Optimal Robust Decision trees

Research question

What are the advantages of an adapted version of the Murtree algorithm when applied to computing optimal robust decision trees, compared to state-of-the-art solutions?

Background

- Optimal trees can be found with dynamic programming. This approach is more feasible than other existing solutions. [1]
- No dynamic programming approach has been applied to robust decision trees
- Alternative solutions do not scale beyond depth two

References

[1] Emir Demirovi´c, Anna Lukina, Emmanuel Hebrard, Jeffrey Chan, James Bailey, Christopher Leckie, Kotagiri Ramamohanarao, and Peter J. Stuckey. Murtree: Optimal decision trees via dynamic programming and search. J. Mach. Learn. Res., 23(1), jan 2022.

[2] Dani¨el Vos and Sicco Verwer. Efficient training of robust decision trees against adversarial examples. In Marina Meila and Tong Zhang, editors, Proceedings of the 38th International Conference on Machine Learning, volume 139 of Proceedings of Machine Learning Research, pages 10586–10595. PMLR, 18–24 Jul 2021.

[3] Dani¨el Vos and Sicco Verwer. Robust optimal classification trees against adversarial examples. Proceedings of the AAAI Conference on Artificial Intelligence, 36(8):8520–8528, Jun. 2022.

[4] Jacobus GM van der Linden, Mathijs M de Weerdt, and Emir Demirovic. Optimal decision trees for separable ´ objectives: Pushing the limits of dynamic programming. arXiv e-prints, pages arXiv–2305, 2023.

Method

Dynamic programming approach

Optimal decision tree

$$C(D,d) = \begin{cases} \min(D^+, D^-), & \text{if } d = 0. \\ \min_{f \in F}(C(D_f, d-1) + C(D_{\bar{f}}, d-1)) & \text{if } d > 0. \end{cases}$$

Figure 1

Recurrence relation for DP approach to solving a binary decision tree.
D: set of instances
D+/-: set of instances labeled true/false
D_f/-f: set of instances where feature is true/false
d: depth
F: set of features
C(D,d): minimum cost

Optimal Robust decision tree

$$C(D,F,d) = \begin{cases} \{(F, \emptyset, \emptyset, min(D^+, D^-) + min(D_a^+, D_a^-), min(D^+, D^-) + |D_a|\}, & \text{if } d = 0, \\ \text{candidates}\left(\bigcup_{f \in \mathcal{F}} \text{merge}\left(C(D_f, f \cup F, d-1), C(D_{\bar{f}}, \bar{f} \cup F, d-1)\right)\right), & \text{if } d > 0. \end{cases}$$

Figure 2

Recurrence relation for DP approach to solving a robust binary decision tree.
candidates: returns possible subtrees that might be part of the optimal solution
merge: combines the sets of subtrees for the feature split into one set of subtrees

Existing Alternatives

ROCT
An approach that relies on combinatorial optimization approaches such as mixed integer programming or maximum satisfiability
[3]

Brute-force
Simply trying out all possible valid tree combinations

GROOT
An alternative that can find robust decision trees but does not guarrantee optimality
[2]

Modelling adversary

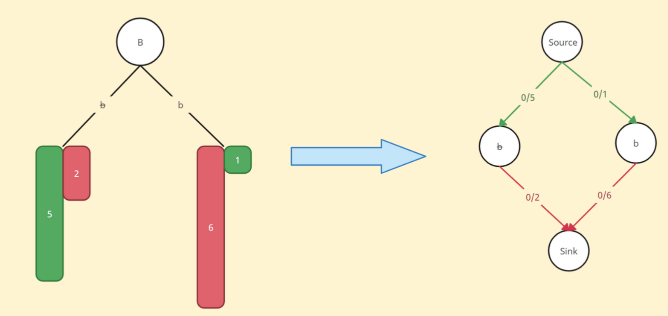

Figure 3

An decision tree can be modelled as a network flow problem.

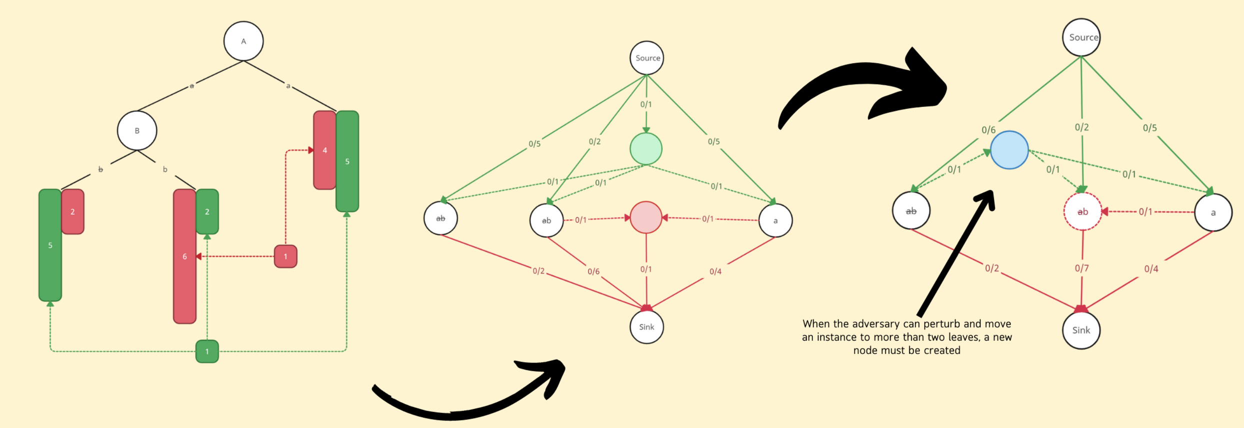

Figure 4

The same can be done for a decision tree with adversarial samples. This can be advantageous when it comes to computing upper and lower bounds.

Experiment Design

- Measure runtimes of our algorithm and bruteforce algorithm, under varying conditions.
- Plot how the dynamic programming approach scales in runtime with different maximum depths and amount of features
- Plot the difference in runtime between the DP approach and bruteforce algorithm

Experiment results:

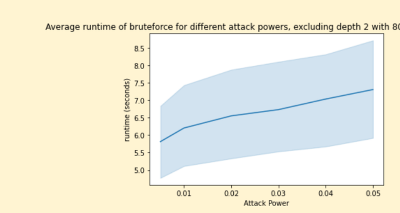

Figure 5
Average runtime for brute-force model when varying runtime

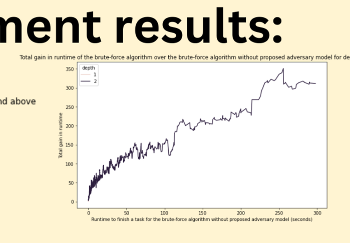

Figure 6
Total speedup of proposed model over brute-force model

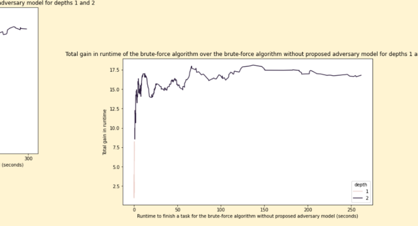

Figure 8
Total speedup of the brute-force model over a brute-force model that does not use the proposed adversary model.

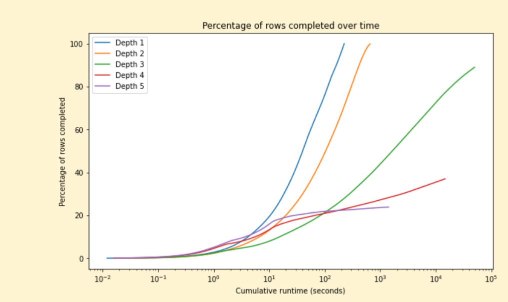

Figure 9
Percentage of optimal trees computed over time, for proposed algorithm.

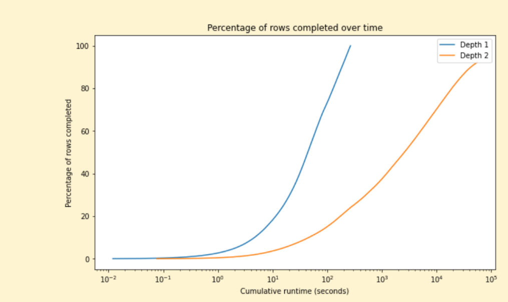

Figure 9
Percentage of optimal trees computed over time, for brute-force model.

Conclusion

- Our experiments that this model outperforms a bruteforce approach at a linear rate with respect to the runtime of the bruteforce model
- The model can comforteably scale up to depth 3, finding optimal decision trees of max depth 4 and 5 is feasible.
- The model might be improved with a more specialized algorithm to solve the max-flow problem, and possibly a more efficient network flow representaton of the problem

Limitations

This approach only considers binary trees with binary labels, it has not yet been extended to a more general version with continuous data and continuous labels or more than 2 labels. This makes the approach difficult to compare with other state-of-the-art methods, such as ROCT.

Future work

- Extend to a more general version with continuous data, continuous labels and non-binary labels. Followed with a direct comparison to state-of-the-art models.
- Optimizing robust trees under other objectives such as MSE, fairness, or survival analysis.

Contact me

Email: a.b.c.bien@student.tudelft.nl
Phone: +31 6 94 20 69 42