Learning Optimal Classification Trees Using a Binary Linear Program Formulation (Extended Abstract) *

Sicco Verwer[1] and Yingqian Zhang[2]

[1] Delft University of Technology
[2] Eindhoven University of Technology

Introduction. Decision trees have gained increasing popularity these years due to their effectiveness in solving classification and regression problems and their capability to explain prediction results. Learning an optimal decision tree with a predefined depth is NP-hard. Hence, greedy based heuristics such as CART and ID3 have been widely used to construct sub-optimal trees. Recent years have seen an increasing number of work that employ various Mathematical Optimization (MO) methods to build better quality decision trees. A limitation of the state-of-the-art Mathematical Optimization formulations, i.e., [1, 2], for this problem is that they create constraints and variables for every row in the training data. Consequently, the solving time of finding decision trees increases dramatically with the problem size. We formulate the problem of constructing the optimal decision tree of a given depth as an *binary linear program.* We call our method BinOCT, a Binary encoding for constructing Optimal Classification Trees. Our novel formulation models the selection of decision threshold via a binary search procedure encoded using a type of big-M constraints. This requires a very small number of binary decision variables and is therefore able to find good quality solutions within limited time. Noteworthy is that the number of decision variables is largely independent of the number of training data rows: it only depends logarithmically on the number of unique feature values. We show using experiments that BinOCT outperforms existing MO based approaches on a variety of data sets in terms of accuracy and computation time.

BinOCT model. In contrast to earlier formulations that use continuous or integer decision thresholds for each internal node, we represent decision thresholds using only binary variables. When the feature used in the decision is binary, the formulation is intuitive, as explained below. Assume we are learning a tree consisting of a single decision node. Let $l_{r,1}$ and $l_{r,2}$ be binaries indicating that data row r reaches leaf 1 in the left and leaf 2 in the right branch from the root node. A row has to reach a single leaf:

$$l_{r,1} + l_{r,2} = 1.$$

* The full paper was published in [3]

Let t_n be a binary variable for the binary decision threshold for node n, that is, depending on the value of t_n, a data row goes to the left or right branch of node n. Hence, leaf 1 is reached by row r when t_n is 0. Leaf 2 is reached by row r when t_n is 1. This can be encoded by adding the following two constraints

$$l_{r,1} + t_n \leq 1 \text{ and } l_{r,2} - t_n \leq 0.$$

These constraints force $l_{r,1}$ to be 0 when t_n equals 1 and $l_{r,2}$ to 0 when t_n equals 0. The first constraint then guarantees that the leaf not forced to 0 is reached by row r. A naive formulation based on this intuition requires at least one such constraint for every row in the training data. We significantly reduce this number by the observation that we can simply sum the above constraints of threshold checking for all data rows r with feature value v_r equal to 1, i.e.,

$$\sum_{r:v_r=1} l_{r,1} + M \cdot t_n \leq M \text{ and } \sum_{r:v_r=1} l_{r,2} - M \cdot t_n \leq 0,$$

where $M = \sum_{r:v_r=1} 1$. Like before, $l_{r,1} = 0$ when $t_n = 1$, and $l_{r,2} = 0$ when $t_n = 0$, only now this is forced at once for all rows in the training data. In addition to summing over all rows at once, we use a novel binary encoding of the decision tree branching thresholds, see [3] for details. Overall, our encoding requires $O(2^K(F+C+log(T_{max})))$ decision variables and $O(R+2^K(F \cdot T_{all}+C))$ constraints, where K is the tree depth, F is the number of features, C is the number of classes, R is the number of data rows, T_{max} is the maximum number of possible decision thresholds for any feature, and T_{all} is the total number of decision thresholds over all features. Most importantly, the first term does not depend on R, while the second term depends only linearly on R.

Experiments We formulate the learning problem using the proposed formulation. The resulting BinOCT model is passed to the optimization solver CPLEX 12.8.0, which returns the best solution (i.e., a classification tree with highest accuracy) it can find within 10 minutes. We tested our method on 16 datasets. The number of data rows in these datasets range from 124 to 4601, and the number of features from 4 to 57. For depths 2 and 3, BinOCT clearly outperforms both the classical CART method and the previously proposed OCT encoding [1]. For depth 4, BinOCT still outperforms them, but only slightly. Overfitting seems to be a problem for BinOCT, which we will address in future work. The full experimental results are available in [3], and the code and datasets used can be found online at https://github.com/SiccoVerwer/binoct.

References

1. Bertsimas, D., Dunn, J.: Optimal classification trees. Machine Learning **106**(7), 1039–1082 (2017)
2. Verwer, S., Zhang, Y.: Learning decision trees with flexible constraints and objectives using integer optimization. In: CPAIOR. pp. 94–103. Springer (2017)
3. Verwer, S., Zhang, Y.: Learning optimal classification trees using a binary linear program formulation. In: AAAI. pp. 1625–1632 (2019)