Machine Learning for Causal Inference with Continuous Treatments

Eray Turkel (eturkel@stanford.edu)

Motivation
Causal inference can be seen as a missing data problem and the problem becomes much more complicated in the case of continuous treatments.

Benchmark Comparison

Our methods focus on further improving the density estimation step by using conditional Generative Adversarial Networks (Adversarial Networks).

Application: Effect of Political Ads on Donations

We use data from Urban and Niebler (2014), where the authors measure the causal effect of ads on money raised in 16000 US zipcode areas where candidates aired ads "unintentionally" in 2008. They discretize the treatment, which leads to loss of information. We will estimate the full response function. We split the data and use the training set to train our models. We use a conditional kernel density estimator and pick the optimal bandwidths through cross-validated maximum likelihood, using 50 restarts and an Epanechnikov kernel. For GBM, we train the model using 15000 trees and pick the optimal number of trees to use at prediction time through 10 fold cross validation.

Simulation Results and Future Work

Our results are higher than the binary-discretized results obtained in the paper, and we see some decreasing marginal returns to advertising that a simple binary analysis would miss. The flexible linear model fits a U-shaped curve which is difficult to interpret, probably because the function is not flexible enough. Future work will focus on further improving the density estimation step by using modern techniques, such as using conditional Generative Adversarial Networks.

Estimating Response Functions

Under the assumption of selection on observables and unconfoundedness, existing literature estimates response functions in two steps. The first step involves estimating the propensity score. This is usually done by estimating two separate parametric models through maximum likelihood. We propose the use of more flexible and modern machine learning methods. We use conditional kernel density estimators (also called Parzen Window Estimators) for the first step, and neural networks and gradient boosted trees for the second step. Our benchmark comparison will be against a widely used method in the literature, briefly described below.

Y_i = \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip} + \epsilon_i

Simulation Results

To evaluate the accuracy of our proposed method against existing methods, we need to test it in a context with a known data generating process. We run simulation studies with N=2000, 20 covariates, and different levels of confounding, sparsity, and non-linearities in the response function. For the neural network, we use a two-layer network with 4 and 2 nodes and a linear output, trained with regular gradient descent for 100000 epochs, randomly started 5 times, and TanH activation. For the GBM, we use 5000 trees, and use cross validation to determine the optimal number of trees to use on test data. In all settings, we use an unseen test set generated through the same process to evaluate the accuracy of the predictions.

Simulation Setup

Y_i = \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip} X_{ip}^2 + \beta_{p+2} X_{ip} X_{ip}^3... + \eta T_i + \epsilon_i

Results and Future Work

Our results are higher than the binary-discretized results obtained in the paper, and we see some decreasing marginal returns to advertising that a simple binary analysis would miss. The flexible linear model fits a U-shaped curve which is difficult to interpret, probably because the function is not flexible enough. Future work will focus on further improving the density estimation step by using modern techniques, such as using conditional Generative Adversarial Networks.

References


