Complex Bootstrap Binary

Hun

2/9/2022

Generating data with true log odds ratio and its standard deviation

```r
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")
pre_data <- defData(pre_data, varname = "A",
                    formula = "0.5*L1 + 0.27*L2 -0.17*L3",
                dist = "binary", link = "logit")
pre_data <- defData(pre_data, varname = "Y",
                    formula = "0.5*A + 0.8*L2 + -0.1*L3",
                dist = "binary", link = "logit")

df <- genData(5000, pre_data)

expit <- function(beta) {
    return(exp(beta)/(1 + exp(beta)))
}

ATE <- expit(sum(0.5 + 0.8*df$L2 - 0.1*df$L3)) - expit(sum(0.8*df$L2 - 0.1*df$L3))
# this is not true ATE

# True log odds ratio: 0.5
```

Complex Bootstrap

```r
nboot <- 1000

boot_samples <- vector(mode = "list", length = nboot)

matched_boot_df <- vector(mode = "list", length = nboot)

results <- vector(mode = "list", length = nboot)

boot_log_OR <- vector(mode = "list", length = nboot)
```

```
boot_se_log_OR <- vector(mode = "list", length = nboot)

boot_CI_log_OR <- vector(mode = "list", length = nboot)

count_true_value <- vector(length = nboot)

boots <- data.frame(id = 1:nboot,
                    se_OR = NA,
                    log_OR = NA)

for (i in 1:nboot) {

  boot_samples[[i]] <- sample_n(df, 1000, replace = FALSE)

  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")

  results[[i]] <- boots

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

  results[[i]]$log_OR <- summary(bootmod)$coeff[2,1]
  results[[i]]$se_OR <- summary(bootmod)$coeff[2,2]

  boot_log_OR[[i]] <- mean(results[[i]]$log_OR)

  boot_se_log_OR[[i]] <- mean(results[[i]]$se_OR)

  boot_CI_log_OR[[i]] <-
    c(boot_log_OR[[i]] - 1.96*boot_se_log_OR[[i]],
       boot_log_OR[[i]] + boot_se_log_OR[[i]])

  count_true_value[i] <-
    between(0.5, range(boot_CI_log_OR[[i]])[1], range(boot_CI_log_OR[[i]])[2])
}
```

Result

```
cat("mean log odds ratio:", mean(unlist(boot_log_OR)))
```

```
## mean log odds ratio: 0.5580097
```

```
cat("se of mean log odds ratio", mean(unlist(boot_se_log_OR)))
```

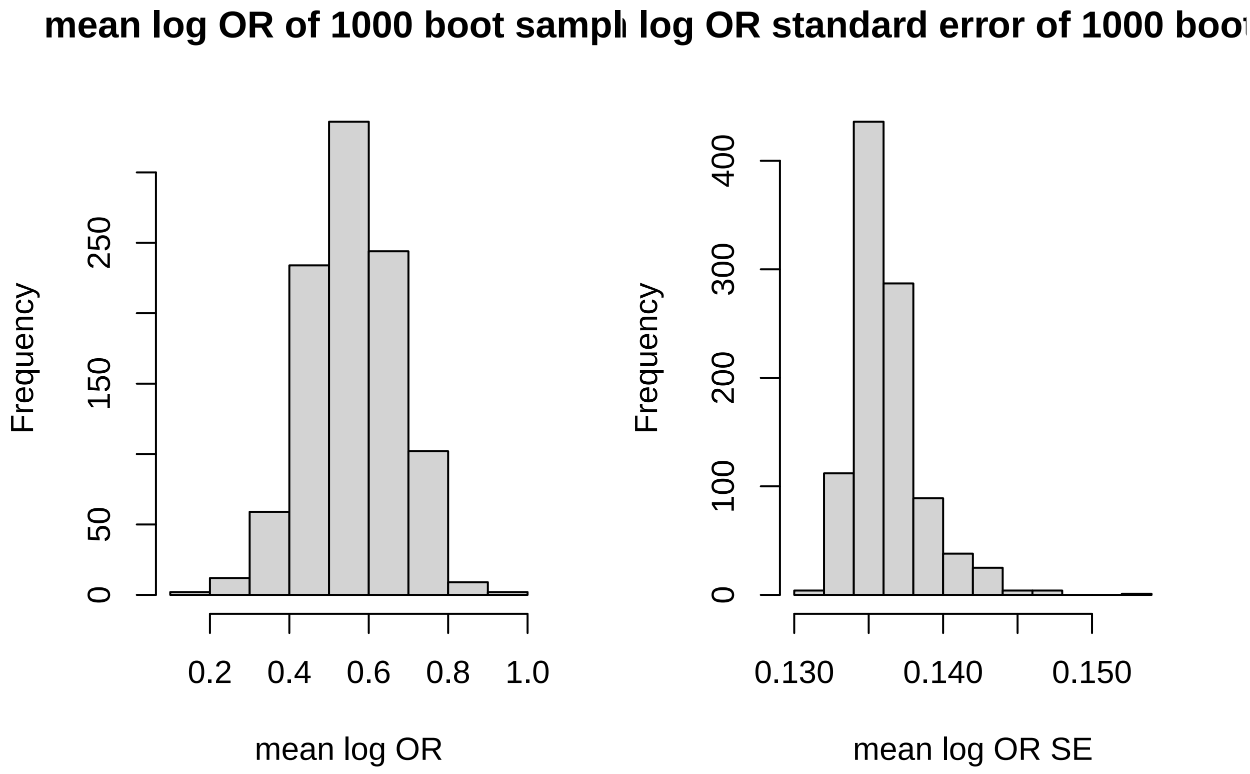
```
## se of mean log odds ratio 0.1362303
```

```r
cat("empirical sd", sd(unlist(boot_log_OR)))
```

```
## empirical sd 0.1151697
```

```r
par(mfrow = c(1,2))
hist(unlist(boot_log_OR), breaks = 10,
     main = " mean log OR of 1000 boot samples", xlab = "mean log OR")

hist(unlist(boot_se_log_OR), breaks = 10,
     main = "mean log OR standard error of 1000 boot samples", xlab = "mean log OR SE")
```



number of bootstrap confidence intervals that contain the true parameter value

```r
# number of bootstrap confidence intervals that contain the true parameter value
tibble(count = count_true_value) %>%
  mutate(count = as.factor(count) %>% fct_relevel("TRUE", "FALSE")) %>%
  group_by(count) %>% count()
```

```
## # A tibble: 2 x 2
## # Groups:   count [2]
##   count     n
##   <fct> <int>
## 1 TRUE    924
## 2 FALSE    76
```