Introduction to R

Answers to Session 4 exercises

Statistical Consulting Centre

1 March, 2017

1. Generate a one-way frequency table for `q1a`.

```
table(sports.df$q1a)


                             Daily               Several times a week
                                 2                                  8
             Several times a month Several times a year or less often
                                66                                649
```

2. Create a new variable called `q1a.sc` (meaning *q1a score*), where `q1a.sc` is of type numeric/integer rather than of type `factor`.

```
q1a.sc <- as.numeric(sports.df$q1a)
```

3. Generate a one-way frequency table of `q1a.sc` and compare it with the one you generated in question 1. Their frequencies should be identical.

```
table(q1a.sc)

q1a.sc
  1   2   3   4
  2   8  66 649
```

4. Repeat the steps in questions 1 – 3 for variables `q1b` to `q1e`, thereby creating new variables `q1b.sc` – `q1e`.

```
table(sports.df$q1b)


                             Daily               Several times a week
                                 3                                  4
             Several times a month Several times a year or less often
                                63                                649
```

```
q1b.sc <- as.numeric(sports.df$q1b)
table(q1b.sc)

q1b.sc
  1   2   3   4
  3   4  63 649

table(sports.df$q1c)


                             Daily               Several times a week
                                35                                277
             Several times a month Several times a year or less often
                               437                                194

q1c.sc <- as.numeric(sports.df$q1c)
table(q1c.sc)

q1c.sc
  1   2   3   4
 35 277 437 194

table(sports.df$q1d)


                             Daily               Several times a week
                                62                                281
             Several times a month Several times a year or less often
                               440                                177

q1d.sc <- as.numeric(sports.df$q1d)
table(q1d.sc)

q1d.sc
  1   2   3   4
 62 281 440 177

table(sports.df$q1e)


                             Daily               Several times a week
                               244                                371
             Several times a month Several times a year or less often
                               187                                110

q1e.sc <- as.numeric(sports.df$q1e)
table(q1e.sc)

q1e.sc
  1   2   3   4
244 371 187 110
```

5. Create a data frame called `mean.df` containing all five score variables (`q1a.sc` – `q1e.sc`) which you've created.

```
mean.df <- data.frame(cbind(q1a.sc, q1b.sc, q1c.sc, q1d.sc, q1e.sc))
## Always check whether mean.df contains what it supposed to contain
dim(mean.df)

[1] 996   5

head(mean.df)

  q1a.sc q1b.sc q1c.sc q1d.sc q1e.sc
1      4      4      4      2      1
2      4     NA      3      4      4
3     NA     NA     NA      3      1
4      4      4      2      3     NA
5      4      4      2      3      2
6     NA      4      2      1      1
```

6. Use `apply()` on `mean.df` to calculate each participant's mean score across variables `q1a.sc` – `q1e.sc`. Name this new variable `nerdy.sc`, meaning *nerdy score*.

```
nerdy.sc <- apply(mean.df, 1, mean, na.rm = TRUE)
```

7. Add the variable `nerdy.sc` to the `mean.df` data frame and use `summary()` to generate the five-number-summary of all *six* variables in `mean.df`.

```
mean.df$nerdy.sc <- nerdy.sc
head(mean.df)

  q1a.sc q1b.sc q1c.sc q1d.sc q1e.sc nerdy.sc
1      4      4      4      2      1     3.00
2      4     NA      3      4      4     3.75
3     NA     NA     NA      3      1     2.00
4      4      4      2      3     NA     3.25
5      4      4      2      3      2     3.00
6     NA      4      2      1      1     2.00

apply(mean.df, 2, summary)

         q1a.sc  q1b.sc q1c.sc q1d.sc q1e.sc nerdy.sc
Min.      1.000   1.000  1.000  1.000  1.000    1.000
1st Qu.   4.000   4.000  2.000  2.000  1.000    2.800
Median    4.000   4.000  3.000  3.000  2.000    3.000
Mean      3.879   3.889  2.838  2.762  2.179    3.006
3rd Qu.   4.000   4.000  3.000  3.000  3.000    3.333
Max.      4.000   4.000  4.000  4.000  4.000    4.000
NA's    271.000 277.000 53.000 36.000 84.000    7.000
```

8. Add the columns of `nerdy.sc` to `sports.df` for future use.

```
sports.df$nerdy.sc <- nerdy.sc
```

9. Use `tapply()` to calculate the mean nerdy score for all ten income levels.

```
with(sports.df, tapply(nerdy.sc, income, mean, na.rm = TRUE))

      > 100 000$  10 000$-15 000$  15 000$-20 000$  20 000$-25 000$ 
        2.951323         2.884722         3.092177         2.930044 
 25 000$-30 000$  30 000$-40 000$  40 000$-50 000$           5 000$ 
        3.075231         3.000517         3.070673         3.026471 
 50 000$-70 000$ 70 000$-100 000$ 
        2.983465         3.063077 
```

10. Income level 1 is shown first in the output of question 9 while income level 10 is shown last. Do you agree with `R`'s default ordering of `income` levels? If not, appropriately order the levels of `Income`.

```
sports.df$income = factor(sports.df$income, levels = c("5 000$", "10 000$-15 000$", "
                                             "20 000$-25 000$", "25 000$-30 000$", "3
                                             "40 000$-50 000$", "50 000$-70 000$", "7
                                             "> 100 000$"))
```

11. Repeat question 9 to check that *your chosen* ordering of `Income` levels has been correctly set.

```
with(sports.df, tapply(nerdy.sc, income, mean, na.rm = TRUE))

          5 000$  10 000$-15 000$  15 000$-20 000$  20 000$-25 000$ 
        3.026471         2.884722         3.092177         2.930044 
 25 000$-30 000$  30 000$-40 000$  40 000$-50 000$  50 000$-70 000$ 
        3.075231         3.000517         3.070673         2.983465 
70 000$-100 000$       > 100 000$ 
        3.063077         2.951323 
```

12. You were introduced to the following function, `mytab()`, in the Session 4 lecture slides.

```
mytab <- function(someinput){
 n <- length(someinput)
 n.missing <- na.check(someinput)
 n.complete <- n - n.missing
 mymean <- round(mean(someinput, na.rm = TRUE), 2)
 mysd <- round(sd(someinput, na.rm = TRUE), 2)
 mystder <- round(mysd/sqrt(n.complete), 2)
 Lower.CI <- round(mymean - 1.96*mystder, 2)
 Upper.CI <- round(mymean + 1.96*mystder, 2)
 c(Complete.obs = n.complete, Missing.obs = n.missing,
   Mean = mymean, Std.Error = mystder,
   Lower.CI = Lower.CI, Upper.CI = Upper.CI)
}
```

It depends on the `na.check()` function, defined earlier, to calculate the number of missing values, i.e., `mytab()` depends on the availability of `na.check()` in order for it to work. Modify `mytab()` so it does *no longer* depends on `na.check()` to calculate the number of missing values. Let's call the modified function `mytab1()`.

```
mytab1 <- function(someinput){
 n <- length(someinput)
 n.missing <- length(which(is.na(someinput)))
 n.complete <- n - n.missing
 mymean <- round(mean(someinput, na.rm = TRUE), 2)
 mysd <- round(sd(someinput, na.rm = TRUE), 2)
 mystder <- round(mysd/sqrt(n.complete), 2)
 Lower.CI <- round(mymean - 1.96*mystder, 2)
 Upper.CI <- round(mymean + 1.96*mystder, 2)
 c(Complete.obs = n.complete, Missing.obs = n.missing,
   Mean = mymean, Std.Error = mystder,
   Lower.CI = Lower.CI, Upper.CI = Upper.CI)
}
```

13. Use `mytab1()` to produce a summary table for all six variables in `mean.df`.

```
apply(mean.df, 2, mytab1)

             q1a.sc q1b.sc q1c.sc q1d.sc q1e.sc nerdy.sc
Complete.obs 725.00 719.00 943.00 960.00 912.00   989.00
Missing.obs  271.00 277.00  53.00  36.00  84.00     7.00
Mean           3.88   3.89   2.84   2.76   2.18     3.01
Std.Error      0.01   0.01   0.03   0.03   0.03     0.02
Lower.CI       3.86   3.87   2.78   2.70   2.12     2.97
Upper.CI       3.90   3.91   2.90   2.82   2.24     3.05
```

14. Use `mytab1()` to produce a summary table of nerdy scores for all ten income levels.

```
tapply(mean.df$nerdy.sc, sports.df$income, mytab1)

$`5 000$`
Complete.obs  Missing.obs         Mean    Std.Error     Lower.CI
       85.00         1.00         3.03         0.05         2.93
    Upper.CI
        3.13

$`10 000$-15 000$`
```

```
Complete.obs  Missing.obs         Mean    Std.Error     Lower.CI 
      120.00         0.00         2.88         0.05         2.78 
    Upper.CI 
        2.98 

$`15 000$-20 000$`
Complete.obs  Missing.obs         Mean    Std.Error     Lower.CI 
       98.00         0.00         3.09         0.05         2.99 
    Upper.CI 
        3.19 

$`20 000$-25 000$`
Complete.obs  Missing.obs         Mean    Std.Error     Lower.CI 
       76.00         0.00         2.93         0.05         2.83 
    Upper.CI 
        3.03 

$`25 000$-30 000$`
Complete.obs  Missing.obs         Mean    Std.Error     Lower.CI 
       72.00         1.00         3.08         0.05         2.98 
    Upper.CI 
        3.18 

$`30 000$-40 000$`
Complete.obs  Missing.obs         Mean    Std.Error     Lower.CI 
      129.00         1.00         3.00         0.05         2.90 
    Upper.CI 
        3.10 

$`40 000$-50 000$`
Complete.obs  Missing.obs         Mean    Std.Error     Lower.CI 
      104.00         2.00         3.07         0.05         2.97 
    Upper.CI 
        3.17 

$`50 000$-70 000$`
Complete.obs  Missing.obs         Mean    Std.Error     Lower.CI 
      127.00         1.00         2.98         0.05         2.88 
    Upper.CI 
        3.08 

$`70 000$-100 000$`
Complete.obs  Missing.obs         Mean    Std.Error     Lower.CI 
       65.00         0.00         3.06         0.06         2.94 
    Upper.CI 
        3.18 

$`> 100 000$`
Complete.obs  Missing.obs         Mean    Std.Error     Lower.CI 
```

```
       63.00         0.00         2.95         0.06         2.83
    Upper.CI
        3.07
```