My Machine Learning Model Paper

So this report talks about the design and making of a political alignment quiz application developed in C plus plus, alongside a computer program that learns stuff in pythn that analyzes collected user responses to improve prediction acuracy over time. The thing mixes interactive survey logic with info-driven insights to identify a user's political leaning based on their answers to a series of questions.

My Rationale for the Chosen ML Model

The computer program that learns stuff selected for this project is a Random Forest thingy. I picked it because of several factors: Random Forests are doesnt mess up to much, handle different types of info well, and provide pretty good results even with relatively small infosets. They also offer shows which questions matter most, which can help identify which survey questions are most influential in predicting political alignment. Given the nature of the input info, numerical scores assigned to political parties based on user responses, Random Forests are well-suited to capture complex patterns and interactions. (McGuire & Delahunt, 2020)

Data Storage and Processing

The user responses are stored in a spreadsheet file with the name 'responses.csv'. Each row in the file represents a single user's interaction with the quiz, including their cumulative scores for each political party, the predicted party based on those scores, and the user's self-identified political label. This organized way enables easy integration into the pythn-based machine learning pipeline. The pythn script reads the CSV using pandas, maps party labels to numeric values, and splits the info into practice and test info. The features used for training are the party scores, and the thing we wanna guess is the user's actual political label.

Improving Predictions Over Time

As more users complete the quiz and their responses are appended to the spreadsheet file, the info set grows, enabling the modle to learn from a broader range of inputs. The training script can be rerun periodically to retrain the modle on the updated infoset, thereby improving its predictive acuracy. This learning again and again ensures that the modle adapts to changing patterns in user responses, becoming more reliable over time. Also like, the use of Random Forest allows the modle to works okay with new stuff from limited info, making it effective even in early stages of putting it online.

Evaluation of my Model Accuracy and Limitations

The modle's performance is evaluated using standard ways to check how good it is: numbers that show how good it guesses. These metrics provide a comprehensive view of how well the modle predicts political alignment. Initial tests show promising results, with acuracy and F1-scores typically above 80%, depending on the size and diversity of the infoset. But like, the modle has limitations. It may struggle with answers that are kinda in the middle that do not strongly favor any party. Also like, if the infoset is not fair, more users identify with one party than others the modle may become biased. To mitigate this, techniques such as fancy ways to fix unfair info can be employed during training. (Siddique et al., 2023)

Conclusion

This project demonstrates the effective integration of a C plus plus survey application with a pythn-based computer program that learns stuff. By collecting user responses and continuously training the modle, the system evolves to provide increasingly accurate predictions of political alignment. The use of Random Forests ensures robustness and easy to understand, while the structured info storage facilitates seamless analysis. Maybe later we can add real-time prediction using a saved modle, changing questions on the fly, and putting it online as a web application.

References

McGuire, S. K., & Delahunt, C. B. (2020). Predicting United States Policy Outcomes with Random Forests. Institute for New Economic Thinking. https://www.ineteconomics.org/uploads/papers/McGuire-and-Delahunt-predictingPolicy\_INET\_25oct2020.pdf

Siddique, S., et al. (2023). Survey on Machine Learning Biases and Mitigation Techniques. Machine Learning and Knowledge Extraction, 4(1), 1. https://www.mdpi.com/2673-6470/4/1/1