COMP 565 Assignment 3 – xAI

This assignment is worth 8% of your total grade and due at **23:59 on October 23, 2023**. The breakdown of the total points are indicated in the section headers below whenever applicable.

1 (1%) Simulation

Simulate a dataset based on the following steps:

1. Simulate 10 independent input features each sampled from a Bernoulli distribution with $\pi = 0.5$ for sample $i \in \{1, \ldots, N\}$: $x_{i,d} \in Bernoulli(0.5) = \pi^{x_{i,d}}(1-\pi)^{1-x_{i,d}}$.
2. Among those 10 input features, set the first 3 features to be *causal*.
3. Simulate real value output using a small neural network with 1 hidden layer with 2 ReLU hidden units and 1 real value output unit (i.e., 3-2-1 architecture):
   (a) Set the input-hidden weights $w^{(1)}_{d,j} = 1$ for $d \in \{1, 2, 3\}$ and hidden output weights $w^{(o)}_j = 1$ for $j \in \{1, 2\}$ hidden units.
   (b) Simulate real value output:

$$y_i = \sum_{j=1}^{2} w^{(o)}_j \max(0, \sum_{d=1}^{3} w^{(1)}_{d,j} x_{d,i})$$

4. Repeat steps 1-3 to generate $i \in \{1, \ldots, N\}$ with $N \geq 100$ samples.
5. Split the data into 80% training and 20% testing

2 (3%) Shapley values

Pretending that you did not know which of the 10 features are causal from your simulation and implement Shapley values to compute the feature attributions of the 20% testing samples.

1. Train a random forest on the 80% training data using scikit-learn with default option on your simulated data (`https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html`).

2. Verify that your RF regressor works well on the test data based on Pearson correlation between the prediction and the groundtruth output from the 20% test data

3. Compute the Shapley value for the feature attribution of each feature to explain individual predictions from the 20% test examples by the trained RF regression model:

$$\phi_d = \sum_{S \subseteq F \setminus \{d\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} \left[f_{S \cup \{d\}}(x_{S \cup \{d\}}) - f_S(x_S) \right]$$

4. Plot a scatterplot with the Shapley values $\phi_d^{(n)}$ on the individual test examples as the x-axis and the ordered indices of the 10 features as the y-axis (where the first 3 are the causal features).

5. Plot the global feature attributions for the 10 features in a bar plot with the mean Shapley values $I_d = \frac{1}{N} \sum_{n=1}^{N} |\phi_d^{(n)}|$

See slide 36 or slide 37 in Lecture 7 for the desirable style of the two plots. You are free to implement a Kernel SHAP but it is completely optional.

3 (4%) DeepLIFT

DeepLIFT explains feature attributions by difference-from-reference. This is done by backpropagating multipliers in chain rule. The multipliers are defined by linear rule, rescale rule, and RevealCancel rule. The feature attributions are the multipliers times the input change.

1. Implement a 1-layer neural network with the same architecture as the network you use to simulate the data except for using 10 input features (i.e., 10-2-1 architecture with ReLU activation).

2. To make your job easier, you may set the input-to-hidden and hidden-output weights to be also the same as simulated weights. Also, you don't have to consider general cases of MLP. As long as your code works with the 10-2-1 network to get the correct feature importance you will receive full mark.

3. Implement the linear and rescale rule using *all-zeros inputs as the reference* to compute the multiplier at each layer $m_{\Delta x_d \Delta y}$.

4. Implement backpropagation of the multipliers to compute the feature importance scores $C_d^{(n)} = m_{\Delta x_d \Delta y} \Delta x_d$ on each individual prediction

5. Plot a scatterplot with the DeepLIFT feature importance values $C_d^{(n)}$ on the individual test examples as the x-axis and the indices of the 10 features as the y-axis (where the first 3 are the causal features).

6. Plot the global feature attributions as bar plot with the mean DeepLIFT feature values $I_d = \frac{1}{N}\sum_{n=1}^{N}|C_d^{(n)}|$

You are free to implement and explore RevealCancel rule or use a different neural network architecture as the ground truth network but it is completely optional.

4 File to submit

Submit your code as a notebook in either iPython notebook or R markdown with respective file names as `COMP565_A3_xai.ipynb` or `COMP565_A3_xai.Rmd`. Your notebook implements *all of the above 3 tasks* in ONE file. As you will generate your own simulated data, your file should be self-contained. We should be able to run your notebook and produce the above plots to evaluate the correctness of your implementation.