ that refers to the data subset belonging to the subgroup $\mathcal{G}_j^k$. Each subgroup can be described by the decision path that leads to the respective terminal node.

4.1.1 Remarks

Continuous dependencies For conditional independence $X_j \perp X_{-j} | G_j^k$ to hold, the chosen decision tree approach has to capture the (potentially complex) dependencies between X_j and X_{-j}. CART can only capture differences in the expected value of $X_j | X_{-j}$ but are insensitive to changes in, for example, the variance. Transformation trees are in principle agnostic to the specified distribution and the default transformation family of distributions is very general, as empirical results suggest (Hothorn and Zeileis 2017). However, the approach is based on the assumption that the dependence can be modeled with a discrete grouping. For example, in the case of linear Gaussian dependencies, the corresponding optimal variable would be linear Gaussian itself, and would be in conflict with our proposed interpretable grouping approach. Even in these settings the approach allows an approximation of the conditional distribution. In the case of simple linear Gaussian dependencies, partitioning the feature space will still *reduce extrapolation*. But we never get rid of it completely, unless there are only individual data points left in each partition, see Fig. 2.

Sparse subgroups Fewer subgroups are generally desirable for two reasons: (1) for a good approximation of the marginal distribution within a subgroup, a sufficient number of observations per group is required, which might lead to fewer subgroups, and (2) a large number of subgroups leads to more complex groups, which reduces

their human-intelligibility and therefore forfeits the added value of the local, subgroup-wise interpretations. As we rely on decision trees, we can adjust the granularity of the grouping using hyperparameters such as the maximum tree depth. By controlling the maximum tree depth, we can control the trade-off between the depth of the tree (and hence its interpretability) and the homogeneity of the distribution within the subgroups.

4.2 Conditional subgroup permutation feature importance (cs-PFI)

We estimate the cs-PFI of feature X_j within a subgroup $\mathcal{G}_j^k$ as:

$$PFI_j^k = \frac{1}{n_k} \sum_{i:\mathbf{x}^{(i)} \in \mathcal{G}_j^k} \left(\frac{1}{M} \sum_{m=1}^{M} L\left(y^{(i)}, \hat{f}\left(\tilde{x}_{j,m}^{(i)}, \mathbf{x}_{-j}^{(i)}\right)\right) - L\left(y^{(i)}, \hat{f}\left(\mathbf{x}^{(i)}\right)\right)\right), \tag{5}$$

where $\tilde{x}_{j,m}^{(i)}$ refers to a feature value obtained from the m-th permutation of x_j within the subgroup k_j. This estimation is exactly the same as the marginal PFI [Eq. (2)], except that it only includes observations from the given subgroup. Algorithm 1 describes the estimation of the cs-PFIs for a given feature on unseen data.

Algorithm 1: Estimate cs-PFI

Input: Model f; data $\mathcal{D}_{train}$, $\mathcal{D}_{test}$; loss L; feature j; no. permutations M

1 Train tree T_j with target X_j and features X_{-j} using $\mathcal{D}_{train}$

2 Compute subgroups $\mathcal{G}_j^k$ for $\mathcal{D}_{test}$ based on terminal nodes of $T_j, k \in \{1, \ldots, K_j\}$

3 **for** $k \in \{1, \ldots, K_j\}$ **do**

4 $\quad L_{orig} := \frac{1}{n_k} \sum_{i:\boldsymbol{x}^{(i)} \in \mathcal{G}_j^k} L(y^{(i)}, \hat{f}(\boldsymbol{x}^{(i)}))$

5 $\quad$ **for** $m \in \{1, \ldots, M\}$ **do**

6 $\quad\quad$ Generate $\tilde{\boldsymbol{x}}_j^m$ by permuting feature values $\boldsymbol{x}_j$ within subgroup $\mathcal{G}_j^k$

7 $\quad\quad L_{perm}^m := \frac{1}{n_k} \sum_{i:\boldsymbol{x}^{(i)} \in \mathcal{G}_j^k} L(y^{(i)}, \hat{f}(\tilde{x}_{j,m}^{(i)}, \boldsymbol{x}_{-j}^{(i)}))$

8 $\quad$ cs-PFI$_j^k = \frac{1}{M} \sum_{m=1}^{M} L_{perm}^m - L_{orig}$

9 cs-PFI$_j = \frac{1}{n} \sum_{k=1}^{K_j} n_k PFI_j^k$

The algorithm has two outcomes: We get local importance values for feature X_j for each subgroup (cs-PFI$_j^k$; Algorithm 1, line 8) and a global conditional feature importance (cs-PFI$_j$; Algorithm 1, line 9). The latter is equivalent to the weighted average of subgroup importances regarding the number of observations within each subgroup (see proof in "Appendix Appendix A)".

$$\text{cs-PFI}_j = \frac{1}{n} \sum_{k=1}^{K_j} n_k PFI_j^k$$

The cs-PFIs needs the same amount of model evaluations as the PFI ($O(nM)$). On top of that comes the cost for training the respective decision trees and making predictions to assign a subgroup to each observation.

Theorem 1 *When feature X_j is independent of features X_{-j} for a given dataset $\mathcal{D}$, each cs-PFI$_j^k$ has the same expectation as the marginal PFI, and an n/n_k-times larger variance, where n and n_k are the number of observations in the data and the subgroup $\mathcal{G}_j^k$.*

The proof of Theorem 1 is shown in "Appendix Appendix B". Theorem 1 has the practical implication that even in the case of applying cs-PFI to an independent feature, we will retrieve the marginal PFI, and not introduce any problematic interpretations. Equivalence in expectation and higher variance under the independence of X_j and X_{-j} holds true even if the partitions $\mathcal{G}_j^k$ would be randomly chosen. Theorem 1 has further consequences regarding overfitting: Assuming a node has already reached independence between X_j and X_{-j}, then further splitting the tree based on noise will not change the expected cs-PFIs.

4.3 Conditional subgroup partial dependence plots (cs-PDPs)

A range of work argues that PDPs are not suitable for inference if features are dependent (Hooker and Mentch 2019; Freiesleben et al. 2022). Conditional PDPs have been suggested as an alternative, but they are difficult to estimate, since they require sampling from the multivariate conditional of the remaining feature $P(X_{-j}|X_j)$. For settings where a human-intelligible partitioning can be learned, we suggest an alternative that does not require to sample from $P(X_{-j}|X_j)$: Instead of computing the global cPDP, we suggest to compute the cs-PDP$_j^k$ for each subgroup $\mathcal{G}_j^k$ using the marginal PDP formula in Eq. (4).

$$\text{cs-PDP}_j^k(x) = \frac{1}{n_k} \sum_{i:\mathbf{x}^{(i)} \in \mathcal{G}_j^k} \hat{f}\left(x, \mathbf{x}_{-j}^{(i)}\right)$$

This results in multiple cs-PDPs per feature, which can be displayed together in the same plot as in Fig. 9. The cs-PDPs allow interesting insight into data and model. First of all, since they do not extrapolate, they allow interesting insight into the data: They describe how prediction and feature of interest covary within specific groups. Secondly, in contrast to the global cPDP, they allow interesting insight into the model: For the global cPDP even features that are not used by the model can have nonzero effects (as illustrated in Fig. 3). Our proposed cs-PDPs only show nonzero effects if the respective variable is causal for the prediction.

4.3.1 Plotting the cs-PDP

The cs-PDP can be plotted in the same way as the PDP, with the exception that we get mutiple effect curves instead of just one. For a more compact view, we propose to

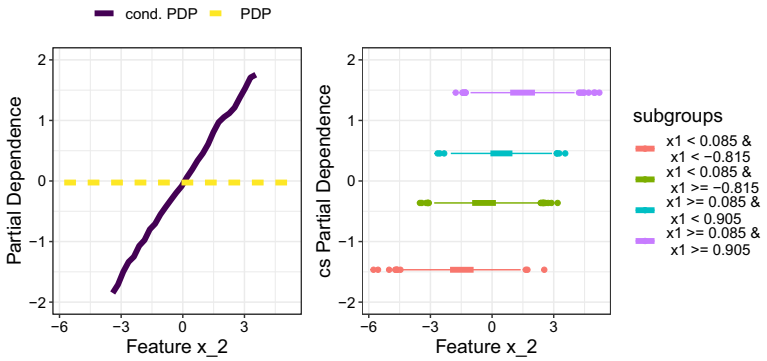

Fig. 3 We simulated a linear model of $y = x_1 + \epsilon$ with $\epsilon \sim N(0, 1)$ and an additional feature X_2 which is correlated with $X1$ (≈ 0.72). The conditional PDP (left) gives the false impression that X_2 has an influence on the target. The cs-PDPs help in this regard, as the effects due to X_1 (changes in intercept) are clearly separated from the effect that X_2 has on the target (slope of the cs-PDPs), which is zero. Unlike the marginal PDP, the cs-PDPs reveals that for increasing X_2 we expect that the prediction increases due to the correlation between X_1 and X_2

plot all cs-PDPs into the same plot. In addition, we suggest to plot the PDPs similar to boxplots, where the dense center quartiles are indicated with a bold line (see Fig. 4). By emphasizing the data density within the subgroups, the user can immediately see where to trust the plot more and where less. We restrict each cs-PDP$_j^k$ to the interval $[min(\boldsymbol{x}_j), max(\boldsymbol{x}_j)]$, with $\boldsymbol{x}_j = (x_j^{(1)}, \ldots, x_j^{(n_j^k)})$.

Equivalently to PFI, the subgroup PDPs approximate the true marginal PDP even if the features are independent.

Theorem 2 *When feature X_j is independent of features X_{-j} for a given dataset $\mathcal{D}$, each cs-PDP$_j^k$ has the same expectation as the marginal PDP, and an n/n_k-times larger variance, where n and n_k are the number of observations in the data and the subgroup $\mathcal{G}_j^k$.*

The proof of Theorem 2 is shown in "Appendix Appendix C". Theorem 2 has the same practical implications as Theorem 1: Even if the features are independent, we will, in expectation, get the marginal PDPs. And when trees are grown deeper than needed, in expectation the cs-PDPs will yield the same curve.

Both the PDP and the set of cs-PDPs need $O(nM)$ evaluations, since $\sum_{k=1}^{K_j} n_k = n$ (and worst case $O(n^2)$ if evaluated at each $x_j^{(i)}$ value). Again, there is an additional cost for training the respective decision trees and making predictions.

5 Experiments

Since for real data sets there are no ground truth values for cPFI and cPDP available, we targeted a diverse set of metrics in our experiments:

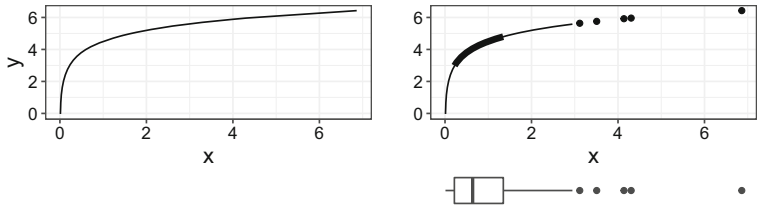

Fig. 4 Left: Marginal PDP. Bottom right: Boxplot showing the distribution of feature X. Top right: PDP with boxplot-like emphasis. In the x-range, the PDP is drawn from $\pm 1.58 \cdot IQR/\sqrt{n}$, where IQR is the range between the 25% and 75% quantile. If this range exceeds $[min(x_j), max(x_j)]$, the PDP is capped. Outliers are drawn as points. The PDP is bold between the 25% and 75% quantiles

– Conditional PFI Ground Truth Simulation: With this simulated experiment, we compared various cPFI methods. Since the data were simulated, we could compute the ground truth cPFI and benchmark all methods accordingly.
– Data fidelity evaluation: This experiment used real data sets to analyze how well the different perturbation methods that underpin the various cPDP/cPFI approaches avoid extrapolation.
– Model fidelity: This experiment evaluates how close the cPDP curves are to the real model predictions.

5.1 Training conditional sampling approaches

To ensure that sampling approaches are not overfitting, we suggest to separate training and sampling, where training covers all estimation steps that involve data. For this purpose, we refer to the training data with $\mathcal{D}_{train}$ and to the data for importance computation with $\mathcal{D}_{test}$. This section both describes how we compared the sampling approaches in the following chapters and serves as a general recommendation for how to use the sampling approaches.

For our cs-permutation, we trained the CART / transformation trees on $\mathcal{D}_{train}$ and permuted X_j of $\mathcal{D}_{test}$ within the terminal nodes of the tree. For CVIRF (Strobl et al. 2008; Debeer and Strobl 2020), which is specific to random forests, we trained the random forest on $\mathcal{D}_{train}$ to predict the target y and permuted X_j of $\mathcal{D}_{test}$ within the terminal nodes. For Model-X knockoffs (Candes et al. 2018), we fitted the second-order knockoffs on $\mathcal{D}_{train}$ and replaced X_j in $\mathcal{D}_{test}$ with its knockoffs. For the imputation approach (Fisher et al. 2019), we trained a random forest on $\mathcal{D}_{train}$ to predict X_j from X_{-j}, and replaced values of X_j in $\mathcal{D}_{test}$ with their random forest predictions plus a random residual. For the interval-based sampling (Apley and Zhu 2016), we computed quantiles of X_j using $\mathcal{D}_{train}$ and perturbed X_j in $\mathcal{D}_{test}$ by moving each observation once to the left and once to the right border of the respective intervals. The marginal permutation (PFI, PDP) required no training, we permuted (i.e., shuffled) the feature X_j in $\mathcal{D}_{test}$.

5.2 Conditional PFI ground truth simulation

We compared our cs-PFI approach using CART (tree cart) and transformation trees (tree trtr), CVIRF (Strobl et al. 2008; Debeer and Strobl 2020), Model-X knockoffs (ko) (Candes et al. 2018) and the imputation approach (impute rf) (Fisher et al. 2019) in ground truth simulations. We simulated the following data generating process: $y^{(i)} = f(\mathbf{x}^{(i)}) = \mathbf{x}_1^{(i)} \cdot \mathbf{x}_2^{(i)} + \sum_{j=1}^{10} x_j^{(i)} + \epsilon^{(i)}$, where $\epsilon^{(i)} \sim N(0, \sigma_\epsilon)$. All features, except feature X_1 followed a Gaussian distribution: $X_j \sim N(0, 1)$. Feature X_1 was simulated as a function of the other features plus noise: $x_1^{(i)} = h(x_{-1}^{(i)}) + \epsilon_x$. We simulated the following scenarios by changing h and ϵ_x:

- In the *independent* scenario, X_1 did not depend on any feature: $h(\mathbf{x}_{-1}^{(i)}) = 0$, $\epsilon_x \sim N(0, 1)$. This scenario served as a test how the different conditional PFI approaches handle the edge case of independence.
- The *linear* scenario introduces a strong correlation of X_1 with feature X_2: $h(\mathbf{x}_{-1}^{(i)}) = \mathbf{x}_2^{(i)}$, $\epsilon_x \sim N(0, 1)$.
- In the *non-linear* scenario, we simulated X_1 as a non-linear function of multiple features: $h(\mathbf{x}_{-1}^{(i)}) = 3 \cdot \mathbb{1}(\mathbf{x}_2^{(i)} > 0) - 3 \cdot \mathbb{1}(\mathbf{x}_2^{(i)} \leq 0) \cdot \mathbb{1}(\mathbf{x}_3^{(i)} > 0)$. Here also the variance of $\epsilon_x \sim N(0, \sigma_x)$ is a function of x: $\sigma_x(\mathbf{x}^{(i)}) = \mathbb{1}(\mathbf{x}_2^{(i)} > 0) + 2 \cdot \mathbb{1}(\mathbf{x}^{(i)} \leq 0) \cdot \mathbb{1}(\mathbf{x}_3^{(i)} > 0) + 5 \cdot \mathbb{1}(\mathbf{x}_2^{(i)} \leq 0) \cdot \mathbb{1}(\mathbf{x}_3^{(i)} \leq 0)$.
- For the *multiple linear dependencies* scenario, we chose X_1 to depend on many features: $h(\mathbf{x}_{-1}^{(i)}) = \sum_{j=2}^{10} x_j^{(i)}$, $\epsilon_x \sim N(0, 5)$.

For each scenario, we varied the number of sampled data points $n \in \{300, 3000\}$ and the number of features $p \in \{9, 90\}$. To "train" each of the cPFI methods, we used $2/3 \cdot n$ (200 or 2000) data points and the rest (100/1000) to compute the cPFI. The experiment was repeated 1000 times. We examined two settings.

- In setting (I), we assumed that the model recovered the true model $\hat{f} = f$.
- In setting (II), we trained a random forest with 100 trees (Breiman 2001).

In both settings, the true conditional distribution of X_1 given the remaining features is known (function h and error distribution is known). Therefore we can compute the ground truth conditional PFI, as defined in Eq. (2), by replacing $\hat{f}$ with f. We generated the samples of X_1 according to g to get the $\tilde{X}_1$ values and compute the increase in loss. The conditonal PFIs differed in settings (I) and (II) since in (I) we used the true f, and in (II) the trained random forest $\hat{f}$.

5.2.1 Conditional PFI ground truth results

For setting (I), the mean squared errors between the estimated conditional PFIs and the ground truth are displayed in Table 2, and the distributions of conditional PFI estimates in Fig. 5. In the *independent scenario*, where conditional and marginal PFI are equal, all methods performed equally well, except in the low n, high p scenario, where the knockoffs sometimes failed. As expected, the variance was higher for all methods when $n = 300$. In the *linear scenario*, the marginal PFI was clearly different from the conditional PFI. There was no clear best performing conditional PFI approach, as

the results differ depending on training size n and number of features p. For low n and low p, knockoffs performed best. For high p, regardless of n, the cs-permutation approaches worked best, which might be due to the feature selection mechanism inherent to trees. The *multiple linear dependencies scenario* was the only scenario in which the cs-PFI approach was consistently outperformed by the other methods. Decision trees already need multiple splits for recovering linear relationships, and in this scenario, multiple linear relationships had to be recovered. Imputation with random forest worked well when multiple linear dependencies are present. For knockoffs, the results were mixed. As expected, the cs-PFI approach worked well in the *non-linear scenario*, and outperformed all other approaches. Knockoffs and imputation with random forests both overestimated the conditional PFI (except for knockoffs for $n = 300$ and $p = 90$). In addition to this bias, they had a larger variance compared to the cs-PFI approaches.

Generally, the transformation trees performed equal to or outperformed CART across all scenarios, except for the multiple linear dependencies scenario. Our cs-PFI approaches worked well in all scenarios, except when multiple (linear) dependencies were present. Even for a single linear dependence, the cs-PFI approaches were on par with knockoffs and imputation, and clearly outperformed both when the relationship was more complex.

In setting (II), a random forest was analyzed, which allowed us to include the conditional variable importance for random forests (CVIRF) by Strobl et al. (2008) and Debeer and Strobl (2020) in the benchmark. The MSEs are displayed in "Appendix Appendix D", Table 6, and the distribution of conditional PFI estimates in "Appendix Appendix D" in Fig. 11. The results for all other approaches are comparable to setting (I). For the low n settings, CVIRF worked as well as the other approaches in the *independent scenario*. It outperformed the other approaches in the *linear scenario* and the *multiple linear scenario* (when n was small). The CVIRF approach consistently underestimated the conditional PFI in all scenarios with high n, even in the *independent scenario*. Therefore, we would recommend to analyze the conditional PFI for random forests using cs-PFI for lower dimensional dependence structures, and imputation for multiple (linear) dependencies.

5.3 Trading interpretability for accuracy

In an additional experiment, we examined the trade-off between the depth of the trees and the accuracy with which we recover the true conditional PFI. For scenario (I), we trained decision trees with different maximal depths (from 1 to 10) and analyzed how the resulting number of subgroups influenced the conditional PFI estimate. The experiment was repeated 1000 times. The deeper the trees, the better the true conditional PFI was approximated. Also no overfitting occured, which is in line with theoretical considerations in Theorem 1. See "Appendix Appendix E" for detailed results.

Table 2 MSE comparing estimated and true conditional PFI (scenario I)

Setting	cs-PFI (cart)	cs-PFI (trtr)	impute rf	ko	mPFI
Independent					
n = 300, p = 10	1.33	1.35	1.67	1.47	1.39
n = 300, p = 90	1.50	1.29	1.46	5.81	1.31
n = 3000, p = 10	0.14	0.15	0.16	0.13	0.15
n = 3000, p = 90	0.15	0.14	0.14	0.18	0.13
Linear					
n = 300, p = 10	4.62	4.30	3.64	2.03	44.83
n = 300, p = 90	5.55	5.26	17.53	11.63	45.36
n = 3000, p = 10	0.40	0.26	0.26	0.63	37.40
n = 3000, p = 90	0.45	0.31	3.55	0.38	36.32
Multi. lin.					
n = 300, p = 10	2443.67	2623.54	1276.41	1583.69	2739.83
n = 300, p = 90	2574.54	2896.47	2141.01	6607.73	2988.68
n = 3000, p = 10	1031.83	900.68	140.98	810.78	1548.37
n = 3000, p = 90	1075.95	1041.10	438.25	185.13	1599.59
Non-linear					
n = 300, p = 10	22.00	17.76	265.73	668.34	1204.17
n = 300, p = 90	19.99	19.81	504.53	131.77	1248.74
n = 3000, p = 10	1.18	1.00	144.77	626.80	1156.32
n = 3000, p = 90	1.17	1.13	206.01	579.02	1136.83

Impute rf: Imputation with a random forest, ko: Model-X knockoffs, mPFI: (marginal) PFI, tree cart: cs-permutation based on CART, tree trtr: cs-permutation based on transformation trees

5.4 Data fidelity evaluation

PDP and PFI work by data intervention, prediction, and subsequent aggregation (Scholbeck et al. 2019). Based on data $\mathcal{D}$, the intervention creates a new data set. In order to compare different conditional sampling approaches, we define a measure of data fidelity to quantify the ability to preserve the joint distribution under intervention. Failing to preserve the joint distribution leads to extrapolation when features are dependent. Model-X knockoffs, for example, are directly motivated by preserving the joint distribution, while others, such as accumulated local effect plots do so more implicitly.

Data fidelity is the degree to which a sample $\tilde{X}_j$ of feature X_j preserves the joint distribution, that is, the degree to which $(\tilde{X}_j, X_{-j}) \sim (X_j, X_{-j})$ In theory, any measure that compares two multivariate distributions can be used to compute the data fidelity. In practice, however, the joint distribution is unknown, which makes measures such as the Kullback-Leibler divergence impractical. We are dealing with two samples, one data set without and one with intervention.

In this classic two-sample test-scenario, the maximum mean discrepancy (MMD) can be used to compare whether two samples come from the same distribution (Fortet

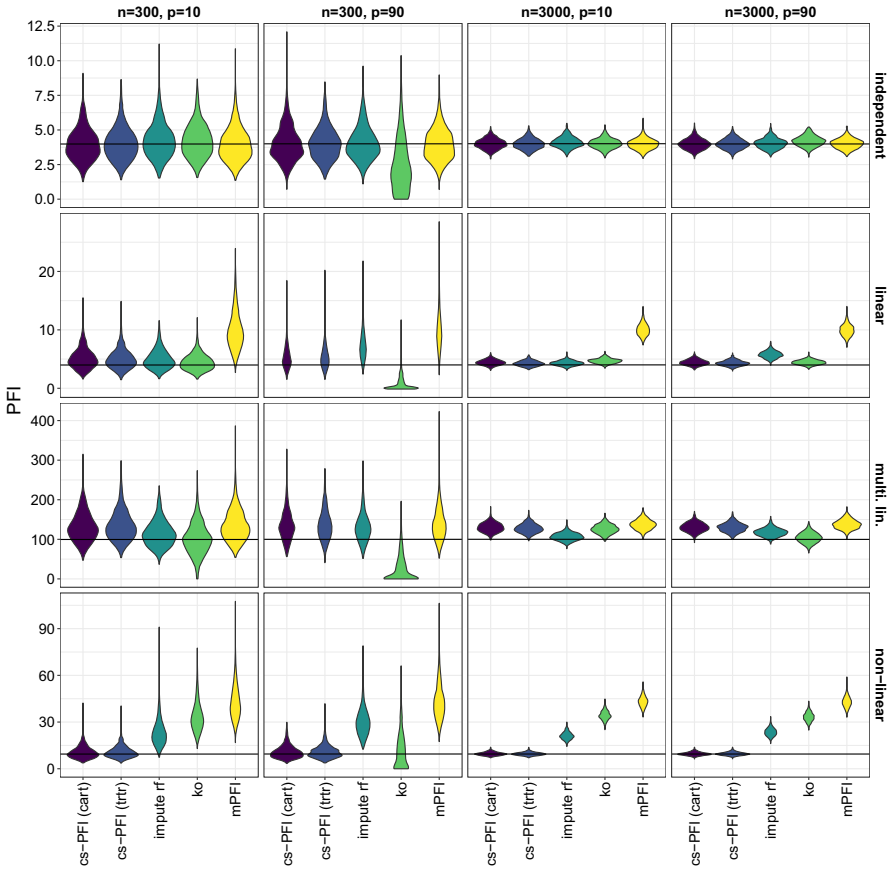

Fig. 5 Setting (I) comparing various conditional PFI approaches on the true model against the true conditional PFI (horizontal line) based on the data generating process

and Mourier 1953; Gretton et al. 2007, 2012; Smola et al. 2007). The empirical MMD is defined as:

$$\text{MMD}(\mathcal{D}, \tilde{\mathcal{D}}) = \frac{1}{n^2} \sum_{x,z \in \mathcal{D}} k(x, z) - \frac{2}{nl} \sum_{x \in \mathcal{D}, z \in \tilde{\mathcal{D}}} k(x, z) + \frac{1}{l^2} \sum_{x,z \in \tilde{D}} k(x, z) \quad (6)$$

where $\mathcal{D} = \{x_j^{(i)}, x_{-j}^{(i)}\}_{i=1}^{n}$ is the original data set and $\tilde{\mathcal{D}} = \{\tilde{x}_j^{(i)}, x_{-j}^{(i)}\}_{i=1}^{l}$ a data set with perturbed $x_j^{(i)}$. For both data sets, we scaled numerical features to a mean of zero and a standard deviation of one. For the kernel k we used the radial basis function kernel for all experiments. For parameter σ of the radial basis function kernel, we chose the median L2-distance between data points which is a common heuristic (Gretton et al. 2012). We measure data fidelity as the negative logarithm of the MMD ($-log(\text{MMD})$) to obtain a more condensed scale where larger values are better.

Definition 1 (*MMD-based Data Fidelity*) Let $\mathcal{D}$ be a dataset, and $\tilde{D}$ be another dataset from the same distribution, but with an additional intervention. We define the data fidelity as: Data Fidelity $= -log(\text{MMD}(\mathcal{D}, \tilde{\mathcal{D}}))$.

We evaluated how different sampling strategies (see Table 1) affect the data fidelity measure for numerous data sets of the OpenML-CC18 benchmarking suite (Bischl et al. 2019). We removed all data sets with 7 or fewer features and data sets with more than 500 features. See "Appendix Appendix F" for an overview of the remaining data sets. For each data set, we removed all categorical features from the analysis, as the underlying sampling strategies of ALE plots and Model-X knockoffs are not well equipped to handle them. We were foremost interested in two questions:

(A) How does cs-permutation compare with other sampling strategies w.r.t. data fidelity?
(B) How do choices of tree algorithm (CART vs. transformation trees) and tree depth parameter affect data fidelity?

In each experiment, we selected a data set, randomly sampled a feature and computed the data fidelity of various sampling strategies as described in the pseudo-code in Algorithm 2.

Algorithm 2: Data Fidelity Experiments

```
Input: OpenML-CC18 data sets, sampling strategies
for data set 𝒟 in OpenML-CC18 do
    Remove prediction target from 𝒟 (only keep it for CVIRF)
    Randomly order features in 𝒟
    for features j ∈ {1, ..., 10} do
        for repetition ∈ {1, ..., 30} do
            Sample min(10.000, n) rows from 𝒟
            Split sample into 𝒟_train (40%), 𝒟_test (30%) and 𝒟_ref (30%)
            for each sampling do
                "Train" sampling approach using 𝒟_train (e.g., construct subgroups, fit
                knockoff-generator, ...)
                Generate conditional sample X̃_j for 𝒟_test
                Estimate data fidelity as −log(MMD(𝒟_ref, 𝒟_test))
return Set of data fidelity estimates
```

For an unbiased evaluation, we split the data into three pieces: $\mathcal{D}_{train}$ (40% of rows), $\mathcal{D}_{test}$ (30% of rows) and $\mathcal{D}_{ref}$ (30% of rows). We used $\mathcal{D}_{train}$ to "train" each sampling method (e.g., train decision trees for cs-permutation, see Sect. 5.1). We used $\mathcal{D}_{ref}$, which we left unchanged and $\mathcal{D}_{test}$, for which the chosen feature was perturbed to estimate the data fidelity. For each data set, we chose 10 features at random, for which sampling was applied. Marginal permutation (which ignores the joint distribution) and "no perturbation" served as lower and upper bounds for data fidelity. For CVIRF, we only used one tree per random forest as we only compared the general perturbation strategy which is the same for each tree.

We repeated all experiments 30 times with different random seeds and therefore different data splits. All in all this produced 12,210 results (42 data sets $\times$ (up to) 10 features $\times$ 30 repetitions) per sampling method. All results are shown in detail in "Appendix Appendix F" (Figs. 13, 14, 15, and 16).

Since the experiments are repeated across the same data sets and the same features, the data fidelity results are not independent. Therefore, we used a random intercept model (Bryk and Raudenbush 1992) to analyze the differences in data fidelity between different sampling approaches. The target variable of the random intercept model was the MMD, the dependent variable was the perturbation method, and we used a random intercept per data and per feature (nested). So, informally: $MMD \sim$ perturbation method $+ (1|$ dataset/feature).

We chose "Marginal Permutation" as the reference category. We fitted two random intercept models: One to compare cs-permutation with fully-grown trees (CART, trtr) with other sampling methods and another one to compare different tree depths.

5.4.1 Results (A) state-of-the-art comparison

Figure 6 shows the effect estimates of different sampling approaches modeled with a random intercept model. The results show that cs-permutation performed better than all other methods. Model-X knockoffs and the imputation approach (with random forests) came in second place and outperformed ALE and CVIRF. Knockoffs were proposed to preserve the joint distribution, but are based on multivariate Gaussian distribution. This seems to be too restrictive for the data sets in our experiments. CVIRF does not have much higher data fidelity than marginal permutation. However, results for CVIRF must be viewed with caution, since data fidelity regards all features equally – regardless of their impact on the model prediction. For example, a feature can be highly correlated with the feature of interest, but might not be used in the random forest. A more informative experiment for comparing CVIRF can be found in Sect. 5.2. Figures 13 and 14 in "Appendix Appendix F" show the individual data fidelity results for the OpenML-CC18 data sets. Not perturbing the feature at all has the highest data fidelity and serves as the upper bound. The marginal permutation serves as a lower baseline. For most data sets, cs-permutation has a higher data fidelity compared to all other sampling approaches. For all the other methods there is at least one data set on which they reach a low data fidelity (e.g., "semeion", "qsar-biodeg" for ALE; "nodel-simulation", "churn" for imputation; "jm1", "pc1" for knockoffs). In contrast, cs-permutation achieves a consistently high data fidelity on all these data sets.

Additionally, we review the data fidelity rankings of the sampling methods in Table 3. The table shows the average ranking of each method according to MMD. First we computed the rank of each perturbation method per dataset, feature and repetition, with rank 1 being the best (lowest MMD). This allows another view on the performance of the perturbation methods. The rankings show a similar picture as the random intercept model estimates, except that Model-X knockoffs have a better average ranking than imputation. This could be the case since on a few data sets (bank-marketing, electricity, see Fig. 13 in "Appendix Appendix F") Model-X knock-

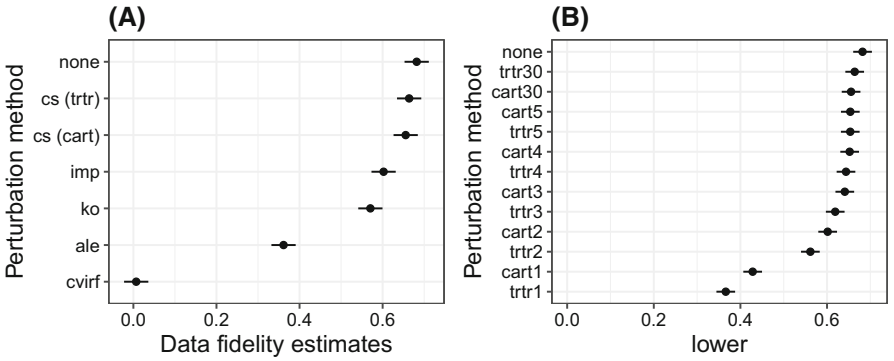

Fig. 6 Linear regression model coefficients and 95% confidence intervals for the effect of different sampling approaches on data fidelity, with (nested) random effects per data set and feature. **A** Comparing different sampling approaches. No perturbation ("none") and permutation ("perm") serve as upper and lower bounds. **B** Comparing cs-permutation using either CART or transformation trees and different tree depths (1, 2, 3, 4, 5 and 30). Marginal permutation is the reference category and therefore is at $x = 0$ and all other perturbation method estimates are relative to this reference

Table 3 Mean ranks and their standard deviation based on data fidelity of various perturbation methods over data sets, features and repetitions

	None	cs (trtr)	ko	cs (cart)	imp	ale	perm	cvirf
Mean ranks	2.50	3.51	3.70	3.76	4.25	4.61	6.82	6.84
SD	0.73	0.87	1.32	0.91	1.37	2.07	1.14	1.14

None: No intervention, which serves as upper benchmark. cart30: cs-permutation with CART with maximal depth of 30. trtr30: cs-permutation with transformation trees with maximal depth of 30. imp: Imputation approach. ko: Model-X knockoffs (Candes et al. 2018). ale: ALE perturbation (Apley and Zhu 2016). cvirf: Conditional variable importance for random forests (Strobl et al. 2008). perm: Unconditional permutation

offs have a very low data fidelity but on most others a higher model fidelity than the imputation method.

5.4.2 Results (B) tree configuration

We included shallow trees with maximum depth parameter from 1 to 5 to analyze the trade-off between tree depth and data fidelity. We included trees with a maximum depth parameter of 30 ("fully-grown" trees as this was the software's limit) as an upper bound for each decision tree algorithm. Figure 6B) shows that the deeper the trees (and the more subgroups), the higher the data fidelity. This is to be expected, since deeper trees allow for a more fine-grained separation of distributions. More importantly, we are interested in the trade-off between depth and data fidelity. Even splitting with a maximum depth of only 1 (two subgroups) strongly improves data fidelity over the simple marginal permutation for most data sets. A maximum depth of two means another huge average improvement in data fidelity, and already puts cs-permutation on par with knockoffs. A depth of three to four is almost as good as a maximum depth parameter of 30 and already outperforms all other methods, while still

Table 4 We selected data sets from OpenML Vanschoren et al. (2014) and Casalicchio et al. (2017) having 1000–8000 instances and a maximum of 50 numerical features

	wine	satellite	wind	space	pollen	quake
No. of rows	6497	6435	6574	3107	3848	2178
No. of features	12	37	15	7	6	4

We excluded data sets with categorical features, since ALE cannot handle them

being interpretable due to their shortness. CART slightly outperforms transformation trees clearly when trees are shallow, which is surprising since transformation trees are, in theory, better equipped to handle changes in the distribution. Deeply grown transformation trees (max. depth of 30) slightly outperform CART. Figures 15 and 16 in "Appendix Appendix F" show data fidelity aggregated by data set.

5.5 Model fidelity

Model fidelity has been defined as how well the predictions of an explanation method approximate the ML model (Ribeiro et al. 2016). Similar to Szepannek (2019), we define model fidelity for feature effects as the mean squared error between model prediction and the prediction of the partial function f_j (which depends only on feature X_j) defined by the feature effect method, for example $f_j(x) = PDP_j(x)$. For a given data instance with observed feature value $x_j^{(i)}$, the predicted outcome of, for example, a PDP can be obtained by the value on the y-axis of the PDP at the observed x_j value.

$$\text{Model\_Fidelity}(\hat{f}, f_j) = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{f}\left(x^{(i)}\right) - f_j\left(x_j^{(i)}\right)\right)^2, \tag{7}$$

where f_j is a feature effect function such as ALE or PDP. For this definition of model fidelity, lower values are more desirable. The better the model fidelity, the closer the effect curve is to the actual model predictions. In order to evaluate ALE plots, they have to be adjusted such that they are on a comparable scale to a PDP (Apley and Zhu 2016): $f_j^{ALE,adj} = f_j^{ALE} + \frac{1}{n}\sum_{i=1}^{n}\hat{f}(x^{(i)})$.

We trained random forests (500 trees), linear models and k-nearest neighbours models (k = 7) on various regression data sets (Table 4). 70% of the data were used to train the ML models and the transformation trees/CARTs. This ensure that results are not over-confident due to overfitting, see also Sect. 5.1. The remaining 30% of the data were used to evaluate model fidelity. For each model and each data set, we measured model fidelity between effect prediction and model prediction [Eq. (7)], averaged across observations and features.

Table 5 shows that the model fidelity of ALE and PDP is similar, while the cs-PDPs have the best model fidelity (lower is better). This is an interesting result since the decision trees for the cs-PDPs are neither based on the model nor on the real target, but solely on the conditional dependence structure of the features. However, the cs-PDPs have the advantage that we obtain multiple plots. We did not aggregate the plots to

Table 5 Mean model fidelity averaged over features in a random forest for various data sets, and the variance across features

	Pollen	Quake	Satellite	Space	Wind	Wine
PDP	10.83 (6.33)	0.03 (0.0)	4.78 (0.03)	0.04 (0.0)	43.98 (33.91)	0.75 (0.0)
ALE	12.33 (19.68)	0.04 (0.0)	4.82 (0.01)	0.04 (0.0)	43.38 (56.71)	0.75 (0.0)
trtr1	9.09 (3.18)	0.03 (0.0)	4.19 (0.59)	0.04 (0.0)	30.36 (41.02)	0.72 (0.0)
cart1	9.06 (3.24)	0.03 (0.0)	3.75 (0.77)	0.04 (0.0)	31.22 (59.23)	0.72 (0.0)
trtr2	8.29 (5.14)	0.03 (0.0)	3.36 (0.52)	0.04 (0.0)	26.47 (50.21)	0.71 (0.0)
cart2	8.12 (6.29)	0.03 (0.0)	3.23 (0.69)	0.04 (0.0)	27.29 (78.63)	0.71 (0.0)

The cPDPs (trtr,cart) always had a lower loss (i.e. higher model fidelity) than PDP and ALE. The loss monotonically decreases with increasing maximum tree depth for subgroup construction

a single conditional PDP, but computed the model fidelity for the PDPs within the subgroups (visualized in Fig. 9). Our cs-PDPs using trees with a maximum depth of 2 have a better model fidelity than using a maximum depth of 1. We limited the analysis to interpretable conditioning and therefore allowed only trees with a maximum depth of 2, since a tree depth of 3 already means up to 8 subgroups which is already an impractical number of PDPs to have in one plot. CART sometimes beats trtr (e.g., on the "satellite" data set) but sometimes trtr has a lower loss (e.g., on the "wind" data set). Using different models (knn or linear model) produced similar results, see "Appendix Appendix G".

6 Application

In the following application, we demonstrate that cs-PDPs and cs-PFI are valuable tools to understand model and data beyond insights given by PFI, PDPs, or ALE plots. We trained a random forest to predict daily bike rentals (Dua and Graff 2017) with given weather and seasonal information. The data ($n = 731$, $p = 9$) was divided into 70% training and 30% test data. The features are not independent (see "Appendix Appendix H")

6.1 cs-PDPs and cs-PFI

To construct the subgroups, we used transformation trees with a maximum tree depth of 2 which limited the number of possible subgroups to 4. We chose transformation trees because they are theoretically more sound and don't require the assumption that the conditional distributions only differ in the means of the other features.

Figure 7 shows that for most features the biggest change in the estimated conditional PFI happens when moving from a maximum depth of 0 (=marginal PFI) to a depth of 2. This makes a maximum depth of 2 a reasonable trade-off between limiting the number of subgroups and accurately approximating the conditional PFI. We compared the marginal and conditional PFI for the bike rental predictions, see Fig. 8.

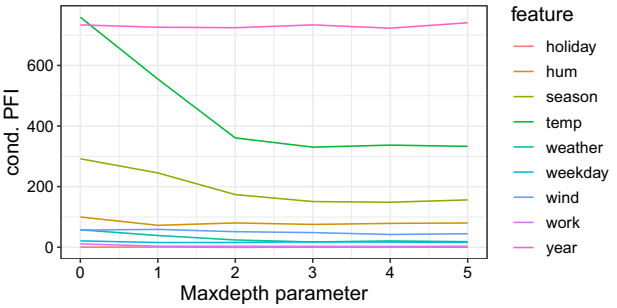

Fig. 7 Conditional feature importance by increasing maximum depth of the trees

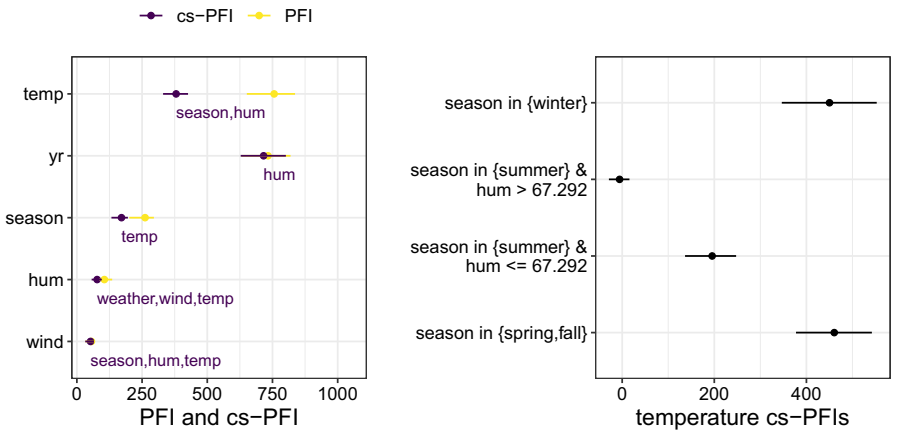

Fig. 8 Left: Comparison of PFI and cs-PFI for a selection of features. For cs-PFI we also show the features that constitute the subgroups. Right: Local cs-PFI of temperature within subgroups. The temperature feature is important in spring, fall and winter, but neglectable on summer days, especially humid ones

The most important features, according to (marginal) PFI, were temperature and year. For the year feature, the marginal and conditional PFI are the same. Temperature is less important when we condition on season and humidity. The season already holds a lot of information about the temperature, so this is not a surprise. When we know that a day is in summer, it is not as important to know the temperature to make a good prediction. On humid summer days, the PFI of temperature is zero. However, in all other cases, it is important to know the temperature to predict how many bikes will be rented on a given day. The disaggregated cs-PFI in a subgroup can be interpreted as "How important is the temperature, given we know the season and the humidity".

We compare PDP, ALE and cs-PDP in Fig. 9. Both ALE and PDP show a monotone increase of predicted bike rentals up until a temperature of 25 °C and a decrease beyond that. The PDP shows a weaker negative effect of very high temperatures which might be caused by extrapolation: High temperature days are combined with e.g. winter. A limitation of the ALE plot is that we should only interpret it locally within each interval that was used to construct the ALE plot. In contrast, our cs-PDP is explicit about the subgroup conditions in which the interpretation of the cs-PDP is valid and shows the

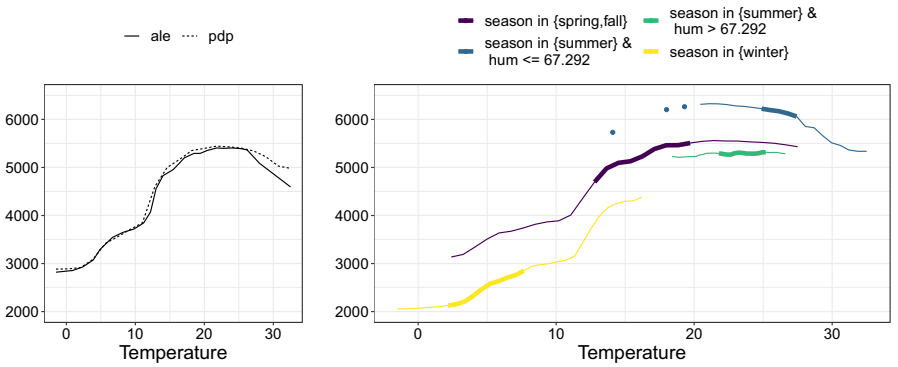

Fig. 9 Effect of temperature on predicted bike rentals. Left: PDP and ALE plot. Right: cs-PDPs for 4 subgroups

distributions in which the feature effect may be interpreted. The local cs-PDPs in subgroups reveal a more nuanced picture: For humid summer days, the temperature has no effect on the bike rentals, and the average number of rentals are below that of days with similar temperatures in spring, fall and drier summer days. The temperature has a slightly negative effect on the predicted number of bike rentals for dry summer days (humidity below 67.3). The change in intercepts of the local cs-PDP can be interpreted as the effect of the grouping feature (season). The slope can be interpreted as the temperature effect within a subgroup.

We also demonstrate the local cs-PDPs for the season, a categorical feature. Figure 10 shows both the PDP and our local cs-PDPs. The normal PDP shows that on average there is no difference between spring, summer and fall and only slightly less bike rentals in winter. The PDP with four subgroups conditional on temperature shows that the marginal PDP is misleading. The PDP indicates that in spring, summer and fall, around 4500 bikes are rented and in winter around 1000 fewer. The cs-PDPs in contrast show that, conditional on temperature, the differences between the seasons are much greater, especially for low temperatures. Only at high temperatures is the number of rented bikes similar between seasons.

7 Discussion

We proposed the cs-PFIs and cs-PDPs, wich are variants of PFI and PDP that work when features are dependent. Both cs-PFIs and cs-PDPs rely on permutations in subgroups based on decision trees. The approach is simple: Train a decision tree to predict the feature of interest and compute the (marginal) PFI/PDP in each terminal node defined by the decision tree.

Compared to other approaches, cs-PFIs and cs-PDPs enable a human comprehensible grouping, which carries information how dependencies affect feature effects and importance. As we showed in various experiments, our methods are on par or outperform other methods in many dependence settings. We therefore recommend

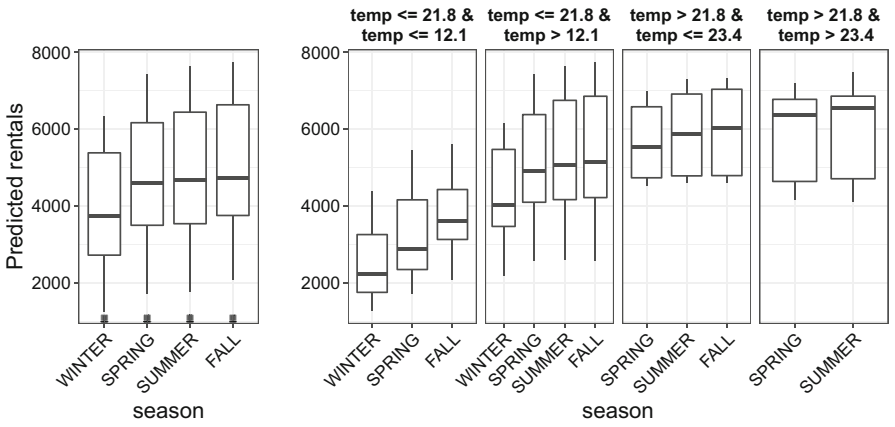

Fig. 10 Effect of season on predicted rentals. Left: PDP. Right: Local cs-PDPs. The cs-PDPs are conditioned on temperature, in which the tree split at 21.5 and at 9.5

using cs-PDPs and cs-PFIs to analyze feature effects and importances when features are dependent. However, due to their construction with decision trees, cs-PFIs and cs-PDPs do not perform well when the feature of interest depends on many other features, but only if it depends on a few features. Especially the interpretability suffers if the tree has to rely on many features. We recommend analyzing the dependence structure beforehand, using the imputation approach with random forests in the case of multiple dependencies, and cs-PFIs in all other cases.

Our framework is flexible regarding the choice of partitioning and we leave the evaluation of the rich selection of possible decision tree and decision rules approaches to future research.

Reproducibility All experiments were conducted using *mlr* (Lang et al. 2019) and R (R Core Team 2017). We used the *iml* package (Molnar et al. 2018) for ALE and PDP, *party*/*partykit* (Hothorn and Zeileis 2015) for CVIRF and *knockoff* (Patterson and Sesia 2020) for Model-X knockoffs. The code for all experiments is available at https://github.com/christophM/paper_conditional_subgroups.

Acknowledgements This project is funded by the Bavarian State Ministry of Science and the Arts, by the Bavarian Research Institute for Digital Transformation (bidt), and supported by the German Federal Ministry of Education and Research (BMBF) (Grant No. 01IS18036A) and by the German Research Foundation (DFG), Emmy Noether Grant 437611051. The authors of this work take full responsibilities for its content.

Funding Open Access funding enabled and organized by Projekt DEAL.

Appendix A Decompose conditional PFI into cs-PFIs

Assuming a perfect construction of G_j, it holds that $X_j \perp X_{-j}|G_j$ and also that $X_j \perp G_j|X_{-j}$ (as G_j is a compression of X_{-j}). Therefore

$$P(X_j|X_{-j}) = P(X_j|X_{-j}, G_j) = P(X_j|G_j). \tag{8}$$

When we sample the replacement $\tilde{x}_j^{(i)}$ for an $x_j^{(i)}$ from the marginal within a group ($P(X_j|G_j = g_j^{(i)})$, e.g., via permutation) we also sample from the conditional $P(X_j|X_{-j} = x_{-j}^{(i)})$. Every data point from the global sample can therefore equivalently be seen as a sample from the marginal within the group, or as a sample from the global conditional distribution.
As follows, the weighted sum of marginal subgroup PFIs coincides with the conditional PFI (cPFI).

$$cPFI = \sum_{i=1}^{n} \frac{1}{n} \left(L\left(f\left(\tilde{x}_j^{(i)}, x_{-j}^{(i)} \right), y^{(i)} \right) - L\left(\hat{f}\left(x_j^{(i)}, x_{-j}^{(i)} \right), y^{(i)} \right) \right) \tag{9}$$

$$= \sum_{k=1}^{K} \frac{n_k}{n} \sum_{i \in \mathcal{G}_k} \frac{1}{n_k} \left(L\left(f\left(\tilde{x}_j^{(i)}, x_{-j}^{(i)} \right), y^{(i)} \right) - L\left(\hat{f}\left(x_j^{(i)}, x_{-j}^{(i)} \right), y^{(i)} \right) \right) \tag{10}$$

$$= \sum_{k=1}^{K} \frac{n_k}{n} PFI^k \tag{11}$$

Appendix B Expectation and variance of the PFI in a subgroup

We show that under feature independence the PFI and a PFI in an arbitrary subgroup have the same expected value and the subgroup k PFI has a higher variance. Let $\tilde{L}^{(i)} = \frac{1}{M} \sum_{m=1}^{M} L(y^{(i)}, \hat{f}(\tilde{x}_{j,m}^{(i)}, x_{-j}^{(i)})$ and $L^{(i)} = L(y^{(i)}, \hat{f}(x_{j,m}^{(i)}, x_{-j}^{(i)})$.

Proof

$$\mathbb{E}_{X_{-j}}[PFI_j] = \mathbb{E}_{X_{-j}} \left[\frac{1}{n} \sum_{i=1}^{n} (\tilde{L}^{(i)} - L^{(i)}) \right]$$

$$= \mathbb{E}_{X_{-j}}[\tilde{L}^{(i)} - L^{(i)}]$$

$$\mathbb{E}[PFI_j^k]_{X_{-j}} = \mathbb{E}_{X_{-j}} \left[\frac{1}{n_k} \sum_{i:x^{(i)} \in \mathcal{G}_j^k} (\tilde{L}^{(i)} - L^{(i)}) \right]$$

$$
\begin{aligned}
&= \frac{1}{n_k}\mathbb{E}_{X_{-j}}\left[\sum_{i:x^{(i)}\in\mathcal{G}_j^k}(\tilde{L}^{(i)} - L^{(i)})\right] \\
&= \frac{1}{n_k}n_k\mathbb{E}_{X_{-j}}\left[(\tilde{L}^{(i)} - L^{(i)})\right] \\
&= \mathbb{E}_{X_{-j}}[PFI_j] \\
\mathbb{V}_{X_{-j}}\left[PFI_j\right] &= \mathbb{V}_{X_{-j}}\left[\frac{1}{n}\sum_{i=1}^{n}(\tilde{L}^{(i)} - L^{(i)})\right] \\
&= \frac{1}{n^2}n\mathbb{V}_{X_{-j}}\left[\tilde{L}^{(i)} - L^{(i)}\right] \\
&= \frac{1}{n}\mathbb{V}_{X_{-j}}\left[\tilde{L}^{(i)} - L^{(i)})\right] \\
\mathbb{V}_{X_{-j}}\left[PFI_j^k]\right] &= \mathbb{V}_{X_{-j}}\left[\frac{1}{n_k}\sum_{i=1}^{n_k}(\tilde{L}^{(i)} - L^{(i)})\right] \\
&= \frac{1}{n_k^2}n_k\mathbb{V}_{X_{-j}}\left[\tilde{L}^{(i)} - L^{(i)}\right] \\
&= \frac{1}{n_k}\mathbb{V}_{X_{-j}}\left[\tilde{L}^{(i)} - L^{(i)})\right] \\
\frac{\mathbb{V}_{X_{-j}}[PFI_j^k]}{\mathbb{V}_{X_{-j}}\left[PFI_j\right]} &= \frac{n}{n_k}
\end{aligned}
$$

□

Appendix C Expectation and variance of the PDP in a subgroup

We show that under feature independence the PDP and a PDP in an arbitrary subgroup have the same expected value and the subgroup k PDP has a higher variance.

Proof

$$
\begin{aligned}
\mathbb{E}_{X_{-j}}[PDP_j(x)] &= \mathbb{E}_{X_{-j}}\left[\hat{f}(x, X_{-j})\right] \\
\mathbb{E}_{X_{-j}}[PDP_j^k(x)] &= \mathbb{E}_{X_{-j}}\left[\frac{1}{n_k}\sum_{i=1}^{n_k}\hat{f}\left(x, x_{-j}^{(i)}\right)\right] = \frac{1}{n_k}n_k\mathbb{E}_{X_{-j}}\left[\hat{f}(x, X_{-j})\right] \\
&= \mathbb{E}_{X_{-j}}\left[\hat{f}(x, X_{-j})\right] \\
\mathbb{V}_{X_{-j}}\left[PDP_j(x)\right] &= \mathbb{V}_{X_{-j}}\left[\frac{1}{n}\sum_{i=1}^{n}\hat{f}\left(x, x_{-j}^{(i)}\right)\right] \\
&= \frac{1}{n^2}n\mathbb{V}_{X_{-j}}\left[\hat{f}(x, X_{-j})\right]
\end{aligned}
$$

$$= \frac{1}{n}\mathbb{V}_{X_{-j}}\left[\hat{f}(x, X_{-j})\right]$$

$$\mathbb{V}_{X_{-j}}\left[PDP_j^k(x)\right] = \mathbb{V}_{X_{-j}}\left[\frac{1}{n_k}\sum_{i=1}^{n_k}\hat{f}\left(x, x_{-j}^{(i)}\right)\right]$$

$$= \frac{1}{n_k^2} n_{k_j} \mathbb{V}_{X_{-j}}\left[\hat{f}(x, X_{-j})\right]$$

$$= \frac{1}{n_k}\mathbb{V}_{X_{-j}}\left[\hat{f}(x, X_{-j})\right]$$

$$\frac{\mathbb{V}_{X_{-j}}[PDP_j^k(x)]}{\mathbb{V}_{X_{-j}}\left[PDP_j(x)\right]} = \frac{n}{n_k}$$

□

Appendix D cPFI ground truth scenario II

This chapter contains the results for the conditional PFI ground truth simulation, scenario II with an intermediate random forest (Table 6 and Fig. 11).

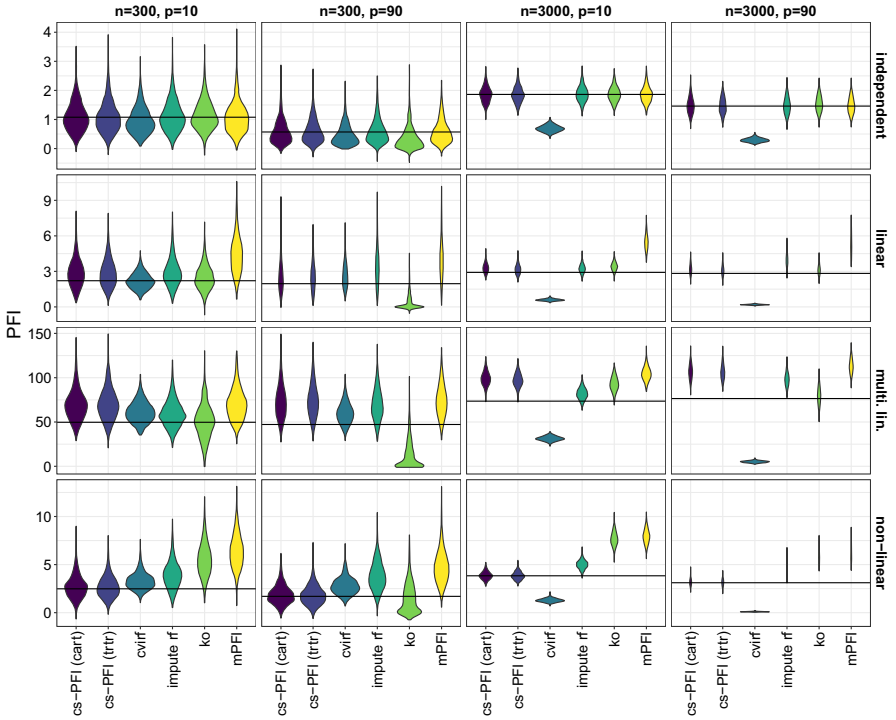

Fig. 11 Experiment (II) comparing various conditional PFI approaches with an intermediary a random forest against the true conditional PFI based on the data generating process

Table 6 MSE comparing estimated and true conditional PFI (for random forest, scenario II)

Setting	cs-PFI (cart)	cs-PFI (trtr)	cvirf	impute rf	ko	mPFI
independent						
n = 300, p = 10	0.26	0.28	0.22	0.27	0.25	0.27
n = 300, p = 90	0.19	0.17	0.14	0.18	0.19	0.17
n = 3000, p = 10	0.07	0.07	1.39	0.07	0.06	0.08
n = 3000, p = 90	0.08	0.08	1.37	0.08	0.08	0.08
linear						
n = 300, p = 10	1.79	1.69	0.45	1.87	1.10	7.11
n = 300, p = 90	1.93	1.88	1.36	4.25	2.93	7.06
n = 3000, p = 10	0.29	0.22	5.41	0.25	0.40	6.80
n = 3000, p = 90	0.32	0.24	6.98	1.66	0.26	7.02
multi. lin.						
n = 300, p = 10	667.79	744.48	275.58	335.40	377.35	726.15
n = 300, p = 90	972.42	1098.74	301.26	823.89	1473.67	1065.26
n = 3000, p = 10	715.41	625.99	1790.45	114.71	454.26	1017.53
n = 3000, p = 90	974.37	945.19	5090.09	532.44	110.94	1416.30
non-linear						
n = 300, p = 10	1.40	1.29	1.37	3.96	12.35	18.51
n = 300, p = 90	1.06	1.03	2.05	6.77	2.38	12.32
n = 3000, p = 10	0.17	0.16	6.53	1.55	15.29	17.56
n = 3000, p = 90	0.15	0.14	9.09	3.28	8.00	11.30

impute rf: Imputation with a random forest, ko: Model-X knockoffs, mPFI: (marginal) PFI, tree cart: cs-permutation based on CART, tree trtr: cs-permutation based on transformation trees, CVIRF: conditional variable importance for random forests

Appendix E cPFI ground truth tree depth

See Fig. 12.

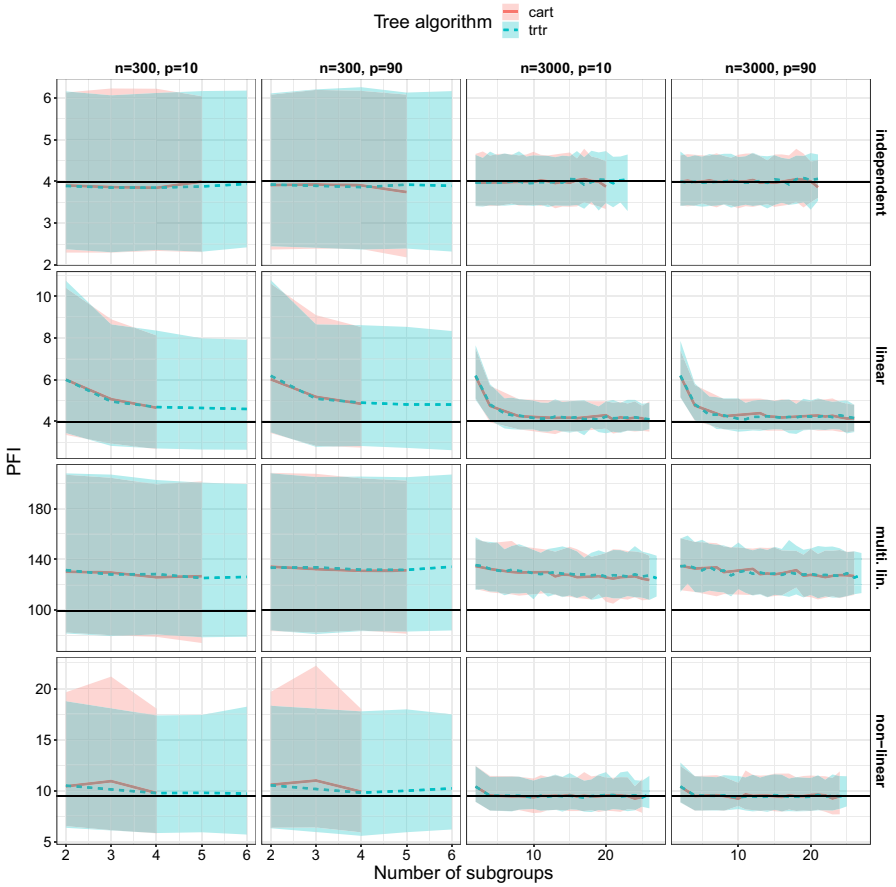

Fig. 12 Conditional PFI estimate using cs-PFI (**cart/tr**ansformation **tr**ee) with increasing number of subgroups (simulation scenario I). Displayed is the median PFI over 1000 repetitions along with the 5% and 95% quartiles

Appendix F Data fidelity on OpenML-CC18 data sets

An overview of data sets from the OpenML-CC18 benchmarking suit. We used a subset of 42 out of 72 data sets with 7 to 500 continuous features (Table 7).

Table 7 Overview of OpenML CC18 data sets used for the data fidelity experiment

OpenML ID	Name	No. Obs.	No. numerical feat.	No. feat.
1049	pc4	1458	38	38
1050	pc3	1563	38	38
1053	jm1	10,880	22	22
1063	kc2	522	22	22
1067	kc1	2109	22	22
1068	pc1	1109	22	22
12	mfeat-factors	2000	217	217
14	mfeat-fourier	2000	77	77
1461	bank-marketing	45,211	8	17
1475	first-order-theorem-proving	6118	52	52
1480	ilpd	583	10	11
1486	nomao	34,465	90	119
1487	ozone-level-8hr	2534	73	73
1494	qsar-biodeg	1055	42	42
1497	wall-robot-navigation	5456	25	25
15	breast-w	683	10	10
1501	semeion	1593	257	257
151	electricity	45,312	8	9
1510	wdbc	569	31	31
16	mfeat-karhunen	2000	65	65
182	satimage	6430	37	37
188	eucalyptus	641	15	20
22	mfeat-zernike	2000	48	48
23517	numerai28.6	96,320	22	22
28	optdigits	5620	63	65
307	vowel	990	11	13
31	credit-g	1000	8	21
32	pendigits	10,992	17	17
37	diabetes	768	9	9
40499	texture	5500	41	41
40701	churn	5000	17	21
40966	MiceProtein	552	78	82
40979	mfeat-pixel	2000	241	241
40982	steel-plates-fault	1941	28	28
40984	segment	2310	19	20
40994	climate-model-simulation-crashes	540	21	21
44	spambase	4601	58	58
4538	GesturePhaseSegmentationProcessed	9873	33	33
458	analcatdata_authorship	841	71	71
54	vehicle	846	19	19
6	letter	20,000	17	17
6332	cylinder-bands	378	19	40

Appendix F.1 Data fidelity results

See Figs. 13, 14, 15 and 16.

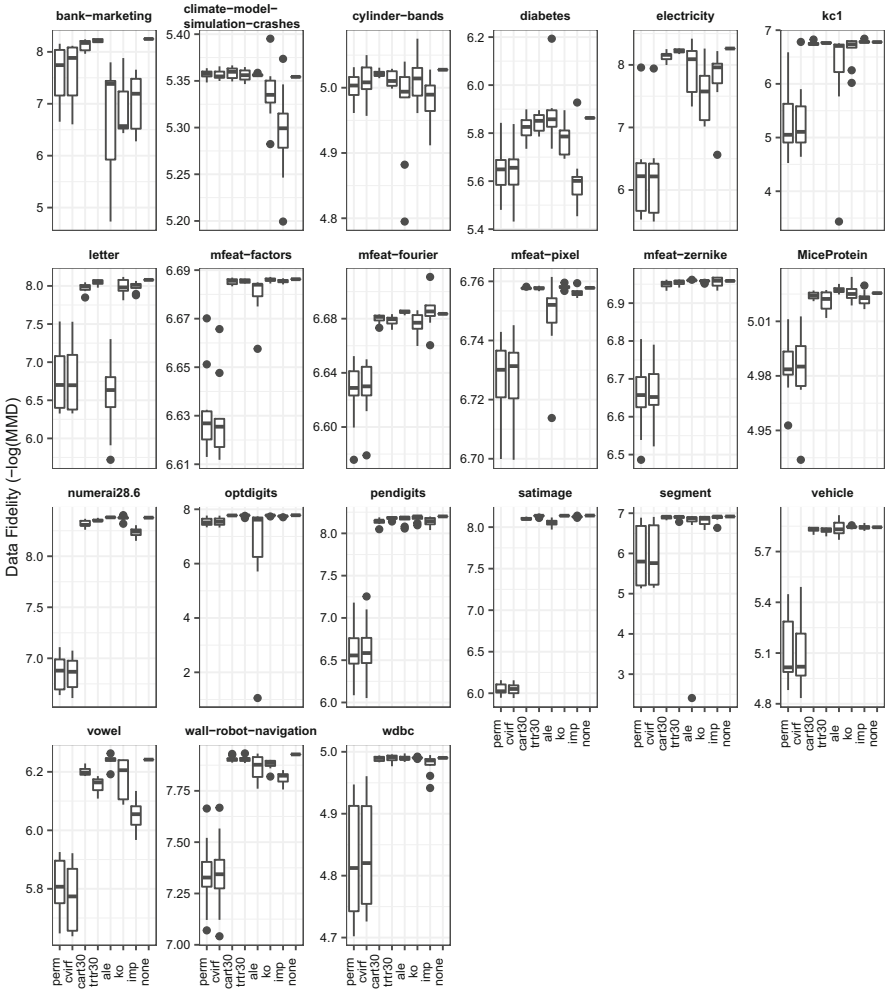

Fig. 13 Data Fidelity experiment with OpenML-CC18 data sets (1/2). Different sampling types are compared: unconditional permutation (perm), cs-permutation (maximal tree depth) with CART (cart30) or transformation trees (trtr30), Model-X knockoffs (ko), data imputation with a random forest (imp), ALE (ale), conditional variable importance for random forests (cvirf) and no permutation (none). Each data point in the boxplot represents one feature and one data set. Results from repeated experiments have been averaged (mean) before using them in the boxplots

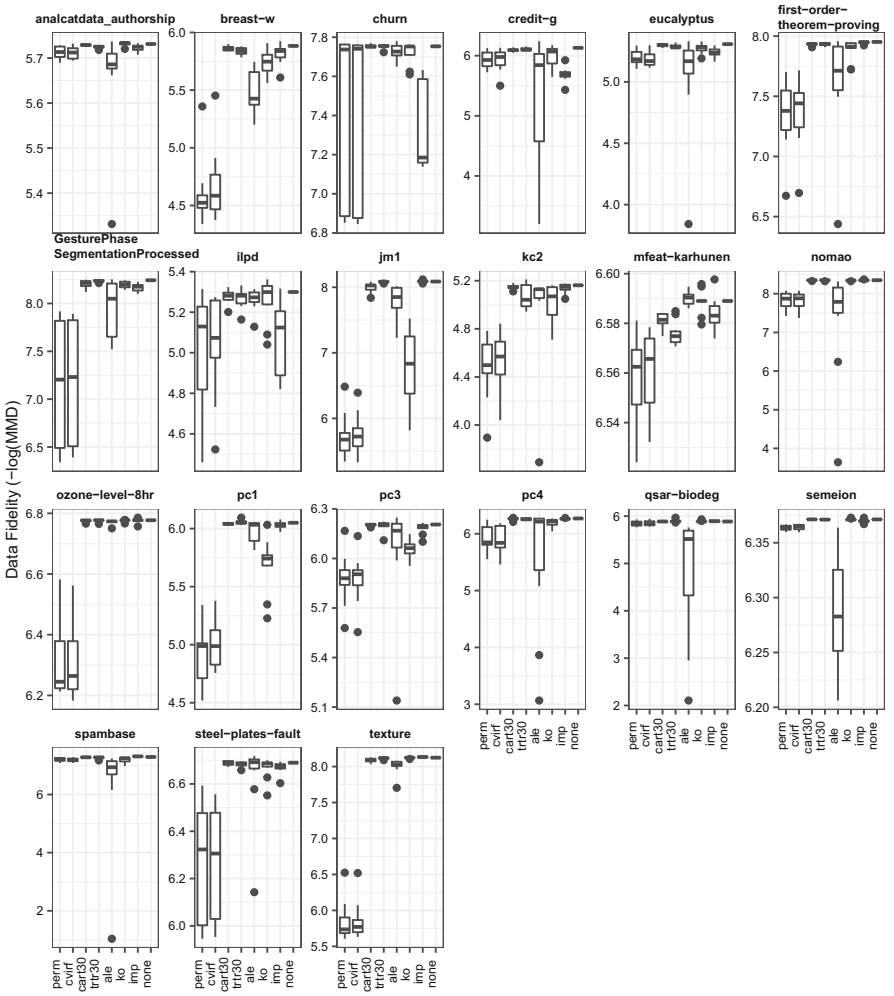

Fig. 14 Data Fidelity experiment with OpenML-CC18 data sets (1/2). Different sampling types are compared: unconditional permutation (perm), cs-permutation (maximal tree depth) with CART (cart30) or transformation trees (trtr30), Model-X knockoffs (ko), data imputation with a random forest (imp), ALE (ale), conditional variable importance for random forests (cvirf) and no permutation (none). Each data point in the boxplot represents one feature and one data set. Results from repeated experiments have been averaged (mean) before using them in the boxplots

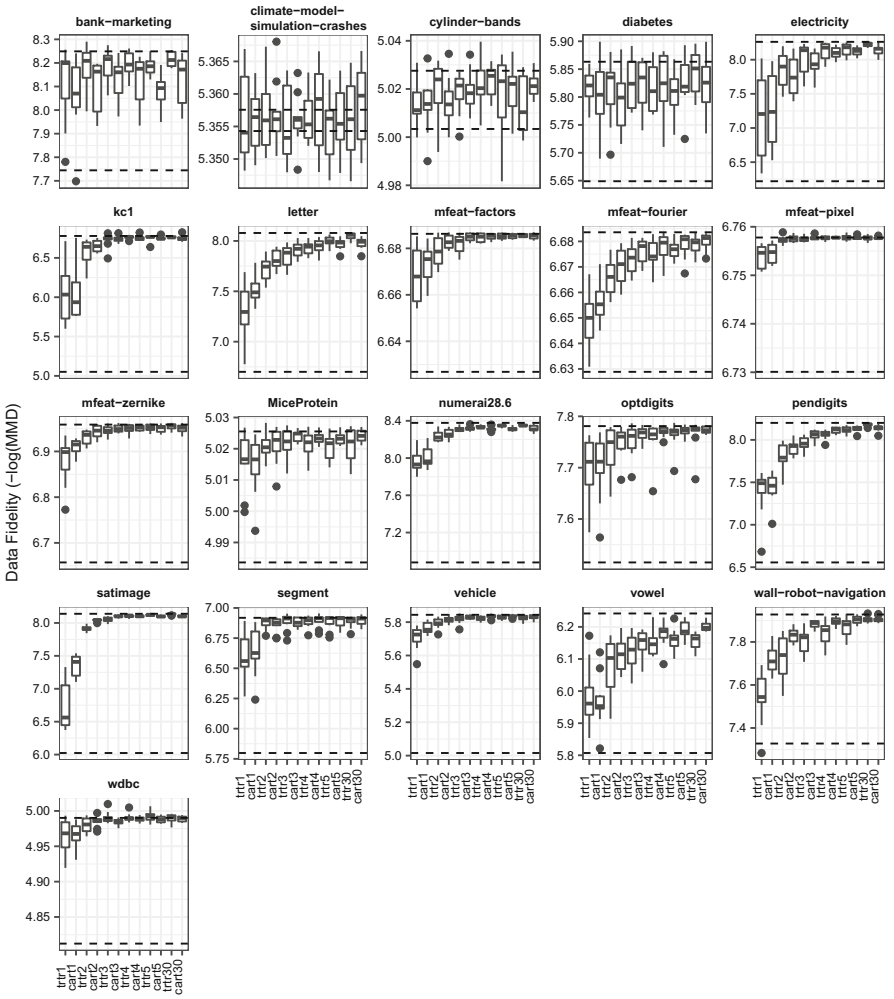

Fig. 15 Data Fidelity experiment with OpenML-CC18 data sets (1/2). Different tree depths and tree types (CART and Transformation Trees) are compared. Unconditional permutation and lack of permutation serve as lower and upper bound for data fidelity and their median data fidelity is plotted as dotted lines. Each data point in the boxplot represents one feature and one data set. Results from repeated experiments have been averaged (mean) before using them in the boxplots

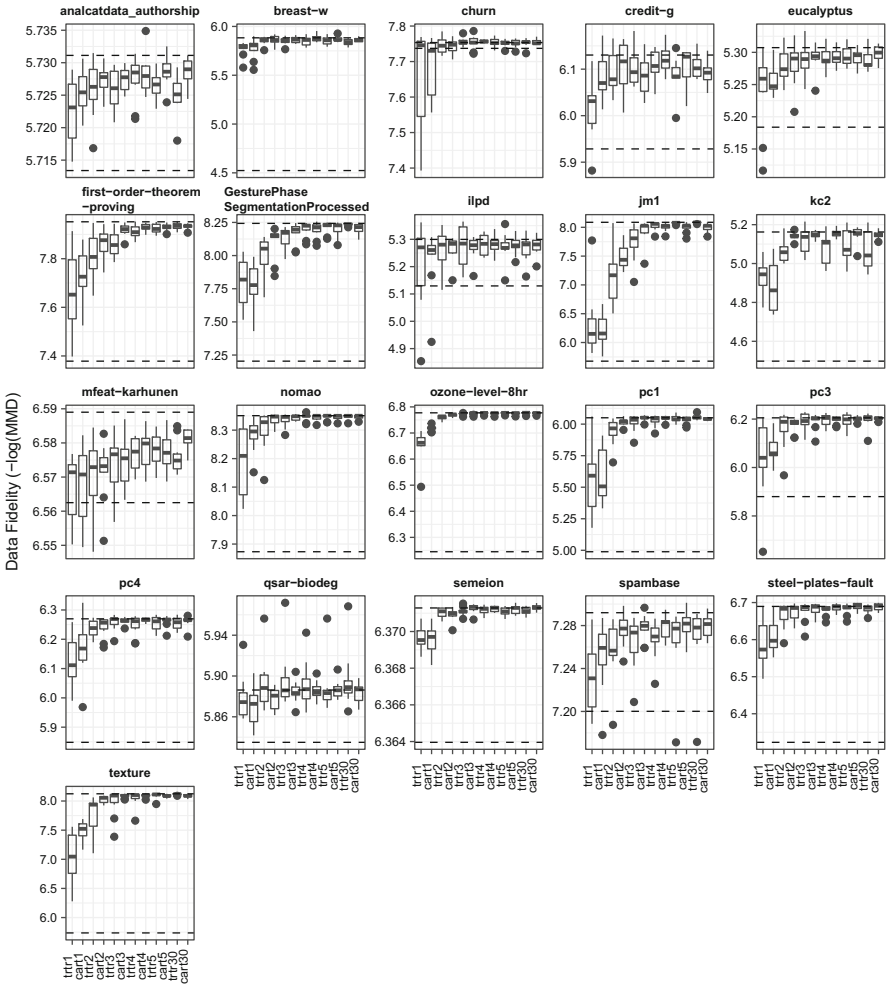

Fig. 16 Data Fidelity experiment with OpenML-CC18 data sets (1/2). Different tree depths and tree types (CART and Transformation Trees) are compared. Unconditional permutation and lack of permutation serve as lower and upper bound for data fidelity and their median data fidelity is plotted as dotted lines. Each data point in the boxplot represents one feature and one data set. Results from repeated experiments have been averaged (mean) before using them in the boxplots

Appendix G Model fidelity plots

See Fig. 17.

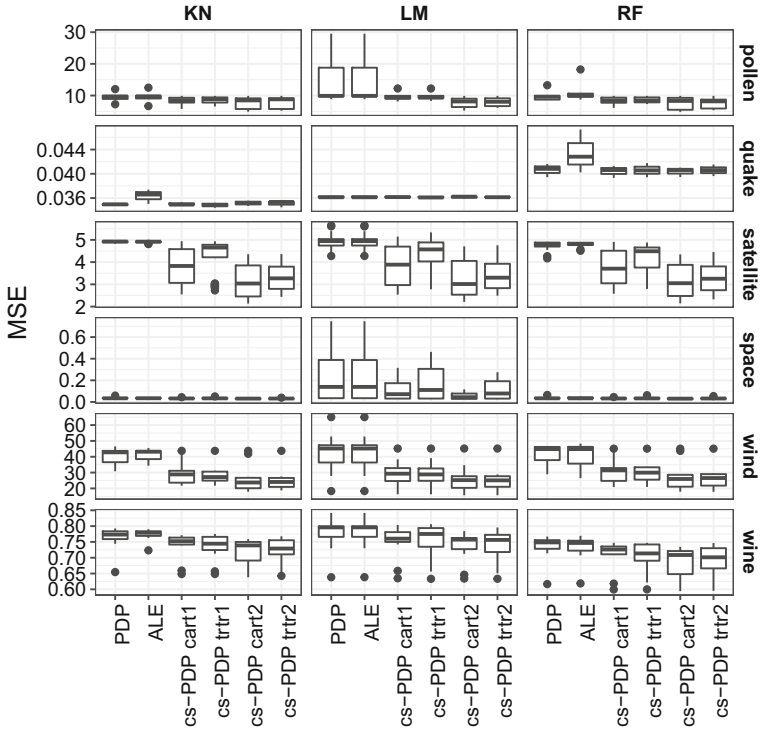

Fig. 17 Comparing the loss between model f and various feature effect methods. Each instance in the boxplot is MSE for one feature, summed over the test data

Appendix H Application: feature dependence analysis

The features in the bike data are dependent. For example, the correlation between temperature and humidity is 0.13. The data contains both categorical and numerical features and we are interested in the multivariate, non-linear dependencies. Thus, correlation is an inadequate measure of dependence. We therefore indicate the degree of dependence by showing the extent to which we can predict each feature from all other features in Table 8. This idea is based on the proportional reduction in loss (Cooil and Rust 1994). Per feature, we trained a random forest to predict that feature from all other features. We measured the proportion of loss explained by each random forest, compared to a constant model to quantify the dependence of the respective feature on all other features. For numerical features, this meant using the R-squared measure. For categorical features, we computed $1 - MMCE(y_{class}, rf(X))/MMCE(y_{class}, x_{mode})$,

where $MMCE$ is the mean misclassification error, y_{class} the true class, $rf()$ the classification function of the random forest and x_{mode} the most frequent class in the training data. We divided the training data into two folds and trained the random forest on one half. Then, we computed the proportion of explained loss on the other half and vice versa. Finally, we averaged the results. The feature "work" can be fully predicted by weekday and holiday. Season, temperature, humidity and weather can be partially predicted and are therefore not independent.

Table 8 Percentage of loss explained by predicting a feature from the remaining features with a random forest

Season	Holiday	Weekday	Temp	Hum	Work	Weather	Year	Wind
46%	25%	12%	66%	42%	100%	44%	10%	11%

References

Apley DW, Zhu J (2016) Visualizing the effects of predictor variables in black box supervised learning models. arXiv preprint arXiv:1612.08468

Bair E, Ohrbach R, Fillingim RB, Greenspan JD, Dubner R, Diatchenko L, Helgeson E, Knott C, Maixner W, Slade GD (2013) Multivariable modeling of phenotypic risk factors for first-onset TMD: the OPPERA prospective cohort study. J Pain 14(12):T102–T115

Bischl B, Casalicchio G, Feurer M, Hutter F, Lang M, Mantovani RG, van Rijn JN, Vanschoren J (2019) Openml benchmarking suites. arXiv preprint arXiv:1708.03731

Boulesteix AL, Wright MN, Hoffmann S, König IR (2020) Statistical learning approaches in the genetic epidemiology of complex diseases. Hum Genet 139(1):73–84

Breiman L (2001) Random forests. Mach Learn 45(1):5–32

Breiman L, Friedman J, Olshen R, Stone C (1984) Classification and regression trees. Wadsworth and Brooks, Boston

Bryk AS, Raudenbush SW (1992) Hierarchical linear models: applications and data analysis methods. Sage Publications Inc, Thousand Oaks

Candes E, Fan Y, Janson L, Lv J (2018) Panning for gold:'model-X'knockoffs for high dimensional controlled variable selection. J R Stat Soc Ser B (Stat Methodol) 80(3):551–577

Casalicchio G, Bossek J, Lang M, Kirchhoff D, Kerschke P, Hofner B, Seibold H, Vanschoren J, Bischl B (2017) OpenML: an R package to connect to the machine learning platform OpenML. Comput Stat 34:977–991

Chen H, Janizek JD, Lundberg S, Lee SI (2020) True to the model or true to the data? arXiv preprint arXiv:2006.16234

Cooil B, Rust RT (1994) Reliability and expected loss: a unifying principle. Psychometrika 59(2):203–216

Debeer D, Strobl C (2020) Conditional permutation importance revisited. BMC Bioinform 21(1):1–30

Dua D, Graff C (2017) UCI machine learning repository. http://archive.ics.uci.edu/ml

Esselman PC, Stevenson RJ, Lupi F, Riseng CM, Wiley MJ (2015) Landscape prediction and mapping of game fish biomass, an ecosystem service of Michigan rivers. N Am J Fish Manag 35(2):302–320

Fisher A, Rudin C, Dominici F (2019) 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):1–81

Fortet R, Mourier E (1953) Convergence de la répartition empirique vers la répartition théorique. Ann Sci l'École Normale Supér 70:267–285

Freiesleben T, König G, Molnar C, Tejero-Cantero A (2022) Scientific inference with interpretable machine learning: Analyzing models to learn about real-world phenomena. arXiv preprint arXiv:2206.05487

Friedman JH et al (1991) Multivariate adaptive regression splines. Ann Stat 19(1):1–67

Frye C, de Mijolla D, Begley T, Cowton L, Stanley M, Feige I (2020) Shapley explainability on the data manifold. arXiv preprint arXiv:2006.01272
Goldstein A, Kapelner A, Bleich J, Pitkin E (2015) Peeking inside the black box: visualizing statistical learning with plots of individual conditional expectation. J Comput Graph Stat 24(1):44–65
Gregorutti B, Michel B, Saint-Pierre P (2017) Correlation and variable importance in random forests. Stat Comput 27(3):659–678
Gretton A, Fukumizu K, Teo CH, Song L, Schölkopf B, Smola AJ et al (2007) A kernel statistical test of independence. Nips Citeseer 20:585–592
Gretton A, Borgwardt KM, Rasch MJ, Schölkopf B, Smola A (2012) A kernel two-sample test. J Mach Learn Res 13(1):723–773
Hooker G (2007) Generalized functional anova diagnostics for high-dimensional functions of dependent variables. J Comput Graph Stat 16(3):709–732
Hooker G, Mentch L (2019) Please stop permuting features: an explanation and alternatives. arXiv preprint arXiv:1905.03151
Hothorn T (2018) Top-down transformation choice. Stat Model 18(3–4):274–298
Hothorn T, Zeileis A (2015) partykit: a modular toolkit for recursive partytioning in R. J Mach Learn Res 16(1):3905–3909
Hothorn T, Zeileis A (2017) Transformation forests. arXiv preprint arXiv:1701.02110
König G, Molnar C, Bischl B, Grosse-Wentrup M (2020) Relative feature importance. arXiv preprint arXiv:2007.08283
Lang M, Binder M, Richter J, Schratz P, Pfisterer F, Coors S, Au Q, Casalicchio G, Kotthoff L, Bischl B (2019) mlr3: a modern object-oriented machine learning framework in R. J Open Source Softw 4:1903
Lei J, G'Sell M, Rinaldo A, Tibshirani RJ, Wasserman L (2018) Distribution-free predictive inference for regression. J Am Stat Assoc 113(523):1094–1111
Molnar C, Bischl B, Casalicchio G (2018) iml: an R package for interpretable machine learning. JOSS 3(26):786
Obringer R, Nateghi R (2018) Predicting urban reservoir levels using statistical learning techniques. Sci Rep 8(1):1–9
Parr T, Wilson JD (2019) A stratification approach to partial dependence for codependent variables. arXiv preprint arXiv:1907.06698
Patterson E, Sesia M (2020) knockoff: the knockoff filter for controlled variable selection. R package version 0.3.3. https://CRAN.R-project.org/package=knockoff
Pintelas E, Liaskos M, Livieris IE, Kotsiantis S, Pintelas P (2020) Explainable machine learning framework for image classification problems: case study on glioma cancer prediction. J Imaging 6(6):37
R Core Team (2017) R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna
Ribeiro MT, Singh S, Guestrin C (2016) Why should I trust you?: Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, ACM, pp 1135–1144
Romano Y, Sesia M, Candès E (2019) Deep knockoffs. J Am Stat Assoc, pp 1–12
Scholbeck CA, Molnar C, Heumann C, Bischl B, Casalicchio G (2019) Sampling, intervention, prediction, aggregation: a generalized framework for model-agnostic interpretations. In: Joint European conference on machine learning and knowledge discovery in databases. Springer, pp 205–216
Smola A, Gretton A, Song L, Schölkopf B (2007) A Hilbert space embedding for distributions. In: International conference on algorithmic learning theory. Springer, pp 13–31
Stachl C, Au Q, Schoedel R, Gosling SD, Harari GM, Buschek D, Völkel ST, Schuwerk T, Oldemeier M, Ullmann T, Hussmann H, Bischl B, Bühner M (2020) Predicting personality from patterns of behavior collected with smartphones. Proc Natl Acad Sci 117(30):17680–17687
Stiglic G, Kocbek P, Fijacko N, Zitnik M, Verbert K, Cilar L (2020) Interpretability of machine learning-based prediction models in healthcare. Wiley Interdiscip Rev Data Min Knowl Discov 10(5):e1379
Strobl C, Boulesteix AL, Kneib T, Augustin T, Zeileis A (2008) Conditional variable importance for random forests. BMC Bioinform 9(1):307
Szepannek G (2019) How much can we see? A note on quantifying explainability of machine learning models. arXiv preprint arXiv:1910.13376
Vanschoren J, Van Rijn JN, Bischl B, Torgo L (2014) OpenML: networked science in machine learning. ACM SIGKDD Explor Newsl 15(2):49–60

Watson DS, Wright MN (2021) Testing conditional independence in supervised learning algorithms. Mach Learn 110(8):2107–2129

Zhao X, Yan X, Yu A, Van Hentenryck P (2020) Prediction and behavioral analysis of travel mode choice: a comparison of machine learning and logit models. Travel Behav Soc 20:22–35