Oversampling with sample function

Asked 1 year, 4 months ago Active 1 year, 4 months ago Viewed 196 times



I would like to create a mtcars dataset where all cylinders have 100 observations. For that, I would sample with replacement the existing observations.

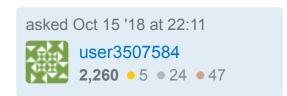
I have tried the following code that, for some reason, does not produce 300 observations.



```
library(data.table)
mtcars <- data.table(mtcars)</pre>
resampling <- list()</pre>
set.seed(3)
cyl <- sort(unique(as.character(mtcars$cyl)))</pre>
for (i in 1:length(cyl)){
  min_obs_cyl <- 100
  dat_cyl <- mtcars[cyl == as.numeric(cyl[i]) ]</pre>
  resampling[[ cyl[i] ]] <- dat_cyl[sample(1:nrow(dat_cyl),</pre>
                                                  size = (min_obs_cyl - nrow(mtcars[cyl ==
cyl[i] ])),
                                                      replace = T),
}
resampling_df <- do.call("rbind", resampling)</pre>
mtcars_oversample <- rbind(mtcars, resampling_df)</pre>
```

I get 307 observations. Anyone knows what I am doing wrong?

sample resampling



3 Answers



I think in this case, you can do the the sampling within groups using data.table 's by= functionality. sample from the .I row counter within each cyl group, and then use this

```
mtcars[mtcars[, sample(.I, 100, replace=TRUE), by=cyl]$V1,]
      mpg cyl disp hp drat wt qsec vs am gear carb
  1: 18.1
           6 225.0 105 2.76 3.460 20.22 1
  2: 17.8
            6 167.6 123 3.92 3.440 18.90 1
                                                    4
 3: 19.2 6 167.6 123 3.92 3.440 18.30 1 0
                                                    4
 4: 19.2
           6 167.6 123 3.92 3.440 18.30 1 0
                                                   4
# 5: 21.0 6 160.0 110 3.90 2.620 16.46 0 1
                                                   4
#296: 15.5
           8 318.0 150 2.76 3.520 16.87 0 0
#297: 19.2 8 400.0 175 3.08 3.845 17.05 0 0
                                                   2
#298: 19.2 8 400.0 175 3.08 3.845 17.05
                                                   2
                                               3
#299: 14.3 8 360.0 245 3.21 3.570 15.84 0 0
                                               3
                                                   4
#300: 15.2
           8 275.8 180 3.07 3.780 18.00 0 0
                                                   3
```

If you need to specify different counts for each group, here's one option. The special object stores the value of the by= argument as a list.

```
grpcnt <- setNames(c(50,100,70), unique(mtcars$cyl))
# 6    4    8
# 50    100    70
mtcars[mtcars[, sample(.I, grpcnt[as.character(.BY[[1]])], replace=TRUE), by=cyl]$V1]</pre>
```

edited Oct 16 '18 at 22:47

answered Oct 15 '18 at 22:26



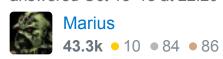
I think this is the best and fastest solution. However, I didn't mention that I would need a different number of observations per cyl group. I guess I would use this solution stackoverflow.com/questions/33495916/... – user3507584 Oct 16 '18 at 6:38

@user3507584 - see my edit for how to adapt different group sizes to this sort of solution. – thelatemail Oct 16 '18 at 22:47

For an alternative solution, you can use dplyr and do:

```
library(dplyr)

mtcars %>%
    group_by(cyl) %>%
    do(sampled = sample_n(., size = 100, replace = TRUE)) %>%
    select(-cyl) %>%
    unnest()
```





Here's another way using dplyr::slice

3

```
mtcars %>%
  group_by(cyl) %>%
  slice(sample(n(), 100, replace = T)) %>%
  ungroup()
```



answered Oct 15 '18 at 22:25

