ISE 625 Project Proposal

Stable decision trees for suicide experience prediction

Adhithya Bhaskar, Michelle Gelman

Problem Context and Background

- **Aim:** Predict suicidal experiences among youth experiencing homelessness (YEH)
- The provided decision tree model is unstable to change in train-test splits
- Can we find a robust model invariant to shifts in distributions that will procude the same best features indicative of suicide ideaiton and attempts?

Dataset Considerations

- Missing data
  - 584, 587 samples remaining for each prediction model from initial listwise deletion method form oriignal 940 total samples
  - 4% of data set mising for suicideideation and attempy (36 and 40 samples respectively)
- Imbalanced classes
  - 83% labeled 2, 16% labeled 1 for suicideidea class
  - 88% labeled 0, 11% labeled 1 for suicideattempt class

Stable Decision Trees

- Bertsimas et al. (2023) proposes a method to create stable decision trees
- 1 of 6 datasets used is publicly available - Breast Cancer dataset (UCI Machine Learning Repository)
- Used to test and compare our implementation
- With satisfactory results, we will apply our implementation to the suicide dataset

Proposed Plan - 1. Understand the instability of provided DT

- Given model exists as 2 python files (for `suicidea` and `suicattemp`)
- Create simple example to deterministically try various splits
- Empirically measure the difference in predicted splits

Proposed Plan - 2. Implement a stable DT (Bertsimas et al. 2023)

1. Train initial set **(T0)** of decision trees on a subset of the data
2. Train second set **(T)** of decision trees on the full dataset
3. Compute **average distance** of each tree in **T** to **T0**
4. Compute performance metrics (AUC) of trees on validation/test set
5. For the trees in **T** we select the Pareto optimal trees by optimizing for predictive performance and distance to **T0**

Proposed Plan - 3. Measuring effectiveness of proposed model

- Compare the performance of stable trees to the provided DT using metrics we define in step 1

Key optimization algorithms to implement:

- Distance between two trees

$$d(\mathcal{T}_1, \mathcal{T}_2) \;=\; \min_{\{x\}} \sum_{p\in\mathcal{P}(\mathcal{T}_1)} \sum_{q\in\mathcal{P}(\mathcal{T}_2)} d(p,q)\, x_{p,q} \;+\; \sum_{p\in\mathcal{P}(\mathcal{T}_1)} w(p)\, x_p$$

- Pareto optimal tree

$$\mathbb{T}^\star = \operatorname{argmax} f(d_b, \alpha_b)$$