Learning to Explain Voting Rules

Extended Abstract

Inwon Kang
Rensselaer Polytechnic Institute
Troy, NY, United States
inwon.kang04@gmail.com

Qishen Han
Rensselaer Polytechnic Institute
Troy, NY, United States
hnickc2017@gmail.com

Lirong Xia
Rensselaer Polytechnic Institute
Troy, NY, United States
xialirong@gmail.com

ABSTRACT

Explaining the outcome of an election is a crucial task to address, especially in the case of complex voting rules. For those without a background in social choice, understanding the result of an election with a complex voting rule can be difficult. One possible way of explaining a voting rule is by using a decision tree structure, allowing the reader to follow the reasoning behind the outcome.

This work proposes a methodology for explaining voting rules using decision-tree-based classifiers. Using simple features, the classifiers can be trained to a high accuracy while maintaining a human-readable size. We test this framework with well-established voting rules – Copeland, Kemeny-Young, Ranked Pairs and Schulze – to generate explanations for each election's outcome. We experiment with different decision tree algorithms on a synthetic dataset to generate explanations for the election outcome. We find that Copeland and Schulze under three candidates can be learned perfectly using an optimized decision tree algorithm, while cases of other rules have high accuracy experimentally.

KEYWORDS

Social Choice, Artificial Intelligence, Explainable Voting, Machine Learning, Explainable AI, XAI

ACM Reference Format:
Inwon Kang, Qishen Han, and Lirong Xia. 2023. Learning to Explain Voting Rules: Extended Abstract. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 3 pages.

1 INTRODUCTION

Nowadays, voting is applied for decision-making in every corner of life, ranging from picking the school mascot to making high-stakes decisions made by multiple supercomputers, and it has become an essential part of many complex systems. While voting can aggregate individual preferences and make collective decisions, it is also important to convince the participants that the outcome of the voting successfully reflects their wishes as a whole. This is a relatively easy task for simple rules like plurality. However, when the complexity of the voting rule grows, explaining the voting outcome can be much more challenging. Without expertise in the area, it's hard for untrained individuals to understand the technical details and insights of voting rules as complicated as Schulze [14]. Therefore, to convince the voters, an explanation of the voting outcome should be created from simple features, for example, the position of a candidate in the rank and the competition between two candidates.

The main question we aim to answer in this paper is as follows: **Can we generate explanations for voting rules via simple features?**

Past literature [3–5, 8, 12, 13] has studied the axiomatic approach for explaining voting outcomes. The axiomatic approach explains the voting outcome as a chain of simple axioms [8]. While this approach is logically solid, the explanations became larger and less effective as the size of the voting profile grows. Suryanarayana et. al. [15] proposed a featured-based explanation method via crowdsourcing and algorithmic generating. Nevertheless, the effectiveness of their method is measured subjectively via surveys of participants without an objective guarantee, e.g. correct or perfect explaining the result. The development of machine learning provides a powerful perspective and tool for analyzing voting rules. Multiple works have demonstrated that deep models such as MLP or GNN can be used to estimate and emulate the behavior of a voting rule [2, 7]. However, these deep models are equally or even more complex than the voting rule itself and still need to improve in terms of interpretability.

This work focuses on explaining voting rules via *decision tree models*. The decision tree model has several advantages. It is powerful enough to learn and characterize a voting rule via a series of conditions on different features, which brings a correctness and completeness guarantee.

2 EXPERIMENTS

2.1 Experiment Settings

Voting Rules. A voting rule chooses a set of winners from a set of candidates according to a voting profile. A set of candidates is defined as C, with $|C| = m$. A vote V is a linear ordering of these candidates, namely $V \in L(C)$ and $|V| = n$, where $L(C)$ denotes all the possible orderings. A voting profile P is composed of n votes, thus $P \in L(C)^n$. Throughout this work, we use m to denote the number of candidates and n to denote the size of the profile. A voting rule $r : L(C)^n \to 2^C \setminus \{\emptyset\}$ maps a profile to a set of winners from the candidates. We run our experiments on four voting rules: Copeland, Ranked Pairs, Schulze, and Kemeny Young. (See [6] for their formal definition)

Feature Extraction. We extract features we refer to as *pairwise margins* from the voting scenarios as inputs to our decision trees. A pairwise margin is defined as the binary difference between every possible pair of pairs of candidates, i.e. the difference between the pairwise victories of two pairs. For example, if $|A \succ B| = 3$ and $|B \succ C| = 2$, the pairwise margin feature is $|A \succ B| > |B \succ C|$ is encoded as 1, while $|B \succ C| > |A \succ B|$ is encoded as -1.

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

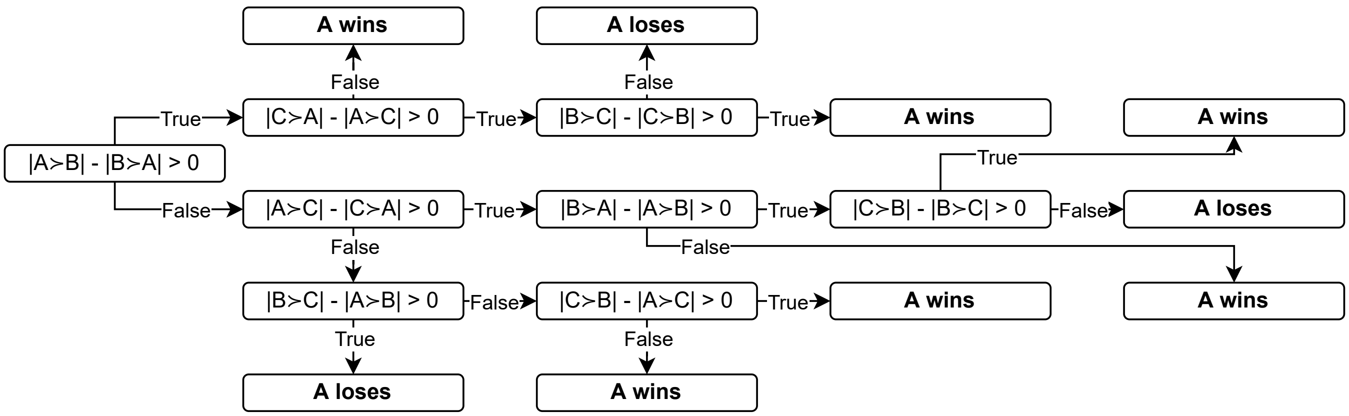

Figure 1: GOSDT tree predicting A's victory for the Copeland rule, $m = 3$.

Decision Tree Algorithms. We consider 4 impelementations of decision tree algorithms, namely XGBoost [9], scikit-learn [11], Generalized Optimized Sparse Decision Tree (GODST) [10], and Hierarchical Shrinkage [1].

Voting Scenario. We conduct our experiment with varying values of m and n and to determine how different candidates and voters influence the learning process. We set n to be even valued so that scenarios with ties are included in the training set. Each profile was generated by randomly shuffling every possible ordering of the candidates. Our we used 10,000 profiles for each experiment. We train the trees for different values of $m \in [3, 5]$, and $n \in [10, 100]$ incremented by 10. All of our experiments were conducted on a system with 2 AMD EPYC 7313 16-Core Processors with 256G RAM running Ubuntu 20.04.6 LTS.

	Copeland	KY	RP	Schulze
XGBoost	1.0	0.99	1.0	1.0
GOSDT	1.0	0.99	1.0	1.0
Scikit-Learn	0.82	0.96	0.95	0.95
HS Decision Tree	0.81	0.96	0.95	0.95

Table 1: Average accuracy score of best trees learned by model for $m = 3, n = 100$

2.2 Training results

Learning for Different Voting Rules. As shown in Table 1, Copeland, Schulze, and Ranked Pairs can be effectively learned by the decision tree models in the $m = 3$ setting. However, the Kemeny-Young rule posed some challenges, with the resulting decision trees covering most of the cases but not failing to achieve a perfect score. This may be due to the limitation of our feature setup, which is more related to the WMG settings and leaves room for future investigation.

Working with a Larger Voting Setup. We also considered larger voting scenarios with $m > 3$ and $n > 100$. We find that with larger m, the decision trees end up being much more complex and that it is difficult to randomly generate a training dataset that covers all possible corner cases that can occur in these settings. One possible remedy for this is to brute-force generate every possible combination of votes. However, number of possible combinations grows exponentially, which can pose a computational challenge. We also find that the value of n does not have a significant impact on the learning process past a certain threshold. This is due to the fact that there are a finite number of possible WMG settings for the WMG based voting rules, so once the number of voters grows past a certain threshold, the possible combinations of pairwise margin features stays the same.

Comparing Different Tree Algorithms. While most models had a high accuracy score for voting rules that satisfy the Condorcet criterion, only XGBoost and GOSDT were able to learn a *perfect* rule that scored 100% accuracy on the test dataset. Upon examining the tree structure, we find that GOSDT indeed produces smaller trees than the classic CART algorithm while still being correct. Compared to the 24 nodes present in the XGBoost tree for the Copeland rule, the GOSDT tree, as can be seen in Figure 1, outputs a tree with only 16 nodes while achieving the same perfect performance.

3 CONCLUSION

We propose a framework for training decision trees to learn the outcome of a voting rule and serve as an explanation. We train different decision tree models on synthetic voting profile data for different voting rules, and compare the performance and efficiency of different decision tree mining algorithms.

From our experiments, we find that voting rules that incorporate rankings into their mechanism (i.e., satisfy the Condorcet criterion) can be well estimated by decision tree models and produce a human-readable diagram. The number of voters in a profile does not impact the performance of the trees while increasing the number of candidates requires a tree with a greater depth.

One direct extension of this work is generating simple decision trees for votes with multiple alternatives. Another future direction is to create new explainable voting rules based on decision trees. We have demonstrated that decision trees can act as proxies for voting rules. If we reverse this process, we can design new voting mechanisms that are both explainable and axiomatically desirable.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their helpful suggestions. LX acknowledges NSF #1453542 and #2007476, and a Google gift fund for support.

REFERENCES

[1] Abhineet Agarwal, Yan Shuo Tan, Omer Ronen, Chandan Singh, and Bin Yu. 2022. Hierarchical Shrinkage: Improving the accuracy and interpretability of tree-based models. In *Proceedings of the 39th International Conference on Machine Learning*. 111–135.

[2] Cem Anil and Xuchan Bao. 2021. Learning to Elect. In *Advances in Neural Information Processing Systems*. 8006–8017.

[3] Arthur Boixel and Ulle Endriss. 2020. Automated Justification of Collective Decisions via Constraint Solving. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*. 168–176.

[4] Arthur Boixel, Ulle Endriss, and Ronald de Haan. 2022. A calculus for computing structured justifications for election outcomes. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*.

[5] Arthur Boixel, Ulle Endriss, and Oliviero Nardi. 2022. Displaying Justifications for Collective Decisions. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*. 5892–5895. Demo Track.

[6] Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia. 2016. *Handbook of Computational Social Choice*. Cambridge University Press.

[7] Dávid Burka, Clemens Puppe, László Szepesváry, and Attila Tasnádi. 2021. Voting: a machine learning approach. *European Journal of Operational Research* (2021).

[8] Olivier Cailloux and Ulle Endriss. 2016. Arguing about Voting Rules. In *Proceedings of the 2016 International Conference on Autonomous Agents Multiagent Systems*. 287–295.

[9] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 785–794.

[10] Jimmy Lin, Chudi Zhong, Diane Hu, Cynthia Rudin, and Margo Seltzer. 2020. Generalized and Scalable Optimal Sparse Decision Trees. In *Proceedings of the 37th International Conference on Machine Learning*. 6150–6160.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[12] Dominik Peters, Ariel D Procaccia, Alexandros Psomas, and Zixin Zhou. 2020. Explainable Voting. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.). 1525–1534.

[13] Ariel D. Procaccia. 2019. *Axioms Should Explain Solutions*. Springer International Publishing, 195–199.

[14] Markus Schulze. 2011. A new monotonic, clone-independent, reversal symmetric, and condorcet-consistent single-winner election method. *Social choice and Welfare* 36, 2 (2011), 267–303.

[15] Sharadhi Alape Suryanarayana, David Sarne, and Sarit Kraus. 2022. Justifying Social-Choice Mechanism Outcome for Improving Participant Satisfaction. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*. 1246–1255.