Exemple of generating a report from a health dataset with the r package *dataepi*

R. Priam*

June 1, 2021

Abstract

The r package dataepi allows to generate a report for the analysis of a dataset via a kx2x2 health table with odds ratios, relative risks, descriptive statistics of the variable and statistical tests for two variables. This document presents un example of first analysis and a description of several functionalities of the package.

Loading of the dataset and preparation of the variables

First the library is loaded with rstudio. The data.frame is in the variable A for analyzing with the r package dataepi. The command for loading the library is *library()* preparing the variable for the package are as follows:

```
> library(MASS)
> data("Pima.tr")
> data("Pima.te")
> A = rbind(Pima.tr,Pima.te)
> A$id=1:nrow(A)
> A$npreg = as.character(A$npreg)
> A$npreg[A$npreg%in%as.character(10:17)]="10_17"
> A$diabet = as.numeric(as.character(A$type)=="Yes")
> A$bp_pb = as.character(as.numeric(as.numeric(as.character(A$bp))>90))
> A$glu_cl3 = as.character(cut(A$glu,c(56,103,129,199),
+                              labels=c("56_103","103_129","129_199")))
> A$skin_cl3 = as.character(cut(A$skin,c(7,24,33,99),
+                               labels=c("7_24","24_33","33_99")))
> A$bmigroups = as.character(cut(A$bmi,c(0,25,30,100),
+                                labels=c("a_normal","b_overweight","c_obese")))
> A$ped_cl3 = as.character(cut(A$ped,c(0.085,0.295,0.557,2.420),
+                              labels=c("low","middle","large")))
> A$age_cl3 = as.character(cut(A$age,c(21,24,33,81),labels=c("21_24","24_33","33_81")))
>
> var_y = "diabet"
> vars_cont = c("age","bp","glu","skin","bmi","ped","age")
> vars_disc = c("npreg","bp_pb","bmigroups","glu_cl3","skin_cl3","ped_cl3","age_cl3")
```

*rpriam@gmail.com

```
> vars_x = c("npreg","bp_pb","bmigroups","skin_cl3","ped_cl3","age_cl3")
> vars_int = NULL
> var_id = "id"
```

Checking last character
It is not allows a digit as last character of the name of a variable, hence, this may be added,

```
> vars_cont = unique(vars_cont)
> vars_disc = unique(vars_disc)
> vars_x    = unique(vars_x)
>
> for (j in 1:ncol(A)) {
+     nv=names(A)[j]
+     if(substr(nv,nchar(nv),nchar(nv))%in%paste(0:9))
+       names(A)[j] = paste(names(A)[j],"_",sep="")
+     if (sum(nv%in%vars_cont))
+       { l = which(vars_cont%in%nv); vars_cont[l] = names(A)[j]; }
+     if (sum(nv%in%vars_disc))
+       { l = which(vars_disc%in%nv); vars_disc[l] = names(A)[j]; }
+     if (sum(nv%in%vars_x))
+       { l = which(vars_x%in%nv); vars_x[l] = names(A)[j]; }
+ }
```

Adding the description of the study (facultative)
The descriptive for the study may be added as,

```
> list_supp = list()
> list_supp$where     = " "
> list_supp$who       = " "
> list_supp$disease   = " "
> list_supp$objective = " "
> list_supp$project   = " "
> list_supp$inex      = " "
```

Checking the variables
Let's have a look to the variables, the corresponding output is as follows.

```
> str(A)
'data.frame': 532 obs. of  16 variables:
 $ npreg    : chr  "5" "7" "5" "0" ...
 $ glu      : int  86 195 77 165 107 97 83 193 142 128 ...
 $ bp       : int  68 70 82 76 60 76 58 50 80 78 ...
 $ skin     : int  28 33 41 43 25 27 31 16 15 37 ...
 $ bmi      : num  30.2 25.1 35.8 47.9 26.4 35.6 34.3 25.9 32.4 43.3 ...
 $ ped      : num  0.364 0.163 0.156 0.259 0.133 ...
 $ age      : int  24 55 35 26 23 52 25 24 63 31 ...
```

```
 $ type     : Factor w/ 2 levels "No","Yes": 1 2 1 1 1 2 1 1 1 2 ...
 $ id       : int  1 2 3 4 5 6 7 8 9 10 ...
 $ diabet   : num  0 1 0 0 0 1 0 0 0 1 ...
 $ bp_pb    : chr  "0" "0" "0" "0" ...
 $ glu_cl3_ : chr  "56_103" "129_199" "56_103" "129_199" ...
 $ skin_cl3_: chr  "24_33" "24_33" "33_99" "33_99" ...
 $ bmigroups: chr  "c_obese" "b_overweight" "c_obese" "c_obese" ...
 $ ped_cl3_ : chr  "middle" "low" "low" "low" ...
 $ age_cl3_ : chr  "21_24" "33_81" "33_81" "24_33" ...
```

It is recognized variables with discrete values, continuous values, binary values or polytomous values. Normally, the dataset may be known because it may have been produced by the investigator of the study, otherwise it is wise to have a look of the number of values of each variable, and when they are not too numerous their unique set.

Preparation of the dataset with the function data_prepare()
A function of the r package prepares the dataset for the other functions,

```
> A0 = A;
> fp = data_prepare(A,var_y,vars_cont,vars_disc,var_id)
> A  = fp$A
> A  = A[,unique(c(var_y,vars_cont,vars_disc,vars_x,var_id))]
```

Checking the variables with the function tab_contents()
A description of the variables for the analysis is as follows,

```
> desc_all          <- tab_contents(A)
> print(desc_all)
    variable   r_class nblevels nbobs
1     diabet   numeric        2   532
2        age   numeric       46   532
3         bp   numeric       42   532
4        glu   numeric      126   532
5       skin   numeric       50   532
6        bmi   numeric      222   532
7        ped   numeric      413   532
8      npreg character       11   532
9      bp_pb character        2   532
10 bmigroups character        3   532
11  glu_cl3_ character        4   532
12 skin_cl3_ character        4   532
13  ped_cl3_ character        4   532
14  age_cl3_ character        4   532
15        id   integer      532   532
```

Checking the generated tables, one by one
A way to get the tables for the analysis leads to,

```
> desc_cont         <- tab_desc_cont(A,vars_cont)
> desc_disc         <- tab_desc_disc(A,vars_disc)
> desc_biv          <- tab_desc2class_cont(A,vars_cont,var_y)
> test_tt           <- tab_tt2classes_cont(A,vars_cont,var_y)
> test_anova        <- tab_ttanova_cont(A,vars_cont,vars_disc)
> test_chi2         <- tab_chi2all(A,c(var_y,vars_disc),pvalue_seuil_ = 0.05)
> or                <- tab_all2x2(A,vars_x,var_y,stat_oddsratio)
> rr                <- tab_all2x2(A,vars_x,var_y,stat_relativerisk)
> gg                <- tab_glmorr(A,vars_x,var_y)
```

Checking the generated tables, all once, with the function rep_compute()
A way to get all thes table for the analysis is as:

```
> au = dataepi::rep_compute(A, var_y, vars_x, vars_cont, vars_disc, vars_int, var_id)
> print(names(au[1:7]))
[1] "Anew"       "desc_all"   "desc_cont"  "desc_disc"  "desc_biv"   "test_tt"    "test_anova"
> print(names(au[8:13]))
[1] "test_chi2" "or"        "rr"        "gg"        "fv"        "args"
> print(au$args$vars_x)
[1] "npreg"     "bp_pb"     "bmigroups" "skin_cl3_" "ped_cl3_"  "age_cl3_"
> print(au$args$var_y)
[1] "diabet"
```

Generating the report with .tex extension with the function rep_write()
The command lines for the report creation before compilation into a pdf or ps file are,

```
> fnl <- paste("./report_dataepi_", data_,".tex",sep="")
> if (file.exists(fnl)) {file.remove(fnl);}
[1] TRUE
> if (!exists("list_supp")) list_supp=NULL;
> wr = dataepi::rep_write(fnl,"tex",A, var_y, vars_x, vars_cont,
+                         vars_disc, vars_int, var_id, list_supp)
```

This function executes the function dataepi::au() and then write the report in the file with the name in the variable fnl. To check the header of the file,

```
> file_tex = read.csv(fnl, header = FALSE)
> print(file_tex[1:10,])
 [1] \\documentclass[12pt]{article}
 [2] \\usepackage[margin=0.7in]{geometry}
 [3] \\usepackage[utf8]{inputenc}
 [4] \\usepackage{graphics}
 [5] \\usepackage{datetime}
 [6] \\usepackage{pdflscape}
 [7]
 [8] \\author{ }
 [9] \\date{\\today (\\currenttime)}
[10] \\title{Report\\footnote{This document is auto-generated from the r package daatepi.} for
```