Complex Bootstrap Binary (V2)

Waveley Qiu

2022-02-16

Generating covariates and finding coefficients for covariates in propensity model.

```
set.seed(20220216)

covariate_coef <- function(desired_prop, n, cov_df) {


   alpha_0 = log(desired_prop/(1 - desired_prop))

   coef_L1 <- sample(cov_df$L1, 10000, replace = TRUE)

   coef_L2 <- sample(cov_df$L2, 10000, replace = TRUE)

   coef_L3 <- sample(cov_df$L3, 10000, replace = TRUE)

   A_logit <- vector(mode = "list",length = length(coef_L2))
   p_A <-  vector(mode = "numeric",length = length(coef_L2))

   u <- alpha_0 + coef_L1[1]*cov_df$L1 + coef_L2[1]*cov_df$L2 - coef_L3[1]*cov_df$L3

   p_A[1] <-  mean(exp(u)/(1 + exp(u)))

   tol <- 0.001

   i = 1

   while (abs(p_A[i] - 0.14) > tol) {

     i = i + 1

     A_logit[[i]] <-
       alpha_0 + coef_L1[i]*cov_df$L1 + coef_L2[i]*cov_df$L2 - coef_L3[i]*cov_df$L3

     p_A[i] <- mean(exp(A_logit[[i]])/(1 + exp(A_logit[[i]])))

     if (abs(p_A[i] - 0.14) < tol) {
       mean_treated_proportion <- p_A[i]
       desired_coef_L1 <- coef_L1[i]
       desired_coef_L2 <- coef_L2[i]
       desired_coef_L3 <- coef_L3[i]
     }
```

```
     if (i > length(coef_L2)) {
       stop("You need better coverage, mate.")
    }

   }

   return(tibble(alpha_0,  mean_treated_proportion,
            desired_coef_L1, desired_coef_L2, desired_coef_L3))
}
```

Generating 100 no boot samples

```
seed_vec <- rnorm(100000, mean = 0, sd = 100) %>% round(0) %>% unique()

generate_no_boot_data <- function(n, size = 5000, seeds = seed_vec) {

  df <- list()

  cov_df <- list()

  pb <- progress_bar$new(format = "generating data... [:bar] :percent eta: :eta", total = n)

    for (i in 1:n) {
    pb$tick()
    set.seed(seeds[i])

    set.seed(seeds[i])
    pre_data <- defData(varname = "L1", formula = "0", variance = 1,
                dist = "normal")

    pre_data <- defData(pre_data, varname = "L2", formula = "0", variance = 1,
                dist = "normal")

    pre_data <- defData(pre_data, varname = "L3", formula = "0", variance = 1,
                dist = "normal")

    cov_df[[i]] <- genData(5000, pre_data)

    cov_coef_df <- covariate_coef(0.14, 2, cov_df[[i]])

    L1_coef <- cov_coef_df$desired_coef_L1
    L2_coef <- cov_coef_df$desired_coef_L2
    L3_coef <- cov_coef_df$desired_coef_L3

    pre_data <- defData(pre_data, varname = "L1_coef", formula = L1_coef)
    pre_data <- defData(pre_data, varname = "L2_coef", formula = L2_coef)
    pre_data <- defData(pre_data, varname = "L3_coef", formula = L3_coef)

    pre_data <- defData(pre_data, varname = "A",
                        formula = "-1.815 + L1_coef*L1 + L2_coef*L2", #+ L3_coef*L3",
                    dist = "binary", link = "logit")

    pre_data <- defData(pre_data, varname = "Y",
```

```
                        formula = "0.437 + 0.767*A + 1.39*L2 -0.7*L3" ,
                    dist = "binary", link = "logit")

    df[[i]] <- genData(size, pre_data)
    df[[i]] <- df[[i]] %>% select(-L1_coef, -L2_coef, -L3_coef)
    }
  return(df)
}

###### change to 100 later ###
no_boot_list <- generate_no_boot_data(10)
```

Complex Bootstrap

```
seed_vec <- rnorm(10000, mean = 10000, sd = 100) %>% round(0) %>% unique()

###### change iter to 100 later ###
generate_boots <- function(df, iter = 10, seeds = seed_vec){
  pb3$tick()
  boot_samples <- list()
  matched_boot_df <- list()
  boot_log_or <- tibble()

    for (i in 1:iter) {
      set.seed(seeds[[i]])

      boot_samples[[i]] <- sample_n(df, nrow(df), replace = TRUE)

      matched <- matchit(A ~ L1 + L2 + L3, data = boot_samples[[i]],
                         distance = "glm", link = "logit",
                         method = "nearest", ratio = 1)

      matched_boot_df[[i]] <- match.data(matched, distance = "ps")

      bootmod <- glm(Y ~ A + ps, data = matched_boot_df[[i]],
                     weights = weights, family = binomial)

      sum_bootmod <- summary(bootmod)

      estimate_val <- sum_bootmod$coef[2,1]
      to_bind_rows_estimate <- tibble(estimate = estimate_val)
      boot_log_or <- bind_rows(boot_log_or, to_bind_rows_estimate)
    }

       results <-
       tibble(
         mean_log_odds = mean(boot_log_or$estimate),
         sd_log_odds = sd(boot_log_or$estimate)
       )

  return(results)
}
```

Run Complex Bootstrap

5:54 PM Start 6:15/20 PM End

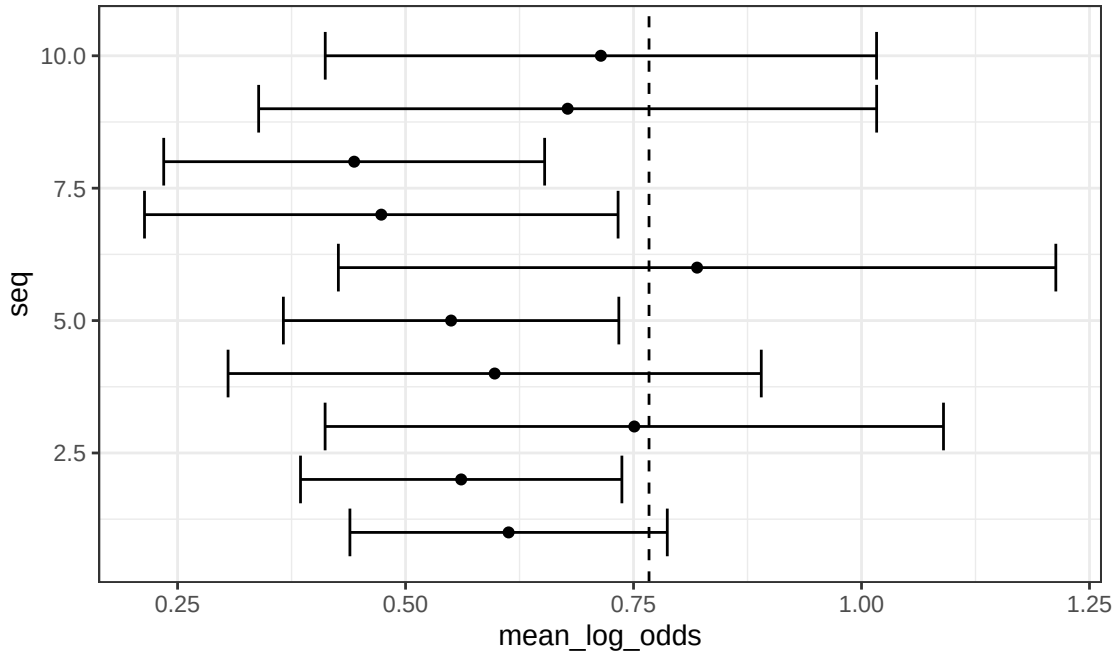
```
nb_tib <- tibble(nb = no_boot_list)

pb3 <- progress_bar$new(format = "bootstrapping... [:bar] :percent eta: :eta", total = nrow(nb_tib))

result_list <- nb_tib %>% mutate(res_tib = map(.x = nb, ~generate_boots(.x)))

fin_estimate_df <- result_list %>% unnest(res_tib) %>% select(-nb)
```

1000 Confidence Intervals Coverage Rate

```
ci_and_coverage <- function(fin_estimates){

    fin_estimates <-
      fin_estimates %>%
      mutate(lower_ci = mean_log_odds - 1.96*sd_log_odds,
             upper_ci = mean_log_odds + 1.96 * sd_log_odds,
             count_true = lower_ci <= 0.767 & 0.767 <= upper_ci)

  return(fin_estimates)
}
complex_coverage <- ci_and_coverage(fin_estimate_df)

coverage_rate <-
    complex_coverage %>%
    select(count_true) %>%
    sum() %>%
    paste0("%")
coverage_rate
```

```
## [1] "6%"
```

```
complex_coverage <- complex_coverage %>%
    mutate(seq = seq(1:nrow(complex_coverage)))

plot = complex_coverage %>%
    ggplot(aes(x = mean_log_odds, y = seq)) +
    geom_point() +
    geom_errorbar(aes(xmin = lower_ci, xmax = upper_ci)) +
    geom_vline(xintercept = 0.767, linetype = "dashed")

plot
```


seq
10.0
7.5
5.0
2.5
0.25
0.50
0.75
1.00
1.25
mean_log_odds