Simulation_with_Python_experiment_data

2024-05-13

Preparing the data

```r
library(banditsCI)
setwd(dirname(rstudioapi::getSourceEditorContext()$path))
gammahat <- as.matrix(read.csv('scores.csv', header = FALSE))
probs_array <- as.matrix(read.csv('probs.csv', header = FALSE))

set.seed(123)

policy1_main <- list(
  # includes all matrices in policy1 and policy0
  matrix(
    c(rep(1, nrow(gammahat)), rep(0, nrow(gammahat)), rep(0, nrow(gammahat))),
    nrow = nrow(gammahat)),
  matrix(
    c(rep(0, nrow(gammahat)), rep(1, nrow(gammahat)), rep(0, nrow(gammahat))),
    nrow = nrow(gammahat)),
  matrix(
    c(rep(0, nrow(gammahat)*(ncol(gammahat)-1)), rep(1, nrow(gammahat))),
    nrow = nrow(gammahat)))
```

Running the simulation

```r
# main effects
output_estimates <- output_estimates(policy1 = policy1_main,
                 gammahat = gammahat,
                 probs_array = probs_array,
                 floor_decay = 0.7)

# Get estimates for treatment effects of policies as contrast to control
# \delta(w_1, w_2) = E[Y_t(w_1) - Y_t(w_2)].
# In Hadad et al. (2021) there are two approaches.
## The first approach: use the difference in AIPW scores as the unbiased scoring
## rule for \delta (w_1, w_2)
### The following function implements the first approach by subtracting policy0,
### the control arm, from all the arms in policy1, except for the control arm
### itself.
out_full_te1.1 <- output_estimates(
  policy0 = policy1_main[[1]],
  policy1 = list(policy1_main[[3]]),
  contrasts = "combined",
  gammahat = gammahat,
  probs_array = probs_array,
```

```r
  floor_decay = 0.7)

out_full_te1.2 <- output_estimates(
  policy0 = policy1_main[[2]],
  policy1 = list(policy1_main[[3]]),
  contrasts = "combined",
  gammahat = gammahat,
  probs_array = probs_array,
  floor_decay = 0.7)

## The second approach takes asymptotically normal inference about
## \delta(w_1, w_2): \delta ^ hat (w_1, w_2) = Q ^ hat (w_1) - Q ^ hat (w_2)
out_full_te2.1 <- output_estimates(
  policy0 = policy1_main[[1]],
  policy1 = list(policy1_main[[3]]),
  contrasts = "separate",
  gammahat = gammahat,
  probs_array = probs_array,
  floor_decay = 0.7)

out_full_te2.2 <- output_estimates(
  policy0 = policy1_main[[2]],
  policy1 = list(policy1_main[[3]]),
  contrasts = "separate",
  gammahat = gammahat,
  probs_array = probs_array,
  floor_decay = 0.7)
```

Function to compare the results

```r
# Compare the two approaches for uniform and non_contextual_two_point
compare_methods <- function(output_estimates,
                            out_full_te1.1,
                            out_full_te1.2,
                            out_full_te2.1,
                            out_full_te2.2) {
  # Initialize an empty data frame to hold the comparison data
  comparison_df <- data.frame(method = character(),
                              estimate = numeric(),
                              std_error = numeric(),
                              contrasts = character(),
                              policy = integer(),
                              from = character(),
                              stringsAsFactors = FALSE)

  # Function to process and append data
  process_data <- function(data, policy_num, contrasts, from) {
    for (method in c("uniform", "non_contextual_twopoint")) {
      if (method %in% rownames(data)) {
        row <- data.frame(
          method = method,
          estimate = data[method, "estimate"],
```

```r
          std_error = data[method, "std.error"],
          contrasts = contrasts,
          policy = policy_num,
          from = from,
          stringsAsFactors = FALSE
        )
        comparison_df <<- rbind(comparison_df, row)
      }
    }
  }

  # Process and append data for each subset and condition
  process_data(output_estimates[[1]], "0", "main effect", "output_estimates[[1]]")
  process_data(output_estimates[[2]], "1", "main effect", "output_estimates[[2]]")
  process_data(output_estimates[[3]], "2", "main effect", "output_estimates[[3]]")
  process_data(out_full_te1.1[[1]], "(0,1)", "combined", "out_full_te1.1[[1]]")
  process_data(out_full_te1.2[[1]], "(0,2)", "combined", "out_full_te1.2[[1]]")
  process_data(out_full_te2.1[[1]], "(0,1)", "separate", "out_full_te2.1[[1]]")
  process_data(out_full_te2.2[[1]], "(0,2)", "separate", "out_full_te2.2[[1]]")

  return(comparison_df)
}
```

Comparing the results

```r
comparison_df <- compare_methods(output_estimates,
                                 out_full_te1.1,
                                 out_full_te1.2,
                                 out_full_te2.1,
                                 out_full_te2.2)
# Show the comparison data frame sorted by method
comparison_df <- comparison_df[order(comparison_df$method), ]

# print the comparison data frame as a table
knitr::kable(comparison_df)
```

	method	estimate	std_error	contrasts	policy	from
2	non_contextual_twopoint	0.8267759	0.1125030	main effect	0	output_estimates[[1]]
4	non_contextual_twopoint	0.9138038	0.0580545	main effect	1	output_estimates[[2]]
6	non_contextual_twopoint	1.1012259	0.0104059	main effect	2	output_estimates[[3]]
8	non_contextual_twopoint	0.3891761	0.2723742	combined	(0,1)	out_full_te1.1[[1]]
10	non_contextual_twopoint	0.4351515	0.2386683	combined	(0,2)	out_full_te1.2[[1]]
12	non_contextual_twopoint	0.2744500	0.1129833	separate	(0,1)	out_full_te2.1[[1]]
14	non_contextual_twopoint	0.1874221	0.0589797	separate	(0,2)	out_full_te2.2[[1]]
1	uniform	0.7198237	0.2557261	main effect	0	output_estimates[[1]]
3	uniform	0.7106815	0.2011189	main effect	1	output_estimates[[2]]
5	uniform	1.1055150	0.0106840	main effect	2	output_estimates[[3]]
7	uniform	0.3856913	0.2559502	combined	(0,1)	out_full_te1.1[[1]]
9	uniform	0.3948335	0.2014011	combined	(0,2)	out_full_te1.2[[1]]
11	uniform	0.3856913	0.2559491	separate	(0,1)	out_full_te2.1[[1]]
13	uniform	0.3948335	0.2014024	separate	(0,2)	out_full_te2.2[[1]]