**Background**

My colleague wanted to be able to do some simple analysis around health  
care using the Centers for Disease Control and Prevention  
([https://www.cdc.gov](https://www.cdc.gov/)), National Center for Health Statistics  
(<https://www.cdc.gov/nchs/index.htm>), National Health Interview Survey  
(<https://www.cdc.gov/nchs/nhis/nhis_2016_data_release.htm>). They  
wanted a series of cross tabulated sets of summary data for variable  
pairings (for example whether or not the respondent had a formal health  
care provider by region of the country). They wanted one Excel  
“workbook” with 12 worksheets each one of which was the summary of  
counts for a pair of variables. From there they could use Excel’s native  
plotting tools to make the graphs they needed.

A little sleuthing around CRAN helped me discover openxlsx which seems  
to be quite active, well maintained, and have a variety of features I  
would need. In my case that involved a download and install first (but  
I’ll comment it out in this version for you). As long as I’m at it  
I’ll load dplyr and ggplot2 (they’ll figure more prominently in my  
next post).

knitr::opts\_chunk$set(echo = TRUE, warning = FALSE)

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(ggplot2)

theme\_set(theme\_bw()) # set theme to my personal preference

# install.packages("openxlsx")

require(openxlsx)

## Loading required package: openxlsx

**Download and structure the data**

The data my colleague wanted was from 2014 and she was kind enough to  
provide the URL for a compressed zipfile  
<ftp://ftp.cdc.gov/pub/Health_Statistics/NCHS/Datasets/NHIS/2016/personsxcsv.zip>.  
Full documentation about the file and its contents and the methodology  
is here <https://www.cdc.gov/nchs/nhis/nhis_2016_data_release.htm>.

After downloading and uncompressing it into a local project directory  
(it’s about 77Mb of data, with over 100,000 rows and more than 600  
columns). I could go to work on processing it to get what was really  
needed for my colleague.

FullFile <- read.csv(file = "personsx.csv")

dim(FullFile)

## [1] 103789 606

# 606 variables too many whittle it down with wild cards

**A little dplyr to make our life easier**

While strictly speaking nothing in the next few steps requires dplyr  
(they can all be done in base R) I will showcase a couple ways dplyr  
can make your data analysis faster and easier.

First off, call me old-fashioned but, I abhor having lots of data in  
working memory that I know for a fact I’ll never use. While my Mac has  
plenty of space let’s showcase dplyr’s ability to help us rapidly  
reduce down to a more manageable dataset. A quick look at the data  
dictionary provided by the **CDC** shows we’ll never use any of the  
variables in the dataset that start with “L” or with “INT” so let’s make  
them go away.

FullFile <- select(FullFile, -starts\_with("L"))

FullFile <- select(FullFile, -starts\_with("INT"))

dim(FullFile)

## [1] 103789 289

# 289 is still big but ...

The other thing we notice is that the variables are all coded as integers when  
we know they are truly factors. Since we know we’re going to analyze  
them as factors let’s take a moment to recode which will also mean an  
opportunity to provide more user friendly labels so we’re not constantly  
referring to the code book to see what “1” really represents in the  
data.

The function is recode\_factor and allows us to map factor labels onto  
the integers to provide a more useful dataset.

FullFile$REGION <- recode\_factor(FullFile$REGION,

`1` = "Northeast",

`2` = "Midwest",

`3` = "South",

`4` = "West")

FullFile$SEX <- recode\_factor(FullFile$SEX, `1` = "Male", `2` = "Female")

FullFile$RACERPI2 <- recode\_factor(FullFile$RACERPI2,

`1` = "White only",

`2` = "Black/African American only",

`3` = "AIAN only",

`4` = "Asian only",

`5` = "Race group not releasable",

`6` = "Multiple race")

FullFile$PDMED12M <- recode\_factor(FullFile$PDMED12M, `1` = "Yes", `2` = "No")

## Warning: Unreplaced values treated as NA as .x is not compatible. Please

## specify replacements exhaustively or supply .default

summary(FullFile$PDMED12M)

## Yes No NA's

## 6744 96986 59

Uh oh, what are we being warned about? recode\_factor has a nice  
rational solution for cases where we don’t specify all the possible  
choices for a variable (see ?recode\_factor) it simply assigns them an  
NA value. In our case that’s just what we want.

Finishing the rest (and suppressing the  
warnings)…

FullFile$PNMED12 <- recode\_factor(FullFile$PNMED12, `1` = "Yes", `2` = "No")

FullFile$PNMED12M <- recode\_factor(FullFile$PNMED12M, `1` = "Yes", `2` = "No")

FullFile$NOTCOV <- recode\_factor(FullFile$NOTCOV, `1` = "Not covered", `2` = "Covered")

FullFile$COVER <- recode\_factor(FullFile$COVER,

`1` = "Private",

`2` = "Medicaid and other public",

`3` = "Other coverage",

`4` = "Uninsured",

`5` = "Do not know")

FullFile$PLNWRKS1 <- recode\_factor(FullFile$PLNWRKS1,

`1` = "Through employer",

`2` = "Through union",

`3` = "Through workplace, but don't know if employer or union",

`4` = "Through workplace, self-employed or professional association",

`5` = "Purchased directly",

`6` = "Through Healthcare.gov or the Affordable Care Act",

`7` = "Through a state/local government or community program",

`8` = "Other",

`9` = "Through school",

`10` = "Through parents",

`11` = "Through relative other than parents")

FullFile$HCSPFYR <- recode\_factor(FullFile$HCSPFYR,

`0` = "Zero",

`1` = "Less than $500",

`2` = "$500 - $1,999",

`3` = "$2,000 - $2,999",

`4` = "$3,000 - $4,999",

`5` = "$5,000 or more")

FullFile$MEDBILL <- recode\_factor(FullFile$MEDBILL, `1` = "Yes", `2` = "No")

FullFile$MEDBPAY <- recode\_factor(FullFile$MEDBPAY, `1` = "Yes", `2` = "No")

# I am thinking that earnings can be collapsed into three attributes: low; medium; high

FullFile$EARNINGS <- recode\_factor(FullFile$ERNYR,

`1` = "$01-$34,999",

`2` = "$01-$34,999",

`3` = "$01-$34,999",

`4` = "$01-$34,999",

`5` = "$01-$34,999",

`6` = "$01-$34,999",

`7` = "$35,000-$74,999",

`8` = "$35,000-$74,999",

`9` = "$35,000-$74,999",

`10` = "$35,000-$74,999",

`11` = "$75,000 and over")

# Education the same: low; medium; high

FullFile$EDUCATION <- recode\_factor(FullFile$EDUC1,

`0` = "HSchool Grad or less",

`1` = "HSchool Grad or less",

`2` = "HSchool Grad or less",

`3` = "HSchool Grad or less",

`4` = "HSchool Grad or less",

`5` = "HSchool Grad or less",

`6` = "HSchool Grad or less",

`7` = "HSchool Grad or less",

`8` = "HSchool Grad or less",

`9` = "HSchool Grad or less",

`10` = "HSchool Grad or less",

`11` = "HSchool Grad or less",

`12` = "HSchool Grad or less",

`13` = "HSchool Grad or less",

`14` = "HSchool Grad or less",

`15` = "Some college or AA degree",

`16` = "Some college or AA degree",

`17` = "Some college or AA degree",

`18` = "Bachelor's or higher",

`19` = "Bachelor's or higher",

`20` = "Bachelor's or higher",

`21` = "Bachelor's or higher")

# Age also collapsed: low; medium; high

FullFile$AGE <- cut(FullFile$AGE\_P,

breaks = c(-Inf, 18, 61, Inf),

labels = c("Less than 18", "18 to 60", "More than 60"),

right = FALSE)

table(FullFile$EDUC1,FullFile$EDUCATION)

##

## HSchool Grad or less Some college or AA degree Bachelor's or higher

## 0 3424 0 0

## 1 1679 0 0

## 2 1633 0 0

## 3 1787 0 0

## 4 1716 0 0

## 5 1822 0 0

## 6 2726 0 0

## 7 1860 0 0

## 8 2644 0 0

## 9 2908 0 0

## 10 2884 0 0

## 11 3186 0 0

## 12 1774 0 0

## 13 2290 0 0

## 14 18413 0 0

## 15 0 14822 0

## 16 0 5413 0

## 17 0 2966 0

## 18 0 0 13883

## 19 0 0 5956

## 20 0 0 955

## 21 0 0 1018

## 96 0 0 0

## 97 0 0 0

## 98 0 0 0

## 99 0 0 0

One easy way to see whether your recoding has had the desired effect is  
to make a simple table that maps the original values to the new values.  
That’s what I’ve done here for **EDUCATION** which shows the collapse of  
categories and the fact that values like 99 have been mapped to NA.

Finally let’s grab just the variables we really need including the newly  
recoded versions and make them a new dataset and take the full version  
out of  
memory.

OfInterest <- select(FullFile, AGE, REGION, SEX, EDUCATION, EARNINGS, PDMED12M, PNMED12M, NOTCOV, MEDBILL)

str(OfInterest)

## 'data.frame': 103789 obs. of 9 variables:

## $ AGE : Factor w/ 3 levels "Less than 18",..: 2 2 2 1 1 2 2 2 3 2 ...

## $ REGION : Factor w/ 4 levels "Northeast","Midwest",..: 3 4 4 4 4 4 3 3 4 3 ...

## $ SEX : Factor w/ 2 levels "Male","Female": 1 2 1 1 2 2 2 1 1 1 ...

## $ EDUCATION: Factor w/ 3 levels "HSchool Grad or less",..: 2 1 3 1 NA 3 1 1 3 2 ...

## $ EARNINGS : Factor w/ 3 levels "$01-$34,999",..: 1 2 2 NA NA 3 1 NA 2 3 ...

## $ PDMED12M : Factor w/ 2 levels "Yes","No": 2 1 2 2 2 2 2 2 2 2 ...

## $ PNMED12M : Factor w/ 2 levels "Yes","No": 2 2 2 2 2 2 2 2 2 2 ...

## $ NOTCOV : Factor w/ 2 levels "Not covered",..: 2 2 2 2 2 2 2 2 2 2 ...

## $ MEDBILL : Factor w/ 2 levels "Yes","No": 2 1 1 1 1 2 2 2 2 2 ...

summary(OfInterest)

## AGE REGION SEX

## Less than 18:25680 Northeast:16883 Male :50257

## 18 to 60 :58025 Midwest :20507 Female:53532

## More than 60:20084 South :35826

## West :30573

## EDUCATION EARNINGS PDMED12M

## HSchool Grad or less :50746 $01-$34,999 :20390 Yes : 6744

## Some college or AA degree:23201 $35,000-$74,999 :12927 No :96986

## Bachelor's or higher :21812 $75,000 and over: 6413 NA's: 59

## NA's : 8030 NA's :64059

## PNMED12M NOTCOV MEDBILL

## Yes : 4978 Not covered:10506 Yes :16282

## No :98741 Covered :92181 No :87096

## NA's: 70 NA's : 1102 NA's: 411

##

rm(FullFile)

**Driving Excel**

Everything we have done so far has been in preparation for actually  
providing my colleague the data she wanted in Excel so now on to the  
main event. In this post I am deliberately going to **NOT** build  
functions or use ggplot to make the graphs she wanted. Those will be  
topics for future posts.

You may have noticed that a table is almost exactly what she wants. If  
the variables of interest are AGE and whether or not the person has  
health care coverage (NOTCOV) then the command would be  
table(OfInterest$AGE,OfInterest$NOTCOV) and we have results we need.  
The openxlsx package provides a write.xlsx function that will accept the  
table command and produce a properly formatted workbook as output. The  
code below will produce SimpleExcelExample.xlsx in your working  
directory. The number of options is legion and ?write.xlsx will  
display them for you.

table(OfInterest$AGE,OfInterest$NOTCOV)

##

## Not covered Covered

## Less than 18 1312 24193

## 18 to 60 8730 48484

## More than 60 464 19504

write.xlsx(table(OfInterest$AGE,OfInterest$NOTCOV), file = "SimpleExcelExample.xlsx")

**Finishing up the manual way**

The last thing I will describe in this post is how to make one workbook  
with multiple sheets. Where each sheet represents one variable pairing  
(e.g. AGE by NOTCOV). We’ll even label the tabs (worksheets) in a  
thoughtful way. So my colleague can easily see which sheet corresponds  
to which variable pairing.

The code below follows this pattern

1. Create a new empty workbook object wb <- createWorkbook()
2. Invent a name for the **tab** or worksheet inside the workbook  
   NameofSheet
3. Make a table for a pair of variables TheData <-  
   table(OfInterest$EDUCATION,OfInterest$NOTCOV)
4. Add a worksheet (tab) into the workbook addWorksheet
5. Write the table we made onto the worksheet with writeData
6. Repeat steps 2 through 5 **12** times

### Manual and painful way

## Create a new workbook

wb <- createWorkbook()

# education by each of the other 4 variables

NameofSheet <- "CoverageByEducation"

TheData <- table(OfInterest$EDUCATION,OfInterest$NOTCOV)

addWorksheet(wb = wb, sheetName = NameofSheet)

writeData(wb = wb, sheet = NameofSheet, x = TheData, borders = "n")

NameofSheet <- "MedbillByEducation"

TheData <- table(OfInterest$EDUCATION,OfInterest$MEDBILL)

addWorksheet(wb = wb, sheetName = NameofSheet)

writeData(wb = wb, sheet = NameofSheet, x = TheData, borders = "n")

NameofSheet <- "PDMED12MByEducation"

TheData <- table(OfInterest$EDUCATION,OfInterest$PDMED12M)

addWorksheet(wb = wb, sheetName = NameofSheet)

writeData(wb = wb, sheet = NameofSheet, x = TheData, borders = "n")

NameofSheet <- "PNMED12MByEducation"

TheData <- table(OfInterest$EDUCATION,OfInterest$PNMED12M)

addWorksheet(wb = wb, sheetName = NameofSheet)

writeData(wb = wb, sheet = NameofSheet, x = TheData, borders = "n")

# earnings by each of the other 4 variables

NameofSheet <- "CoverageByEarnings"

TheData <- table(OfInterest$EARNINGS,OfInterest$NOTCOV)

addWorksheet(wb = wb, sheetName = NameofSheet)

writeData(wb = wb, sheet = NameofSheet, x = TheData, borders = "n")

NameofSheet <- "MedbillByEarnings"

TheData <- table(OfInterest$EARNINGS,OfInterest$MEDBILL)

addWorksheet(wb = wb, sheetName = NameofSheet)

writeData(wb = wb, sheet = NameofSheet, x = TheData, borders = "n")

NameofSheet <- "PDMED12MByEarnings"

TheData <- table(OfInterest$EARNINGS,OfInterest$PDMED12M)

addWorksheet(wb = wb, sheetName = NameofSheet)

writeData(wb = wb, sheet = NameofSheet, x = TheData, borders = "n")

NameofSheet <- "PNMED12MByEarnings"

TheData <- table(OfInterest$EARNINGS,OfInterest$PNMED12M)

addWorksheet(wb = wb, sheetName = NameofSheet)

writeData(wb = wb, sheet = NameofSheet, x = TheData, borders = "n")

# age by each of the other 4 variables

NameofSheet <- "CoverageByAge"

TheData <- table(OfInterest$AGE,OfInterest$NOTCOV)

addWorksheet(wb = wb, sheetName = NameofSheet)

writeData(wb = wb, sheet = NameofSheet, x = TheData, borders = "n")

NameofSheet <- "MedbillByAge"

TheData <- table(OfInterest$AGE,OfInterest$MEDBILL)

addWorksheet(wb = wb, sheetName = NameofSheet)

writeData(wb = wb, sheet = NameofSheet, x = TheData, borders = "n")

NameofSheet <- "PDMED12MByAge"

TheData <- table(OfInterest$AGE,OfInterest$PDMED12M)

addWorksheet(wb = wb, sheetName = NameofSheet)

writeData(wb = wb, sheet = NameofSheet, x = TheData, borders = "n")

NameofSheet <- "PNMED12MByAge"

TheData <- table(OfInterest$AGE,OfInterest$PNMED12M)

addWorksheet(wb = wb, sheetName = NameofSheet)

writeData(wb = wb, sheet = NameofSheet, x = TheData, borders = "n")

#

saveWorkbook(wb, "BetterExcelExample.xlsx", overwrite = TRUE) ## save to working directory

**All done (not really)**

Hopefully this post helps you understand how to use the openxlsx  
package to have R drive Excel to help you with your data analysis. In  
my next post I’ll build on this scaffolding to discuss how to make these  
very same graphs in ggplot2 (which IMHO runs circles around Excel for  
scientific plotting), as well as making this all more efficient through  
the use of functions to take care of some of the more repetitive chores.