Correlation and Unintended Biases on Univariate and Multivariate Decision Trees

Mattia Setzu
Department of Computer Science
University of Pisa
Pisa, Italy
mattia.setzu@unipi.it

Salvatore Ruggieri
Department of Computer Science
University of Pisa
Pisa, Italy
salvatore.ruggieri@unipi.it

***Abstract*—Decision Trees are accessible, interpretable, and well-performing classification models. A plethora of variants with increasing expressiveness has been proposed in the last forty years. We contrast the two families of *univariate* DTs, whose split functions partition data through axis-parallel hyperplanes, and *multivariate* DTs, whose splits instead partition data through oblique hyperplanes. The latter include the former, hence multivariate DTs are in principle more powerful. Surprisingly enough, however, univariate DTs consistently show comparable performances in the literature. We analyze the reasons behind this, both with synthetic and real-world benchmark datasets. Our research questions test whether the pre-processing phase of removing correlation among features in datasets has an impact on the relative performances of univariate vs multivariate DTs. We find that existing benchmark datasets are likely biased towards favoring univariate DTs.**

I. Introduction

Decision Trees (DTs) [1] are accessible, interpretable, and well-performing classification models that are commonly used in practice. These models are widely available across software libraries and are standard baselines in industry and academic communities [2]. DTs are inherently transparent [3], which may facilitate the inclusion of stakeholders in understanding and assessing model behaviour. Moreover, ensembles of DTs outperform deep learning models on tabular data [4]. Trending research topics include the design of DT learning algorithms able to reproduce the behavior of complex and opaque models (e.g., of a tree ensemble [5]–[8]), and the design of DT learning algorithms able to adapt to distributions of data different from the one of the training dataset [9].

The condition tested at internal nodes of a decision tree can be broadly categorized as *univariate* or *multivariate*, the former yielding axis-parallel splits, and thus producing univariate DTs (UDTs), and the latter yielding oblique splits, and thus producing multivariate DTs (MDTs). These two families of DTs have found wildly different degrees of success in practical use, and in the scientific research. Multivariate DTs are, in principle, more expressive than the univariate ones, being capable of linearly separating the space of instances. Such an expressivity comes at a higher computational cost and an interpretational gap. Surprisingly enough, when compared head-to-head [10], there is not a clear-cut performance difference between the two types of DTs. For this, MDTs are often dismissed in favor of univariate ones – an application of the Occam's razor.

In this paper, we analyze the reasons behind this unexpected behavior by analytically comparing the two families of DTs, and by identifying a bias in standard dataset pre-processing practices that favors UDTs. We further support our theoretical results with an extensive experimentation on both synthetic and real-world datasets, which confirms the above bias.

The paper is structured as follows. In Section II, we define background notions on DTs. In Section III, we pose our research questions. Experimental results are discussed in Section IV. Finally, in Section V, we summarize our conclusions.

II. Background

A. Decision Trees

A decision tree T is a tree data structure comprised of N nodes $\{n_i\}_{i=1}^{N}$, either internal nodes or leaves. Each internal node points to a number of child nodes, the degree of the node, based on the possible outcomes of the split function at the node. Leaf nodes terminate the tree and are associated with task-dependent values, e.g., in a classification task, a leaf holds an estimated probability distribution over class labels. Top-down tree learning algorithms, such as CART [11] and C4.5 [12], follow a greedy approach, in which the tree structure and the split functions are learned recursively. Optimal decision tree learning algorithms [13] compile a complete binary tree of given depth to a Mixed Integer Linear Programming (MILP) problem. For uniformity, in the former approach, we also restrict to a tree depth stopping criteria and to binary split functions. Moreover, we disregard pre-processing steps, such as feature selection [14], and post-processing steps, such as tree simplification [15]. For a recent survey on DTs, see [16].

B. Split functions

We assume binary split functions (or, simply, splits) $f_i : \mathcal{X} \rightarrow \{0, 1\}$, that at an internal node n_i deterministically redirect an instance $x \in \mathcal{X} \subseteq \mathrm{R}^m$ towards the left ($f_i(x) = 0$) or the right ($f_i(x) = 1$) child node. Some approaches learn a fuzzy split condition [17], for which the instance is probabilistically redirected to one of the child nodes. Split functions can vary in complexity, from being as simple as a univariate split [11], [12] or as complex as a neural network [18]. The most common split functions are of the form *1-of-1* splits, that is, they are comprised of a single boolean condition, for which instances are redirected according to the result of the condition. In multiconditional DTs, such as Trepan [19], split functions consist of testing k boolean conditions, h of which must be satisfied – an h-of-k split policy. This introduces a significant cost in terms of both learning complexity and interpretability. In the rest of this paper, we restrict to uniconditional *linear* split functions of the form:

$$f_i(x) = \mathbb{1}(\alpha_i^T x \leq \beta_i) \tag{1}$$

where $\alpha_i \in \mathbb{R}^m, \beta_i \in \mathbb{R}$ are the parameters of the split function f_i. Univariate splits implement univariate separating hyperplanes, that is, α_i is 0 but for a single component for which it is 1. Multivariate splits allow for more than one non-zero component, hence implementing separating hyperplanes with an arbitrary number of non-zero components.

C. Univariate splits

Let us denote by n_0, n_1, and n_2 a parent node and its two child nodes respectively. Also, let X_i and Y_i, for $i = 0, 1, 2$, be the features and class attributes respectively, of the data instances at the three nodes. Univariate splits test a condition $x_j \leq \beta$, where x_j with $j \in 1, \ldots, m$ is a feature in X_0 and and β is in the domain of x_j. According to [20], we distinguish the following strategies.

Impurity-based splits. Impurity measures the heterogeneity of class label frequency for instances at a node. We can define vectors p_1 and p_2 holding the empirical class distribution of the instances at n_1 and n_2, i.e., called $\mathit{freq}(y, Y_i)$ the frequency of class label y in Y_i:

$$p_1 = \left\langle \frac{\mathit{freq}(y, Y_1)}{\mid Y_1 \mid} \right\rangle_{y \in \mathcal{Y}} \qquad p_2 = \left\langle \frac{\mathit{freq}(y, Y_2)}{\mid Y_2 \mid} \right\rangle_{y \in \mathcal{Y}} \tag{2}$$

Given p_1 and p_2, measuring impurity amounts to measuring their distance to the standard basis of the class space: the larger the distance, the more impure the split. Conventionally, the distance is then mapped to a single a scalar through a weighted sum. Typically, Euclidean or cosine distance is adopted [21]. A notable impurity measure is the Gini, namely the expected probability of incorrectly labeling a random instance:

$$\mathit{Gini}(n_i) = 1 - \sum_{y \in \mathcal{Y}} \left(\frac{\mathit{freq}(y, Y_i)}{\mid Y_i \mid} \right)^2$$

The difference between the Gini of the parent node and the weighted average Gini of the child nodes quantifies the increase in impurity after a candidate split. Gini-based approaches, such as CART [11], select the split which maximizes such an increase.

Information Gain splits. Entropy measures the information contained within a random variable based on the uncertainty of its events [22]. The standard is *Shannon's entropy* [23], which for a node n_i is defined as:

$$H(n_i) = \sum_{y \in \mathcal{Y}} -\frac{\mathit{freq}(y, Y_i)}{\mid Y_i \mid} \log \frac{\mathit{freq}(y, Y_i)}{\mid Y_i \mid}$$

Therefore, entropy is the expected information of the class distribution at the node n_i. The difference of entropy of the parent node and the weighted average entropy of the child nodes is the Information Gain of a split:

$$IG = H(n_0) - \sum_{i \in \{1,2\}} \frac{\mid Y_i \mid}{\mid Y_0 \mid} H(n_i)$$

Selecting the split for which the Information Gain is maximized is a popular strategy, which has been adopted in several learning algorithms, including ID3 [24] and GID3 [25], C4 [26] and C4.5 [27].

Statistical test splits. Other strategies look for the best split through purely statistical tests or confidence intervals. Starting from the contingency tables of true and predicted class labels at the nodes n_0, n_1 and n_2, a number of statistical measures of associations can be considered to be maximized [28]–[30].

D. Multivariate splits

Unlike univariate splits, the search space of parameters α_i's and β_i's in (1) for multivariate splits cannot be efficiently enumerated. Instead, they are determined through specialized optimization algorithms. Such an optimization introduces a significant computational overhead.

Margin optimization. Given their success in a host of applications, Support Vector Machines [31] (SVMs) are a natural candidate for optimizing class separation. Provided with linear kernels, SVM-based multivariate trees have been implemented in SVM Trees [32], Geometric Trees [33], and in Reflector-based Trees such as CartOpt [34] and HHCart [35].

Linear optimization. Linear Trees are families of DTs that directly optimize a linear model at each node. Linear Discriminant Trees [10] and QUEST [36] build on top of Fisher's linear discriminant, a statistical method to find linear combinations separating samples. Linear Machines [37] relies on the encoding of a DT into a set of one-vs-all linear models. Weighted Oblique Decision Trees [38] look to optimize a differentiable formulation of entropy in which samples are first mapped through a nonlinear function.

Composed optimization. Trees based on composed optimization aim to leverage univariate splits to either learn mixed trees, or use the univariate split as an initialization for the multivariate one. For instance, OC1 [39] initializes each multivariate split with a univariate one derived from CART, which is then replaced by a better multivariate one learned through a series of optimization techniques. Directly extending its univariate counterpart, CART-LC [11] optimizes a local linear model, which is then simplified by pruning coefficients. Model Trees [40] instead take a mixed approach and first build a CART Tree, by progressively replacing the univariate splits at the penultimate layer with multivariate ones, each iteration pruning the replaced subtree, only to stop when no further performance gain is achieved.

Other forms of optimization. Several learning algorithms employ complex nonlinear and nondifferentiable optimization techniques: Genetic Trees [41] and Evolutionary Trees [42]

	Mean $\pm$ Stdev	Min	Max
Dataset size	196,907 $\pm$ 1,301,496	68	9,999,889
Dimensionality	649.22 $\pm$ 2,942.14	4	20,001
Dimensionality ratio	20,585.62 $\pm$ 144,532.71	0.00	1,111,098.77

TABLE I: Summary of the 57 benchmark datasets.

employ genetic programming, and Annealing Trees [43] employ simulated annealing. Another noteworthy family of approaches departs from a greedy search by encoding the tree learning into a global MILP optimization problem [13], [44]–[46]. The resulting DTs are called optimal trees.

E. Feature correlation

In this paper, we focus on feature correlation as a major property of interest due to its generality and possible impact on DT performances. Correlation, and even more so multicollinearity, relates multiple features by quantifying the linear relationship occurring among them. Generally, given two linearly dependent features x_i, x_j, we can express one in linear terms of the other, i.e.:

$$x_i = \alpha x_j + \beta + \epsilon,$$

for suitable $\alpha, \beta \in \mathbb{R}$ and small error ϵ. The same relationship can be found in sets of Δ features $x_i, \ldots, x_{i+\Delta}$, in which case we are dealing with *multicollinearity*, that is:

$$x_i = \sum_{j=i+1}^{\Delta} \alpha_j x_j + \beta + \epsilon,$$

again for suitable $\alpha_{i+1}, \ldots, \alpha_{i+\Delta}, \beta$ and small error ϵ. Perfect multicollinearity is rare in practice, and sets of features are said to enjoy collinearity even if they are in an approximately linear relationship. Multicollinearity leads to several problems in regression analysis, which include high instability in the optimization algorithm itself and large variance in the expected results. As a consequence, removal of highly correlated, or approximately collinear, sets of features has been always a standard practice in dataset pre-processing [47]. Standard metrics to measure collinearity, such as Variance Inflation Factors [48], are tailored to specific cases, e.g., linear regression, hence dataset- and task-agnostic proxy metrics, such as correlation, are typically used instead. Correlation can be quantified in many forms, each trying to detect a slightly different relationship between pairs of features (x_i, x_j). Pearson's correlation ρ^P measures linear correlation through normalized covariance. Spearman's correlation ρ^S instead aims to measure monotonic correlation through ranks of features. Kendall's correlation ρ^τ, also known as Kendall's τ, measures concordance of order, that is, the probability of observing a difference between concordant and discordant pairs w.r.t. the orderings assigned by x_i and x_j.

III. Research Questions

Regarding univariate splits, [49] observes that the "choice of split-selection metric typically has little effect on accuracy, but can profoundly affect complexity and the effectiveness and efficiency of pruning". Regarding Linear Discriminant Trees, while "the proposed method is accurate, learns fast, and generates small trees, [...] results indicate that in the majority of cases, a univariate method suffices" [10].

The reasons for the above conclusions have not been sufficiently clarified in the (forty years old) literature on DT learning. We intend to answer the following questions.

RQ1. *Do feature correlation or other factors, such as complexity of decision boundary and label noise, impact on the performances of the split functions?* In particular, on the relative strength of univariate vs multivariate splits? By answering this research question we aim to understand the relationship between the correlation among the features and the UDTs/MDTs performances. Detecting such an impact will allow to guide practitioners on the choice of UDTs vs MDTs.

RQ2. *Are standard benchmark datasets used for evaluating DT learning algorithms biased?* We analyze standard benchmark datasets that have been and/or are currently used in the literature. Identifying common patterns of such datasets, i.e., with regard to feature correlation, will then allow us to understand whether experiments in the literature have relied on biased collections of datasets.

RQ3. *Does the bias in benchmark datasets transfer to a biased evaluation of the performances of DT learning algorithms?* On the basis of the answers to RQ1 and RQ2, we want to understand if bias in benchmark datasets plus dependence of split functions from correlation of features and other factors, turn out to impact the experimental results in favor of UDTs vs MDTs.

IV. Experiments

A. Benchmark datasets

Following [50], we base the bulk of our analysis on a large subset of UCI[1] datasets. Moreover, we include several datasets from the OpenML repository[2], and from the Kaggle platform[3], for a total of 57 datasets[4]. We present a full list of datasets in Table VII in the Appendix, and a short summary in Table I. Datasets vary wildly in size, ranging from ≈ 70 to ≈ 10M instances, in dimensionality, ranging from 4 to 20,000 features, and in the size to dimensionality ratio, ranging from 0.009 to 110,000. We left in datasets only the numeric attributes, since we restrict to linear split functions of the form (1). Finally, feature values are z-score normalized.

B. Synthetic datasets

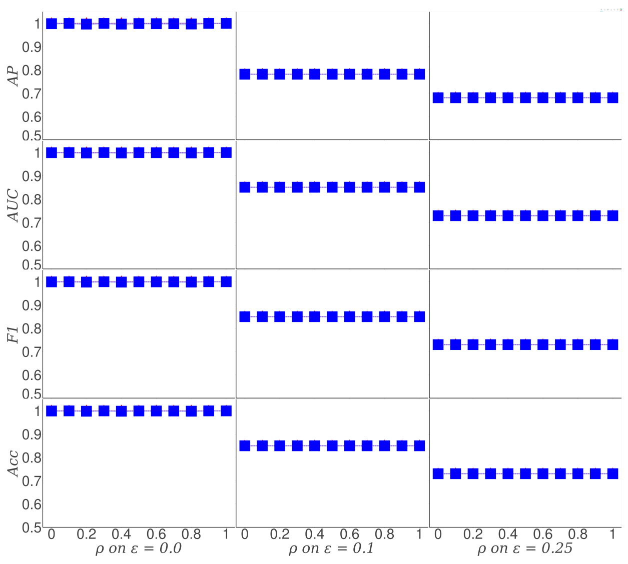
To best estimate the effects of feature correlation and label noise on DT learning algorithms, we experiment also

[1]https://archive-beta.ics.uci.edu

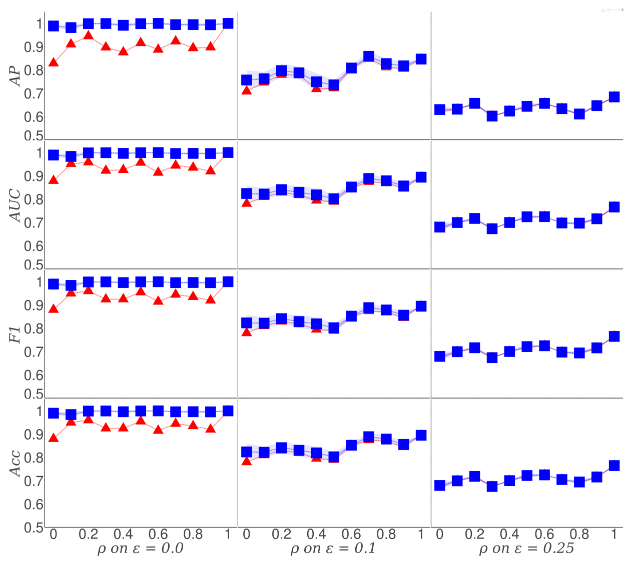
[2]https://www.openml.org

[3]https://www.kaggle.com

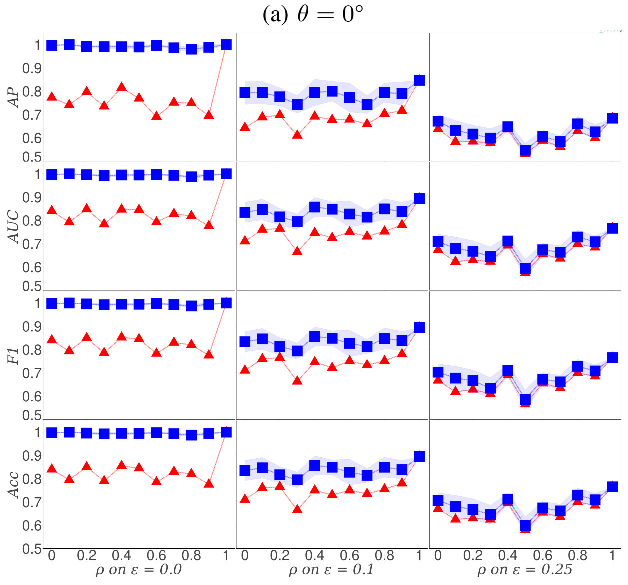
[4]Several datasets from the original paper [50] have been removed due to their extremely low number of features.



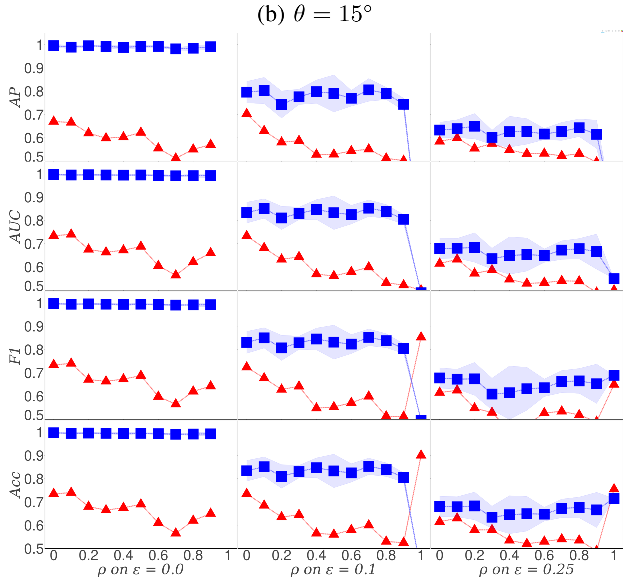
(a) $\theta = 0°$



(b) $\theta = 15°$



(c) $\theta = 30°$



(d) $\theta = 45°$

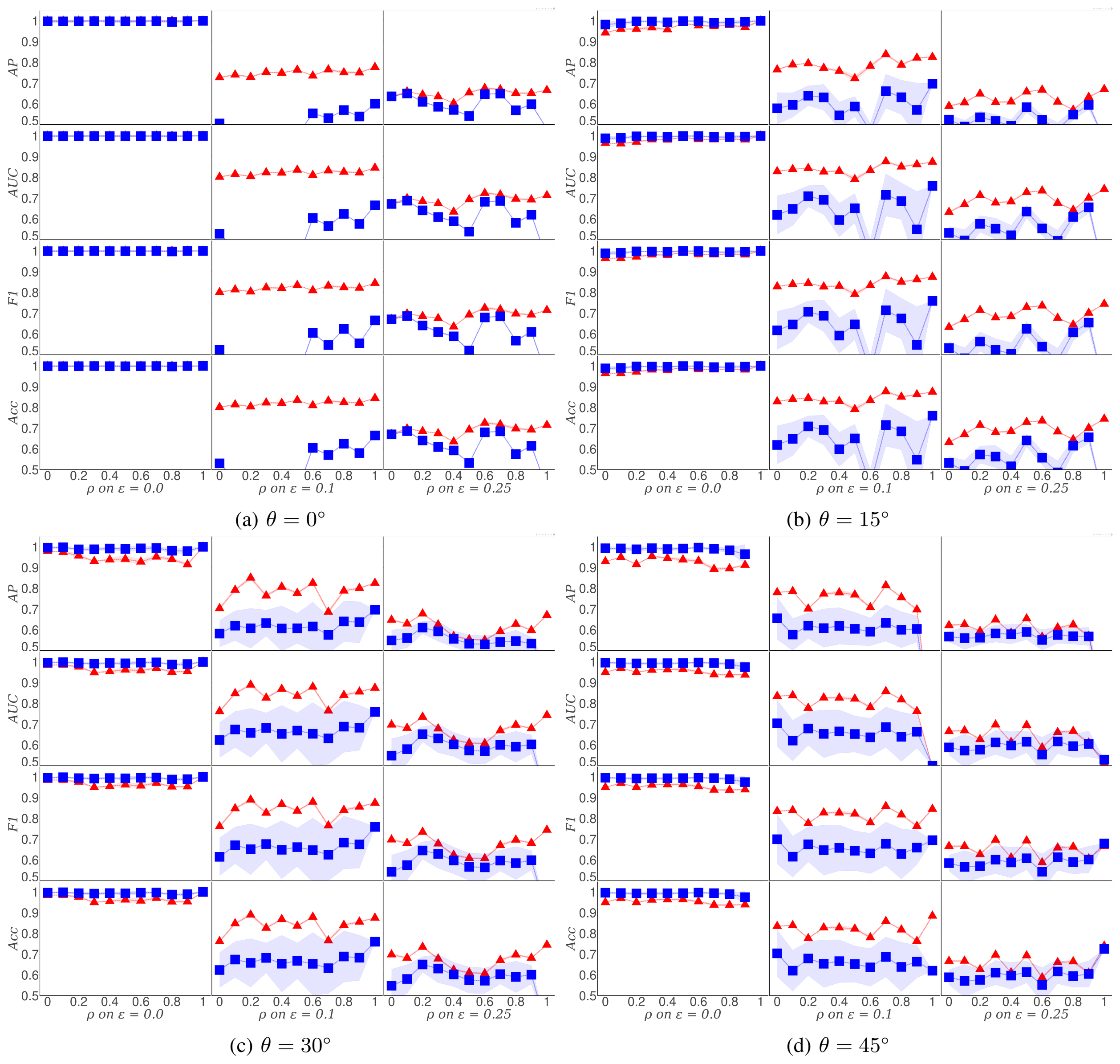
Fig. 1: Performances of univariate and multivariate DTs with maximum depth of 1 on synthetic datasets with slope angle θ, correlation ρ and noise ϵ. Legend: red triangles for UDTs, blue squares for MDTs.

θ	Accuracy		F1		AUC		AP	
	Test outcome	p value	Test outcome	p value	Test outcome	p value	Test outcome	p value
$0°$	0.005	10^{-1}	0.005	10^{-1}	0.005	10^{-1}	0.005	10^{-1}
$15°$	-0.136	10^{-28}	-0.136	10^{-28}	-0.132	10^{-27}	-0.144	10^{-30}
$30°$	-0.159	10^{-32}	-0.159	10^{-32}	-0.158	10^{-32}	-0.159	10^{-32}
$45°$	-0.270	10^{-68}	-0.270	10^{-68}	-0.273	10^{-69}	-0.246	10^{-59}

TABLE II: Paired t-test of the performance of univariate and multivariate DTs with maximum depth of 16 on synthetic datasets with $\epsilon = 0$ and slop angle θ.

with synthetic data in which we control for these factors. We generate datasets with two normally-distributed features, X_1 and X_2, including $1,000$ instances, and with Pearson's correlation[5] $\rho^P = \rho$ ranging from 0 to 1 in 0.1 steps. The binary class label is defined as $Y = \mathbb{1}(X_2 > m \cdot X_1)$,

[5] X_1 and X_2 are generated as follows. Starting from $X_1, Z \sim N(0,1)$, we have that $X_2 = \rho \cdot X_1 + \sqrt{1-\rho^2} \cdot Z \sim N(0,1)$ and $Cor(X_1, X_2) = \rho$.



(a) $\theta = 0°$ (b) $\theta = 15°$ (c) $\theta = 30°$ (d) $\theta = 45°$

Fig. 2: Performances of univariate and multivariate DTs with maximum depth of 16 on synthetic datasets with slope angle θ, correlation ρ and noise ϵ. Legend: red triangles for UDTs, blue squares for MDTs.

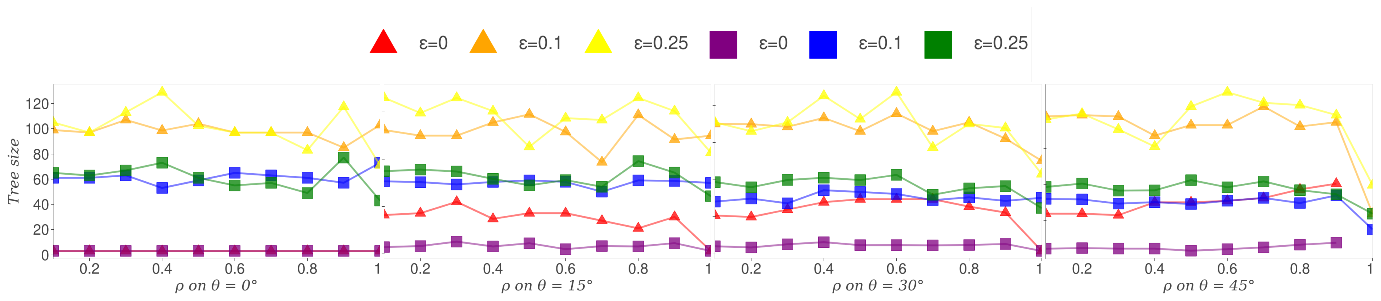
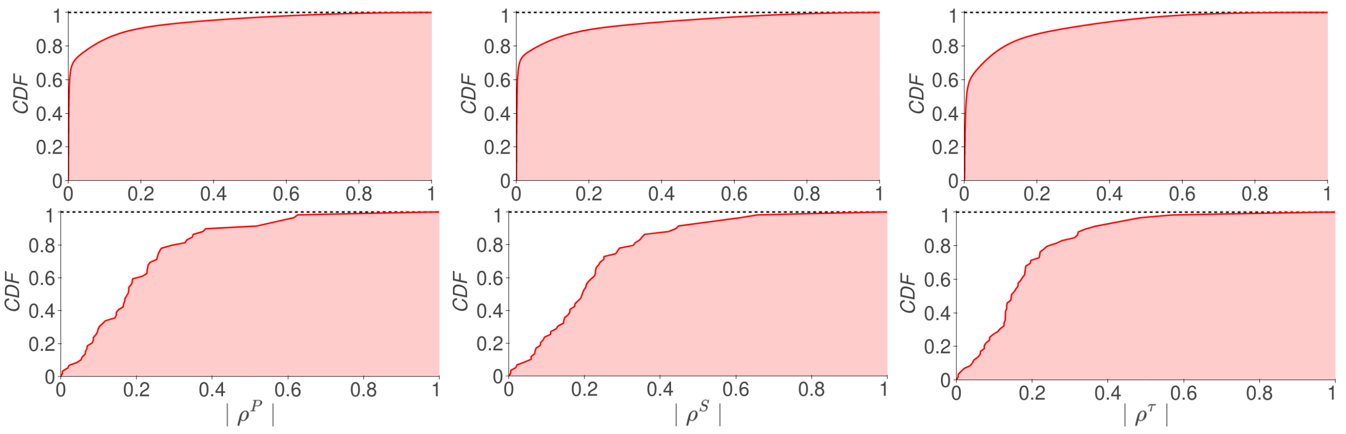


Fig. 3: Model complexity of univariate and multivariate DTs with maximum depth of 16 on synthetic datasets with slope angle θ, correlation ρ and noise ϵ. Legend: triangles for UDTs, squares for MDTs.



(a) Pearson (b) Spearman. (c) Kendall.

Fig. 4: Empirical CDFs of absolute feature correlation over the benchmark datasets. Top: any pair of features in a same dataset. Bottom: mean correlation per dataset.

where the parameter $m = tan(\theta)$ is the slope of the decision boundary and θ is the slope angle. Moreover, each dataset is further replicated and perturbed to introduce some noise by randomly flipping each label according to the outcome of a Bernoulli trial with parameter $\epsilon \in \{0, 0.1, 0.25\}$. The definition of the class feature Y is specifically intended to distinguish the cases when multivariate splits are theoretically the best solution ($0° \ll \theta \ll 90°$) from cases where axis-parallel splits are sufficient ($\theta \approx 0°$ or $\theta \approx 90°$) . Experiments will then show the impact of feature correlation and label noise in such contexts.

C. DT learning algorithms

For univariate DTs, we use CART [11], while for multivariate DTs we implemented an Omnivariate Tree [10]. Our implementation tests several splits at each node, namely: an SVM split [32], a gradient-SVM split, a Ridge split, a Least Squares split, an Elastic Net split [51], a Lasso split, and a CART split. At each node, all splits are evaluated, and the one yielding the best Gini is selected. As in composed optimization DTs such as OC1 [39], we include a CART split in the candidates in case the data can be best separated with a univariate split. Due to their extremely large computational cost on moderate-to-large datasets, we do not include Optimal Trees in the experimentation.

D. Performance and model complexity metrics

Each synthetic and benchmark dataset is split into 90% training set and 10% test set by a stratified hold-out method. We evaluate the learned DTs through several performance metrics on the test set, including: F-measure (F1), AUC-ROC (AUC), Average Precision (AP), and Accuracy (Acc). As for model complexity, we consider: the size of the DT, and, for MDTs, also the fraction of non-zero coefficients at its nodes.

E. Experimental Results

Let us consider our research questions.

RQ1. Do feature correlation, decision boundary slope, and label noise impact on the performances of the split functions?

Figure 1 shows the mean and standard deviation of the performances of DTs with a single univariate or multivariate split. Each quadrant referes to synthetic datasets with increasing slope angle $\theta \in \{0°, 15°, 30°, 45°\}$. Let's first consider the case of no noise, i.e., $\epsilon = 0$. By construction of the synthetic datasets, angles close to 0° should lead to axis-parallel splits, while angles close to 45° should favor oblique splits. This is apparent in the plots, where for $\theta = 0°$ there is no difference between UDTs and MDTs, and for $\theta = 45°$ the MDTs perform much better than UDTs. Moreover, for $\theta = 45°$, the gap between MDTs and UDTs increases with correlation ρ. This can be explained by observing that the larger the ρ the more the data instances are closer to the decision boundary $Y = \mathbb{1}(X_2 > X_1)$ of $\theta = 45°$, hence it becomes more and more difficult to separate them with axis-parallel splits. For moderate ($\epsilon = 0.1$) to large ($\epsilon = 0.25$) noise, the gap between MDTs and UDTs decreases. Intuitively, the decision boundary becomes more complex, and neither an oblique nor an axis-parallel split are adequate.

Figure 2 reports the same analysis for DTs of maximum depth 16. For zero noise, the performance gap is smaller than in the previous case, as UDTs are now larger and can better separate the decision boundary. However, Table II shows that such differences are statistically significant. For moderate noise ($\epsilon = 0.1$), something surprising can be observed: UDTs perform better than MDTs, which in addition appear to be unstable. We explain this by the fact that MDTs overfit the decision boundary with unnecessarily complex oblique splits. This is particularly striking for $\theta = 0°$, where axis-parallel splits are enough. For large noise ($\epsilon = 0.25$), the performances of UDTs are still better than those of MDTs, but the gap is smaller – as the irregularity of the decision boundary makes both axis-parallel and oblique splits inadequate. Regarding model complexity, Figure 3 shows that, regardless of label

	Accuracy	F1	AUC	AP
UDTs				
TR	0.932 ± 0.091	0.928 ± 0.098	0.886 ± 0.133	0.818 ± 0.230
TS	0.844 ± 0.125	0.837 ± 0.134	0.771 ± 0.163	0.635 ± 0.281
MDTs				
TR	0.811 ± 0.213	0.816 ± 0.211	0.785 ± 0.203	0.687 ± 0.312
TS	0.720 ± 0.207	0.720 ± 0.215	0.683 ± 0.195	0.558 ± 0.296

TABLE III: Performances of UDTs and MDTs on benchmark datasets. TR = training set. TS = test set.

	Accuracy	F1	AUC	AP
mean	$.122 \pm .170$	$.116 \pm .169$	$.900 \pm .148$	$.794 \pm .135$
min	$-.143$	$-.212$	$-.196$	$-.302$
max	$.681$	$.770$	$.642$	$.486$
UDT wins	49	47	46	46
MDT wins	8	10	11	11

TABLE IV: Performance gap between UDTs and MDTs on benchmark datasets (test set).

	Tree size	Non-zero coefficients ratio
UDTs	57.00 ± 146.719	-
MDTs	59.00 ± 75.544	0.409 ± 0.241

TABLE V: Model complexity of UDTs and MDTs on benchmark datasets.

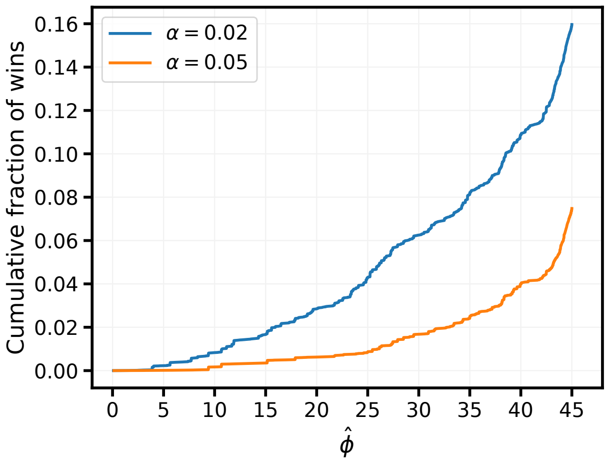


Fig. 5: Cumulative weighted fraction of multivariate splits wins in benchmark datasets at the variation of slope angle estimate $\hat{\phi}$ and threshold α.

noise, UDTs are consistently larger than MDTs, with up to 50% additional nodes.

In summary, the presence of noise appears to have a major impact in favor of UDTs performances, but at the cost of higher model complexity. In absence of noise, MDTs are better than UDTs if the decision boundary is actually oblique. In such a case, the larger the correlation between features the larger the gap with UDTs.

Partition	Size	Correlation		$\hat{\phi}$
MDT $\gg$ UDT	4	$\lvert\rho^\tau\rvert$	0.448 ± 0.132	$29.66° \pm 13.61°$
		$\lvert\rho^P\rvert$	0.310 ± 0.391	
		$\lvert\rho^S\rvert$	0.449 ± 0.247	
UDT $\gg$ MDT	8	$\lvert\rho^\tau\rvert$	0.035 ± 0.123	$21.81° \pm 13.44°$
		$\lvert\rho^P\rvert$	0.200 ± 0.306	
		$\lvert\rho^S\rvert$	0.148 ± 0.167	
MDT $\approx$ UDT	45	$\lvert\rho^\tau\rvert$	0.188 ± 0.166	$22.06° \pm 14.05°$
		$\lvert\rho^P\rvert$	0.218 ± 0.183	
		$\lvert\rho^S\rvert$	0.218 ± 0.183	

TABLE VI: Correlation and $\hat{\phi}$ estimate in benchmark datasets, by group.

RQ2. Are standard benchmark datasets used for evaluating decision tree algorithms biased?

Figure 4 shows the empirical cumulative distributions (CDFs) of Pearson, Spearman, and Kendal correlation coefficients over the benchmark datasets. The top CDFs regard the correlation of any pair of features in the same dataset. The bottom CDFs regard the mean correlation at a dataset level. The feature correlation is extremely low over all pairs of features, as shown by the steep CDF curves at the top. Small mean correlation is also apparent in the CDFs at dataset level, with half of them having a correlation lower or equal than 0.2. Such a bias in favor of low correlated features can be traced back to the long-standing issue of collinearity in linear models, for which inference algorithms are able to converge faster and to less biased solutions.

According to the experiments on synthetic data, the other factors affecting performance regard the decision boundary (the slope angle θ) and the noise (ϵ). However, we do not have ground truth knowledge to measure them over the benchmark datasets. We proceed with a (weak) approximation as follows. Consider any distinct pair x_1, x_2 of features in a dataset[6]. We test whether the decision boundary w.r.t. these two-dimensional features can be better described by an univariate or a multivariate split. To this end, we train three Linear SVMs using as features: (1) x_1 only, (2) x_2 only, and (3) both x_1 and x_2. We say that the multivariate split from (3) *wins* over the univariate splits in (1, 2) if the accuracy[7] of the Linear SVM using (3) is greater than $1+\alpha$ times the accuracy of both Linear SVMs using (1) or (2). Here, α is a threshold to ensure that the improvement in accuracy is substantial. We experiment with $\alpha = 0.02$ (two percent) and $\alpha = 0.05$ (five percent). The fraction of pairs of features for which the multivariate split wins – or simply, the *fraction of wins* – is a raw metric of degree of "obliqueness" of the decision boundary of the dataset. Moreover, we estimate the slope angle of the Linear SVM in (3) as $\hat{\theta} = tan^{-1}(-\beta_1/\beta_2)$, where β_1, β_2 are the coefficients of x_1 and x_2 respectively. This approximation is weak if the unknown noise (ϵ) is not small, or if the true

[6]For the high dimensional datasets *arcene*, *dexter*, and *gisette*, we considered random samples of 1M, 500K and 100K pairs respectively.

[7]Since we are not building predictive models in this task, the whole dataset is used both for training and for calculating the accuracy.

decision boundary is not linear in x_1, x_2. We perform the following transformation that makes the slope angle estimate independent of the order of the two features x_1 and x_2 and of the sign of the angle:

$$\hat{\phi} = \min\{\ |\hat{\theta}|,\ 90^\circ - |\hat{\theta}|\ \}$$

Thus, $\hat{\phi} \in [0^\circ, 45^\circ]$. Figure 5 shows the cumulative weighed[8] fraction of wins at the variation of $\hat{\phi}$ for different thresholds α. Remarkably, multivariate splits win over univariate splits with an improvement of at least 2% in less than 16% of the pairs, and with an improvement of at least 5% in less than 8% of the pairs. Therefore, decision boundaries of the benchmark datasets are mostly axis-parallel. Moreover, the skewed distributions in Figure 5 also highlight that multivariate splits win mostly for large values of the slope angle $\hat{\phi}$.

In summary, the benchmarks datasets exhibit distributions of feature correlation and of decision boundary slope that are skewed towards low correlation values and approximatively axis-parallel decision boundaries.

RQ3. Does the bias in benchmark datasets transfer to a biased evaluation of the performance of DT learning algorithms?

Table III reports the aggregate predictive performances of univariate and multivariate DTs on the benchmark datasets. Univariate DTs consistently achieves better Accuracy, F1, Average Precision, and AUC, as also shown in Table IV, alongside the number of wins per model. Interestingly, univariate DTs are better also on the training set, which suggests that the decision boundaries in the datasets lead multivariate DTs to largely overfit the data. The complexity of the learnt models, shown in Table V, is in favor of multivariate DTs, especially for what regards variability of tree size. However, the ratio of non-zero coefficients is, on average, quite large – and this may limit the benefit of having smaller trees if the objective is to achieve model comprehensibility.

The vast majority of datasets have performance gaps between UDTs and MDTs within one standard deviation from the mean, for each of the considered metrics. Thus, we characterize a partition of datasets into three groups: when MDTs perform better than UDTs by at least one standard deviation (MDT $\gg$ UDT) for the F1 metric, the opposite case (UDT $\gg$ MDT), and when the performances are within one standard deviation (UDT $\approx$ MDT). Table VI reports the mean $\pm$ stdev of the correlation coefficients, and of the slope angle estimate $\hat{\phi}$ over the pairs of features of the datasets in each group. For the small group MDT $\gg$ UDT, correlation is moderate-to-large and $\hat{\phi}$ is larger than for the other two groups. This is consistent with the results of experiments with synthetic data, where MDTs perform better than UDTs for large slope angles and large correlations. For the other two groups UDT $\approx$ MDT and UDT $\gg$ MDT, correlation is low and the estimated slop angles is, on average, lower than for the MDT $\gg$ UDT group. Again, this is in line with the case when oblique splits do not outperform axis-paralell splits. There is no clear difference between the groups UDT $\approx$ MDT and UDT $\gg$ MDT in terms of slope angle, while correlations coefficients of UDT $\gg$ MDT are smaller than those of UDT $\approx$ MDT. From our theoretical analysis, we can relate the different performance gaps of those two groups partly to different feature correlation and partly to different degrees of label noise. Unfortunately, without ground truth on the class attributes, we cannot test the latter hypothesis.

In summary, the observed performances of UDTs and MDTs can be largely explained by the factors analyzed in this paper: feature correlation, slope of the decision boundary, and, we conjecture, label noise. Based on the fact that benchmark datasets are skewed towards low correlation and axis-parallel decision boundaries, the fact that the observed performances are mostly in favor of UDTs are expected. The practitioner, then, should be warned about making the general conclusion that UDTs are always better than MDTs.

[8]Each pair of features is weighted by the inverse of the total number of pairs in the dataset. This is made necessary by the large difference in feature dimensionality among the datasets.

V. Conclusions

We have compared two families of DTs which differ as per expressive power of the split function: univariate and multivariate DTs. The latter generates smaller trees, and, in absence of noise, performs better than univariate DTs. Moreover, the larger the feature correlation the larger is the improvement in performance. This was established in answering RQ1. Next, we observed that standard benchmark datasets are preprocessed to remove correlation. Also, we found their decision boundary is approximatively "axis-parallel". This was established in answering RQ2. Finally, we tested whether the previous two answers lead to the conclusion that the better performances of univariate DTs observed in the literature is due to biases in the benchmark datasets, namely skewed correlation and "axis-parallel" decision boundaries. Our results support that the observed differences in performances can be explained by such factors. Consequently, practitioners are advised to test such factors on their datasets before making a choice whether to use MDTs or UDTs.

Datasets and code

The benchmark datasets can be downloaded from https://huggingface.co/mstz. Experimental Python code and scripts are available at https://github.com/msetzu/Univariate-vs-multivariate-decision-trees.

Acknowledgment

Research partly funded by PNRR - M4C2 - Investimento 1.3, Partenariato Esteso PE00000013 - "FAIR - Future Artificial Intelligence Research" - Spoke 1 "Human-centered AI", funded by the European Commission under the NextGeneration EU programme, and by the Excellent Science European Research Council (ERC) programme for the XAI project - "Science and technology for the explanation of AI decision making" (g.a. No. 834756). This work reflects only the authors' views and the European Research Executive Agency (REA) is not responsible for any use that may be made of the information it contains.

References

[1] L. Rokach and O. Maimon, "Decision trees," in *The Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach, Eds. Springer, 2005, pp. 165–192.

[2] C. Rudin, "A renaissance for decision tree learning," https://www.youtube.com/watch?v=bY7WEr6lcuY, 2016, keynote at PAPIs 2016.

[3] ——, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nat. Mach. Intell.*, vol. 1, no. 5, pp. 206–215, 2019.

[4] L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why do tree-based models still outperform deep learning on typical tabular data?" in *NeurIPS*, 2022.

[5] A. I. Weinberg and M. Last, "Selecting a representative decision tree from an ensemble of decision-tree models for fast big data classification," *J. Big Data*, vol. 6, p. 23, 2019.

[6] T. Vidal and M. Schiffer, "Born-again tree ensembles," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 119. PMLR, 2020, pp. 9743–9753.

[7] V. Bonsignori, R. Guidotti, and A. Monreale, "Deriving a single interpretable model by merging tree-based classifiers," in *DS*, ser. Lecture Notes in Computer Science, vol. 12986. Springer, 2021, pp. 347–357.

[8] E. Dudyrev and S. O. Kuznetsov, "Summation of decision trees," in *FCA4AI@IJCAI*, ser. CEUR Workshop Proceedings, vol. 2972. CEUR-WS.org, 2021, pp. 99–104.

[9] J. M. Álvarez, K. M. Scott, B. Berendt, and S. Ruggieri, "Domain adaptive decision trees: Implications for accuracy and fairness," in *FAccT*. ACM, 2023, pp. 423–433.

[10] O. T. Yildiz and E. Alpaydin, "Linear discriminant trees," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 19, no. 3, pp. 323–353, 2005.

[11] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Wadsworth, 1984.

[12] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986.

[13] D. Bertsimas and J. Dunn, "Optimal classification trees," *Mach. Learn.*, vol. 106, no. 7, pp. 1039–1082, 2017.

[14] S. Ruggieri, "Complete search for feature selection in decision trees," *J. Mach. Learn. Res.*, vol. 20, pp. 104:1–104:34, 2019.

[15] F. Esposito, D. Malerba, and G. Semeraro, "A comparative analysis of methods for pruning decision trees," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 5, pp. 476–491, 1997.

[16] V. G. Costa and C. E. Pedreira, "Recent advances in decision trees: an updated survey," *Artif. Intell. Rev.*, vol. 56, no. 5, pp. 4765–4800, 2023.

[17] J. Adamo, "Fuzzy decision trees," *Fuzzy sets and systems*, vol. 4, no. 3, pp. 207–219, 1980.

[18] P. Kontschieder, M. Fiterau, A. Criminisi, and S. R. Bulò, "Deep neural decision forests," in *IJCAI*. IJCAI/AAAI Press, 2016, pp. 4190–4194.

[19] M. W. Craven and J. W. Shavlik, "Extracting tree-structured representations of trained networks," in *NIPS*. MIT Press, 1995, pp. 24–30.

[20] K. Grabczewski, *Meta-Learning in Decision Tree Induction*, ser. Studies in Computational Intelligence. Springer, 2014, vol. 498.

[21] U. M. Fayyad and K. B. Irani, "The attribute selection problem in decision tree generation," in *AAAI*. AAAI Press / The MIT Press, 1992, pp. 104–110.

[22] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Wiley, 2001.

[23] T. Maszczyk and W. Duch, "Comparison of Shannon, Renyi and Tsallis entropy used in decision trees," in *ICAISC*, ser. Lecture Notes in Computer Science, vol. 5097. Springer, 2008, pp. 643–651.

[24] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986.

[25] U. M. Fayyad, "Branching on attribute values in decision tree generation," in *Proceedings of the 12th National Conference on Artificial Intelligence, Seattle, WA, USA, July 31 - August 4, 1994, Volume 1*, B. Hayes-Roth and R. E. Korf, Eds. AAAI Press / The MIT Press, 1994, pp. 601–606.

[26] J. R. Quinlan, "Probabilistic decision trees," in *Machine Learning*. Elsevier, 1990, pp. 140–152.

[27] ——, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[28] J. Mingers, "An empirical comparison of selection measures for decision-tree induction," *Mach. Learn.*, vol. 3, pp. 319–342, 1989.

[29] G. Katz, A. Shabtai, L. Rokach, and N. Ofek, "Confdtree: Improving decision trees using confidence intervals," in *ICDM*. IEEE Computer Society, 2012, pp. 339–348.

[30] R. D. Rosa and N. Cesa-Bianchi, "Splitting with confidence in decision trees with application to stream mining," in *IJCNN*. IEEE, 2015, pp. 1–8.

[31] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[32] F. Nie, W. Zhu, and X. Li, "Decision tree SVM: an extension of linear SVM for non-linear classification," *Neurocomputing*, vol. 401, pp. 153–159, 2020.

[33] N. Manwani and P. S. Sastry, "Geometric decision tree," *IEEE Trans. Syst. Man Cybern. Part B*, vol. 42, no. 1, pp. 181–192, 2012.

[34] B. L. Robertson, C. J. Price, and M. Reale, "Cartopt: a random search method for nonsmooth unconstrained optimization," *Comput. Optim. Appl.*, vol. 56, no. 2, pp. 291–315, 2013.

[35] D. C. Wickramarachchi, B. L. Robertson, M. Reale, C. J. Price, and J. Brown, "HHCART: an oblique decision tree," *Comput. Stat. Data Anal.*, vol. 96, pp. 12–23, 2016.

[36] W.-Y. Loh and Y.-S. Shih, "Split selection methods for classification trees," *Statistica Sinica*, pp. 815–840, 1997.

[37] C. E. Brodley and P. E. Utgoff, "Multivariate decision trees," *Mach. Learn.*, vol. 19, no. 1, pp. 45–77, 1995.

[38] B. Yang, S. Shen, and W. Gao, "Weighted oblique decision trees," in *AAAI*. AAAI Press, 2019, pp. 5621–5627.

[39] S. K. Murthy, S. Kasif, and S. Salzberg, "A system for induction of oblique decision trees," *J. Artif. Intell. Res.*, vol. 2, pp. 1–32, 1994.

[40] J. R. Quinlan, "Learning with continuous classes," in *Australian Joint Conf. on Artificial Intelligence*, vol. 92. World Scientific, 1992, pp. 343–348.

[41] M. Kretowski, "An evolutionary algorithm for oblique decision tree induction," in *ICAISC*, ser. Lecture Notes in Computer Science, vol. 3070. Springer, 2004, pp. 432–437.

[42] E. Cantú-Paz and C. Kamath, "Inducing oblique decision trees with evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 7, no. 1, pp. 54–68, 2003.

[43] D. G. Heath, S. Kasif, and S. Salzberg, "Induction of oblique decision trees," in *IJCAI*. Morgan Kaufmann, 1993, pp. 1002–1007.

[44] C. Bessiere, E. Hebrard, and B. O'Sullivan, "Minimising decision tree size as combinatorial optimisation," in *CP*, ser. Lecture Notes in Computer Science, vol. 5732. Springer, 2009, pp. 173–187.

[45] S. Verwer and Y. Zhang, "Learning decision trees with flexible constraints and objectives using integer optimization," in *CPAIOR*, ser. Lecture Notes in Computer Science, vol. 10335. Springer, 2017, pp. 94–103.

[46] H. Zhu, P. Murali, D. T. Phan, L. M. Nguyen, and J. Kalagnanam, "A scalable mip-based method for learning optimal multivariate decision trees," in *NeurIPS*, 2020.

[47] D. E. Farrar and R. R. Glauber, "Multicollinearity in regression analysis: The problem revisited," *Review of Economics and Statistics*, vol. 49, p. 92–107, 1967.

[48] D. A. Belsley, E. Kuh, and R. E. Welsch, *Regression diagnostics: Identifying influential data and sources of collinearity*. John Wiley & Sons, 2005.

[49] J. K. Martin, "An exact probability metric for decision tree splitting and stopping," *Mach. Learn.*, vol. 28, no. 2-3, pp. 257–291, 1997.

[50] M. F. Delgado, E. Cernadas, S. Barro, and D. G. Amorim, "Do we need hundreds of classifiers to solve real world classification problems?" *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3133–3181, 2014.

[51] J. K. Tay, B. Narasimhan, and T. Hastie, "Elastic net regularization paths for all generalized linear models," *J. Stat. Softw.*, vol. 106, no. 1, 2023.

Appendix

Dataset	# instances	# features	Accuracy		F1		AUC		AP	
			UDT	*MDT*	*UDT*	*MDT*	*UDT*	*MDT*	*UDT*	*MDT*
acute_inflammation	120	8	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
adult	36631	7	0.840	0.361	0.830	0.381	0.729	0.448	0.497	0.222
arcene	100	10000	0.7	0.55	0.700	0.390	0.707	0.5	0.594	0.45
arhythmia	68	279	0.714	0.571	0.726	0.589	0.725	0.625	0.826	0.773
australian_credit	690	7	0.724	0.608	0.724	0.609	0.720	0.611	0.611	0.510
balance_scale	625	4	0.872	0.856	0.857	0.866	0.473	0.602	0.08	0.120
bank	45211	9	0.885	0.493	0.851	0.579	0.552	0.492	0.168	0.115
blood	748	3	0.773	0.493	0.727	0.524	0.575	0.400	0.306	0.218
breast	683	9	0.948	0.919	0.949	0.919	0.946	0.909	0.882	0.825
car	1728	6	0.979	0.861	0.979	0.857	0.980	0.813	0.940	0.658
contraceptive	1473	8	0.701	0.559	0.697	0.554	0.686	0.579	0.686	0.618
compas	4534	12	0.985	0.886	0.985	0.863	0.975	0.671	0.923	0.438
covertype	581012	54	0.808	0.507	0.809	0.502	0.796	0.460	0.640	0.348
dexter	2599	20000	0.559	0.498	0.539	0.335	0.559	0.498	0.535	0.499
electricity	45312	8	0.813	0.652	0.813	0.654	0.806	0.650	0.705	0.525
fertility	99	6	0.85	0.8	0.875	0.837	0.916	0.888	0.4	0.333
german	1000	17	0.695	0.695	0.697	0.694	0.644	0.634	0.399	0.392
gisette	7000	5000	0.942	0.965	0.942	0.964	0.942	0.965	0.925	0.951
glass	214	9	0.860	0.930	0.860	0.923	0.462	0.654	0.069	0.213
heart_failure	299	12	0.733	0.633	0.729	0.633	0.677	0.576	0.459	0.360
heloc	10459	23	0.698	0.509	0.697	0.503	0.695	0.504	0.652	0.524
higgs	98049	28	0.681	0.523	0.681	0.522	0.679	0.526	0.646	0.542
hill	606	100	0.5	0.606	0.394	0.606	0.500	0.606	0.5	0.563
hypo	3269	23	0.909	0.640	0.897	0.719	0.610	0.576	0.177	0.102
ipums	299285	8	0.949	0.640	0.937	0.733	0.633	0.560	0.248	0.071
lrs	531	100	0.925	0.915	0.923	0.912	0.844	0.816	0.633	0.589
magic	19020	10	0.854	0.813	0.850	0.804	0.818	0.761	0.839	0.796
madelon	2000	500	0.73	0.527	0.729	0.527	0.73	0.527	0.666	0.514
house16	22784	16	0.854	0.770	0.853	0.767	0.818	0.711	0.871	0.807
ionosphere	351	34	0.887	0.859	0.888	0.855	0.885	0.827	0.900	0.846
musk	476	166	0.802	0.854	0.800	0.854	0.792	0.851	0.704	0.767
nbfi	8308	29	0.913	0.777	0.895	0.820	0.530	0.528	0.082	0.075
ozone	1847	72	0.921	0.910	0.920	0.910	0.691	0.649	0.226	0.170
page_blocks	5473	9	0.967	0.931	0.966	0.926	0.906	0.748	0.719	0.429
phoneme	5404	5	0.851	0.392	0.852	0.401	0.833	0.453	0.635	0.275
pima	768	8	0.714	0.370	0.702	0.368	0.656	0.408	0.477	0.316
pol	15000	48	0.969	0.393	0.969	0.396	0.968	0.326	0.973	0.600
pums	299285	10	0.949	0.693	0.937	0.772	0.633	0.643	0.248	0.092
planning	182	12	0.648	0.594	0.612	0.552	0.513	0.449	0.303	0.285
post_operative	87	8	0.722	0.5	0.677	0.525	0.561	0.469	0.322	0.266
seeds_0	210	7	0.833	0.976	0.822	0.975	0.767	0.964	0.650	0.952
seeds_1	210	7	0.976	0.976	0.976	0.975	0.982	0.964	0.933	0.952
seeds_2	210	7	0.928	0.952	0.927	0.952	0.910	0.946	0.838	0.886
segment	2310	18	0.997	0.991	0.997	0.991	0.998	0.969	0.985	0.948
shuttle	43500	9	0.999	0.999	0.999	0.999	0.998	0.998	0.999	0.999
sonar	208	60	0.761	0.738	0.758	0.732	0.756	0.731	0.694	0.670
spambase	4601	57	0.904	0.669	0.903	0.609	0.894	0.592	0.826	0.481
speeddating	1048	62	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
steel_plates	1941	27	0.915	0.915	0.916	0.917	0.740	0.754	0.296	0.309
student_performance	1000	6	0.83	0.775	0.827	0.780	0.785	0.764	0.845	0.835
sydt	9999889	8	0.979	0.672	0.978	0.704	0.956	0.705	0.979	0.875
toxicity	171	1203	0.542	0.457	0.529	0.449	0.445	0.357	0.297	0.294
twonorm	7400	20	0.826	0.875	0.826	0.874	0.826	0.875	0.765	0.858
vertebral_column	310	6	0.870	0.838	0.869	0.838	0.839	0.815	0.871	0.856
wall_following	5456	24	0.997	0.993	0.997	0.993	0.995	0.993	0.982	0.959
wine_origin	179	13	0.916	0.944	0.917	0.944	0.916	0.937	0.803	0.868
wine	6497	12	0.989	0.308	0.989	0.219	0.986	0.539	0.991	0.772

TABLE VII: Performance of UDTs and MDTs per dataset. Train-test stratified split 90%-10%.