Simulation with Python Experiment Data (Contextual)

2024-10-20

1. Preparing the data

```r
library(reticulate)
```

```
## Warning: package 'reticulate' was built under R version 4.3.3
```

```r
library(banditsCI)

# Read in data generated in python notebook

gammahat <- as.matrix(read.csv('gammahat.csv', header = FALSE))
muxs <- as.matrix(read.csv('muxs.csv', header = FALSE))

# probabilities
np <- import("numpy")
# 3 dimensions: time, contexts, treatment arms
probs_array <- np$load("probs.npy")


# Define the policies we want to evaluate
policy1_main <- list(
  # includes all matrices in policy1 and policy0
  matrix(
    c(rep(1, nrow(gammahat)),
      rep(0, 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)),
      rep(0, nrow(gammahat))),
    nrow = nrow(gammahat)),
  matrix(
    c(rep(0, nrow(gammahat)),
      rep(0, nrow(gammahat)),
      rep(1, nrow(gammahat)),
      rep(0, nrow(gammahat))),
    nrow = nrow(gammahat)),
  matrix(
    c(rep(0, nrow(gammahat)),
      rep(0, nrow(gammahat)),
      rep(0, nrow(gammahat)),
      rep(1, nrow(gammahat))),
```

```
    nrow = nrow(gammahat)))
```

2. Estimation

2.1 Main Effects

```
best_mtx <- matrix(0, nrow = nrow(gammahat), ncol = ncol(gammahat))
best_mtx[, which.max(colMeans(muxs))] <- 1

output_estimates_best <- output_estimates(policy1 = list(best_mtx),
                                          gammahat = gammahat,
                                          probs_array = probs_array,
                                          floor_decay = 0.8)

uniform_var <- estimate(0:(nrow(gammahat)-1),
                        gammahat,
                        best_mtx)
output_estimates_best[[1]] <- rbind(output_estimates_best[[1]],
                                    'uniform_var' = c(uniform_var['estimate'],
                                                      sqrt(uniform_var['var'])))


optimal_mtx <- matrix(0, nrow = nrow(gammahat), ncol = ncol(gammahat))
optimal_mtx[cbind(1:nrow(gammahat), apply(muxs, 1, which.max))] <- 1

output_estimates_optimal <- output_estimates(policy1 = list(optimal_mtx),
                                             gammahat = gammahat,
                                             probs_array = probs_array,
                                             floor_decay = 0.8)

uniform_var <- estimate(0:(nrow(gammahat)-1),
                        gammahat,
                        optimal_mtx)
output_estimates_optimal[[1]] <- rbind(output_estimates_optimal[[1]],
                                       'uniform_var' = c(
                                         uniform_var['estimate'],
                                         sqrt(uniform_var['var'])))
```

2.2 Treatment effects in contrast to control

$\delta(w_1, w_2) = E[Y_t(w_1) - Y_t(w_2)]$.

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 <- output_estimates(
  policy0 = best_mtx,
  policy1 = list(optimal_mtx),
  contrasts = "combined",
  gammahat = gammahat,
  probs_array = probs_array,
  floor_decay = 0.8)
```

```
uniform_var <- estimate(0:(nrow(gammahat)-1),
                        gammahat,
                        optimal_mtx-best_mtx)
out_full_te1[[1]] <- rbind(out_full_te1[[1]],
                           'uniform_var' = c(
                             uniform_var['estimate'],
                             sqrt(uniform_var['var'])))
```

The second approach takes asymptotically normal inference about $\delta(w_1, w_2) : \delta^h at(w_1, w_2) = Q^h at(w_1) - Q^h at(w_2)$

```
out_full_te2 <- output_estimates(
  policy0 = best_mtx,
  policy1 = list(optimal_mtx),
  contrasts = "separate",
  gammahat = gammahat,
  probs_array = probs_array,
  floor_decay = 0.8)

out_full_te1[[1]] <- rbind(out_full_te1[[1]],
                           'uniform_var' = c(
                             output_estimates_optimal[[1]]['uniform_var','estimate'] - output_estimates_
                             sqrt(output_estimates_optimal[[1]]['uniform_var','std.error']^2 +
                                    output_estimates_best[[1]]['uniform_var','std.error']^2)
                           ))
```

3. Compare the results

```
# Compare the two approaches for uniform and non_contextual_two_point
comparison_df <- data.frame(method = character(),
                            estimate = numeric(),
                            std_error = numeric(),
                            var = numeric(),
                            contrasts = character(),
                            policy = integer(),
                            stringsAsFactors = FALSE)

# Function to process and append data
process_data <- function(data, policy_num, contrasts) {
  for (method in c("non_contextual_minvar", "contextual_minvar",
                   "non_contextual_stablevar", "contextual_stablevar",
                   "uniform_var")) {
    if (method %in% rownames(data)) {
      row <- data.frame(
        method = method,
        estimate = data[method, "estimate"],
        std_error = data[method, "std.error"],
        var = data[method, "std.error"]^2,
        contrasts = contrasts,
        policy = policy_num,
        stringsAsFactors = FALSE
      )
      comparison_df <<- rbind(comparison_df, row)
```

```r
    }
  }
}

# Process and append data for each subset and condition
process_data(output_estimates_best[[1]], "best", "main effect")
process_data(output_estimates_optimal[[1]], "optimal", "main effect")
process_data(out_full_te1[[1]], "(best,optimal)", "combined")
process_data(out_full_te2[[1]], "(best,optimal)", "separate")


comparison_df <- comparison_df[order(comparison_df$policy, decreasing = TRUE), ]

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

	method	estimate	std_error	var	contrasts	policy
6	non_contextual_minvar	0.4017075	0.0563778	0.0031785	main effect	optimal
7	contextual_minvar	0.5147350	0.0181728	0.0003303	main effect	optimal
8	non_contextual_stablevar	0.4450447	0.0577781	0.0033383	main effect	optimal
9	contextual_stablevar	0.5011886	0.0222223	0.0004938	main effect	optimal
10	uniform_var	0.4646648	0.0569579	0.0032442	main effect	optimal
1	non_contextual_minvar	0.1876929	0.0395924	0.0015676	main effect	best
2	contextual_minvar	0.1950278	0.0261016	0.0006813	main effect	best
3	non_contextual_stablevar	0.1369466	0.0624188	0.0038961	main effect	best
4	contextual_stablevar	0.1656435	0.0367070	0.0013474	main effect	best
5	uniform_var	-0.0529567	0.3033878	0.0920442	main effect	best
11	non_contextual_minvar	0.3175327	0.0794133	0.0063065	combined	(best,optimal)
12	contextual_minvar	0.2869980	0.0545568	0.0029764	combined	(best,optimal)
13	non_contextual_stablevar	0.3751744	0.1153043	0.0132951	combined	(best,optimal)
14	contextual_stablevar	0.3403858	0.0908967	0.0082622	combined	(best,optimal)
15	uniform_var	0.5176216	0.3027359	0.0916490	combined	(best,optimal)
16	non_contextual_minvar	0.2140146	0.0688913	0.0047460	separate	(best,optimal)
17	contextual_minvar	0.3197071	0.0318048	0.0010115	separate	(best,optimal)
18	non_contextual_stablevar	0.3080981	0.0850554	0.0072344	separate	(best,optimal)
19	contextual_stablevar	0.3355451	0.0429096	0.0018412	separate	(best,optimal)