Wrangling categorical data in R: Supplemental Appendices

Amelia McNamara*
Program in Statistical and Data Sciences, Smith College
and
Nicholas J Horton
Department of Mathematics and Statistics, Amherst College

March 19, 2018

Abstract

Supplemental appendices to accompany Wrangling categorical data in R. A preprint of this article is available at `https://peerj.com/preprints/3163/`. The full article appears in The American Statistician [McNamara and Horton, 2018].

Keywords: statistical computing; data derivation; data science; data management

*Corresponding author email: `amcnamara@smith.edu`

Supplementary Appendix A: Loading the data

Since this paper appears in a reproducible special issue, we want to make sure our data ingestion process is as reproducible as possible. We are using the General Social Survey (GSS) data, which includes many years of data (1972-2014) and many possible variables (150-800 variables, depending on the year) [Smith et al., 2015, NORC at the University of Chicago, 2016]. However, the GSS data has some idiosyncrasies. So, we are attempting good-enough practices for data ingest [Wilson et al., 2016].

The major issue related to reproducibility is the fact that the dataset is not available through an API. For SPSS and Stata users, yearly data are available for direct download on the website. For more format possibilities, users must go through an online wizard to select variables and years for the data they wish to download [NORC at the University of Chicago, 2016]. For this paper, we selected a subset of the demographic variables and the year 2014. The possible output options from the wizard are Excel (either data and metadata or metadata only), SPSS, SAS, Stata, DDI, or R script. We selected both the Excel and R formats to look at the differences.

The R format provided by the GSS is actually a Stata file and custom R script using the **foreign** package to do the translation. Here is the result of that process.

```
library(dplyr)
source('../data/GSS.r')
glimpse(GSS)

## Observations: 2,538
## Variables: 17
## $ YEAR     <int> 2014, 2014, 2014, 2014, 2014, 2014, 2014, 2014, 2014,...
## $ ID_      <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16...
## $ WRKSTAT  <int> 1, 1, 4, 2, 5, 1, 9, 1, 8, 1, 7, 8, 5, 1, 6, 2, 2, 1,...
## $ PRESTIGE <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ MARITAL  <int> 3, 1, 3, 1, 1, 1, 1, 1, 5, 1, 1, 5, 3, 1, 5, 1, 3, 5,...
## $ CHILDS   <int> 0, 0, 1, 2, 3, 1, 2, 2, 4, 3, 2, 0, 5, 2, 0, 3, 3, 0,...
## $ AGE      <int> 53, 26, 59, 56, 74, 56, 63, 34, 37, 30, 43, 56, 69, 4...
## $ EDUC     <int> 16, 16, 13, 16, 17, 17, 12, 17, 10, 15, 5, 11, 8, 11,...
## $ SEX      <int> 1, 2, 1, 2, 2, 2, 1, 1, 2, 2, 2, 1, 1, 2, 2, 1, 2, 1,...
```

```
## $ RACE     <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 1, 1, 1, 1, 1, 3, 1,...
## $ INCOM16  <int> 2, 3, 2, 2, 4, 4, 2, 3, 3, 1, 1, 2, 2, 2, 2, 3, 2, 3,...
## $ INCOME   <int> 12, 12, 12, 12, 13, 12, 13, 12, 10, 12, 9, 9, 10, 11,...
## $ RINCOME  <int> 12, 12, 0, 9, 0, 12, 13, 12, 0, 12, 0, 0, 0, 11, 12, ...
## $ INCOME72 <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ PARTYID  <int> 5, 5, 6, 5, 3, 6, 6, 8, 3, 3, 3, 3, 3, 1, 3, 6, 1, 3,...
## $ FINRELA  <int> 4, 4, 2, 4, 3, 4, 9, 3, 2, 3, 8, 5, 1, 1, 3, 3, 2, 3,...
## $ SEXORNT  <int> 3, 3, 3, 3, 3, 9, 0, 0, 3, 3, 3, 3, 3, 0, 3, 3, 0, 0,...
```

Obviously, the result is less than ideal, lacking many of the qualities for sharing data for collaboaration [Ellis and Leek, 2018]. All of the factor variables are encoded as integers, but their level labels have been lost. We have to look at a codebook to determine if `SEX == 1` indicates male or female. We would rather preserve the integrated level labels. In order to do this, our best option is to use the Excel file and use the **readxl** package to load it.

```
library(readxl)
GSS <- read_excel("../data/GSS.xls")
glimpse(GSS)

## Observations: 2,540
## Variables: 17
## $ Gss year for this respondent                       <dbl> 2014, 2014...
## $ Respondent id number                               <dbl> 1, 2, 3, 4...
## $ Labor force status                                 <chr> "Working f...
## $ Rs occupational prestige score  (1970)             <dbl> 0, 0, 0, 0...
## $ Marital status                                     <chr> "Divorced"...
## $ Number of children                                 <dbl> 0, 0, 1, 2...
## $ Age of respondent                                  <chr> "53.000000...
## $ Highest year of school completed                   <dbl> 16, 16, 13...
## $ Respondents sex                                    <chr> "Male", "F...
## $ Race of respondent                                 <chr> "White", "...
## $ Rs family income when 16 yrs old                   <chr> "Below ave...
## $ Total family income                                <chr> "$25000 or...
## $ Respondents income                                 <chr> "$25000 or...
## $ Total family income                                <chr> "Not appli...
## $ Political party affiliation                        <chr> "Not str r...
```

```
## $ Opinion of family income                              <chr> "Above ave...
## $ Sexual orientation                                    <chr> "Heterosex...
```

This is a little better. Now the character strings are preserved. But, the data is not yet usable in an analysis. One problem is some of the variable names include spaces, so they are hard to use. Also, one variable name is repeated, perhaps because of an error in the data wizard. To fix these issues, the variables must be renamed the variables such that all variables have unique names without spaces.

```
names(GSS) <- make.names(names(GSS), unique=TRUE)
names(GSS)

##  [1] "Gss.year.for.this.respondent......................"
##  [2] "Respondent.id.number"
##  [3] "Labor.force.status"
##  [4] "Rs.occupational.prestige.score...1970."
##  [5] "Marital.status"
##  [6] "Number.of.children"
##  [7] "Age.of.respondent"
##  [8] "Highest.year.of.school.completed"
##  [9] "Respondents.sex"
## [10] "Race.of.respondent"
## [11] "Rs.family.income.when.16.yrs.old"
## [12] "Total.family.income"
## [13] "Respondents.income"
## [14] "Total.family.income.1"
## [15] "Political.party.affiliation"
## [16] "Opinion.of.family.income"
## [17] "Sexual.orientation"
```

These names are an improvement, but now some are full of periods. For ease of coding, more human readable names are preferable. As with all the tasks in this paper, there is a fragile way to do this in **base** R, but we'll use the more robust `rename()` function from the **dplyr** package. `rename()`

```
library(dplyr)
GSS <- GSS %>%
  rename(Year = Gss.year.for.this.respondent.....................,
         ID = Respondent.id.number,
         LaborStatus = Labor.force.status,
         OccupationalPrestigeScore = Rs.occupational.prestige.score...1970.,
         MaritalStatus = Marital.status,
         NumChildren = Number.of.children,
         Age = Age.of.respondent,
         Sex = Respondents.sex,
         HighestSchoolCompleted = Highest.year.of.school.completed,
         Race = Race.of.respondent,
         ChildhoodFamilyIncome = Rs.family.income.when.16.yrs.old,
         TotalFamilyIncome = Total.family.income,
         RespondentIncome = Respondents.income,
         PoliticalParty = Political.party.affiliation,
         OpinionOfIncome = Opinion.of.family.income,
         SexualOrientation = Sexual.orientation)
names(GSS)

##  [1] "Year"                      "ID"
##  [3] "LaborStatus"               "OccupationalPrestigeScore"
##  [5] "MaritalStatus"             "NumChildren"
##  [7] "Age"                       "HighestSchoolCompleted"
##  [9] "Sex"                       "Race"
## [11] "ChildhoodFamilyIncome"     "TotalFamilyIncome"
## [13] "RespondentIncome"          "Total.family.income.1"
## [15] "PoliticalParty"            "OpinionOfIncome"
## [17] "SexualOrientation"

GSS <- GSS %>%
  select(-Total.family.income.1)
```

With the data loaded and the names adjusted, the cleaned data can be written to a new file for use in the body of the paper.

```
library(readr)
write_csv(GSS, path="../data/GSScleaned.csv")
```

A version of this file is used as our motivating example.

Supplementary Appendix B: Closing exercise

We have included the following as a possible supplementary exercise.

Subjects in the HELP study were also categorized into categories of primary and secondary drug and alcohol involvement, as displayed in the following table.

```
library(mosaic)
library(mosaicData)
HELPbase <- HELPfull %>%
  filter(TIME == 0)
tally( ~ PRIM_SUB + SECD_SUB, data=HELPbase)

##          SECD_SUB
## PRIM_SUB  0  1  2  3  4  5  6  7  8
##        1 99  0 57 13  1  3 11  0  1
##        2 51 84  0  6  0  0 15  0  0
##        3 57 28 29  0  0  6  5  1  2
##        6  0  1  0  0  0  0  0  0  0
```

The following coding of substance use involvement was used in the study.

value	description
0	None
1	Alcohol
2	Cocaine
3	Heroin
4	Barbituates
5	Benzos
6	Marijuana
7	Methadone
8	Opiates

Create a new variable called `primsub` combining the primary and secondary substances into a categorical variable with values corresponding to primary and secondary substances of the form: `alcohol only`,
`cocaine only`, `heroin only`,
`alcohol-cocaine`, `cocaine-alcohol`, or `other`. Code any group with fewer than 5 entries as `alcohol-other`, `cocaine-other`, or `heroin-other`. If `PRIM\_SUB == 6` make the `primsub` variable missing.

How many subjects are there in the `alcohol-none` group? How many subjects are there in the `alcohol-other` group? What are the three most common groups?

SOLUTION:

```
HELPbase <- HELPbase %>%
  mutate(
    primary= recode(PRIM_SUB,
      `1`="alcohol",
      `2`="cocaine",
      `3`="heroin",
      `4`="barbituates",
      `5`="benzos",
      `6`="marijuana",
      `7`="methadone",
      `8`="opiates"),
    second=recode(SECD_SUB,
      `0`="none",
      `1`="alcohol",
      `2`="cocaine",
      `3`="heroin",
      `4`="barbituates",
      `5`="benzos",
      `6`="marijuana",
      `7`="methadone",
      `8`="opiates"),
    title=paste0(primary, "-", second)
)
```

```
tally(~ primary, data=HELPbase)
```

```
##
##   alcohol   cocaine    heroin marijuana
##       185       156       128         1
```

```
tally(~ second, data=HELPbase)
```

```
##
##     alcohol barbituates      benzos     cocaine      heroin   marijuana
##         113           1           9          86          19          31
##   methadone        none     opiates
##           1         207           3
```

```
counts <- HELPbase %>%
  group_by(primary, second) %>%
  summarise(observed=n())

merged <- left_join(HELPbase, counts, by=c("primary", "second"))
```

```
merged <- merged %>%
  mutate(
    title =
      case_when(
        observed < 5 & primary == "alcohol" ~ "alcohol-other",
        observed < 5 & primary == "cocaine" ~ "cocaine-other",
        observed < 5 & primary == "heroin" ~ "heroin-other",
        TRUE ~ title),
    title = ifelse(primary == "marijuana", NA, title))

tally(~ title + observed, data=merged)
```

```
##                     observed
## title                 1  2  3  5  6 11 13 15 28 29 51 57 84 99
##   alcohol-cocaine     0  0  0  0  0  0  0  0  0  0  0 57  0  0
##   alcohol-heroin      0  0  0  0  0  0 13  0  0  0  0  0  0  0
##   alcohol-marijuana   0  0  0  0  0 11  0  0  0  0  0  0  0  0
```

```
##   alcohol-none       0  0  0  0  0  0  0  0  0  0  0  0  0 99
##   alcohol-other      2  0  3  0  0  0  0  0  0  0  0  0  0  0
##   cocaine-alcohol    0  0  0  0  0  0  0  0  0  0  0  0 84  0
##   cocaine-heroin     0  0  0  0  6  0  0  0  0  0  0  0  0  0
##   cocaine-marijuana  0  0  0  0  0  0  0 15  0  0  0  0  0  0
##   cocaine-none       0  0  0  0  0  0  0  0  0  0 51  0  0  0
##   heroin-alcohol     0  0  0  0  0  0  0  0 28  0  0  0  0  0
##   heroin-benzos      0  0  0  0  6  0  0  0  0  0  0  0  0  0
##   heroin-cocaine     0  0  0  0  0  0  0  0  0 29  0  0  0  0
##   heroin-marijuana   0  0  0  5  0  0  0  0  0  0  0  0  0  0
##   heroin-none        0  0  0  0  0  0  0  0  0  0  0 57  0  0
##   heroin-other       1  2  0  0  0  0  0  0  0  0  0  0  0  0
##   <NA>               1  0  0  0  0  0  0  0  0  0  0  0  0  0
```

```
tally(~ title == "alcohol-none", data=merged)

##
##  TRUE FALSE  <NA>
##    99   370     1

tally(~ title == "alcohol-other", data=merged)

##
##  TRUE FALSE  <NA>
##     5   464     1

sort(tally(~ title, data=merged), decreasing=TRUE)[1:3]

##
##    alcohol-none cocaine-alcohol alcohol-cocaine
##              99              84              57
```

References

Shannon E Ellis and Jeffrey T Leek. How to share data for collaboration. *The American Statistician*, 71(5), 2018.

Amelia McNamara and Nicholas J Horton. Wrangling categorical data in R. *The American Statistician*, 71(5), 2018.

NORC at the University of Chicago. GSS data explorer. `https://gssdataexplorer.norc.org/`, 2016.

Tom W Smith, Peter Mardsen, Michael Hout, and Jibum Kim. General social surveys, 1972-2014 [machine-readable data file], 2015.

Greg Wilson, Jennifer Bryan, Karen Cranston, Justin Kitzes, Lex Nederbragt, and Tracy K Teal. Good enough practices for scientific computing. `https://swcarpentry.github.io/good-enough-practices-in-scientific-computing/`, 2016.