Exporting Tables in R to Word

2024-03-14

Contents

1 Learning outcome:

By the end of the practical, students should be able to export output tables to Word. Should I learn this? Well, you can skip this exporting process and do things manually in Word directly. We do this to improve the quality of the report (and minimise errors!)

2 Prep

2.1 Create R_Tables folder

We aim to export several Tables. To enable easy access while we are writing up the report, we will generate a folder dedicated to the output tables. Please generate a folder, `R_Tables`.

2.2 Install eoffice package

We also need a package **eoffice**. This allows us to export a descriptive statistics table and a correlation table. We do not export a **t-test table** as the result is not satisfactory (You may manually create a table in Word for reporting the t-test result). In terms of regression tables, you can combine all models in one table to export by using **modelsummary** package.

```
# install.packages("modelsummary") # Regression tables
install.packages("eoffice")        # Export to Word
```

2.3 Open Practical_Regression_1.R

Open the **Practical_Regression_Exporting.R** file.

3 Descriptive statistics Option 1

Let's get a descriptive table.

```
health <- read_rds("health_cleaned.rds")

desc_var <- health %>%
      select(-hserial, -pserial, -gender,
             -employed)

psych::describe(desc_var, fast = TRUE)
```

```
##              vars    n  mean    sd   min    max  range   se
## bmival          1 5673 27.48  5.42 13.55  65.28  51.73 0.07
## socialclass     2 5673  4.17  2.13  0.00   7.00   7.00 0.03
## age             3 5673 45.53 14.16 16.00  70.00  54.00 0.19
## fruitveg        4 5673  3.67  2.68  0.00  30.00  30.00 0.04
## alcohol         5 5673 12.28 21.60  0.00 461.50 461.50 0.29
## gender_num      6 5673  0.55  0.50  0.00   1.00   1.00 0.01
## employed_num    7 5673  0.67  0.47  0.00   1.00   1.00 0.01
```

3.1 First attempt

Let's test it to see how R exports the table to Word. To do so, we first save the table in an object, **a**. Following this, We load the **eoffice** library, and use the function, **totable**. Then, type **a**, which is the object table you intend to export to, before adding a file name in the Word document.

```
a <- psych::describe(desc_var, fast = TRUE)
library("eoffice")
totable(a, "R_Tables/Desc.docx")
```

Go to the file explorer and open the exported Word document, `Desc.docx`.

vars	n	mean	sd	min	max	range	se
1	5,673	27.4804546	5.4177175	13.55082	65.27721	51.72639	0.071929983
2	5,673	4.1697515	2.1317471	0.00000	7.00000	7.00000	0.028302793
3	5,673	45.5288207	14.1551793	16.00000	70.00000	54.00000	0.187935569
4	5,673	3.6664023	2.6834035	0.00000	30.00000	30.00000	0.035627027
5	5,673	12.2795641	21.6003682	0.00000	461.50000	461.50000	0.286783898
6	5,673	0.5494447	0.4975931	0.00000	1.00000	1.00000	0.006606447
7	5,673	0.6730125	0.4691540	0.00000	1.00000	1.00000	0.006228867

What's your verdict? Unfortunately, the default setting does not meet our needs. There are several issues including excessive decimal places, missing variable names and the last two columns are unnecessary. We'll address this one at a time.

3.2 Prep before exporting

3.2.1 Remove excessive decimal places

To keep the decimal places up to 2, we use `round`

```
##                vars    n  mean    sd   min    max  range   se
## bmival            1 5673 27.48  5.42 13.55  65.28  51.73 0.07
## socialclass       2 5673  4.17  2.13  0.00   7.00   7.00 0.03
## age               3 5673 45.53 14.16 16.00  70.00  54.00 0.19
## fruitveg          4 5673  3.67  2.68  0.00  30.00  30.00 0.04
## alcohol           5 5673 12.28 21.60  0.00 461.50 461.50 0.29
## gender_num        6 5673  0.55  0.50  0.00   1.00   1.00 0.01
## employed_num      7 5673  0.67  0.47  0.00   1.00   1.00 0.01
```

3.2.2 Save the variable names

Let's save the variable names by using `row.names(object)`. We treat the names as character.

```
Variable <- as.character(row.names(a))
Variable

## [1] "bmival"       "socialclass"  "age"          "fruitveg"     "alcohol"
## [6] "gender_num"   "employed_num"
```

3.2.3 Merge variable names and the table in two decimal places

We use `data.frame` to merge two objects. Let's print the first seven columns, which is what we want. Now, we are all set.

```
desc <- data.frame(Variable, a2)
desc[, 1:7]
```

```
##                 Variable vars    n  mean    sd   min    max
## bmival            bmival    1 5673 27.48  5.42 13.55  65.28
## socialclass   socialclass    2 5673  4.17  2.13  0.00   7.00
## age                   age    3 5673 45.53 14.16 16.00  70.00
## fruitveg         fruitveg    4 5673  3.67  2.68  0.00  30.00
## alcohol           alcohol    5 5673 12.28 21.60  0.00 461.50
## gender_num     gender_num    6 5673  0.55  0.50  0.00   1.00
## employed_num employed_num    7 5673  0.67  0.47  0.00   1.00
```

3.3 Second attempt

If we proceed to export, the pre-existing file will be overwritten. If the `Desc` Word file is open, close the file first. Let's export the first seven columns by specifying `desc[, 1:7]` in front of the filename.

```
totable(desc[, 1:7], "R_Tables/Desc.docx")
```

Now, open the Word file and see the result. The categorical variables seem not correct. Let's try again with another approach by using `tableone` package.

4 Descriptive statistics Option 2

You are already familiar with `CreateTableOne`. We will learn one option, `showAllLevels`, that allows us to `print` all categories for categorical variables. The option, `showAllLevels` can be used in conjunction with `print` Before that, we declare which variables we will summarise.

```
library(tableone)
# declare vars to summarise
vars <-  c("bmival", "socialclass", "age", "fruitveg",
         "alcohol", "gender", "employed")
# showAllLevels includes all categories
a <- print(CreateTableOne(data = health[, vars]),
      showAllLevels = TRUE )

##
##                          level Overall
##   n                             5673
##   bmival (mean (SD))            27.48 (5.42)
##   socialclass (mean (SD))        4.17 (2.13)
##   age (mean (SD))               45.53 (14.16)
##   fruitveg (mean (SD))           3.67 (2.68)
##   alcohol (mean (SD))           12.28 (21.60)
##   gender (%)             Men     2556 (45.1)
##                          Women   3117 (54.9)
##   employed (%)           No      1855 (32.7)
##                          Yes     3818 (67.3)
```

The next step is to repeat the code that merges variable names. Let's give a new filename, `Desc2` and export the table.

```
# save the variable names
Variable <- as.character(row.names(a))
# merge variable names
desc <- data.frame(Variable, a)
totable(desc, "R_Tables/Desc2.docx")
```

Now, open the Word file and see what the table looks like.

5 Correlation matrix

Let's produce a correlation matrix.

```
source("SK_Functions_corstars.R")
corr <- fn_corstars(desc_var)
corr

##                  bmival socialclass        age   fruitveg    alcohol
gender_num
## bmival
## socialclass  -0.045***
## age           0.189***    0.024
## fruitveg     -0.020       0.191***   0.097***
## alcohol      -0.018       0.044***   0.026     -0.066***
## gender_num   -0.017      -0.039**   -0.002      0.056***  -0.182***
## employed_num -0.060***    0.241***  -0.278***   0.027*     0.018     -0.114***
```

Again, we follow the steps above by adding variable names. You will see a new Variable column.

```
Variable <- as.character(row.names(corr))
corr2 <- data.frame(Variable, corr)
corr2

##                  Variable     bmival socialclass        age   fruitveg
## bmival             bmival
## socialclass   socialclass -0.045***
## age                  age  0.189***    0.024
## fruitveg         fruitveg -0.020       0.191***   0.097***
## alcohol           alcohol -0.018       0.044***   0.026     -0.066***
## gender_num     gender_num -0.017      -0.039**   -0.002      0.056***
## employed_num employed_num -0.060***    0.241***  -0.278***   0.027*
##                 alcohol gender_num
## bmival
## socialclass
## age
## fruitveg
## alcohol
## gender_num   -0.182***
## employed_num  0.018     -0.114***
```

Let's give a new filename, `Correlation` and export the table.

```
totable(corr2, "R_Tables/Correlation.docx")
```

6 Regression models

We fit two models and combine regression models in one table.

```
model1 <- lm(bmival ~ socialclass, data = health)
model2 <- lm(bmival ~ socialclass + age + gender +
      employed + fruitveg + alcohol, data = health)
```

We then save the list of all models.

```
library(modelsummary)
models <- list(model1, model2)
```

Along with coefficients, we also need other model fit information. Specify it at `gof_map`.

```
# gof = goodness of fit. Declare what to print.
fit <- c("nobs", "r.squared", "adj.r.squared" )
modelsummary(models, stars = TRUE, gof_map = fit)
```

	(1)	(2)
(Intercept)	27.957***	24.962***
	(0.158)	(0.329)
socialclass	-0.114***	-0.113**
	(0.034)	(0.035)
age		0.074***
		(0.005)
genderWomen		-0.224
		(0.145)
employedYes		0.049
		(0.163)
fruitveg		-0.062*
		(0.027)

	(1)	(2)
alcohol		-0.007*
		(0.003)
Num.Obs.	5673	5673
R2	0.002	0.040
R2 Adj.	0.002	0.039

+ p < 0.1, * p < 0.05, ** p < 0.01, *** p < 0.001

To export, we simply add `output = "FILENAME"`.

```
modelsummary(models, stars = TRUE, gof_map = fit,
        output = "R_Tables/Regression.docx")
```

The tables are not perfect, but almost in report-ready format. Well-done.