Scalable Optimal Multiway-Split Decision Trees with Constraints

Shivaram Subramanian*, Wei Sun*
IBM Research, Yorktown Heights, New York, USA
{subshiva, sunw}@us.ibm.com

Abstract
There has been a surge of interest in learning optimal decision trees using mixed-integer programs (MIP) in recent years, as heuristic-based methods do not guarantee optimality and find it challenging to incorporate constraints that are critical for many practical applications. However, existing MIP methods that build on an *arc-based* formulation do not scale well as the number of binary variables is in the order of $O(2^dN)$, where $d$ and $N$ refer to the depth of the tree and the size of the dataset. Moreover, they can only handle sample-level constraints and linear metrics. In this paper, we propose a novel *path-based* MIP formulation where the number of decision variables is independent of $N$. We present a scalable column generation algorithm to solve the MIP. Our framework produces a multiway-split tree which is more interpretable than the standard binary-split trees due to its shorter rules. Our method can incorporate nonlinear metrics such as F1-score and a broader class of constraints. We demonstrate its efficacy with extensive experiments. We report superior or competitive results compared to the state-of-art MIP-based methods with up to a 24X reduction in runtime.

1 Introduction
Decision trees are among the most popular machine learning models as the tree structure is visually easy to understand. As learning an optimal decision tree is NP-hard (Laurent and Rivest 1976), popular algorithms such as CART (Breiman et al. 1984), ID3 (Quinlan 1986) and C4.5 (Quinlan 2014) rely on greedy heuristics to construct trees. Motivated by the heuristic nature of the traditional methods, there have been many efforts across different fields to learn optimal decision trees (ODT), e.g., dynamic programming (Lin et al. 2020), constraint programming (Verhaeghe et al. 2020), Boolean satisfiability (Narodytska et al. 2018), itemset mining (Aglin, Nijssen, and Schaus 2020). In particular, recent advances in modern optimization has facilitated a nascent stream of research that leverages mixed-integer programming (MIP) to train globally optimal trees with constraints (Bertsimas and Dunn 2017; Aghaei, Azizi, and Vayanos 2019; Verwer and Zhang 2019; Aghaei, Gómez, and Vayanos 2021) - this is the methodology we are focusing on in this paper.

Prior MIP-based methods rely on an *arc-based* formulation that require a large number of binary decision variables to identify splitting conditions at branch nodes as well as label and sample assignments to leaf nodes. This approach has several drawbacks: (1) the optimization problem becomes easily intractable as the number of binary variables and constraints increases linearly with training data. Hence, experiments are typically restricted to datasets with no more than a few thousand samples. (2) Prior MIP frameworks can only handle linear metrics. In many applications, it is desirable to consider nonlinear metrics, e.g., F1-score is preferred over accuracy to evaluate machine learning models trained on imbalanced datasets. (3) With the arc-based formulation, it is challenging to impose constraints on individual decision rules and feature combinations. One such example occurs in the medical field, where doctors have to arrive at an appropriate diagnosis while taking into account the costs of medical tests (Lomax and Vadera 2013). (4) The vast majority of the decision tree literature focuses on binary-split trees, where each node can have at most two child nodes. Multiway-split trees (see Figure 1 for an example) whose branching condition may contain several values are often more intuitive and comprehensible (Fulton, Kasif, and Salzberg 1995).

In this paper, we attempt to address these prior limitations by proposing a scalable MIP-based framework to train constrained optimal multiway-split trees (OMT). Our contributions are five-fold.
Novel path-based MIP formulation Unlike arc-based MIP, we explicitly model decision rules or paths in a tree. The high level idea is as follows - we first define a feature graph that admits every possible combination of input features as decision rule candidates. The objective of the MIP is to identify a subset of rules from this potentially huge rule space to minimize the prediction error. The feature graph naturally embeds the hierarchical structure of a tree, and we add a constraint to model another defining property of decision trees where each sample can only be assigned to a single rule.

Scalable algorithm Unlike existing MIP ODTs whose binary decision variables are in the order of $O(2^d \times N)$, where $d$ and $N$ refer to the depth of the tree and the size of training data, the number of binary decision variables in our formulation is independent of $N$. It equals the number of candidate decision rules defined in the feature graph. While this number can also be excessive in the worst case, we employ column generation (CG) to dynamically generate a relatively small number of candidates to recover a near-optimal solution. While existing MIP approaches can manage no more than a few thousand training samples, an initial implementation of our proposed CG algorithm effectively solved a $10^6$ sample dataset.

Flexible framework Our framework can be applied to both classification and regression settings. The path-based model allows us to incorporate any path-level metric, including nonlinear ones, which enter the MIP as input parameters. We also show how a broader class of constraints including path-level, and attribute-level constraints, which are difficult to express within existing arc-based formulations, can be easily incorporated into our framework.

Enhanced interpretability with multiway-splits To the best of our knowledge, we are the first to train optimal multiway-split trees for prediction tasks, where a node may have more than two child nodes. Since a feature seldom appears more than once in any root-to-leaf path, multiway-split trees are easier to comprehend than their binary counterparts (Fulton, Kasif, and Salzberg 1995).

Extensive numerical results We conduct extensive computational studies on public datasets and report the first MIP-based results for large datasets containing up to 1,008,372 samples, which are an order of magnitude larger than the prior datasets analyzed in the MIP-based ODT literature. We show that our framework is either competitive or improves upon the state-of-the-art MIP methods in terms of out-of-sample accuracy and achieves up to a 24-fold reduction in runtime.

3 Problem Formulation

We consider a dataset which consists of $N$ samples, $\{(x_i, y_i)\}_{i=1}^N$, where $x_i \in \mathcal{X}^k$ are features which are assumed to be categorical. We will discuss how to handle numerical features in Section 5, and present experiments on datasets with both categorical and numerical features in Section 6. $y_i$ is the outcome, which can be a discrete label for classification or a continuous quantity for regression.

A binary-split tree of depth $d$ can have at most $2^d$ leaf nodes. In a multiway-split tree, each node may have more than two children. Thus, we use the depth of a tree $d$, and the number of leaf nodes $l$, which are user-specified parameters to define the complexity of a tree. Figure 1 shows a multiway-split tree with $d = 3$ and $l = 8$.

In our framework, we explicitly model individual decision paths from the root to the leaf nodes. We begin by defining a feature graph that contains every possible combination of input features, followed by introducing a MIP optimization problem to identify a subset of paths to form the tree.
The search space \( P \) may become impractical for high-dimensional datasets.

**Proposition 1** The search space \( P \) is \( O(n^p) \).

Later in Section 4, we describe our algorithm which uses a large-scale optimization technique to intelligently search through the space, and only paths that can improve the current solution are generated on the fly.

**Path-Based MIP Formulation**

We denote a decision rule as \( j \in \{1, \cdots, L\} \), where \( L = |P| \). Let \( S_j \in [N] \) be the subset of observations which fall into rule \( j \). Denote \( \xi_j \) as a user-specified metric associated with rule \( j \), e.g., misclassification error in rule \( j \) in a classification task, squared loss or absolute loss for regression.

Let \( z_j \) be a binary decision variable which indicates whether rule \( j \) is selected (1) or not (0), for \( j = 1, \cdots, L \). We define non-negative slack variables \( s_i \) and a positive penalty cost \( c_i \) for each sample \( i \). A MIP formulation to determine an optimal multiway-split tree (OMT) can be written as follows,

\[
\min \sum_{j=1}^{L} \xi_j z_j + \sum_{i=1}^{N} c_i s_i
\]

s.t. \[
\sum_{j=1}^{L} a_{ij} z_j + s_i = 1, \quad \forall i = 1, \cdots, N \tag{1}
\]

\[
\sum_{j=1}^{L} z_j \leq L
\]

\[
z_j \in \{1, 0\}, \quad \forall j = 1, \cdots, L
\]

\[
s_i \geq 0, \quad \forall i = 1, \cdots, N
\]

The input parameter \( a_{ij} = 1 \) if sample \( i \) satisfies the conditions specified in rule \( j \), and 0 otherwise. While several rules may contain the same data sample, the set partitioning constraint (1) with a sufficiently large penalty \( c_i \) ensures that each sample is ultimately assigned to exactly one rule. The cardinality constraint (2) stipulates that no more than \( L \) rules are active in the optimal solution \( z^* \).

It is well-known that set partitioning problems are \( NP \)-hard (Hartmanns 1982). Thus, while optimal solutions can be obtained in practice for moderate sized instances (Atamtürk, Nemhauser, and Savelsbergh 1996), the problem may easily become intractable to solve directly as the cardinality of feasible rules grows exponentially with the feature space. We will present an efficient algorithm to overcome this computational challenge in the next section.

### 4 Column Generation

Column generation (CG) is a classical technique to solve large MIPs for which it is not practical to explicitly generate all columns (variables) of the problem (Lübbecke and Desrosiers 2005). Specifically, we first consider a restricted master program (RMP) version of OMT, where we 1) consider only a subset of paths \( \hat{L} \), which is typically much smaller than \( L \) and is determined dynamically, and 2) relax the integrality constraints on \( z_j \) to \( 0 \leq z_j \leq 1 \) for all \( j = 1, \cdots, \hat{L} \).

Denote the dual variables associated with the set partitioning constraints in (1) and the cardinality constraint in (2) as \( \lambda_i \) and \( \mu \) respectively. The dual formulation of RMP can be written as follows,

\[
\max \sum_{i=1}^{N} \lambda_i + n\mu
\]

s.t. \[
\sum_{i=1}^{N} a_{ij} \lambda_i + \mu \leq \xi_j, \quad \forall j = 1, \cdots, \hat{L} \tag{3}
\]

\[
\mu \leq 0, \quad \lambda_i \leq c_i, \quad \forall i = 1, \cdots, N
\]

Note that since \( z_j \leq 1 \) is implied by Eq (1), we are only left with \( z_j \geq 0 \) in the primal RMP.

Based on the dual feasibility constraint for path \( j \) in (3), the reduced cost for path \( j \) is equal to

\[
rc_j = \xi_j - \left( \sum_{i=1}^{N} a_{ij} \lambda_i + \mu \right). \tag{4}
\]
When RMP is solved to optimality, dual feasibility is guaranteed only for the rules included in \( \hat{L} \). A path violating the dual constraint (3) has a negative reduced cost and must be added to the RMP for the next iteration. To identify paths that maximally violate Eq (3), we need to solve \( \min r_c \), where \( r_c \) gives the subproblem, as the search for some \( K \) paths having the most negative reduced cost, which is a \( K \)-shortest path problem (KSP) over the feature graph \( G \), where the cost on a path is its reduced cost defined in (4). If no violating path exists, we obtain dual feasibility, and the CG has converged.

The high-level view of the CG procedure is as follows: we start by solving RMP with an empty set of candidate rules with \( L = \emptyset \), which sets slack variables \( s_l \) to 1.0 and yields an initial dual solution. Next, we use these dual values to solve the subproblem to identify up to \( K \) candidate rules having the most negative reduced cost. They are added to \( L \) and the RMP is re-optimized to obtain a new primal-dual solution, and the process iterates. The CG procedure converges to an optimal solution of the LP relaxation of OMT when dual feasibility is achieved, i.e., \( r_c \geq 0 \) for all \( j = 1, \ldots, M \). After the CG procedure has converged, the binary restrictions are reimposed on \( z \) and we solve the resultant Master-MIP.

The CG procedure can be implemented exactly for provable optimality, or approximately to obtain a near-optimal decision tree. Prior works (Barnhart et al. 1998) have proposed a branch-and-price technique to solve the Master-MIP to provable optimality, which also requires the subproblem to be solved exactly (Desrosiers and Lübbecke 2005).

For simplicity and replicability, we employ a CG heuristic that solves the resultant Master-MIP directly using a standard optimization package (CPLEX 2021). As noted in Lübbecke and Desrosiers 2005, the role of the subproblem is to provide a potential column or to prove that none exists. Since any negative reduced cost path contributes to this aim, we employ a heuristic subproblem algorithm that is a path-dependent adaptation of a \( K \)-shortest path method for acyclic graphs with additive arc costs (Horne 1980). Our KSP method is similar to the resource-constrained shortest path heuristics used in CG applications (Desrosiers and Lübbecke 2005; Desaulniers, Desrosiers, and Solomon 2006) and has polynomial-time complexity (additional details are given in the appendix). In our experiments, the CG procedure terminates when the dual solution converges to within a tolerance or we reach a maximum iteration limit.

To preserve interpretability, shallow trees are often preferred, where the depth \( d \) is small. The following result shows how this constraint dramatically shrinks the CG search space, especially for high-dimensional datasets.

**Proposition 2** The constrained search space is \( O((\xi_\text{K})^d) \).

Comparing this result to the unconstrained setting shown in Proposition 1, we see that restricting each path to contain no more than \( d \) features significantly reduces the total number of feasible paths, and the worst-case value for \( L \ll L \). Considering an example with \( d = \eta = 2 \), \( L \) is bounded by \( O(k^d) \) versus \( L \) is \( O(2^d) \). Prior MIP models are restricted to a fixed-depth binary tree representation which requires categorical attributes to be encoded into a generic set of binary features. Doing so prevents them from exploiting the beneficial hierarchical feature graph structure used in our approach.

**5 Flexible Framework**

**Nonlinear Metrics**

Nonlinear metrics such as the F1-score, Matthews correlation coefficient, and Fowlkes–Mallows index, are often used to evaluate the performance of machine learning models trained on imbalanced data (Demirovic and Stuckey 2021). Existing MIP-based decision trees only consider linear metrics. In our approach, once the feature graph is defined, we know the samples \( S_j \) which satisfy this rule and we can compute the metric associated with this rule in the subproblem. These nonlinear metrics may be represented as \( \xi_t \) and enter RMP as an input parameter in the objective as shown in Section 3. They can also enter RMP as constraints and we provide an example of incorporating F1-score in the appendix. This transformative modeling capability of CG is invaluable when solving optimization problems that are highly nonlinear and nonconvex in their original form.

**Constraints Enforcement**

Existing optimal classification trees in the literature have been extended to incorporate constraints to address fairness and data imbalance issues. In this section, we show how our method provides an elegant and unified framework to handle constraints, including those which existing arc-based MIP formulations cannot efficiently manage.

**Path-level constraints** Prior MIP-based decision trees typically manage \( \text{sample-level} \) constraints, e.g., constraining precision or recall conditioned on samples’ class labels (Aghaei, Gómez, and Vayanos 2021; Günlük et al. 2021), or fairness metrics such as statistical parity conditioned on sensitive features (Aghaei, Azizi, and Vayanos 2019; Aghaei, Gómez, and Vayanos 2021). However, none of the existing MIP-based methods are able to efficiently handle constraints at \( \text{path-level} \), as the notion of “path” is not explicitly defined in the arc-based formulation.

Consider the example of cost-sensitive decision trees, which is motivated by the medical domain. Often, doctors must arrive at a diagnosis by taking into account the economic constraints faced by a patient when different test options involve a tradeoff between accuracy and measurement cost (Lomax and Vadera 2013; Núñez 1991). Denoting the cost associated with each decision rule \( j \) which consists of several medical tests as \( \rho_j \) and the budget as \( C \), we specify the following constraint: \( \rho_j z_j \leq C \) for all \( j = 1, \ldots, L \). This constraint which ensures that each of the final selected diagnostic methods (with \( z_j = 1 \)) is staying within the budget, can be processed within the KSP subproblem. Additional examples can be found in the appendix.

In general, \( \text{path-level} \) constraints can be expressed as a set of polyhedral inequalities on \( z \), i.e., \( \sum_{j=1}^L \rho_{m_j} z_j \geq q_m, \forall m = 1, \ldots, M \). Let \( r_m \geq 0 \) denote the dual variables corresponding to these constraints. Then, the reduced cost
The constraints influence the subproblem through the resultant reduced costs to ensure that paths that are more likely to be feasible are added to the RMP.

**Attribute-level constraints** Existing MIP methods are also challenged by *attribute-level* constraints that represent complex nonlinear conditions involving several features that can not be abstracted efficiently into linear constraints, e.g., disallowing certain feature combinations. These constraints are easily handled within the KSP subproblem as a feasibility check while extending a partial path to the next node in $G$. A practical example of an attribute-level requirement is the need to preserve attribute hierarchy. For example, in the medical domain, doctors may perform temperature checks or blood tests before proceeding to more advanced tests. Ensuring such hierarchy makes the decision tree more reliable and comprehensible for domain experts (Nanfack, Temple, and Fréay 2022). This can be easily achieved by appropriately arranging the nodes in the feature graph.

**Cumulative Binning on Numerical Features**

For tree-based approaches, numerical input are typically handled via thresholding. For example, consider a numerical feature with values in $[0, 1]$ being divided into 3 intervals, e.g., $[0, 0.33)$, $[0.33, 0.67)$ and $[0.67, 1.0]$. One can transform this numerical feature to a categorical feature with 3 values, and create 3 nodes representing this feature in the graph. This approach may be limiting as a binary-split tree can branch on conditions such as $x \leq 0.67$ or $x > 0.3$. To address this limitation, we employ *cumulative binning*, where intervals can be overlapping. We create additional nodes representing intervals $[0, 0.67)$, $[0.33, 1.0)$, yielding a total of 5 nodes. The coverage constraints (1) ensure that the final rule set does not contain overlapping samples. In our experiments, quantile discretization is used to create intervals. More generally, with $\kappa$ intervals, cumulative binning results in $O(\kappa^2)$ nodes, i.e., a gain in a rule’s expressiveness at the cost of higher computational effort.

### 6 Experiments

While OMT is a general framework which can also be used in regression settings, we focus on classification tasks in our experiments as most of the existing ODTs are classification trees. We group our experiments by dataset size, where small/medium datasets are loosely defined as having a few thousand samples, and a dataset is considered large if it has many thousands of samples and several hundred binary features (Dash, Gunluk, and Wei 2018).

To benchmark our approach OMT, we implement the following MIP-based methods, i.e., FlowOCT (Aghaei, Gómez, and Vayanos 2021), BinOCT (Verwer and Zhang 2019) and OCT (Bertsimas and Dunn 2017). Although CART cannot produce constrained decision rules, we still include it as a baseline. As the benchmarks produce binary-split trees of a given depth $d$, we construct a comparable multiway-split tree via the cardinality constraint that restricts the number of leaf nodes $l$ to be at most $2^d$, i.e., $l = 2^d$, and limit the rule length to $d$.

**Small/Medium Datasets**

We evaluate the same 12 classification datasets from the UCI repository (Dua and Graff 2017) that have been used in FlowOCT (Aghaei, Gómez, and Vayanos 2021), which is considered the state-of-the-art ODT. We closely follow the experiment setup in Aghaei, Gómez, and Vayanos 2021 to construct decision trees with $d \in \{2, 3, 4, 5\}$. We create 5 random splits for each dataset into training (50%), validation (25%), and test sets (25%). A time limit of 20 minutes is imposed on each experiment, in contrast to the one hour limit used in Aghaei, Gómez, and Vayanos 2021.

Table 1 reports the out-of-sample accuracy averaged over five splits for $d \in \{4, 5\}$ (the complete results across all depths can be found in the appendix). Best accuracy in a given row is reported in **bold**. Among 48 (data, depth) combinations, OMT dominates other methods in 56.3% of them, FlowOCT 33.3%, OCT 29.2%, BinOCT 8.3%, and CART 14.6% (including ties). These results demonstrate that our OMT method achieves competitive and often superior results compared to MIP-based binary-split ODT models.

Next, we analyze the solution time taken by different MIP-based methods. We define an “instance” as a unique (dataset, depth, data split) combination. With 5 random data splits, there are $12 \times 4 \times 5 = 240$ instances in total. In Figure 3, the x-axis shows the solution time in seconds, while the y-axis shows the number of instances solved. With the MIP-based ODT benchmarks, when the solution time is under the time limit, implying the instance is solved to optimality or near optimality. Figure 3 shows that OMT solves most instances before the 20 minutes time limit, followed by FlowOCT, BinOCT and OCT. Specifically, 188 out of 240 instances was solved under 1 minute by OMT, in contrast to

![Figure 3: Runtime performance on a total of 240 instances](image-url)
which grows by another 4.5X when d
mark, FlowOCT ignoring constraints considered in this work.
deriving in one instance, and no wins for
stances at d = 3. In contrast, our method on the six largest datasets analyzed in the
2020, we focus on one hour to account for the larger data size. As in Zhu et al.
and 5 is 10.3X and 24.6X respectively. In contrast, the runtime increases are merely 1.4X, 0.68X, 0.4X as d increases from 2 to 5. Overall, our speedup over FlowOCT at d = 4 and 5 is 10.3X and 24.6X respectively. The exact OMT speedup is likely to be even higher if we compare runtime to achieve the same solution quality.

The strong CG performance can also be explained by examining the gap between the objective value achieved by the Master-MIP (ν_{MIP}) and the final RMP (ν_{RMP}), i.e., Δ = (ν_{MIP} − ν_{RMP})/ν_{MIP}. A table that reports on this MIP-LP gap with respect to d is included in the appendix. In particular, the median gap is no more than 0.3% across all depths, suggesting that the path-based LP relaxation is a relatively strong approximation of the nonconvex discrete OMT model. This tight gap also underscores the advantage of converging to a relatively small subset of high quality paths from which an effective decision tree can be distilled using a standard MIP solver.

### Large Datasets

We test our method on the six largest datasets analyzed in the MIP-based ODT literature (Zhu et al. 2020). We follow the same experiment setup described earlier but limit runtime to one hour to account for the larger data size. As in Zhu et al. 2020, we focus on d = {2, 3}. Table 3 summarizes the average out-of-sample accuracy of different MIP approaches. The left entries under column “CART” shows the CART results that we obtain. None of the arc-based MIP methods are able to obtain an optimal solution on any of the 12 large instances at d = 3 within the time limit, with FlowOCT winning in one instance, and no wins for OCT and BinOCT. On the other hand, OMT dominates 6 out of 12, and CART

<table>
<thead>
<tr>
<th>dataset</th>
<th>N</th>
<th>d</th>
<th>OMT</th>
<th>OCT</th>
<th>BinOCT</th>
<th>FlowOCT</th>
<th>CART</th>
</tr>
</thead>
<tbody>
<tr>
<td>soybean-small</td>
<td>47</td>
<td>4</td>
<td>0.88±0.139</td>
<td>0.94±0.048</td>
<td>0.83±0.22</td>
<td>0.94±0.083</td>
<td>1.0±0.0</td>
</tr>
<tr>
<td>soybean-small</td>
<td>47</td>
<td>5</td>
<td>0.95±0.075</td>
<td>0.97±0.096</td>
<td>0.72±0.21</td>
<td>0.97±0.083</td>
<td>1.0±0.0</td>
</tr>
<tr>
<td>monks-3</td>
<td>122</td>
<td>4</td>
<td>0.98±0.008</td>
<td>0.99±0.015</td>
<td>0.98±0.011</td>
<td>0.99±0.011</td>
<td>0.99±0.007</td>
</tr>
<tr>
<td>monks-3</td>
<td>122</td>
<td>5</td>
<td>0.98±0.008</td>
<td>0.978±0.014</td>
<td>0.983±0.015</td>
<td>0.99±0.012</td>
<td>0.99±0.007</td>
</tr>
<tr>
<td>monks-1</td>
<td>124</td>
<td>4</td>
<td>1.0±0.0</td>
<td>1.0±0.029</td>
<td>1.0±0.0</td>
<td>1.0±0.03</td>
<td>1.0±0.03</td>
</tr>
<tr>
<td>monks-1</td>
<td>124</td>
<td>5</td>
<td>1.0±0.0</td>
<td>0.9±0.14</td>
<td>1.0±0.0</td>
<td>1.0±0.03</td>
<td>0.8±0.064</td>
</tr>
<tr>
<td>hayes-roth</td>
<td>132</td>
<td>4</td>
<td>0.75±0.075</td>
<td>0.75±0.038</td>
<td>0.64±0.138</td>
<td>0.8±0.066</td>
<td>0.55±0.087</td>
</tr>
<tr>
<td>hayes-roth</td>
<td>132</td>
<td>5</td>
<td>0.77±0.089</td>
<td>0.75±0.076</td>
<td>0.575±0.066</td>
<td>0.817±0.029</td>
<td>0.78±0.058</td>
</tr>
<tr>
<td>monks-2</td>
<td>169</td>
<td>4</td>
<td>0.63±0.045</td>
<td>0.66±0.031</td>
<td>0.581±0.027</td>
<td>0.662±0.023</td>
<td>0.598±0.033</td>
</tr>
<tr>
<td>monks-2</td>
<td>169</td>
<td>5</td>
<td>0.779±0.03</td>
<td>0.662±0.05</td>
<td>0.607±0.03</td>
<td>0.662±0.055</td>
<td>0.651±0.055</td>
</tr>
<tr>
<td>house-votes-84</td>
<td>232</td>
<td>4</td>
<td>0.96±0.046</td>
<td>0.97±0.01</td>
<td>0.914±0.03</td>
<td>0.971±0.01</td>
<td>0.96±0.02</td>
</tr>
<tr>
<td>house-votes-84</td>
<td>232</td>
<td>5</td>
<td>0.95±0.026</td>
<td>0.97±0.01</td>
<td>0.96±0.01</td>
<td>0.971±0.01</td>
<td>0.96±0.02</td>
</tr>
<tr>
<td>spect</td>
<td>267</td>
<td>4</td>
<td>0.83±0.04</td>
<td>0.801±0.086</td>
<td>0.746±0.065</td>
<td>0.791±0.079</td>
<td>0.731±0.026</td>
</tr>
<tr>
<td>spect</td>
<td>267</td>
<td>5</td>
<td>0.80±0.039</td>
<td>0.791±0.06</td>
<td>0.721±0.048</td>
<td>0.796±0.085</td>
<td>0.731±0.03</td>
</tr>
<tr>
<td>breast-cancer</td>
<td>277</td>
<td>4</td>
<td>0.74±0.046</td>
<td>0.724±0.038</td>
<td>0.662±0.091</td>
<td>0.743±0.079</td>
<td>0.69±0.058</td>
</tr>
<tr>
<td>breast-cancer</td>
<td>277</td>
<td>5</td>
<td>0.669±0.094</td>
<td>0.738±0.0</td>
<td>0.567±0.157</td>
<td>0.676±0.036</td>
<td>0.671±0.049</td>
</tr>
<tr>
<td>balance-scale</td>
<td>625</td>
<td>4</td>
<td>0.78±0.016</td>
<td>0.747±0.048</td>
<td>0.707±0.006</td>
<td>0.699±0.01</td>
<td>0.769±0.033</td>
</tr>
<tr>
<td>balance-scale</td>
<td>625</td>
<td>5</td>
<td>0.77±0.012</td>
<td>0.735±0.01</td>
<td>0.565±0.1</td>
<td>0.72±0.045</td>
<td>0.762±0.013</td>
</tr>
<tr>
<td>tic-tac-toe</td>
<td>958</td>
<td>4</td>
<td>0.80±0.039</td>
<td>0.776±0.073</td>
<td>0.78±0.021</td>
<td>0.757±0.032</td>
<td>0.758±0.019</td>
</tr>
<tr>
<td>tic-tac-toe</td>
<td>958</td>
<td>5</td>
<td>0.82±0.061</td>
<td>0.71±0.025</td>
<td>0.812±0.029</td>
<td>0.788±0.05</td>
<td>0.778±0.046</td>
</tr>
<tr>
<td>car-evaluation</td>
<td>1728</td>
<td>4</td>
<td>0.879±0.009</td>
<td>0.796±0.076</td>
<td>0.848±0.01</td>
<td>0.823±0.016</td>
<td>0.842±0.029</td>
</tr>
<tr>
<td>car-evaluation</td>
<td>1728</td>
<td>5</td>
<td>0.91±0.005</td>
<td>0.742±0.041</td>
<td>0.815±0.052</td>
<td>0.8±0.016</td>
<td>0.857±0.019</td>
</tr>
<tr>
<td>kr-vs-kp</td>
<td>3196</td>
<td>4</td>
<td>0.96±0.006</td>
<td>0.847±0.094</td>
<td>0.938±0.012</td>
<td>0.94±0.011</td>
<td>0.94±0.011</td>
</tr>
<tr>
<td>kr-vs-kp</td>
<td>3196</td>
<td>5</td>
<td>0.96±0.003</td>
<td>0.652±0.098</td>
<td>0.847±0.164</td>
<td>0.94±0.057</td>
<td>0.94±0.011</td>
</tr>
</tbody>
</table>

Table 1: Mean ± standard deviation of out of sample accuracy on the small/medium datasets.

<table>
<thead>
<tr>
<th>d</th>
<th>OMT</th>
<th>FlowOCT</th>
<th>BinOCT</th>
<th>OCT</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3.9</td>
<td>1.9</td>
<td>6.1</td>
<td>33.9</td>
</tr>
<tr>
<td>3</td>
<td>9.3</td>
<td>32.5</td>
<td>727.8</td>
<td>1200.4</td>
</tr>
<tr>
<td>4</td>
<td>15.6</td>
<td>177.0</td>
<td>1201.9</td>
<td>1202.5</td>
</tr>
<tr>
<td>5</td>
<td>21.9</td>
<td>561.4</td>
<td>1203.5</td>
<td>1205.9</td>
</tr>
</tbody>
</table>

Table 2: Median runtime (seconds) for MIP-based methods.
Table 3: Out-of-sample accuracy, using the large datasets from (Zhu et al. 2020).

<table>
<thead>
<tr>
<th>dataset</th>
<th>N</th>
<th>k</th>
<th>d</th>
<th>OMT</th>
<th>OCT</th>
<th>BinOCT</th>
<th>FlowOCT</th>
<th>CART</th>
<th>S1O</th>
</tr>
</thead>
<tbody>
<tr>
<td>pendigits</td>
<td>7494</td>
<td>16</td>
<td>2</td>
<td>0.395*</td>
<td>0.348</td>
<td>0.291</td>
<td>0.329</td>
<td>0.352/0.362</td>
<td>0.389</td>
</tr>
<tr>
<td>pendigits</td>
<td>7494</td>
<td>16</td>
<td>3</td>
<td>0.685*</td>
<td>0.517</td>
<td>0.347</td>
<td>0.459</td>
<td>0.558/0.579</td>
<td>0.625</td>
</tr>
<tr>
<td>avila</td>
<td>10430</td>
<td>3</td>
<td>2</td>
<td>0.475</td>
<td>0.461</td>
<td>0.096</td>
<td>0.509</td>
<td>0.501/0.503</td>
<td>0.526*</td>
</tr>
<tr>
<td>avila</td>
<td>10430</td>
<td>3</td>
<td>3</td>
<td>0.524</td>
<td>0.417</td>
<td>0.403</td>
<td>0.504</td>
<td>0.527/0.535</td>
<td>0.558*</td>
</tr>
<tr>
<td>EEG</td>
<td>14980</td>
<td>14</td>
<td>2</td>
<td>0.658</td>
<td>0.602</td>
<td>0.584</td>
<td>0.648</td>
<td>0.624/0.586</td>
<td>0.665*</td>
</tr>
<tr>
<td>EEG</td>
<td>14980</td>
<td>14</td>
<td>3</td>
<td>0.690*</td>
<td>0.572</td>
<td>0.493</td>
<td>0.649</td>
<td>0.659/0.642</td>
<td>0.665</td>
</tr>
<tr>
<td>HTRU</td>
<td>17898</td>
<td>8</td>
<td>2</td>
<td>0.973</td>
<td>0.977</td>
<td>0.705</td>
<td>0.956</td>
<td>0.978/0.973</td>
<td>0.978*</td>
</tr>
<tr>
<td>HTRU</td>
<td>17898</td>
<td>8</td>
<td>3</td>
<td>0.977</td>
<td>0.978</td>
<td>0.552</td>
<td>0.973</td>
<td>0.979/0.981*</td>
<td>0.979</td>
</tr>
<tr>
<td>shuttle</td>
<td>43500</td>
<td>9</td>
<td>2</td>
<td>0.968*</td>
<td>0.821</td>
<td>0.285</td>
<td>0.920</td>
<td>0.939/0.938</td>
<td>0.940</td>
</tr>
<tr>
<td>shuttle</td>
<td>43500</td>
<td>9</td>
<td>3</td>
<td>0.984</td>
<td>0.793</td>
<td>0.390</td>
<td>0.914</td>
<td>0.996/0.997</td>
<td>0.995*</td>
</tr>
<tr>
<td>skin</td>
<td>245057</td>
<td>3</td>
<td>2</td>
<td>0.875*</td>
<td>0.899</td>
<td>0.774</td>
<td>0.802</td>
<td>0.907/0.806</td>
<td>0.863</td>
</tr>
<tr>
<td>skin</td>
<td>245057</td>
<td>3</td>
<td>3</td>
<td>0.967*</td>
<td>0.793</td>
<td>0.855</td>
<td>0.801</td>
<td>0.965/0.871</td>
<td>0.949</td>
</tr>
</tbody>
</table>

Table 4: Out-of-sample accuracy on additional large datasets

<table>
<thead>
<tr>
<th>dataset</th>
<th>N</th>
<th>k</th>
<th>d</th>
<th>OMT</th>
<th>CART</th>
</tr>
</thead>
<tbody>
<tr>
<td>MiniBooNE</td>
<td>130065</td>
<td>50</td>
<td>2</td>
<td>0.850</td>
<td>0.835</td>
</tr>
<tr>
<td>MiniBooNE</td>
<td>130065</td>
<td>50</td>
<td>3</td>
<td>0.873</td>
<td>0.865</td>
</tr>
<tr>
<td>crop mapping</td>
<td>325834</td>
<td>175</td>
<td>2</td>
<td>0.769</td>
<td>0.679</td>
</tr>
<tr>
<td>crop mapping</td>
<td>325834</td>
<td>175</td>
<td>3</td>
<td>0.841</td>
<td>0.797</td>
</tr>
<tr>
<td>covertype</td>
<td>581012</td>
<td>54</td>
<td>2</td>
<td>0.667</td>
<td>0.668</td>
</tr>
<tr>
<td>covertype</td>
<td>581012</td>
<td>54</td>
<td>3</td>
<td>0.684</td>
<td>0.677</td>
</tr>
<tr>
<td>susy</td>
<td>1008372</td>
<td>18</td>
<td>2</td>
<td>0.744</td>
<td>0.748</td>
</tr>
<tr>
<td>susy</td>
<td>1008372</td>
<td>18</td>
<td>3</td>
<td>0.760</td>
<td>0.755</td>
</tr>
</tbody>
</table>

In this paper, we propose a scalable and flexible mixed-integer optimization framework based on column generation (CG) to identify optimal multiway-split decision trees with constraints. We present a novel path-based MIP formulation for decision trees where the number of columns or binary variables is independent of the training data size. Our method can generate both classification and regression trees by minimizing any of the commonly used nonlinear error metrics while still solving a linear MIP model. A first “plain vanilla” CG implementation is tested on several public datasets ranging up to a million data samples. We are able to achieve a performance that is comparable to or superior than those achieved by the state-of-art optimal classification tree methods in the literature while consuming only a small fraction of their runtime. Our CG approach is also able to seamlessly handle a wide variety of practical constraints that cannot be efficiently managed by prior ODT models or popular heuristics like CART.

The current framework has to discretize numerical features. One way to handle numerical features without discretization is to employ a preprocessing step like BinOCT [Verwer & Zhang 2019], which we leave as future work. For deeper trees (e.g., d ≥ 10), the computational benefit of Proposition-2 diminishes and a more sophisticated CG implementation may be required.
References


