Make a fake variable using the CHIS Data Dictionary

# Intro

Get the rchisdd() function  
<https://github.com/mikejacktzen/ucla-chis-puf/blob/main/rchisdd/rchisdd.R>

source(paste0(getwd(),"/rchisdd.R"))  
args(rchisdd)

function (info\_dd, lgl\_floor = TRUE)   
NULL

Using the rchisdd() function to create a fake random chis dummy data variable, according to user provided information from the data dictionary.

#' @param info\_dd a data.frame that contains c('VALUE','FREQ') if categorical or c('N','MIN','MAX','MEAN') if count/continuous  
#' @param lgl\_floor TRUE/FALSE if you want the output values rounded to their integer floor  
#'  
#' @return a numeric vector of values generated according to the user supplied `info\_dd`

This is useful when there is a variable in the dac confidential data dictionary

<https://healthpolicy.ucla.edu/chis/data/confidential-data/Documents/2021/CHIS%202021%20Data%20Dictionary%20-%20Source%20-%20Adult%20Oct%202022.pdf>

but you do not find it in the dummy data file you have.

You can intentionally create an example place holder of that variable in your dummy data set with the rchisdd() function. After all, the dummy data set is a place holder for the real data in the dac.

The dummy variable you create will mimic the ‘name’ and the marginal distribution / frequency that you specify. This is information you find from the data dictionary.

Once in the dac, you can assume the variable exists in the confidential dac dataset and you do not need to create it inside the dac.

library(dplyr)

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':  
  
 filter, lag

The following objects are masked from 'package:base':  
  
 intersect, setdiff, setequal, union

library(haven)  
  
fp\_parent = getwd()  
fp\_dat\_dummy = paste0(fp\_parent,"\\ADULT\_STATA\_DUMMY\_2021\\Stata\\ADULT\_with\_format.dta")  
df\_dummy = haven::read\_dta(fp\_dat\_dummy) %>% rename\_all(toupper)

Note the number of rows in the dummy dataset. Use it as the overall sample size n\_ss

# sample size of dataset  
nrow(df\_dummy); n\_ss = 24453

[1] 24453

In the data dictionary:

A categorical variable will have a Value with its respective Frequency.

A continuous variable will have the eligible universe sample size N, Min, Max, and Mean.

Below, we show you how to use the above information to create a placeholder fake variable and attach it to your dummy dataset.

# Categorical

We will pretend the 2021 dummy data does **NOT** have the categorical **INTVLANG2** (even though it does).

df\_dummy$INTVLANG2 %>% summary

Min. 1st Qu. Median Mean 3rd Qu. Max.   
 1.000 1.000 1.000 1.135 1.000 8.000

INTVLANG2\_original = df\_dummy$INTVLANG2  
  
# remove INTVLANG2  
df\_dummy\_pretend = df\_dummy %>% select(-INTVLANG2)

For each data dictionary ‘value’, replicate that value ‘freq’ number of times.

Store the values you see from the data dictionary into a vector called value. Likewise, store its paired ‘frequency’ into a vector called freq.

info\_dd = data.frame(VALUE = c(1,2,3,4,5,8),  
 FREQ = c(22606, 1116,129,201,396,5))  
  
vec\_val\_freq = rchisdd(info\_dd)

Loading required package: purrr

# attach labels (if desired) not done here  
  
INTVLANG2\_fake = vec\_val\_freq  
  
'INTVLANG2' %in% names(df\_dummy\_pretend)

[1] FALSE

df\_dummy\_pretend$INTVLANG2 <- INTVLANG2\_fake  
'INTVLANG2' %in% names(df\_dummy\_pretend)

[1] TRUE

df\_dummy\_pretend$INTVLANG2 %>% table

.  
 1 2 3 4 5 8   
22606 1116 129 201 396 5

# Continuous

For a continuous variable, our function will squeeze/stretch a vector of values to your specified [min,max] range while approximately preserving your specified mean. It will try to solve for a power p that satisfies the above.

## Continuous: Universe is entire sample

We focus on the continuous variable SRAGE and we pretend SRAGE is not on the dummy data set.

# pretend   
SRAGE\_original = df\_dummy$SRAGE  
  
# remove SRAGE  
df\_dummy\_pretend = df\_dummy %>% select(-SRAGE)

For SRAGE, all N=24453 samples in the dummy data are eligible to take on values in the range [18,117] with a mean of 53.52 .

Save this information from the data dictionary into a data.frame

# continuous version  
# SRAGE  
  
# N MIN MAX MEAN from data dictionary  
info\_dd = data.frame(N = 24453,  
 MIN = 18,  
 MAX = 117,  
 MEAN = 53.52)  
  
vec\_val\_cont = rchisdd(info\_dd)

Rounding to the floor since lgl\_floor=TRUE

length(vec\_val\_cont)

[1] 24453

info\_dd$N

[1] 24453

n\_ss

[1] 24453

# note any difference between number sample size of dataset and eligible universe size  
n\_ss - info\_dd$N

[1] 0

# SRAGE is applicable to all n\_ss, so (n\_ss - pick\_params$N) == 0  
# do not need the final step to fill ddict value for the 'rest' of the (n\_ss - pick\_params$N) rows  
  
SRAGE\_fake = vec\_val\_cont  
  
'SRAGE' %in% names(df\_dummy\_pretend)

[1] FALSE

df\_dummy\_pretend$SRAGE <- SRAGE\_fake  
'SRAGE' %in% names(df\_dummy\_pretend)

[1] TRUE

df\_dummy\_pretend$SRAGE %>% summary

Min. 1st Qu. Median Mean 3rd Qu. Max.   
 18.00 44.00 52.00 53.02 61.00 117.00

## Continuous: Universe is a subset

We focus on the continuous variable AC210 and we pretend AC210 is not on the dummy data set.

# pretend   
AC210\_original = df\_dummy$AC210  
  
# remove AC210  
df\_dummy\_pretend = df\_dummy %>% select(-AC210)

Similar to the continuous example above, but the UNIVERSE information in the data dictionary should be reviewed and incorporated.

Extra end-step to append negative values for ineligible / non-universe cases

UNIVERSE: ADULTS WHO HAVE HAD AN ALCOHOLIC BEVERAGE IN PAST 30 DAYS

That is, an individual’s response in AC210 is non-negative when their response in AC208 == 1

# AC210 N MIN MAX MEAN from data dictionary  
info\_dd = data.frame(N = 14493,  
 MIN = 0,  
 MAX = 90,  
 MEAN = 2.14)  
  
vec\_val\_cont = rchisdd(info\_dd)

Rounding to the floor since lgl\_floor=TRUE

length(vec\_val\_cont)

[1] 14493

info\_dd$N

[1] 14493

n\_ss

[1] 24453

# note any difference between number sample size of dataset and eligible universe size  
n\_ss - info\_dd$N

[1] 9960

Notice that info\_dd$N is 14493 which is less than the dummy data number of rows n\_ss=24453

That is the summary information of the data dictionary for AC210 applies to the 14493 eligible cases, while the remaining cases n\_ss - info\_dd$N needs to take on a value for ineligible.

### Fill in non-Universe Values

# do need the final step to fill ddict value for the 'rest' of the (n\_ss - pick\_params$N) non-universe rows  
  
# fake drawn values for eligible cases  
# rep -1 for the ineligible  
  
AC210\_fake = c(vec\_val\_cont,   
 rep(x=-1,times=(n\_ss - info\_dd$N))   
 )  
  
'AC210' %in% names(df\_dummy\_pretend)

[1] FALSE

df\_dummy\_pretend$AC210 <- AC210\_fake  
'AC210' %in% names(df\_dummy\_pretend)

[1] TRUE

df\_dummy\_pretend$AC210 %>% summary

Min. 1st Qu. Median Mean 3rd Qu. Max.   
-1.0000 -1.0000 0.0000 0.6496 0.0000 90.0000

df\_dummy\_pretend %>% filter(AC210 >= 0) %>% select(AC210) %>% unlist %>% summary

Min. 1st Qu. Median Mean 3rd Qu. Max.   
 0.000 0.000 0.000 1.783 2.000 90.000

df\_dummy\_pretend %>% filter(AC210 < 0) %>% select(AC210) %>% unlist %>% table

.  
 -1   
9960

# Optional Labels

To add your own labels, see haven::labelled()

<https://haven.tidyverse.org/reference/labelled.html>

library(haven)  
?haven::labelled()