TEND_Data_Prep

Andrew Crena

2023-05-28

R Markdown

Set WD

```
library(here)
```

```
## here() starts at C:/Users/adcre/OneDrive/Documents/Desktop_RStudio
```

```
here::i_am("TEND_Group_Testing.Rmd") # tells you full file path script's current location
```

```
## here() starts at C:/Users/adcre/OneDrive/Documents/Desktop_RStudio
```

```
here() # returns file path of where the script is current saved
```

```
## [1] "C:/Users/adcre/OneDrive/Documents/Desktop_RStudio"
```

```
setwd(here()) # sets the working directory to be wherever your source file or Rmd is
getwd()
```

```
## [1] "C:/Users/adcre/OneDrive/Documents/Desktop_RStudio"
```

Libraries/Packages

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(readr)
library(here)
```

Data Preparation for analysis

```
# This markdown was created when I realized how many important concepts are covered in
# just the preparation for the 'fun stuff' the analysis! Throughout this code chunk and
# possibly more, you will find various methods of data cleaning or preparation. I would
# like to have many of these skills as a proficiency in order to be of use to my team!
# Brain_Data <- read.csv("Brain_Data.csv", header=T, sep=",", na.strings=c("NA", "888", "999"))
Brain_Data <- read.csv("Brain_Data.csv", header=T, sep=",", na.strings=c("NA", "888", "999"))
Behavioral_Data<-read.csv("Bx_Data.csv", header=T, sep=",", na.strings=c("NA", "888", "999"))

# Merge the data sets
DATA <- merge(Brain_Data, Behavioral_Data, "ID", all=T)
View(DATA)

# The most common form of data is factors, numbers, strings and characters. They are all
# understood by R in different ways and certain functions need data to be in specific types
# to be understood correctly. Below are examples of rewriting the variable "ID" as a
# factor. And also creating a new variables Age as the numeric version of "Age.at.V1".
# To call a variable within a data set you say [name of dataset]$[name of varible],
# if the name of the variable does not exist, it will CREATE the variable in that data
# set. So in this case, "Age" did not exist, it was just Age.at.V1 and Age.at.V2 in our
# sheet, so instead we created a number version of "Age.at.V1" now called "Age"
DATA$ID<- as.factor(DATA$ID)
DATA$Age<- as.numeric(DATA$Age_at_Bx)

# It will be a very good habit to not only check what form your data or columns are
# represented as, it will be very useful to learn the functions that change your code
# into a more effective form.
#
# attach() and detach()
# If you have several variables you are manipulating and don't want to call the data
# frame they belong to each time ([data frame]$[varibale you want]) you can "attach"
# the data set so that R assumes all variables you call until you "detach" it will be
# in the data frame you are looking at. Below is an example of scoring a measurement
# of several components. Instead of having to write "DATA$RADS_DM" etc. for each subscale
# we are adding, we can just write the variable name. THIS WAS COPY+PASTED FROM
# JOHANNA's Rmd
attach(DATA)
DATA$RADS_total_rescore<- RADS_DM + RADS_AN + RADS_NS + RADS_SC
detach(DATA)
View(DATA)
# This would be useful in a situation where you are constantly having to call a column
# that is nested within a variable, or multiple. This could get confusing when you have
# performed significant exploration and have many variables in your environment

# attach() and detach()
# If you have several variables you are manipulating and don't want to call the data
# frame they belong to each time, you can use attach() and detach() to tell the software
```

```
# to call only to the data frame you specify. This prevents you from having to type out
# the long code that calls to a specific column nested in a data set
attach(DATA)
DATA$RADS_total_rescore<- RADS_DM + RADS_AN + RADS_NS + RADS_SC
detach(DATA)

# recode() COPY+PASTED FROM JOHANNA'S Rmd
# #Here is how you can easily recode something using the "recode" function, the first
# value will be what the data currently is, and the second value is what you now want
# it to be. For example, we want to round up all of the scores in one column or variable,
# 'Puberty_rounded'.
DATA$Puberty_rounded <- recode(DATA$Puberty, '1.5' = 2.0, '2.5' = 3.0, '3.5' = 4.0,
                               '4.5' = 5.0)
# Dplyr has some very useful functions as well, but this is useful to know

# creating new variable and column
# This is how you can add a new variable with NA values
DATA$attempt_reason<- NA

# assigning NA to conditions in a column
# By using the variable assignment operator '<-', we can assign NA avlues to a specified
# column in a df, and add conditions so that only subjects with scores below 21, for
# example, get reassigned to NA
DATA$Bx_to_Scan_Days[DATA$Bx_to_Scan_Days > 21] <- NA

# rename()
# If you wanted to rename variables or names of columns, you can change them with
# rename(), even doing multiple rows simultaneously
DATA_Final<- rename(DATA,"Hx_Attempt" = "Hx.Attempt", "Age_First.Attempt" = "Age.First.Attempt","Number

# Let's now save our new data sheet using the writexl library
library(writexl)
write_xlsx(DATA_Final, "Example_DATA_Export.xlsx")
```

Quick Check!

Johanna gave a very useful tip for quickly checking your data (certain columns, variables) before running anlyses or continuing with your exploratory analysis. Using View() and data.frame() in conjuction with one another gives you a quick glimpse at a quickly-made df of your specified columns.

```
# This is a great way of double-checking that your preparation hasn't changed the data.
# Think of it as a way of looking twice before turning!
View(data.frame(DATA_Final$ID, DATA_Final$CDRSR_total, DATA_Final$Sex, DATA_Final$Puberty))
```