Demographic Health Survey Data in R

Pakistan Demographic Health Survey Data

Table of contents

Prof.Dr. Zahid Asghar, School of Economics, QAU

## Purpose

Main purpose of this post is to explain how to download and read DHS data in R, how to create variables, make tables and cross-tables and run a simple logit regression. I am using in this post is IR which is the individual (women’s) recode file. For more details about files you can visit [dhs](dhsprogram.com). USAID main sponsoring agency of DHS data has recently started using open community software for DHS data. I am using it for Pakistani data 2017-18. Main outline of this post is as follows:

1. Reading dhs files in R from STATA (one can do with with SPSS and other formats as well)
2. Creating new variables , tables, cross-tables, surveydesign and regression. I shall mainly use tidyverse package but you can use base R if you like.

## Install R and RStudio (Urdu/Hindi langues for this heading only)

Once you have installed R, Rtools (in some cases using Windows) and RStudio<https://youtu.be/ZNBZevfYgo0>, you need to install a package. A package is like fixing a bulb and you have to do it only once. Once you have installed relevant package(s), next is to recall it using library() which is like switching on a button of a bulb. There is base-R commands but very easy and useful is to learn tidyverse which consists of many useful packages, we will talk later on.

As always, the first thing we will do is load the tidyverse.

*Note: If you haven’t yet installed the tidyverse, you’ll first have to run the code install.packages(“tidyverse”).* You may get some warnings when you recall it. Ignore these warnings and go ahead. What are errors, warnings and messages, you may watch my video <https://youtu.be/SRSNdCZ_QnE>.

## DHS data in R

Please have the dataset downloaded and ready to use. I shall use the IR file.You can download the model dataset [here](https://www.dhsprogram.com/data/Model-Datasets.cfm) For the code below to work, you must save the dataset in the same folder as your R scripts.

Install and load the packages you need. You only need to install packages once. After installing a package you can load the package using the library command.

### Load relevant packages

library(tidyverse) # its an umbrella library containing 9 important and widely used packages of R  
library(naniar) # to use replace\_with\_na function  
library(haven) # to read STATA/SPSS data  
library(here) # to check what is your file path  
library(labelled) # to get labels of vairables  
library(pollster)  
library(gt) # to get tables  
library(survey) # For analysis of complex survey data sets  
here::here() # check your path/directory and make sure your data is in this directory

[1] "D:/RepTemplates/pdhsurvey"

## Load data

As I have my data in a folder data , so I am giving a parth inside here of the relevant folder and then data.

# open your dataset  
#pkir <- read\_dta(here("data","PKIR71FL.dta"))  
#   
# save(pkir, file = 'data/pkir.RData') #you can save data in R. It will reduce file size and if its correctly save, you can just click on data and it will be uploaded with the command as follows:  
  
load("D:/RepTemplates/pdhs/data/pkir.RData")

## Modern contraceptive use

To recode modern contraceptive use into a binary variable, first I check category labels of v313 representing methods used for family planning.

print\_labels(pkir$v313) # print\_labels command for variable labels

Labels:  
 value label  
 0 no method  
 1 folkloric method  
 2 traditional method  
 3 modern method

Recode v313 as 1 if female is using modern method and 0 if otherwise. For this I use mutate for creating a new variable modfp. To create survey weight variables wtand finally a table of summary statistics is calculated.

# creating a new categorical variable using mutate and case\_when commands  
  
pkir <- pkir |>  
 mutate(modfp =  
 case\_when(v313 == 3 ~ 1,  
 v313 < 3 ~ 0 )) |>  
 set\_value\_labels(modfp = c(yes = 1, no = 0)) |>  
 set\_variable\_labels(modfp ="Currently used any modern method")  
  
# creating the sampling weight variable.   
pkir <- pkir |> mutate(wt=v005/1000000)  
#pkir$wt <- pkir$v005/1000000  
  
# check with final report Table 7.3  
topline(pkir, modfp, wt ) |> gt()

| Currently used any modern method | Frequency | Percent | Valid Percent | Cumulative Percent |
| --- | --- | --- | --- | --- |
| no | 9353.449 | 75.65067 | 75.65067 | 75.65067 |
| yes | 3010.551 | 24.34933 | 24.34933 | 100.00000 |

There are 24% females who are using modern family planning methods.

## Example 2

Number of ANC visits in 4 categories that match the table in the final report age of child needed for this indicator. If b19\_01 is not available in the data use v008 - b3\_01

pkir <- pkir |>  
 mutate(age = b19\_01)  
  
pkir <- pkir |>  
 mutate(age = v008 - b3\_01)  
  
  
pkir <- pkir |>  
 mutate(ancvisits =  
 case\_when(  
 m14\_1 == 0 ~ 0 ,  
 m14\_1 == 1 ~ 1 ,  
 m14\_1 %in% c(2,3) ~ 2 ,  
 m14\_1>=4 & m14\_1<=90 ~ 3 ,  
 m14\_1>90 ~ 9 ,  
 age>=60 ~ 99 )) |>  
 replace\_with\_na(replace = list(ancvisits = c(99))) |>  
 set\_value\_labels(ancvisits = c(none = 0, "1" = 1, "2-3"=2, "4+"=3, "don't know/missing"=9 )) |>  
 set\_variable\_labels(ancvisits = "Number of ANC visits")  
  
# check with final report Table 9.2  
topline(pkir, ancvisits, wt ) |> gt() |> fmt\_number(decimals = 2)

| Number of ANC visits | Frequency | Percent | Valid Percent | Cumulative Percent |
| --- | --- | --- | --- | --- |
| none | 816.91 | 6.61 | 12.17 | 12.17 |
| 1 | 556.75 | 4.50 | 8.30 | 20.47 |
| 2-3 | 1,856.84 | 15.02 | 27.67 | 48.14 |
| 4+ | 3,452.49 | 27.92 | 51.45 | 99.59 |
| don't know/missing | 27.83 | 0.23 | 0.41 | 100.00 |
| (Missing) | 5,653.18 | 45.72 | NA | NA |

library(gt)  
  
topline(pkir, ancvisits, wt ) |>   
 gt() |> fmt\_number(decimals = 2)

| Number of ANC visits | Frequency | Percent | Valid Percent | Cumulative Percent |
| --- | --- | --- | --- | --- |
| none | 816.91 | 6.61 | 12.17 | 12.17 |
| 1 | 556.75 | 4.50 | 8.30 | 20.47 |
| 2-3 | 1,856.84 | 15.02 | 27.67 | 48.14 |
| 4+ | 3,452.49 | 27.92 | 51.45 | 99.59 |
| don't know/missing | 27.83 | 0.23 | 0.41 | 100.00 |
| (Missing) | 5,653.18 | 45.72 | NA | NA |

### Setting the survey design using the svydesign command from the survey package ###  
# the survey design will be saved in an object named mysurvey, you can change this name to another name  
mysurvey<-svydesign(id=pkir$v021, data=pkir, strata=pkir$v022, weight=pkir$wt, nest=T)  
options(survey.lonely.psu="adjust")   
  
# the option (survey.lonely.psu="adjust") is used to adjust for the case if there is single PSU within a stratum   
  
# now that we have set the survey design, we can use the survey package commands. See below for some examples.

# Modern contraceptive use indicator. This indicator was also produced in the RecodingDHSVars.R  
pkir <- pkir %>%  
 mutate(modfp =   
 case\_when(v313 == 3~ 1,  
 v313<3~ 0)) %>%   
 set\_value\_labels(modfp = c(yes = 1, no = 0)) %>%  
 set\_variable\_labels(modfp ="Currently used any modern method")  
  
# to get frequencies  
svytable(~modfp, mysurvey)

modfp  
 0 1   
9353.449 3010.551

# to get proportions  
prop.table(svytable(~modfp, mysurvey))

modfp  
 0 1   
0.7565067 0.2434933

#you can save the result in an object  
table1 <- prop.table(svytable(~v106, mysurvey))  
  
#then you can refer to this object later on  
table1

v106  
 0 1 2 3   
0.4917832 0.1647189 0.2121359 0.1313621

#If you want sampling error (SE) and confidence intervals (CI), you can use svyby with by=1 for total  
svyby(~modfp,by=1, design=mysurvey, FUN=svymean, vartype=c("se", "ci"))

by modfp se ci\_l ci\_u  
1 1 0.2434933 0.006556505 0.2306428 0.2563438

#if you want to do this for a categorical variable, then you should add as.factor  
#below we did not need as.factor since we had a binary variable  
#for instance education level, v106, has four categories. You can estimate the SE and cI for each category.  
svyby(~as.factor(v106),by=1, design=mysurvey, FUN=svymean, vartype=c("se", "ci"))

by as.factor(v106)0 as.factor(v106)1 as.factor(v106)2 as.factor(v106)3  
1 1 0.4917832 0.1647189 0.2121359 0.1313621  
 se.as.factor(v106)0 se.as.factor(v106)1 se.as.factor(v106)2  
1 0.01558661 0.006194613 0.01019698  
 se.as.factor(v106)3 ci\_l.as.factor(v106)0 ci\_l.as.factor(v106)1  
1 0.00814183 0.461234 0.1525777  
 ci\_l.as.factor(v106)2 ci\_l.as.factor(v106)3 ci\_u.as.factor(v106)0  
1 0.1921502 0.1154044 0.5223323  
 ci\_u.as.factor(v106)1 ci\_u.as.factor(v106)2 ci\_u.as.factor(v106)3  
1 0.1768601 0.2321216 0.1473198

# Crosstabulation of modfp (modern FP use) and place of education (v106)  
table2 <- svyby(~modfp, by = ~v106, design=mysurvey, FUN=svymean, vartype=c("se", "ci"))  
  
table2

v106 modfp se ci\_l ci\_u  
0 0 0.2101258 0.009068765 0.1923513 0.2279002  
1 1 0.2744924 0.016388275 0.2423720 0.3066129  
2 2 0.2650724 0.012835547 0.2399152 0.2902296  
3 3 0.2946932 0.017590575 0.2602163 0.3291701

## chi-square results

svychisq(~modfp+v106, mysurvey)  
  
# to see how the survey commands are used for regressions, please check the regression R script.   
# To do this for several variables at once, you can do the following  
  
# List of variables to crosstabulate with your outcome  
variables <- c("v013","v106", "v190","v025","v024")   
  
results <- list() # Empty list to store the crosstabulation results  
  
# Loop through the variables  
for (var in variables) {  
 # Crosstabulation using svyby  
 crosstab <- svyby(~ modfp, by = as.formula(paste("~", var)), design = mysurvey, FUN = svymean, vartype = c("se", "ci"))  
   
 chi\_square <- svychisq(as.formula(paste("~ modfp +", var)), design = mysurvey)  
   
 # Store results in list  
 results[[var]] <- list(crosstab = crosstab, chi\_square = chi\_square)  
}  
  
# Access the crosstabulation results for each variable  
for (var in variables) {  
 print(paste("Crosstabulation for", var))  
 print(results[[var]])  
}

## Compute linear model

reg1 <- svyglm(modfp~1+v025+as.factor(v013),  
 design = mysurvey,  
 family = binomial(link = 'logit'))

Warning in eval(family$initialize): non-integer #successes in a binomial glm!

library(broom)

Warning: package 'broom' was built under R version 4.3.1

tidy(reg1)

# A tibble: 8 × 5  
 term estimate std.error statistic p.value  
 <chr> <dbl> <dbl> <dbl> <dbl>  
1 (Intercept) -2.37 0.226 -10.5 4.43e-23  
2 v025 -0.235 0.0685 -3.43 6.60e- 4  
3 as.factor(v013)2 0.865 0.211 4.10 4.88e- 5  
4 as.factor(v013)3 1.40 0.216 6.45 2.96e-10  
5 as.factor(v013)4 1.87 0.214 8.74 4.96e-17  
6 as.factor(v013)5 1.96 0.212 9.23 1.12e-18  
7 as.factor(v013)6 2.08 0.222 9.33 5.33e-19  
8 as.factor(v013)7 1.64 0.227 7.26 1.81e-12

# unadjusted with place of residence   
reg1 <- svyglm(modfp~ 1+ v025 , design=mysurvey, family=binomial(link="logit"))

Warning in eval(family$initialize): non-integer #successes in a binomial glm!

# to see the results  
summary(reg1)

Call:  
svyglm(formula = modfp ~ 1 + v025, design = mysurvey, family = binomial(link = "logit"))  
  
Survey design:  
svydesign(id = pkir$v021, data = pkir, strata = pkir$v022, weight = pkir$wt,   
 nest = T)  
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) -0.64717 0.10343 -6.257 9.24e-10 \*\*\*  
v025 -0.30138 0.06782 -4.444 1.12e-05 \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
(Dispersion parameter for binomial family taken to be 1.000081)  
  
Number of Fisher Scoring iterations: 4

# to get ORs  
ORreg1 <- exp(reg1$coefficients )  
ORreg1

(Intercept) v025   
 0.5235268 0.7397958

# multivariate models with place of residence (v025), education (v106), and current age (v013)  
reg2 <- svyglm(modfp~ 1+ v025 + as.factor(v106) + as.factor(v013), design=mysurvey, family=binomial(link="logit"))

Warning in eval(family$initialize): non-integer #successes in a binomial glm!

tidy(reg2)

# A tibble: 11 × 5  
 term estimate std.error statistic p.value  
 <chr> <dbl> <dbl> <dbl> <dbl>  
 1 (Intercept) -2.79 0.245 -11.4 1.66e-26  
 2 v025 -0.100 0.0697 -1.43 1.52e- 1  
 3 as.factor(v106)1 0.389 0.0959 4.06 5.86e- 5  
 4 as.factor(v106)2 0.360 0.0852 4.22 2.92e- 5  
 5 as.factor(v106)3 0.412 0.105 3.93 9.77e- 5  
 6 as.factor(v013)2 0.842 0.210 4.01 7.23e- 5  
 7 as.factor(v013)3 1.38 0.216 6.37 4.79e-10  
 8 as.factor(v013)4 1.85 0.213 8.69 7.61e-17  
 9 as.factor(v013)5 1.97 0.212 9.31 6.24e-19  
10 as.factor(v013)6 2.13 0.224 9.50 1.37e-19  
11 as.factor(v013)7 1.71 0.229 7.48 4.23e-13

## checking for correlations

create a subset of data with variables you want to check

cordata <- subset(pkir, select=c(v025, v106, v013 ,v190))  
  
mycordata <- cor(cordata)  
# we see that v025 and v190 may be highly correlated  
  
# to produce a correlation plot  
#install.packages("corrplot")  
library(corrplot)

Warning: package 'corrplot' was built under R version 4.3.1

corrplot 0.92 loaded

corrplot(mycordata)

