Data Science for Biological, Medical and Health Research: Notes for 432

Thomas E. Love, Ph.D.

Built 2021-01-29 17:53:59

Contents

Introduction

These Notes provide a series of examples using R to work through issues that are likely to come up in PQHS/CRSP/MPHP 432.

While these Notes share some of the features of a textbook, they are neither comprehensive nor completely original. The main purpose is to give students in 432 a set of common materials on which to draw during the course. In class, we will sometimes:

- reiterate points made in this document,
- amplify what is here,
- simplify the presentation of things done here,
- use new examples to show some of the same techniques,
- refer to issues not mentioned in this document,

but what we don't (always) do is follow these notes very precisely. We assume instead that you will read the materials and try to learn from them, just as you will attend classes and try to learn from them. We welcome feedback of all kinds on this document or anything else via Piazza.

What you will mostly find are brief explanations of a key idea or summary, accompanied (most of the time) by R code and a demonstration of the results of applying that code.

Everything you see here is available to you as HTML or PDF. You will also have access to the R Markdown files, which contain the code which generates everything in the document, including all of the R results. We will demonstrate the use of R Markdown (this document is generated with the additional help of an R package called bookdown) and R Studio (the "program" which we use to interface with the R language) in class.

To download the data and R code related to these notes, visit the appropriate link at the 432 course website.

R Packages used in these notes

Here, we'll load in some packages used in these notes. The list of R Packages we will use in 432 is more extensive, and is available on our course website.

```
library(here)
library(janitor)
library(magrittr)

library(tableone)

library(broom)
library(haven)
library(janitor)
library(patchwork)
library(Hmisc)
library(rms)

library(visdat)
library(naniar)
library(caret)
library(simputation)
library(car)
library(mice)
library(leaps)
library(lars)
library(Epi)
library(pROC)
library(ROCR)
library(VGAM)
library(ggridges)
library(pander)
library(arm)
```

```
library(survival)
library(survminer)
library(kableExtra)

## and of course, we conclude with...

library(tidymodels)
library(tidyverse)
```

```
specify_decimal <- function(x, k) format(round(x, k), nsmall=k)
```

General Theme for ggplot work

```
theme_set(theme_bw())
```

Data used in these notes

All data sets used in these notes are available on our Data and Code website.

Dr. Love is in the process of moving all of the data loads below to their individual chapters.

```
prost <- read_csv("data/prost.csv")
pollution <- read_csv("data/pollution.csv")

bonding <- read_csv("data/bonding.csv")
cortisol <- read_csv("data/cortisol.csv")
emphysema <- read_csv("data/emphysema.csv")
resect <- read_csv("data/resect.csv")
colscr <- read_csv("data/screening.csv")
colscr2 <- read_csv("data/screening2.csv")
authorship <- read_csv("data/authorship.csv")
hem <- read_csv("data/hem.csv")
leukem <- read_csv("data/leukem.csv")
```

Chapter 1

Building Table 1

Many scientific articles involve direct comparison of results from various exposures, perhaps treatments. In 431, we studied numerous methods, including various sorts of hypothesis tests, confidence intervals, and descriptive summaries, which can help us to understand and compare outcomes in such a setting. One common approach is to present what's often called Table 1. Table 1 provides a summary of the characteristics of a sample, or of groups of samples, which is most commonly used to help understand the nature of the data being compared.

1.1 Data load

Let's load two data sets for this Chapter. All data sets used in these notes are available on our Data and Code website.

```
fakestroke <- read_csv("data/fakestroke.csv")
```

```
-- Column specification --------------------------------------------------------
cols(
  studyid = col_character(),
  trt = col_character(),
  age = col_double(),
  sex = col_character(),
  nihss = col_double(),
  location = col_character(),
  hx.isch = col_character(),
  afib = col_double(),
  dm = col_double(),
  mrankin = col_character(),
  sbp = col_double(),
```

```
  iv.altep = col_character(),
  time.iv = col_double(),
  aspects = col_double(),
  ia.occlus = col_character(),
  extra.ica = col_double(),
  time.rand = col_double(),
  time.punc = col_double()
)
```

```
bloodbrain <- read_csv("data/bloodbrain.csv")
```

```
-- Column specification --------------------------------------------------------
cols(
  case = col_double(),
  brain = col_double(),
  liver = col_double(),
  tlratio = col_double(),
  solution = col_character(),
  sactime = col_double(),
  postin = col_double(),
  sex = col_character(),
  wt.init = col_double(),
  wt.loss = col_double(),
  wt.tumor = col_double()
)
```

1.2 Two examples from the *New England Journal of Medicine*

1.2.1 A simple Table 1

Table 1 is especially common in the context of clinical research. Consider the excerpt below, from a January 2015 article in the *New England Journal of Medicine* (Tolaney et al. 2015).

Table 1. Baseline Characteristics of the Patients.*

Characteristic	Patients (N=406)
	no. (%)
Age group	
<50 yr	132 (32.5)
50–59 yr	137 (33.7)
60–69 yr	96 (23.6)
≥70 yr	41 (10.1)
Sex	
Female	405 (99.8)
Male	1 (0.2)
Race†	
White	351 (86.5)
Black	28 (6.9)
Asian	11 (2.7)
Other	16 (3.9)

This (partial) table reports baseline characteristics on age group, sex and race, describing 406 patients with HER2-positive[1] invasive breast cancer that began the protocol therapy. Age, sex and race (along with severity of illness) are the most commonly identified characteristics in a Table 1.

In addition to the measures shown in this excerpt, the full Table also includes detailed information on the primary tumor for each patient, including its size, nodal status and histologic grade. Footnotes tell us that the percentages shown are subject to rounding, and may not total 100, and that the race information was self-reported.

1.2.2 A group comparison

A more typical Table 1 involves a group comparison, for example in this excerpt from Roy et al. (2008). This Table 1 describes a multi-center randomized clinical trial comparing two different approaches to caring for patients with heart failure and atrial fibrillation[2].

[1] HER2 = human epidermal growth factor receptor type 2. Over-expression of this occurs in 15-20% of invasive breast cancers, and has been associated with poor outcomes.

[2] The complete Table 1 appears on pages 2668-2669 of Roy et al. (2008), but I have only reproduced the first page and the footnote in this excerpt.

Table 1. Baseline Characteristics of the Patients.*

Variable	Rhythm-Control Group (N=682)	Rate-Control Group (N=694)
Male sex (%)	78	85
Age (yr)	66±11	67±11
Body-mass index†	27.8±5.4	28.0±5.1
Nonwhite race (%)‡	16	13
NYHA class III or IV (%)		
At baseline	32	31
During previous 6 mo	76	76
Predominant cardiac diagnosis (%)§		
Coronary artery disease	48	48
Valvular heart disease	5	5
Nonischemic cardiomyopathy	36	39
Congenital heart disease	1	1
Hypertensive heart disease	10	7

The article provides percentages, means and standard deviations across groups, but note that it does not provide p values for the comparison of baseline characteristics. This is a common feature of NEJM reports on randomized clinical trials, where we anticipate that the two groups will be well matched at baseline. Note that the patients in this study were *randomly* assigned to either the rhythm-control group or to the rate-control group, using blocked randomization stratified by study center.

1.3 The MR CLEAN trial

Berkhemer et al. (2015) reported on the MR CLEAN trial, involving 500 patients with acute ischemic stroke caused by a proximal intracranial arterial occlusion. The trial was conducted at 16 medical centers in the Netherlands, where 233 were randomly assigned to the intervention (intraarterial treatment plus usual care) and 267 to control (usual care alone.) The primary outcome was the modified Rankin scale score at 90 days; this categorical scale measures functional outcome, with scores ranging from 0 (no symptoms) to 6 (death). The fundamental conclusion of Berkhemer et al. (2015) was that in patients with acute ischemic stroke caused by a proximal intracranial occlusion of the anterior circulation, intraarterial treatment administered within 6 hours after stroke onset was effective and safe.

Here's the Table 1 from Berkhemer et al. (2015).

Table 1. Baseline Characteristics of the 500 Patients.*

Characteristic	Intervention (N=233)	Control (N=267)
Age — yr		
Median	65.8	65.7
Interquartile range	54.5–76.0	55.5–76.4
Male sex — no. (%)	135 (57.9)	157 (58.8)
NIHSS score†		
Median (interquartile range)	17 (14–21)	18 (14–22)
Range	3–30	4–38
Location of stroke in left hemisphere — no. (%)	116 (49.8)	153 (57.3)
History of ischemic stroke — no. (%)	29 (12.4)	25 (9.4)
Atrial fibrillation — no. (%)	66 (28.3)	69 (25.8)
Diabetes mellitus — no. (%)	34 (14.6)	34 (12.7)
Prestroke modified Rankin scale score — no. (%)‡		
0	190 (81.5)	214 (80.1)
1	21 (9.0)	29 (10.9)
2	12 (5.2)	13 (4.9)
>2	10 (4.3)	11 (4.1)
Systolic blood pressure — mm Hg§	146±26.0	145±24.4
Treatment with IV alteplase — no. (%)	203 (87.1)	242 (90.6)
Time from stroke onset to start of IV alteplase — min		
Median	85	87
Interquartile range	67–110	65–116
ASPECTS — median (interquartile range)¶	9 (7–10)	9 (8–10)
Intracranial arterial occlusion — no./total no. (%)‖		
Intracranial ICA	1/233 (0.4)	3/266 (1.1)
ICA with involvement of the M1 middle cerebral artery segment	59/233 (25.3)	75/266 (28.2)
M1 middle cerebral artery segment	154/233 (66.1)	165/266 (62.0)
M2 middle cerebral artery segment	18/233 (7.7)	21/266 (7.9)
A1 or A2 anterior cerebral artery segment	1/233 (0.4)	2/266 (0.8)
Extracranial ICA occlusion — no./total no. (%)‖**	75/233 (32.2)	70/266 (26.3)
Time from stroke onset to randomization — min††		
Median	204	196
Interquartile range	152–251	149–266
Time from stroke onset to groin puncture — min		
Median	260	NA
Interquartile range	210–313	

The Table was accompanied by the following notes.

* The intervention group was assigned to intraarterial treatment plus usual care, and the control group was assigned to usual care alone. Plus–minus values are means ±SD. ICA denotes internal carotid artery, IV intravenous, and NA not applicable.

† Scores on the National Institutes of Health Stroke Scale (NIHSS) range from 0 to 42, with higher scores indicating more severe neurologic deficits. The NIHSS is a 15-item scale, and values for 30 of the 7500 items were missing (0.4%). The highest number of missing items for a single patient was 6.

‡ Scores on the modified Rankin scale of functional disability range from 0 (no symptoms) to 6 (death). A score of 2 or less indicates functional independence.

§ Data on systolic blood pressure at baseline were missing for one patient assigned to the control group.

¶ The Alberta Stroke Program Early Computed Tomography Score (ASPECTS) is a measure of the extent of stroke. Scores ranges from 0 to 10, with higher scores indicating fewer early ischemic changes. Scores were not available for four patients assigned to the control group: noncontrast computed tomography was not performed in one patient, and three patients had strokes in the territory of the anterior cerebral artery.

‖ Vessel imaging was not performed in one patient in the control group, so the level of occlusion was not known.

** Extracranial ICA occlusions were reported by local investigators.

†† Data were missing for two patients in the intervention group.

1.4 Simulated `fakestroke` data

Consider the simulated data, available on our Data and Code website in the `fakestroke.csv` file, which I built to let us mirror the Table 1 for MR CLEAN (Berkhemer et al. 2015). The `fakestroke.csv` file contains the following 18 variables for 500 patients.

Variable	Description
`studyid`	Study ID # (z001 through z500)
`trt`	Treatment group (Intervention or Control)
`age`	Age in years
`sex`	Male or Female
`nihss`	NIH Stroke Scale Score (can range from 0-42; higher scores indicate more severe neurological deficits)
`location`	Stroke Location - Left or Right Hemisphere
`hx.isch`	History of Ischemic Stroke (Yes/No)
`afib`	Atrial Fibrillation (1 = Yes, 0 = No)
`dm`	Diabetes Mellitus (1 = Yes, 0 = No)
`mrankin`	Pre-stroke modified Rankin scale score (0, 1, 2 or > 2) indicating functional disability - complete range is 0 (no symptoms) to 6 (death)
`sbp`	Systolic blood pressure, in mm Hg
`iv.altep`	Treatment with IV alteplase (Yes/No)
`time.iv`	Time from stroke onset to start of IV alteplase (minutes) if iv.altep=Yes
`aspects`	Alberta Stroke Program Early Computed Tomography score, which measures extent of stroke from 0 - 10; higher scores indicate fewer early ischemic changes
`ia.occlus`	Intracranial arterial occlusion, based on vessel imaging - five categories[3]
`extra.ica`	Extracranial ICA occlusion (1 = Yes, 0 = No)
`time.rand`	Time from stroke onset to study randomization, in minutes
`time.punc`	Time from stroke onset to groin puncture, in minutes (only if Intervention)

Here's a quick look at the simulated data in `fakestroke`.

```
fakestroke
```

```
# A tibble: 500 x 18
   studyid trt     age sex   nihss location hx.isch  afib    dm mrankin   sbp
   <chr>   <chr> <dbl> <chr> <dbl> <chr>    <chr>   <dbl> <dbl> <chr>   <dbl>
```

[3] The five categories are Intracranial ICA, ICA with involvement of the M1 middle cerebral artery segment, M1 middle cerebral artery segment, M2 middle cerebral artery segment, A1 or A2 anterior cerebral artery segment

```
 1 z001    Cont~    53 Male     21 Right    No          0     0 2         127
 2 z002    Inte~    51 Male     23 Left     No          1     0 0         137
 3 z003    Cont~    68 Fema~    11 Right    No          0     0 0         138
 4 z004    Cont~    28 Male     22 Left     No          0     0 0         122
 5 z005    Cont~    91 Male     24 Right    No          0     0 0         162
 6 z006    Cont~    34 Fema~    18 Left     No          0     0 2         166
 7 z007    Inte~    75 Male     25 Right    No          0     0 0         140
 8 z008    Cont~    89 Fema~    18 Right    No          0     0 0         157
 9 z009    Cont~    75 Male     25 Left     No          1     0 2         129
10 z010    Inte~    26 Fema~    27 Right    No          0     0 0         143
# ... with 490 more rows, and 7 more variables: iv.altep <chr>, time.iv <dbl>,
#   aspects <dbl>, ia.occlus <chr>, extra.ica <dbl>, time.rand <dbl>,
#   time.punc <dbl>
```

1.5 Building Table 1 for `fakestroke`: Attempt 1

Our goal, then, is to take the data in `fakestroke.csv` and use it to generate a Table 1 for the study that compares the 233 patients in the Intervention group to the 267 patients in the Control group, on all of the other variables (except study ID #) available. I'll use the `tableone` package of functions available in R to help me complete this task. We'll make a first attempt, using the `CreateTableOne` function in the `tableone` package. To use the function, we'll need to specify:

- the `vars` or variables we want to place in the rows of our Table 1 (which will include just about everything in the `fakestroke` data except the `studyid` code and the `trt` variable for which we have other plans, and the `time.punc` which applies only to subjects in the Intervention group.)
  - A useful trick here is to use the `dput` function, specifically something like `dput(names(fakestroke))` can be used to generate a list of all of the variables included in the `fakestroke` tibble, and then this can be copied and pasted into the `vars` specification, saving some typing.
- the `strata` which indicates the levels want to use in the columns of our Table 1 (for us, that's `trt`)

```
fs.vars <- c("age", "sex", "nihss", "location",
          "hx.isch", "afib", "dm", "mrankin", "sbp",
          "iv.altep", "time.iv", "aspects",
          "ia.occlus", "extra.ica", "time.rand")

fs.trt <- c("trt")

att1 <- CreateTableOne(data = fakestroke,
                       vars = fs.vars,
                       strata = fs.trt)
print(att1)
```

	Stratified by trt			
	Control	Intervention	p	test
n	267	233		
age (mean (SD))	65.38 (16.10)	63.93 (18.09)	0.343	
sex = Male (%)	157 (58.8)	135 (57.9)	0.917	
nihss (mean (SD))	18.08 (4.32)	17.97 (5.04)	0.787	
location = Right (%)	114 (42.7)	117 (50.2)	0.111	
hx.isch = Yes (%)	25 (9.4)	29 (12.4)	0.335	
afib (mean (SD))	0.26 (0.44)	0.28 (0.45)	0.534	
dm (mean (SD))	0.13 (0.33)	0.12 (0.33)	0.923	
mrankin (%)			0.922	
> 2	11 (4.1)	10 (4.3)		
0	214 (80.1)	190 (81.5)		
1	29 (10.9)	21 (9.0)		
2	13 (4.9)	12 (5.2)		
sbp (mean (SD))	145.00 (24.40)	146.03 (26.00)	0.647	
iv.altep = Yes (%)	242 (90.6)	203 (87.1)	0.267	
time.iv (mean (SD))	87.96 (26.01)	98.22 (45.48)	0.003	
aspects (mean (SD))	8.65 (1.47)	8.35 (1.64)	0.033	
ia.occlus (%)			0.795	
A1 or A2	2 (0.8)	1 (0.4)		
ICA with M1	75 (28.2)	59 (25.3)		
Intracranial ICA	3 (1.1)	1 (0.4)		
M1	165 (62.0)	154 (66.1)		
M2	21 (7.9)	18 (7.7)		
extra.ica (mean (SD))	0.26 (0.44)	0.32 (0.47)	0.150	
time.rand (mean (SD))	213.88 (70.29)	202.51 (57.33)	0.051	

1.5.1 Some of this is very useful, and other parts need to be fixed.

1. The 1/0 variables (`afib`, `dm`, `extra.ica`) might be better if they were treated as the factors they are, and reported as the Yes/No variables are reported, with counts and percentages rather than with means and standard deviations.
2. In some cases, we may prefer to re-order the levels of the categorical (factor) variables, particularly the `mrankin` variable, but also the `ia.occlus` variable. It would also be more typical to put the Intervention group to the left and the Control group to the right, so we may need to adjust our `trt` variable's levels accordingly.
3. For each of the quantitative variables (`age`, `nihss`, `sbp`, `time.iv`, `aspects`, `extra.ica`, `time.rand` and `time.punc`) we should make a decision whether a summary with mean and standard deviation is appropriate, or whether we should instead summarize with, say, the median and quartiles. A mean and standard deviation really only yields an appropriate summary when

the data are least approximately Normally distributed. This will make the p values a bit more reasonable, too. The `test` column in the first attempt will soon have something useful to tell us.

4. If we'd left in the `time.punc` variable, we'd get some warnings, having to do with the fact that `time.punc` is only relevant to patients in the Intervention group.

1.5.2 `fakestroke` Cleaning Up Categorical Variables

Let's specify each of the categorical variables as categorical explicitly. This helps the `CreateTableOne` function treat them appropriately, and display them with counts and percentages. This includes all of the 1/0, Yes/No and multi-categorical variables.

```
fs.factorvars <- c("sex", "location", "hx.isch", "afib", "dm",
                   "mrankin", "iv.altep", "ia.occlus", "extra.ica")
```

Then we simply add a `factorVars = fs.factorvars` call to the `CreateTableOne` function.

We also want to re-order some of those categorical variables, so that the levels are more useful to us. Specifically, we want to:

- place Intervention before Control in the `trt` variable,
- reorder the `mrankin` scale as 0, 1, 2, > 2, and
- rearrange the `ia.occlus` variable to the order[4] presented in Berkhemer et al. (2015).

To accomplish this, we'll use the `fct_relevel` function from the `forcats` package (loaded with the rest of the core `tidyverse` packages) to reorder our levels manually.

```
fakestroke <- fakestroke %>%
    mutate(trt = fct_relevel(trt, "Intervention", "Control"),
           mrankin = fct_relevel(mrankin, "0", "1", "2", "> 2"),
           ia.occlus = fct_relevel(ia.occlus, "Intracranial ICA",
                                   "ICA with M1", "M1", "M2",
                                   "A1 or A2")
           )
```

[4] We might also have considered reordering the `ia.occlus` factor by its frequency, using the `fct_infreq` function

1.6 `fakestroke` Table 1: Attempt 2

```
att2 <- CreateTableOne(data = fakestroke,
                       vars = fs.vars,
                       factorVars = fs.factorvars,
                       strata = fs.trt)
print(att2)
```

```
                       Stratified by trt
                        Intervention   Control        p      test
  n                        233            267
  age (mean (SD))         63.93 (18.09)  65.38 (16.10)  0.343
  sex = Male (%)           135 (57.9)     157 (58.8)    0.917
  nihss (mean (SD))       17.97 (5.04)   18.08 (4.32)   0.787
  location = Right (%)     117 (50.2)     114 (42.7)    0.111
  hx.isch = Yes (%)         29 (12.4)      25 ( 9.4)    0.335
  afib = 1 (%)              66 (28.3)      69 (25.8)    0.601
  dm = 1 (%)                29 (12.4)      34 (12.7)    1.000
  mrankin (%)                                           0.922
     0                     190 (81.5)     214 (80.1)
     1                      21 ( 9.0)      29 (10.9)
     2                      12 ( 5.2)      13 ( 4.9)
     > 2                    10 ( 4.3)      11 ( 4.1)
  sbp (mean (SD))        146.03 (26.00) 145.00 (24.40)  0.647
  iv.altep = Yes (%)       203 (87.1)     242 (90.6)    0.267
  time.iv (mean (SD))     98.22 (45.48)  87.96 (26.01)  0.003
  aspects (mean (SD))      8.35 (1.64)    8.65 (1.47)   0.033
  ia.occlus (%)                                         0.795
     Intracranial ICA        1 ( 0.4)       3 ( 1.1)
     ICA with M1            59 (25.3)      75 (28.2)
     M1                    154 (66.1)     165 (62.0)
     M2                     18 ( 7.7)      21 ( 7.9)
     A1 or A2                1 ( 0.4)       2 ( 0.8)
  extra.ica = 1 (%)         75 (32.2)      70 (26.3)    0.179
  time.rand (mean (SD)) 202.51 (57.33) 213.88 (70.29)  0.051
```

The categorical data presentation looks much improved.

1.6.1 What summaries should we show?

Now, we'll move on to the issue of making a decision about what type of summary to show for the quantitative variables. Since the `fakestroke` data are just simulated and only match the summary statistics of the original results, not the details, we'll adopt the decisions made by Berkhemer et al. (2015), which

were to use medians and interquartile ranges to summarize the distributions of all of the continuous variables **except** systolic blood pressure.

- Specifying certain quantitative variables as *non-normal* causes R to show them with medians and the 25th and 75th percentiles, rather than means and standard deviations, and also causes those variables to be tested using non-parametric tests, like the Wilcoxon signed rank test, rather than the t test. The `test` column indicates this with the word `nonnorm`.
  - In real data situations, what should we do? The answer is to look at the data. I would not make the decision as to which approach to take without first plotting (perhaps in a histogram or a Normal Q-Q plot) the observed distributions in each of the two samples, so that I could make a sound decision about whether Normality was a reasonable assumption. If the means and medians are meaningfully different from each other, this is especially important.
  - To be honest, though, if the variable in question is a relatively unimportant covariate and the p values for the two approaches are nearly the same, I'd say that further investigation is rarely important,
- Specifying *exact* tests for certain categorical variables (we'll try this for the `location` and `mrankin` variables) can be done, and these changes will be noted in the `test` column, as well.
  - In real data situations, I would rarely be concerned about this issue, and often choose Pearson (approximate) options across the board. This is reasonable so long as the number of subjects falling in each category is reasonably large, say above 10. If not, then an exact test may be a tiny improvement.
  - Paraphrasing Rosenbaum (2017), having an exact rather than an approximate test result is about as valuable as having a nice crease in your trousers.

To finish our Table 1, then, we need to specify which variables should be treated as non-Normal in the `print` statement - notice that we don't need to redo the `CreateTableOne` for this change.

```
print(att2,
      nonnormal = c("age", "nihss", "time.iv", "aspects", "time.rand"),
      exact = c("location", "mrankin"))
```

```
                         Stratified by trt
                          Intervention            Control
  n                          233                     267
  age (median [IQR])        65.80 [54.50, 76.00]    65.70 [55.75, 76.20]
  sex = Male (%)             135 (57.9)              157 (58.8)
  nihss (median [IQR])      17.00 [14.00, 21.00]    18.00 [14.00, 22.00]
  location = Right (%)       117 (50.2)              114 (42.7)
  hx.isch = Yes (%)           29 (12.4)               25 ( 9.4)
  afib = 1 (%)                66 (28.3)               69 (25.8)
```

```
dm = 1 (%)                    29 (12.4)                34 (12.7)
mrankin (%)
   0                         190 (81.5)               214 (80.1)
   1                          21 ( 9.0)                29 (10.9)
   2                          12 ( 5.2)                13 ( 4.9)
   > 2                        10 ( 4.3)                11 ( 4.1)
sbp (mean (SD))           146.03 (26.00)           145.00 (24.40)
iv.altep = Yes (%)           203 (87.1)               242 (90.6)
time.iv (median [IQR])     85.00 [67.00, 110.00]    87.00 [65.00, 116.00]
aspects (median [IQR])      9.00 [7.00, 10.00]       9.00 [8.00, 10.00]
ia.occlus (%)
   Intracranial ICA           1 ( 0.4)                 3 ( 1.1)
   ICA with M1               59 (25.3)                75 (28.2)
   M1                       154 (66.1)               165 (62.0)
   M2                        18 ( 7.7)                21 ( 7.9)
   A1 or A2                   1 ( 0.4)                 2 ( 0.8)
extra.ica = 1 (%)            75 (32.2)                70 (26.3)
time.rand (median [IQR]) 204.00 [152.00, 249.50] 196.00 [149.00, 266.00]
                         Stratified by trt
                          p      test
n
age (median [IQR])        0.579 nonnorm
sex = Male (%)            0.917
nihss (median [IQR])      0.453 nonnorm
location = Right (%)      0.106 exact
hx.isch = Yes (%)         0.335
afib = 1 (%)              0.601
dm = 1 (%)                1.000
mrankin (%)               0.917 exact
   0
   1
   2
   > 2
sbp (mean (SD))           0.647
iv.altep = Yes (%)        0.267
time.iv (median [IQR])    0.596 nonnorm
aspects (median [IQR])    0.075 nonnorm
ia.occlus (%)             0.795
   Intracranial ICA
   ICA with M1
   M1
   M2
   A1 or A2
extra.ica = 1 (%)         0.179
time.rand (median [IQR])  0.251 nonnorm
```

1.7 Obtaining a more detailed Summary

If this was a real data set, we'd want to get a more detailed description of the data to make decisions about things like potentially collapsing categories of a variable, or whether or not a normal distribution was useful for a particular continuous variable, etc. You can do this with the `summary` command applied to a created Table 1, which shows, among other things, the effect of changing from normal to non-normal p values for continuous variables, and from approximate to "exact" p values for categorical factors.

Again, as noted above, in a real data situation, we'd want to plot the quantitative variables (within each group) to make a smart decision about whether a t test or Wilcoxon approach is more appropriate.

Note in the summary below that we have some missing values here. Often, we'll present this information within the Table 1, as well.

```
summary(att2)
```

```
     ### Summary of continuous variables ###

trt: Intervention
           n miss p.miss mean sd median p25 p75 min max  skew  kurt
age      233    0    0.0   64 18     66  54  76  23  96 -0.34 -0.52
nihss    233    0    0.0   18  5     17  14  21  10  28  0.48 -0.74
sbp      233    0    0.0  146 26    146 129 164  78 214 -0.07 -0.22
time.iv  233   30   12.9   98 45     85  67 110  42 218  1.03  0.08
aspects  233    0    0.0    8  2      9   7  10   5  10 -0.56 -0.98
time.rand 233   2    0.9  203 57    204 152 250 100 300  0.01 -1.16
------------------------------------------------------------
trt: Control
           n miss p.miss mean sd median p25 p75 min max   skew  kurt
age      267    0    0.0   65 16     66  56  76  24  94 -0.296 -0.28
nihss    267    0    0.0   18  4     18  14  22  11  25  0.017 -1.24
sbp      267    1    0.4  145 24    145 128 161  82 231  0.156  0.08
time.iv  267   25    9.4   88 26     87  65 116  44 130  0.001 -1.32
aspects  267    4    1.5    9  1      9   8  10   5  10 -1.071  0.36
time.rand 267   0    0.0  214 70    196 149 266 120 360  0.508 -0.93

p-values
             pNormal pNonNormal
age      0.342813660 0.57856976
nihss    0.787487252 0.45311695
sbp      0.647157646 0.51346132
time.iv  0.003073372 0.59641104
aspects  0.032662901 0.07464683
time.rand 0.050803672 0.25134327
```

```
Standardize mean differences
             1 vs 2
age       0.08478764
nihss     0.02405390
sbp       0.04100833
time.iv   0.27691223
aspects   0.19210662
time.rand 0.17720957

================================================================================

     ### Summary of categorical variables ###

trt: Intervention
       var   n miss p.miss            level freq percent cum.percent
       sex 233    0    0.0           Female   98    42.1        42.1
                                       Male  135    57.9       100.0

  location 233    0    0.0             Left  116    49.8        49.8
                                       Right  117    50.2       100.0

   hx.isch 233    0    0.0               No  204    87.6        87.6
                                        Yes   29    12.4       100.0

      afib 233    0    0.0                0  167    71.7        71.7
                                          1   66    28.3       100.0

        dm 233    0    0.0                0  204    87.6        87.6
                                          1   29    12.4       100.0

   mrankin 233    0    0.0                0  190    81.5        81.5
                                          1   21     9.0        90.6
                                          2   12     5.2        95.7
                                       > 2   10     4.3       100.0

  iv.altep 233    0    0.0               No   30    12.9        12.9
                                        Yes  203    87.1       100.0

 ia.occlus 233    0    0.0 Intracranial ICA    1     0.4         0.4
                                ICA with M1   59    25.3        25.8
                                         M1  154    66.1        91.8
                                         M2   18     7.7        99.6
                                  A1 or A2    1     0.4       100.0

 extra.ica 233    0    0.0                0  158    67.8        67.8
```

```
                                          1   75    32.2       100.0

------------------------------------------------------------
trt: Control
       var   n miss p.miss            level freq percent cum.percent
       sex 267    0    0.0           Female  110    41.2        41.2
                                      Male  157    58.8       100.0

  location 267    0    0.0             Left  153    57.3        57.3
                                     Right  114    42.7       100.0

   hx.isch 267    0    0.0               No  242    90.6        90.6
                                       Yes   25     9.4       100.0

      afib 267    0    0.0                0  198    74.2        74.2
                                          1   69    25.8       100.0

        dm 267    0    0.0                0  233    87.3        87.3
                                          1   34    12.7       100.0

   mrankin 267    0    0.0                0  214    80.1        80.1
                                          1   29    10.9        91.0
                                          2   13     4.9        95.9
                                       > 2   11     4.1       100.0

  iv.altep 267    0    0.0               No   25     9.4         9.4
                                       Yes  242    90.6       100.0

 ia.occlus 267    1    0.4 Intracranial ICA    3     1.1         1.1
                           ICA with M1   75    28.2        29.3
                                    M1  165    62.0        91.4
                                    M2   21     7.9        99.2
                                A1 or A2    2     0.8       100.0

 extra.ica 267    1    0.4                0  196    73.7        73.7
                                          1   70    26.3       100.0


p-values
            pApprox    pExact
sex       0.9171387 0.8561188
location  0.1113553 0.1056020
hx.isch   0.3352617 0.3124683
afib      0.6009691 0.5460206
dm        1.0000000 1.0000000
mrankin   0.9224798 0.9173657
```

```
iv.altep  0.2674968 0.2518374
ia.occlus 0.7945580 0.8189090
extra.ica 0.1793385 0.1667574

Standardize mean differences
              1 vs 2
sex       0.017479025
location  0.151168444
hx.isch   0.099032275
afib      0.055906317
dm        0.008673478
mrankin   0.062543164
iv.altep  0.111897009
ia.occlus 0.117394890
extra.ica 0.129370206
```

In this case, I have simulated the data to mirror the results in the published Table 1 for this study. In no way have I captured the full range of the real data, or any of the relationships in that data, so it's more important here to see what's available in the analysis, rather than to interpret it closely in the clinical context.

1.8 Exporting the Completed Table 1 from R to Excel or Word

Once you've built the table and are generally satisfied with it, you'll probably want to be able to drop it into Excel or Word for final cleanup.

1.8.1 Approach A: Save and open in Excel

One option is to **save the Table 1** to a `.csv` file within our `data` subfolder (note that the `data` folder must already exist), which you can then open directly in Excel. This is the approach I generally use. Note the addition of some `quote`, `noSpaces` and `printToggle` selections here.

```
fs.table1save <- print(att2,
      nonnormal = c("age", "nihss", "time.iv", "aspects", "time.rand"),
      exact = c("location", "mrankin"),
      quote = FALSE, noSpaces = TRUE, printToggle = FALSE)

write.csv(fs.table1save, file = "data/fs-table1.csv")
```

When I then open the `fs-table1.csv` file in Excel, it looks like this:

	A	B	C	D	E
1		Intervention	Control	p	test
2	n	233	267		
3	age (median [IQR])	65.80 [54.50, 76.00]	65.70 [55.75, 76.20]	0.579	nonnorm
4	sex = Male (%)	135 (57.9)	157 (58.8)	0.917	
5	nihss (median [IQR])	17.00 [14.00, 21.00]	18.00 [14.00, 22.00]	0.453	nonnorm
6	location = Right (%)	117 (50.2)	114 (42.7)	0.111	
7	hx.isch = Yes (%)	29 (12.4)	25 (9.4)	0.335	
8	afib = 1 (%)	66 (28.3)	69 (25.8)	0.601	
9	dm = 1 (%)	29 (12.4)	34 (12.7)	1	
10	mrankin (%)			0.922	
11	0	190 (81.5)	214 (80.1)		
12	1	21 (9.0)	29 (10.9)		
13	2	12 (5.2)	13 (4.9)		
14	>2	10 (4.3)	11 (4.1)		
15	sbp (mean (sd))	146.03 (26.00)	145.00 (24.40)	0.647	
16	iv.altep = Yes (%)	203 (87.1)	242 (90.6)	0.267	
17	time.iv (median [IQR])	85.00 [67.00, 110.00]	87.00 [65.00, 116.00]	0.596	nonnorm
18	aspects (median [IQR])	9.00 [7.00, 10.00]	9.00 [8.00, 10.00]	0.075	nonnorm
19	ia.occlus (%)			0.795	
20	Intracranial ICA	1 (0.4)	3 (1.1)		
21	ICA with M1	59 (25.3)	75 (28.2)		
22	M1	154 (66.1)	165 (62.0)		
23	M2	18 (7.7)	21 (7.9)		
24	A1 or A2	1 (0.4)	2 (0.8)		
25	extra.ica = 1 (%)	75 (32.2)	70 (26.3)	0.179	
26	time.rand (median [IQR])	204.00 [152.00, 249.50]	196.00 [149.00, 266.00]	0.251	nonnorm
27	time.punc (median [IQR])	260.00 [212.00, 313.00]	NA [NA, NA]	NA	nonnorm
28					

And from here, I can either drop it directly into Word, or present it as is, or start tweaking it to meet formatting needs.

1.8.2 Approach B: Produce the Table so you can cut and paste it

```
print(att2,
      nonnormal = c("age", "nihss", "time.iv", "aspects", "time.rand"),
      exact = c("location", "mrankin"),
      quote = TRUE, noSpaces = TRUE)
```

This will look like a mess by itself, but if you:

1. copy and paste that mess into Excel
2. select Text to Columns from the Data menu
3. select Delimited, then Space and select Treat consecutive delimiters as one

you should get something usable again.

Or, in Word,

1. insert the text
2. select the text with your mouse
3. select Insert ... Table ... Convert Text to Table
4. place a quotation mark in the "Other" area under Separate text at ...

After dropping blank columns, the result looks pretty good.

1.9 A Controlled Biological Experiment - The Blood-Brain Barrier

My source for the data and the following explanatory paragraph is page 307 from Ramsey and Schafer (2002). The original data come from Barnett et al. (1995).

> The human brain (and that of rats, coincidentally) is protected from the bacteria and toxins that course through the bloodstream by something called the blood-brain barrier. After a method of disrupting the barrier was developed, researchers tested this new mechanism, as follows. A series of 34 rats were inoculated with human lung cancer cells to induce brain tumors. After 9-11 days they were infused with either the barrier disruption (BD) solution or, as a control, a normal saline (NS) solution. Fifteen minutes later, the rats received a standard dose of a particular therapeutic antibody (L6-F(ab')2. The key measure of the effectiveness of transmission across the brain-blood barrier is the ratio of the antibody concentration in the brain tumor to the antibody concentration in normal tissue outside the brain. The rats were then sacrificed, and the amounts of antibody in the brain tumor and in normal tissue from the liver were measured. The study's primary objective is to determine whether the antibody concentration in the tumor increased when the blood-barrier disruption infusion was given, and if so, by how much?

1.10 The `bloodbrain.csv` file

Consider the data, available on our Data and Code website in the `bloodbrain.csv` file, which includes the following variables:

Variable	Description
`case`	identification number for the rat (1 - 34)
`brain`	an outcome: Brain tumor antibody count (per gram)
`liver`	an outcome: Liver antibody count (per gram)
`tlratio`	an outcome: tumor / liver concentration ratio

Variable	Description
solution	the treatment: BD (barrier disruption) or NS (normal saline)
sactime	a design variable: Sacrifice time (hours; either 0.5, 3, 24 or 72)
postin	covariate: Days post-inoculation of lung cancer cells (9, 10 or 11)
sex	covariate: M or F
wt.init	covariate: Initial weight (grams)
wt.loss	covariate: Weight loss (grams)
wt.tumor	covariate: Tumor weight (10^{-4} grams)

And here's what the data look like in R.

```
bloodbrain
```

```
# A tibble: 34 x 11
    case  brain  liver tlratio solution sactime postin sex   wt.init wt.loss
   <dbl>  <dbl>  <dbl>   <dbl> <chr>      <dbl>  <dbl> <chr>   <dbl>   <dbl>
 1     1  41081 1.46e6  0.0282 BD           0.5     10 F         239     5.9
 2     2  44286 1.60e6  0.0276 BD           0.5     10 F         225     4
 3     3 102926 1.60e6  0.0642 BD           0.5     10 F         224    -4.9
 4     4  25927 1.78e6  0.0146 BD           0.5     10 F         184     9.8
 5     5  42643 1.35e6  0.0316 BD           0.5     10 F         250     6
 6     6  31342 1.79e6  0.0175 NS           0.5     10 F         196     7.7
 7     7  22815 1.63e6  0.0140 NS           0.5     10 F         200     0.5
 8     8  16629 1.62e6  0.0103 NS           0.5     10 F         273     4
 9     9  22315 1.57e6  0.0142 NS           0.5     10 F         216     2.8
10    10  77961 1.06e6  0.0735 BD           3       10 F         267     2.6
# ... with 24 more rows, and 1 more variable: wt.tumor <dbl>
```

1.11 A Table 1 for bloodbrain

Barnett et al. (1995) did not provide a Table 1 for these data, so let's build one to compare the two solutions (BD vs. NS) on the covariates and outcomes, plus the natural logarithm of the tumor/liver concentration ratio (tlratio). We'll opt to treat the sacrifice time (sactime) and the days post-inoculation of lung cancer cells (postin) as categorical rather than quantitative variables.

```
bloodbrain <- bloodbrain %>%
    mutate(logTL = log(tlratio))

dput(names(bloodbrain))
```

```
c("case", "brain", "liver", "tlratio", "solution", "sactime",
"postin", "sex", "wt.init", "wt.loss", "wt.tumor", "logTL")
```

OK - there's the list of variables we'll need. I'll put the outcomes at the bottom of the table.

```
bb.vars <- c("sactime", "postin", "sex", "wt.init", "wt.loss",
             "wt.tumor", "brain", "liver", "tlratio", "logTL")

bb.factors <- c("sactime", "sex", "postin")

bb.att1 <- CreateTableOne(data = bloodbrain,
                          vars = bb.vars,
                          factorVars = bb.factors,
                          strata = c("solution"))
summary(bb.att1)
```

```
     ### Summary of continuous variables ###

solution: BD
          n miss p.miss   mean    sd median    p25   p75    min   max  skew
wt.init  17    0      0    243 3e+01  2e+02  2e+02 3e+02  2e+02 3e+02 -0.39
wt.loss  17    0      0      3 5e+00  4e+00  1e+00 6e+00 -5e+00 1e+01 -0.10
wt.tumor 17    0      0    157 8e+01  2e+02  1e+02 2e+02  2e+01 4e+02  0.53
brain    17    0      0  56043 3e+04  5e+04  4e+04 8e+04  6e+03 1e+05  0.29
liver    17    0      0 672577 7e+05  6e+05  2e+04 1e+06  2e+03 2e+06  0.35
tlratio  17    0      0      2 3e+00  1e-01  6e-02 3e+00  1e-02 9e+00  1.58
logTL    17    0      0     -1 2e+00 -2e+00 -3e+00 1e+00 -4e+00 2e+00  0.08
         kurt
wt.init   0.7
wt.loss   0.2
wt.tumor  1.0
brain    -0.6
liver    -1.7
tlratio   1.7
logTL    -1.7
------------------------------------------------------------
solution: NS
          n miss p.miss   mean    sd median    p25    p75    min   max  skew
wt.init  17    0      0    240 3e+01  2e+02  2e+02  3e+02  2e+02 3e+02  0.33
wt.loss  17    0      0      4 4e+00  3e+00  2e+00  7e+00 -4e+00 1e+01 -0.09
wt.tumor 17    0      0    209 1e+02  2e+02  2e+02  3e+02  3e+01 5e+02  0.63
brain    17    0      0  23887 1e+04  2e+04  1e+04  3e+04  1e+03 5e+04  0.30
liver    17    0      0 664975 7e+05  7e+05  2e+04  1e+06  9e+02 2e+06  0.40
tlratio  17    0      0      1 2e+00  5e-02  3e-02  9e-01  1e-02 7e+00  2.27
logTL    17    0      0     -2 2e+00 -3e+00 -3e+00 -7e-02 -5e+00 2e+00  0.27
          kurt
wt.init  -0.48
wt.loss   0.08
```

```
wt.tumor  0.77
brain    -0.35
liver    -1.56
tlratio   4.84
logTL    -1.61

p-values
             pNormal  pNonNormal
wt.init  0.807308940 0.641940278
wt.loss  0.683756156 0.876749808
wt.tumor 0.151510151 0.190482094
brain    0.001027678 0.002579901
liver    0.974853609 0.904045603
tlratio  0.320501715 0.221425879
logTL    0.351633525 0.221425879

Standardize mean differences
             1 vs 2
wt.init  0.08435244
wt.loss  0.14099823
wt.tumor 0.50397184
brain    1.23884159
liver    0.01089667
tlratio  0.34611465
logTL    0.32420504

================================================================================

     ### Summary of categorical variables ###

solution: BD
     var  n miss p.miss level freq percent cum.percent
 sactime 17    0    0.0   0.5    5    29.4        29.4
                          3    4    23.5        52.9
                         24    4    23.5        76.5
                         72    4    23.5       100.0

  postin 17    0    0.0     9    1     5.9         5.9
                         10   14    82.4        88.2
                         11    2    11.8       100.0

     sex 17    0    0.0     F   13    76.5        76.5
                          M    4    23.5       100.0

------------------------------------------------------------
solution: NS
```

```
     var  n miss p.miss level freq percent cum.percent
 sactime 17    0    0.0   0.5    4    23.5        23.5
                          3    5    29.4        52.9
                         24    4    23.5        76.5
                         72    4    23.5       100.0

  postin 17    0    0.0     9    2    11.8        11.8
                         10   13    76.5        88.2
                         11    2    11.8       100.0

     sex 17    0    0.0     F   13    76.5        76.5
                          M    4    23.5       100.0


p-values
          pApprox pExact
sactime 0.9739246      1
postin  0.8309504      1
sex     1.0000000      1

Standardize mean differences
           1 vs 2
sactime 0.1622214
postin  0.2098877
sex     0.0000000
```

Note that, in this particular case, the decisions we make about normality vs. non-normality (for quantitative variables) and the decisions we make about approximate vs. exact testing (for categorical variables) won't actually change the implications of the p values. Each approach gives similar results for each variable. Of course, that's not always true.

1.11.1 Generate final Table 1 for `bloodbrain`

I'll choose to treat `tlratio` and its logarithm as non-Normal, but otherwise, use t tests, but admittedly, that's an arbitrary decision, really.

```
print(bb.att1, nonnormal = c("tlratio", "logTL"))
```

```
                       Stratified by solution
                        BD                      NS
  n                            17                         17
  sactime (%)
     0.5                        5 (29.4)                   4 (23.5)
     3                          4 (23.5)                   5 (29.4)
     24                         4 (23.5)                   4 (23.5)
```

```
   72                          4 (23.5)               4 (23.5)
postin (%)
   9                           1 ( 5.9)               2 (11.8)
   10                         14 (82.4)              13 (76.5)
   11                          2 (11.8)               2 (11.8)
sex = M (%)                    4 (23.5)               4 (23.5)
wt.init (mean (SD))       242.82 (27.23)         240.47 (28.54)
wt.loss (mean (SD))         3.34 (4.68)            3.94 (3.88)
wt.tumor (mean (SD))      157.29 (84.00)         208.53 (116.68)
brain (mean (SD))       56043.41 (33675.40)    23887.18 (14610.53)
liver (mean (SD))      672577.35 (694479.58)  664975.47 (700773.13)
tlratio (median [IQR])      0.12 [0.06, 2.84]      0.05 [0.03, 0.94]
logTL (median [IQR])       -2.10 [-2.74, 1.04]    -2.95 [-3.41, -0.07]
                      Stratified by solution
                       p      test
n
sactime (%)             0.974
   0.5
   3
   24
   72
postin (%)              0.831
   9
   10
   11
sex = M (%)             1.000
wt.init (mean (SD))     0.807
wt.loss (mean (SD))     0.684
wt.tumor (mean (SD))    0.152
brain (mean (SD))       0.001
liver (mean (SD))       0.975
tlratio (median [IQR])  0.221 nonnorm
logTL (median [IQR])    0.221 nonnorm
```

Or, we can get an Excel-readable version placed in a `data` subfolder, using

```
bb.t1 <- print(bb.att1, nonnormal = c("tlratio", "logTL"), quote = FALSE,
               noSpaces = TRUE, printToggle = FALSE)

write.csv(bb.t1, file = "data/bb-table1.csv")
```

which, when dropped into Excel, will look like this:

	A	B	C	D	E
1		BD	NS	p	test
2	n	17	17		
3	sex = M (%)	4 (23.5)	4 (23.5)	1	
4	sactime (%)			0.974	
5	0.5	5 (29.4)	4 (23.5)		
6	3	4 (23.5)	5 (29.4)		
7	24	4 (23.5)	4 (23.5)		
8	72	4 (23.5)	4 (23.5)		
9	postin (%)			0.831	
10	9	1 (5.9)	2 (11.8)		
11	10	14 (82.4)	13 (76.5)		
12	11	2 (11.8)	2 (11.8)		
13	wt.init (mean (sd))	242.82 (27.23)	240.47 (28.54)	0.807	
14	wt.loss (mean (sd))	3.34 (4.68)	3.94 (3.88)	0.684	
15	wt.tumor (mean (sd))	157.29 (84.00)	208.53 (116.68)	0.152	
16	brain (mean (sd))	56043.41 (33675.40)	23887.18 (14610.53)	0.001	
17	liver (mean (sd))	672577.35 (694479.58)	664975.47 (700773.13)	0.975	
18	tlratio (median [IQR])	0.12 [0.06, 2.84]	0.05 [0.03, 0.94]	0.221	nonnorm
19	logTL (median [IQR])	-2.10 [-2.74, 1.04]	-2.95 [-3.41, -0.07]	0.221	nonnorm
20					

One thing I would definitely clean up here, in practice, is to change the presentation of the p value for `sex` from 1 to > 0.99, or just omit it altogether. I'd also drop the `computer-ese` where possible, add units for the measures, round **a lot**, identify the outcomes carefully, and use notes to indicate deviations from the main approach.

1.11.2 A More Finished Version (after Cleanup in Word)

Table 1. Comparing Rats Receiving BD to those Receiving NS on Available Covariates and Design Variables, and Key Outcomes

	Barrier Disruption (BD: treatment)	Normal Saline (NS: control)	p
# of Rats	17	17	
Sex = Male	4 (23.5)	4 (23.5)	-
Sacrifice Time (hours)			0.97
0.5	5 (29.4)	4 (23.5)	
3	4 (23.5)	5 (29.4)	
24	4 (23.5)	4 (23.5)	
72	4 (23.5)	4 (23.5)	
Days post-inoculation of lung cancer cells			0.83
9	1 (5.9)	2 (11.8)	
10	14 (82.4)	13 (76.5)	
11	2 (11.8)	2 (11.8)	
Initial Weight (g)	243 (27)	240 (29)	0.81
Weight Loss (g)	3.3 (4.7)	3.9 (3.9)	0.68
Tumor Weight (10^{-4} g)	157.3 (84.0)	208.5 (116.7)	0.15
Key Outcomes: mean (sd) unless otherwise indicated			
Brain Tumor Antibody Count (per g)	56,043 (33,675)	23,887 (14,611)	0.001
Liver Antibody Count (per g)	672,577 (694,480)	664,975 (700,773)	0.98
Tumor/Liver Ratio (median [Q25, Q75])	0.12 [0.06, 2.84]	0.05 [0.03, 0.94]	0.22
Natural Log of Tumor/Liver Ratio (median [Q25, Q75])	-2.10 [-2.74, 1.04]	-2.95 [-3.41, -0.07]	0.22

Table 1 Notes:

- Categorical variables are summarized with counts, percentages and p values based on approximate chi-square tests.
- Continuous variables, unless otherwise indicated, are summarized with means, standard deviations and p values based on t tests.
- The Tumor / Liver ratio and its natural logarithm are summarized with the median and quartiles and a p value from a non-parametric (Wilcoxon signed rank) test.

Chapter 2

BRFSS SMART Data Building

The Centers for Disease Control analyzes Behavioral Risk Factor Surveillance System (BRFSS) survey data for specific metropolitan and micropolitan statistical areas (MMSAs) in a program called the Selected Metropolitan/Micropolitan Area Risk Trends of BRFSS (SMART BRFSS.)

In this work, we will focus on data from the 2017 SMART, and in particular on data from the state of Ohio, and from the Cleveland-Elyria, OH, Metropolitan Statistical Area. The purpose of this survey is to provide localized health information that can help public health practitioners identify local emerging health problems, plan and evaluate local responses, and efficiently allocate resources to specific needs.

In this chapter, I describe some cleaning of the BRFSS SMART data, and break it out into national, statewide, and local samples.

The data files produced by this chapter include:

- `smart_ohio.Rds` which includes data on approximately 100 variables for over 7000 subjects in six MMSAs that are at least partially located in the state of Ohio.
- `smart_cle.Rds` which includes data on those same variables for a little over 1000 subjects in the Cleveland-Elyria-Lorain OH MMSA.

2.1 Key resources

- the "raw" data, in the form of the 2017 SMART BRFSS MMSA Data, found in a zipped SAS Transport Format file. The data were released in

October 2018.
- the MMSA Variable Layout which simply lists the variables included in the data file
- the Calculated Variables PDF which describes the risk factors by data variable names - there is also an online summary matrix of these calculated variables.
- the lengthy 2017 Survey Questions PDF which lists all questions asked as part of the BRFSS in 2017
- the enormous Codebook for the 2017 BRFSS Survey PDF which identifies the variables by name for us.

Also, for each subject, we are also provided with a sampling weight, in `_MMSAWT`, which will help us incorporate the sampling design later. These weights are at the MMSA level, and are used for generating MMSA-level estimates for variables in the data set. Details on the weighting methodology are available at this PDF.

2.2 Ingesting the Raw Data

To create the data files we'll use, I used the `read_xpt` function from the `haven` package to bring in the SAS XPT data file that is provided by CDC. The codes I used (but won't use in these Notes) were:

```
smart_raw <- read_xpt("MMSA2017/MMSA2017.xpt")
```

This gives the nationwide data, which has 230,875 rows and 177 columns.

But for the purposes of putting these Notes online, I needed to crank down the sample size enormously. To that end, I created a new data file, which I developed by

- importing the MMSA2017.xpt file as above
- filtering away all observations except those from MMSAs which include Ohio in their name, and
- saving the result, which now has 7,412 rows and 177 columns.

The code (again, not run here) that I used to filter to the OH-based MMSAs was:

```
smart_ohio_raw <- smart_raw %>%
    filter(str_detect(MMSANAME, "OH"))

write_csv(smart_ohio_raw, "data/smart_ohio_raw.csv")
```

So, for purposes of these notes, our complete data set is actually coming from `smart_ohio_raw.csv` and consists only of the 7,412 observations associated with the six MMSAs that include Ohio in their names.

2.3 Ingesting from our CSV file

Note that the `smart_ohio_raw.csv` and other data files we're developing in this Chapter are available on our Data and Code website

```
smart_ohio_raw <- read_csv("data/smart_ohio_raw.csv")

dim(smart_ohio_raw)
```

```
[1] 7412  177
```

2.4 What does the raw data look like?

Here is a list of all variable names included in this file. We're not going to use all of those variables, but this will give you a sense of what is available.

```
names(smart_ohio_raw)
```

```
  [1] "DISPCODE" "STATERE1" "SAFETIME" "HHADULT"  "GENHLTH"  "PHYSHLTH"
  [7] "MENTHLTH" "POORHLTH" "HLTHPLN1" "PERSDOC2" "MEDCOST"  "CHECKUP1"
 [13] "BPHIGH4"  "BPMEDS"   "CHOLCHK1" "TOLDHI2"  "CHOLMED1" "CVDINFR4"
 [19] "CVDCRHD4" "CVDSTRK3" "ASTHMA3"  "ASTHNOW"  "CHCSCNCR" "CHCOCNCR"
 [25] "CHCCOPD1" "HAVARTH3" "ADDEPEV2" "CHCKIDNY" "DIABETE3" "DIABAGE2"
 [31] "LMTJOIN3" "ARTHDIS2" "ARTHSOCL" "JOINPAI1" "SEX"      "MARITAL"
 [37] "EDUCA"    "RENTHOM1" "NUMHHOL2" "NUMPHON2" "CPDEMO1A" "VETERAN3"
 [43] "EMPLOY1"  "CHILDREN" "INCOME2"  "INTERNET" "WEIGHT2"  "HEIGHT3"
 [49] "PREGNANT" "DEAF"     "BLIND"    "DECIDE"   "DIFFWALK" "DIFFDRES"
 [55] "DIFFALON" "SMOKE100" "SMOKDAY2" "STOPSMK2" "LASTSMK2" "USENOW3"
 [61] "ECIGARET" "ECIGNOW"  "ALCDAY5"  "AVEDRNK2" "DRNK3GE5" "MAXDRNKS"
 [67] "FRUIT2"   "FRUITJU2" "FVGREEN1" "FRENCHF1" "POTATOE1" "VEGETAB2"
 [73] "EXERANY2" "EXRACT11" "EXEROFT1" "EXERHMM1" "EXRACT21" "EXEROFT2"
 [79] "EXERHMM2" "STRENGTH" "SEATBELT" "FLUSHOT6" "FLSHTMY2" "PNEUVAC3"
 [85] "SHINGLE2" "HIVTST6"  "HIVTSTD3" "HIVRISK5" "CASTHDX2" "CASTHNO2"
 [91] "CALLBCKZ" "WDUSENOW" "WDINFTRK" "WDHOWOFT" "WDSHARE"  "NAMTRIBE"
 [97] "NAMOTHR"  "_URBNRRL" "_STSTR"   "_IMPSEX"  "_RFHLTH"  "_PHYS14D"
[103] "_MENT14D" "_HCVU651" "_RFHYPE5" "_CHOLCH1" "_RFCHOL1" "_MICHD"
[109] "_LTASTH1" "_CASTHM1" "_ASTHMS1" "_DRDXAR1" "_LMTACT1" "_LMTWRK1"
[115] "_LMTSCL1" "_PRACE1"  "_MRACE1"  "_HISPANC" "_RACE"    "_RACEG21"
[121] "_RACEGR3" "_AGEG5YR" "_AGE65YR" "_AGE80"   "_AGE_G"   "WTKG3"
[127] "_BMI5"    "_BMI5CAT" "_RFBMI5"  "_EDUCAG"  "_INCOMG"  "_SMOKER3"
[133] "_RFSMOK3" "_ECIGSTS" "_CURECIG" "DRNKANY5" "_RFBING5" "_DRNKWEK"
[139] "_RFDRHV5" "FTJUDA2_" "FRUTDA2_" "GRENDA1_" "FRNCHDA_" "POTADA1_"
[145] "VEGEDA2_" "_MISFRT1" "_MISVEG1" "_FRTRES1" "_VEGRES1" "_FRUTSU1"
[151] "_VEGESU1" "_FRTLT1A" "_VEGLT1A" "_FRT16A"  "_VEG23A"  "_FRUITE1"
[157] "_VEGETE1" "_TOTINDA" "_MINAC11" "_MINAC21" "_PACAT1"  "_PAINDX1"
```

```
[163] "_PA150R2" "_PA300R2" "_PA30021" "_PASTRNG" "_PAREC1"  "_PASTAE1"
[169] "_RFSEAT2" "_RFSEAT3" "_FLSHOT6" "_PNEUMO2" "_AIDTST3" "_MMSA"
[175] "_MMSAWT"  "SEQNO"    "MMSANAME"
```

2.5 Cleaning the BRFSS Data

2.5.1 Identifying Information

The identifying variables for each subject are gathered in `SEQNO`, which I'll leave alone.

- Each statistical (geographic) area is identified by a `_MMSA` variable, which I'll rename `mmsa_code`, and by an `MMSANAME` which I'll rename as `mmsa_name`
- For each subject, we are also provided with a sampling weight, in `_MMSAWT`, which will help us incorporate the sampling design later in the semester. We'll rename this as `mmsa_wt`. Details on the weighting methodology are available at https://www.cdc.gov/brfss/annual_data/2017/pdf/2017_SMART_BRFSS_MMSA_Methodology-508.pdf

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(mmsa_code = `_MMSA`,
           mmsa_name = `MMSANAME`,
           mmsa_wt = `_MMSAWT`)

smart_ohio_raw %>% count(mmsa_code, mmsa_name)
```

```
# A tibble: 6 x 3
  mmsa_code mmsa_name                                                    n
      <dbl> <chr>                                                    <int>
1     17140 Cincinnati, OH-KY-IN, Metropolitan Statistical Area       1737
2     17460 Cleveland-Elyria, OH, Metropolitan Statistical Area       1133
3     18140 Columbus, OH, Metropolitan Statistical Area               2033
4     19380 Dayton, OH, Metropolitan Statistical Area                  587
5     26580 Huntington-Ashland, WV-KY-OH, Metropolitan Statistical Area  1156
6     45780 Toledo, OH, Metropolitan Statistical Area                  766
```

Those names are very long. I'll build some shorter ones, by dropping everything after the comma.

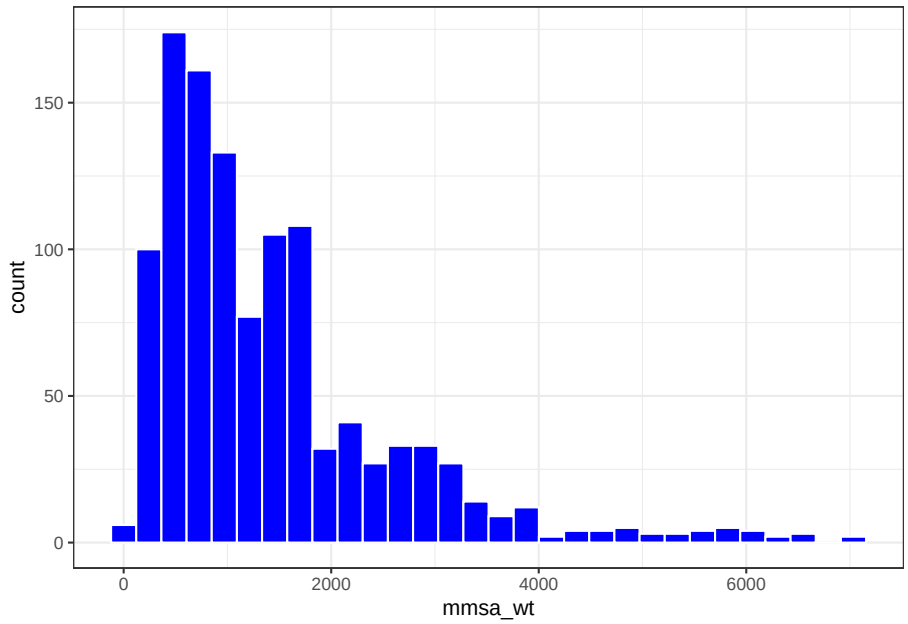
```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(mmsa = str_replace_all(string = mmsa_name, pattern="\\,.*$",replacement=" ")

smart_ohio_raw %>% count(mmsa, mmsa_name)
```

```
# A tibble: 6 x 3
  mmsa                mmsa_name                                          n
```

```
  <chr>               <chr>                                              <int>
1 "Cincinnati "       Cincinnati, OH-KY-IN, Metropolitan Statistical Area  1737
2 "Cleveland-Elyria " Cleveland-Elyria, OH, Metropolitan Statistical Area  1133
3 "Columbus "         Columbus, OH, Metropolitan Statistical Area          2033
4 "Dayton "           Dayton, OH, Metropolitan Statistical Area             587
5 "Huntington-Ashlan~ Huntington-Ashland, WV-KY-OH, Metropolitan Statisti~ 1156
6 "Toledo "           Toledo, OH, Metropolitan Statistical Area             766
```

And here are the sampling weights for the subjects in the Cleveland-Elyria MSA.

```
smart_ohio_raw %>%
    filter(mmsa_code == 17460) %>%
    ggplot(., aes(x = mmsa_wt)) +
    geom_histogram(bins = 30, fill = "blue", col = "white")
```



2.5.2 Survey Method

2.5.2.1 `DISPCODE` and its cleanup to `completed`

`DISPCODE` which is 1100 if the subject completed the interview, and 1200 if they partially completed the interview. We'll create a variable called `completed` that indicates (1 = complete, 0 = not) whether the subject completed the interview.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(completed = 12 - (DISPCODE/100))
```

```
smart_ohio_raw %>% count(DISPCODE, completed)
```

```
# A tibble: 2 x 3
  DISPCODE completed     n
     <dbl>     <dbl> <int>
1     1100         1  6277
2     1200         0  1135
```

2.5.2.2 STATERE1 and SAFETIME and their reduction to landline

BRFSSS is conducted by telephone. The next two variables help us understand whether the subject was contacted via land line or via cellular phone.

- STATERE1 is 1 if the subject is a resident of the state (only asked of people in the land line version of the survey).
- SAFETIME is 1 if this is a safe time to talk (only asked of people in the cell phone version of the survey).
- We'll use STATERE1 and SAFETIME to create an indicator variable landline that specifies how the respondent was surveyed (1 = land line, 0 = cell phone), as follows...

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(landline = replace_na(STATERE1, 0))

smart_ohio_raw %>% count(STATERE1, SAFETIME, landline)
```

```
# A tibble: 2 x 4
  STATERE1 SAFETIME landline     n
     <dbl>    <dbl>    <dbl> <int>
1        1       NA        1  3649
2       NA        1        0  3763
```

2.5.2.3 HHADULT and its cleanup to hhadults

- HHADULT is the response to "How many members of your household, including yourself, are 18 years of age or older?"
  - The permitted responses range from 1-76, with special values 77 for Don't Know/Not Sure and 99 for refused, with BLANK for missing or not asked.
  - So we should change all numerical values above 76 to NA for our analyses (the blanks are already regarded as NAs by R in the ingestion process.)

```
smart_ohio_raw %>% tabyl(HHADULT)
```

```
 HHADULT    n      percent valid_percent
       1  274 0.0369670804   0.236206897
       2  603 0.0813545602   0.519827586
       3  170 0.0229357798   0.146551724
       4   73 0.0098488937   0.062931034
       5   28 0.0037776579   0.024137931
       6    4 0.0005396654   0.003448276
       7    3 0.0004047491   0.002586207
       8    1 0.0001349164   0.000862069
      10    1 0.0001349164   0.000862069
      11    1 0.0001349164   0.000862069
      99    2 0.0002698327   0.001724138
      NA 6252 0.8434970318            NA
```

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(hhadults = HHADULT,
           hhadults = replace(hhadults, hhadults > 76, NA))

smart_ohio_raw %>% count(HHADULT, hhadults) %>% tail()
```

```
# A tibble: 6 x 3
  HHADULT hhadults     n
    <dbl>    <dbl> <int>
1       7        7     3
2       8        8     1
3      10       10     1
4      11       11     1
5      99       NA     2
6      NA       NA  6252
```

2.5.3 Health Status (1 item)

The next variable describes relate to the subject's health status.

2.5.3.1 `GENHLTH` and its cleanup to `genhealth`

- `GENHLTH`, the General Health variable, which is the response to "Would you say that in general your health is . . . "
  - 1 = Excellent
  - 2 = Very good
  - 3 = Good
  - 4 = Fair
  - 5 = Poor
  - 7 = Don't know/Not sure
  - 9 = Refused

  - BLANK = Not asked or missing

To clean up the `GENHLTH` data into a new variable called `genhealth` we'll need to - convince R that the 7 and 9 values are in fact best interpreted as `NA`, - and perhaps change the variable to a factor and incorporate the names into the levels.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(genhealth = fct_recode(factor(GENHLTH),
                                  "1_Excellent" = "1",
                                  "2_VeryGood" = "2",
                                  "3_Good" = "3",
                                  "4_Fair" = "4",
                                  "5_Poor" = "5",
                                  NULL = "7",
                                  NULL = "9"))

smart_ohio_raw %>% count(GENHLTH, genhealth)
```

```
# A tibble: 7 x 3
  GENHLTH genhealth       n
    <dbl> <fct>       <int>
1       1 1_Excellent  1057
2       2 2_VeryGood   2406
3       3 3_Good       2367
4       4 4_Fair       1139
5       5 5_Poor        428
6       7 <NA>           10
7       9 <NA>            5
```

2.5.4 Healthy Days - Health-Related Quality of Life (3 items)

The next three variables describe the subject's health-related quality of life.

2.5.4.1 `PHYSHLTH` and its cleanup to `physhealth`

PHYSHLTH', the Number of Days Physical Health Not Good variable, which is the response to "Now thinking about your physical health, which includes physical illness and injury, for how many days during the past 30 days was your physical health not good?"

- Values of 1-30 are numeric and reasonable.
- A value of 88 indicates "none" and should be recoded to 0.
- 77 is the code for Don't know/Not sure
- 99 is the code for Refused

- BLANK indicates Not asked or missing, and R recognizes this as `NA` properly.

To clean up `PHYSHLTH` to a new variable called `physhealth`, we'll need: - to convince R that the 77 and 99 values are in fact best interpreted as `NA`, and - to convince R that the 88 should be interpreted as 0.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(physhealth = PHYSHLTH,
           physhealth = replace(physhealth, physhealth %in% c(77, 99), NA),
           physhealth = replace(physhealth, physhealth == 88, 0))

smart_ohio_raw %>% count(PHYSHLTH, physhealth) %>% tail()
```

```
# A tibble: 6 x 3
  PHYSHLTH physhealth     n
     <dbl>      <dbl> <int>
1       28         28    12
2       29         29    14
3       30         30   677
4       77         NA   123
5       88          0  4380
6       99         NA    15
```

Note that we present the `tail` of the counts in this case so we can see what happens to the key values (77, 88, 99) of our original variable `PHYSHLTH`.

2.5.4.2 `MENTHLTH` and its cleanup to `menthealth`

MENTHLTH', the Number of Days Mental Health Not Good variable, which is the response to "Now thinking about your mental health, which includes stress, depression, and problems with emotions, for how many days during the past 30 days was your mental health not good?"

- This is coded just like the `PHYSHLTH` variable, so we need to do the same cleaning we did there.

To clean up `MENTHLTH` to a new variable called `menthealth`, we'll need: - to convince R that the 77 and 99 values are in fact best interpreted as `NA`, and - to convince R that the 88 should be interpreted as 0.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(menthealth = MENTHLTH,
           menthealth = replace(menthealth, menthealth %in% c(77, 99), NA),
           menthealth = replace(menthealth, menthealth == 88, 0))

smart_ohio_raw %>% count(MENTHLTH, menthealth) %>% tail()
```

```
# A tibble: 6 x 3
  MENTHLTH menthealth     n
     <dbl>      <dbl> <int>
1       28         28     7
2       29         29    10
3       30         30   475
4       77         NA    86
5       88          0  4823
6       99         NA    28
```

2.5.4.3 POORHLTH and its cleanup to poorhealth

POORHLTH, the Poor Physical or Mental Health variable, which is the response to "During the past 30 days, for about how many days did poor physical or mental health keep you from doing your usual activities, such as self-care, work, or recreation?"

- Again, we recode just like the PHYSHLTH variable.

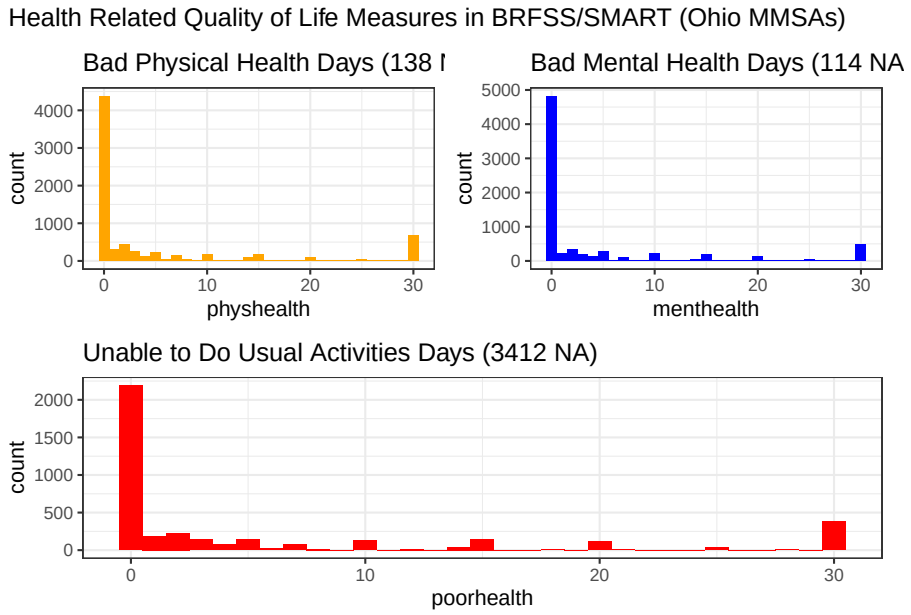
```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(poorhealth = POORHLTH,
           poorhealth = replace(poorhealth, poorhealth %in% c(77, 99), NA),
           poorhealth = replace(poorhealth, poorhealth == 88, 0))

smart_ohio_raw %>% count(POORHLTH, poorhealth) %>% tail()
```

```
# A tibble: 6 x 3
  POORHLTH poorhealth     n
     <dbl>      <dbl> <int>
1       29         29     4
2       30         30   382
3       77         NA    64
4       88          0  2194
5       99         NA    11
6       NA         NA  3337
```

There's a lot more missingness in the poorhealth counts than in the other health-related quality of life measures. There's also a strong mode at 0, and a smaller mode at 30 in each variable.

```
p1 <- ggplot(smart_ohio_raw, aes(x = physhealth)) +
    geom_histogram(binwidth = 1, fill = "orange") +
    labs(title = paste0("Bad Physical Health Days (",
                        sum(is.na(smart_ohio_raw$physhealth)),
                        " NA)"))

p2 <- ggplot(smart_ohio_raw, aes(x = menthealth)) +
```

```
    geom_histogram(binwidth = 1, fill = "blue") +
    labs(title = paste0("Bad Mental Health Days (",
                        sum(is.na(smart_ohio_raw$menthealth)),
                        " NA)"))

p3 <- ggplot(smart_ohio_raw, aes(x = poorhealth)) +
    geom_histogram(binwidth = 1, fill = "red") +
    labs(title = paste0("Unable to Do Usual Activities Days (",
                        sum(is.na(smart_ohio_raw$poorhealth)),
                        " NA)"))

(p1 + p2) / p3 +
    plot_annotation(title = "Health Related Quality of Life Measures in BRFSS/SMART (Ohio MMSAs)"
```



2.5.5 Health Care Access (4 items)

The next four variables relate to the subject's health care access.

2.5.5.1 `HLTHPLN1` and its cleanup to `healthplan`

`HLTHPLN1`, the Have any health care coverage variable, is the response to "Do you have any kind of health care coverage, including health insurance, prepaid

plans such as HMOs, or government plans such as Medicare, or Indian Health Service?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused

To clean up the `HLTHPLN1` data into a new variable called `healthplan` we'll - convince R that the 7 and 9 values are in fact best interpreted as `NA`, - and turn it into an indicator variable, e.g., we will leave the variable as numeric, but change the values to 1 = Yes and 0 = No.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(healthplan = HLTHPLN1,
           healthplan = replace(healthplan, healthplan %in% c(7, 9), NA),
           healthplan = replace(healthplan, healthplan == 2, 0))

smart_ohio_raw %>% count(HLTHPLN1, healthplan)
```

```
# A tibble: 4 x 3
  HLTHPLN1 healthplan     n
     <dbl>      <dbl> <int>
1        1          1  6994
2        2          0   398
3        7         NA    10
4        9         NA    10
```

2.5.5.2 `PERSDOC2` and its cleanup to `hasdoc` and to `numdocs2`

`PERSDOC2`, the Multiple Health Care Professionals variable, is the response to "Do you have one person you think of as your personal doctor or health care provider?" where if the response is "No," the survey then asks "Is there more than one or is there no person who you think of as your personal doctor or health care provider?"

- 1 = Yes, only one
- 2 = More than one
- 3 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

To clean up the `PERSDOC2` data into a new variable called `hasdoc` we'll - convince R that the 7 and 9 values are in fact best interpreted as `NA`, - and turn it into an indicator variable, e.g., we will leave the variable as numeric, but change the values to 1 = Yes and 0 = No, so that the original 1 and 2 become 1, and the original 3 becomes 0.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(hasdoc = PERSDOC2,
           hasdoc = replace(hasdoc, hasdoc %in% c(7, 9), NA),
           hasdoc = replace(hasdoc, hasdoc %in% c(1, 2), 1),
           hasdoc = replace(hasdoc, hasdoc == 3, 0))

smart_ohio_raw %>% count(PERSDOC2, hasdoc)
```

```
# A tibble: 5 x 3
  PERSDOC2 hasdoc     n
     <dbl>  <dbl> <int>
1        1      1  5784
2        2      1   623
3        3      0   990
4        7     NA    14
5        9     NA     1
```

2.5.5.3 MEDCOST and its cleanup to costprob

MEDCOST, the Could Not See Doctor Because of Cost variable, is the response to "Was there a time in the past 12 months when you needed to see a doctor but could not because of cost?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

This is just like HLTHPLAN.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(costprob = MEDCOST,
           costprob = replace(costprob, costprob %in% c(7, 9), NA),
           costprob = replace(costprob, costprob == 2, 0))

smart_ohio_raw %>% count(MEDCOST, costprob)
```

```
# A tibble: 4 x 3
  MEDCOST costprob     n
    <dbl>    <dbl> <int>
1       1        1   714
2       2        0  6680
3       7       NA    14
4       9       NA     4
```

2.5.5.4 CHECKUP1 and its cleanup to t_checkup

CHECKUP1, the Length of time since last routine checkup variable, is the response to "About how long has it been since you last visited a doctor for a routine checkup? [A routine checkup is a general physical exam, not an exam for a specific injury, illness, or condition.]"

- 1 = Within past year (anytime less than 12 months ago)
- 2 = Within past 2 years (1 year but less than 2 years ago)
- 3 = Within past 5 years (2 years but less than 5 years ago)
- 4 = 5 or more years ago
- 7 = Don't know/Not sure
- 8 = Never
- 9 = Refused
- BLANK = Not asked or missing

To clean up the CHECKUP1 data into a new variable called t_checkup we'll - convince R that the 7 and 9 values are in fact best interpreted as NA, - relabel options 1, 2, 3, 4 and 8 while turning the variable into a factor.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(t_checkup = fct_recode(factor(CHECKUP1),
                                  "1_In-past-year" = "1",
                                  "2_1-to-2-years" = "2",
                                  "3_2-to-5-years" = "3",
                                  "4_5_plus_years" = "4",
                                  "8_Never" = "8",
                                  NULL = "7",
                                  NULL = "9"))

smart_ohio_raw %>% count(CHECKUP1, t_checkup)
```

```
# A tibble: 7 x 3
  CHECKUP1 t_checkup          n
     <dbl> <fct>          <int>
1        1 1_In-past-year  5803
2        2 2_1-to-2-years   714
3        3 3_2-to-5-years   413
4        4 4_5_plus_years   376
5        7 <NA>              68
6        8 8_Never           32
7        9 <NA>               6
```

2.5.6 Blood Pressure (2 measures)

2.5.6.1 BPHIGH4 and its cleanup to bp_high

BPHIGH4 is asking about awareness of a hypertension diagnosis. It's the response to the question: "Have you EVER been told by a doctor, nurse or other health professional that you have high blood pressure?" In addition, if the answer was "Yes" and the respondent is female, they were then asked "Was this only when you were pregnant?"

The available codes are:

- 1 = Yes
- 2 = Yes, but female told only during pregnancy
- 3 = No
- 4 = Told borderline high or pre-hypertensive
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

To clean up the BPHIGH4 data into a new variable called bp_high we'll - convince R that the 7 and 9 values are in fact best interpreted as NA, - relabel (and re-order) options 1, 2, 3, 4 while turning the variable into a factor.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(bp_high = fct_recode(factor(BPHIGH4),
                                "0_No" = "3",
                                "1_Yes" = "1",
                                "2_Only_while_pregnant" = "2",
                                "4_Borderline" = "4",
                                NULL = "7",
                                NULL = "9"),
           bp_high = fct_relevel(bp_high,
                                "0_No", "1_Yes",
                                "2_Only_while_pregnant",
                                "4_Borderline"))

smart_ohio_raw %>% count(BPHIGH4, bp_high)
```

```
# A tibble: 6 x 3
  BPHIGH4 bp_high                 n
    <dbl> <fct>                <int>
1       1 1_Yes                 3161
2       2 2_Only_while_pregnant   67
3       3 0_No                  4114
4       4 4_Borderline            49
5       7 <NA>                    19
6       9 <NA>                     2
```

2.5.6.2 BPMEDS and its cleanup to bp_meds

BPMEDS is the response to the question "Are you currently taking medicine for your high blood pressure?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

To clean up the BPMEDS data into a new variable called bp_meds we'll treat it just as we did with HLTHPLN1 and - convince R that the 7 and 9 values are in fact best interpreted as NA, - and turn it into an indicator variable, e.g., we will leave the variable as numeric, but change the values to 1 = Yes and 0 = No.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(bp_meds = BPMEDS,
           bp_meds = replace(bp_meds, bp_meds %in% c(7, 9), NA),
           bp_meds = replace(bp_meds, bp_meds == 2, 0))

smart_ohio_raw %>% count(BPMEDS, bp_meds)
```

```
# A tibble: 5 x 3
  BPMEDS bp_meds     n
   <dbl>   <dbl> <int>
1      1       1  2675
2      2       0   481
3      7      NA     4
4      9      NA     1
5     NA      NA  4251
```

What is the relationship between our two blood pressure variables? Only the people with bp_meds = "1_Yes" were asked the bp_meds question.

```
smart_ohio_raw %>% tabyl(bp_high, bp_meds)
```

```
              bp_high   0    1  NA_
                 0_No   0    0 4114
                1_Yes 481 2675    5
 2_Only_while_pregnant   0    0   67
          4_Borderline   0    0   49
                  <NA>   0    0   21
```

2.5.7 Cholesterol (3 items)

2.5.7.1 CHOLCHK1 and its cleanup to t_chol

CHOLCHK1, the Length of time since cholesterol was checked, is the response to "Blood cholesterol is a fatty substance found in the blood. About how long has it been since you last had your blood cholesterol checked?"

- 1 = Never
- 2 = Within past year (anytime less than 12 months ago)
- 3 = Within past 2 years (1 year but less than 2 years ago)
- 4 = Within past 5 years (2 years but less than 5 years ago)
- 5 = 5 or more years ago
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

To clean up the CHOLCHK1 data into a new variable called t_chol we'll - convince R that the 7 and 9 values are in fact best interpreted as NA, - relabel options 1, 2, 3, 4 and 8 while turning the variable into a factor.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(t_chol = fct_recode(factor(CHOLCHK1),
                               "1_Never" = "1",
                               "2_In-past-year" = "2",
                               "3_1-to-2-years" = "3",
                               "4_2-to-5-years" = "4",
                               "5_5_plus_years" = "5",
                               NULL = "7",
                               NULL = "9"))

smart_ohio_raw %>% count(CHOLCHK1, t_chol)
```

```
# A tibble: 8 x 3
  CHOLCHK1 t_chol             n
     <dbl> <fct>          <int>
1        1 1_Never          424
2        2 2_In-past-year  5483
3        3 3_1-to-2-years   559
4        4 4_2-to-5-years   289
5        5 5_5_plus_years   272
6        7 <NA>             376
7        9 <NA>               8
8       NA <NA>               1
```

The next two measures are not gathered from the people who answered "Never" to this question.

2.5.7.2 TOLDHI2 and its cleanup to chol_high

TOLDHI2 is asking about awareness of a diagnosis of high cholesterol. It's the response to the question: "Have you EVER been told by a doctor, nurse or other health professional that your blood cholesterol is high?"

The available codes are:

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

To clean up the TOLDHI2 data into a new variable called chol_high we'll treat it like BPMEDS and HLTHPLN1 - convince R that the 7 and 9 values are in fact best interpreted as NA, - and turn it into an indicator variable, e.g., we will leave the variable as numeric, but change the values to 1 = Yes and 0 = No.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(chol_high = TOLDHI2,
           chol_high = replace(chol_high, chol_high %in% c(7, 9), NA),
           chol_high = replace(chol_high, chol_high == 2, 0))

smart_ohio_raw %>% count(TOLDHI2, chol_high)
```

```
# A tibble: 5 x 3
  TOLDHI2 chol_high     n
    <dbl>     <dbl> <int>
1       1         1  2612
2       2         0  4286
3       7        NA    70
4       9        NA     4
5      NA        NA   440
```

2.5.7.3 CHOLMED1 and its cleanup to chol_meds

CHOLMED1 is the response to the question "Are you currently taking medicine prescribed by a doctor or other health professional for your blood cholesterol?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

To clean up the CHOLMED1 data into a new variable called chol_meds we'll treat it just as we did with HLTHPLN1 and - convince R that the 7 and 9 values are in

fact best interpreted as NA, - and turn it into an indicator variable, e.g., we will leave the variable as numeric, but change the values to 1 = Yes and 0 = No.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(chol_meds = CHOLMED1,
           chol_meds = replace(chol_meds, chol_meds %in% c(7, 9), NA),
           chol_meds = replace(chol_meds, chol_meds == 2, 0))

smart_ohio_raw %>% count(CHOLMED1, chol_meds)
```

```
# A tibble: 4 x 3
  CHOLMED1 chol_meds     n
     <dbl>     <dbl> <int>
1        1         1  1781
2        2         0   826
3        7        NA     5
4       NA        NA  4800
```

2.5.8 Chronic Health Conditions (14 items)

2.5.8.1 Self-reported diagnosis history (11 items)

The next few variables describe whether or not the subject meets a particular standard, and are all coded in the raw data the same way:

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

and we'll recode them all to 1 = Yes, 0 = No, otherwise NA, as we've done previously.

The questions are all started with "Has a doctor, nurse, or other health professional ever told you that you had any of the following? For each, tell me Yes, No, or you're Not sure."

Original	Revised	Details
CVDINFR4	hx_mi	(Ever told) you had a heart attack, also called a myocardial infarction?
CVDCRHD4	hx_chd	(Ever told) you had angina or coronary heart disease?
CVDSTRK3	hx_stroke	(Ever told) you had a stroke?
ASTHMA3	hx_asthma	(Ever told) you had asthma?
ASTHNOW	now_asthma	Do you still have asthma? (only asked of those with Yes in ASTHMA3)

Original	Revised	Details
CHCSCNCR	hx_skinc	(Ever told) you had skin cancer?
CHCOCNCR	hx_otherc	(Ever told) you had any other types of cancer?
CHCCOPD1	hx_copd	(Ever told) you have Chronic Obstructive Pulmonary Disease or COPD, emphysema or chronic bronchitis?
HAVARTH3	hx_arthr	(Ever told) you have some form of arthritis, rheumatoid arthritis, gout, lupus, or fibromyalgia? (Arthritis diagnoses include: rheumatism, polymyalgia rheumatica; osteoarthritis (not osteporosis); tendonitis, bursitis, bunion, tennis elbow; carpal tunnel syndrome, tarsal tunnel syndrome; joint infection, etc.)
ADDEPEV2	hx_depress	(Ever told) you that you have a depressive disorder, including depression, major depression, dysthymia, or minor depression?
CHCKIDNY	hx_kidney	(Ever told) you have kidney disease? Do NOT include kidney stones, bladder infection or incontinence.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(hx_mi = CVDINFR4,
           hx_mi = replace(hx_mi, hx_mi %in% c(7, 9), NA),
           hx_mi = replace(hx_mi, hx_mi == 2, 0),
           hx_chd = CVDCRHD4,
           hx_chd = replace(hx_chd, hx_chd %in% c(7, 9), NA),
           hx_chd = replace(hx_chd, hx_chd == 2, 0),
           hx_stroke = CVDSTRK3,
           hx_stroke = replace(hx_stroke, hx_stroke %in% c(7, 9), NA),
           hx_stroke = replace(hx_stroke, hx_stroke == 2, 0),
           hx_asthma = ASTHMA3,
           hx_asthma = replace(hx_asthma, hx_asthma %in% c(7, 9), NA),
           hx_asthma = replace(hx_asthma, hx_asthma == 2, 0),
           now_asthma = ASTHNOW,
           now_asthma = replace(now_asthma, now_asthma %in% c(7, 9), NA),
           now_asthma = replace(now_asthma, now_asthma == 2, 0),
           hx_skinc = CHCSCNCR,
           hx_skinc = replace(hx_skinc, hx_skinc %in% c(7, 9), NA),
           hx_skinc = replace(hx_skinc, hx_skinc == 2, 0),
           hx_otherc = CHCOCNCR,
           hx_otherc = replace(hx_otherc, hx_otherc %in% c(7, 9), NA),
           hx_otherc = replace(hx_otherc, hx_otherc == 2, 0),
           hx_copd = CHCCOPD1,
           hx_copd = replace(hx_copd, hx_copd %in% c(7, 9), NA),
           hx_copd = replace(hx_copd, hx_copd == 2, 0),
           hx_arthr = HAVARTH3,
           hx_arthr = replace(hx_arthr, hx_arthr %in% c(7, 9), NA),
```

```
            hx_arthr = replace(hx_arthr, hx_arthr == 2, 0),
            hx_depress = ADDEPEV2,
            hx_depress = replace(hx_depress, hx_depress %in% c(7, 9), NA),
            hx_depress = replace(hx_depress, hx_depress == 2, 0),
            hx_kidney = CHCKIDNY,
            hx_kidney = replace(hx_kidney, hx_kidney %in% c(7, 9), NA),
            hx_kidney = replace(hx_kidney, hx_kidney == 2, 0))
```

We definitely should have written a function to do that, of course.

2.5.8.2 `_ASTHMS1` and its cleanup to `asthma`

`_ASTHMS1` categorizes subjects by asthma status as:

- 1 = Current
- 2 = Former
- 3 = Never
- 9 = Don't Know / Not Sure / Refused / Missing

We'll turn this into a factor with appropriate levels and NA information.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(asthma = fct_recode(
        factor(`_ASTHMS1`),
        "Current" = "1",
        "Former" = "2",
        "Never" = "3",
        NULL = "9"))

smart_ohio_raw %>% count(`_ASTHMS1`, asthma)
```

```
# A tibble: 4 x 3
  `_ASTHMS1` asthma      n
       <dbl> <fct>   <int>
1          1 Current   734
2          2 Former    248
3          3 Never    6376
4          9 <NA>       54
```

2.5.8.3 `DIABETE3` and its cleanup to `hx_diabetes` and `dm_status`

`DIABETE3`, the (Ever told) you have diabetes variable, is the response to "(Ever told) you have diabetes (If Yes and respondent is female, ask Was this only when you were pregnant?. If Respondent says pre-diabetes or borderline diabetes, use response code 4.)"

- 1 = Yes
- 2 = Yes, but female told only during pregnancy
- 3 = No
- 4 = No, pre-diabetes or borderline diabetes
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

I'll create one variable called `hx_diabetes` which is 1 if `DIABETE3` = 1, and 0 otherwise, with appropriate NAs, like our other variables. Then I'll create `dm_status` to include all of this information in a factor, but again recode the missing values properly.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(hx_diabetes = DIABETE3,
           hx_diabetes = replace(hx_diabetes, hx_diabetes %in% c(7, 9), NA),
           hx_diabetes = replace(hx_diabetes, hx_diabetes %in% 2:4, 0),
           dm_status = fct_recode(factor(DIABETE3),
                                  "Diabetes" = "1",
                                  "Pregnancy-Induced" = "2",
                                  "No-Diabetes" = "3",
                                  "Pre-Diabetes" = "4",
                                  NULL = "7",
                                  NULL = "9"),
           dm_status = fct_relevel(dm_status,
                                   "No-Diabetes",
                                   "Pre-Diabetes",
                                   "Pregnancy-Induced",
                                   "Diabetes"))

smart_ohio_raw %>% count(DIABETE3, hx_diabetes, dm_status)
```

```
# A tibble: 6 x 4
  DIABETE3 hx_diabetes dm_status            n
     <dbl>       <dbl> <fct>             <int>
1        1           1 Diabetes           1098
2        2           0 Pregnancy-Induced    67
3        3           0 No-Diabetes        6100
4        4           0 Pre-Diabetes        133
5        7          NA <NA>                 12
6        9          NA <NA>                  2
```

2.5.8.4 `DIABAGE2` and its cleanup to `dm_age`

`DIABAGE2`, the Age When Told Diabetic variable, is the response to "How old were you when you were told you have diabetes?" It is asked only of people with

`DIABETE3 = 1` (Yes).

- The response is 1-97, with special values 98 for Don't Know/Not Sure and 99 for refused, with BLANK for missing or not asked. People 97 years of age and above were listed as 97.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(dm_age = DIABAGE2,
           dm_age = replace(dm_age, dm_age > 97, NA))

smart_ohio_raw %>% count(DIABAGE2, dm_age) %>% tail()
```

```
# A tibble: 6 x 3
  DIABAGE2 dm_age     n
     <dbl>  <dbl> <int>
1       84     84     1
2       85     85     2
3       90     90     1
4       98     NA    61
5       99     NA     4
6       NA     NA  6314
```

2.5.9 Arthritis Burden (4 items)

The first two measures are only asked of people with `hx_arthr` = 1, and are coded as:

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

and we'll recode them to 1 = Yes, 0 = No, otherwise NA, as we've done previously.

2.5.9.1 `LMTJOIN3` (Limited because of joint symptoms), and its cleanup to `arth_lims`

This is the response to "Are you now limited in any way in any of your usual activities because of arthritis or joint symptoms?"

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(arth_lims = LMTJOIN3,
           arth_lims = replace(arth_lims, arth_lims %in% c(7, 9), NA),
           arth_lims = replace(arth_lims, arth_lims == 2, 0))

smart_ohio_raw %>% count(hx_arthr, LMTJOIN3, arth_lims)
```

```
# A tibble: 6 x 4
  hx_arthr LMTJOIN3 arth_lims     n
     <dbl>    <dbl>     <dbl> <int>
1        0       NA        NA  4587
2        1        1         1  1378
3        1        2         0  1388
4        1        7        NA    17
5        1        9        NA     2
6       NA       NA        NA    40
```

2.5.9.2 ARTHDIS2 (Does Arthritis Affect Whether You Work), and its cleanup to arth_work

This is the response to "Do arthritis or joint symptoms now affect whether you work, the type of work you do or the amount of work you do?"

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(arth_work = ARTHDIS2,
           arth_work = replace(arth_work, arth_work %in% c(7, 9), NA),
           arth_work = replace(arth_work, arth_work == 2, 0))

smart_ohio_raw %>% count(ARTHDIS2, arth_work)
```

```
# A tibble: 5 x 3
  ARTHDIS2 arth_work     n
     <dbl>     <dbl> <int>
1        1         1   925
2        2         0  1808
3        7        NA    42
4        9        NA    10
5       NA        NA  4627
```

2.5.9.3 ARTHSOCL (Social Activities Limited Because of Joint Symptoms) and its cleanup to arth_soc

This is the response to "During the past 30 days, to what extent has your arthritis or joint symptoms interfered with your normal social activities, such as going shopping, to the movies, or to religious or social gatherings?"

The responses are:

- 1 = A lot
- 2 = A little
- 3 = Not at all
- 7 = Don't know/Not sure
- 9 = Refused

- BLANK = Not asked or missing

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(arth_soc = fct_recode(factor(ARTHSOCL),
                                 "A lot" = "1",
                                 "A little" = "2",
                                 "Not at all" = "3",
                                 NULL = "7",
                                 NULL = "9"))

smart_ohio_raw %>% count(ARTHSOCL, arth_soc)
```

```
# A tibble: 6 x 3
  ARTHSOCL arth_soc       n
     <dbl> <fct>      <int>
1        1 A lot        606
2        2 A little     734
3        3 Not at all  1427
4        7 <NA>          15
5        9 <NA>           3
6       NA <NA>        4627
```

2.5.9.4 JOINPAI1 (How Bad Was Joint Pain - scale of 0-10) and its cleanup to joint_pain

This is the response to the following question: "Please think about the past 30 days, keeping in mind all of your joint pain or aching and whether or not you have taken medication. On a scale of 0 to 10 where 0 is no pain or aching and 10 is pain or aching as bad as it can be, DURING THE PAST 30 DAYS, how bad was your joint pain ON AVERAGE?"

The available values are 0-10, plus codes 77 (Don't Know / Not Sure), 99 (Refused) and BLANK.

To clean up `JOINPAI1` to a new variable called `joint_pain`, we'll need to convince R that the 77 and 99 values are, like `BLANK`, in fact best interpreted as `NA`.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(joint_pain = JOINPAI1,
           joint_pain = replace(joint_pain, joint_pain %in% c(77, 99), NA))

smart_ohio_raw %>% count(JOINPAI1, joint_pain) %>% tail()
```

```
# A tibble: 6 x 3
  JOINPAI1 joint_pain     n
     <dbl>      <dbl> <int>
1        8          8   277
```

```
2        9          9    72
3       10         10   158
4       77         NA    28
5       99         NA     5
6       NA         NA  4627
```

2.5.10 Demographics (25 items)

2.5.10.1 `_AGEG5YR`, which we'll edit into `agegroup`

The `_AGEG5YR` variable is a calculated variable (by CDC) obtained from the subject's age. Since the `age` data are not available, we instead get these groupings, which we'll rearrange into the `agegroup` factor.

`_AGEG5YR`	Age range	`agegroup`
1	18 <= AGE <= 24	18-24
2	25 <= AGE <= 29	25-29
3	30 <= AGE <= 34	30-34
4	35 <= AGE <= 39	35-39
5	40 <= AGE <= 44	40-44
6	45 <= AGE <= 49	45-49
7	50 <= AGE <= 54	50-54
8	55 <= AGE <= 59	55-59
9	60 <= AGE <= 64	60-64
10	65 <= AGE <= 69	65-69
11	70 <= AGE <= 74	70-74
12	75 <= AGE <= 79	75-79
13	AGE >= 80	80plus
14	Don't Know, Refused or Missing	NA

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(agegroup = fct_recode(factor(`_AGEG5YR`),
                                 "18-24" = "1",
                                 "25-29" = "2",
                                 "30-34" = "3",
                                 "35-39" = "4",
                                 "40-44" = "5",
                                 "45-49" = "6",
                                 "50-54" = "7",
                                 "55-59" = "8",
                                 "60-64" = "9",
                                 "65-69" = "10",
                                 "70-74" = "11",
```

```
                                  "75-79" = "12",
                                  "80-96" = "13",
                                  NULL = "14"))

smart_ohio_raw %>% count(`_AGEG5YR`, agegroup)
```

```
# A tibble: 14 x 3
   `_AGEG5YR` agegroup     n
        <dbl> <fct>    <int>
 1          1 18-24      448
 2          2 25-29      327
 3          3 30-34      375
 4          4 35-39      446
 5          5 40-44      426
 6          6 45-49      509
 7          7 50-54      604
 8          8 55-59      786
 9          9 60-64      837
10         10 65-69      810
11         11 70-74      685
12         12 75-79      499
13         13 80-96      592
14         14 <NA>        68
```

2.5.10.2 _MRACE1 recoded to race

We'll create three variables describing race/ethnicity. The first comes from the _MRACE1 variable categorized by CDC, and the available responses are:

- 1 = White only
- 2 = Black or African-American only
- 3 = American Indian or Alaskan Native only
- 4 = Asian only
- 5 = Native Hawaiian or Pacific Islander only
- 6 = Other race only
- 7 = Multiracial
- 77 = Don't know / Not Sure
- 99 = Refused
- BLANK = Missing

We'll create a factor out of this information, with appropriate level names.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(race = fct_recode(factor(`_MRACE1`),
                                  "White" = "1",
                                  "Black or African A" = "2",
```

```
                                    "Amer Indian or Alaskan" = "3",
                                    "Asian" = "4",
                                    "Hawaiian or Pac Island" = "5",
                                    "Other Race" = "6",
                                    "Multiracial" = "7",
                                    NULL = "77",
                                    NULL = "99"))

smart_ohio_raw %>% count(`_MRACE1`, race)
```

```
# A tibble: 9 x 3
  `_MRACE1` race                      n
      <dbl> <fct>                 <int>
1         1 White                  6177
2         2 Black or African A      739
3         3 Amer Indian or Alaskan   66
4         4 Asian                   115
5         5 Hawaiian or Pac Island    5
6         6 Other Race               43
7         7 Multiracial             153
8        77 <NA>                     14
9        99 <NA>                    100
```

2.5.10.3 _HISPANC recoded to hispanic

The _HISPANC variable specifies whether or not the respondent is of Hispanic or Latinx origin. The available responses are:

- 1 = Hispanic, Latinx or Spanish origin
- 2 = Not of Hispanic, Latinx or Spanish origin
- 9 = Don't Know, Refused, or Missing

We'll turn the 9s into NA, and create an indicator variable (1 = Hispanic or Latinx, 0 = not)

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(hispanic = 2 - `_HISPANC`,
           hispanic = replace(hispanic, hispanic < 0, NA))

smart_ohio_raw %>% count(`_HISPANC`, hispanic)
```

```
# A tibble: 3 x 3
  `_HISPANC` hispanic     n
       <dbl>    <dbl> <int>
1          1        1   146
2          2        0  7217
```

```
3          9       NA    49
```

2.5.10.4 _RACEGR3 recoded to race_eth

The _RACEGR3 variable is a five-level combination of race and ethnicity. The responses are:

- 1 = White non-Hispanic
- 2 = Black non-Hispanic
- 3 = Other race non-Hispanic
- 4 = Multiracial non-Hispanic
- 5 = Hispanic
- 9 = Don't Know / Not Sure / Refused

We'll create a factor out of this information, with appropriate level names.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(race_eth = fct_recode(
        factor(`_RACEGR3`),
        "White non-Hispanic" = "1",
        "Black non-Hispanic" = "2",
        "Other race non-Hispanic" = "3",
        "Multiracial non-Hispanic" = "4",
        "Hispanic" = "5",
        NULL = "9"))

smart_ohio_raw %>% count(`_RACEGR3`, race_eth)
```

```
# A tibble: 6 x 3
  `_RACEGR3` race_eth                    n
       <dbl> <fct>                   <int>
1          1 White non-Hispanic       6086
2          2 Black non-Hispanic        725
3          3 Other race non-Hispanic   193
4          4 Multiracial non-Hispanic  143
5          5 Hispanic                  146
6          9 <NA>                      119
```

2.5.10.5 SEX recoded to female

The available levels of SEX are:

- 1 = Male
- 2 = Female
- 9 = Refused

We'll recode that to `female` = 1 for Female, 0 Male, otherwise NA. Note the trick here is to subtract one from the coded `SEX` to get the desired `female`, but this requires that we move 9 to NA, rather than 9.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(female = SEX - 1,
           female = replace(female, female == 8, NA))

smart_ohio_raw %>% count(SEX, female)
```

```
# A tibble: 2 x 3
    SEX female     n
  <dbl>  <dbl> <int>
1     1      0  3136
2     2      1  4276
```

2.5.10.6 `MARITAL` status, revised to `marital`

The available levels of `MARITAL` are:

- 1 = Married
- 2 = Divorced
- 3 = Widowed
- 4 = Separated
- 5 = Never married
- 6 = A member of an unmarried couple
- 9 = Refused
- BLANK = Not asked or missing

We'll just turn this into a factor, and move 9 to NA.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(marital = fct_recode(factor(MARITAL),
                                "Married" = "1",
                                "Divorced" = "2",
                                "Widowed" = "3",
                                "Separated" = "4",
                                "Never_Married" = "5",
                                "Unmarried_Couple" = "6",
                                NULL = "9"))

smart_ohio_raw %>% count(MARITAL, marital)
```

```
# A tibble: 7 x 3
  MARITAL marital             n
    <dbl> <fct>           <int>
1       1 Married          3668
```

```
2       2 Divorced          1110
3       3 Widowed            978
4       4 Separated          142
5       5 Never_Married     1248
6       6 Unmarried_Couple   208
7       9 <NA>                58
```

2.5.10.7 `EDUCA` recoded to `educgroup`

The available levels of `EDUCA` (Education Level) are responses to: "What is the highest grade or year of school you completed?"

- 1 = Never attended school or only kindergarten
- 2 = Grades 1 through 8 (Elementary)
- 3 = Grades 9 through 11 (Some high school)
- 4 = Grade 12 or GED (High school graduate)
- 5 = College 1 year to 3 years (Some college or technical school)
- 6 = College 4 years or more (College graduate)
- 9 = Refused
- BLANK = Not asked or missing

We'll just turn this into a factor, and move 9 to NA.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(educgroup = fct_recode(factor(EDUCA),
                                  "Kindergarten" = "1",
                                  "Elementary" = "2",
                                  "Some_HS" = "3",
                                  "HS_Grad" = "4",
                                  "Some_College" = "5",
                                  "College_Grad" = "6",
                                  NULL = "9"))

smart_ohio_raw %>% count(EDUCA, educgroup)
```

```
# A tibble: 7 x 3
  EDUCA educgroup        n
  <dbl> <fct>        <int>
1     1 Kindergarten     3
2     2 Elementary     117
3     3 Some_HS        332
4     4 HS_Grad       2209
5     5 Some_College  2079
6     6 College_Grad  2646
7     9 <NA>            26
```

2.5.10.8 RENTHOM1 recoded to home_own

The available levels of RENTHOM1 (Own or Rent Home) are responses to: "Do you own or rent your home? (Home is defined as the place where you live most of the time/the majority of the year.)"

- 1 = Own
- 2 = Rent
- 3 = Other Arrangement
- 7 = Don't know/Not Sure
- 9 = Refused
- BLANK = Not asked or missing

We'll recode as home_own = 1 if they own their home, and 0 otherwise, and dealing with missingness properly.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(home_own = RENTHOM1,
           home_own = replace(home_own, home_own %in% c(7,9), NA),
           home_own = replace(home_own, home_own %in% c(2,3), 0))

smart_ohio_raw %>% count(RENTHOM1, home_own)
```

```
# A tibble: 5 x 3
  RENTHOM1 home_own     n
     <dbl>    <dbl> <int>
1        1        1  5216
2        2        0  1793
3        3        0   348
4        7       NA    28
5        9       NA    27
```

2.5.10.9 CPDEMO1A and its cleanup to cell_own

CPDEMO1A is the response to "Including phones for business and personal use, do you have a cell phone for personal use?"

Available responses are:

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

and we'll recode them to 1 = Yes, 0 = No, otherwise NA, as we've done previously.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(cell_own = 2 - CPDEMO1A,
           cell_own = replace(cell_own, cell_own < 0, NA))

smart_ohio_raw %>% count(CPDEMO1A, cell_own)
```

```
# A tibble: 5 x 3
  CPDEMO1A cell_own     n
     <dbl>    <dbl> <int>
1        1        1  2930
2        2        0   698
3        7       NA     2
4        9       NA    19
5       NA       NA  3763
```

2.5.10.10 VETERAN3 and its cleanup to veteran

VETERAN3, the Are You A Veteran variable, is the response to "Have you ever served on active duty in the United States Armed Forces, either in the regular military or in a National Guard or military reserve unit? (Active duty does not include training for the Reserves or National Guard, but DOES include activation, for example, for the Persian Gulf War.)"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(veteran = VETERAN3,
           veteran = replace(veteran, veteran %in% c(7, 9), NA),
           veteran = replace(veteran, veteran == 2, 0))

smart_ohio_raw %>% count(VETERAN3, veteran)
```

```
# A tibble: 3 x 3
  VETERAN3 veteran     n
     <dbl>   <dbl> <int>
1        1       1   927
2        2       0  6479
3        9      NA     6
```

2.5.10.11 EMPLOY1 and its cleanup to employment

EMPLOY1, the Employment Status variable, is the response to "Are you currently ... ?"

- 1 = Employed for wages
- 2 = Self-employed
- 3 = Out of work for 1 year or more
- 4 = Out of work for less than 1 year
- 5 = A homemaker
- 6 = A student
- 7 = Retired
- 8 = Unable to work
- 9 = Refused
- BLANK = Not asked or missing

We'll just turn this into a factor, and move 9 to NA.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(employment = fct_recode(factor(EMPLOY1),
                                   "Employed_for_wages" = "1",
                                   "Self-employed" = "2",
                                   "Outofwork_1yearormore" = "3",
                                   "Outofwork_lt1year" = "4",
                                   "Homemaker" = "5",
                                   "Student" = "6",
                                   "Retired" = "7",
                                   "Unable_to_work" = "8",
                                   NULL = "9"))

smart_ohio_raw %>% count(EMPLOY1, employment)
```

```
# A tibble: 9 x 3
  EMPLOY1 employment               n
    <dbl> <fct>                <int>
1       1 Employed_for_wages    3119
2       2 Self-employed          466
3       3 Outofwork_1yearormore  254
4       4 Outofwork_lt1year      134
5       5 Homemaker              411
6       6 Student                190
7       7 Retired               2202
8       8 Unable_to_work         603
9       9 <NA>                    33
```

2.5.10.12 `CHILDREN` and its cleanup to `kids`

`CHILDREN`, the Number of Children in Household variable, is the response to "How many children less than 18 years of age live in your household?"

- 1-87 = legitimate responses
- 88 = None
- 99 = Refused
- BLANK = Not asked or missing

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(kids = CHILDREN,
           kids = replace(kids, kids == 99, NA),
           kids = replace(kids, kids == 88, 0))

smart_ohio_raw %>% count(CHILDREN, kids) %>% tail()
```

```
# A tibble: 6 x 3
  CHILDREN  kids     n
     <dbl> <dbl> <int>
1        6     6     7
2        7     7     5
3        8     8     2
4       12    12     1
5       88     0  5449
6       99    NA    43
```

2.5.10.13 `INCOME2` to `incomegroup`

The available levels of `INCOME2` (Income Level) are responses to: "Is your annual household income from all sources . . . "

- 1 = Less than $10,000
- 2 = $10,000 to less than $15,000
- 3 = $15,000 to less than $20,000
- 4 = $20,000 to less than $25,000
- 5 = $25,000 to less than $35,000
- 6 = $35,000 to less than $50,000
- 7 = $50,000 to less than $75,000
- 8 = $75,000 or more
- 77 = Don't know/Not sure
- 99 = Refused
- BLANK = Not asked or missing

We'll just turn this into a factor, and move 77 and 99 to NA.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(incomegroup = fct_recode(factor(`INCOME2`),
                                    "0-9K" = "1",
                                    "10-14K" = "2",
                                    "15-19K" = "3",
                                    "20-24K" = "4",
                                    "25-34K" = "5",
                                    "35-49K" = "6",
                                    "50-74K" = "7",
                                    "75K+" = "8",
                                    NULL = "77",
                                    NULL = "99"))

smart_ohio_raw %>% count(`INCOME2`, incomegroup)
```

```
# A tibble: 11 x 3
   INCOME2 incomegroup     n
     <dbl> <fct>       <int>
 1       1 0-9K          285
 2       2 10-14K        306
 3       3 15-19K        477
 4       4 20-24K        589
 5       5 25-34K        685
 6       6 35-49K        922
 7       7 50-74K        928
 8       8 75K+         1910
 9      77 <NA>          610
10      99 <NA>          678
11      NA <NA>           22
```

2.5.10.14 `INTERNET` and its cleanup to `internet30`

`INTERNET`, the Internet use in the past 30 days variable, is the response to "Have you used the internet in the past 30 days?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(internet30 = INTERNET,
           internet30 = replace(internet30, internet30 %in% c(7, 9), NA),
           internet30 = replace(internet30, internet30 == 2, 0))
```

```
smart_ohio_raw %>% count(INTERNET, internet30)
```

```
# A tibble: 5 x 3
  INTERNET internet30     n
     <dbl>      <dbl> <int>
1        1          1  6020
2        2          0  1335
3        7         NA    10
4        9         NA    10
5       NA         NA    37
```

2.5.10.15 WTKG3 is weight_kg

WTKG3 is computed by CDC, as the respondent's weight in kilograms with two implied decimal places. We calculate the actual weight in kg, with the following:

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(weight_kg = WTKG3/100)

smart_ohio_raw %>% count(WTKG3, weight_kg) %>% tail()
```

```
# A tibble: 6 x 3
  WTKG3 weight_kg     n
  <dbl>     <dbl> <int>
1 19051      191.     1
2 19278      193.     1
3 19504      195.     1
4 20412      204.     2
5 20865      209.     1
6    NA       NA    462
```

2.5.10.16 HEIGHT3 is replaced with height_m

HEIGHT3 is strangely gathered to allow people to specify their height in either feet and inches or in meters and centimeters.

- 200-711 indicates height in feet (first digit) and inches (second two digits)
- 9000 - 9998 indicates height in meters (second digit) and centimeters (last two digits)
- 7777 = Don't know/Not sure
- 9999 = Refused

Note that there is one impossible value of 575 in the data set. We'll make that an NA, and we'll also make NA any heights below 3 feet, or above 2.24 meters. Specifically, we calculate the actual height in meters, with the following:

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(height_m = case_when(
        HEIGHT3 >= 300 & HEIGHT3 <= 511 ~ round((12*floor(HEIGHT3/100) + (HEIGHT3 - 10
        HEIGHT3 >= 600 & HEIGHT3 <= 711 ~ round((12*floor(HEIGHT3/100) + (HEIGHT3 - 10
        HEIGHT3 >= 9000 & HEIGHT3 <= 9224 ~ ((HEIGHT3 - 9000)/100)))

smart_ohio_raw %>% count(HEIGHT3, height_m) %>% tail()
```

```
# A tibble: 6 x 3
  HEIGHT3 height_m     n
    <dbl>    <dbl> <int>
1     607     2.01     2
2     608     2.03     6
3     609     2.06     1
4    7777    NA       27
5    9999    NA       86
6      NA    NA       67
```

2.5.10.17 `bmi` is calculated from `height_m` and `weight_kg`

We'll calculate body-mass index from height and weight.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(bmi = round(weight_kg/(height_m)^2,2))

smart_ohio_raw %>% count(height_m, weight_kg, bmi)# %>% tail()
```

```
# A tibble: 1,806 x 4
   height_m weight_kg   bmi     n
      <dbl>     <dbl> <dbl> <int>
 1     1.35      39.0  21.4     1
 2     1.35      52.2  28.6     1
 3     1.4       89.8  45.8     1
 4     1.42      31.8  15.8     1
 5     1.42      45.4  22.5     1
 6     1.42      55.8  27.7     1
 7     1.42      58.5  29.0     1
 8     1.42      59.9  29.7     1
 9     1.42      60.8  30.1     1
10     1.42      71.2  35.3     1
# ... with 1,796 more rows
```

2.5.10.18 `bmigroup` is calculated from `bmi`

We'll then divide the respondents into adult BMI categories, in the usual way.

- BMI < 18.5 indicates underweight
- BMI from 18.5 up to 25 indicates normal weight
- BMI from 25 up to 30 indicates overweight
- BMI of 30 and higher indicates obesity

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(bmigroup = factor(cut2(as.numeric(bmi),
                             cuts = c(18.5, 25.0, 30.0))))

smart_ohio_raw %>% count(bmigroup)
```

```
# A tibble: 5 x 2
  bmigroup        n
* <fct>       <int>
1 [13.3,18.5)   119
2 [18.5,25.0)  2017
3 [25.0,30.0)  2445
4 [30.0,75.5]  2338
5 <NA>          493
```

2.5.10.19 PREGNANT and its cleanup to pregnant

PREGNANT, the Pregnancy Status variable, is the response to "To your knowledge, are you now pregnant?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing (includes SEX = male)

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(pregnant = PREGNANT,
           pregnant = replace(pregnant, pregnant %in% c(7, 9), NA),
           pregnant = replace(pregnant, pregnant == 2, 0))

smart_ohio_raw %>% count(PREGNANT, pregnant)
```

```
# A tibble: 5 x 3
  PREGNANT pregnant     n
     <dbl>    <dbl> <int>
1        1        1    41
2        2        0  1329
3        7       NA     3
4        9       NA     3
5       NA       NA  6036
```

2.5.10.20 DEAF and its cleanup to deaf

DEAF, the Are you deaf or do you have serious difficulty hearing variable, is the response to "Are you deaf or do you have serious difficulty hearing?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(deaf = DEAF,
           deaf = replace(deaf, deaf %in% c(7, 9), NA),
           deaf = replace(deaf, deaf == 2, 0))

smart_ohio_raw %>% count(DEAF, deaf)
```

```
# A tibble: 5 x 3
   DEAF  deaf     n
  <dbl> <dbl> <int>
1     1     1   708
2     2     0  6551
3     7    NA    15
4     9    NA     4
5    NA    NA   134
```

2.5.10.21 BLIND and its cleanup to blind

BLIND, the Blind or Difficulty seeing variable, is the response to "Are you blind or do you have serious difficulty seeing, even when wearing glasses?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(blind = BLIND,
           blind = replace(blind, blind %in% c(7, 9), NA),
           blind = replace(blind, blind == 2, 0))

smart_ohio_raw %>% count(BLIND, blind)
```

```
# A tibble: 5 x 3
  BLIND blind     n
```

```
  <dbl> <dbl> <int>
1     1     1   415
2     2     0  6834
3     7    NA    14
4     9    NA     1
5    NA    NA   148
```

2.5.10.22 DECIDE and its cleanup to decide

DECIDE, the Difficulty Concentrating or Remembering variable, is the response to "Because of a physical, mental, or emotional condition, do you have serious difficulty concentrating, remembering, or making decisions?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(decide = DECIDE,
           decide = replace(decide, decide %in% c(7, 9), NA),
           decide = replace(decide, decide == 2, 0))

smart_ohio_raw %>% count(DECIDE, decide)
```

```
# A tibble: 5 x 3
  DECIDE decide     n
   <dbl>  <dbl> <int>
1      1      1   870
2      2      0  6348
3      7     NA    30
4      9     NA     2
5     NA     NA   162
```

2.5.10.23 DIFFWALK and its cleanup to diffwalk

DIFFWALK, the Difficulty Walking or Climbing Stairs variable, is the response to "Do you have serious difficulty walking or climbing stairs?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(diffwalk = DIFFWALK,
           diffwalk = replace(diffwalk, diffwalk %in% c(7, 9), NA),
           diffwalk = replace(diffwalk, diffwalk == 2, 0))

smart_ohio_raw %>% count(DIFFWALK, diffwalk)
```

```
# A tibble: 5 x 3
  DIFFWALK diffwalk     n
     <dbl>    <dbl> <int>
1        1        1  1482
2        2        0  5738
3        7       NA    19
4        9       NA     2
5       NA       NA   171
```

2.5.10.24 DIFFDRES and its cleanup to diffdress

DIFFDRES, the Difficulty Dressing or Bathing variable, is the response to "Do you have difficulty dressing or bathing?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(diffdress = DIFFDRES,
           diffdress = replace(diffdress, diffdress %in% c(7, 9), NA),
           diffdress = replace(diffdress, diffdress == 2, 0))

smart_ohio_raw %>% count(DIFFDRES, diffdress)
```

```
# A tibble: 5 x 3
  DIFFDRES diffdress     n
     <dbl>     <dbl> <int>
1        1         1   352
2        2         0  6868
3        7        NA    12
4        9        NA     1
5       NA        NA   179
```

2.5.10.25 DIFFALON and its cleanup to diffalone

DIFFALON, the Difficulty Doing Errands Alone variable, is the response to "Because of a physical, mental, or emotional condition, do you have difficulty doing errands alone such as visiting a doctor's office or shopping?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(diffalone = DIFFALON,
           diffalone = replace(diffalone, diffalone %in% c(7, 9), NA),
           diffalone = replace(diffalone, diffalone == 2, 0))

smart_ohio_raw %>% count(DIFFALON, diffalone)
```

```
# A tibble: 5 x 3
  DIFFALON diffalone     n
     <dbl>     <dbl> <int>
1        1         1   636
2        2         0  6560
3        7        NA    15
4        9        NA     4
5       NA        NA   197
```

2.5.11 Tobacco Use (2 items)

2.5.11.1 SMOKE100 and its cleanup to smoke100

SMOKE100, the Smoked at Least 100 Cigarettes variable, is the response to "Have you smoked at least 100 cigarettes in your entire life? [Note: 5 packs = 100 cigarettes]"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(smoke100 = SMOKE100,
           smoke100 = replace(smoke100, smoke100 %in% c(7, 9), NA),
           smoke100 = replace(smoke100, smoke100 == 2, 0))

smart_ohio_raw %>% count(SMOKE100, smoke100)
```

```
# A tibble: 5 x 3
  SMOKE100 smoke100     n
     <dbl>    <dbl> <int>
1        1        1  3294
2        2        0  3881
3        7       NA    31
4        9       NA     4
5       NA       NA   202
```

2.5.11.2 _SMOKER3 and its cleanup to smoker

_SMOKER3, is a calculated variable which categorizes subjects by their smoking status:

- 1 = Current smoker who smokes daily
- 2 = Current smoker but not every day
- 3 = Former smoker
- 4 = Never smoked
- 9 = Don't Know / Refused / Missing

We'll reclassify this as a factor with appropriate labels and NAs.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(smoker = fct_recode(factor(`_SMOKER3`),
                               "Current_daily" = "1",
                               "Current_not_daily" = "2",
                               "Former" = "3",
                               "Never" = "4",
                               NULL = "9"))

smart_ohio_raw %>% count(`_SMOKER3`, smoker)
```

```
# A tibble: 5 x 3
  `_SMOKER3` smoker                n
       <dbl> <fct>             <int>
1          1 Current_daily       990
2          2 Current_not_daily   300
3          3 Former             1999
4          4 Never              3881
5          9 <NA>                242
```

2.5.12 E-Cigarettes (2 items)

2.5.12.1 ECIGARET and its cleanup to ecig_ever

ECIGARET, the Ever used an e-cigarette variable, is the response to "Have you ever used an e-cigarette or other electronic vaping product, even just one time, in your entire life?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(ecig_ever = ECIGARET,
           ecig_ever = replace(ecig_ever, ecig_ever %in% c(7, 9), NA),
           ecig_ever = replace(ecig_ever, ecig_ever == 2, 0))

smart_ohio_raw %>% count(ECIGARET, ecig_ever)
```

```
# A tibble: 5 x 3
  ECIGARET ecig_ever     n
     <dbl>     <dbl> <int>
1        1         1  1354
2        2         0  5799
3        7        NA     9
4        9        NA     3
5       NA        NA   247
```

2.5.12.2 _ECIGSTS and its cleanup to ecigs

_ECIGSTS, is a calculated variable which categorizes subjects by their smoking status:

- 1 = Current and uses daily
- 2 = Current user but not every day
- 3 = Former user
- 4 = Never used e-cigarettes
- 9 = Don't Know / Refused / Missing

We'll reclassify this as a factor with appropriate labels and NAs.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(ecigs = fct_recode(factor(`_ECIGSTS`),
                              "Current_daily" = "1",
                              "Current_not_daily" = "2",
                              "Former" = "3",
```

```
                                  "Never" = "4",
                                  NULL = "9"))

smart_ohio_raw %>% count(`_ECIGSTS`, ecigs)
```

```
# A tibble: 5 x 3
  `_ECIGSTS` ecigs                n
       <dbl> <fct>              <int>
1          1 Current_daily        102
2          2 Current_not_daily    165
3          3 Former              1085
4          4 Never               5799
5          9 <NA>                 261
```

2.5.13 Alcohol Consumption (6 items)

2.5.13.1 ALCDAY5 and its cleanup to alcdays

ALCDAY5, the Days in past 30 had alcoholic beverage variable, is the response to "During the past 30 days, how many days per week or per month did you have at least one drink of any alcoholic beverage such as beer, wine, a malt beverage or liquor?"

- 101-107 = # of days per week (101 = 1 day per week, 107 = 7 days per week)
- 201-230 = # of days in past 30 days (201 = 1 day in last 30, 230 = 30 days in last 30)
- 777 = Don't know/Not sure
- 888 = No drinks in past 30 days
- 999 = Refused
- BLANK = Not asked or Missing

We're going to convert this to a single numeric value. Answers in days per week (in the past 7 days) will be converted (after rounding) to days in the past 30. This is a little bit of a mess, really, but we can do it.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(alcdays = as.numeric(ALCDAY5)) %>%
    mutate(alcdays = replace(alcdays, alcdays == 888, 0),
           alcdays = replace(alcdays, alcdays %in% c(777, 999), NA)) %>%
    mutate(alcdays = case_when(ALCDAY5 > 199 & ALCDAY5 < 231 ~ ALCDAY5 - 200,
                               ALCDAY5 > 100 & ALCDAY5 < 108 ~ round((ALCDAY5 - 100)*30
                               TRUE ~ alcdays))

smart_ohio_raw %>% count(ALCDAY5, alcdays)
```

```
# A tibble: 39 x 3
   ALCDAY5 alcdays     n
     <dbl>   <dbl> <int>
 1     101       4   263
 2     102       9   197
 3     103      13   142
 4     104      17    76
 5     105      21    53
 6     106      26    18
 7     107      30   114
 8     201       1   621
 9     202       2   448
10     203       3   233
# ... with 29 more rows
```

2.5.13.2 AVEDRNK2 and its cleanup to avgdrinks

AVEDRNK2, the Avg alcoholic drinks per day in past 30 variable, is the response to "One drink is equivalent to a 12-ounce beer, a 5-ounce glass of wine, or a drink with one shot of liquor. During the past 30 days, on the days when you drank, about how many drinks did you drink on the average? (A 40 ounce beer would count as 3 drinks, or a cocktail drink with 2 shots would count as 2 drinks.)"

- 1-76 = # of drinks per day
- 77 = Don't know/Not sure
- 99 = Refused
- BLANK = Not asked or Missing (always happens when ALCDAY5 = 777, 888 or 999)

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(avgdrinks = AVEDRNK2,
           avgdrinks = replace(avgdrinks, avgdrinks > 76, NA))

smart_ohio_raw %>% count(AVEDRNK2, avgdrinks) %>% tail()
```

```
# A tibble: 6 x 3
  AVEDRNK2 avgdrinks     n
     <dbl>     <dbl> <int>
1       42        42     1
2       60        60     2
3       76        76     1
4       77        NA    46
5       99        NA     5
6       NA        NA  3876
```

2.5.13.3 MAXDRNKS and its cleanup to maxdrinks

MAXDRINKS, the most drinks on a single occasion in the past 30 days variable, is the response to "During the past 30 days, what is the largest number of drinks you had on any occasion?"

- 1-76 = # of drinks
- 77 = Don't know/Not sure
- 99 = Refused
- BLANK = Not asked or Missing (always happens when ALCDAY5 = 777, 888 or 999)

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(maxdrinks = MAXDRNKS,
           maxdrinks = replace(maxdrinks, maxdrinks > 76, NA))

smart_ohio_raw %>% count(MAXDRNKS, maxdrinks) %>% tail()
```

```
# A tibble: 6 x 3
  MAXDRNKS maxdrinks     n
     <dbl>     <dbl> <int>
1       42        42     1
2       48        48     1
3       76        76     2
4       77        NA    94
5       99        NA    11
6       NA        NA  3899
```

2.5.13.4 _RFBING5 and its cleanup to binge

_RFBING5 identifies binge drinkers (males having five or more drinks on one occasion, females having four or more drinks on one occasion in the past 30 days)

The values are

- 1 = No
- 2 = Yes
- 9 = Don't Know / Refused / Missing

People who reported no alcdays are reported here as "No," so we'll adjust this into an indicator variable, and create the necessary NAs.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(binge = `_RFBING5` - 1,
           binge = replace(binge, binge > 1, NA))

smart_ohio_raw %>% count(`_RFBING5`, binge)
```

```
# A tibble: 3 x 3
  `_RFBING5` binge     n
       <dbl> <dbl> <int>
1          1     0  6035
2          2     1  1000
3          9    NA   377
```

2.5.13.5 _DRNKWEK and its cleanup to drinks_wk

_DRNKWEK provides the computed number of alcoholic drinks per week, with two implied decimal places. The code 99900 is used for "Don't know / Not sure / Refused / Missing" so we'll fix that, and also divide by 100 to get an average with a decimal point.

Note: We're also going to treat all results of 100 or more drinks per week as incorrect, and thus indicate them as missing data here.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(drinks_wk = `_DRNKWEK` / 100,
           drinks_wk = replace(drinks_wk, drinks_wk > 99, NA))

smart_ohio_raw %>% count(`_DRNKWEK`, drinks_wk) %>% tail(12)
```

```
# A tibble: 12 x 3
   `_DRNKWEK` drinks_wk     n
        <dbl>     <dbl> <int>
 1       9333      93.3     2
 2      10000      NA       1
 3      10500      NA       2
 4      11667      NA       1
 5      14000      NA       2
 6      16800      NA       2
 7      17500      NA       1
 8      18200      NA       1
 9      28000      NA       1
10      29400      NA       1
11      53200      NA       1
12      99900      NA     379
```

2.5.13.6 _RFDRHV5 and its cleanup to drink_heavy

_RFDRHV5 identifies heavy drinkers (males having 14 or more drinks per week, females having 7 or more drinks per week)

The values are

- 1 = No
- 2 = Yes
- 9 = Don't Know / Refused / Missing

People who reported no `alcdays` are reported here as "No," so we'll adjust this into an indicator variable, and create the necessary NAs.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(drink_heavy = `_RFDRHV5` - 1,
           drink_heavy = replace(drink_heavy, drink_heavy > 1, NA))

smart_ohio_raw %>% count(`_RFDRHV5`, drink_heavy)
```

```
# A tibble: 3 x 3
  `_RFDRHV5` drink_heavy     n
       <dbl>       <dbl> <int>
1          1           0  6607
2          2           1   426
3          9          NA   379
```

2.5.14 Fruits and Vegetables (8 items)

2.5.14.1 `_FRUTSU1` and its cleanup to `fruit_day`

`_FRUTSU1` provides the computed number of fruit servings consumed per day, with two implied decimal places. We'll divide by 100 to insert the decimal point.

Note: We're also going to treat all results exceeding 16 servings per day as implausible, and thus indicate them as missing data here, following some CDC procedures.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(fruit_day = `_FRUTSU1` / 100,
           fruit_day = replace(fruit_day, fruit_day > 16, NA))

smart_ohio_raw %>% count(`_FRUTSU1`, fruit_day) %>% tail()
```

```
# A tibble: 6 x 3
  `_FRUTSU1` fruit_day     n
       <dbl>     <dbl> <int>
1        913      9.13     1
2       1000     10        4
3       1400     14        1
4       3000     NA        1
5       7600     NA        1
6         NA     NA      555
```

2.5.14.2 `_VEGESU1` and its cleanup to `veg_day`

`_VEGESU1` provides the computed number of vegetable servings consumed per day, with two implied decimal places. We'll divide by 100 to insert the decimal point.

Note: We're also going to treat all results exceeding 23 servings per day as implausible, and thus indicate them as missing data here, following some CDC procedures.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(veg_day = `_VEGESU1` / 100,
           veg_day = replace(veg_day, veg_day > 23, NA))

smart_ohio_raw %>% count(`_VEGESU1`, veg_day) %>% tail()
```

```
# A tibble: 6 x 3
  `_VEGESU1` veg_day     n
       <dbl>   <dbl> <int>
1       1414    14.1     1
2       1603    16.0     1
3       1891    18.9     1
4       2167    21.7     1
5       3150    NA       1
6         NA    NA     666
```

2.5.14.3 `FTJUDA2_` and its cleanup to `eat_juice`

`FTJUDA2_` provides the servings of fruit juice consumed per day, with two implied decimal places. We'll divide by 100 to insert the decimal point.

Note: We're also going to treat all results exceeding 16 servings per day as implausible, and thus indicate them as missing data here.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(eat_juice = `FTJUDA2_` / 100,
           eat_juice = replace(eat_juice, eat_juice > 16, NA))

smart_ohio_raw %>% count(`FTJUDA2_`, eat_juice) %>% tail()
```

```
# A tibble: 6 x 3
  FTJUDA2_ eat_juice     n
     <dbl>     <dbl> <int>
1      500         5     6
2      600         6     1
3      700         7     1
4     1200        12     1
```

```
5     7500        NA     1
6       NA        NA   469
```

2.5.14.4 FRUTDA2_ and its cleanup to eat_fruit

FRUTDA2_ provides the servings of fruit consumed per day, with two implied decimal places. We'll divide by 100 to insert the decimal point.

Note: We're also going to treat all results exceeding 16 servings per day as implausible, and thus indicate them as missing data here.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(eat_fruit = `FRUTDA2_` / 100,
           eat_fruit = replace(eat_fruit, eat_fruit > 16, NA))

smart_ohio_raw %>% count(`FRUTDA2_`, eat_fruit) %>% tail()
```

```
# A tibble: 6 x 3
  FRUTDA2_ eat_fruit     n
     <dbl>     <dbl> <int>
1      700         7     5
2      800         8     3
3      900         9     1
4     1000        10     1
5     3000        NA     1
6       NA        NA   456
```

2.5.14.5 GRENDA1_ and its cleanup to eat_greenveg

GRENDA1_ provides the servings of dark green vegetables consumed per day, with two implied decimal places. We'll divide by 100 to insert the decimal point.

Note: We're also going to treat all results exceeding 16 servings per day as implausible, and thus indicate them as missing data here.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(eat_greenveg = `GRENDA1_` / 100,
           eat_greenveg = replace(eat_greenveg, eat_greenveg > 16, NA))

smart_ohio_raw %>% count(`GRENDA1_`, eat_greenveg) %>% tail()
```

```
# A tibble: 6 x 3
  GRENDA1_ eat_greenveg     n
     <dbl>        <dbl> <int>
1      700         7        4
2      786         7.86     1
3      800         8        2
```

```
4     2000        NA        1
5     3000        NA        1
6       NA        NA      447
```

2.5.14.6 FRNCHDA_ and its cleanup to eat_fries

FRNCHDA_ provides the servings of french fries consumed per day, with two implied decimal places. We'll divide by 100 to insert the decimal point.

Note: We're also going to treat all results exceeding 16 servings per day as implausible, and thus indicate them as missing data here.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(eat_fries = `FRNCHDA_` / 100,
           eat_fries = replace(eat_fries, eat_fries > 16, NA))

smart_ohio_raw %>% count(`FRNCHDA_`, eat_fries) %>% tail()
```

```
# A tibble: 6 x 3
  FRNCHDA_ eat_fries     n
     <dbl>     <dbl> <int>
1      300         3        9
2      314         3.14     1
3      400         4        3
4      500         5        1
5      700         7        1
6       NA        NA      453
```

2.5.14.7 POTADA1_ and its cleanup to eat_potato

POTADA1_ provides the servings of potatoes consumed per day, with two implied decimal places. We'll divide by 100 to insert the decimal point.

Note: We're also going to treat all results exceeding 16 servings per day as implausible, and thus indicate them as missing data here.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(eat_potato = `POTADA1_` / 100,
           eat_potato = replace(eat_potato, eat_potato > 16, NA))

smart_ohio_raw %>% count(`POTADA1_`, eat_potato) %>% tail()
```

```
# A tibble: 6 x 3
  POTADA1_ eat_potato     n
     <dbl>      <dbl> <int>
1      314       3.14     1
2      329       3.29     1
```

```
3      400       4        3
4      471       4.71     1
5      700       7        1
6       NA      NA      501
```

2.5.14.8 VEGEDA2_ and its cleanup to eat_otherveg

VEGEDA2_ provides the servings of other vegetables consumed per day, with two implied decimal places. We'll divide by 100 to insert the decimal point.

Note: We're also going to treat all results exceeding 16 servings per day as implausible, and thus indicate them as missing data here.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(eat_otherveg = `VEGEDA2_` / 100,
           eat_otherveg = replace(eat_otherveg, eat_otherveg > 16, NA))

smart_ohio_raw %>% count(`VEGEDA2_`, eat_otherveg) %>% tail()
```

```
# A tibble: 6 x 3
  VEGEDA2_ eat_otherveg     n
     <dbl>        <dbl> <int>
1      600            6     3
2      700            7    11
3      800            8     1
4     1000           10     2
5     1100           11     1
6       NA           NA   509
```

2.5.15 Exercise and Physical Activity (8 items)

2.5.15.1 _TOTINDA and its cleanup to exerany

_TOTINDA, the Exercise in Past 30 Days variable, is the response to "During the past month, other than your regular job, did you participate in any physical activities or exercises such as running, calisthenics, golf, gardening, or walking for exercise?"

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused
- BLANK = Not asked or missing

This is just like HLTHPLAN.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(exerany = `_TOTINDA`,
           exerany = replace(exerany, exerany %in% c(7, 9), NA),
           exerany = replace(exerany, exerany == 2, 0))

smart_ohio_raw %>% count(`_TOTINDA`, exerany)
```

```
# A tibble: 3 x 3
  `_TOTINDA` exerany     n
       <dbl>   <dbl> <int>
1          1       1  4828
2          2       0  2137
3          9      NA   447
```

2.5.15.2 _PACAT1 and its cleanup to activity

_PACAT1 contains physical activity categories, estimated from responses to the BRFSS. The categories are:

- 1 = Highly Active
- 2 = Active
- 3 = Insufficiently Active
- 4 = Inactive
- 9 = Don't Know / Not Sure / Refused / Missing

So we'll create a factor.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(activity = factor(`_PACAT1`),
           activity = fct_recode(activity,
                                 "Highly_Active" = "1",
                                 "Active" = "2",
                                 "Insufficiently_Active" = "3",
                                 "Inactive" = "4",
                                 NULL = "9"))

smart_ohio_raw %>% count(`_PACAT1`, activity)
```

```
# A tibble: 5 x 3
  `_PACAT1` activity                 n
      <dbl> <fct>                <int>
1         1 Highly_Active         2053
2         2 Active                1132
3         3 Insufficiently_Active 1293
4         4 Inactive              2211
5         9 <NA>                   723
```

2.5.15.3 _PAINDX1 and its cleanup to rec_aerobic

_PAINDX1 indicates whether the respondent's stated levels of physical activity meet recommendations for aerobic activity. The responses are:

- 1 = Yes
- 2 = No
- 9 = Don't know/Not sure/Refused/Missing

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(rec_aerobic = 2 - `_PAINDX1`,
           rec_aerobic = replace(rec_aerobic, rec_aerobic < 0, NA))

smart_ohio_raw %>% count(`_PAINDX1`, rec_aerobic)
```

```
# A tibble: 3 x 3
  `_PAINDX1` rec_aerobic     n
       <dbl>       <dbl> <int>
1          1           1  3228
2          2           0  3504
3          9          NA   680
```

2.5.15.4 _PASTRNG and its cleanup to rec_strength

_PASTRNG indicates whether the respondent's stated levels of physical activity meet recommendations for strength-building activity. The responses are:

- 1 = Yes
- 2 = No
- 9 = Don't know/Not sure/Refused/Missing

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(rec_strength = 2 - `_PASTRNG`,
           rec_strength = replace(rec_strength, rec_strength < 0, NA))

smart_ohio_raw %>% count(`_PASTRNG`, rec_strength)
```

```
# A tibble: 3 x 3
  `_PASTRNG` rec_strength     n
       <dbl>        <dbl> <int>
1          1            1  1852
2          2            0  5004
3          9           NA   556
```

2.5.15.5 EXRACT11 and its cleanup to exer1_type

Respondents are asked "What type of physical activity or exercise did you spend the most time doing during the past month?" and these responses are gathered into a set of 76 named categories, including an "other" category. Codes 77 (Don't Know / Not Sure) and 99 (Refused) are dropped into NA in my code below, and Code 98 ("Other type of activity") remains. Then I went through the tedious work of converting the factor levels from numbers to names, following the value labels provided by BRFSS.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(exer1_type = factor(EXRACT11),
           exer1_type = fct_recode(
               exer1_type,
               "Active Gaming Devices" = "1",
               "Aerobics video or class" = "2",
               "Backpacking" = "3",
               "Badminton" = "4",
               "Basketball" = "5",
               "Bicycling machine" = "6",
               "Bicycling" = "7",
               "Boating" = "8",
               "Bowling" = "9",
               "Boxing" = "10",
               "Calisthenics" = "11",
               "Canoeing" = "12",
               "Carpentry" = "13",
               "Dancing" = "14",
               "Elliptical machine" = "15",
               "Fishing" = "16",
               "Frisbee" = "17",
               "Gardening" = "18",
               "Golf with cart" = "19",
               "Golf without cart" = "20",
               "Handball" = "21",
               "Hiking" = "22",
               "Hockey" = "23",
               "Horseback riding" = "24",
               "Hunting large game" = "25",
               "Hunting small game" = "26",
               "Inline skating" = "27",
               "Jogging" = "28",
               "Lacrosse" = "29",
               "Mountain climbing" = "30",
               "Mowing lawn" = "31",
               "Paddleball" = "32",
```

```
                "Painting house" = "33",
                "Pilates" = "34",
                "Racquetball" = "35",
                "Raking lawn" = "36",
                "Running" = "37",
                "Rock climbing" = "38",
                "Rope skipping" = "39",
                "Rowing machine" = "40",
                "Rugby" = "41",
                "Scuba diving" = "42",
                "Skateboarding" = "43",
                "Skating" = "44",
                "Sledding" = "45",
                "Snorkeling" = "46",
                "Snow blowing" = "47",
                "Snow shoveling" = "48",
                "Snow skiing" = "49",
                "Snowshoeing" = "50",
                "Soccer" = "51",
                "Softball/Baseball" = "52",
                "Squash" = "53",
                "Stair Climbing" = "54",
                "Stream fishing" = "55",
                "Surfing" = "56",
                "Swimming" = "57",
                "Swimming in laps" = "58",
                "Table tennis" = "59",
                "Tai Chi" = "60",
                "Tennis" = "61",
                "Touch football" = "62",
                "Volleyball" = "63",
                "Walking" = "64",
                "Waterskiing" = "66",
                "Weight lifting" = "67",
                "Wrestling" = "68",
                "Yoga" = "69",
                "Child Care" = "71",
                "Farm Work" = "72",
                "Household Activities" = "73",
                "Martial Arts" = "74",
                "Upper Body Cycle" = "75",
                "Yard Work" = "76",
                "Other Activities" = "98",
                NULL = "77",
                NULL = "99")
```

```
    )
```

```
Warning: Problem with `mutate()` input `exer1_type`.
i Unknown levels in `f`: 3, 17, 21, 32, 36, 41, 42, 45, 47, 53, 55, 56, 59
i Input `exer1_type` is `fct_recode(...)`.
```

The warning generated here is caused by the fact that some of the available types of exercise were not mentioned by people in our sample. Looking at the last few results, we can see how many people fell into several categories.

```
smart_ohio_raw %>% count(EXRACT11, exer1_type) %>% tail()
```

```
# A tibble: 6 x 3
  EXRACT11 exer1_type           n
     <dbl> <fct>            <int>
1       75 Upper Body Cycle     6
2       76 Yard Work           78
3       77 <NA>                10
4       98 Other Activities   276
5       99 <NA>                 4
6       NA <NA>              2588
```

The most common activities are:

```
smart_ohio_raw %>% count(exer1_type, sort = TRUE) %>% head(10)
```

```
# A tibble: 10 x 2
   exer1_type                 n
   <fct>                  <int>
 1 Walking                 2605
 2 <NA>                    2602
 3 Running                  324
 4 Other Activities         276
 5 Gardening                242
 6 Weight lifting           189
 7 Aerobics video or class  103
 8 Bicycling machine        103
 9 Bicycling                 96
10 Golf with cart            90
```

2.5.15.6 EXRACT21 and its cleanup to exer2_type

As a follow-up, respondents are asked "What other type of physical activity gave you the next most exercise during the past month?" and these responses are also gathered into the same set of 76 named categories, including an "other" category, but now also adding a "No Other Activity" category (code 88). Codes 77 (Don't Know / Not Sure) and 99 (Refused) are dropped into NA in my code below, and

Code 98 ("Other type of activity") remains. Then I went through the tedious work of converting the factor levels from numbers to names, following the value labels provided by BRFSS. I'm sure there's a better way to do this.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(exer2_type = factor(EXRACT21),
           exer2_type = fct_recode(
               exer2_type,
               "Active Gaming Devices" = "1",
               "Aerobics video or class" = "2",
               "Backpacking" = "3",
               "Badminton" = "4",
               "Basketball" = "5",
               "Bicycling machine" = "6",
               "Bicycling" = "7",
               "Boating" = "8",
               "Bowling" = "9",
               "Boxing" = "10",
               "Calisthenics" = "11",
               "Canoeing" = "12",
               "Carpentry" = "13",
               "Dancing" = "14",
               "Elliptical machine" = "15",
               "Fishing" = "16",
               "Frisbee" = "17",
               "Gardening" = "18",
               "Golf with cart" = "19",
               "Golf without cart" = "20",
               "Handball" = "21",
               "Hiking" = "22",
               "Hockey" = "23",
               "Horseback riding" = "24",
               "Hunting large game" = "25",
               "Hunting small game" = "26",
               "Inline skating" = "27",
               "Jogging" = "28",
               "Lacrosse" = "29",
               "Mountain climbing" = "30",
               "Mowing lawn" = "31",
               "Paddleball" = "32",
               "Painting house" = "33",
               "Pilates" = "34",
               "Racquetball" = "35",
               "Raking lawn" = "36",
               "Running" = "37",
               "Rock climbing" = "38",
```

```
                "Rope skipping" = "39",
                "Rowing machine" = "40",
                "Rugby" = "41",
                "Scuba diving" = "42",
                "Skateboarding" = "43",
                "Skating" = "44",
                "Sledding" = "45",
                "Snorkeling" = "46",
                "Snow blowing" = "47",
                "Snow shoveling" = "48",
                "Snow skiing" = "49",
                "Snowshoeing" = "50",
                "Soccer" = "51",
                "Softball/Baseball" = "52",
                "Squash" = "53",
                "Stair Climbing" = "54",
                "Stream fishing" = "55",
                "Surfing" = "56",
                "Swimming" = "57",
                "Swimming in laps" = "58",
                "Table tennis" = "59",
                "Tai Chi" = "60",
                "Tennis" = "61",
                "Touch football" = "62",
                "Volleyball" = "63",
                "Walking" = "64",
                "Waterskiing" = "66",
                "Weight lifting" = "67",
                "Wrestling" = "68",
                "Yoga" = "69",
                "Child Care" = "71",
                "Farm Work" = "72",
                "Household Activities" = "73",
                "Martial Arts" = "74",
                "Upper Body Cycle" = "75",
                "Yard Work" = "76",
                "No Other Activity" = "88",
                "Other Activities" = "98",
                NULL = "77",
                NULL = "99")
    )
```

```
Warning: Problem with `mutate()` input `exer2_type`.
i Unknown levels in `f`: 3, 21, 30, 39, 41, 46, 50, 62
i Input `exer2_type` is `fct_recode(...)`.
```

```
smart_ohio_raw %>% count(EXRACT21, exer2_type) %>% tail()
```

```
# A tibble: 6 x 3
  EXRACT21 exer2_type            n
     <dbl> <fct>             <int>
1       76 Yard Work           153
2       77 <NA>                 26
3       88 No Other Activity  1854
4       98 Other Activities    246
5       99 <NA>                 19
6       NA <NA>               2627
```

The most common activity types in this group are:

```
smart_ohio_raw %>% count(exer2_type, sort = TRUE) %>% head(10)
```

```
# A tibble: 10 x 2
   exer2_type               n
   <fct>                <int>
 1 <NA>                  2672
 2 No Other Activity     1854
 3 Walking                629
 4 Weight lifting         272
 5 Other Activities       246
 6 Gardening              202
 7 Household Activities   169
 8 Yard Work              153
 9 Running                148
10 Bicycling              118
```

2.5.15.7 `_MINAC11` and its cleanup to `exer1_min`

`_MINAC11` is minutes of physical activity per week for the first activity (listed as `exer1_type` above.) Since there are only about 10,080 minutes in a typical week, we'll treat as implausible any values larger than 4200 minutes (which would indicate 70 hours per week.)

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(exer1_min = `_MINAC11`,
           exer1_min = replace(exer1_min, exer1_min > 4200, NA))

smart_ohio_raw %>% count(`_MINAC11`, exer1_min) %>% tail()
```

```
# A tibble: 6 x 3
  `_MINAC11` exer1_min     n
       <dbl>     <dbl> <int>
1       3780      3780     8
```

```
2       3959       3959     1
3       3960       3960     1
4       4193       4193     6
5      27000         NA     1
6         NA         NA  2760
```

2.5.15.8 `_MINAC21` and its cleanup to `exer2_min`

`_MINAC21` is minutes of physical activity per week for the second activity (listed as `exer2_type` above.) Again, we'll treat as implausible any values larger than 4200 minutes (which would indicate 70 hours per week.)

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(exer2_min = `_MINAC21`,
           exer2_min = replace(exer2_min, exer2_min > 4200, NA))

smart_ohio_raw %>% count(`_MINAC21`, exer2_min) %>% tail()
```

```
# A tibble: 6 x 3
  `_MINAC21` exer2_min     n
       <dbl>     <dbl> <int>
1       3360      3360     3
2       3780      3780     7
3       4193      4193     3
4       6120        NA     1
5       8400        NA     1
6         NA        NA  2770
```

2.5.16 Seatbelt Use (1 item)

2.5.16.1 `SEATBELT` and its cleanup to `seatbelt`

This question asks "How often do you use seat belts when you drive or ride in a car?" Possible responses are:

- 1 = Always
- 2 = Nearly always
- 3 = Sometimes
- 4 = Seldom
- 5 = Never
- 7 = Don't know / Not sure
- 8 = Never drive or ride in a car
- 9 = Refused

We'll treat codes 7, 8 and 9 as NA, and turn this into a factor.

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(seatbelt = fct_recode(factor(SEATBELT),
                                 "Always" = "1",
                                 "Nearly_always" = "2",
                                 "Sometimes" = "3",
                                 "Seldom" = "4",
                                 "Never" = "5",
                                 NULL = "7",
                                 NULL = "8",
                                 NULL = "9"))

smart_ohio_raw %>% count(SEATBELT, seatbelt)
```

```
# A tibble: 9 x 3
  SEATBELT seatbelt          n
     <dbl> <fct>         <int>
1        1 Always         6047
2        2 Nearly_always   409
3        3 Sometimes       191
4        4 Seldom           81
5        5 Never           148
6        7 <NA>              7
7        8 <NA>             21
8        9 <NA>              2
9       NA <NA>            506
```

2.5.17 Immunization (3 items)

2.5.17.1 `FLUSHOT6` and its cleanup to `vax_flu`

`FLUSHOT6` gives the response to "During the past 12 months, have you had either a flu shot or a flu vaccine that was sprayed in your nose?" The responses are:

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(vax_flu = 2 - FLUSHOT6,
           vax_flu = replace(vax_flu, vax_flu < 0, NA))

smart_ohio_raw %>% count(FLUSHOT6, vax_flu)
```

```
# A tibble: 5 x 3
  FLUSHOT6 vax_flu     n
```

```
     <dbl>   <dbl> <int>
1        1       1  3453
2        2       0  3410
3        7      NA    26
4        9      NA     3
5       NA      NA   520
```

2.5.17.2 PNEUVAC3 and its cleanup to vax_pneumo

PNEUVAC3 gives the response to "A pneumonia shot or pneumococcal vaccine is usually given only once or twice in a person's lifetime and is different from the flu shot. Have you ever had a pneumonia shot?" The responses are:

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(vax_pneumo = 2 - PNEUVAC3,
           vax_pneumo = replace(vax_pneumo, vax_pneumo < 0, NA))

smart_ohio_raw %>% count(PNEUVAC3, vax_pneumo)
```

```
# A tibble: 5 x 3
  PNEUVAC3 vax_pneumo     n
     <dbl>      <dbl> <int>
1        1          1  3112
2        2          0  3262
3        7         NA   509
4        9         NA     3
5       NA         NA   526
```

2.5.17.3 SHINGLE2 and its cleanup to vax_shingles

SHINGLE2 gives the response to "Have you ever had the shingles or zoster vaccine?" The responses are:

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(vax_shingles = 2 - SHINGLE2,
           vax_shingles = replace(vax_shingles, vax_shingles < 0, NA))
```

```
smart_ohio_raw %>% count(SHINGLE2, vax_shingles)
```

```
# A tibble: 4 x 3
  SHINGLE2 vax_shingles     n
     <dbl>        <dbl> <int>
1        1            1  1503
2        2            0  2979
3        7           NA    78
4       NA           NA  2852
```

2.5.18 HIV/AIDS (2 items)

2.5.18.1 `HIVTST6` and its cleanup to `hiv_test`

`HIVTST6` gives the response to "Have you ever been tested for HIV? Do not count tests you may have had as part of a blood donation. Include testing fluid from your mouth." The responses are:

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(hiv_test = 2 - HIVTST6,
           hiv_test = replace(hiv_test, hiv_test < 0, NA))

smart_ohio_raw %>% count(HIVTST6, hiv_test)
```

```
# A tibble: 5 x 3
  HIVTST6 hiv_test     n
    <dbl>    <dbl> <int>
1       1        1  2017
2       2        0  4565
3       7       NA   260
4       9       NA    14
5      NA       NA   556
```

2.5.18.2 `HIVRISK5` and its cleanup to `hiv_risk`

`HIVRISK5` gives the response to "I am going to read you a list. When I am done, please tell me if any of the situations apply to you. You do not need to tell me which one. You have injected any drug other than those prescribed for you in the past year. You have been treated for a sexually transmitted disease or STD

in the past year. You have given or received money or drugs in exchange for sex in the past year." The responses are:

- 1 = Yes
- 2 = No
- 7 = Don't know/Not sure
- 9 = Refused

```
smart_ohio_raw <- smart_ohio_raw %>%
    mutate(hiv_risk = 2 - HIVRISK5,
           hiv_risk = replace(hiv_risk, hiv_risk < 0, NA))

smart_ohio_raw %>% count(HIVRISK5, hiv_risk)
```

```
# A tibble: 5 x 3
  HIVRISK5 hiv_risk     n
     <dbl>    <dbl> <int>
1        1        1   277
2        2        0  6537
3        7       NA     2
4        9       NA    17
5       NA       NA   579
```

2.6 Imputing Age and Income as Quantitative from Thin Air

This section is purely for teaching purposes. I would never use the variables created in this section for research work.

2.6.1 `age_imp`: Imputing Age Data

I want a quantitative age variable, so I'm going to create an imputed `age_imp` value for each subject based on their `agegroup`. For each age group, I will assume that each of the ages represented by a value in that age group will be equally likely, and will draw from the relevant uniform distribution to impute age.

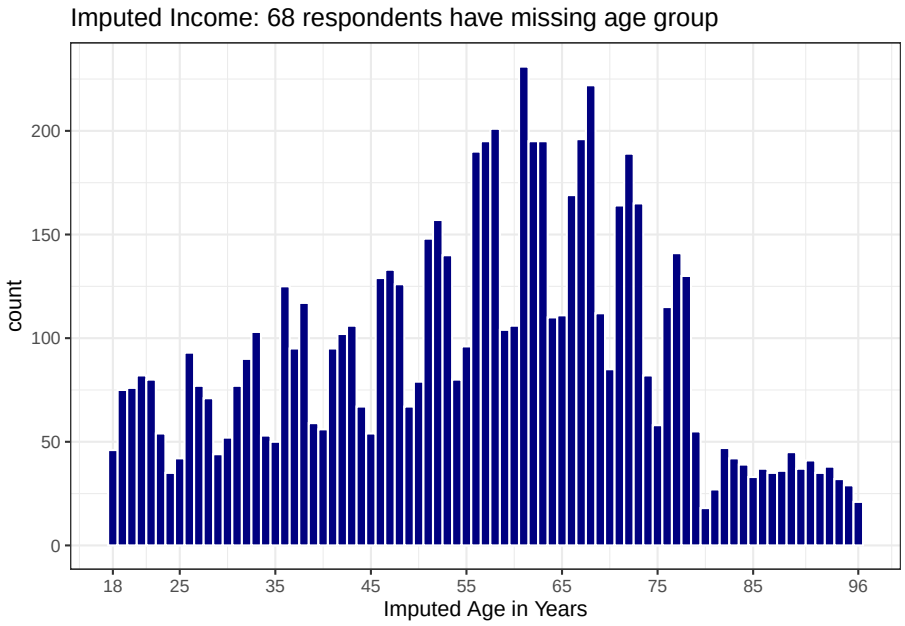
```
set.seed(2020432002)

smart_ohio_raw <- smart_ohio_raw %>%
    mutate(age_low = as.numeric(str_sub(as.character(agegroup), 1, 2))) %>%
    mutate(age_high = as.numeric(str_sub(as.character(agegroup), 4, 5))) %>%
    rowwise() %>%
    mutate(age_imp = ifelse(!is.na(agegroup),
                            round(runif(1, min = age_low, max = age_high),0),
```

```
                              NA))

smart_ohio_raw %>% count(agegroup, age_imp) #%>% tail()
```

```
# A tibble: 80 x 3
# Rowwise:
   agegroup age_imp     n
   <fct>      <dbl> <int>
 1 18-24         18    46
 2 18-24         19    75
 3 18-24         20    76
 4 18-24         21    82
 5 18-24         22    80
 6 18-24         23    54
 7 18-24         24    35
 8 25-29         25    42
 9 25-29         26    93
10 25-29         27    77
# ... with 70 more rows
```

Here is a histogram of the `age_imp` variable.

```
ggplot(smart_ohio_raw, aes(x = age_imp)) +
    geom_histogram(fill = "navy", col = "white",
                   binwidth = 1) +
    scale_x_continuous(breaks = c(18, 25, 35, 45, 55, 65, 75, 85, 96)) +
    labs(x = "Imputed Age in Years",
         title = paste0("Imputed Income: ",
                        sum(is.na(smart_ohio_raw$age_imp)),
                        " respondents have missing age group"))
```



2.6.2 `inc_imp`: Imputing Income Data

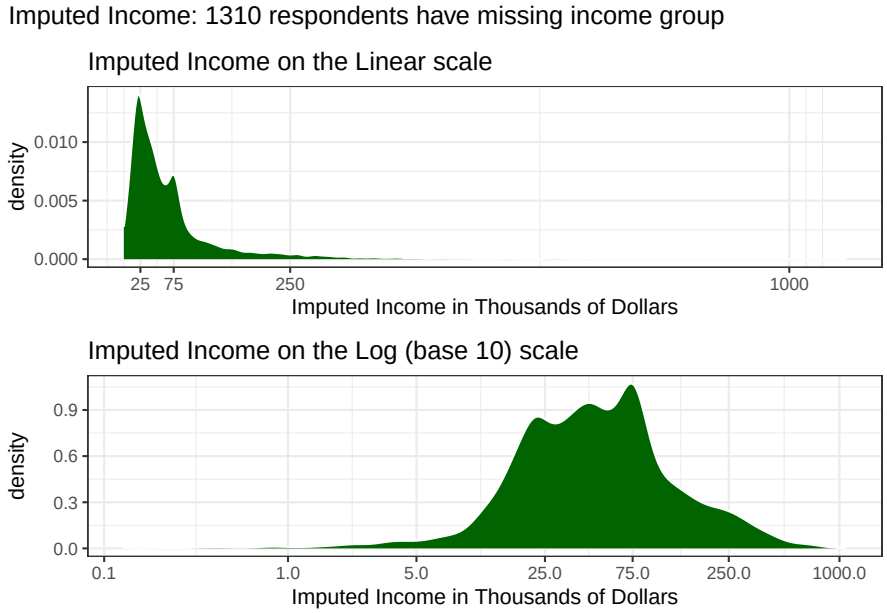
I want a quantitative income variable, so I'm going to create an imputed `inc_imp` value for each subject based on their `incomegroup`. For most income groups, I will assume that each of the incomes represented by a value in that income group will be equally likely, and will draw from the relevant uniform distribution to impute income. The exception is the highest income group, where I will impute a value drawn from a distribution that places all values at $75,000 or more, but has a substantial right skew and long tail.

```
set.seed(2020432001)

smart_ohio_raw <- smart_ohio_raw %>%
    mutate(inc_imp = case_when(
        incomegroup == "0-9K" ~ round(runif(1, min = 100, max = 9999)),
        incomegroup == "10-14K" ~ round(runif(1, min = 10000, max = 14999)),
        incomegroup == "15-19K" ~ round(runif(1, min = 15000, max = 19999)),
        incomegroup == "20-24K" ~ round(runif(1, min = 20000, max = 24999)),
        incomegroup == "25-34K" ~ round(runif(1, min = 25000, max = 34999)),
        incomegroup == "35-49K" ~ round(runif(1, min = 35000, max = 49999)),
        incomegroup == "50-74K" ~ round(runif(1, min = 50000, max = 74999)),
        incomegroup == "75K+" ~ round((rnorm(n = 1, mean = 0, sd = 300)^2) + 74999)))

smart_ohio_raw %>% count(incomegroup, inc_imp) %>% tail()
```

```
# A tibble: 6 x 3
# Rowwise:
  incomegroup inc_imp     n
  <fct>         <dbl> <int>
1 75K+         774009     1
2 75K+         798174     1
3 75K+         806161     1
4 75K+         847758     1
5 75K+        1085111     1
6 <NA>             NA  1310
```

Here are density plots of the `inc_imp` variable. The top picture shows the results on a linear scale, and the bottom shows them on a log (base 10) scale.

```
p1 <- ggplot(smart_ohio_raw, aes(x = inc_imp/1000)) +
    geom_density(fill = "darkgreen", col = "white") +
    labs(x = "Imputed Income in Thousands of Dollars",
         title = "Imputed Income on the Linear scale") +
    scale_x_continuous(breaks = c(25, 75, 250, 1000))

p2 <- ggplot(smart_ohio_raw, aes(x = inc_imp/1000)) +
    geom_density(fill = "darkgreen", col = "white") +
    labs(x = "Imputed Income in Thousands of Dollars",
         title = "Imputed Income on the Log (base 10) scale") +
    scale_x_log10(breaks = c(0.1, 1, 5, 25, 75, 250, 1000))

p1 / p2 +
    plot_annotation(title =
                        paste0("Imputed Income: ", sum(is.na(smart_ohio_raw$inc_imp)),
```



2.7 Clean Data in the State of Ohio

There are six MMSAs associated with the state of Ohio. We're going to create a `smart_ohio` that includes each of them. First, I'll ungroup the data that I created earlier, so I get a clean tibble.

```
smart_ohio_raw <- smart_ohio_raw %>% ungroup()
```

Next, I'll select the variables I want to retain (they are the ones I created, plus `SEQNO`.)

```
smart_ohio <- smart_ohio_raw %>%
    select(SEQNO, mmsa, mmsa_code, mmsa_name, mmsa_wt, completed,
           landline, hhadults,
           genhealth, physhealth, menthealth, poorhealth,
           agegroup, age_imp, race, hispanic, race_eth,
           female, marital, kids, educgroup, home_own,
           veteran, employment, incomegroup, inc_imp,
           cell_own, internet30,
           weight_kg, height_m, bmi, bmigroup,
           pregnant, deaf, blind, decide,
           diffwalk, diffdress, diffalone,
           smoke100, smoker, ecig_ever, ecigs,
           healthplan, hasdoc, costprob, t_checkup,
```

```
            bp_high, bp_meds,
            t_chol, chol_high, chol_meds,
            asthma, hx_asthma, now_asthma,
            hx_mi, hx_chd, hx_stroke, hx_skinc, hx_otherc,
            hx_copd, hx_depress, hx_kidney,
            hx_diabetes, dm_status, dm_age,
            hx_arthr, arth_lims, arth_work, arth_soc,
            joint_pain, alcdays, avgdrinks, maxdrinks,
            binge, drinks_wk, drink_heavy,
            fruit_day, veg_day, eat_juice, eat_fruit,
            eat_greenveg, eat_fries, eat_potato,
            eat_otherveg, exerany, activity, rec_aerobic,
            rec_strength, exer1_type, exer2_type,
            exer1_min, exer2_min, seatbelt,
            vax_flu, vax_pneumo, vax_shingles,
            hiv_test, hiv_risk)

saveRDS(smart_ohio, "data/smart_ohio.Rds")

write_csv(smart_ohio, "data/smart_ohio.csv")
```

The `smart_ohio` file should contain 99 variables, describing 7412 respondents.

2.8 Clean Cleveland-Elyria Data

2.8.1 Cleveland - Elyria Data

The `mmsa_name` variable is probably the simplest way for us to filter our data down to the MMSA we are interested in. Here, I'm using the `str_detect` function to identify the values of `mmsa_name` that contain the text "Cleveland."

```
smart_cle <- smart_ohio %>%
  filter(str_detect(mmsa_name, 'Cleveland'))

saveRDS(smart_cle, "data/smart_cle.Rds")
```

In the Cleveland-Elyria MSA, we have 1133 observations on the same 99 variables.

We'll build a variety of smaller subsets from these data, eventually.

Chapter 3

Dealing with Missingness: Single Imputation

3.1 Selecting Some Variables from the `smart_cle` data

```
smart_cle <- readRDS("data/smart_cle.Rds")

smart_cle1 <- smart_cle %>%
  select(SEQNO, physhealth, genhealth, bmi,
         age_imp, female, race_eth, internet30,
         smoke100, activity, drinks_wk, veg_day)
```

The `smart_cle.Rds` data file available on the Data and Code page of our website describes information on 99 variables for 1133 respondents to the BRFSS 2017, who live in the Cleveland-Elyria, OH, Metropolitan Statistical Area. The variables in the `smart_cle1.csv` file are listed below, along with the items that generate these responses.

Variable	Description
`SEQNO`	respondent identification number (all begin with 2016)
`physhealth`	Now thinking about your physical health, which includes physical illness and injury, for how many days during the past 30 days was your physical health not good?
`genhealth`	Would you say that in general, your health is ... (five categories: Excellent, Very Good, Good, Fair or Poor)
`bmi`	Body mass index, in kg/m^2
`age_imp`	Age, imputed, in years

Variable	Description
female	Sex, 1 = female, 0 = male
race_eth	Race and Ethnicity, in five categories
internet30	Have you used the internet in the past 30 days? (1 = yes, 0 = no)
smoke100	Have you smoked at least 100 cigarettes in your life? (1 = yes, 0 = no)
activity	Physical activity (Highly Active, Active, Insufficiently Active, Inactive)
drinks_wk	On average, how many drinks of alcohol do you consume in a week?
veg_day	How many servings of vegetables do you consume per day, on average?

```
str(smart_cle1)
```

```
tibble [1,133 x 12] (S3: tbl_df/tbl/data.frame)
 $ SEQNO     : num [1:1133] 2.02e+09 2.02e+09 2.02e+09 2.02e+09 2.02e+09 ...
 $ physhealth: num [1:1133] 4 0 0 0 0 2 2 0 0 0 ...
 $ genhealth : Factor w/ 5 levels "1_Excellent",..: 1 1 3 3 3 2 3 2 4 1 ...
 $ bmi       : num [1:1133] NA 23.1 26.9 26.5 24.2 ...
 $ age_imp   : num [1:1133] 51 28 37 36 88 43 23 34 58 54 ...
 $ female    : num [1:1133] 1 1 1 1 0 0 0 0 0 1 ...
 $ race_eth  : Factor w/ 5 levels "White non-Hispanic",..: 1 1 3 1 1 1 1 3 2 1 ...
 $ internet30: num [1:1133] 1 1 0 1 1 1 1 1 1 1 ...
 $ smoke100  : num [1:1133] 1 0 0 1 1 1 0 0 0 1 ...
 $ activity  : Factor w/ 4 levels "Highly_Active",..: 4 4 3 1 1 NA 1 1 1 1 ...
 $ drinks_wk : num [1:1133] 0.7 0 0 4.67 0.93 0 2 0 0 0.47 ...
 $ veg_day   : num [1:1133] NA 3 4.06 2.07 1.31 NA 1.57 0.83 0.49 1.72 ...
```

3.2 smart_cle1: Seeing our Missing Data

The naniar package provides several useful functions for summarizing missingness in our data set. Like all tidy data sets, our smart_cle1 tibble contains rows which describe observations, sometimes called *cases*, and also contains columns which describe variables.

Overall, there are 1133 cases, and 1133 observations in our smart_cle1 tibble.

- We can obtain a count of the number of missing cells in the entire tibble.

```
smart_cle1 %>% n_miss()
```

```
[1] 479
```

- We can use the `miss_var_summary` function to get a sorted table of each variable by number missing.

```
miss_var_summary(smart_cle1) %>% knitr::kable()
```

variable	n_miss	pct_miss
activity	109	9.6204766
veg_day	101	8.9143866
bmi	91	8.0317741
drinks_wk	66	5.8252427
smoke100	40	3.5304501
race_eth	26	2.2947926
physhealth	24	2.1182701
age_imp	11	0.9708738
internet30	7	0.6178288
genhealth	4	0.3530450
SEQNO	0	0.0000000
female	0	0.0000000

- Or we can use the `miss_var_table` function to tabulate the number of variables that have each observed level of missingness.

```
miss_var_table(smart_cle1)
```

```
# A tibble: 11 x 3
   n_miss_in_var n_vars pct_vars
 *         <int>  <int>    <dbl>
 1             0      2    16.7
 2             4      1     8.33
 3             7      1     8.33
 4            11      1     8.33
 5            24      1     8.33
 6            26      1     8.33
 7            40      1     8.33
 8            66      1     8.33
 9            91      1     8.33
10           101      1     8.33
11           109      1     8.33
```

- Or we can get a count for a specific variable, like `activity`:

```
smart_cle1 %>% select(activity) %>% n_miss()
```

```
[1] 109
```

- We can also use `prop_miss_case` or `pct_miss_case` to specify the proportion (or percentage) of missing observations across an entire data set, or within a specific variable.

```
prop_miss_case(smart_cle1)
```

```
[1] 0.2127096
```

```
smart_cle1 %>% select(activity) %>% pct_miss_case(.)
```

```
[1] 9.620477
```

- We can also use `prop_miss_var` or `pct_miss_var` to specify the proportion (or percentage) of variables with missing observations across an entire data set.

```
prop_miss_var(smart_cle1)
```

```
[1] 0.8333333
```

```
pct_miss_var(smart_cle1)
```

```
[1] 83.33333
```

- We use `miss_case_table` to identify the number of missing values for each of the cases (rows) in our tibble.

```
miss_case_table(smart_cle1)
```

```
# A tibble: 7 x 3
  n_miss_in_case n_cases pct_cases
*          <int>   <int>     <dbl>
1              0     892    78.7
2              1     129    11.4
3              2      51     4.50
4              3      22     1.94
5              4      21     1.85
6              5      10     0.883
7              6       8     0.706
```

- Use `miss_case_summary` to specify individual observations and count their missing values.

```
miss_case_summary(smart_cle1)
```

```
# A tibble: 1,133 x 3
    case n_miss pct_miss
   <int>  <int>    <dbl>
 1    17      6     50
 2    42      6     50
 3   254      6     50
 4   425      6     50
 5   521      6     50
 6   729      6     50
```

```
 7   757      6     50
 8  1051      6     50
 9    89      5     41.7
10    94      5     41.7
# ... with 1,123 more rows
```

The case numbers identified here are row numbers. Extract the data for case 17, for instance, with the `slice` function.

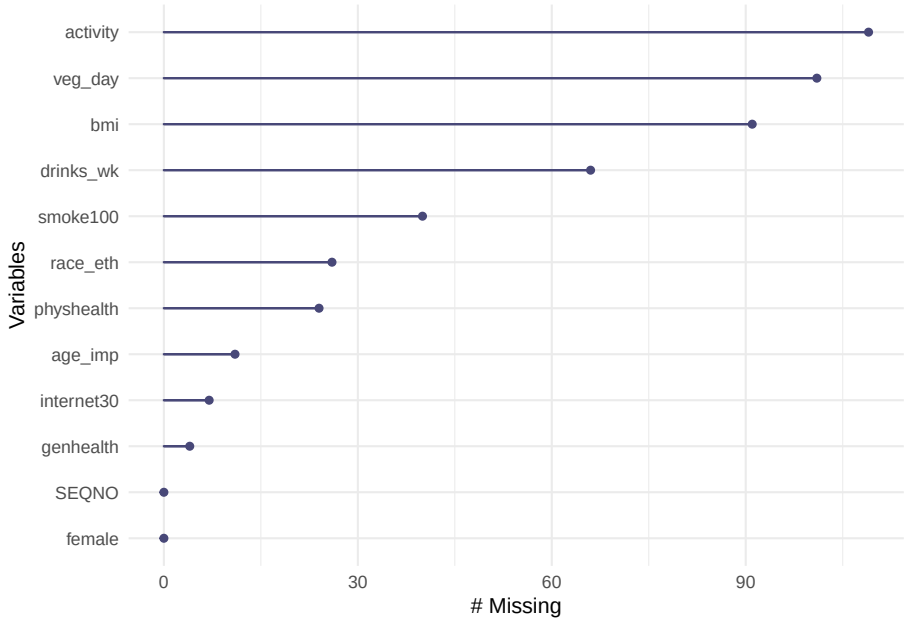
```
smart_cle1 %>% slice(17)
```

```
# A tibble: 1 x 12
   SEQNO physhealth genhealth   bmi age_imp female race_eth internet30 smoke100
   <dbl>      <dbl> <fct>     <dbl>   <dbl>  <dbl> <fct>         <dbl>    <dbl>
1 2.02e9          0 1_Excell~    NA      50      0 White n~          NA       NA
# ... with 3 more variables: activity <fct>, drinks_wk <dbl>, veg_day <dbl>
```

3.2.1 Plotting Missingness

The `gg_miss_var` function plots the number of missing observations in each variable in our data set.

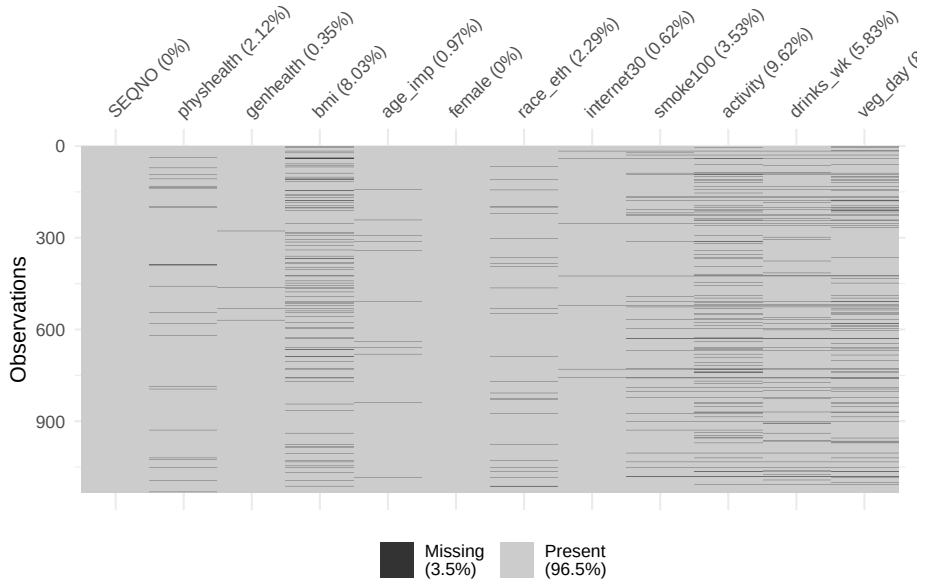
```
gg_miss_var(smart_cle1)
```



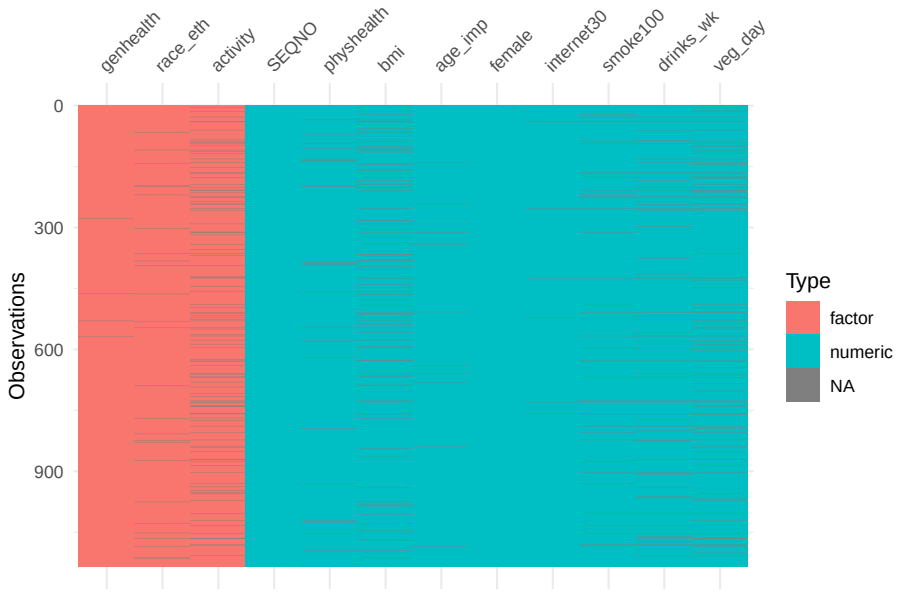
So the most commonly missing variable is `activity` which, as we've seen, has 109 missing values.

To get a general sense of the missingness in our data, we might use either the `vis_dat` or the `vis_miss` function from the `visdat` package.

```
vis_miss(smart_cle1)
```




```
vis_dat(smart_cle1)
```



3.3 Missing-data mechanisms

My source for this description of mechanisms is Chapter 25 of Gelman and Hill (2007), and that chapter is available at this link.

1. **MCAR = Missingness completely at random**. A variable is missing completely at random if the probability of missingness is the same for all units, for example, if for each subject, we decide whether to collect the `diabetes` status by rolling a die and refusing to answer if a "6" shows up. If data are missing completely at random, then throwing out cases with missing data does not bias your inferences.
2. **Missingness that depends only on observed predictors**. A more general assumption, called **missing at random** or **MAR**, is that the probability a variable is missing depends only on available information. Here, we would have to be willing to assume that the probability of nonresponse to `diabetes` depends only on the other, fully recorded variables in the data. It is often reasonable to model this process as a logistic regression, where the outcome variable equals 1 for observed cases and 0 for missing. When an outcome variable is missing at random, it is acceptable to exclude the missing cases (that is, to treat them as NA), as long as the regression controls for all the variables that affect the probability of missingness.
3. **Missingness that depends on unobserved predictors**. Missingness is no longer "at random" if it depends on information that has not been

recorded and this information also predicts the missing values. If a particular treatment causes discomfort, a patient is more likely to drop out of the study. This missingness is not at random (unless "discomfort" is measured and observed for all patients). If missingness is not at random, it must be explicitly modeled, or else you must accept some bias in your inferences.

4. **Missingness that depends on the missing value itself.** Finally, a particularly difficult situation arises when the probability of missingness depends on the (potentially missing) variable itself. For example, suppose that people with higher earnings are less likely to reveal them.

Essentially, situations 3 and 4 are referred to collectively as **non-random missingness**, and cause more trouble for us than 1 and 2.

3.4 Options for Dealing with Missingness

There are several available methods for dealing with missing data that are MCAR or MAR, but they basically boil down to:

- Complete Case (or Available Case) analyses
- Single Imputation
- Multiple Imputation

3.5 Complete Case (and Available Case) analyses

In **Complete Case** analyses, rows containing NA values are omitted from the data before analyses commence. This is the default approach for many statistical software packages, and may introduce unpredictable bias and fail to include some useful, often hard-won information.

- A complete case analysis can be appropriate when the number of missing observations is not large, and the missing pattern is either MCAR (missing completely at random) or MAR (missing at random.)
- Two problems arise with complete-case analysis:
  1. If the units with missing values differ systematically from the completely observed cases, this could bias the complete-case analysis.
  2. If many variables are included in a model, there may be very few complete cases, so that most of the data would be discarded for the sake of a straightforward analysis.
- A related approach is *available-case* analysis where different aspects of a problem are studied with different subsets of the data, perhaps identified on the basis of what is missing in them.

3.6 Single Imputation

In **single imputation** analyses, NA values are estimated/replaced *one time* with *one particular data value* for the purpose of obtaining more complete samples, at the expense of creating some potential bias in the eventual conclusions or obtaining slightly *less* accurate estimates than would be available if there were no missing values in the data.

- A single imputation can be just a replacement with the mean or median (for a quantity) or the mode (for a categorical variable.) However, such an approach, though easy to understand, underestimates variance and ignores the relationship of missing values to other variables.
- Single imputation can also be done using a variety of models to try to capture information about the NA values that are available in other variables within the data set.
- The `simputation` package can help us execute single imputations using a wide variety of techniques, within the pipe approach used by the `tidyverse`. Another approach I have used in the past is the `mice` package, which can also perform single imputations.

3.7 Multiple Imputation

Multiple imputation, where NA values are repeatedly estimated/replaced with multiple data values, for the purpose of obtaining mode complete samples *and* capturing details of the variation inherent in the fact that the data have missingness, so as to obtain *more* accurate estimates than are possible with single imputation.

- We'll postpone the discussion of multiple imputation for a while.

3.8 Approach 1: Building a Complete Case Analysis: `smart_cle1_cc`

In the 431 course, we usually dealt with missing data by restricting our analyses to respondents with complete data on all variables. Let's start by doing that here. We'll create a new tibble called `smart_cle1_cc` which includes all respondents with complete data on all of these variables.

```
smart_cle1_cc <- smart_cle1 %>%
  drop_na()

dim(smart_cle1_cc)
```

```
[1] 892  12
```

Our `smart_cle1_cc` tibble now has many fewer observations than its predecessors, but all of the variables in this complete cases tibble have no missing observations.

Data Set	Rows	Columns	Missingness?
`smart_cle`	1133	99	Quite a bit.
`smart_cle1`	1133	12	Quite a bit.
`smart_cle1_cc`	892	12	None.

3.9 Approach 2: Single Imputation to create `smart_cle1_sh`

Next, we'll create a data set which has all of the rows in the original `smart_cle1` tibble, but deals with missingness by imputing (estimating / filling in) new values for each of the missing values. To do this, we'll make heavy use of the `simputation` package in R.

The `simputation` package is designed for single imputation work. Note that we'll eventually adopt a **multiple imputation** strategy in some of our modeling work, and we'll use some specialized tools to facilitate that later.

To begin, we'll create a "shadow" in our tibble to track what we'll need to impute.

```
smart_cle1_sh <- bind_shadow(smart_cle1)

names(smart_cle1_sh)
```

```
 [1] "SEQNO"          "physhealth"     "genhealth"      "bmi"
 [5] "age_imp"        "female"         "race_eth"       "internet30"
 [9] "smoke100"       "activity"       "drinks_wk"      "veg_day"
[13] "SEQNO_NA"       "physhealth_NA" "genhealth_NA"   "bmi_NA"
[17] "age_imp_NA"     "female_NA"      "race_eth_NA"    "internet30_NA"
[21] "smoke100_NA"    "activity_NA"    "drinks_wk_NA"   "veg_day_NA"
```

Note that the `bind_shadow()` function doubles the number of variables in our tibble, specifically by creating a new variable for each that takes the value `!NA` or `NA`. For example, consider

```
smart_cle1_sh %>% count(activity, activity_NA)
```

```
# A tibble: 5 x 3
  activity              activity_NA     n
  <fct>                 <fct>       <int>
```

```
1 Highly_Active          !NA           338
2 Active                 !NA           173
3 Insufficiently_Active  !NA           201
4 Inactive               !NA           312
5 <NA>                   NA            109
```

The `activity_NA` variable takes the value `!NA` (meaning not missing) when the value of the `activity` variable is known, and takes the value `NA` for observations where the `activity` variable is missing. This background tracking will be helpful to us when we try to assess the impact of imputation on some of our summaries.

3.9.1 What Type of Missingness Do We Have?

There are three types of missingness that we might assume in any given setting: missing completely at random (MCAR), missing at random (MAR) and missing not at random (MNAR). Together, MCAR and MAR are sometimes called *ignorable* non-response, which essentially means that imputation provides a way to useful estimates. MNAR or missing NOT at random is sometimes called non-ignorable missingness, implying that even high-quality imputation may not be sufficient to provide useful information to us.

Missing Completely at Random means that the missing data points are a random subset of the data. Essentially, there is nothing that makes some data more likely to be missing than others. If the data truly match the standard for MCAR, then a complete-case analysis will be about as good as an analysis after single or multiple imputation.

Missing at Random means that there is a systematic relationship between the observed data and the missingness mechanism. Another way to say this is that the missing value is not related to the reason why it is missing, but is related to the other variables collected in the study. The implication is that the missingness can be accounted for by studying the variables with complete information. Imputation strategies can be very helpful here, incorporating what we know (or think we know) about the relationships between the results that are missing and the results that we see.

- Wikipedia provides a nice example. If men are less likely to fill in a depression survey, but this has nothing to do with their level of depression after accounting for the fact that they are male, then the missingess can be assumed MAR.
- Determining whether missingness is MAR or MNAR can be tricky. We'll spend more time discussing this later.

Missing NOT at Random means that the missing value is related to the reason why it is missing.

- Continuing the Wikipedia example, if men failed to fill in a depression survey because of their level of depression, then this would be MNAR.

- Single imputation is most helpful in the MAR situation, although it is also appropriate when we assume MCAR.
- Multiple imputation will, similarly, be more helpful in MCAR and MAR situations than when data are missing NOT at random.

It's worth noting that many people are unwilling to impute values for outcomes or key predictors in a modeling setting, but are happy to impute for less important covariates. For now, we'll assume MCAR or MAR for all of the missingness in our `smart_cle1` data, which will allow us to adopt a single imputation strategy.

3.9.2 Single imputation into `smart_cle1_sh`

Which variables in `smart_cle1_sh` contain missing data?

```
miss_var_summary(smart_cle1_sh)
```

```
# A tibble: 24 x 3
   variable   n_miss pct_miss
   <chr>       <int>    <dbl>
 1 activity      109    9.62
 2 veg_day       101    8.91
 3 bmi            91    8.03
 4 drinks_wk      66    5.83
 5 smoke100       40    3.53
 6 race_eth       26    2.29
 7 physhealth     24    2.12
 8 age_imp        11    0.971
 9 internet30      7    0.618
10 genhealth       4    0.353
# ... with 14 more rows
```

We will impute these variables using several different strategies, all supported nicely by the `simputation` package.

These include imputation methods based solely on the distribution of the complete cases of the variable being imputed.

- `impute_median`: impute the median value of all non-missing observations into the missing values for the variable
- `impute_rhd`: random "hot deck" imputation involves drawing at random from the complete cases for that variable

Also available are imputation strategies that impute predicted values from models using other variables in the data set besides the one being imputed.

- `impute_pmm`: imputation using predictive mean matching
- `impute_rlm`: imputation using robust linear models
- `impute_cart`: imputation using classification and regression trees

- `impute_knn`: imputation using k-nearest neighbors methods

3.9.3 Imputing Binary Categorical Variables

Here, we'll arbitrarily impute our 1/0 variables as follows:

- For `internet30` we'll use the `impute_rhd` approach to draw a random observation from the existing set of 1s and 0s in the complete `internet30` data.
- For `smoke100` we'll use a method called predictive mean matching (`impute_pmm`) which takes the result from a model based on the (imputed) `internet30` value and whether or not the subject is `female`, and converts it to the nearest value in the observed `smoke100` data. This is a good approach for imputing discrete variables.

These are completely arbitrary choices, for demonstration purposes.

```
set.seed(2020001)
smart_cle1_sh <- smart_cle1_sh %>%
  data.frame() %>%
    impute_rhd(.,
               internet30 ~ 1) %>%
    impute_pmm(., smoke100 ~ internet30 + female) %>%
  tbl_df()
```

```
Warning: `tbl_df()` is deprecated as of dplyr 1.0.0.
Please use `tibble::as_tibble()` instead.
This warning is displayed once every 8 hours.
Call `lifecycle::last_warnings()` to see where this warning was generated.
```

```
smart_cle1_sh %>% count(smoke100, smoke100_NA)
```

```
# A tibble: 4 x 3
  smoke100 smoke100_NA     n
     <dbl> <fct>       <int>
1        0 !NA           579
2        0 NA             21
3        1 !NA           514
4        1 NA             19
```

```
smart_cle1_sh %>% count(internet30, internet30_NA)
```

```
# A tibble: 4 x 3
  internet30 internet30_NA     n
       <dbl> <fct>         <int>
1          0 !NA             207
2          0 NA                1
3          1 !NA             919
```

```
4          1 NA                6
```

Other approaches that may be used with 1/0 variables include `impute_knn` and `impute_pmm`.

3.9.4 Imputing Quantitative Variables

We'll demonstrate a different approach for imputing each of the quantitative variables with missing observations. Again, we're making purely arbitrary decisions here about what to include in each imputation. In practical work, we'd want to be a bit more thoughtful about this.

Note that I'm choosing to use `impute_pmm` with the `physhealth` and `age_imp` variables. This is (in part) because I want my imputations to be integers, as the other observations are for those variables. `impute_rhd` would also accomplish this.

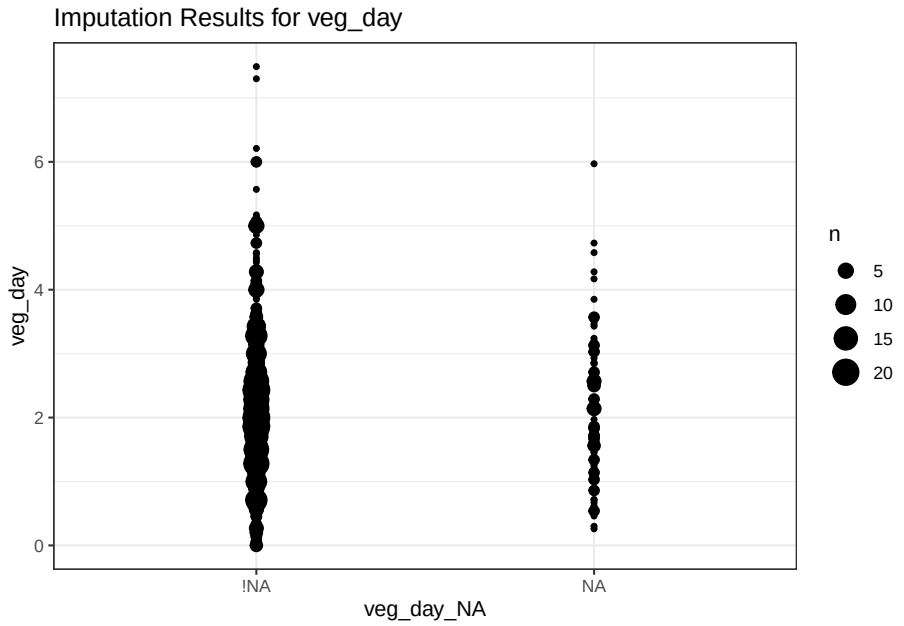
```
set.seed(2020001)
smart_cle1_sh <- smart_cle1_sh %>%
  data.frame() %>%
    impute_rhd(., veg_day ~ 1) %>%
    impute_median(., drinks_wk ~ 1) %>%
    impute_pmm(., physhealth ~
                 drinks_wk + female + smoke100) %>%
    impute_pmm(., age_imp ~ drinks_wk + physhealth) %>%
    impute_rlm(., bmi ~ physhealth + smoke100) %>%
  tbl_df()
```

3.9.5 Imputation Results

Let's plot a few of these results, so we can see what imputation has done to the distribution of these quantities.

1. `veg_day`

```
ggplot(smart_cle1_sh, aes(x = veg_day_NA, y = veg_day)) +
  geom_count() +
  labs(title = "Imputation Results for veg_day")
```



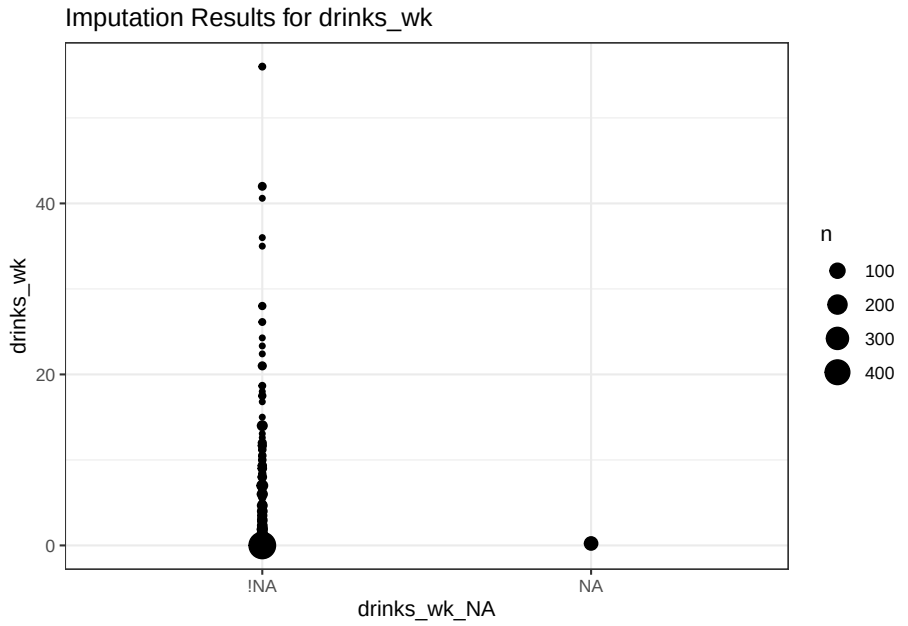

```
smart_cle1_sh %$%
  mosaic::favstats(veg_day ~ veg_day_NA)
```

```
Registered S3 method overwritten by 'mosaic':
  method                           from
  fortify.SpatialPolygonsDataFrame ggplot2
```

```
  veg_day_NA  min     Q1 median   Q3  max     mean       sd    n missing
1        !NA 0.00 1.2675   1.72 2.42 7.49 1.912548 1.038403 1032       0
2         NA 0.26 1.3400   1.86 2.72 5.97 2.085050 1.062316  101       0
```

2. `drinks_wk` for which we imputed the median value...

```
ggplot(smart_cle1_sh, aes(x = drinks_wk_NA, y = drinks_wk)) +
  geom_count() +
  labs(title = "Imputation Results for drinks_wk")
```



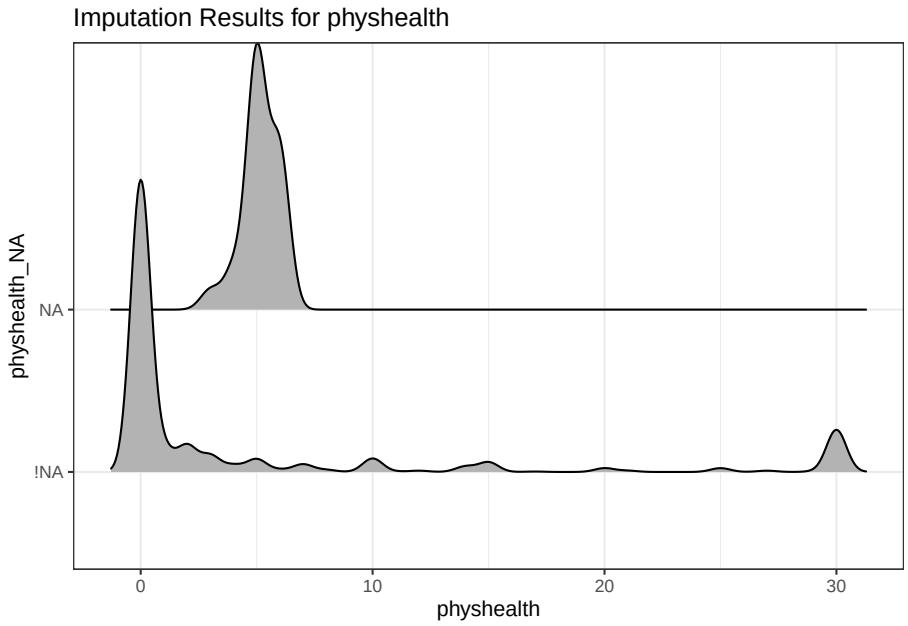

```
smart_cle1_sh %>% filter(drinks_wk_NA == "NA") %>%
  tabyl(drinks_wk)
```

```
 drinks_wk  n percent
      0.23 66       1
```

3. `physhealth`, a count between 0 and 30. . .

```
ggplot(smart_cle1_sh,
       aes(x = physhealth, y = physhealth_NA)) +
  geom_density_ridges() +
  labs(title = "Imputation Results for physhealth")
```

```
Picking joint bandwidth of 0.426
```

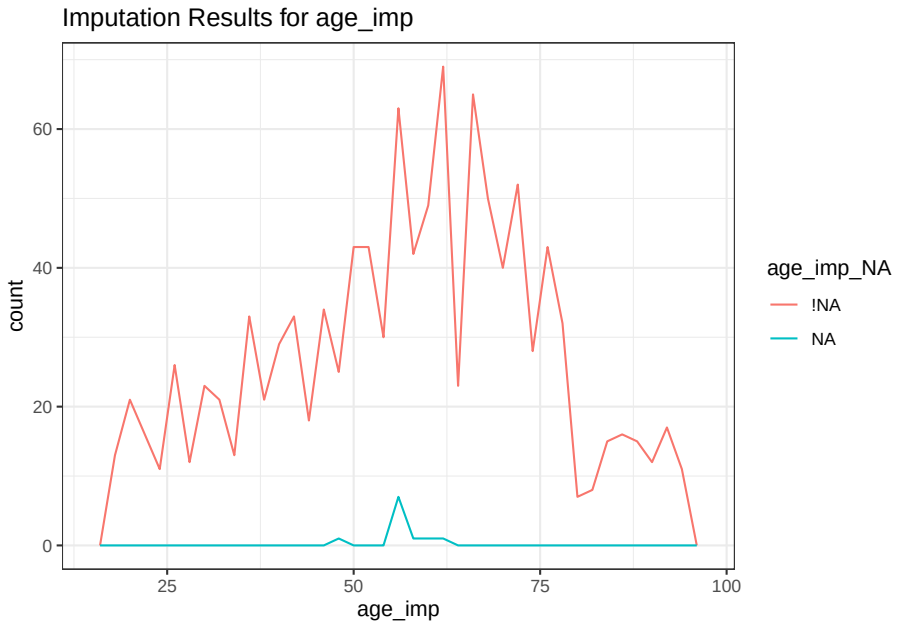



```
smart_cle1_sh %>% filter(physhealth_NA == "NA") %>%
  tabyl(physhealth)
```

```
 physhealth  n    percent
          3  1 0.04166667
          4  2 0.08333333
          5 13 0.54166667
          6  8 0.33333333
```

4. `age_imp`, in (integer) years

```
ggplot(smart_cle1_sh,
       aes(x = age_imp, color = age_imp_NA)) +
  geom_freqpoly(binwidth = 2) +
  labs(title = "Imputation Results for age_imp")
```

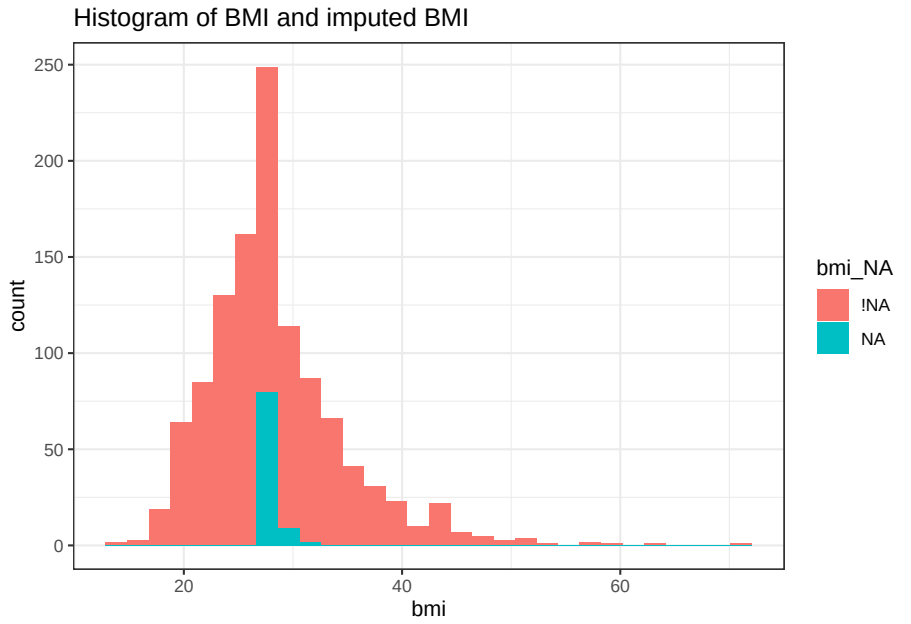



```
smart_cle1_sh %>% filter(age_imp_NA == "NA") %>%
  tabyl(age_imp)
```

```
 age_imp n    percent
      48 1 0.09090909
      57 7 0.63636364
      58 1 0.09090909
      61 1 0.09090909
      63 1 0.09090909
```

5. `bmi` or body mass index

```
ggplot(smart_cle1_sh, aes(x = bmi, fill = bmi_NA)) +
  geom_histogram(bins = 30) +
  labs(title = "Histogram of BMI and imputed BMI")
```




```
smart_cle1_sh %$% mosaic::favstats(bmi ~ bmi_NA)
```

```
  bmi_NA     min      Q1   median       Q3      max     mean        sd    n
1    !NA 13.3000 24.1100 27.30000 31.68000 70.56000 28.40947 6.6289286 1042
2     NA 27.0693 27.0693 27.50229 27.66574 30.75898 27.66057 0.8964101   91
  missing
1       0
2       0
```

3.9.6 Imputing Multi-Categorical Variables

The three multi-categorical variables we have left to impute are `activity`, `race_eth` and `genhealth`, and each is presented as a factor in R, rather than as a character variable.

We'll arbitrarily decide to impute

- `activity` and `genhealth` with a classification tree using `physhealth`, `bmi` and `smoke100`,
- and then impute `race_eth` with a random draw from the distribution of complete cases.

```
set.seed(2020001)
smart_cle1_sh <- smart_cle1_sh %>%
  data.frame() %>%
    impute_cart(., activity + genhealth ~
```

```
                  physhealth + bmi + smoke100) %>%
    impute_rhd(., race_eth ~ 1) %>%
  tbl_df()
```

Let's check our results.

```
smart_cle1_sh %>% count(activity_NA, activity)
```

```
# A tibble: 6 x 3
  activity_NA activity                  n
  <fct>       <fct>                 <int>
1 !NA         Highly_Active           338
2 !NA         Active                  173
3 !NA         Insufficiently_Active   201
4 !NA         Inactive                312
5 NA          Highly_Active            90
6 NA          Inactive                 19
```

```
smart_cle1_sh %>% count(race_eth_NA, race_eth)
```

```
# A tibble: 9 x 3
  race_eth_NA race_eth                    n
  <fct>       <fct>                   <int>
1 !NA         White non-Hispanic        805
2 !NA         Black non-Hispanic        222
3 !NA         Other race non-Hispanic    24
4 !NA         Multiracial non-Hispanic   22
5 !NA         Hispanic                   34
6 NA          White non-Hispanic         19
7 NA          Black non-Hispanic          4
8 NA          Multiracial non-Hispanic    2
9 NA          Hispanic                    1
```

```
smart_cle1_sh %>% count(genhealth_NA, genhealth)
```

```
# A tibble: 7 x 3
  genhealth_NA genhealth       n
  <fct>        <fct>       <int>
1 !NA          1_Excellent   164
2 !NA          2_VeryGood    383
3 !NA          3_Good        364
4 !NA          4_Fair        158
5 !NA          5_Poor         60
6 NA           2_VeryGood      3
7 NA           3_Good          1
```

And now, we should have no missing values in the data, at all.

```
miss_case_table(smart_cle1_sh)
```

```
# A tibble: 1 x 3
  n_miss_in_case n_cases pct_cases
*          <int>   <int>     <dbl>
1              0    1133       100
```

3.9.7 Saving the new tibbles

```
saveRDS(smart_cle1_cc, here("data", "smart_cle1_cc.Rds"))
saveRDS(smart_cle1_sh, here("data", "smart_cle1_sh.Rds"))
```

Barnett, Peggy A., Simon Roman-Golstein, Fred Ramsey, and others. 1995. "Differential Permeability and Quantitative MR Imaging of a Human Lung Carcinoma Brain Xenograft in the Nude Rat." *American Journal of Pathology* 146(2): 436–49. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1869863/.

Berkhemer, Olvert A., Puck S. S. Fransen, Debbie Buemer, and others. 2015. "A Randomized Trial of Intraarterial Treatment for Acute Ischemic Stroke." *New England Journal of Medicine* 372: 11–20. http://www.nejm.org/doi/full/10.1056/NEJMoa1411587.

Gelman, Andrew, and Jennifer Hill. 2007. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. New York: Cambridge University Press.

Ramsey, Fred L., and Daniel W. Schafer. 2002. *The Statistical Sleuth: A Course in Methods of Data Analysis*. Second Edition. Pacific Grove, CA: Duxbury.

Rosenbaum, Paul R. 2017. *Observation and Experiment: An Introduction to Causal Inference*. Cambridge, MA: Harvard University Press.

Roy, Denis, Mario Talajic, Stanley Nattel, and others. 2008. "Rhythm Control Versus Rate Control for Atrial Fibrillation and Heart Failure." *New England Journal of Medicine* 358: 2667–77. http://www.nejm.org/doi/full/10.1056/NEJMoa0708789.

Tolaney, Sara M, William T. Barry, T. Dang Chau, and others. 2015. "Adjuvant Paclitaxel and Trastuzumab for Node-Negative, Her2-Positive Breast Cancer." *New England Journal of Medicine* 372: 134–41. http://www.nejm.org/doi/full/10.1056/NEJMoa1406281.