MIRROR data manipulation

Thijs Noordzij

26 mei 2016

# Open MIRROR-file from SPSS .sav file extension

## Require haven package

Haven package is faster than foreign package, and loads the data without error messages. Variable labels are stored in the "label" attribute of each variable.

Hadley Wickham and Evan Miller (2015). haven: Import SPSS, Stata and SAS Files. R package version 0.2.0. <https://CRAN.R-project.org/package=haven>

if (require("haven")){  
 print("haven is loaded correctly")  
} else {  
 print("trying to install haven")  
 install.packages("haven")  
 if (require("haven")){  
 print("haven installed and loaded")  
 } else {  
 stop("could not install haven")  
 }  
}  
citation("haven")

## Load data

It takes almost 1 minute to load the data.

data <- read\_spss("C:/Users/Thijs/Documents/MIRROR/data/mirror\_formatie\_ tm2014.sav")

The output of system.time() of previous code section:

|  |  |  |
| --- | --- | --- |
| user | system | elapsed |
| 40.72 | 0.33 | 41.08 |

# Recoding missing values correctly

## Define NA in FUNGRP(\_PVE1), BRIN, GEBDAT, GESLACHT

### FUNGRP

NA defined as in SPSS file for MIRROR provided by DUO. FUNGRP also has 2431 system.missings in SPSS.

missing\_FUNGRP <- c(-1)  
data$FUNGRP <- lapply(data$FUNGRP, function(x) replace(x, x %in% missing\_FUNGRP, NA))  
data$FUNGRP <- as.numeric(data$FUNGRP)

The output of system.time() of previous section of code:

|  |  |  |
| --- | --- | --- |
| user | system | elapsed |
| 29.65 | 0.13 | 29.92 |

### FUNGRP\_PVE

NA defined as in SPSS file for MIRROR provided by DUO. FUNGRP also has 2869 system.missings in SPSS.

missing\_FUNGRP\_PVE1 <- c(-1)  
data$FUNGRP\_PVE1 <- lapply(data$FUNGRP\_PVE1, function(x) replace(x, x %in% missing\_FUNGRP\_PVE1, NA))  
data$FUNGRP\_PVE1 <- as.numeric(data$FUNGRP\_PVE1)

### FUNGRP\_FMIX1

NA defined as in SPSS file for MIRROR provided by DUO, and "-2" added. FUNGRP also has 8424 system.missings in SPSS.

missing\_FUNGRP\_FMIX1 <- c(-1, -2)  
data$FUNGRP\_FMIX1 <- lapply(data$FUNGRP\_FMIX1, function(x) replace(x, x %in% missing\_FUNGRP\_FMIX1, NA))  
data$FUNGRP\_FMIX1 <- as.numeric(data$FUNGRP\_FMIX1)

### BRIN

NA defined as in SPSS file for MIRROR provided by DUO.

missing\_BRIN <- c("O")   
data$BRIN <- lapply(data$BRIN, function(x) replace(x, x %in% missing\_BRIN, NA))

The output of system.time() of previous section of code:

|  |  |  |
| --- | --- | --- |
| user | system | elapsed |
| 30.81 | 0.02 | 30.83 |

There are many more unlikely, such as "O", "VMB", "0000", "P&O", "PR", "SB O", "STAF", "TEAM". In a [next section](#ValidateBRIN) this will be handled.

### GEBDAT

NA defined as in SPSS file for MIRROR provided by DUO.

missing\_GEBDAT <- c(-1)   
data$GEBDAT <- lapply(data$GEBDAT, function(x) replace(x, x %in% missing\_GEBDAT, NA))

The output of system.time() of previous section of code:

|  |  |  |
| --- | --- | --- |
| user | system | elapsed |
| 54.51 | 0.11 | 54.81 |

### GESLACHT

NA defined as in SPSS file for MIRROR provided by DUO, "" (empty string) added.

missing\_GESLACHT <- c("", "O")   
data$GESLACHT <- lapply(data$GESLACHT, function(x) replace(x, x %in% missing\_GESLACHT, NA))

The output of system.time() of previous section of code:

|  |  |  |
| --- | --- | --- |
| user | system | elapsed |
| 30.50 | 0.06 | 30.60 |

## Total missing values per variable

sum(is.na(data$BRIN))  
sum(is.na(data$GEBDAT))  
sum(is.na(data$GESLACHT))  
sum(is.na(data$FUNGRP))  
sum(is.na(data$FUNGRP\_PVE1))  
sum(is.na(data$FUNGRP\_FMIX1[data$JAAR >= 2002]))

|  |  |  |
| --- | --- | --- |
| variable | missing values | comments |
| BRIN | 1393 |  |
| GEBDAT | 4156 |  |
| GESLACHT | 4059 | if " " is added as missing value, else 0 |
| FUNGRP | 10559 | that includes 2431 system.missing in SPSS |
| FUNGRP\_PVE1 | 9067 | that includes 2869 system.missing in SPSS |
| FUNGRP\_FMIX1 | 8424 | number of NA in JAAR >= 2002. In JAAR < 2002 all values are NA |

## Validate BRIN

BRIN is a unique identifier for a school. BRIN has a standard format: 4 characters, number, number, character, character (e.g. "11AO"). Some BRIN in the MIRROR-file do not represent the school where the staff (teacher or otherwise) works. These 'BRIN' represent staff working under direct responsibility of the board. These BRIN have a free format, which deviates from the previously mentioned standard format. Possibly, these BRIN are also not identical for the same school in different years.

valid.brin is a function that identifies (in)correct values for BRIN. This function is used to make a new variable in the data of the MIRROR-file (data$validbrin). valid.brin is sourced from a separate file.

source('E:/Google Drive/Promotie/Analyse/Teacher-attrition-with-MIRROR-data/validBRIN.R')  
data$validbrin <- lapply(data$BRIN, valid.brin)  
data$validbrin <- as.logical(data$validbrin)

The validBRIN funtion works, but it takes ca. 35 minutes to complete the operation.

## Valid BRIN in complete dataset

mytable <- table(data$validbrin, exclude = NULL)  
mytable  
prop.table(mytable)

|  |  |  |  |
| --- | --- | --- | --- |
|  | FALSE | TRUE | NA |
| n | 42243 | 6407015 | 1393 |
| proportion | 0.0065486414 | 0.9932354114 | 0.0002159472 |

## Valid BRIN per year

Invalid BRIN appear in 2008 and later years. It remains a small percentage of all observations, max 2%. In the following checks, only records of 2008 and later will be used.

mytable <- table(data$JAAR, data$validbrin, exclude = NULL)  
mytable  
prop.table(mytable,1)

## Valid BRIN per job (FUNGRP\_PVE1 and FUNGRP\_FMIX1)

Invalid BRIN are concentrated in non-teaching jobs, but are <5% of these records. For teachers, >99% of the records have a valid BRIN. This is checked for two different variables for job, with very similar results.

# FUNGRP\_PVE1  
mytable1 <- table(data$FUNGRP\_PVE1[data$JAAR >= 2008], data$validbrin[data$JAAR >= 2008], exclude = NULL)  
rownames(mytable1)<-c("Directie", "OP", "OOP", "OBP", "LIO", "<NA>")  
mytable1  
prop.table(mytable1,1)  
  
# FUNGRP\_FMIX1  
mytable2 <- table(data$FUNGRP\_FMIX1[data$JAAR >= 2008], data$validbrin[data$JAAR >= 2008], exclude = NULL)  
rownames(mytable2)<-c("Directie", "OP", "OOP", "OBP", "LIO", "<NA>")  
mytable2  
prop.table(mytable2,1)  
  
prop.table(mytable1,1) - prop.table(mytable2,1)

# Potential problems with linking to other data

To link the teachers of this dataset reliable to other data, some key variables have to be complete. In most cases these variables are enough to identify persons: year (JAAR), school (BRIN), date of birth (GEBDAT), sex (GESLACHT), job (FUNGRP\_PVE1). If a record has a missing value in one (or more) of these records, it is considered weak. In many cases, job might not be necessary for a reliable link, so a check with and without FUNGRP\_PVE1 is done.

13862 records (0.2%) have at least one missing value in the key variables necessary for linking, including the variable for job. If we exclude the variable for job, 5553 records (0.09%) with at least one missing key value remain. For further analysis in this paragraph, I use the more conservative estimate, which included the variable for job as source for potential problems with linking.

data$weaklink <- apply(data[, c("JAAR", "BRIN", "GEBDAT", "GESLACHT", "FUNGRP\_PVE1")], 1, anyNA)  
sum(data$weaklink)  
sum(data$weaklink)/length(data$weaklink)  
data$weaklink2 <- apply(data[, c("JAAR", "BRIN", "GEBDAT", "GESLACHT")], 1, anyNA)  
sum(data$weaklink2)  
sum(data$weaklink2)/length(data$weaklink2)

## Relation between 'weak' records and years

Records with at least one missing key value seem to concentrate in the years 2008 and later. 2010 has the most, but still only ca. 1,5% of the records of that year.

mytable <- table(data$JAAR, data$weaklink, exclude = NULL)  
mytable  
prop.table(mytable, 1)

## Relation between 'weak' records and job

A large proportion of the weak records have missing values for job: 9067 out of 13862 weak records (65%) have a missing value for the job variable. The records that don't have a missing value for job are almost never weak (<0,1%), with a slightly higher proportion of weak records (0,2%) with the teachers in training (LIO).

mytable <- table(data$FUNGRP\_PVE1, data$weaklink, exclude = NULL)  
rownames(mytable)<-c("Directie", "OP", "OOP", "OBP", "LIO", "<NA>")  
mytable  
prop.table(mytable, 1)

The weak records in 2010 (2010 has the most weak records, [check here](#weakyear)) are also almost completely caused by a missing value for the job variable (and possibly also other missing values). 5959 out of the 6062 (98%) weak records in 2010 have a missing value for the job variable.

mytable <- table(data$FUNGRP\_PVE1[data$JAAR==2010], data$weaklink[data$JAAR==2010], exclude = NULL)  
rownames(mytable)<-c("Directie", "OP", "OOP", "OBP", "LIO", "<NA>")  
mytable  
prop.table(mytable, 1)

## Relation between 'weak' records and invalid BRIN

There seems to be no obvious link between the invalid BRIN and the missing key values. 86,7% of records with a missing key value have a valid BRIN and 3,2% have an invalid BRIN. The remaining 10% of the records with missing key values has a missing BRIN (), which makes sense because a missing BRIN is one of the criteria to mark a record as problematic.

mytable <- table(data$validbrin, data$weaklink, exclude = NULL)  
mytable  
prop.table(mytable, 2)  
# mytable2 <- table(data$validbrin, data$weaklink)  
# mytable2  
# prop.table(mytable2, 2)

# Duplicate records

library(plyr)

# check for persons (id\_2015) with one or more recors with one or more NA's in variables used for linking to other data.

id\_data <- ddply(data1, "id\_2015", summarise, koppelprobleem\_record = max(koppelprobleem)) sum(id\_data$koppelprobleem\_record) # 16868 out of 629793 persons (2.68%)

# idem for most recent year (2014)

id\_data\_2014 <- ddply(data1[data1$JAAR==2014,], "id\_2015", summarise, koppelprobleem\_record = max(koppelprobleem)) sum(id\_data\_2014$koppelprobleem\_record) # 505 out of 332418 persons (0,15%)

# idem for most recent 6 years (2008 - 2014)

id\_data\_20092014 <- ddply(subset(data1, JAAR >= 2008), "id\_2015", summarise, koppelprobleem\_record = max(koppelprobleem)) sum(id\_data\_20092014$koppelprobleem\_record) # 16759 out of 478643 persons (3,50%)

head(data1[which(data1$JAAR==(2009:2014)), ]) head(data1[data1$JAAR==2014, ]) head(subset(data1, JAAR >= 2008))

min(c(T,T,F))

duplicated(testdata)

# Create variable for entry and exit into the labor force.

testdata2 <- ddply(testdata, "id\_2015", summarise, first\_year = min(JAAR)) # summarise function aggregates data frame by "id\_2015" testdata3 <- ddply(testdata, "id\_2015", summarise, last\_year = max(JAAR)) # user system elapsed # 2.52 0.03 2.55

testdata4 <- merge(testdata2, testdata3) testdata5 <- merge(testdata, testdata4)

head(testdata2) head(testdata3) head(testdata4) head(testdata5)

testdata6 <- ddply(testdata, c("id\_2015", "FUNGRP"), summarise, last\_year = max(JAAR)) # FUNGRP is used for MIRROR, a model for labour market estimates. Most likely the best variable to use to determine if someone is a teacher. head(testdata6)

# To do:

# 1. How to get the value labels out of the SPSS-file?

# The not-sophisticated way: <http://www.statmethods.net/input/valuelabels.html>

dataFUNGRP\_PVE1, levels = c(-1,1,2,3,4,5), labels = c("onbekend", "directie, management", "onderwijzend personeel, leraren", "onderwijsondersteunend personeel", "beheer- en administratief personeel", "leraren in opleiding")) # Dit geeft problemen bij het benoemen van NA's in eerdere syntax.

# Or, alternatively:

# dataFUNGRP\_PVE1,

# levels = c(1,2,3,4,5),

# labels = c("directie/management", "OP", "OOP", "OBP", "LIO"))

# 2. Convert some variables to nominal variables (factor?)

# 3. Create date variable out of GEBDAT

# 4. Fill missing GEBDAT is known within same id\_2015? Maybe not: id\_2015 might not be very reliable if date of birth is missing.

{r setup, include=FALSE} knitr::opts\_chunk$set(echo = TRUE)

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

{r cars} summary(cars)

## Including Plots

You can also embed plots, for example:

{r pressure, echo=FALSE} plot(pressure)

Note that the echo = FALSE parameter was added to the code chunk to prevent printing of the R code that generated the plot.