STATS 506 Problem Set #6

Haiming Li

Stratified Bootstrap

```r
# get data
lahman <- dbConnect(RSQLite::SQLite(), "./lahman_1871-2022.sqlite")
data <- na.omit(dbGetQuery(lahman, "SELECT teamID, PO, A, InnOuts FROM Fielding"))
dbDisconnect(lahman)
```

a. Here's the implementation without parallel processing.

```r
system.time({
# create a list that contain individual team data
teams <- unique(data$teamID)
teams_data <- vector('list', length = length(unique(data$teamID)))
for (i in 1:length(teams)) {
  teams_data[[i]] <- data[data$teamID == teams[i],]
}

# creat bootstrap sample
n_iter <- 1000
samples_naive <- vector('list', n_iter)

for (i in 1:n_iter) {
  sample_i <- NULL
  # helper function to sample each team
  sample_team <- function(team_data) {
    team_sample <- team_data[sample(1:nrow(team_data),
                                    size = nrow(team_data), replace = TRUE), ]
  }
  samples_naive[[i]] <- Reduce(rbind, lapply(teams_data, sample_team))
}})
```

```
   user  system elapsed 
582.803  23.706 607.130 
```

Here's the implementation with parallel package

```r
system.time({
# team data
teams <- unique(data$teamID)
teams_data <- split(data, data$teamID)

n_iter <- 1000
# helper function for one bootstrap sample
bootstrap_sample <- function(iter) {
    sample_team <- function(team_data) {
      team_data[sample(1:nrow(team_data), size = nrow(team_data), replace = TRUE), ]
    }
    do.call(rbind, lapply(teams_data, sample_team))
}

samples_parallel <- mclapply(1:n_iter, bootstrap_sample, mc.cores = 6)
})
```

```
   user  system elapsed 
118.764  67.917 232.129 
```

Here's the implementation with future package

```r
system.time({
  # Create a list containing individual team data
  teams <- unique(data$teamID)
  teams_data <- split(data, data$teamID)

  plan(multisession, workers = 6)

  # helper function for one bootstrap sample
  n_iter <- 1000
  bootstrap_sample <- function(iter) {
    sample_team <- function(team_data) {
      team_data[sample(1:nrow(team_data), size = nrow(team_data), replace = TRUE), ]
    }
    do.call(rbind, lapply(teams_data, sample_team))
  }
  future_list <- lapply(1:n_iter, function(x) future(bootstrap_sample(), seed = NULL))
  samples_future <- lapply(future_list, value)
})
```

```
   user  system elapsed 
 75.306  12.104 156.030 
```

b. Here’s the estimation of RF and its standard error for each method.

```
# helper function calculate RF for single sample
RF <- function(sample) {
  sample$RF <- 3 * (sample$PO + sample$A) / sample$InnOuts
  res <- aggregate(RF ~ teamID, data = sample, FUN = mean)
  res$RF[is.infinite(res$RF)] <- 0 # fix division by 0
  return(res)
}

# calculate mean RF and SE for a list of samples
calc_metrics <- function(samples) {
  res <- mclapply(samples, RF, mc.cores = 6)
  combined_df <- do.call(rbind, res)
  result <- aggregate(RF ~ teamID, data = combined_df,
                        FUN = function(x) c(mean = mean(x),
                                            se = sd(x) / sqrt(length(x))))
  return(result)
}

# calculate metric for each method and combine them
res_naive <- calc_metrics(samples_naive)
res_parallel <- calc_metrics(samples_parallel)
res_future <- calc_metrics(samples_future)

# show results
res_naive %>%
  inner_join(res_parallel, by = 'teamID') %>%
  inner_join(res_future, by = 'teamID') %>%
  mutate(
    RF_naive = RF.x[, 1],
    SE_naive = RF.x[, 2],
    RF_parallel = RF.y[, 1],
    SE_parallel = RF.y[, 2],
    RF_future = RF[, 1],
    SE_future = RF[, 2]
  ) %>%
  select(teamID, RF_naive, SE_naive, RF_parallel,
         SE_parallel, RF_future, SE_future)
```

```
    teamID    RF_naive     SE_naive RF_parallel  SE_parallel   RF_future
1      ALT 0.388147797 0.0015805836 0.386569748 0.0015719210 0.384937764
2      ANA 0.414634671 0.0004827884 0.414935846 0.0005056575 0.415586604
3      ARI 0.366117474 0.0002727785 0.366205439 0.0002659045 0.365457654
4      ATL 0.044552301 0.0039108238 0.051102856 0.0041466716 0.057996647
5      BAL 0.145713028 0.0060037937 0.137770253 0.0059280256 0.139318899
6      BFN 0.451180900 0.0006641044 0.451352532 0.0006422091 0.452239878
7      BFP 0.462553920 0.0016906170 0.462798441 0.0017022874 0.460468358
8      BL1 0.442787724 0.0009222230 0.442976054 0.0009174196 0.441751886
9      BL2 0.401170078 0.0005125457 0.400209532 0.0004968817 0.401762984
10     BL3 0.442888725 0.0012366641 0.445644107 0.0012695297 0.444553319
11     BL4 0.431758327 0.0018438224 0.430938933 0.0018022627 0.431026391
12     BLA 0.494286452 0.0009885552 0.493698734 0.0010132467 0.494646579
13     BLN 0.449848768 0.0005962324 0.448765219 0.0005853419 0.449715793
14     BLU 0.385968398 0.0013423627 0.384457303 0.0013229197 0.386611899
15     BOS 0.402228549 0.0001751654 0.401884432 0.0001685651 0.401756119
16     BR1 0.438941776 0.0013867741 0.437878166 0.0013935502 0.437430009
17     BR2 0.400260059 0.0008536512 0.399663272 0.0008216315 0.398824447
18     BR3 0.448649594 0.0007198231 0.446566614 0.0007195582 0.447498678
19     BR4 0.438887240 0.0014884695 0.438223429 0.0014775705 0.438223545
20     BRO 0.442689189 0.0003157864 0.442707537 0.0003075404 0.442205176
21     BRP 0.436253017 0.0017633761 0.431796555 0.0017121776 0.433678585
22     BS1 0.451287808 0.0010019729 0.453535769 0.0009851510 0.451543297
23     BS2 0.433245030 0.0015358363 0.431753196 0.0015731683 0.433157307
24     BSN 0.456582020 0.0002848271 0.455553405 0.0002856983 0.456138917
25     BSP 0.430171699 0.0014618851 0.431023150 0.0014847677 0.431805873
26     BSU 0.420330240 0.0017875212 0.418025161 0.0017819572 0.418080016
27     CAL 0.415739991 0.0002630245 0.416315225 0.0002707624 0.416345511
28     CH1 0.427009106 0.0019378762 0.430923647 0.0019023090 0.427791820
29     CH2 0.435167119 0.0011604554 0.437987827 0.0011364912 0.437497188
30     CHA 0.001611198 0.0008045138 0.001207017 0.0006962548 0.002425074
31     CHN 0.415098589 0.0001466312 0.415307264 0.0001502214 0.414973831
32     CHP 0.475247080 0.0016376483 0.474792154 0.0016771569 0.477124178
33     CHU 0.340955345 0.0011688315 0.344217947 0.0011879486 0.344463024
34     CIN 0.139518096 0.0060693750 0.155299572 0.0062107973 0.142805376
35     CL1 0.383170398 0.0012989664 0.384125145 0.0012492688 0.385347027
```

36	CL2	0.434311410	0.0007249062	0.436059346	0.0007347618	0.434239334
37	CL3	0.459186914	0.0009171559	0.457526125	0.0008871330	0.461618817
38	CL4	0.163556175	0.0068028729	0.161614179	0.0067800415	0.170180301
39	CL5	0.441272758	0.0014165827	0.437446656	0.0013674092	0.439545443
40	CL6	0.410989388	0.0009033388	0.412265833	0.0008988428	0.411867185
41	CLE	0.143145806	0.0060789589	0.139464983	0.0060406768	0.146117326
42	CLP	0.407749210	0.0014502498	0.403887062	0.0013885915	0.407161296
43	CN1	0.441029755	0.0007278405	0.441266810	0.0007411536	0.440906594
44	CN2	0.438932451	0.0006588940	0.438951017	0.0006646419	0.439840541
45	CN3	0.435648031	0.0015758785	0.435423579	0.0015867134	0.438410188
46	CNU	0.443663083	0.0016362167	0.445622175	0.0016609354	0.444996271
47	COL	0.389571404	0.0002506015	0.389867221	0.0002458504	0.389542564
48	DET	0.153766478	0.0062414315	0.154797084	0.0062566496	0.144059213
49	DTN	0.421815234	0.0005580180	0.422221159	0.0005399143	0.422108425
50	ELI	0.522970273	0.0020444707	0.525561374	0.0020033405	0.528360598
51	FLO	0.370655991	0.0003184452	0.370261623	0.0003206898	0.370817319
52	FW1	0.442791702	0.0016715409	0.441518082	0.0015321116	0.442198665
53	HAR	0.420017762	0.0014091111	0.417379583	0.0014255633	0.415591341
54	HOU	0.401518878	0.0001931155	0.401626576	0.0001961500	0.401327988
55	HR1	0.381691427	0.0011635158	0.384967305	0.0011670206	0.384023188
56	IN1	0.409852049	0.0021261335	0.408587539	0.0021188698	0.410826965
57	IN2	0.461118208	0.0014781249	0.462257716	0.0014904594	0.460740327
58	IN3	0.418744875	0.0009796115	0.420277210	0.0010057879	0.420325599
59	KC1	0.407397594	0.0004146733	0.407810299	0.0004064443	0.407509848
60	KC2	0.380874918	0.0008064784	0.379935303	0.0008305917	0.380365894
61	KCA	0.390700969	0.0001901893	0.390624716	0.0001822241	0.390938302
62	KCN	0.439719979	0.0015029292	0.436409819	0.0016298926	0.439746992
63	KCU	0.418852750	0.0009448216	0.417243848	0.0009770939	0.417248382
64	KEO	0.510557079	0.0034488607	0.507671871	0.0035293405	0.508829098
65	LAA	0.376604074	0.0003055422	0.376423076	0.0002780240	0.376679847
66	LAN	0.138531717	0.0059607747	0.143660795	0.0060089983	0.146028711
67	LS1	0.531076253	0.0018026833	0.531130355	0.0017825294	0.530415244
68	LS2	0.165844944	0.0071572524	0.166339354	0.0071474951	0.178987185
69	LS3	0.489182650	0.0004911719	0.489511015	0.0004888953	0.489272210
70	MIA	0.362997842	0.0003904545	0.364265660	0.0003634978	0.364069910
71	MID	0.460097732	0.0019289944	0.457684868	0.0019029034	0.460067793
72	MIL	0.359087721	0.0002663960	0.358822345	0.0002678587	0.358854323
73	MIN	0.391434790	0.0001787514	0.391539191	0.0001883965	0.391787744
74	ML1	0.440919288	0.0004559475	0.441267840	0.0004804980	0.441415407
75	ML2	0.439962998	0.0017390310	0.436370921	0.0017107225	0.437639688
76	ML3	0.465211696	0.0020266964	0.462916210	0.0019582849	0.462602327
77	ML4	0.054550703	0.0044269724	0.057022465	0.0045100872	0.052476593
78	MLA	0.418469808	0.0012272150	0.417752931	0.0012295128	0.419622529

79	MLU	0.514759111	0.0039458586	0.519016201	0.0040626398	0.510493041
80	MON	0.063020680	0.0047289895	0.056309192	0.0045107357	0.054949432
81	NH1	0.440429212	0.0015530255	0.440053633	0.0015462532	0.440589666
82	NY1	0.439906965	0.0002849201	0.440356929	0.0002919683	0.440363648
83	NY2	0.389746049	0.0007040009	0.390714634	0.0007380635	0.390557265
84	NY3	0.379808606	0.0012566960	0.379636879	0.0012690857	0.380421429
85	NY4	0.171095198	0.0068463699	0.167347442	0.0068080688	0.155995754
86	NYA	0.403748862	0.0001637129	0.403938348	0.0001652251	0.403960666
87	NYN	0.402203087	0.0001740611	0.402626822	0.0001726361	0.402395811
88	NYP	0.422247746	0.0014684894	0.423649022	0.0015749729	0.423348803
89	OAK	0.018086164	0.0026364172	0.020519708	0.0028010060	0.018552259
90	PH1	0.453764211	0.0008424148	0.452324783	0.0008805824	0.452722504
91	PH2	0.463048888	0.0010557921	0.464547817	0.0010001373	0.464946588
92	PH3	0.461789949	0.0027356533	0.466059203	0.0028243131	0.469033464
93	PH4	0.412756383	0.0004822260	0.411927644	0.0004929563	0.412785038
94	PHA	0.168127777	0.0070848220	0.176512559	0.0071721019	0.175053356
95	PHI	0.401178769	0.0001469971	0.400958043	0.0001482305	0.400661152
96	PHN	0.451288105	0.0012590264	0.452419711	0.0012974957	0.453358955
97	PHP	0.439086814	0.0017554849	0.442610402	0.0018029544	0.437999009
98	PHU	0.482504405	0.0014551411	0.479088463	0.0014904468	0.479093158
99	PIT	0.410946740	0.0001516630	0.411126004	0.0001483754	0.410927920
100	PRO	0.464932881	0.0007328814	0.465437425	0.0007165378	0.463489846
101	PT1	0.442426700	0.0007189805	0.441818053	0.0007154973	0.442746549
102	PTP	0.448922232	0.0016637974	0.447089248	0.0017310767	0.449255175
103	RC1	0.566828323	0.0025780863	0.573429196	0.0025292446	0.571963585
104	RC2	0.477613711	0.0014470927	0.474910669	0.0014069416	0.473979932
105	RIC	0.510682531	0.0021956031	0.508276888	0.0021734956	0.507398383
106	SDN	0.393893460	0.0001908377	0.394235585	0.0001844594	0.394139324
107	SE1	0.416456242	0.0014208973	0.417625366	0.0013694666	0.417284333
108	SEA	0.143831524	0.0059140408	0.139237812	0.0058622638	0.144356964
109	SFN	0.401462660	0.0001803657	0.401540149	0.0001749290	0.401370580
110	SL1	0.335031540	0.0019445241	0.337800050	0.0020890870	0.335696195
111	SL2	0.393134303	0.0018536066	0.395174994	0.0018700176	0.390330586
112	SL3	0.398938755	0.0012991425	0.397247978	0.0012598491	0.398633333
113	SL4	0.436370917	0.0005355294	0.435788966	0.0005239712	0.436327508
114	SL5	0.375127182	0.0009571660	0.376299523	0.0009458689	0.375248052
115	SLA	0.465615888	0.0005438720	0.465682141	0.0005553177	0.464813066
116	SLN	0.150696708	0.0063715998	0.156944691	0.0064249150	0.165402471
117	SLU	0.393818265	0.0012717536	0.392285855	0.0012379448	0.393244511
118	SPU	0.467452454	0.0028206936	0.460625915	0.0028561836	0.468257018
119	SR1	0.463567042	0.0012133601	0.465559970	0.0011918222	0.465784432
120	SR2	0.414595282	0.0012725643	0.412689274	0.0011957503	0.412068826
121	TBA	0.356118382	0.0002580922	0.356687255	0.0002492044	0.356367094

```
122    TEX 0.149511302 0.0059420858 0.148001324 0.0059319046 0.143333145
123    TL1 0.456216489 0.0015391882 0.455613255 0.0014728178 0.457311608
124    TL2 0.419888848 0.0019131293 0.420993376 0.0019756380 0.420106148
125    TOR 0.375358334 0.0002090845 0.375156104 0.0002010477 0.375272753
126    TRN 0.479100238 0.0009620704 0.480896308 0.0009326311 0.481976039
127    TRO 0.471595423 0.0012887659 0.472707272 0.0012955890 0.471330253
128    WAS 0.373788699 0.0002694745 0.373994317 0.0002566868 0.373790041
129    WIL 0.379985382 0.0018600884 0.383462703 0.0020149974 0.382572542
130    WOR 0.425335179 0.0010960064 0.424715660 0.0011280986 0.424096556
131    WS1 0.463507193 0.0003801142 0.463748162 0.0003820761 0.463832892
132    WS2 0.398765359 0.0004163956 0.398785385 0.0004246963 0.399282803
133    WS3 0.408211096 0.0014030607 0.411512093 0.0014283056 0.411207624
134    WS4 0.468551346 0.0018504198 0.463208633 0.0018402766 0.466447344
135    WS5 0.408139132 0.0016109950 0.410451703 0.0016378906 0.405803704
136    WS6 0.413513130 0.0017889585 0.415937447 0.0018883581 0.414354774
137    WS7 0.451885631 0.0015477993 0.447449578 0.0014991446 0.449913885
138    WS8 0.447771397 0.0006629957 0.448469664 0.0006742009 0.448577812
139    WS9 0.438407493 0.0011466583 0.435978647 0.0011811544 0.438908410
140    WSU 0.401292426 0.0011042798 0.401915861 0.0011088763 0.400835879
       SE_future
1   0.0015716854
2   0.0004963227
3   0.0002752415
4   0.0043859460
5   0.0059423034
6   0.0006448398
7   0.0017054557
8   0.0009754416
9   0.0005166048
10  0.0012796787
11  0.0018779383
12  0.0009933906
13  0.0005937806
14  0.0013941427
15  0.0001690022
16  0.0013588653
17  0.0008071216
18  0.0007346303
19  0.0014324223
20  0.0003129371
21  0.0017489520
22  0.0010023004
23  0.0015026039
```

```
24  0.0002800006
25  0.0014876831
26  0.0017797480
27  0.0002688440
28  0.0018238037
29  0.0011154364
30  0.0009875968
31  0.0001499874
32  0.0016807452
33  0.0011641454
34  0.0061041781
35  0.0012006531
36  0.0007424918
37  0.0009611729
38  0.0068570952
39  0.0013748641
40  0.0008712087
41  0.0060984511
42  0.0014535650
43  0.0007375390
44  0.0006413065
45  0.0015815588
46  0.0015613803
47  0.0002528647
48  0.0061579250
49  0.0005343788
50  0.0019493703
51  0.0003157221
52  0.0015464034
53  0.0014701754
54  0.0001939409
55  0.0012170624
56  0.0020046507
57  0.0014460564
58  0.0009883886
59  0.0004032481
60  0.0008584414
61  0.0001890015
62  0.0015732908
63  0.0009472926
64  0.0035071388
65  0.0002967478
66  0.0060297312
```

```
67  0.0018293596
68  0.0072557846
69  0.0005037560
70  0.0003827923
71  0.0018674118
72  0.0002526660
73  0.0001853164
74  0.0004670498
75  0.0017452375
76  0.0019909004
77  0.0043737223
78  0.0011916813
79  0.0039956427
80  0.0044592268
81  0.0015700514
82  0.0002862942
83  0.0007271510
84  0.0012494257
85  0.0066980754
86  0.0001731095
87  0.0001790483
88  0.0015543503
89  0.0026735535
90  0.0008544844
91  0.0010329119
92  0.0027902892
93  0.0004832898
94  0.0071279811
95  0.0001565178
96  0.0012789933
97  0.0017670301
98  0.0015498459
99  0.0001509398
100 0.0007296715
101 0.0007077996
102 0.0016542476
103 0.0025704475
104 0.0014308918
105 0.0021536839
106 0.0001872844
107 0.0014170900
108 0.0059103453
109 0.0001743144
```

```
110 0.0020455221
111 0.0018060605
112 0.0012353575
113 0.0005052588
114 0.0009180567
115 0.0005311854
116 0.0064907726
117 0.0012364435
118 0.0028102149
119 0.0012842388
120 0.0012748290
121 0.0002526794
122 0.0058933939
123 0.0014979103
124 0.0019683895
125 0.0002081378
126 0.0009599182
127 0.0013121845
128 0.0002448549
129 0.0019313651
130 0.0011386809
131 0.0003964981
132 0.0004242206
133 0.0014273171
134 0.0018280793
135 0.0016239184
136 0.0017364279
137 0.0015052002
138 0.0006857866
139 0.0012062756
140 0.0011126284
```

c. As shown by the timing in part a, the naive approach is of course the slowest as it does not have any parallel processing. The implementation with future package seems to be faster than the implementation with parallel package. It might due to the fact that future package utilize concurrent programming. For this particular task, being able to dynamically schedule resources would improve the performance.

Link to GitHub

- GitHub Repo of this Pset