Modified Causal Forest: Optimal Policy Estimation

Section 1: General information

Welcome to the mcf estimation and optimal policy package.
This report provides you with a summary of specifications and results. More detailed information can be found in the respective output files. Figures and data (in csv-format, partly to recreate the figures on your own) are provided in the output path as well.

Output information for OPTIMAL POLICY ANALYSIS
Path for all outputs:
Q:\SEW\Projekte\MLechner\Projekte und Angebote\Unicef\Kasachstan\Workshops\Astana\Wednesday_examples/example/outputOPTBPS
Detailed text output:
Q:\SEW\Projekte\MLechner\Projekte und Angebote\Unicef\Kasachstan\Workshops\Astana\Wednesday_examples/example/outputOPTBPS/OptPolicy.0.6.0.txt
Summary text output:
Q:\SEW\Projekte\MLechner\Projekte und Angebote\Unicef\Kasachstan\Workshops\Astana\Wednesday_examples/example/outputOPTBPS/OptPolicy.0.6.0_Summary.txt

BACKGROUND

The optimal policy module offers three (basic) algorithms that can be used to exploit fine grained knowledge about effect heterogeneity to obtain decision rules. The current version is implemented for discrete treatments only.

The BEST_POLICY_SCORE algorithm is based on assigning the treatment that has the highest impact at the unit (e.g., individual) level. If the treatment heterogeneity is known (not estimated), this will lead to the best possible result. This algorithm is computationally not burdensome. However, it will not be easy to understand how the implied rules depends on the features of the unit. Its statistical properties are also not clear (for estimated treatment heterogeneity) and there is a certain danger of overfitting, which could lead to an unsatisfactory out-of-training-sample performance.

The BPS_CLASSIFIER classifier algorithm runs a classifier for each of the allocations obtained by the BEST_POLICY_SCORE algorithm. One advantage of this approach compared to the BEST_POLICY_SCORE algorithm is that prediction of the allocation of (new) observations is fast because it does not require to recompute the policy score (as it is the case with the BEST_POLICY_SCORE algorithm). The specific classifier is selected among four different classifiers from scikit-learn, namely a simple neural network, two classification random forests with minimum leaf size of 2 and 5, and ADDABoost. The selection is a made according to the out-of-sample performance of the Accuracy Score of scikit-learn.

The POLICY TREE algorithm builds optimal shallow decision trees. While these trees are unlikely to lead to gloablly optimal allocations, and are computationally much more expensive, they have the advantage that the decision rule is much easier to understand and that some statistical properties are known, at least for certain versions of such decision trees (e.g. Zhou, Athey, Wager, 2023). The basic algorithmic implementation follows the recursive algorithm suggested by Zhou, Athey, Wager (2023) with three (more substantial) deviations (=extensions).

Modified Causal Forest: Optimal Policy Estimation

Extension 1: Since using One Hot Encoding for categorical variables may lead to rather extreme leaves for such variables with many different values when building (shallow) trees (splitting one value against the rest), a more sophisticated procedure is used that allows to have with several values of the categorical variables on both sides of the split.
Extension 2: Constraints are allowed for. They are handled in a sequential manner: First, an approximate treatment-specific cost vector is obtained and used to adjust the policy score accordingly. Second, trees that violate the constraints are removed (to some extent, optional).
Extensions 3: There are a several options implemented to reduce the computational burden, which are discussed below in the section showing the implementation of the policy score.

References
-Zhou, Z., S. Athey, S. Wager (2023): Offline Multi-Action Policy Learning: Generalization and Optimization, Operations Research, INFORMS, 71(1), 148-183.

Modified Causal Forest: Optimal Policy Estimation

Section 2: Optimal Policy

METHOD
The assignment rule is based on allocating units to the treatment with the highest score.

VARIABLES provided
Policy scores: y_pot0, y_pot1, y_pot2
IATEs relative to first treatment state: iate1vs0, iate2vs0
Treatment dependent variables for descriptive analysis: zero, ite1vs0, ite2vs0, x_cont0, iate1vs0, iate2vs0
Variables determining prioritisation of units in case of binding constraints for the best_policy_score method: x_unord0
Treatment: treat
Identifier: id
Oderered features of units: x_cont0, x_cont1, x_cont2, x_ord0, x_ord1, x_ord2
Categorical / unorderered features of units: x_unord0, x_unord1, x_unord2
Features used for variable importance statistics without transformations: x_cont0, x_cont1, x_cont2
Features that are transformed to indicator/dummy variables for variable importance computations (only): x_unord0

COSTS
No user provided costs of specific treatments.

RESTRICTIONS of treatment shares
The following restrictions on the treatment shares are specified 100%, 100%, 30.0%.

Section 2.1: Optimal Policy: Training

COMPUTATION
6 logical cores are used for processing.

DATA PREPARATION
Variables without variation are removed.
Variables that are perfectly correlated with other variables are removed.
Dummy variables with less than 10 observations in the smaller group are removed.
Rows with any missing values for variables needed for training are removed.

RESTRICTIONS on treatment shares
Restrictions are ignored if they are not binding.
If they are binding, then several methods are used to enforce them (almost) exactly:
1) Prioritize units that benefit most.
2) Deny a random selection of units their best option.
3) Prioritize units with higher values of x_unord0.

Modified Causal Forest: Optimal Policy Estimation

Section 2.2: Optimal Policy: Evaluation of Allocation(s)

Main evaluation results.

Note: The output files contain relevant additional information, like a descriptive analysis of the treatment groups and variable importance statistics.

Evaluation of treatment allocation

Allocation	*Value function*	*Share of 0 in %*	*Share of 1 in %*	*Share of 2 in %*
All bb	1.6566	19.3	40.6	40.1
All bb_restrict_random	0.6364	33.2	36.7	30.1
All bb_restrict_largest_gain	1.6154	26.3	43.6	30.1
All bb_restrict_largest_gain_random_order	1.5663	22.8	47.1	30.1
All bb_restrict_largest_gain_x_unord0	1.5582	22.8	47.1	30.1
All observed	0.5182	33.3	33.4	33.3
All random	0.6532	30.2	34.0	35.8
Switchers bb	1.6876	19.68	39.94	40.38
Switchers bb_restrict_random	0.6689	33.93	36.64	29.43
Switchers bb_restrict_largest_gain	1.6167	27.58	41.94	30.48
Switchers bb_restrict_largest_gain_random_order	1.573	24.05	46.06	29.88
Switchers bb_restrict_largest_gain_x_unord0	1.5626	23.71	44.86	31.43
Switchers random	0.68	30.38	33.38	36.24

Note: Allocation analysed is the SAME as the one obtained from the training data.