Master's Thesis

Interpretation of black box models using tree-based surrogate models

Department of Statistics
Ludwig-Maximilians-Universität München

Sofia Maria Loibl

Munich, March 24th, 2023

Submitted in partial fulfillment of the requirements for the degree of M.Sc. in Statistik
mit wirtschafts- und sozialwissenschaftlicher Ausrichtung
Supervised by Dr. Giuseppe Casalicchio

Abstract

Surrogate models allow complex but powerful black box machine learning models to be interpreted retrospectively. In this work, the following requirement is placed on a surrogate model: It should divide the feature space in subregions where interpretable models that only include main effects can approximate the black box model. In this way, both good interpretability and high performance should be achieved. For the generation of such models, four different model-based tree algorithms are compared: SLIM, GUIDE, MOB and CTree. Selection bias, performance, interpretability and stability of the different methods are investigated. In the presence of subgroup-specific main effects, SLIM and GUIDE convince with their results regarding interpretability and performance.

Contents

1 Introduction

Various machine learning algorithms achieve outstanding predictive performance nowadays. However, most of them are complex black box models that, unlike traditional statistical methods such as linear regression, are not intrinsically interpretable (Hu et al., 2020). Especially in highly regulated industries such as insurance, this is a problem (Henckaerts et al., 2022). In order to maintain the good performance of complex black box models but still achieve a certain degree of interpretability, there exist different methods with which models can be interpreted post-hoc.

One option are so called surrogate models. The idea is to approximate the predictions of black box models by intrinsically interpretable models (Molnar, 2019). But in order for a surrogate model's explanations to be trusted, it must itself perform well in approximating the black box predictions. There are basically two approaches for generating surrogate models: global surrogate models, which cover the entire feature space and can be interpreted globally, but often do not perform that well, or local surrogate models, which only explain individual observations. The focus of this work is on a mixture of the two approaches, which are called subregional surrogate models here. The idea is to find subregions in the feature space in which intrinsically interpretable models can be fitted.

Model-based tree (MBT) algorithms are a promising class to generate such subregional models. The concept of MBTs is based on classical regression trees like CART (Breiman et al., 1984) but a MBT contains models instead of constant values in the subregions (called nodes or leaf nodes). It is hence a combination of decision rules and models. If intrinsically interpretable models are chosen for the models, a principally interpretable MBT is derived. The degree of interpretability and performance, though, depends on the complexity of the decision rules (i.e. depth of the tree or number of leaf nodes) and of the models. In order to enable a good interpretability of the models in the leaf nodes, this thesis sets the constraint that only main effect models are fitted in the nodes. In the ideal case, interactions should then be handled by splits, so that the models in the leaf nodes are free from interaction effects. To ensure that the splits are actually due to interactions and not to nonlinearities, it is necessary that potentially nonlinear main effects are modelled appropriately. Although in some cases linear modelling of the main effects is sufficient, it should also be possible to fit more complex models, such as (penalized) polynomial models or generalised additive models (GAM). The goal is thus an additive decomposition of a black box function by combining decision rules and additive main effect models. In other words subregions in the feature space should be found, so that the global black box model can be replaced by subregional main effect models.

In my research, I found four different algorithms that can generate MBTs. The most recent

approach Surrogate Locally-Interpretable Models (SLIM) by Hu et al. (2020) allows a high flexibility in the choice of model classes in the leaf nodes. SLIM uses an exhaustive search to find the best split point through all possible splitting variables. In this thesis SLIM is contrasted with three common algorithms for creating MBTs: Model-based recursive Partitioning (MOB) (Zeileis et al., 2008), Conditional Inference Trees (CTree) (Hothorn et al., 2006) and Regression tress with unbiased variable selection and interaction detection (GUIDE) (Loh, 2002). These algorithms differ from SLIM mainly in that the search for the best split point is a two-step process. First, the best split variable is selected by a hypothesis test, which differs between the three algorithms. Then, in a second step, the best split point for this variable is searched. Since certain prerequisites must be fulfilled for the hypothesis tests, the choice of model classes for these algorithms is limited to some extent.

The comparison of the four algorithms is made with the condition that main effect only models are fitted in the nodes and all features are included in the search for the splitting variable. First, it is examined how the algorithms differ in terms of selection bias. Afterwards, they are compared with regard to their performance, interpretability and stability, as tree algorithms often suffer from poor stability (Fokkema, 2020). In the subsequent application to data sets from the area of life insurance, their suitability as surrogate models is examined.

The main results of this thesis are that SLIM and GUIDE are particularly successful in terms of performance, interpretability and stability when subgroup specific maineffects are present in the data. Smooth interactions are partially better modelled by MOB and CTree. In general for data with smooth interactions all four algorithms produce models with a large number of subregions when high performance is required. The MBTs are then difficult to interpret and less suitable as surrogate models. Furthermore, when applied to the insurance data, it is noticeable that the deeper the MBTs are, the more similar the performance of the different algorithms becomes.

The thesis is structured as follows: In chapter 2, an overview of surrogate models is given and MBTs are placed in this setting. Thereafter, the four MBT algorithms are described in chapter 3. Chapter 4 examines the extent to which the algorithms suffer from selection bias. In chapter 5, different simulation scenarios are used to examine how the algorithms differ in terms of performance, interpretability and stability. In addition, the interpretability and performance of the SLIM algorithm are examined when the complexity of the leaf node models is varied. Finally, in chapter 6 the algorithms are used as surrogate models for the modelling of insurance data and different SLIM models are used for interpretation.

2 Related work

Surrogate models are one possibility to explain complex black box models post-hoc. According to Molnar (2019) the purpose of (interpretable) surrogate models is to approximate the predictions of the underlying model as accurately as possible and to be interpretable at the same time. Depending on whether the goal is to achieve a global interpretation of a black box model (model explanation) or only to explain the results of individual input instances (outcome explanation), global or local surrogate models can be used (Maratea and Ferone, 2021). An advantage of using surrogate models as interpretable machine learning (IML) -method is, that it is a model-agnostic approach, i.e., it does not require any information about the inner workings of the black box model (Molnar, 2019).

The concept of global surrogate models is very simple. To explain the underlying model, the predictions for some input data are first calculated using this black box model. Then, an arbitrarily interpretable surrogate model is trained on the input data and the predictions. Finally, it can be measured how well the surrogate model reproduces the black-box predictions (Molnar, 2019). If the performance is good enough, the interpretation can be made according to the chosen surrogate model.

Molnar (2019) cites the flexibility in choosing the surrogate model and the simplicity of the approach as advantages of this method. However, this flexibility is also a challenge, as the choice of an appropriate surrogate model is a trade-off between high performance due to the potentially higher complexity of the model class and its interpretability. For example, linear models are relatively easy to interpret, but cannot uncover nonlinear relationships that may have been modeled in the underlying black box model. Molnar (2019) provides an overview of interpretable models and their strengths and weaknesses, such as generalized additive models (GAM), decision trees, or sparse linear models (e.g. lasso regression).

In addition, other promising high-performance models that can be interpreted to a certain degree are being developed. Generalized additive models plus interactions (GA2M) of Lou et al. (2013) are an example of an extension of generalized additive models that allow a small number of two-way interactions to be efficiently modeled in addition to the nonlinear main effects.

One aspect that must be remembered when using global surrogate models is that only conclusions about the underlying black box model can be drawn, not about the actual data generating process (Molnar, 2019).

If only a single prediction is to be explained instead of the entire model, local interpretable model-agnostic explanations (LIME) can be used (Ribeiro et al., 2016). Instead of having

to find a (possibly highly complex) interpretable model on the entire feature space, only a surrogate model for a small neighborhood around the instance of interest is trained. This is realized by observing how the predictions of the black-box model behave when small variations of the instance of interest are introduced into the model. Like global surrogates, LIME is very flexible in the choice of an appropriate surrogate model. The advantage over global surrogate models is that it is easier to find a good compromise between model complexity and interpretability, since the model only needs to fit a very small range of the feature space. Even sparse main effect models could imitate the black box model well in the small area of the feature space, for example. One difficulty, however, is choosing an appropriate neighborhood for the instance of interest to which the model is to be fitted. (Molnar, 2019)

A approach that tries to combine the advantages of global and local surrogate methods, i.e., to obtain models that are easy to understand on the one hand, but at the same time cover the entire feature space without sacrificing performance, is summarized here under the term subregional surrogate model. The idea is to subdivide the feature space into appropriate subregions where fitting easy-to-interpret models is sufficient. With K-LIME (K local interpretable model-agnostic explanations), Hall et al. (2017) pursue the approach of unsupervised partitioning of the feature space into K subregions using a K-means clustering algorithm. SLIM (Hu et al., 2020), on the other hand, uses a supervised approach (MBT) to subdivides the feature space according to a given objective. According to (Hu et al., 2018), SLIM is superior to K-LIME, which is mainly because K-LIME does not include the objective in the clustering. Another recent approach, called maidrr (Model-Agnostic Interpretable Data-driven suRRogate) is presented in Henckaerts et al. (2022). There, the feature space is partitioned on the basis of partial dependence plots and then linear models in categorical format are fitted in the subregions. The categorical format is intended to increase interpretability, but thereby has a negative effect on the performance.

In this thesis, the SLIM algorithm is used and compared with other algorithms that can create MBTs.

3 Model-based trees

In the following, MBTs are introduced and the different algorithms are presented.

3.1 Model and notation

Following (Zeileis et al., 2008) and (Seibold et al., 2016), let $M((y; \mathbf{x}); \ )$ be a parametric model, that describes the target $y \ 2 \ Y$ as a function of a feature matrix $\mathbf{x} \ 2 \ X$ with p features $\mathbf{x_1}; :::\mathbf{x_p}$ and a vector of parameters $\ 2 \ \Theta$. When the model is used as a surrogate model, y is the prediction of the black box model that should be explained. If it must be explicitly stated that the response is the prediction of a black-box model, the notation y^{bb} is used. Given n observations $(y; \mathbf{x}) = (y^{(1)}; \mathbf{x}^{(1)}); :::; (y^{(n)}; \mathbf{x}^{(n)})$ the model can be fitted by minimizing some objective function $\Psi((y; \mathbf{x}); \ )$ yielding the parameter estimate $\hat{}$

$$\hat{} = \arg\min_{2} \sum_{i=1} \Psi(y^{(i)}; \mathbf{x}^{(i)}; \ ): \tag{1}$$

In this thesis $M((y; \mathbf{x}); \ )$ is restricted to additive main effect regression models. However, if the data includes interactions, a single global model is not sufficient to fit all n observations well. To deal with interactions, the idea is to partition the feature space X into subregions $fB_b g_{b=1;:::;B}$ and search for locally well fitting main effect models in these regions. The global objective function thus expands to

$$\sum_{b=1} \sum_{i2I_b} \Psi(y^{(i)}; \mathbf{x}^{(i)}; \ _b) \tag{2}$$

and has to be minimized over all partitions $fB_b g$ (with corresponding indexes $I_b; b = 1; :::; B$) and local optimal parameter $\ _b$ for each partition. As (Zeileis et al., 2008) points out, it is very difficult to find the optimal partition, because the number of possible partitions quickly becomes too large for an exhaustive search.

3.2 Model-based tree algorithms

In this chapter, four algorithms are described that aim to find a partition that is close to the optimal one and the associated estimators for $\ _b$ through binary recursive partitioning. All approaches use a greedy forward search to optimize the objective function Ψ locally in each step (Zeileis et al., 2008). The resulting global models are called Model-based trees . A recursive partitioning algorithm for MBTs can generally be divided into the following steps:

1. Start with the partition (node) $I_0 = \{1, \ldots, n\}$
2. Fit the model to all observations in the current node $\{y^{(i)}, \mathbf{x}^{(i)}\}, i \in I_b$ by estimating $\hat{}_b$ via minimization of the objective function Ψ
3. Find the local optimal splitpoint for this node
4. If no stop criterion is met (e.g. depth of the tree, improvement of the objective through split, significance of parameter instability) split the node in two child nodes and repeat the steps 2-4.

The algorithms SLIM, MOB, CTree and GUIDE considered in this work can be divided into two groups. SLIM falls into the group of biased recursive partitioning algorithms, which also includes classical methods like AID (Morgan and Sonquist, 1963) and CART (Breiman et al., 1984). These algorithms use an exhaustive search to select the optimal split point in step 3. This can lead to the phenomenon that, even if the strength of all feature effects (or in our case interaction effects) are the same, for example features with many possible split points are more often chosen as split variables than features with few possible split points. This phenomenon is called selection bias and is explained in more detail in chapter 4 and examined for the four MBT algorithms. MOB, CTree and GUIDE are assigned to unbiased recursive partitioning. Instead of comparing the objective for all possible split points and variables, step 3 of the recursive partitioning algorithm is split into 2 steps:

1. Select the variable that has the highest association with the response as splitting (partitioning) variable. The tests to determine the most significant association differ between the methods.
2. Search for the best split point only within this variable (e.g. by exhaustive search or again by hypothesis testing)

(Schlosser et al., 2019)

3.2.1 SLIM

SLIM by Hu et al. (2020) is explicitly designed to create surrogate MBTs that contain main effect models in the leafnodes and are split by interactions. The assumption here is that if the main effects in the nodes are well fitted, any lack of fit must come from interactions. Therefore, it is not necessary to specify in advance which features should be used as splitting variables and which as regressor variables, but each feature can take on both roles and the selection is done automatically in the recursive partition algorithm.

After fitting the model to all observations in step 2 the objective function is calculated, e.g. the sum of squared errors (SSE). In step 3 SLIM performs an exhaustive search to find the optimal split point. The optimization problem in each recursion step includes the optimization of the objective of the left node and the right node and is given by

$$\min_{j \in \{1,\ldots,p\}} \left(\min_{s \in S_j} \left(\min_{c_l} \sum_{i \in I_l^s} \Psi(y^{(i)}, \mathbf{x}^{(i)}, c_l) + \min_{c_r} \sum_{i \in I_r^s} \Psi(y^{(i)}, \mathbf{x}^{(i)}, c_r) \right) \right), \tag{3}$$

where S_j is the set of all possible split points s (or split sets) regarding variable $\mathbf{x}_j$. For numeric variables, this set consists either of all unique values of $\mathbf{x}_j$ or, in order to reduce the calculation effort, of sample quantiles (e.g. 100 quantiles). For a specific split point s, the left node I_l^s is defined as the set of indices $i \in \{1, \ldots, n\}$ for which $x_j^{(i)} \leq s$ holds and the right node I_r^s is its complement. For categorical variables, all possibilities of dividing the levels of $\mathbf{x}_j$ into two disjoint sets are considered as potential split sets S_j. I_l^s is the set of all indices for which $x_j^{(i)} \in s$ applies and I_r^s is its complement.

The feature $\mathbf{x}_j$ and split point s which minimize (3) are selected as splitting variable and split point, if this leads to a large enough improvement of the objective function.

Since the computational effort for estimating all possible child models becomes very large as the number of possible partitioning variables increases, Hu et al. (2020) have developed an efficient algorithm for estimating them for the case of linear regression, linear regression with B-spline transformed features and ridge regression. A detailed description of this algorithm can be found in Hu et al. (2020).

To avoid overfitting and to obtain a small interpretable tree, the use of pruning is necessary. Hu et al. (2020) use the approach of backpruning, i.e. a deep tree is first fitted and then leaves that do not fulfil certain criteria are pruned back. In order to keep the computational effort as low as possible, prepruning is used in this thesis. This means that a split is only performed if the improvement of the objective through the split is at least $impr \in [0, 1]$ times the improvement of the objective in the parent node. In addition, it is possible to set a value for R^2 (defined in chapter 5) above which splitting in a node will not continue.

3.2.2 MOB

MOB generally distinguishes between regressor variables $\mathbf{x_j}$, which are only used to fit the models in the nodes, and pure partitioning variables $\mathbf{z_j}$. In (Zeileis et al., 2008), however, an overlapping of roles is explicitly not excluded. In order to make the method comparable with SLIM and to have the advantage that the roles do not have to be assigned

beforehand, all features should be able to enter the MBT both as main effect and in the splits as interactions. In this way, the application of MOB in this thesis differs from other application examples as in Seibold et al. (2016) or Thomas et al. (2018).
After an initial model has been fitted in step 2, MOB examines whether the corresponding parameter estimates $\hat{}_b$ are stable. If there is some overall instability, the feature whose parameter estimate has the most significant instability is chosen as splitting variable.
To investigate this, the so called score function is considered, which is defined as the gradient of the objective function regarding the parameter vector $_b$ - provided that it exists:

$$((y;\mathbf{x})\,;\ _b) = \frac{@\Psi\,((y;\mathbf{x})\,;\ _b)}{@\ _b}: \tag{4}$$

(Zeileis et al., 2008)
If the scores - ordered by the potential split variable - do not fluctuate randomly around zero, this indicates that there is parameter instability which could potentially be captured by splitting the data using this variable as partitioning variable (Schlosser et al., 2019). To test the null hypothesis of parameter stability with the so called M-fluctuation test, MOB captures systematic deviations from zero through the empirical fluctuation process

$$W_j(t) = \hat{J}\ ^{1=2} n\ ^{1=2} \sum_{i=1}^{bntc} \hat{}_{(x_j^{(i)})} \quad (0 \quad t \quad 1); \tag{5}$$

where $\hat{}_{(x_j^{(i)})}$ are the scores ordered by $\mathbf{x}_j$ and $\hat{J}$ is an estimate of the covariance matrix $cov(\ (Y;\hat{}))$. (Zeileis et al., 2008) In contrast to the definition in Zeileis et al. (2008), $\mathbf{x}_j = \mathbf{z}_j$ was set in (5) as no distinction is made between regressor and partitioning variables.
According to Zeileis et al. (2008) and Zeileis and Hornik (2007), under the null hypothesis of parameter stability the empirical fluctuation process $W_j(t)$ converges to a Brownian Bridge W_0. By applying a scalar test function to the empirical fluctuation process and the Brownian Bridge, a test statistic and the theoretical limiting distribution can be derived. An overview of possible scalar functions that can be used for this purpose can be found in (Zeileis et al., 2008) and in more detail in (Zeileis and Hornik, 2007). The variable for which the M-fluctuation test detects the most significant parameter instability is used as splitting variable. The choice of the optimal split point with respect to this variable is then made by means of an exhaustive search, analogous to SLIM. As prepruning criterion,

MOB uses the Bonferroni-adjusted p-value of the M-fluctuation test. That means a split is only performed if the instability is significant at a given significance level *alpha*.
A limitation of MOB is that only objective functions can be selected for which the gradient regarding $_b$ exists. This means, for example, that it is not possible to fit lasso models in the nodes.

3.2.3 CTree

CTree was originally developed as a non-parametric regression tree (i.e. for constant fits in the nodes) but can also be used for MBTs. CTree follows a very similar approach to MOB and also tries to detect parameter instability by analysing the dependency between potential splitting variables and a transformation $h()$ of the response y. A common transformation used in MBTs is the score function, i.e. $h(y) =$, but other transformations of the response variable that can detect instabilities in the estimates could also be used. A simple alternative would be to use the residuals. According to (Schlosser et al., 2019), however, the use of the scores - if they exist - is preferable, since instabilities can be better detected with them.
CTree uses a linear association test to test the independence between the scores and the potential partition variables. To measure the association between the scores and a potential splitting variable $\mathbf{x}_j; j = 1; :::; p$ a linear statistics of the form

$$T_j(y; \mathbf{x}) = vec \left(\sum_{i=1} \mathbf{x}_j^{(i)} \hat{}(i)T \right) \quad 2 \ \mathbf{R}^{p_j q} \tag{6}$$

is used which is derived from the more general definition in (Hothorn et al., 2006).
To standardize the test statistic T_j, the conditional expectation and covariance of T_j under the null hypothesis of independence between y and $\mathbf{x}_j$ given all permutations of $fy; \mathbf{x}g$ are calculated. Different transformations can be used to map the multivariate test statistic T_j to a standardized univariate test statistic. The default setting in the R package `partykit` (Hothorn et al., 2015) is a quadratic transformation for numerical features. According to (Hothorn et al., 2006) the transformed test statistic follows an asymptotic ^{2} distribution under the null hypothesis. As in MOB, the splitting variable, for which the p-value of the test is the smallest is selected as splitting variable. A Bonferroni-adjusted p-value is again used as prepruning criterion.
Unlike the other methods, the split point is not selected by an exhaustive search, but with the help of a linear test statistic. The discrepancy between two subsets is measured with a two-sample linear test statistic for each possible binary split. The split that maximises the discrepancy is chosen as split point. (Hothorn et al., 2006)

Schlosser et al. (2019) state that the linear test used in CTree has higher power in detecting smooth relationships between the scores and the splitting variables compared to the M-fluctuation test in MOB. MOB, on the other hand, has a higher ability in detecting abrupt changes.

3.2.4 GUIDE

GUIDE (Loh, 2002) uses residual-based categorical association tests to detect instabilities. For this purpose, ^{2}- independence tests between the dichotomized residuals of the fitted model and the categorized features are performed and the p-values of these so-called curvature tests are calculated. In addition to the curvature tests, GUIDE explicitly searches for interactions, which is promising in the application desired here. For this again, ^{2}- independence tests are performed. Instead of categorizing only one variable, a new categorical variable is created for each feature pair by combining two features for the interaction test. If the smallest p-value comes from a curvature test, the corresponding feature is chosen as the partitioning variable. If the smallest p-value is from an interaction test, the categorical feature involved, if any, is preferably chosen as the splitting variable. If both potential features are categorical, the variable for which the p-value of the curvature test is smaller is chosen. In the case of two numerical variables, the choice is made by evaluating the potential child models after splitting with respect to both variables. Subsequently, a bootstrap selection bias correction is performed. In the original GUIDE algorithm developed by (Loh, 2002), numerical features can be used both as regressor variables and as splitting variables. Categorical features, on the other hand, can only take on the role of splitting variables. This is justified by the fact that a disproportionately large number of degrees of freedom are consumed in the parameter estimation of categorical variables. In the following simulations, unless explicitly stated otherwise, the variant is used in which all features can fulfil both roles.
One advantage of GUIDE is that the score function does not have to exist. This makes the choice of objective very flexible and allows, for example, to fit lasso regression models in the nodes.

3.3 Software implementation

No implementation of SLIM was found, which is why an implementation in R (R Core Team, 2022) was carried out as part of this work. The code from the package `customtrees` from Casalicchio (2020) and from Herbinger et al. (2022) was used as basis. The source code for the SLIM implementation and the applications in this thesis can be found at `https://github.com/slds-lmu/msc_2022_loibl_thesis`.

The software implementation of GUIDE falls in the category "Algorithms with a closed-source, free-of-charge implementation" (Loh, 2014). The binary executable is available under `https://pages.stat.wisc.edu/~loh/guide.html`. Unfortunately, the source code is not accessible and the method cannot be easily adapted to other objectives, although this would be theoretically possible. For this reason, residual based ^{2}- independence tests for finding the best split variable were incorporated as an option of the SLIM implementation in R in this work. Pruning is therefore carried out in the same way as for SLIM. Bootstrap bias correction was also adopted. Even though following all the steps of the original paper (Loh, 2002) the replication of an experiment presented in the paper did not lead to the same results, as can be seen in the appendix in Table 20. Whenever GUIDE is discussed in the following, it refers to the R implementation achieved in this work. This implementation shows good results as can be seen in the simulation in chapter 5 and the insurance use case.
MOB and CTree are implemented in the R package `partykit` (Hothorn and Zeileis, 2015) in the functions `mob`() (Zeileis et al., 2008) and `ctree`() (Hothorn et al., 2006).

3.4 Comparison

In Table 1 a comparison of the different MBT algorithms is given. SLIM and GUIDE are particularly interesting because of their flexibility in choosing different objective functions. In the case of SLIM, one open question is whether the exhaustive search, in contrast to the two-step methods, leads to a selection bias. This will be examined in the following chapter 4.

	Split point selection	Test	Flexibility	Implementation
SLIM	exhaustive search	-	high	-
MOB	two-step	score-based fluctuation	low	R package
CTree	two-step	score-based Permutation	low	R package
GUIDE	two-step	residual-based ^{2}	high	binary executable

Table 1: Comparison of MBT algorithms

Performance, interpretability and stability of the algorithms are empirically compared by simulations in chapter 5.

4 Selection bias and splitting strategies

4.1 Definition and motivation

According to (Hothorn et al., 2006) an algorithm for recursive partitioning is called unbiased when, under the conditions of the null hypothesis of independence between a response y and feature $\mathbf{x}_1, \ldots, \mathbf{x}_p$ the probability of selecting feature $\mathbf{x}_j$ is $1/p$ for all $j = 1, \ldots, p$ regardless of the measurement scales or number of missing values.
This definition may be surprising at first, since if there is no dependency between the target and the features, one would not want to perform a split anyway and would try to prevent this through appropriate pruning procedures. However, the idea behind this is that if, in the case of independence, a feature with, for example, a large number of possible split points is selected more frequently as a splitting variable than a feature with a smaller number of split points, the former could also be incorrectly selected more frequently as a splitting variable if there is a dependency on the response for the second feature. (Loh, 2014)
In the discussion on the Paper Fifty Years of Classification and Regression Trees of (Loh, 2014), Carolin Strobl's statement on selection bias is therefore very clear:

"One should think that the results shown here [...] are so clear that any statistically educated person should never want to use a biased recursive partitioning algorithm again."

So if SLIM is the only biased algorithm, it would have to be discarded immediately according to this statement. However, this will be investigated in detail in the following.

4.2 Simulation independence scenarios

In order to empirically investigate whether the four algorithms presented actually correspond to the respective groups (biased and unbiased) using the definition of Hothorn et al. (2006), two different simulation were carried out.
In the first scenario only numerical features are used, whereas in the second scenario numerical features and additionally one binary and two categorical features are considered.

Independence numeric

The data in the first scenario is defined as follows:

- $\mathbf{x}_1, \mathbf{x}_2 \quad U(0, 1)$

- $\mathbf{x}_3$ uniformly distributed on the set $\{0; 0.1; ...; 0.9; 1\}$
- $\mathbf{x}_4$ uniformly distributed on the set $\{0; 0.01; ...; 0.99; 1\}$
- $y \quad N(0; 1)$
- sample size $n = 1000$
- 1000 simulation runs

The resulting frequencies are shown in Figure 1 and are consistent with the expectations. SLIM clearly prefers to select features with a higher number of split points and therefore is biased. The other three methods seem to select all four features about the same number of times.

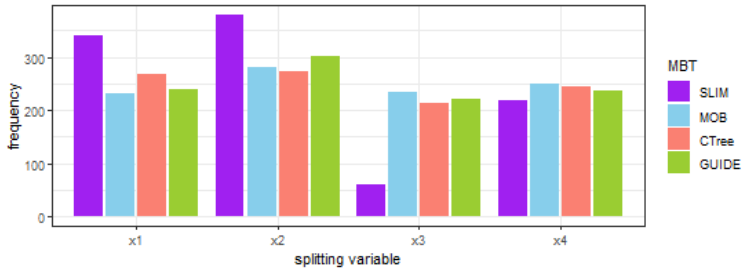

Figure 1: Simulated frequencies of selected splitting features for scenario independence numeric

Selection bias independence mixed

In the second scenario, the data is simulated as follows:

- $\mathbf{x}_1; \mathbf{x}_2 \quad U(0; 1)$
- $\mathbf{x}_3$ uniformly distributed on the set $\{0; 0.1; ...; 0.9; 1\}$
- $\mathbf{x}_4 \quad Bern(0.5)$
- $\mathbf{x}_5$ uniformly distributed on 5 factor levels (15 possible splits)
- $\mathbf{x}_6$ uniformly distributed on 8 factor levels (128 possible splits)
- $y \quad N(0; 1)$
- sample size $n = 1000$
- 1000 simulation runs

The addition of categorical features in this scenario results in a different picture for the frequencies, as shown in Figure 2. Although the numerical variables are chosen with approximately equal frequency in the so-called "unbiased" methods, there are large deviations in the binary and categorical variables especially for MOB and CTree, which calls the designation "unbiased" into question. A possible explanation for this selection bias could be that due to the large number of parameters that have to be estimated for the categorical features in the modelling step, random dependencies to the target variable are already caught well in the model and therefore these variables are used less frequently as splitting variables.
In (Hothorn et al., 2006) the unbiasedness of CTree is empirically investigated for cases, in which a strict separation between regressor variables and partitioning variables is kept. Therefore, the result shown here should not principally question the label, but it does not seem appropriate when there is an overlap or, as in our case, even congruence between the two roles.

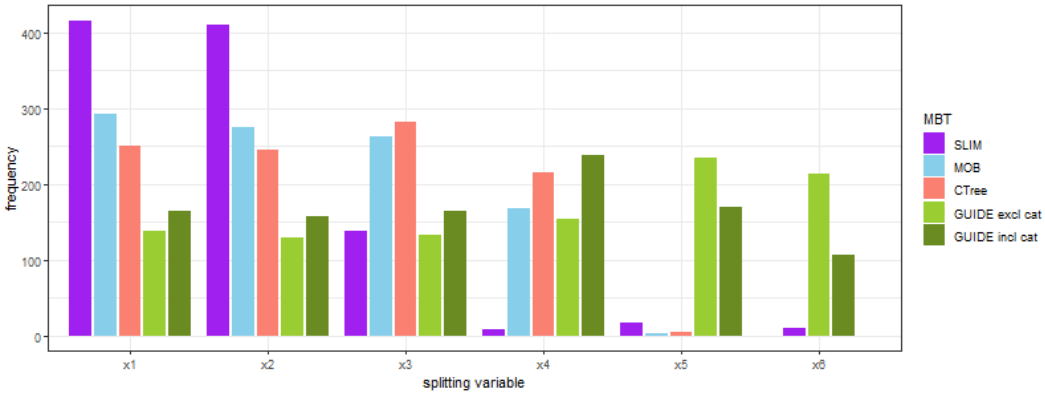

Figure 2: Simulated frequencies of selected splitting features for scenario independence mixed

For GUIDE two different variants were evaluated in this scenario. "GUIDE excl cat" corresponds to the original version, in which categorical features are only used as splitting and not as regressor variables. In "GUIDE incl cat", on the other hand, the categorical features are also used as regressors. The fact that the frequencies of the categorical variables are smaller in "GUIDE incl. cat" could be explained analogously to the selection bias of MOB and CTree.
The binary feature $\mathbf{x_4}$ is almost never chosen as a splitting variable with SLIM, in contrast to the other variables. However, this can be attributed to the selection bias in numerical features with different numbers of split points, as the binary variable can be seen as numerical variable with only one possible split point. What is more interesting is that the numerical variable $\mathbf{x_2}$ with 10 possible split points is chosen by SLIM much more

frequently as splitting variable than the variable $\mathbf{x}_6$ with 128 possible splits, so that the bias cannot be determined by the number of possible splits alone.
The difference between the categorical ($\mathbf{x}_5$ and $\mathbf{x}_6$) and the numerical variables can probably be explained in a similar way as with the other methods, but in reverse. With SLIM, the potential split is executed first and then models in the child nodes are fitted. If a numerical feature is used as partitioning variable, the parameters for all factor levels are estimated in both child models, whereby random dependencies with y can be modelled very accurately. If, on the other hand, splitting is done according to a categorical variable, only a subset of the factor levels can be used for modelling in both child nodes.
As a possible correction approach for the selection bias in SLIM, I investigated how the number of quantiles considered as potential split points affects the selection bias and the performance of SLIM. The simulation setup and results are shown in the appendix A.2. In summary, although selection bias can be reduced for numerical variables in the case of an independent target variable, this correction can lead to unpredictable biases if interactions are actually present. For this reason, the approach is not recommended. For all simulations in this chapter, each unique value was seen as a potential split point for SLIM.

4.3 Simulation interaction scenarios

While so far only the frequencies with which different feature types are selected in the case of independence were shown, it is empirically investigated in the following which feature types the different MBT algorithms tend to select when interactions are actually present. Four different scenarios are examined for this. These are, in a sense, pair comparisons of different feature types. In all scenarios, the data generating process consists of two pairs of interactions, whereby the interactions have the same strength. Each pair consists of a numerical variable that enters linearly into the interaction and another variable that splits the linear effect into two subgroups. In all four scenarios the numerical variables which define the linear effect are $\mathbf{x}_2; \mathbf{x}_4 \quad U(0; 1)$.
The features $\mathbf{x}_1$ and $\mathbf{x}_3$ are responsible for the subgroups. They are uniformly distributed on sets, which are given in Table 2, as well as the data generating processes.

scenario	$\mathbf{x}_1$	$\mathbf{x}_3$	$f(\mathbf{x})$
numerical vs numerical	$[0; 1]$	$f0; 0.1; \ldots; 0.9; 1g$	$1_{(\mathbf{x}_1 \quad mean(\mathbf{x}_1))}\mathbf{x}_2 + 1_{(\mathbf{x}_3 \quad mean(\mathbf{x}_3))}\mathbf{x}_4$
binary vs categorical	$f0; 1g$	$fa; b; c; d; e; fg$	$1_{(\mathbf{x}_1=0)}\mathbf{x}_2 + 1_{(\mathbf{x}_3 2 fa;b;cg)}\mathbf{x}_4$
numerical vs binary	$[0; 1]$	$f0; 1g$	$1_{(\mathbf{x}_1 \quad mean(\mathbf{x}_1))}\mathbf{x}_2 + 1_{(\mathbf{x}_3=0)}\mathbf{x}_4$
numerical vs categorical	$[0; 1]$	$fa; b; c; d; e; fg$	$1_{(\mathbf{x}_1 \quad mean(\mathbf{x}_1))}\mathbf{x}_2 + 1_{(\mathbf{x}_3 2 fa;b;cg)}\mathbf{x}_4$

Table 2: Simulation scenarios selection bias interaction

All experiments are repeated 1000 times. The error terms are set to $N(0; 0.1 \cdot sd(f(x))$ and the data generating processes are $y = f(x) + \varepsilon$.
First of all, there is the question of how selection bias should be defined in these scenarios. Transferring the definition from the independence case, one possibility would be to require that the frequency of being selected as a split variable for all features is 1/4. However, this does not take into account that splits according to x_1 and x_3, i.e. according to the split features that define the subgroups, produce a considerable greater improvement in performance than splits according to the smooth features x_2 and x_4. If x_1 and x_3 are chosen preferentially, one should not regard this as selection bias but as a good splitting strategy to get the smallest and best performing trees possible. What could be expected from an unbiased procedure, however, is that both interaction pairs are chosen equally often for the first split.

Figure 3: Simulated frequencies of selected splitting features for the four interaction scenarios

Figure 3 shows the frequencies of the first selected split variable in each scenario for each MBT. It can be seen that in none of the scenarios and for none of the MBTs the results seem to be equally distributed among all four features. As a general rule, it is noticeable that SLIM and GUIDE always prefer the subgroup-defining variables (x_1 and x_3) for splitting, while CTree always prefers the smooth variables. For MOB, the behaviour varies depending on the feature type.
In Scenario "numerical vs numerical", the distribution for MOB and CTree is at least evenly distributed between the two interactions pairs. One could therefore speak of unbiasedness here. However, while CTree only uses the linear features x_2 and x_4 as the first

split variable, MOB splits according to the subgroup-de ning features, which is preferable in practice. In GUIDE and SLIM, only the features x_1 and x_3 are selected, but there seems to be a preference for the variable x_1, which has a larger number of potential split points.

In the scenario "binary vs categorical", the distribution between the two interaction pairs seems to be roughly equal for SLIM, MOB and CTree. In contrast to the rst scenario, MOB does not split according to the subgroup-de ning variables but with regard to the linear features. In GUIDE, only the binary variable is selected, so there is a clear preference for the binary variable over the categorical variable.

In the remaining scenarios, a strong selection bias is evident in all MBTs. SLIM, for example, gives considerable preference to numerical subgroup-de ning variables over binary and categorical variables.

Obviously, these scenarios do not represent a complete overview of possible interactions. However, since MBTs can show their strengths especially in the presence of subgroup-speci c maine ects (i.e. good interpretability through small trees but good performance at the same time), only such scenarios are compared here. These simulation results should sensitise to the fact that in certain cases variables could only be selected as splitting variables because they have certain properties and not because their share in an interaction is actually the largest, even with the so-called unbiased methods.

Finally it should be noted that even if selection bias is present in the independence case, this does not necessarily lead to the wrong variable being selected for splitting. In particular, this can often be prevented by pruning. However, as far as could be observed here, selection bias is very di cult to control or predict and should therefore always be considered when using these algorithms.

5 Comparison of performance, interpretability and stability

Since it was shown in the previous chapter that selection bias is a problem for all MBTs and therefore the so-called unbiased methods should not be generally preferred to SLIM, the algorithms are compared on the basis of other important properties. Desirable properties of an MBT are high performance, good interpretability and stability. In this chapter, simulation scenarios and their results are described with which an empirical comparison of these properties for the different MBT algorithms can be made. In addition, a comparison of the performance and interpretability of SLIM MBTs is made when objectives of different complexity are used for modelling.

5.1 Evaluation measures

To evaluate the desired properties the following definitions and measurements are used.

5.1.1 Performance

A central aspect in the comparison of the different MBT algorithms is their performance. To find out how well an MBT algorithm performs as a stand-alone machine learning model, the MBT is fitted to the original response. In the following, the R^2 is used as performance measure of accuracy, which is defined as

$$R^2(f y; \hat{y} g) = 1 \quad \frac{\sum_{i=1}^{n} \hat{y}^{(i)} \quad y^2}{\sum_{i=1}^{n} (y^{(i)} \quad y)^2}; \tag{7}$$

where $\hat{y}^{(i)}; i = 1; :::; n$ are the predictions of the MBT model and y is the arithmetic mean of the target y.

To determine how well the respective MBT algorithm works as a surrogate model, i.e. how well the predictions of a black box model are replicated by the MBT, $R^2 \quad y^{bb}; \hat{y}$ is used to measure the so-called fidelity.

Accuracy and fidelity are measured on both training and test data in order to make a statement about the generalisation performance of the MBTs.

5.1.2 Interpretability

When evaluating the suitability of an MBT algorithm as a surrogate model, the trade-o between performance and interpretability is a key concern. While performance is easy to measure, the concept of interpretability is much more abstract. According to Doshi-Velez and Kim (2017) interpretability is de ned as the ability to explain or present something to a person in understandable terms. If, for di erent MBT algorithms to be compared, the models in the leaf nodes are de ned as equally complex (e.g. linear regression models), these do not have to be taken into account when comparing interpretability, but only the structure of the tree is relevant. Therefore the number of leaf nodes is used as the most important criterion for interpretability here. The underlying assumption is: the fewer leaf nodes a tree has, the easier its structure is to understand and to explain to another person.
As an additional measure, the number of di erent features used for splitting is included (if it varies between the algorithms). The idea here is that the splitting of the feature space is easier to understand if it is done in few dimensions.

5.1.3 Stability

A well-known weakness of recursive partitioning algorithms is that the resulting decision trees are often unstable, meaning that slight uctuations in the training data can lead to large di erences in the models (Fokkema, 2020). Depending on whether di erences between two decision trees mean di erent predictions for the same observations or di erent structural properties of a tree, such as the number of leaf nodes, we speak of semantic or structural instability (Wang et al., 2018). Since we are looking for MBTs that are easy to interpret, it is not enough to look at semantic stability. It would be desirable for an MBT that has been tted twice on slightly di erent training data to partition the feature space in the same way. To compare two trees that have the same number of leaf nodes, an additional data set (evaluation data) is used that is clustered according to the decision rules found for the training data. The subregions of an MBT are de ned as the clustering of the evaluation data set according to the decision rules learned by tting the MBT to a training data set. The assumption is now that if the clustering of the evaluation data is identical for both MBTs, the interpretation of the decision rules is also identical, even if the order of split features in the two MBTs is reversed and the rules therefore appear di erent at rst glance. In order to measure the similarity of subregions found by MBTs trained on slightly di erent data the Rand index (RI) (Rand, 1971) is used. The RI de nes the similarity of two clusterings A; B of n observations each by the proportion of the number of observation pairs that are either assigned to the same partition in both

clusterings (n_{11}) or to different partitions in both clusterings (n_{00}) measured against the total number of observation pairs.

$$RI(A;B) = \frac{n_{11} + n_{00}}{\binom{n}{2}} \tag{8}$$

(Gates and Ahn, 2017)
The higher the RI for a pair of trees with the same number of leaf nodes, the more similar the subregions defined by these trees and the more stable the MBT algorithm is in a concrete simulation scenario.
Besides the similarity of subregions with the same number of leaf nodes, the range of the number of leaf nodes is used as a measure of stability. The more the number of leaf nodes fluctuates between different simulation runs for an MBT algorithm, the more unstable it is considered to be.
According to (Hu et al., 2020), the stability problem with SLIM is less if it is used as a surrogate and not as stand-alone model on the original data. However, this is not shown in the paper. Therefore, for the following basic scenarios this claim is empirically investigated by comparing the stability of MBTs on the original data with the stability of MBTs used as surrogate models.

5.2 Basic scenarios

Three simple scenarios that differ mainly in the type of interaction(s) they contain (smooth interaction vs subgroup depending effects) are used for the evaluation. Based on these scenarios the different MBT algorithms are compared with respect to performance, interpretability and stability.

5.2.1 Simulation setting

Since the number of leaf nodes and the performance strongly depend on the prepruning value of impr for SLIM and GUIDE and alpha for MOB and CTree, the simulations are carried out for different values of these parameters for all three scenarios. In addition, two different sample sizesn are chosen. All MBT algorithms are fitted as standalone models on the original data and as surrogate models on a correctly specified linear model (lm) or GAM and on an XGBoost model with correctly specified interactions. The configurations of the XGBoost models are listed in the appendix in Table 24. Table 3 lists all varied factors and their levels. In total, there are 3 2 3 3 = 54 variants for each of the 4 MBT algorithms. Settings that are fixed in all variants are a maximum tree depth of 6

and a minimum node size (number of observation in a node) of 50. The experiments are repeated 100 times.

Varied factors	levels
Scenario	linear smooth, linear categorical, linear mixed
Sample size n	1500, 7500 ($\frac{2}{3}$ training, $\frac{1}{3}$ test data)
Prepruning parameters	alpha $\in \{0.05, 0.01, 0.001\}$; impr $\in \{0.05, 0.1, 0.15\}$
Usage of MBT	standalone, surrogate for lm, surrogate for XGBoost

Table 3: Simulation setting basic scenarios

Performance and Interaction measures are calculated in each simulation run. The RIs are calculated following the simulation based on pairwise comparisons of clusterings of an evaluation data set. The simulation setup is as follows:

1. simulate evaluation data (50000 observations) from the data generating process
2. for each simulation run in 1 : 100 runs:
    1. simulate data and perform train/test split
    2. train MBT on the training data, calculate performance measures on train and test set and extract the number of leaf nodes
    3. save the clustering of the whole evaluation data defined through the trained MBT
3. for each of the $(100(100 - 1)/2 = 4950$ MBT pairs
    1. sample 1000 observation ids from the evaluation data sets
    2. if both trees have the same number of leaf nodes, calculate the RI for the two clusterings of the evaluation data subset

5.2.2 Main results

The most important findings from the simulations of the basic scenarios are:

- SLIM and GUIDE perform better on subgroup detection tasks (Scenario linear categorical and linear mixed) in terms of better performance and interpretability than MOB and CTree. CTree is the weakest in this task.
- MOB and CTree produce more stable trees in scenarios with smooth interactions (Linear smooth and linear mixed). This is probably mainly due to the different pruning techniques. There is need for improvement for SLIM and GUIDE.

- The statement that stability increases when MBTs are used as surrogate models is confirmed in most results. An exception is the scenario linear categorical with a GAM as black box model, which does not model the subgroups well enough. As a result, SLIM and GUIDE can no longer detect them so well and the stability decreases considerably.
- A fundamental problem is that smooth interactions can often only be modelled well by a large number of binary splits, which makes MBTs difficult to interpret on such data. In such cases, MBTs are probably not the best choice for a surrogate model and other models such as GA2M (Lou et al., 2013) or compboost (Schalk et al., 2018) should be considered.

In the following, the basic simulation scenarios and a part of the detailed results are presented and discussed. Additional results can be found in the appendix A.3.1.

5.2.3 Linear smooth

Simulation scenario The data generating processes of the basic scenarios linear smooth includes one smooth two-way interaction and is defined as follows:

- $x_1, \ldots, x_3 \quad U(-1, 1)$
- $f(x) = x_1 + 4x_2 + 3x_2x_3$
- $\quad N(0, 0.1 \cdot sd(f(x))$
- $y = f(x) +$

Results For a comparison of the different MBT algorithms as standalone model on the scenario linear smooth the aggregated results are listed in Table 4 for sample size n = 1500.

model	impr	mean n leaves	n leaves min	n leaves max	mean R^2_{train}	sd R^2_{train}	mean R^2_{test}	sd R^2_{test}
SLIM	0.15	2.27	2	5	0.9584	0.0072	0.9557	0.0076
SLIM	0.10	10.09	5	15	0.9860	0.0057	0.9835	0.0059
SLIM	0.05	14.75	12	18	0.9909	0.0006	0.9884	0.0009
GUIDE	0.15	2.25	2	5	0.9582	0.0071	0.9555	0.0072
GUIDE	0.10	9.81	5	14	0.9859	0.0058	0.9834	0.0060
GUIDE	0.05	14.60	11	17	0.9907	0.0006	0.9883	0.0009
	alpha							
MOB	0.001	9.48	8	13	0.9898	7e-04	0.9876	0.0011
MOB	0.010	11.02	8	14	0.9902	7e-04	0.9879	0.0011
MOB	0.050	12.54	9	15	0.9906	6e-04	0.9882	0.0010
CTree	0.001	11.35	9	14	0.9900	6e-04	0.9881	0.0010
CTree	0.010	12.74	10	15	0.9904	6e-04	0.9884	0.0010
CTree	0.050	13.76	11	16	0.9905	6e-04	0.9885	0.0010
lm					0.9902	0.0006	0.9901	0.0008
XGBoost					0.9858	0.0008	0.9768	0.0018

Table 4: Mean simulation results on 100 simulation runs as standalone models on scenario linear smooth with sample size n = 1500 for different values of impr and alpha

The results of SLIM and GUIDE are very close to each other. Noticeable is the large variation in the number of leaf nodes and thus also in the performance of SLIM and GUIDE for different values of impr. The variation of alpha has a much smaller impact on the results for MOB and CTree.
Since the trade off between interpretability and performance must be taken into account when comparing the models, the performance of the different MBTs is compared depending on the number of leaf nodes. At $impr = 0.1$ and $alpha = 0.001$, all four models have a similar mean number of leaf nodes, which is why this setting is used for a more detailed comparison.

Figure 4: Pairwise plot of the number of leaf nodes vs the accuracy measure R^2 train scenario linear smooth with $n = 1500$, $alpha = 0.001$, $impr = 0.1$

In Figure 4 it can be seen that, as expected, the performance accuracy increases with increasing number of leaf nodes for all models. Noticeable, however, are the two different performance levels of SLIM and GUIDE for trees with 7 to 9 leaf nodes. This is due to the fact that trees with different symmetry properties are generated by SLIM and GUIDE in this range. While in the MBTs with higher performance the distribution of the observations on the different leaf nodes is approximately equal, in the group with lower performance one leaf node each is generated with almost half of the observations and the remaining half is distributed on the remaining leaf nodes. Examples for trees with different symmetry properties are shown in the Figures 22 and 23 in the appendix. There one can also see that the shown strongly asymmetric tree has only just fallen short of the prepruning criterion impr in the big node. The choice of the parameter impr therefore has a decisive influence on the resulting tree at this point.
Figure 5 shows the accuracy depending on the number of leaf nodes for the four different MBTs. The numbers below each box indicate how many of the 100 trees created with each of the four algorithms have the respective number of leaf nodes. For example, 8 of

Figure 5: Test accuracy R^2 vs number of leaf nodes scenario linear smooth with $n = 1500$, alpha $= 0.001$, impr $= 0.1$

the 100 SLIM trees have 10 leaf nodes and the corresponding box is created from these 10 results. Note that the numbers do not add up to a hundred, as there is also a small number of trees with leaf nodes outside the range shown here. The results of SLIM and GUIDE are divided into two groups for 8 and 9 leaf nodes for this plot. Trees with low symmetry are here defined as trees in which the largest leaf node contains at least 350 observations. The results between the two groups differ greatly, as already recognised in Figure 4. Figure shows that accuracy is very high in all four MBT. Over the range shown, MOB and CTree (from number of leaves = 9) have slightly higher performance than SLIM and GUIDE with the same number of leaf nodes, i.e. MOB and CTree achieve a better trade-off between performance and interpretability in this scenario as standalone model. As can be seen in Figure 24 in the appendix, SLIM and GUIDE have a higher average maximum leaf size (lower symmetry) over the entire comparable range of leaf nodes than MOB and GUIDE, which could be a reason for the slightly worse performance. However, when the models are used as surrogate models on lm predictions, the performance of the different MBTs with the same number of leaf nodes is very close (see appendix Figures 25, 26). Overall, it should be noted that the number of leaf nodes required to achieve a similar performance as the correctly specified linear model is very high, considering that it is actually a very simple data generating process. This is because the smooth interaction can only be well approximated through many binary splits. As in this scenario the number of possible split points is identically for all features, selection bias is not a problem.
As can be seen in Figure 4, the number of leaf nodes fluctuates considerably more for SLIM and GUIDE than for MOB and CTree. Figure 6 shows that this is also the case when the MBTs are used as surrogate models.
This observation is already a sign of lower stability of SLIM and GUIDE compared to

Figure 6: Number of leaf nodes for scenario linear smooth with $n = 1500, alpha = 0.001, impr = 0.1$

Figure 7: Rand Indices for scenario linear smooth with $n = 1500, alpha = 0.001, impr = 0.1$

MOB and CTree. Moreover, in Figure 7 the rand indices of the different algorithms are plotted for tree pairs with identical number of leaf nodes. The numbers below the boxes in this plot indicate the number of tree pairs (from 4950 pairs total), in which both trees have the according number of leaf nodes.
There one can see that even with identical numbers of leaf nodes, MOB and CTree seem to generate more stable trees.

5.2.4 Linear categorical

Simulation scenario Numerical and binary features with linear effects and subgroup specific linear effects:

- $x_1, x_2 \sim U(-1, 1), x_3 \sim Bern(0.5)$,
- $f(x) = x_1 - 8x_2 + 16x_2 1_{(x_3=0)} + 8x_2 1_{(x_1>mean(x_1))}$

$\hat{} \quad N(0; 0.1 \cdot sd(f(x))$

$\hat{} \quad y = f(x) +$

The scenario is based on a simulation example in (Herbinger et al., 2022).

Results In this scenario, the data-generating process is not determined by a smooth interaction, but can be fully described by main effect models in four subgroups. This would require a split at the empirical mean of feature x_1 and a split with respect to the binary variable x_3. SLIM and GUIDE differ greatly from MOB and CTree in their ability to find these splits, which can be seen in Table 5. The column share x_2 indicates what proportion of all split observations were split on average with respect to the variable x_2. For SLIM and GUIDE this value is 0 for all values of impr, which indicates, that all splits where performed regarding the variables x_1 and x_3. Consequently, the selection bias of SLIM does not lead to the binary variable x_3 not being selected as a splitting variable here. This corresponds to the results from chapter 4.3 in which is shown that SLIM prefers the variable that defines a subgroup in a two-way interaction as splitting variable. MOB and CTree, on the other hand, split almost only with respect to the variable x_2 and therefore achieve a worse mean performance than SLIM and GUIDE (except for impr = 0.15) despite a considerable higher number of leaf nodes.

model	impr	n leaves	n leaves min	n leaves max	R^2_{train}	sd R^2_{train}	R^2_{test}	sd R^2_{test}	share x_2
SLIM	0.15	2.00	2	2	0.8272	0.0071	0.8250	0.0111	0.0000
SLIM	0.10	3.99	3	4	0.9884	0.0069	0.9870	0.0078	0.0000
SLIM	0.05	4.00	4	4	0.9891	0.0010	0.9878	0.0028	0.0000
GUIDE	0.15	2.00	2	2	0.8272	0.0071	0.8250	0.0111	0.0000
GUIDE	0.10	3.99	3	4	0.9884	0.0069	0.9870	0.0078	0.0000
GUIDE	0.05	4.00	4	4	0.9891	0.0010	0.9878	0.0028	0.0000
	alpha								
MOB	0.001	12.77	10	15	0.9661	0.0083	0.9558	0.0091	0.9095
MOB	0.010	14.40	12	16	0.9736	0.0067	0.9641	0.0076	0.8761
MOB	0.050	14.85	13	17	0.9747	0.0063	0.9654	0.0071	0.8682
CTree	0.001	11.87	10	14	0.9484	0.0030	0.9390	0.0052	0.9976
CTree	0.010	12.82	11	15	0.9498	0.0030	0.9404	0.0051	0.9939
CTree	0.050	13.61	11	16	0.9508	0.0029	0.9411	0.0048	0.9923
lm					0.9702	0.0018	0.9694	0.0029	
XGBoost					0.9876	0.0015	0.9778	0.0031	

Table 5: Mean simulation results on 100 simulation runs as stand alone models on scenario linear categorical with sample size $n = 1500$ for different values of impr and alpha

In contrast to the scenario linear smooth, the variation in the number of leaf nodes for fixed values of impr is also small for SLIM and GUIDE. This indicates a high stability of SLIM and CTree on this scenario.

Figure 8 shows the performance by number of leaf nodes for impr = 0.05 and alpha = 0.001. There one can see again that SLIM and GUIDE achieve a very high R^2 despite the low number of leaf nodes.

It is also noticeable that MOB performs better than CTree with the same number of leaf nodes. This could be because, according to (Schlosser et al., 2019), the M-fluctuation

Figure 8: Test accuracy R^2 vs number of leaf nodes scenario linear categorical with $n = 1500; alpha = 0:001; impr = 0:05$

test used in MOB has a higher ability in detecting abrupt changes than the linear test statistic used in CTree, which was also seen in chapter 4.3.
If SLIM and GUIDE are tted as surrogate models on the predictions of a GAM (linear main e ects, tensorproduct interaction), the number of leaf nodes increases and the variability also increases, as can be seen in Table 27 in the appendix. This indicates that the interactions with the GAM are not tted well enough, despite a $R^2_{test} = 0:97$ see Table 5). The statement that SLIM is more stable when used as a surrogate model than when used as a standalone model (Hu et al., 2020) is therefore not true in this case. Nevertheless, the mean performance of SLIM and GUIDE is better than that of the other MBTs, as can be seen in Table 27 in the appendix.

5.2.5 Linear mixed

Simulation scenario Numerical and binary features with linear e ects, subgroup dependent and smooth two- and three-way interactions:

- $x_1; x_2 \quad U(\ 1; 1), x_3; x_4 \quad Bern(0:5)$
- $f(c) = 4x_2 + 2x_4 + 4x_2x_1 + 8x_21_{(x_3=0)} + 8x_1x_21_{(x_4=1)}$
- $N(0; 0:1 \ sd(f(x))$
- $y = f(x) +$

Results The scenario linear mixed contains both smooth interactions that can only be handled by splits with respect to the numerical variables x_1 and x_2, and subgroups de ned by the binary variables x_3 and x_4. The average proportion of splits carried out with respect to the two numerical variables is shown in Table 6. For $alpha = 0:001$ and

model	impr	n leaves	n leaves min	n leaves max	R^2_{train}	sd R^2_{train}	R^2_{test}	sd R^2_{test}	share x_1 x_2
SLIM	0.15	3.40	2	11	0.8853	0.0309	0.8771	0.0336	0.9592
SLIM	0.10	13.07	8	16	0.9801	0.0073	0.9743	0.0083	0.8793
SLIM	0.05	14.73	13	16	0.9830	0.0019	0.9774	0.0029	0.8794
GUIDE	0.15	3.29	2	11	0.8839	0.0305	0.8758	0.0329	0.9598
GUIDE	0.10	12.52	7	15	0.9789	0.0087	0.9732	0.0091	0.8581
GUIDE	0.05	14.19	12	16	0.9822	0.0024	0.9766	0.0032	0.8516
	alpha								
MOB	0.001	14.52	13	17	0.9802	0.0017	0.9725	0.0027	0.9679
MOB	0.010	14.73	13	17	0.9803	0.0017	0.9726	0.0028	0.9672
MOB	0.050	14.80	13	17	0.9804	0.0017	0.9727	0.0028	0.9664
CTree	0.001	14.80	12	17	0.9802	0.0017	0.9731	0.0023	0.9989
CTree	0.010	14.98	13	17	0.9802	0.0017	0.9731	0.0023	0.9978
CTree	0.050	15.03	13	17	0.9802	0.0017	0.9731	0.0023	0.9978
XGBoost					0.9859	0.0014	0.9682	0.0042	
lm					0.9902	0.0006	0.9898	0.0008	

Table 6: Mean simulation results on 100 simulation runs as stand alone models on scenario linear mixed with sample size $n = 1500$ for different values of impr and alpha

impr $= 0.05$ the mean number of leaf nodes is similar for all four MBT algorithms. The table shows that the average performance of SLIM and GUIDE for this setting is higher than that of the other two MBTs and at the same time the mean proportion of splits with respect to the numerical variable is lower. This means that SLIM and GUIDE use the binary variables more often to reveal the subgroups defined by them, while MOB and even more pronounced CTree split almost exclusively along the numerical features and thus perform worse despite having the same mean number of leaf nodes.
In this example, SLIM and GUIDE achieve a better trade-off between performance and interpretability than MOB and CTree. In Figure 9 and Figure 10 this is broken down in more detail.

Figure 9: Test accuracy R^2 vs number of leaf nodes scenario linear mixed with $n = 1500$, alpha $= 0.001$, impr $= 0.05$

For the comparison of stability one can look at Figure 11. On the one hand, it can be seen that MOB and CTree seem to be somewhat more stable than SLIM and GUIDE with the same number of leaf nodes, but the difference is not great. Furthermore, it can be seen that in most cases the stability of trees with the same number of leaf nodes is higher when they are used as surrogates than when they are used as standalone models.

Figure 10: Share of split observations by features x_3, x_4 vs number of leaf nodes scenario linear mixed with $n = 1500, alpha = 0.001, impr = 0.05$

Figure 11: Rand Indices for scenario linear mixed with $n = 1500, alpha = 0.001, impr = 0.05$

Here, the assertion of Hu et al. (2020) that SLIM is more stable when used as a surrogate is true and also applies to the other algorithms.

5.3 Linear smooth with noise features

This scenario examines how noise features that have no influence on y affect the MBTs. The scenario linear smooth is used as a basis, to which six noise variables are added. In addition to the usual models, SLIM models are fitted with lasso regression models (Tibshirani, 1996; Friedman et al., 2010). Lasso models allow the fitting of sparse models, i.e. a feature selection in the nodes takes place automatically. The strength of the feature selection depends strongly on the penalisation parameter. Three different variants are used to choose this parameter. In all SLIM lasso MBTs, the penalisation parameter is selected using the BIC criterion (Sabourin et al., 2015). However, in the case $df = 3$ or $df = 2$, the additional restriction is defined that the effective degrees of freedom (df) must not exceed this value. This enforces especially sparse models.

- $x_1, ..., x_{10} \sim U(-1, 1)$
- $f(x) = x_1 + 4x_2 + 3x_2x_3$
- $\varepsilon \sim N(0, 0.1 \cdot sd(f(x)))$
- $y = f(x) + \varepsilon$

The MBTs are fitted as standalone models and surrogates on lm predictions on a data set with sample size $n = 3000$ (2000 training, 1000 test observations) using the pruning parameter settings $alpha = 0.001$ and $impr = 0.1$. The simulation is repeated 250 times. The aim of the simulation is to investigate whether the noise variables are incorrectly chosen as splitting variables.

black box	MBT	freq x_{noise}	share x_3	n leaves	n l min	n l max	R^2_{train}	sd R^2_{train}	R^2_{test}	sd R^2_{test}
standalone	SLIM	0.072	0.4654	11.988	5	17	0.9880	0.0048	0.9854	0.0049
standalone	SLIM lasso	0.092	0.4655	11.048	5	16	0.9871	0.0049	0.9852	0.0051
standalone	SLIM lasso df 3	0.028	0.5182	9.732	4	14	0.9863	0.0046	0.9848	0.0050
standalone	SLIM lasso df 2	0.028	0.9990	9.648	4	15	0.9864	0.0044	0.9852	0.0047
standalone	GUIDE	0.104	0.4780	11.788	5	16	0.9880	0.0047	0.9854	0.0048
standalone	MOB	0.000	0.5038	11.096	8	14	0.9901	0.0005	0.9878	0.0007
standalone	CTree	0.000	0.5087	13.140	10	16	0.9904	0.0004	0.9882	0.0007
standalone	lm						0.9901	0.0004	0.9901	0.0006
lm	SLIM	0.000	0.4900	14.036	8	16	0.9987	0.0018	0.9984	0.0019
lm	SLIM lasso	0.000	0.4574	13.736	8	16	0.9985	0.0023	0.9982	0.0027
lm	SLIM lasso df 3	0.000	0.5076	14.008	8	16	0.9985	0.0023	0.9983	0.0026
lm	SLIM lasso df 2	0.000	1.0000	11.160	5	14	0.9979	0.0018	0.9977	0.0020
lm	GUIDE	0.000	0.4909	14.096	8	16	0.9988	0.0018	0.9984	0.0019
lm	MOB	0.000	0.5073	15.960	15	16	0.9995	0.0000	0.9993	0.0001
lm	CTree	0.000	0.5003	15.564	13	16	0.9994	0.0001	0.9992	0.0001

Table 7: Mean simulation results of stand alone model and surrogates on lm predictions on scenario linear smooth with noise features

Table 7 shows the mean simulation results. freq x_{noise} is the proportion of trees in which at least one of the noise variables is used as a splitting variable. It can be seen that the noise variables are only used as splitting variables for the specified settings if the MBTs are used as standalone models. However, MOB and CTree do not select false variables

even then. For the SLIM MBTs it can be seen that freqx_{noise} decreases with a strong penalty (df = 2 and df = 3). GUIDE, on the other hand, has the most problems with the noise variables and the highest value for freqx_{noise}. An interesting e ect of using a lasso model with a maximum of 2 e ective degrees of freedom is that for the splitting almost exclusively variablex_3 is used, although otherwisex_2 and x_3 are mostly chosen similarly often. This could be becausex_3 does not have a main e ect on the response, butx_2 and x_1 do. Due to the strong penalisation, only these two features are used for modelling in the leafs. Sincex_3 is not included in the modelling, it is instead more often used as a splitting variable. The interpretability of this MBT is thus increased on the one hand by the very sparse models in the leaf nodes and on the other hand by the fact that the splits are almost only carried out with respect tox_3, which is easier to keep track of than if the feature space is split with respect to several di erent variables. The average number of leaf nodes for SLIM with lasso df 2 is smaller than for the standard SLIM MBT, which is why the performance cannot be directly compared. However, it can be seen that the training performance of SLIM with lasso df 2 is further below that of the standard SLIM than the test performance. This indicates that with the strong penalisation, over tting can also be reduced. In the appendix in gures 27 and 28 there is a more detailed presentation of the individual results. It is particularly noticeable that, as with the basic scenario linear smooth, all SLIM MBTs and GUIDE generate some strongly asymmetrical trees, which then have a considerable poorer performance.

5.4 Linear smooth with correlated features

While in the basic scenarios the features are independent of each other, this scenario examines how a correlations between features in uence the MBTs. For this purpose, the "linear smooth" scenario is modi ed so that there is a correlation betweenx_1 and x_3.

- $x_1, \ldots, x_3 \sim U(-1, 1), Corr(x_1, x_3) = \rho_{13} \in 0.1, 0.5, 0.9$
- $f(x) = x_1 + 4x_2 + 3x_2x_3$
- $\varepsilon \sim N(0, 0.1 \cdot sd(f(x)))$
- $y = f(x) + \varepsilon$

It is to be examined whether the modi cation causes the featurex_1 to be incorrectly selected as a splitting variable.

For the simulation, the MBTs are tted on a data set with sample size$n = 1500$ (1000 training, 500 test observations) using the pruning parameter settings$alpha = 0.001$ and $impr = 0.01$. The parameters are chosen so that the average number of leaf nodes is

similar for the different MBTs. The trees are fitted both as standalone models on the original data and as surrogate models on correctly specified linear models. The simulation is repeated 250 times.

black box	MBT	ρ_{13}	x_1 frequency	n leaves	n leaves min	n leaves max	R^2_{train}	sd R^2_{train}	R^2_{test}	sd R^2_{test}
standalone	SLIM	0.1	0.088	10.012	3	16	0.9863	0.0055	0.9836	0.0061
standalone	SLIM Ridge	0.1	0.096	9.756	4	15	0.9860	0.0057	0.9834	0.0063
standalone	GUIDE	0.1	0.052	9.792	3	15	0.9862	0.0057	0.9836	0.0061
standalone	MOB	0.1	0.000	9.392	8	12	0.9898	0.0006	0.9876	0.0010
standalone	CTree	0.1	0.000	11.312	9	14	0.9901	0.0006	0.9881	0.0010
standalone	SLIM	0.5	0.152	10.052	3	15	0.9867	0.0050	0.9841	0.0055
standalone	SLIM Ridge	0.5	0.112	9.612	3	15	0.9862	0.0051	0.9837	0.0056
standalone	GUIDE	0.5	0.120	9.776	3	15	0.9870	0.0046	0.9846	0.0052
standalone	MOB	0.5	0.000	9.048	8	11	0.9899	0.0006	0.9878	0.0010
standalone	CTree	0.5	0.000	10.776	9	14	0.9901	0.0006	0.9882	0.0010
standalone	SLIM	0.9	0.328	10.216	4	16	0.9873	0.0043	0.9850	0.0046
standalone	SLIM Ridge	0.9	0.292	9.968	4	16	0.9872	0.0043	0.9848	0.0048
standalone	GUIDE	0.9	0.448	9.996	3	17	0.9874	0.0042	0.9852	0.0047
standalone	MOB	0.9	0.048	8.704	8	11	0.9899	0.0005	0.9879	0.0010
standalone	CTree	0.9	0.084	10.280	8	14	0.9901	0.0005	0.9882	0.0009
lm	SLIM	0.1	0.000	12.060	5	16	0.9967	0.0047	0.9959	0.0055
lm	SLIM Ridge	0.1	0.000	11.388	5	16	0.9962	0.0049	0.9954	0.0057
lm	GUIDE	0.1	0.000	12.132	5	16	0.9967	0.0047	0.9960	0.0055
lm	MOB	0.1	0.000	15.764	14	16	0.9994	0.0001	0.9993	0.0001
lm	CTree	0.1	0.000	15.256	13	17	0.9993	0.0001	0.9992	0.0001
lm	SLIM	0.5	0.000	12.008	5	16	0.9970	0.0042	0.9963	0.0050
lm	SLIM Ridge	0.5	0.000	11.584	4	16	0.9967	0.0043	0.9960	0.0051
lm	GUIDE	0.5	0.000	11.968	5	16	0.9974	0.0036	0.9968	0.0043
lm	MOB	0.5	0.000	15.768	14	16	0.9995	0.0001	0.9994	0.0001
lm	CTree	0.5	0.000	15.220	13	17	0.9994	0.0001	0.9993	0.0001
lm	SLIM	0.9	0.000	12.212	5	16	0.9977	0.0033	0.9972	0.0039
lm	SLIM Ridge	0.9	0.000	11.772	5	16	0.9975	0.0034	0.9970	0.0039
lm	GUIDE	0.9	0.000	12.364	5	16	0.9978	0.0033	0.9973	0.0038
lm	MOB	0.9	0.004	15.752	14	17	0.9996	0.0001	0.9995	0.0001
lm	CTree	0.9	0.000	15.144	14	17	0.9995	0.0001	0.9994	0.0001

Table 8: Mean results correlated data

Table 8 shows the average performance and the number of leaf nodes of MBTs as well as the frequency of trees where x_1 is wrongly chosen as the splitting variable. If the MBTs are used as standalone, SLIM and GUIDE choose feature x_1 as splitting variable already at a very low correlation of 0.1 in some cases. As the correlation increases, this frequency increases. MOB and CTree, on the other hand, only use the variables actually involved in the interaction for the splits at $\rho \in \{0.1, 0.5\}$, but at a very high correlation rho = 0.9 they also include x_1 for the split in a small number of cases. Why CTree and MOB react less sensitively to correlated features could not be clarified here. It seems that the scores reflect well which feature is actually responsible for the instability. Since in regression tasks the use of ridge regression is recommended for correlated features (McDonald, 2009), SLIM is also fitted with ridge regression models in the nodes. The choice of the penalisation parameter is made in each recursion step by minimising the BIC. Instead of the SSE, the penalised SSE is used as objective function. However, this does not result in any systematic improvement. If the MBTs are used as surrogates for the linear regression predictions instead of standalone models, there are hardly any false splits in this scenario. So it seems that the problem of correlated features is reduced when the models are used as surrogates.

5.5 Nonlinear e ects

While the main goal so far has been to compare the di erent MBT algorithms on simple scenarios with linear main e ects, the following scenario will investigate how the choice of di erent objective functions a ects the interpretability and performance of SLIM MBTs when nonlinear main e ects are included in the data generating process. As it is very exible in the selection of the objective, only the SLIM algorithm is used. The data is de ned as follows:

- $x_1, \ldots, x_5 \sim U(-1, 1), x_6 \sim Bern(0.5)$,
- $f(x) = x_1 + 2x_2^2 + x_3 \log(abs(x_3)) + x_4 x_5 + x_1 x_4 1_{(x_6=0)}$
- $\sim N(0, 0.1 \cdot sd(f(x)))$
- $y = f(x) +$

SLIM is tted on the data with four di erent objectives.

1. linear regression model (lm)
2. polynomial regression of degree 2 with lasso penalisation (penalised poly)
3. linear regression with unpenalised linear B-spline transformations of the features (B-splines)
4. Generalized additive models with integrated smoothness estimation (Wood, 2011) (GAM)

Since the models in the leaf nodes are of di erent complexity for the di erent objectives, it is not su cient in this case to use only the number of leaf nodes as a measure of interpretability. In addition, the following criteria are considered:

- e ective degrees of freedoms of the leaf node models (lm and penalised poly), as sparse models are easier to interpret
- proportion of observation splits where the split is actually due to an interaction rather than an incompletely modelled main e ect
- interpretable weights versus merely visually interpretability

setting	impr	R^2
setting 1	0.05	0.9
setting 2	0.10	1

Table 9: Prepruning settings for scenario nonlinear e ects

In order to enable a comparison of the interpretability, theR^2 is used for prepruning in this scenario in addition to the prepruning withimpr . As soon as theR^2 exceeds a certain value in a node, it is no longer split.
Trees with two di erent prepruning settings are trained on the data
The simulated data contains 3000 observations (2000 train/ 1000 test). The SLIM MBTs are tted both as standalone models and as surrogate models on predictions of an XGBoost model. The hyperparameter of the XGBoost model are listed in the appendix in Table 31.

		n leaves			n split features					
blackbox	model	n leaves	n l min	n l max	n split feat	n sf min	n sf max	% main e ect	df	sd df
standalone	lm	17.48	13	28	4.76	4	6	0.4525	6.5763	0.1578
standalone	penalised poly	4.36	3	7	2.16	2	3	0.3605	8.1847	0.8960
standalone	B-Splines	2.04	2	3	1.00	1	1	0.0000		
standalone	GAM	2.04	2	3	1.00	1	1	0.0000		
XGBoost	lm	19.82	2	33	4.76	1	6	0.6125	6.8671	0.1438
XGBoost	penalised poly	3.58	2	7	2.08	1	4	0.3880	8.5499	1.1147
XGBoost	B-Splines	1.86	1	3	0.84	0	1	0.0000		
XGBoost	GAM	1.90	1	4	0.88	0	2	0.0000		

Table 10: Mean interpretability simulation results for scenario nonlinear e ects variant 1

Table 10 lists the mean results regarding the interpretability of the di erent SLIM trees for setting 1, in which the prepruning in a node already applies at aR^2 of 0.9. For SLIM with linear regression models in the nodes, this results in large trees which can hardly be used for interpretation purposes. In addition, in average 45% (61% if used as surrogate) of all split observations are not split with respect to an interaction, but with respect to an insu ciently modelled main e ect. Moreover, in some simulation runs all features are used as splitting variables, which further increases the lack of interpretability. The fact that the models in the leaf nodes are very simple can hardly compensate for this.
SLIM in conjunction with penalised polynomial regression in the nodes, on the other hand, achieves comparable performance with far fewer splits. There are also considerable fewer di erent features used for splitting, but the proportion of observations where the splits are performed with respect to a main e ect variable is still high, albeit better than with SLIM lm.
Both SLIM with linear B-Spline transformed variables and GAMs require on average only two leaf nodes, i.e. one split, to achieve aR^2 of 0:9. The models in the leaf nodes can only be interpreted visually, but since the number of models is very small, the degree of interpretability is comparatively high. Moreover, these models actually only split by

interactions, as the nonlinear main e ects are already modelled su ciently well.
If it is not explicitly required that the model weights are directly interpretable and it should also be ensured that they are actually split according to interactions, models with splines are the best choice in this case. GAMs are preferable to unpenalised B-Splines in terms of their generalisation error, but the computational e ort is much higher.

black box	model	R2 train	R2 train sd	R2 test	R2 test sd	time
standalone	lm	0.9243	0.0062	0.9113	0.0054	53.9438
standalone	penalised poly	0.9204	0.0092	0.9161	0.0108	46.6282
standalone	B-Splines	0.9282	0.0036	0.9195	0.0049	15.2456
standalone	GAM	0.9253	0.0034	0.9226	0.0048	417.2874
XGBoost	lm	0.9200	0.0228	0.9023	0.0190	55.7771
XGBoost	penalised poly	0.9213	0.0079	0.9150	0.0087	35.4093
XGBoost	B-Splines	0.9382	0.0118	0.9296	0.0120	11.4670
XGBoost	GAM	0.9348	0.0118	0.9289	0.0117	384.7403
XGBoost	XGBoost	0.9386	0.0295	0.9199	0.0362	3.1163

Table 11: Mean performance simulation results for scenario nonlinear e ects variant 1

The results for setting 2 are listed in the appendix in the tables 32 and 33

6 Insurance use case

In the following, SLIM is used in various con gurations as a surrogate for modelling the bene t present values of two ctitious insurance tari s from the TRAIL.X (TRustworthy Arti cial Intelligence in Life Insurance) research project (msg insurit, n.d.). The results are interpreted and the delity is compared with the MBT algorithms GUIDE, MOB and CTree.

6.1 Data set K2204

The data set K2204 contains data for a (ctitious) endowment insurance tari . With this tari , a single bene t is paid both in the event of survival and death of the policyholder. The data set includes the features sex (1 = male, 0 = female), age and duration and the two targets bene t present value (BPV) and premium present value (PPV). The targets were modelled using two di erent black box models. In the following, only the BPV is used. The results for the PBV are very similar (interpreted the other way round) and can be found in the appendix. Since the data set with 5994 observations is rather small, all observations were used for training. For this data set therefore only training performance of the surrogate models is considered. A special characteristic of K2204 is a correlation between the features age and duration, as can be seen in Figure 12. When interpreting a MBT for this data set, it must therefore be taken into account that a split with regard to one of the features can have an in uence on the value range of the other feature.

Figure 12: Features age and duration in the K2204 data set

6.1.1 Shallow MBTs with linear models

In a rst step, SLIM is tted as surrogate to the black box predictions of BPV (BPV_pred) with linear regression models in the nodes. The maximum depth is set to 3 and an

improvement in the objective ($impr$) of at least 0.1 of the previous improvement is set as prepruning parameter. The resulting tree is shown in Figure 13.

Figure 13: SLIM tree for K2204 with linear models

Basic observations across all subregions are:

- Gender male has a positive effect on BPV_{pred}
- age has a positive effect on BPV_{pred}
- duration has a negative effect on BPV_{pred}

The strength of the effects, however, differs in the different subregions found by SLIM. The five leaf nodes can be roughly divided into two regions with similar effects:

- Region 1 (Nodes 2,8): High duration (30) or high age ($>$ (48) and medium duration (between 13 and 30)
- Region 2 (Nodes 5,6,7): Low - medium duration with low age or high age with low duration (12)

In Region 1 sex male and age seem to have a higher positive effect on BPV_{pred} than in region 2. The negative effect of duration, on the other hand, is smaller in region 1. This indicates a nonlinearity of duration.
If SLIM is fitted as a standalone model instead of a surrogate model in the same configuration, the differences in the split points are very small. This indicates that the black box

model captures the underlying relationships very well. The corresponding tree is shown in the appendix in Figure 29.

In the following, the delity of the di erent MBT with algorithms with linear models is compared. For this purpose, all four algorithms were tted with a maximum depth of 3. For SLIM and GUIDE, impr is set to 0.05 and for MOB and CTreealpha is set to 0.05. Furthermore, a minimum node size of 200 observations is required. The MBTs are compared with a baseline model, which is a linear regression model on the entire feature space. In addition to theR^2 and the MSE, the mean absolute error (MAE) and the maximum absolute error (max AE) are included as measures of delity. The max AE is particularly important here, as it is strictly regulated in order not to discriminate against any individual. The results are listed in Table 12. It shows that all MBTs achieve considerable improvement over the baseline model.

	R^2	MSE	MAE	max AE	n leaves
linear baseline model	0.985101	0.000201	0.011393	0.064709	1
SLIM	0.999233	0.000010	0.002272	0.020526	8
GUIDE	0.999276	0.000010	0.002142	0.020526	8
MOB	0.998527	0.000020	0.003149	0.024504	8
CTree	0.995091	0.000066	0.005740	0.042931	8

Table 12: Fidelity of K2204 linear baseline model and linear MBTs

GUIDE achieves the best performance slightly ahead of SLIM. CTree obtains the worst performance. Since all algorithms generate MBTs with the same number of leaf nodes, the di erence in performance must be explained by di erent split features or points. Table 13 lists the share of observations that were split with respect to the di erent features.

	age	duration	sex
SLIM	0.28	0.67	0.05
GUIDE	0.28	0.72	0.00
MOB	0.10	0.77	0.13
CTree	0.00	0.96	0.04

Table 13: Share of observations split by the di erent features K2204 linear MBTs

It is noticeable that SLIM and GUIDE split by age more often than the other two algorithms. Furthermore, it should be considered that particularly the SLIM tree could be in uenced by selection bias and therefore sex with only one possible split point could be disadvantaged as a splitting variable. SLIM actually chooses sex less often as splitting variable than MOB and CTree, but sex is also rarely chosen by GUIDE and no selection bias in the independence case between numerical variables with di erent numbers of splitting variables could be found in GUIDE in chapter 4. Moreover, the e ect of selection bias would not be so dramatic here, since the two variables with which sex could interact - and probably does to a small extent, as the varying parameter estimates indicate - are

chosen as splitting variables. The small interaction of sex with age and duration is thus reflected in the model, even if sex is not chosen as a splitting variable.

6.1.2 MBTs with B-spline models

In order to better capture nonlinearities and thus reduce the risk of splitting with respect to nonlinearities instead of interactions, all MBT algorithms are fitted with B-spline transformed feature age and duration as surrogate models. Two different maximal depths are used, 3 for interpretable shallow trees and 6 for deep trees with higher fidelity. The settings for alpha, improve and minimum node size remain the same as for the MBTs with linear models. Again, a baseline model is fitted, in this case a regression model with the feature sex and B-spline transformed features age and duration on the entire feature space.
Figure 14 plots the prediction of the baseline B-spline model and the two B-spline SLIM surrogates against BPV$_{pred}$ to visualise performance improvement. This shows a considerable improvement from the baseline model to the shallow tree. For the deep tree, the observations seem to be even somewhat closer to the identity line.

Figure 14: B-spline surrogate predictions vs. BPV$_{pred}$ for K2204

The fidelity results for all B-spline surrogates are listed in Table 14 .

	R^2	MSE	MAE	max AE	n leaves
B-spline baseline model	0.9943311	7.63e-05	0.0067811	0.0378568	1
SLIM shallow	0.9994176	7.80e-06	0.0018244	0.0167730	8
GUIDE shallow	0.9993900	8.20e-06	0.0019011	0.0165545	8
MOB shallow	0.9992115	1.06e-05	0.0022236	0.0182034	8
CTree shallow	0.9990918	1.22e-05	0.0024194	0.0183906	8
SLIM deep	0.9997514	3.30e-06	0.0010766	0.0126778	21
GUIDE deep	0.9997301	3.60e-06	0.0011730	0.0116453	20
MOB deep	0.9996858	4.20e-06	0.0013448	0.0119785	21
CTree deep	0.9997091	3.90e-06	0.0012936	0.0123686	20

Table 14: Fidelity of K2204 B-spline baseline model and B-spline MBTs

The improvement of the shallow MBTs over the baseline model is large, but not as substantial as in the trees with linear models without B-spline transformations. This is

probably due to the fact that the splits in the MBTs with linear models also handled nonlinearities that could not be adjusted in the linear baseline model. In the baseline model with B-spline transformations, on the other hand, the nonlinearities are already taken into account and the splits then actually capture primarily the interactions, which is what is desired. The deeper splitting improves the performance of all MBTs considerably. The MAE of the deep trees, for example, is only 52%-61% of the MAE of the shallow trees. CTree achieves the greatest improvement and obtains better delity in the deep trees than MOB (except for max AE). The better performance of all MBTs, however, comes at the cost of interpretability. Additionally, there is an increased risk of over tting.

	age	duration	sex
SLIM shallow	0.38	0.62	0.00
GUIDE shallow	0.30	0.70	0.00
MOB shallow	0.08	0.84	0.08
CTree shallow	0.23	0.71	0.07
SLIM deep	0.35	0.60	0.04
GUIDE deep	0.20	0.78	0.02
MOB deep	0.08	0.76	0.16
CTree deep	0.20	0.67	0.13

Table 15: Share of observations split by the di erent features K2204 B-spline MBTs

Table 15 shows the proportions of observations that were split according to the di erent features. For SLIM, GUIDE and MOB the share results of the shallow trees are similar to the MBTs with linear models. Shallow CTree, on the other hand, selects age in 23% of split observations as splitting variable, whereas it was not used at all for splitting in CTree with linear models. It performs the worst in terms of delity in this case as well, but not as considerable as in the previous setting. With the deep trees, the values for share and also the delity values of the di erent MBTs move closer together.
For the interpretation, the shallow SLIM B-spline tree shown in Figure 15 is analysed in more detail.
In order to investigate the e ects of the splits on the feature e ects more closely, the splits with regard to duration and age are analysed separately. The nodes 1,5,13 and 14 together comprise the entire feature space, whereby the sub-regions are determined by splits with regard to the feature duration. Figure 16 shows the input-output relation (feature e ects) of the features estimated in the B-spline models in the di erent nodes. Note that the curves are centred.
As with SLIM with linear models, it can be seen that the positive e ect of sex male seems to increase with increasing duration. The same applies to age. The seemingly negative e ect of age at low duration and high age is probably due to extrapolation, as there are only few observations in this area, which is shown in Figure 30 in the appendix.
With the feature duration, it can be seen that the negative e ect seems to decrease with

Figure 15: SLIM tree for K2204 with B-spline models

Figure 16: Input-output relation of features in nodes split by duration for SLIM tree with B-splines and depth 3

increasing duration, which is again just a nonlinearity.

To investigate the e ect of splits with respect to feature age, on the one hand nodes 4,7 and 8 are compared, which cover the feature space for duration 25, and on the other

hand nodes 11 and 12, which cover the feature space for < 25 duration $<= 40$. In the Figures 17 and 18 the corresponding input-output relations of the features are shown.

Figure 17: Input-output relation of features in nodes with duration < 25 split by age for SLIM tree with B-splines and depth 3

Figure 18: Input-output relation of features in nodes with $25 <$ duration $<= 40$ split by age for SLIM tree with B-splines and depth 3

Again, the interpretation is consistent with the results from SLIM with linear models. The positive effect of sex and age is increased by increasing age, while the negative effect of duration is reduced. The slightly negative effect of age at high age and low duration should again be viewed with caution and is probably due to the poor data situation in this area.

Finally, a deep SLIM tree with B-spline models is used as a standalone model for BPV and its accuracy is compared to the accuracy of the black box model. The result is shown in Table 16. The accuracy of SLIM is worse than that of the black box model for all evaluation measures. However, the difference is most apparent for the max AE. While the max AE is explicitly minimised in the black box model, the MSE is minimised with SLIM. In general, a different loss function would also be possible with SLIM. In this case, either the max AE could be minimised only in the split selection, or also in the modelling in the nodes. When optimising the max AE in the split selection, it must be noted that the algorithm is principally designed for additive loss functions. In order to compare the joint performance of the child models with that of the parent model, the maximum should probably be used instead. However, it is questionable whether the data would be split according to interactions in this case, especially if the models were nevertheless fitted using least squared regression.

	R^2	MSE	MAE	max AE
SLIM	0.9997757	3.1e-06	0.0010172	0.0118951
Blackbox model	0.9998316	2.3e-06	0.0009441	0.0047668

Table 16: Accuracy of standalone B-spline SLIM MBTs and of the black box model K2204

6.2 Data set R1_08

The data set R108 contains the data of an annuity insurance tariff. Instead of a single benefit in the endowment case, a lifelong annuity is paid out. R108 includes, in addition to sex, age and duration, the features birthyear, payment_period, in_year_payments_exkasso (categorical with 4 levels), guarantee_period and increment_factor. The value range of the features age and duration is limited to a range in which no correlation exists (25 age 35 and 30 duration 40). Here, as well, the BPV and BPV_pred predicted by a black box model are analysed as target variables. For the application of the MBTs, subsets of the training and test data sets including BPV and BPV_pred are drawn, each with 100000 observations.

In order to model nonlinearities in the nodes, MBTs with B-spline models are used. As with K2204, trees with two different depths are evaluated, shallow trees with a maximum depth of 3 and deep trees with a maximum depth of 7. impr and alpha are set to 0.05 and a minimum node size of 500 observations is required.

For the interpretation of the splits and models, the shallow SLIM tree is examined in more detail. Its structure is shown in Figure 19. If the tree is fitted on the test data instead of the training data, exactly the same splits result. The same applies if the tree is used as

a standalone model for BPV instead of a surrogate model. This indicates that the black box model replicates the true data and relationships well.

Figure 19: Shallow SLIM tree for K108 with B-spline models

It is noticeable that the feature space in the first splits is only divided with regard to increment_factor. The effects of these splits on the other features are shown in Figure 20. The effects of the features payment_period and guarantee_period are not shown because they are very small and therefore not visible on the scale at all. The interaction with increment_factor is clearly visible in all the features shown. For all features, an increasing increment_factor increases the feature effects.
The consequences of the splits with respect to duration are shown in Figure 21. It indicates that across all features a higher duration weakens the feature main effects.
Since increment_factor interacts with so many features, the splits are very effective here and bring a great improvement in performance. Table 18 lists the share of the splits by the different features. From this it can be seen that MOB and CTree split shallow MBTs considerably less by increment_factor than SLIM and GUIDE. At the same time, their performance is behind that of SLIM and GUIDE, as can be seen in Table 17. SLIM achieves the best performance.
By using deep trees, the fidelity increases and the results of the different algorithms move closer together. The values for share also get closer. Overall, the differences between the different algorithms are therefore less pronounced with the deep trees than with the shal-

Figure 20: Input-output relation of features in nodes split by increment factor for SLIM tree with B-splines and depth 3

Figure 21: Input-output relation of features in nodes split by duration for SLIM tree with B-splines and depth 3 (nodes 11-14)

low trees. Although the deep trees achieve a high training performance, an interpretation is hardly possible with this high number of leaf nodes. In addition, the test performance

di ers more strongly from the training performance than with the shallow trees, which is a sign of over tting. The almost identical values for max AE have been checked and are correct. They result from the same observation for all four algorithms and the observation falls in nodes with almost identical subregions. This shows again that the MTBs or, more precisely, the subregions found by the di erent algorithms and thus also their performance move closer together when deep MBTs are formed.

	train				test				
	R^2	MSE	MAE	max AE	R^2	MSE	MAE	max AE	n leaves
B-spline	0.927748	3.534102	1.291762	16.412655	0.927267	3.558816	1.295778	16.413200	1
SLIM shallow	0.997432	0.125606	0.235344	4.296943	0.997397	0.127386	0.236761	4.299219	8
GUIDE shallow	0.996540	0.169233	0.262321	5.006522	0.996496	0.171433	0.263770	5.003334	8
MOB shallow	0.994944	0.247289	0.324474	5.236252	0.994942	0.247489	0.324895	5.336406	8
CTree shallow	0.993049	0.340023	0.401982	5.236252	0.993055	0.339838	0.401632	5.336406	8
SLIM deep	0.999882	0.005794	0.043939	1.079069	0.999870	0.006350	0.045885	1.126732	105
GUIDE deep	0.999844	0.007652	0.052102	1.079069	0.999832	0.008219	0.054046	1.126732	95
MOB deep	0.999861	0.006823	0.047428	1.079069	0.999848	0.007460	0.049459	1.126732	108
CTree deep	0.999839	0.007851	0.058806	1.079069	0.999825	0.008539	0.061363	1.126732	106

Table 17: delity of K1_08 B-spline baseline model and B-spline MBTs K2204

	sex	age	duration	birth_year	increment
SLIM shallow	0.00	0.00	0.12	0.00	0.88
GUIDE shallow	0.00	0.12	0.00	0.00	0.88
MOB shallow	0.00	0.12	0.33	0.00	0.54
CTree shallow	0.00	0.33	0.33	0.00	0.33
SLIM deep	0.10	0.08	0.24	0.05	0.54
GUIDE deep	0.06	0.10	0.22	0.05	0.56
MOB deep	0.11	0.15	0.20	0.03	0.52
CTree deep	0.00	0.17	0.29	0.05	0.49

Table 18: Share of observations split by the di erent features K1_08 B-spline MBTs

The feature payment_period, in_year_payments_exkasso and guarantee_period are never chosen as splitting variables and are therefore not listed in the table.
Finally two SLIM trees with B-spline models are tted as standalone models for BPV and their accuracy is compared with the black box model. The result is shown in Table 19. For this data set, too, the deviations are most considerable for the max AE

	train				test			
	R^2	MSE	MAE	max AE	R^2	MSE	MAE	max AE
SLIM shallow	0.9974321	0.1256075	0.2353447	4.3046812	0.9973966	0.1273854	0.2367417	4.3074899
SLIM deep	0.9998814	0.0057987	0.0439357	1.0862638	0.9998702	0.0063532	0.0458758	1.1344576
Blackbox	1.0000000	0.0000023	0.0011006	0.0096827	1.0000000	0.0000023	0.0010968	0.0099144

Table 19: Accuracy of standalone B-spline SLIM MBTs and of the black box model K1_08

The results of the application of the MBT algorithms were discussed with an expert and the plausibility of the resulting interpretations was veri ed.

7 Conclusion

In order to obtain a surrogate model that is able to partition the feature space in such a way that in the subregions main e ect only models can approximate the black box model well, four di erent MBT algorithms were compared in this thesis.
The comparison of the selection bias showed that according to the classical de nition (i.e. independence scenario) and with numerical variables only indeed only SLIM shows a selection bias, while the other methods seem to be unbiased. A correction approach to eliminate the selection bias under numerical variables in SLIM was not successful. When adding categorical variables, however, GUIDE, MOB and CTree also showed a selection bias. If, instead of the independence scenario, scenarios were examined in which interactions were actually present, a bias was also found there for all algorithms. This problem should always be considered when applying the algorithms without de ning the roles of the features (regressor vs partitiong) in advance, as in this thesis. In extreme cases, it could lead to the phenomenon that features are chosen as splitting variables only because of their scale, although other features actually have a stronger share of interactions.
When examining the splitting behaviour of the algorithms for scenarios with two interactions of the same size with di erent feature types, it was notable that SLIM, GUIDE and partly also MOB choose variables for splitting that reveal subgroups much more frequently than CTree. This enables them to generate considerable smaller and yet better performing trees than CTree (and MOB), if subgroup depending main e ects exist. This was also shown in the subsequent simulations, with the aim of comparing interpretability, performance and interpretability of the di erent algorithms.
As a fundamental problem of all algorithms, however, it turned out that smooth interactions can often only be modelled well by a large number of binary splits, which makes a global interpretation of MBTs di cult or even impossible for such data. In such cases, MBTs are probably not the best choice for surrogate models and models like GA2M (Lou et al., 2013) should be preferred.
One advantage that SLIM and GUIDE have over the other two algorithms is that they are more exible in the selection of di erent (for example penalized) models in the nodes. However, there is potential for improvement for SLIM and GUIDE in the area of pruning. Some simulations showed that the size of the trees varies greatly for both algorithms and that sometimes (incorrectly) highly asymmetric trees are created. To improve this, hyperparameter tuning of the prepruning parameters would be an option. For SLIM, however, this could lead to a considerable computational e ort because of the exhaustive search, whereas GUIDE could be better suited for this.

The simulation example with nonlinear e ects as well as the application of MBTs on insurance data sets have shown that it is recommended to use non-linear models in the leafnodes. This improves the performance considerably and also ensures that the split is actually based on interactions. The split e ects can then be interpreted visually, which was carried out in the insurance data use case. If it is necessary that the parameter estimators of the models are directly interpretable, linear models or polynomial (penalised) models are an alternative.
MBT algorithms, especially SLIM and GUIDE, are a promising addition - although not a universal solution - to the possible model classes for surrogate models. Through the combination of decision rules and (nonlinear) main e ect models, a relatively high performance and interpretability can be achieved at the same time. However, interpretability decreases very quickly with a high number of subregions. A compromise must therefore always be found here. In general, it is advisable to use di erent IML methods simultaneously to check the plausibility of the individual results or to compare them with expert knowledge, which was done in the case of the insurance data sets.

References

Breiman, L., Friedman, J., Stone, C. J. and Olshen, R. A. (1984).Classi cation and Regression Trees,CRC Press.

Casalicchio, G. (2020). customtrees: Custom decision trees.
URL: https://github.com/giuseppec/customtrees

Csardi, G. and Nepusz Tamas (2006). The igraph software package for complex network research,InterJournal Complex Systems : 1695.
URL: https://igraph.org

Doshi-Velez, F. and Kim, B. (2017). Towards a rigorous science of interpretable machine learning, arXiv preprint arXiv:1702.08608 .

Dowle, M. and Srinivasan, A. (2021). data.table: Extension of data.frame.
URL: https://CRAN.R-project.org/package=data.table

Fokkema, M. (2020). Fitting prediction rule ensembles with r package pre,Journal of Statistical Software92(12).

Friedman, J., Hastie, T. and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent,Journal of Statistical Software33(1): 1.

Gates, A. J. and Ahn, Y.-Y. (2017). The impact of random models on clustering similarity, arXiv preprint arXiv:1701.06508 .

Hall, P., Gill, N., Kurka, M. and Phan, W. (2017). Machine learning interpretability with h2o driverless ai,H2O. ai .
URL: http://docs.h2o.ai/driverless-ai/latest-stable/docs/booklets/MLIBooklet.pdf

Henckaerts, R., Antonio, K. and Côté, M.-P. (2022). When stakes are high: Balancing accuracy and transparency with model-agnostic interpretable data-driven surrogates, Expert Systems with Applications202: 117230.
URL: https://www.sciencedirect.com/science/article/pii/s0957417422006042

Herbinger, J., Bischl, B. and Casalicchio, G. (2022). Repid: Regional e ect plots with implicit interaction detection, International Conference on Arti cial Intelligence and Statistics, pp. 10209{10233.

Hothorn, T., Hornik, K. and Zeileis, A. (2006). Unbiased recursive partitioning: A conditional inference framework,Journal of Computational and Graphical Statistics 15(3): 651{674.

Hothorn, T., Hornik, K. and Zeileis, A. (2015). ctree: Conditional inference trees, The comprehensive R archive network.

Hothorn, T. and Zeileis, A. (2015). partykit: A modular toolkit for recursive partytioning in r, Journal of Machine Learning Research 16: 3905–3909.
URL: https://www.jmlr.org/papers/volume16/hothorn15a/hothorn15a.pdf

Hu, L., Chen, J., Nair, V. N. and Sudjianto, A. (2018). Locally interpretable models and effects based on supervised partitioning (lime-sup), arXiv preprint arXiv:1806.00663 .

Hu, L., Chen, J., Nair, V. N. and Sudjianto, A. (2020). Surrogate locally-interpretable models with supervised machine learning algorithms, arXiv preprint arXiv:2007.14528 .

Kassambara, A. (2020). ggpubr: 'ggplot2' based publication ready plots.
URL: https://CRAN.R-project.org/package=ggpubr

Lang, M., Binder, M., Richter, J., Schratz, P., Pfisterer, F., Coors, S., Au, Q., Casalicchio, G., Kotthoff, L. and Bischl, B. (2019). mlr3: A modern object-oriented machine learning framework in r, Journal of Open Source Software
URL: https://joss.theoj.org/papers/10.21105/joss.01903

Lang, M., Bischl, B. and Surmann, D. (2017). batchtools: Tools for r to work on batch systems, The Journal of Open Source Software 2(10).
URL: https://doi.org/10.21105/joss.00135

Loh, W.-Y. (2002). Regression trees with unbiased variable selection and interaction detection, Statistica sinica pp. 361–386.

Loh, W.-Y. (2014). Fifty years of classification and regression trees, International Statistical Review 82(3): 329–348.

Lou, Y., Caruana, R., Gehrke, J. and Hooker, G. (2013). Accurate intelligible models with pairwise interactions, in R. Ghani, T. E. Senator, P. Bradley, R. Parekh, J. He, R. L. Grossman, R. Uthurusamy, I. S. Dhillon and Y. Koren (eds), Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, New York, NY, USA, pp. 623–631.

Maratea, A. and Ferone, A. (2021). Pitfalls of local explainability in complex black-box models, WILF .

McDonald, G. C. (2009). Ridge regression, Wiley Interdisciplinary Reviews: Computational Statistics 1(1): 93–100.

Molnar, C. (2019). Interpretable machine learning: A guide for making Black Box Models interpretable, Lulu, Morisville, North Carolina.

Morgan, J. and Sonquist, J. (1963). Problems in the analysis of survey data, and a proposal, Journal of the American Statistical Association 58(302): 415–434.

msg insurit (n.d.). Automating policy migration with machine learning.
URL: https://msg-insurit.com/your-challenges/machine-learning-insurance/

R Core Team (2022). R: A language and environment for statistical computing.
URL: https://www.R-project.org/

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods, Journal of the American Statistical Association 66(336): 846–850.

Ribeiro, M. T., Singh, S. and Guestrin, C. (2016). Why should i trust you?: Explaining the predictions of any classifier, Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, Association for Computing Machinery, New York, NY, USA, pp. 1135–1144.

Sabourin, J. A., Valdar, W. and Nobel, A. B. (2015). A permutation approach for selecting the penalty parameter in penalized model selection, Biometrics 71(4): 1185–1194.

Schalk, D., Thomas, J. and Bischl, B. (2018). compboost: Modular framework for component-wise boosting, JOSS 3(30): 967.

Schloerke, B., Cook Di, Larmarange, J., Briatte, F., Marbach, M., Thoen, E., Elberg, A. and Crowley, J. (2021). Ggally: Extension to 'ggplot2'.
URL: https://CRAN.R-project.org/package=GGally

Schlosser, L., Hothorn, T. and Zeileis, A. (2019). The power of unbiased recursive partitioning: A unifying view of ctree, mob, and guide, arXiv preprint arXiv:1906.10179 .

Seibold, H., Zeileis, A. and Hothorn, T. (2016). Model-based recursive partitioning for subgroup analyses, The International Journal of Biostatistics 12(1): 45–63.

Thomas, M., Bornkamp, B. and Seibold, H. (2018). Subgroup identification in dose-finding trials via model-based recursive partitioning, Statistics in medicine 37(10): 1608–1624.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society: Series B (Methodological) 58(1): 267–288.

Wang, L., Li, Q., Yu, Y. and Liu, J. (2018). Region compatibility based stability assessment for decision trees, Expert Systems with Applications 105: 112–128.

Wickham, H. (2016). ggplot2: Elegant Graphics for Data Analysis, Springer-Verlag New York.
URL: https://ggplot2.tidyverse.org

Wickham, H. (2022). stringr: Simple, consistent wrappers for common string operations.
URL: https://CRAN.R-project.org/package=stringr

Wickham, H., Francois, R., Henry, L. and Muller, K. (2022). dplyr: A grammar of data manipulation.
URL: https://CRAN.R-project.org/package=dplyr

Wood, S. N. (2011). Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models, Journal of the Royal Statistical Society Series B: Statistical Methodology 73(1): 3–36.

Zeileis, A. and Hornik, K. (2007). Generalized m- uctuation tests for parameter instability, Statistica Neerlandica 61(4): 488–508.

Zeileis, A., Hothorn, T. and Hornik, K. (2008). Model-based recursive partitioning, Journal of Computational and Graphical Statistics 17(2): 492–514.

List of Figures

List of Tables

A Appendix

A.1 GUIDE replication

To validate the R implementation of GUIDE, the selection bias simulation example from (Loh, 2002) was replicated for independent features. For this, GUIDE was simulated both without bootstrap bias correction and with correction in 1000 repetitions. All settings were chosen to the best of my knowledge as described in the paper. In Table 20 the results from the paper are listed in the first two columns and in the following two columns the results from the replication. In the replication the features $x_1; x_2; x_2$ are also selected in the uncorrected version to a certain extent as splitting variables. However, the bias is notable. The effect of the bias correction can also be seen, although not as considerable as in the results from the paper.

	original biased	original corrected	R biased	R corrected
x_1	0	178	106	143
x_2	0	232	129	180
x_3	0	200	118	179
x_4	469	181	293	224
x_5	531	209	354	274

Table 20: Comparison of the results on selection bias and bias correction from the paper (Loh, 2002) with my R implementation

A.2 Selection bias correction approach SLIM

By means of a simulation, I investigated how the number of quantiles (n.quantiles) used as potential splitpoints in SLIM affects the selection bias for numerical features. The data is defined as follows:

- $x_1; x_2 \quad U(0; 1)$
- x_3 uniformly distributed on equidistant grids of length 10, 25, 50 and 100 on the interval [0,1] (i.e. four different settings)
- $y \quad N(0; 1)$

For each of the four simulation scenarios, SLIM trees are fitted with n.quantiles $2 \{100; 75; 50; 25; 10; 2\}$ and one exact tree (each unique value as potential splitpoint) in each simulation run. The experiment is repeated 3000 times.
In Table 21 the frequency of the selected features and the resulting train and test mean MSE for the four different grid sizes and seven different values of n.quantiles is listed.

x_3 gridsize \ n.quantiles		exact	100	75	50	25	10	2
10	x_1	1439	1388	1353	1309	1249	1120	1039
10	x_2	1316	1222	1224	1217	1153	1033	969
10	x_3	245	390	423	474	598	847	992
10	MSE_{train}	0.9841	0.9850	0.9851	0.9855	0.9863	0.9877	0.9906
10	MSE_{test}	1.0193	1.0180	1.01789	1.0174	1.0165	1.0154	1.0110
25	x_1	1294	1206	1169	1133	1042	995	992
25	x_2	1244	1154	1138	1088	1033	1006	1012
25	x_3	462	640	693	779	925	999	996
25	MSE_{train}	0.9827	0.9833	0.9835	0.9838	0.9844	0.9857	0.9897
25	MSE_{test}	1.0208	1.0198	1.0197	1.0194	1.0187	1.0172	1.0129
50	x_1	1246	1125	1085	1057	1043	1018	1043
50	x_2	1120	1035	1018	984	955	948	950
50	x_3	634	840	897	959	1002	1034	1007
50	MSE_{train}	0.9833	0.9840	0.9841	0.9844	0.9849	0.9863	0.9902
50	MSE_{test}	1.0189	1.0179	1.01770	1.0173	1.0164	1.0150	1.0108
100	x_1	1105	989	998	981	967	942	977
100	x_2	1138	1048	1010	1031	1048	1030	1026
100	x_3	757	963	992	988	985	1028	997
100	MSE_{train}	0.9826	0.9832	0.9834	0.9837	0.9844	0.9859	0.9898
100	MSE_{test}	1.0166	1.0157	1.0155	1.0151	1.0146	1.0129	1.0087

Table 21: Frequency of feature selection and mean MSE for four di erent grid sizes and seven di erent values of n.quantiles

There one can see that the selection bias in all the variations shown decreases with decreasing values of n.quantiles. However, if n.quantiles is chosen equal to the number of unique values ofx_3 the selection bias does not seem to be completely eliminated. Only when n.quantiles is chosen smaller is selection bias no longer visible.
For the MSE, it can be observed that as the number of n.quantiles decreases, theMSE_{train} increases and theMSE_{test} decreases. ThatMSE_{train} increases with decreasing n.quantiles was to be expected and is due to the fact that if fewer quantiles are used as potential splitpoint set, the optimal splitpoint may not fall into this set. The values ofMSE_{test} are not very meaningful here, as there is no dependency between the features and the target. This means that the forced split is merely an adjustment to the random error terms, which naturally look di erent on the test data. A better generalisation is therefore obtained in this case if the t to the training errors is not exact.
The scenarios independence numeric and independence mixed from chapter 4 were also simulated with varying values of n.quantiles. The results are listed in Table 22. In the case of scenario independence numeric, the approach seems to be successful. However, here again, only with n.quantiles = 2 selection bias is no longer visible. In the independence mixed scenario, the frequencies also change, but with no value of n.quantiles an approxi-

mately unbiased result is found. If n.quantiles is chosen too small, the bias turns in the other direction (i.e. categorical variables are preferably used for the first split). Within the numerical variables ($x_1; ...x_3$) the balance seems to succeed. The binary variable x_4 is not chosen sufficiently often in any variant. For the MSE, the results correspond to those of the previous simulation.

n.quantiles / x_3 scenario		exact	100	75	50	25	10	2
numeric	x1	1062	957	946	913	875	808	757
numeric	x2	1033	919	899	879	828	759	757
numeric	x3	205	284	293	349	446	646	750
numeric	x4	700	840	862	859	851	787	736
numeric	MSE_{train}	0.9800	0.9807	0.9808	0.9812	0.9820	0.9836	0.9868
numeric	MSE_{test}	1.0194	1.0187	1.0182	1.0179	1.0170	1.0160	1.0112
mixed	x1	1259	1186	1185	1153	1095	954	458
mixed	x2	1248	1184	1150	1126	1034	891	485
mixed	x3	362	465	488	534	638	769	463
mixed	x4	19	25	28	29	35	55	203
mixed	x5	61	78	83	89	113	188	778
mixed	x6	50	61	65	68	84	142	612
mixed	MSE_{train}	0.9528	0.9535	0.9538	0.9541	0.9550	0.9567	0.9606
mixed	MSE_{test}	1.0515	1.0498	1.0493	1.0486	1.0475	1.0450	1.0399

Table 22: Frequency of feature selection and mean MSE for scenarios selection bias independence numeric and independence mixed and seven different values of n.quantiles

In order to investigate whether the correction approach, which seems to work at least for numerical variables, also leads to unbiased variable selection in the presence of interactions, the numerical vs numerical scenario from chapter 4.3 is simulated for different values of n.quantiles. The results are listed in Table 23.
x_3 only takes on 11 different values in this scenario. In the independence case, it would therefore be expected that the selection bias decreases with a decreasing number of quantiles and is no longer recognisable at least for n.quantiles = 2. Here, however, a different picture emerges. If all possible splitpoints are included, the variable x_1 with more possible splitpoints is preferred, as expected. With decreasing values of n.quantiles the relationship changes, but no direct linear relationship can be seen. Rather, the selection bias reverses at n.quantiles = 100 and then increases sharply and decreases again somewhat at lower values. However, the bias seems to be smallest when all possible split points are used for the exhaustive search.
From this we can conclude that unbiasedness in the independence case does not automatically lead to unbiasedness in the presence of interactions. However, it is precisely the selection bias in the case of present interactions that is important. Since the effect of the

correction approach on the selection bias in the presence of interactions is hardly generally predictable, this approach is therefore not recommended. Instead, n.quantiles should be chosen with regard to the criteria train and test MSE and, if necessary, computational time.

n.quantiles	exact	100	75	50	25	10	2
x_1	1643.0000	1272.0000	1135.0000	924.0000	447.0000	592.0000	1809.0000
x_3	1357.0000	1728.0000	1865.0000	2076.0000	2553.0000	2408.0000	1191.0000
MSE_{train}	0.0373	0.0375	0.0376	0.0377	0.0380	0.0379	0.0393
MSE_{test}	0.0391	0.0393	0.0393	0.0393	0.0392	0.0392	0.0404

Table 23: Frequency of feature selection and mean MSE for scenario selection bias numerical vs numerical and seven di erent values of n.quantiles

A.3 Simulation results

A.3.1 Basic scenarios

XGBoost con gurations In Table 24 the XGBoost hyperparameter settings, which were used for the simulation of the basic scenarios are listed.

	linear smooth	linear categorical	linear mixed
max_depth	5	3	5
eta	0.5	0.5	0.5
alpha	1	0.5	2
gamma	2	1	3.5
nrounds	400	350	500

Table 24: XGBoost hyperparameters for basic scenarios after hyperparametertuning

Simulation Results The Figures 23 and 22 each show an example of a SLIM tree with high and low symmetry on the scenario linear smooth with each 8 leafnodes. The value impr in the plot is the potential relative improvement of the objective through the best split for this node in relation to the improvement in the parent node. In the tree 22, the required improvement of 10% in node 2 is just missed, which is why this node is not split further.

Figure 22: SLIM tree with high symmetry on scenario linear smooth

Figure 23: SLIM tree with low symmetry on scenario linear smooth

In the following tables and figures the mean simulation results of the basic scenarios, which are not included in chapter 5 are shown.

black box	MBT	impr	alpha	n leaves	n leaves min	n leaves max	R^2_{train}	sd R^2_{train}	R^2_{test}	sd R^2_{test}
lm	SLIM	0.15		2.06	2	3	0.9650	0.0043	0.9631	0.0046
lm	SLIM	0.10		12.11	5	16	0.9965	0.0052	0.9958	0.0060
lm	SLIM	0.05		15.70	14	16	0.9995	0.0001	0.9993	0.0001
lm	GUIDE	0.15		2.07	2	3	0.9651	0.0044	0.9632	0.0049
lm	GUIDE	0.10		12.03	5	16	0.9965	0.0051	0.9957	0.0060
lm	GUIDE	0.05		15.75	14	16	0.9995	0.0001	0.9993	0.0001
lm	MOB		0.001	15.78	14	16	0.9994	0.0001	0.9993	0.0001
lm	MOB		0.010	15.78	14	16	0.9994	0.0001	0.9993	0.0001
lm	MOB		0.050	15.78	14	16	0.9994	0.0001	0.9993	0.0001
lm	CTree		0.001	15.22	13	17	0.9993	0.0001	0.9992	0.0001
lm	CTree		0.010	15.22	13	17	0.9993	0.0001	0.9992	0.0001
lm	CTree		0.050	15.22	13	17	0.9993	0.0001	0.9992	0.0001
lm							0.9902	0.0006	0.9901	0.0008
XGBoost	SLIM	0.15		2.31	2	6	0.9665	0.0069	0.9629	0.0079
XGBoost	SLIM	0.10		7.33	2	14	0.9850	0.0060	0.9814	0.0062
XGBoost	SLIM	0.05		14.30	8	17	0.9948	0.0010	0.9909	0.0017
XGBoost	GUIDE	0.15		2.26	2	5	0.9664	0.0067	0.9628	0.0077
XGBoost	GUIDE	0.10		6.92	2	14	0.9847	0.0061	0.9811	0.0062
XGBoost	GUIDE	0.05		14.15	8	17	0.9945	0.0010	0.9906	0.0017
XGBoost	MOB		0.001	10.89	8	13	0.9944	0.0005	0.9904	0.0011
XGBoost	MOB		0.010	11.96	9	15	0.9946	0.0005	0.9906	0.0011
XGBoost	MOB		0.050	12.86	11	15	0.9948	0.0005	0.9908	0.0011
XGBoost	CTree		0.001	12.09	9	15	0.9940	0.0006	0.9900	0.0012
XGBoost	CTree		0.010	13.21	10	15	0.9943	0.0006	0.9902	0.0013
XGBoost	CTree		0.050	14.09	11	17	0.9944	0.0006	0.9904	0.0012
XGBoost							0.9858	0.0008	0.9768	0.0018

Table 25: Mean simulation results on 100 simulation runs as surrogate models on scenario linear smooth with sample size $n = 1500$ for different values of impr and alpha

Figure 24: Maximum leaf size of standalone MBTs vs number of leaf nodes scenario linear smooth with $n = 1500$; $alpha = 0.001$; $impr = 0.1$

Figure 25: Test delity R^2 as surrogate on lm predictions vs number of leaf nodes scenario linear smooth with $n = 1500; alpha = 0:001; impr = 0:1$

Figure 26: Test delity R^2 as surrogate on XGBoost predictions vs number of leaf nodes scenario linear smooth with $n = 1500; alpha = 0:001; impr = 0:1$

black box	MBT	impr	alpha	n leaves	n leaves min	n leaves max	R^2_{train}	sd R^2_{train}	R^2_{test}	sd R^2_{test}
standalone	SLIM	0.15		2.04	2	3	0.9542	0.0034	0.9536	0.0036
standalone	SLIM	0.10		36.94	15	58	0.9895	0.0024	0.9880	0.0024
standalone	SLIM	0.05		55.84	50	61	0.9910	0.0002	0.9890	0.0004
standalone	GUIDE	0.15		2.03	2	3	0.9540	0.0030	0.9534	0.0031
standalone	GUIDE	0.10		36.23	14	52	0.9894	0.0023	0.9881	0.0024
standalone	GUIDE	0.05		56.35	47	62	0.9909	0.0002	0.9891	0.0003
standalone	MOB		0.001	17.15	16	21	0.9901	0.0002	0.9893	0.0004
standalone	MOB		0.010	19.14	16	22	0.9902	0.0002	0.9894	0.0004
standalone	MOB		0.050	21.71	18	26	0.9903	0.0002	0.9895	0.0004
standalone	CTree		0.001	19.19	17	22	0.9901	0.0002	0.9894	0.0004
standalone	CTree	0	0.010	21.58	17	25	0.9902	0.0002	0.9895	0.0004
standalone	CTree		0.050	24.56	21	30	0.9903	0.0002	0.9895	0.0003
lm	SLIM	0.15		2.00	2	2	0.9628	0.0009	0.9625	0.0012
lm	SLIM	0.10		52.19	43	60	0.9999	0.0001	0.9998	0.0001
lm	SLIM	0.05		63.99	63	64	1.0000	0.0000	1.0000	0.0000
lm	GUIDE	0.15		2.00	2	2	0.9628	0.0009	0.9625	0.0012
lm	GUIDE	0.10		52.37	43	60	0.9999	0.0001	0.9998	0.0001
lm	GUIDE	0.05		63.99	63	64	1.0000	0.0000	1.0000	0.0000
lm	MOB		0.001	64.00	64	64	1.0000	0.0000	1.0000	0.0000
lm	MOB		0.010	64.00	64	64	1.0000	0.0000	1.0000	0.0000
lm	MOB		0.050	64.00	64	64	1.0000	0.0000	1.0000	0.0000
lm	CTree		0.001	63.60	62	64	1.0000	0.0000	1.0000	0.0000
lm	CTree		0.010	63.60	62	64	1.0000	0.0000	1.0000	0.0000
lm	CTree	5	0.050	63.60	62	64	1.0000	0.0000	1.0000	0.0000
lm							0.9901	0.0002	0.9901	0.0003
XGBoost	SLIM	0.15		2.22	2	7	0.9647	0.0063	0.9643	0.0063
XGBoost	SLIM	0.10		19.23	3	40	0.9908	0.0062	0.9902	0.0062
XGBoost	SLIM	0.05		49.05	34	56	0.9978	0.0003	0.9972	0.0003
XGBoost	CTree	0.15		32.81	26	41	0.9972	0.0002	0.9965	0.0003
XGBoost	CTree	0.10		36.36	29	43	0.9973	0.0002	0.9966	0.0003
XGBoost	CTree	0.05		39.37	32	47	0.9974	0.0002	0.9967	0.0003
XGBoost	MOB		0.001	38.44	29	47	0.9979	0.0002	0.9972	0.0003
XGBoost	MOB		0.010	42.62	31	53	0.9980	0.0002	0.9973	0.0003
XGBoost	MOB		0.050	46.02	39	55	0.9981	0.0002	0.9974	0.0003
XGBoost	GUIDE		0.001	2.21	2	8	0.9645	0.0061	0.9641	0.0062
XGBoost	GUIDE		0.010	19.27	3	38	0.9907	0.0062	0.9900	0.0062
XGBoost	GUIDE		0.050	49.90	34	58	0.9977	0.0003	0.9970	0.0004
XGBoost							0.9877	0.0003	0.9852	0.0006

Table 26: Mean simulation results on 100 simulation runs as stand alone and surrogate models on scenario linear smooth with sample size $n = 7500$ for different values of impr and alpha

black box	MBT	impr	alpha	n leaves	n leaves min	n leaves max	R^2_{train}	sd R^2_{train}	R^2_{test}	sd R^2_{test}
gam	SLIM	0.15		2.00	2	2	0.8528	0.0064	0.8513	0.0108
gam	SLIM	0.10		2.64	2	4	0.8972	0.0432	0.8937	0.0440
gam	SLIM	0.05		8.56	4	16	0.9910	0.0029	0.9893	0.0039
gam	GUIDE	0.15		2.00	2	2	0.8528	0.0064	0.8513	0.0108
gam	GUIDE	0.10		2.64	2	4	0.8972	0.0432	0.8937	0.0440
gam	GUIDE	0.05		6.06	4	13	0.9875	0.0031	0.9859	0.0038
gam	MOB		0.001	13.53	11	15	0.9773	0.0020	0.9718	0.0028
gam	MOB		0.010	14.28	13	16	0.9784	0.0020	0.9728	0.0029
gam	MOB		0.050	14.92	13	16	0.9797	0.0021	0.9740	0.0028
gam	CTree		0.001	13.89	11	16	0.9773	0.0018	0.9720	0.0028
gam	CTree		0.010	14.47	12	16	0.9779	0.0017	0.9725	0.0027
gam	CTree		0.050	14.86	13	16	0.9783	0.0016	0.9729	0.0028
gam							0.9702	0.0018	0.9694	0.0029
XGBoost	SLIM	0.15		2.00	2	2	0.8321	0.0075	0.8323	0.0118
XGBoost	SLIM	0.10		4.00	4	4	0.9923	0.0012	0.9870	0.0029
XGBoost	SLIM	0.05		4.00	4	4	0.9923	0.0012	0.9870	0.0029
XGBoost	GUIDE	0.15		2.00	2	2	0.8321	0.0075	0.8323	0.0118
XGBoost	GUIDE	0.10		4.00	4	4	0.9923	0.0012	0.9870	0.0029
XGBoost	GUIDE	0.05		4.00	4	4	0.9923	0.0012	0.9870	0.0029
XGBoost	MOB		0.001	13.45	11	16	0.9793	0.0063	0.9729	0.0069
XGBoost	MOB		0.010	14.38	13	16	0.9831	0.0055	0.9765	0.0066
XGBoost	MOB		0.050	14.63	13	16	0.9837	0.0052	0.9771	0.0062
XGBoost	CTree		0.001	11.96	10	14	0.9602	0.0030	0.9545	0.0049
XGBoost	CTree		0.010	12.76	10	15	0.9612	0.0033	0.9550	0.0050
XGBoost	CTree		0.050	13.46	10	16	0.9623	0.0036	0.9558	0.0052
XGBoost							0.9876	0.0015	0.9778	0.0031

Table 27: Mean simulation results on 100 simulation runs as surrogate models on scenario linear categorical with sample size $n = 1500$ for different values of impr and alpha

black box	MBT	impr	alpha	n leaves	n leaves min	n leaves max	R^2_{train}	sd R^2_{train}	R^2_{test}	sd R^2_{test}
standalone	SLIM	0.15		2.00	2	2	0.8277	0.0032	0.8267	0.0048
standalone	SLIM	0.10		4.00	4	4	0.9887	0.0008	0.9886	0.0011
standalone	SLIM	0.05		4.00	4	4	0.9887	0.0008	0.9886	0.0011
standalone	GUIDE	0.15		2.00	2	2	0.8277	0.0032	0.8267	0.0048
standalone	GUIDE	0.10		4.00	4	4	0.9887	0.0008	0.9886	0.0011
standalone	GUIDE	0.05		4.00	4	4	0.9887	0.0008	0.9886	0.0011
standalone	MOB		0.001	41.23	26	48	0.9896	0.0004	0.9868	0.0011
standalone	MOB		0.010	45.50	27	53	0.9899	0.0003	0.9871	0.0011
standalone	MOB		0.050	49.24	31	58	0.9902	0.0003	0.9874	0.0011
standalone	CTree		0.001	22.51	20	25	0.9499	0.0014	0.9462	0.0022
standalone	CTree		0.010	24.39	21	27	0.9502	0.0014	0.9464	0.0023
standalone	CTree		0.050	26.62	22	30	0.9507	0.0016	0.9468	0.0023
gam	SLIM	0.15		2.00	2	2	0.8532	0.0029	0.8527	0.0043
gam	SLIM	0.10		2.22	2	4	0.8681	0.0299	0.8668	0.0302
gam	SLIM	0.05		27.66	13	42	0.9940	0.0036	0.9935	0.0039
gam	GUIDE	0.15		2.00	2	2	0.8532	0.0029	0.8527	0.0043
gam	GUIDE	0.10		2.22	2	4	0.8681	0.0299	0.8668	0.0302
gam	GUIDE	0.05		14.56	4	31	0.9886	0.0038	0.9882	0.0040
gam	MOB		0.001	61.11	55	64	0.9973	0.0002	0.9966	0.0003
gam	MOB		0.010	62.42	59	64	0.9973	0.0002	0.9966	0.0002
gam	MOB		0.050	63.15	61	64	0.9974	0.0002	0.9967	0.0002
gam	CTree		0.001	33.93	31	38	0.9789	0.0007	0.9769	0.0011
gam	CTree		0.010	36.76	34	43	0.9793	0.0008	0.9772	0.0012
gam	CTree		0.050	39.43	36	45	0.9798	0.0010	0.9776	0.0013
gam							0.9701	0.0010	0.9698	0.0014
XGBoost	SLIM	0.15		2.00	2	2	0.8335	0.0033	0.8334	0.0048
XGBoost	SLIM	0.10		4.00	4	4	0.9949	0.0009	0.9942	0.0011
XGBoost	SLIM	0.05		4.00	4	4	0.9949	0.0009	0.9942	0.0011
XGBoost	GUIDE	0.15		2.00	2	2	0.8335	0.0033	0.8334	0.0048
XGBoost	GUIDE	0.10		4.00	4	4	0.9949	0.0009	0.9942	0.0011
XGBoost	GUIDE	0.05		4.00	4	4	0.9949	0.0009	0.9942	0.0011
XGBoost	MOB		0.001	53.91	49	60	0.9987	0.0002	0.9974	0.0007
XGBoost	MOB		0.010	55.38	49	60	0.9988	0.0002	0.9974	0.0007
XGBoost	MOB		0.050	56.40	50	61	0.9988	0.0002	0.9974	0.0007
XGBoost	CTree		0.001	24.16	19	29	0.9600	0.0016	0.9577	0.0021
XGBoost	CTree		0.010	26.39	21	32	0.9605	0.0015	0.9581	0.0021
XGBoost	CTree		0.050	28.97	22	36	0.9612	0.0017	0.9587	0.0023
XGBoost							0.9880	0.0009	0.9852	0.0009

Table 28: Mean simulation results on 100 simulation runs as stand alone and surrogate models on scenario linear categorical with sample size $n = 7500$ for different values of impr and alpha

black box	MBT	impr	alpha	n leaves	n leaves min	n leaves max	R^2_{train}	sd R^2_{train}	R^2_{test}	sd R^2_{test}
lm	SLIM	0.15		3.20	2	13	0.8879	0.0309	0.8806	0.0331
lm	SLIM	0.10		13.07	5	16	0.9875	0.0098	0.9843	0.0108
lm	SLIM	0.05		14.78	12	16	0.9913	0.0020	0.9885	0.0028
lm	GUIDE	0.15		3.17	2	13	0.8872	0.0308	0.8799	0.0329
lm	GUIDE	0.10		12.66	7	16	0.9866	0.0095	0.9834	0.0106
lm	GUIDE	0.05		14.38	12	16	0.9905	0.0022	0.9876	0.0029
lm	MOB		0.001	14.99	13	17	0.9882	0.0016	0.9838	0.0021
lm	MOB		0.010	14.99	13	17	0.9882	0.0016	0.9838	0.0021
lm	MOB		0.050	14.99	13	17	0.9882	0.0016	0.9838	0.0021
lm	CTree		0.001	15.05	13	17	0.9880	0.0016	0.9841	0.0019
lm	CTree		0.010	15.05	13	17	0.9880	0.0016	0.9841	0.0019
lm	CTree		0.050	15.05	13	17	0.9880	0.0016	0.9841	0.0019
lm							0.9902	0.0006	0.9898	0.0008
XGBoost	SLIM	0.15		4.47	2	13	0.9067	0.0336	0.9013	0.0339
XGBoost	SLIM	0.10		12.80	7	16	0.9832	0.0089	0.9724	0.0103
XGBoost	SLIM	0.05		14.80	12	17	0.9870	0.0018	0.9764	0.0044
XGBoost	GUIDE	0.15		4.37	2	13	0.9059	0.0335	0.9005	0.0339
XGBoost	GUIDE	0.10		12.48	6	16	0.9822	0.0098	0.9715	0.0112
XGBoost	GUIDE	0.05		14.47	12	17	0.9863	0.0022	0.9758	0.0047
XGBoost	MOB		0.001	14.82	13	17	0.9853	0.0018	0.9745	0.0047
XGBoost	MOB		0.010	14.94	13	17	0.9854	0.0017	0.9746	0.0046
XGBoost	MOB		0.050	14.94	13	17	0.9854	0.0017	0.9746	0.0046
XGBoost	CTree		0.001	15.03	13	17	0.9850	0.0017	0.9743	0.0042
XGBoost	CTree		0.010	15.07	13	17	0.9850	0.0017	0.9743	0.0042
XGBoost	CTree		0.050	15.07	13	17	0.9850	0.0017	0.9743	0.0042
XGBoost							0.9859	0.0014	0.9682	0.0042

Table 29: Mean simulation results on 100 simulation runs as surrogate models on scenario linear mixed with sample size $n = 1500$ for different values of impr and alpha

black box	MBT	impr	alpha	n leaves	n leaves min	n leaves max	R^2_{train}	sd R^2_{train}	R^2_{test}	sd R^2_{test}
standalone	SLIM	0.15		2.30	2	8	0.8637	0.0165	0.8626	0.0176
standalone	SLIM	0.10		38.96	21	54	0.9865	0.0029	0.9849	0.0030
standalone	SLIM	0.05		59.03	51	64	0.9904	0.0005	0.9884	0.0006
standalone	GUIDE	0.15		2.30	2	8	0.8637	0.0165	0.8626	0.0176
standalone	GUIDE	0.10		38.06	21	54	0.9866	0.0028	0.9851	0.0029
standalone	GUIDE	0.05		58.20	45	63	0.9903	0.0005	0.9885	0.0006
standalone	MOB		0.001	48.54	43	55	0.9888	0.0004	0.9861	0.0007
standalone	MOB		0.010	53.17	48	58	0.9891	0.0003	0.9864	0.0007
standalone	MOB		0.050	56.07	51	60	0.9893	0.0003	0.9866	0.0006
standalone	CTree		0.001	51.70	47	57	0.9887	0.0004	0.9858	0.0006
standalone	CTree		0.010	54.85	50	58	0.9889	0.0004	0.9860	0.0006
standalone	CTree		0.050	56.76	52	61	0.9890	0.0003	0.9860	0.0006
lm	SLIM	0.15		2.17	2	16	0.8683	0.0094	0.8677	0.0103
lm	SLIM	0.10		39.69	19	55	0.9956	0.0028	0.9953	0.0030
lm	SLIM	0.05		57.51	46	64	0.9991	0.0005	0.9990	0.0006
lm	GUIDE	0.15		2.17	2	16	0.8683	0.0094	0.8677	0.0103
lm	GUIDE	0.10		40.32	19	55	0.9960	0.0026	0.9956	0.0028
lm	GUIDE	0.05		57.32	46	64	0.9991	0.0005	0.9990	0.0006
lm	MOB		0.001	63.11	60	64	0.9973	0.0002	0.9964	0.0003
lm	MOB		0.010	63.11	60	64	0.9973	0.0002	0.9964	0.0003
lm	MOB		0.050	63.11	60	64	0.9973	0.0002	0.9964	0.0003
lm	CTree		0.001	62.72	59	64	0.9973	0.0002	0.9964	0.0003
lm	CTree		0.010	62.72	59	64	0.9973	0.0002	0.9964	0.0003
lm	CTree		0.050	62.72	59	64	0.9973	0.0002	0.9964	0.0003
lm							0.9901	0.0003	0.9902	0.0004
XGBoost	SLIM	0.15		2.22	2	12	0.8693	0.0121	0.8706	0.0129
XGBoost	SLIM	0.10		39.52	23	53	0.9931	0.0027	0.9914	0.0027
XGBoost	SLIM	0.05		56.86	46	63	0.9964	0.0005	0.9948	0.0007
XGBoost	GUIDE	0.15		2.15	2	9	0.8693	0.0119	0.8705	0.0127
XGBoost	GUIDE	0.10		38.13	22	52	0.9928	0.0027	0.9912	0.0028
XGBoost	GUIDE	0.05		56.72	46	63	0.9963	0.0006	0.9946	0.0007
XGBoost	MOB		0.001	58.76	54	63	0.9961	0.0003	0.9942	0.0005
XGBoost	MOB		0.010	59.47	55	64	0.9961	0.0003	0.9943	0.0005
XGBoost	MOB		0.050	59.69	55	64	0.9961	0.0003	0.9943	0.0005
XGBoost	CTree		0.001	58.51	53	62	0.9956	0.0003	0.9936	0.0005
XGBoost	CTree		0.010	58.95	54	63	0.9956	0.0003	0.9936	0.0005
XGBoost	CTree		0.050	59.09	54	63	0.9956	0.0003	0.9936	0.0005
XGBoost							0.9877	0.0007	0.9836	0.0011

Table 30: Mean simulation results on 100 simulation runs as stand alone and surrogate models on scenario linear mixed with sample size = 7500 for different values of impr and alpha

A.3.2 Linear smooth with noise features

Figures 27 and 28 show the the R^2 test accuracy in scenario linear smooth with noise features. Figure 28 shows that in this scenario asymmetric trees (i.e. trees with a maximum leafsize> 700) lead to poorer performance in SLIM and GUIDE.

Figure 27: Test accuracy R^2 of MBTs on scenario Linear smooth with noise features with $n = 3000$; alpha $= 0.001$; impr $= 0.1$ with nleaves 11

Figure 28: Test accuracy R^2 of SLIM and GUIDE MBTs on scenario Linear smooth with noise features with $n = 3000$; alpha $= 0.001$; impr $= 0.1$ with nleaves 11

A.3.3 Nonlinear e ects

XGBoost con gurations In Table 31 the XGBoost hyperparameter setting for simulation nonlinear is shown.

	nonlinear
max_depth	4
eta	0.825
alpha	0.75
gamma	1
nrounds	700

Table 31: XGBoost hyperparameters for scenario nonlinear after hyperparametertuning

black box	model	n leaves	n l min	n l max	n split feat	n sf min	n sf max	% main e ect	df	sd df
standalone	basic lm	43.06	36	48	5.68	4	6	0.5089	6.5804	0.1249
standalone	penalized poly	19.32	16	22	4.26	3	5	0.2374	8.8768	0.3681
standalone	B-Splines	18.38	13	22	4.26	3	6	0.0126		
standalone	gam	17.60	12	22	3.44	2	4	0.0007		
XGBoost	basic lm	30.38	2	47	4.56	1	6	0.6732	6.8519	0.1558
XGBoost	penalized poly	20.66	16	24	4.70	3	6	0.2327	9.2909	0.4347
XGBoost	B-Splines	17.38	2	25	4.70	1	6	0.0597		
XGBoost	gam	15.28	2	24	4.00	1	6	0.0191		

Table 32: Mean interpretability simulation results nonlinear variant 2

black box	model	R^2_{train}	sd R^2_{train}	R^2_{test}	sd R^2_{test}	time in sec
standalone	basic lm	0.9633	0.0035	0.9469	0.0048	102.2842
standalone	penalized poly	0.9824	0.0015	0.9781	0.0020	195.4385
standalone	B-Splines	0.9929	0.0013	0.9714	0.0029	113.2447
standalone	gam	0.9880	0.0016	0.9835	0.0017	1544.3393
XGBoost	basic lm	0.9198	0.0516	0.9030	0.0417	72.3790
XGBoost	penalized poly	0.9750	0.0033	0.9634	0.0051	206.5870
XGBoost	B-Splines	0.9865	0.0141	0.9591	0.0092	97.9321
XGBoost	gam	0.9786	0.0127	0.9672	0.0115	1728.6021
XGBoost	XGBoost	0.9392	0.0342	0.9216	0.0405	3.0131

Table 33: Mean performance simulation results nonlinear e ects variant 2

A.4 Insurance use case

A.4.1 K2204 BPV

Figure 29: SLIM tree with linear models as standalone model for BPV in data set K2204

Figure 30: Frequency of observations with respect to feature age

A.4.2 K2204 PPV

All plots and tables shown in the chapter 6.1 for the data set k2204 with target BPV are shown here for the target PPV (or PPV_pred). The interpretation can be done analogously to chapter 6.1. The effects found here are merely all larger (since PPV is also larger than BPV) and their direction is exactly reversed.
Figure 31 plots the prediction of the baseline B-spline model and the two B-spline SLIM surrogates against PPV_pred to visualise performance improvement.

	R^2	MSE	MAE	max AE	n leaves
linear baseline model	0.985092	2.523500	1.277544	7.271319	1
SLIM	0.999232	0.130069	0.255018	2.300768	8
GUIDE	0.999275	0.122765	0.240393	2.300768	8
MOB	0.998524	0.249776	0.353372	2.758192	8
CTree	0.995088	0.831537	0.643819	4.808448	8

Table 34: Fidelity of K2204 PPV linear baseline model and linear MBTs

	age	duration	sex
SLIM	0.28	0.67	0.05
GUIDE	0.28	0.72	0.00
MOB	0.10	0.77	0.13
CTree	0.00	0.96	0.04

Table 35: Share of observations split by the di erent features K2204 PPV linear MBTs

Figure 31: B-spline surrogate predictions vs. PPV$_{pred}$ for K2204

The delity results for all B-spline surrogates are listed in Table A.4.2 .
Table A.4.2 shows the proportions of observations that were split according to the di erent features.

	R^2	MSE	MAE	max AE	n leaves
B-spline baseline model	0.9943260	0.9604583	0.7605289	4.249894	1
SLIM shallow	0.9994172	0.0986492	0.2046541	1.880490	8
GUIDE shallow	0.9993896	0.1033306	0.2132615	1.849715	8
MOB shallow	0.9992104	0.1336634	0.2494695	2.063691	8
CTree shallow	0.9990906	0.1539409	0.2713325	2.069590	8
SLIM deep	0.9997505	0.0422310	0.1208647	1.421309	21
GUIDE deep	0.9997299	0.0457276	0.1315567	1.307047	20
MOB deep	0.9996846	0.0533855	0.1510094	1.342777	21
CTree deep	0.9997083	0.0493729	0.1451053	1.385141	20

Table 36: Fidelity of K2204 PPV B-spline baseline model and B-spline MBTs

	age	duration	sex
SLIM shallow	0.38	0.62	0.00
GUIDE shallow	0.30	0.70	0.00
MOB shallow	0.08	0.84	0.08
CTree shallow	0.23	0.71	0.07
SLIM deep	0.35	0.60	0.04
GUIDE deep	0.20	0.78	0.02
MOB deep	0.08	0.76	0.16
CTree deep	0.20	0.67	0.13

Table 37: Share of observations split by the di erent features K2204 PPV B-spline MBTs

Figure 32: SLIM tree for K2204 PPV with B-spline models

	R^2	MSE	MAE	max AE
SLIM	0.9997757	0.0383432	0.1140380	1.3335727
Blackbox model	0.9998305	0.0289638	0.1060365	0.5345493

Table 38: Accuracy of standalone B-spline SLIM MBTs and of the black box model K2204 PPV

Figure 33: Input-output relation of features in nodes split by duration for SLIM tree with B-splines and depth 3

Figure 34: Input-output relation of features in nodes with duration 25 split by age for SLIM tree with B-splines and depth 3

Figure 35: Input-output relation of features in nodes with 25 < duration <= 40 split by age for SLIM tree with B-splines and depth 3

B Electronic appendix

Data, code and figures are provided in electronic form at `https://github.com/slds-lmu/msc_2022_loibl_thesis`. All simulations and evaluations were carried out using the statistical software `R` (R Core Team, 2022). The simulations were carried out on the Linux cluster of the Leibniz Supercomputing Centre using the package `batchtools` (Lang et al., 2017). The author gratefully acknowledge the Leibniz Supercomputing Centre for funding this project by providing computing time on its Linux-Cluster. For data manipulation, the packages `data.table` (Dowle and Srinivasan, 2021), `dplyr` (Wickham et al., 2022) and `stringr` (Wickham, 2022) were used. Visualization was done with `ggplot2` (Wickham, 2016), `GGally` (Schloerke et al., 2021), `ggpubr` (Kassambara, 2020) and `igraph` (Csardi and Nepusz Tamas, 2006). For fitting the black box xgboost models, the package `mlr3` (Lang et al., 2019) was used.

Declaration of authorship

I hereby declare that the report submitted is my own unaided work. All direct or indirect sources used are acknowledged as references. I am aware that the Thesis in digital form can be examined for the use of unauthorized aid and in order to determine whether the report as a whole or parts incorporated in it may be deemed as plagiarism. For the comparison of my work with existing sources I agree that it shall be entered in a database where it shall also remain after examination, to enable comparison with future Theses submitted. Further rights of reproduction and usage, however, are not granted here. This paper was not previously presented to another examination board and has not been published.

Location, date

Name