problemset6

junyi z

GitHub

https://github.com/juny1z/Problemset6.git

a

```
library(DBI)
library(parallel)
library(future)
library(furrr)
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
lahman <- dbConnect(RSQLite::SQLite(), "/Users/zjyyy/Desktop/lahman_1871-2022.sqlite")

fielding <- dbGetQuery(lahman, "SELECT * FROM Fielding") %>%
  mutate(RF = 3 * (PO + A) / InnOuts) %>%
  filter(!is.na(RF) & InnOuts > 0 & !is.na(teamID))

sta_func <- function(data) {
  data %>%
```

```
    group_by(teamID) %>%
    summarise(mean_RF = mean(RF, na.rm = TRUE))
}

stratified_bootstrap <- function(data, group_col, sta_func, n_bootstrap = 1000) {
  replicate(n_bootstrap, {
    sampled_data <- data %>%
      group_by(!!sym(group_col)) %>%
      sample_frac(replace = TRUE) %>%
      ungroup()
    if (nrow(sampled_data) == 0) {
      return(data.frame(teamID = NA, mean_RF = NA))
    }
    sta_func(sampled_data)
  }, simplify = FALSE)
}

# Without any parallel processing
set.seed(123)
bootstrap <- stratified_bootstrap(fielding, "teamID", sta_func, n_bootstrap = 1000)

bootstrap_sum <- bind_rows(bootstrap) %>%
  group_by(teamID) %>%
  summarise(
    mean_RF = mean(mean_RF, na.rm = TRUE),
    se_RF = sd(mean_RF, na.rm = TRUE)
  ) %>%
  mutate(se_RF = ifelse(is.na(se_RF), 0, se_RF)) %>%
  arrange(desc(mean_RF)) %>%
  slice_head(n = 10)

print(bootstrap_sum)
```

```
# A tibble: 10 x 3
   teamID mean_RF se_RF
   <chr>    <dbl> <dbl>
 1 RC1      0.571     0
 2 LS1      0.533     0
 3 ELI      0.525     0
 4 MLU      0.516     0
 5 KEO      0.512     0
 6 RIC      0.508     0
```

```
 7 BLA      0.495     0
 8 LS3      0.489     0
 9 TRN      0.481     0
10 PHU      0.478     0
```

```r
# Parallel
cl <- makeCluster(detectCores() - 1)
clusterExport(cl, varlist = c("fielding", "stratified_bootstrap", "sta_func"), envir = envir
clusterEvalQ(cl, library(dplyr))
```

```
[[1]]
[1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
[7] "methods"   "base"

[[2]]
[1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
[7] "methods"   "base"

[[3]]
[1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
[7] "methods"   "base"

[[4]]
[1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
[7] "methods"   "base"

[[5]]
[1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
[7] "methods"   "base"

[[6]]
[1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
[7] "methods"   "base"

[[7]]
[1] "dplyr"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
[7] "methods"   "base"
```

```r
set.seed(123)
bootstrap_parallel <- parLapply(cl, 1:1000, function(i) {
  stratified_bootstrap(fielding, "teamID", sta_func, n_bootstrap = 1)
})
```

```r
stopCluster(cl)

bootstrap_sum_parallel <- bind_rows(bootstrap_parallel) %>%
  group_by(teamID) %>%
  summarise(
    mean_RF = mean(mean_RF, na.rm = TRUE),
    se_RF = ifelse(n_distinct(mean_RF) > 1, sd(mean_RF, na.rm = TRUE), 0)
  ) %>%
  mutate(se_RF = ifelse(is.na(se_RF), 0, se_RF)) %>%
  arrange(desc(mean_RF)) %>%
  slice_head(n = 10)

print("Results with Parallel Package:")
```

```
[1] "Results with Parallel Package:"
```

```r
print(bootstrap_sum_parallel)
```

```
# A tibble: 10 x 3
   teamID mean_RF se_RF
   <chr>    <dbl> <dbl>
 1 RC1      0.571     0
 2 LS1      0.529     0
 3 ELI      0.526     0
 4 MLU      0.514     0
 5 RIC      0.510     0
 6 KEO      0.509     0
 7 BLA      0.496     0
 8 LS3      0.489     0
 9 PHU      0.482     0
10 TRN      0.478     0
```

```r
# Future
plan(multisession)

set.seed(123)
bootstrap_future <- future_map(1:1000, ~ {
  sampled_data <- fielding %>%
    group_by(teamID) %>%
    sample_frac(replace = TRUE) %>%
```

```
    ungroup()
  sta_func(sampled_data)
}, .options = furrr_options(seed = TRUE))

bootstrap_sum_future <- bind_rows(bind_rows(bootstrap_future)) %>%
  group_by(teamID) %>%
  summarise(
    mean_RF = mean(mean_RF, na.rm = TRUE),
    se_RF = sd(mean_RF, na.rm = TRUE)
  ) %>%
  mutate(se_RF = ifelse(is.na(se_RF), 0, se_RF)) %>%
  arrange(desc(mean_RF)) %>%
  slice_head(n = 10)

print("Results with Future Package:")
```

```
[1] "Results with Future Package:"
```

```
print(bootstrap_sum_future)
```

```
# A tibble: 10 x 3
   teamID mean_RF se_RF
   <chr>    <dbl> <dbl>
 1 RC1      0.574     0
 2 LS1      0.532     0
 3 ELI      0.523     0
 4 RIC      0.514     0
 5 KEO      0.506     0
 6 MLU      0.503     0
 7 BLA      0.495     0
 8 LS3      0.489     0
 9 PHU      0.481     0
10 TRN      0.480     0
```

b

```
final_table <- bootstrap_sum %>%
  rename(mean_RF_no_parallel = mean_RF, se_RF_no_parallel = se_RF) %>%
  left_join(bootstrap_sum_parallel %>%
              rename(mean_RF_parallel = mean_RF, se_RF_parallel = se_RF),
```

```
            by = "teamID"
  ) %>%
  left_join(bootstrap_sum_future %>%
              rename(mean_RF_future = mean_RF, se_RF_future = se_RF),
            by = "teamID"
  ) %>%
  arrange(desc(mean_RF_no_parallel))

print(final_table)
```

```
# A tibble: 10 x 7
   teamID mean_RF_no_parallel se_RF_no_parallel mean_RF_parallel se_RF_parallel
   <chr>                <dbl>             <dbl>            <dbl>          <dbl>
 1 RC1                  0.571                 0            0.571              0
 2 LS1                  0.533                 0            0.529              0
 3 ELI                  0.525                 0            0.526              0
 4 MLU                  0.516                 0            0.514              0
 5 KEO                  0.512                 0            0.509              0
 6 RIC                  0.508                 0            0.510              0
 7 BLA                  0.495                 0            0.496              0
 8 LS3                  0.489                 0            0.489              0
 9 TRN                  0.481                 0            0.478              0
10 PHU                  0.478                 0            0.482              0
# i 2 more variables: mean_RF_future <dbl>, se_RF_future <dbl>
```

c

The mean_RF values differ slightly across the three versions, which is expected due to slight differences in the bootstrap.

Non-Parallel is the slowest since it performs all bootstrap iterations sequentially

Future is comparable to the Parallel, but slightly more flexible due to the future package.