Continuous Simulation

Tucker Morgan - tlm2152

2/16/2022

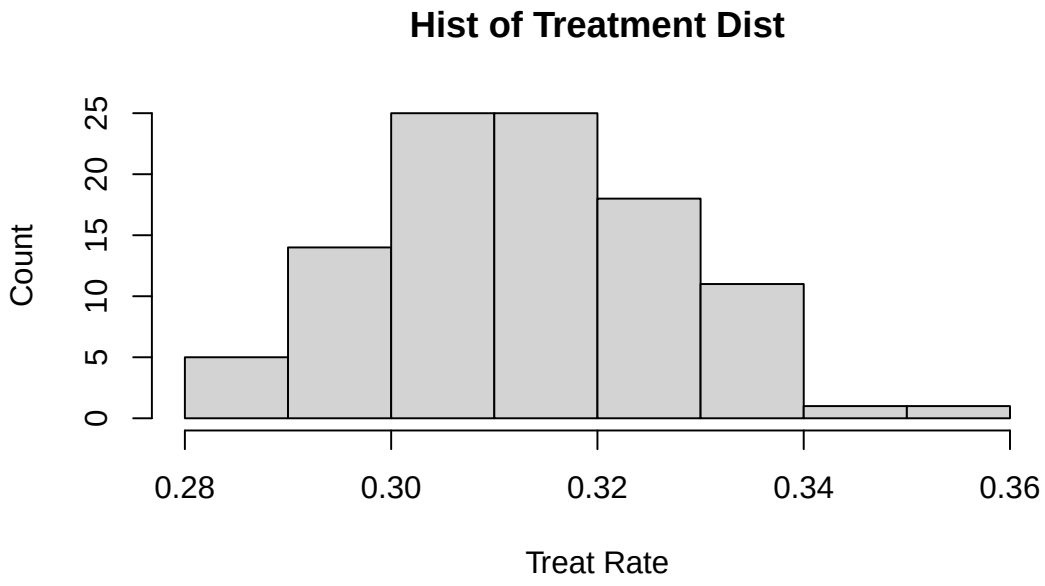
Generating 100 Samples from Our Population

```
source("./shared_code/data_gen_continuous.R")
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

## Loading required package: ggpp

##
## Attaching package: 'ggpp'

## The following object is masked from 'package:ggplot2':
##
##     annotate
```

Let's take a look at one of our "no-boot" aka sub-population data sets.

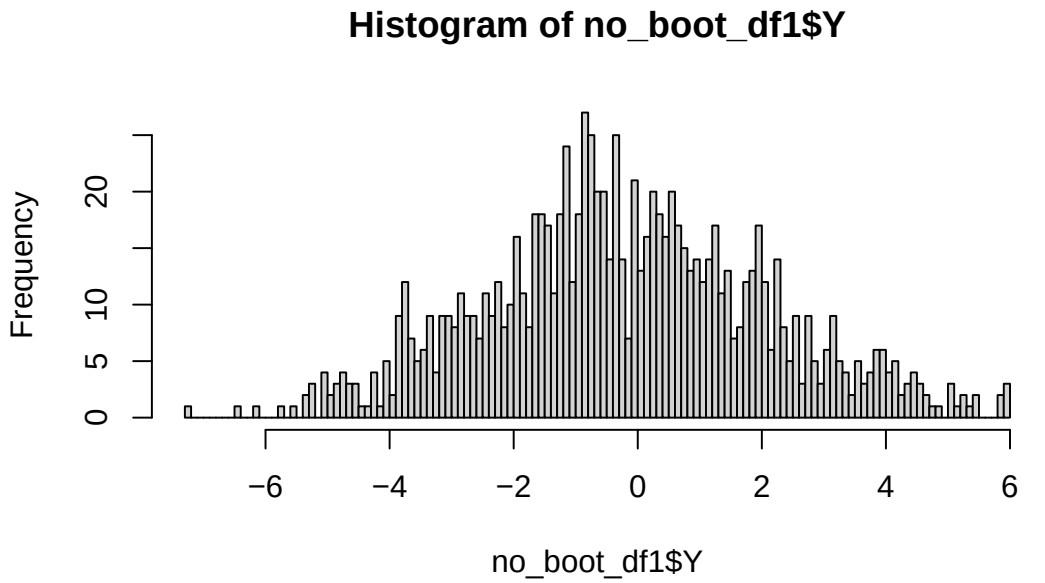
```
hist(map_dbl(1:length(no_boot_list), function(i) mean(no_boot_list[[i]]$A)),
     main = "Hist of Treatment Dist",
     xlab = "Treat Rate",
     ylab = "Count")
```




```
no_boot_df1 <- no_boot_list[[1]]

sum(no_boot_df1$A) / nrow(no_boot_df1) # similar to desired_prop?
```

```
## [1] 0.313
```

```
hist(no_boot_df1$Y, breaks = 100) # continuous distribution of outcome
```

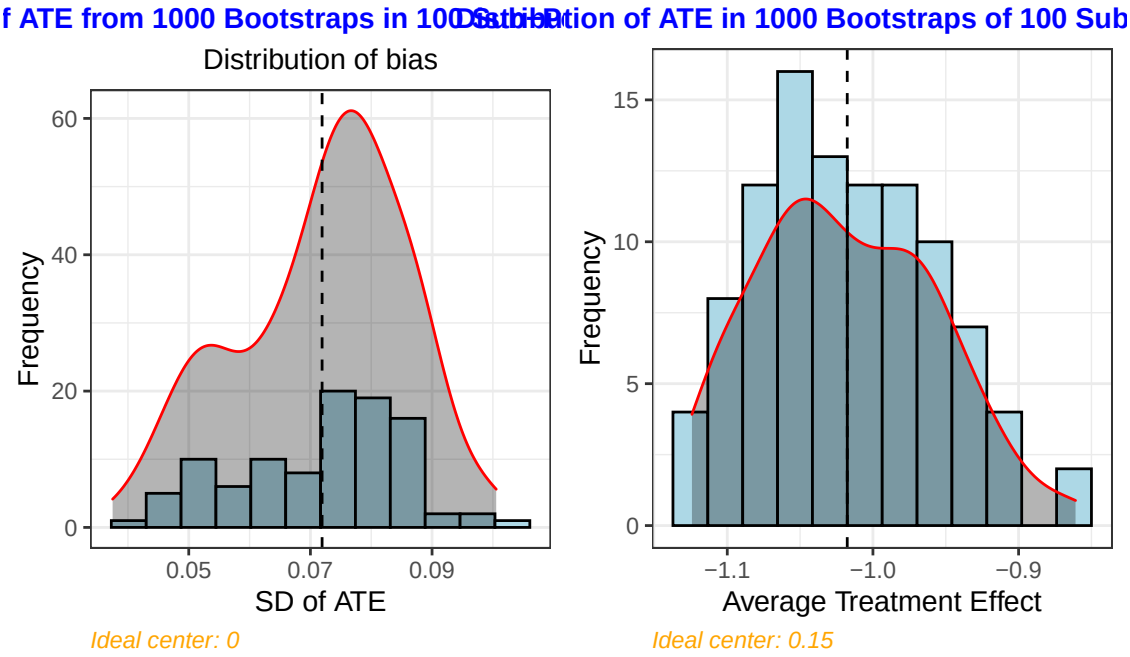


The Simple Bootstrap

```
source("./shared_code/boot_cont_simple.R")
```

Summary of 1000 Simple Bootstraps in 100 Sub-Populations

```
boot_result <-
  boot_estimates %>%
  group_by(seq) %>%
  summarize(ATE = mean(estimate), sd_ATE = sd(estimate))

fig1 <-
  boot_result %>%
  ggplot(aes(x = sd_ATE, color = sd_ATE)) +
  geom_histogram(fill = "light blue",  bins = 12,  color = "black") +
  geom_density(aes(y = ..density..*2), colour = "red",
               fill = "black", alpha = 0.3) +
  geom_vline(xintercept = mean(boot_result$sd_ATE), linetype = "dashed") +
  labs(title = "SD of ATE from 1000 Bootstraps in 100 Sub-Populations",
       subtitle = "Distribution of bias",
  caption = "Ideal center: 0", x = "SD of ATE", y = "Frequency") +
  theme(
  plot.title = element_text(color = "blue", size = 11, face = "bold"),
  plot.subtitle = element_text(color = "black"),
  plot.caption = element_text(color = "orange", face = "italic")
  )


fig2 <-
  boot_result %>%
  ggplot(aes(x = ATE)) +
  geom_histogram(fill = "light blue", bins = 12, color = "black") +
  geom_density(aes(y = ..density..*2), colour = "red",
               fill = "black", alpha = 0.3) +
  geom_vline(xintercept = mean(boot_result$ATE), linetype = "dashed") +
  labs(title = "Distribution of ATE in 1000 Bootstraps of 100 Sub-Populations",
       caption = "Ideal center: 0.15", x = "Average Treatment Effect", y = "Frequency") +
  theme(
  plot.title = element_text(color = "blue", size = 11, face = "bold"),
  plot.caption = element_text(color = "orange", face = "italic")
  )

plot_grid(fig1, fig2)
```




```
rm(boot_estimates, boot_tib, df, matched, matched_df, matched_tib)
```

Running Complex Bootstraps

```
source("./shared_code/boot_cont_complex.R")
```

Confidence Intervals Coverage Rates

Generating Output

```
cont_simple_df <-
  boot_result %>%
  mutate(ci_lower = ATE - 1.96*sd(ATE),
         ci_upper = ATE + 1.96*sd(ATE),
         covered = case_when(
           ci_lower <= beta1 & ci_upper >= beta1 ~ 1,
                                              TRUE ~ 0
         ),
         boot_type = c(0),
         scenario_id = c(6), # this needs to be updated each run
         empirical_mean = continuous_empirical_mean_se$empircal_mean,
         empirical_se = continuous_empirical_mean_se$empircal_se) %>%
  tibble()

cont_complex_df <-
  fin_estimate_df %>%
  mutate(ci_lower = ATE - 1.96*sd(ATE),
```

```
        ci_upper = ATE + 1.96*sd(ATE),
        covered = case_when(
          ci_lower <= beta1 & ci_upper >= beta1 ~ 1,
                                          TRUE ~ 0
        ),
        boot_type = c(1),
        scenario_id = c(6), # this needs to be updated each run
        empirical_mean = continuous_empirical_mean_se$empircal_mean,
        empirical_se = continuous_empirical_mean_se$empircal_se) %>%
  tibble() %>%
  relocate(seq)

cont_df_scen_6 <- bind_rows(cont_simple_df, cont_complex_df) # change df name each run


save(cont_df_scen_6, file = "./output_data/continuous_scen_6.RData") # change object name and file name
```