<u>Manual for the code</u>

In this repository, we provide the code for the research paper ”Heteroscedasticity-aware stratified sampling to improve uplift modeling”. The paper first appeared in its initial version in the thesis ”Statistical approaches to enhance decision support in time series and causality problems”. Later the paper was revised for publication in the EJOR. For this revision, the code was changed. Hence, the code in this repository differs from the code applied in the initial manuscript in the Ph.D. thesis.

Folder structure

To run the code, applicants need the following folder structure in the working directory:

-code/

    -sampling_parameters/

    -experiment_training/

    -experiment evaluation/

    -helper/

    -graphics/

-data/

-results/

    …

The”code” folder can be downloaded directly from the github repository. In the”data” folder, the applicant needs to save the Criteo and Lenta data sets, if s/he wants to run the experiment on these data sets. In the ”results” folder, intermediate and final results get saved. This folder has a relatively complex sub-folder structure. The applicant can download the results.zip file, and unpack it to obtain the required sub-folder structure.

The folder ”code” contains a script with the simulation scenarios and five sub-folders:

The folder ”sampling_parameters” contains the code to obtain sampling parameters. The different data sets require different pre-processing, hence there are separate scripts for the Criteo, Lenta and simulated data. Running the scripts will save the resulting sampling parameters in a corresponding results folder (which the applicant needs to create before running the code). In our experiment, we tried different outcome model types (e.g. complexity restricted, Platt calibrated,…). For each of these outcome model types, the code calculates the HS sampling parameters and saves them. The outcome model types (HS isotonic, HS Platt, HS rest, HS tuned and HS small) in our experiment are described in section 1.1.1 of the supplementary material. The corresponding names in the code are: *'isotonic', 'platt', 'rest', 'tuned', 'small'*.

The code to obtain sampling parameters has a relatively low run time of a few minutes.

The folder”experiment_training” contains the code for the uplift model training part of our computational experiment. Just like for the ”sampling_parameters” folder, there are different scripts for different data sets. Scripts ending with “_leverage.R” provide the code for leverage score sampling.

To run the code for a data set, one needs to set the parameter *samp_param_method.* Here the applicant needs to choose the outcome model type (*'isotonic', 'platt', 'rest', 'tuned' or 'small'*) which s/he wants to apply for HS sampling. When running the code, the sampling parameters of the chosen outcome model (which need to be obtained using the code in the ”sampling_parameters” folder) are read. Then a for loop with 1000 iterations is started. In each of the iterations, the HS sampling procedure and the complete random sampling procedure are applied. Afterwards, various uplift model types are trained on the so generated data sets. Note that the code includes commented out code for training a causal BART. This is because in the original manuscript version (where we conducted the experiment only for a single outcome model type), we applied causal BART. In the revised version, we then had to repeat the experiment for multiple outcome model types, which made it computationally unfeasible to apply a method with such high run time requirements. The training code has a very long run time (a day or even longer for each sampling method and data set.)

The folder ”experiment evaluation” contains the code for the ATE estimation and uplift model evaluation part of the experiment. Again, there are different scripts for the different data sets. In each of these scripts, the applicant needs to select, which of the outcome model types and its HS sampling parameters should be applied, by choosing the parameter *samp_methods*. Because the run time of the scripts for this part of the experiment is moderate, it is possible to choose multiple outcome model types, which then get applied one after another. The experimental procedure in the scripts is the following: A large data set gets loaded, using the chosen outcome model and sampling parameters, a column gets added to the data, which identifies the stratum (S_H or S_L) of an individuum. Some observations of this data set are choosen as a training set, to train outcome models $\hat{\mu}_0(x)$ and $\hat{\mu}_1(x)$. Based on these outcome models, T-learner predictions $\hat{\mu}_1(x) - \hat{\mu}_0(x)$ get generated for all the data (excluding the training sample). Also, predictions of the form $\hat{\varphi}(x) = p \cdot \hat{\mu}_1(x) + (1-p) \cdot \hat{\mu}_0(x)$ get generated for the covariate adjustment. Then a for loop with 1000 iterations starts. In each of these iterations, a test set of size 10,000 is chosen via complete random sampling and via HS sampling from the data. On these data sets, Qini curves of T-learner are build and ATE estimates are generated. When the for-loop is finished, the empirical variance of the Qini curves and ATE estimates gets calculated.

The folder ”helper” contains three scripts. The script ”functions_calibration.R” provides for each of the calibration methods, Platt calibration and isotonic calibration, a function, which can be applied to an outcome model. Such a function then yields a new prediction function as output. This prediction function corresponds to $T(\hat{\mu}(x))$, so the calibration T(), applied to the original outcome model $\hat{\mu}(x)$. The script ”functions_effect_estimation.R” contains a function to calculate the AUQ and a function for stratified estimation of the ATE (equation (3) of the paper). The script ”functions_simulation.R” contains a function to generate the simulated data sets and a function ”get_sampling_parameters()” to obtain HS sampling parameters for a given outcome model and data set. This function implements the algorithm described in section 4.1 of the main paper.

The folder “graphics“ contains the code for the graphics in the paper.