Assignment 1 - Data Cleaning - Instructions

The Eyes of Kasparov

2020-09-03

# Assignment 1, Language development in Autism Spectrum Disorder (ASD) - Brushing up your code skills

## 2. Now let’s take dirty dirty data sets and make them into a tidy one

If you’re not in a project in Rstudio, make sure to set your working directory here. If you created an RStudio project, then your working directory (the directory with your data and code for these assignments) is the project directory.

pacman::p\_load(tidyverse,janitor)

Load the three data sets, after downloading them from dropbox and saving them in your working directory: \* Demographic data for the participants: <https://www.dropbox.com/s/lol8f5m4pgrpmto/demo_train.csv?dl=0> \* Length of utterance data: <https://www.dropbox.com/s/usyauqm37a76of6/LU_train.csv?dl=0> \* Word data: <https://www.dropbox.com/s/8ng1civpl2aux58/token_train.csv?dl=0>

Explore the 3 datasets (e.g. visualize them, summarize them, etc.). You will see that the data is messy, since the psychologist collected the demographic data, the linguist analyzed the length of utterance in May 2014 and the fumbling jack-of-all-trades analyzed the words several months later. In particular: - the same variables might have different names (e.g. participant and visit identifiers) - the same variables might report the values in different ways (e.g. participant and visit IDs) Welcome to real world of messy data :-)

### EDA of data

Before being able to combine the data sets we need to make sure the relevant variables have the same names and the same kind of values.

So:

2a. Identify which variable names do not match (that is are spelled differently) and find a way to transform variable names. Pay particular attention to the variables indicating participant and visit.

### Fixing column names

## From visual inspection we see that SUBJ = Child.ID and VISIT is inconsistently capitalized ##  
  
# fixing demodata  
demodata <- demodata %>%   
 rename(ID = Child.ID,   
 VISIT = Visit)  
  
# fixing loudata  
loudata <- loudata %>%   
 rename(ID = SUBJ)  
  
# fixing worddata  
worddata <- worddata %>%   
 rename(ID = SUBJ)

2b. Find a way to homogeneize the way “visit” is reported (visit1 vs. 1).

# We'll use parse\_number for super simplicity  
demodata <- demodata %>%   
 mutate(VISIT = as\_factor(VISIT))  
  
loudata <- loudata %>%   
 mutate(VISIT = as\_factor(parse\_number(VISIT)))  
  
worddata <- worddata %>%   
 mutate(VISIT = as\_factor(parse\_number(VISIT)))

2c. We also need to make a small adjustment to the content of the Child.ID coloumn in the demographic data. Within this column, names that are not abbreviations do not end with “.” (i.e. Adam), which is the case in the other two data sets (i.e. Adam.). If The content of the two variables isn’t identical the rows will not be merged. A neat way to solve the problem is simply to remove all “.” in all datasets.

Tip: stringr is helpful again. Look up str\_replace\_all Tip: You can either have one line of code for each child name that is to be changed (easier, more typing) or specify the pattern that you want to match (more complicated: look up “regular expressions”, but less typing)

demodata <- demodata %>%   
 mutate(ID = str\_replace\_all(ID, "\\.", ""))  
  
loudata <- loudata %>%   
 mutate(ID = str\_replace\_all(ID, "\\.", ""))  
  
worddata <- worddata %>%   
 mutate(ID = str\_replace\_all(ID, "\\.", ""))

2d. Now that the nitty gritty details of the different data sets are fixed, we want to make a subset of each data set only containig the variables that we wish to use in the final data set. For this we use the tidyverse package dplyr, which contains the function select().

Feel free to rename the variables into something you can remember (i.e. nonVerbalIQ, verbalIQ)

# We are joining the tibbles on ID and VISIT  
all\_data <- demodata %>%   
 left\_join(loudata, by = c("ID", "VISIT")) %>%   
 left\_join(worddata, by = c("ID", "VISIT"))  
  
# Selecting relevant columns  
all\_data <- all\_data %>% select(ID, VISIT, Diagnosis, Ethnicity, Gender, Age, ADOS,   
 MullenRaw, ExpressiveLangRaw, Socialization, MOT\_MLU,   
 CHI\_MLU, types\_MOT, types\_CHI, tokens\_MOT, tokens\_CHI)

2e. Finally we are ready to merge all the data sets into just one.

# A bit of a funky approach   
all\_data <- all\_data %>%   
 mutate(ADOS = if\_else(VISIT == "1", ADOS, NULL),   
 MullenRaw = if\_else(VISIT == 1, MullenRaw, NULL),   
 ExpressiveLangRaw = if\_else(VISIT == "1", ExpressiveLangRaw, NULL),   
 Socialization = if\_else(VISIT == "1", Socialization, NULL)) %>%  
 arrange(ID, VISIT) %>%   
 fill(ADOS, MullenRaw, ExpressiveLangRaw, Socialization)

2g. Final touches

Now we want to \* anonymize our participants (they are real children!). \* make sure the variables have sensible values. E.g. right now gender is marked 1 and 2, but in two weeks you will not be able to remember, which gender were connected to which number, so change the values from 1 and 2 to F and M in the gender variable. For the same reason, you should also change the values of Diagnosis from A and B to ASD (autism spectrum disorder) and TD (typically developing). Tip: Try taking a look at ifelse(), or google “how to rename levels in R”. \* Save the data set using into a csv file. Hint: look into write.csv()

library(digest)  
# Vectorize the hash-function so it doesn't apply to the whole vector  
vdigest <- Vectorize(digest)  
  
  
final\_data <- all\_data %>%   
 # Anonymize our participants using hash  
 mutate(ID = vdigest(ID, algo = "sha1")) %>%   
 # sensible values   
 mutate(Gender = if\_else(Gender == "1", "F", "M"),   
 Diagnosis = if\_else(Diagnosis == "A", "ASD", "TD"))   
  
  
# Write to file  
final\_data %>% write\_csv("final\_data.csv")

1. BONUS QUESTIONS The aim of this last section is to make sure you are fully fluent in the tidyverse. Here’s the link to a very helpful book, which explains each function: <http://r4ds.had.co.nz/index.html>
2. USING FILTER List all kids who:
3. have a mean length of utterance (across all visits) of more than 2.7 morphemes.
4. have a mean length of utterance of less than 1.5 morphemes at the first visit
5. have not completed all trials. Tip: Use pipes to solve this

# 1  
final\_data %>%   
 group\_by(ID) %>%   
 summarise(MLU\_ACROSS = mean(CHI\_MLU)) %>%   
 filter(MLU\_ACROSS > 2.7) %>%   
 head(5)

## `summarise()` ungrouping output (override with `.groups` argument)

## # A tibble: 5 x 2  
## ID MLU\_ACROSS  
## <chr> <dbl>  
## 1 110f030a7c56beb8dc7b8e121c34d468c5aac2f6 2.94  
## 2 2f1925df73d7081fd42c59ffb59cacb8d0f1b940 3.02  
## 3 55ecd46b6416043d3e9387e5dd6442f83d478d54 2.75  
## 4 5fd52431cc2061de1eee06801a942074b7687e1b 2.75  
## 5 63105fa205f0091ea128b6a4aedcb0a06fb0dce9 2.73

# 2  
final\_data %>%   
 filter(VISIT == "1") %>%   
 filter(CHI\_MLU < 1.5) %>%   
 select(ID, CHI\_MLU) %>%   
 head(5)

## # A tibble: 5 x 2  
## ID CHI\_MLU  
## <chr> <dbl>  
## 1 7edc6400c38033887ec1df1fad3a8fcb53c9706c 1.25  
## 2 087fb09b9b1ade8e554db444cadee9546d2a41dd 1.39  
## 3 dbd994484e216f7b00ab0b3b083207e0b3e2aea9 1   
## 4 f31762e7fb06c98966da324dddbb4995f600841c 1.26  
## 5 82e6d2006453e89e75df5b743be5158bc077d105 1.04

# 3   
final\_data %>%   
 group\_by(ID) %>%   
 # count number of trials  
 summarise(num\_trials = n()) %>%   
 filter(num\_trials < 6) %>%   
 head(5)

## `summarise()` ungrouping output (override with `.groups` argument)

## # A tibble: 5 x 2  
## ID num\_trials  
## <chr> <int>  
## 1 0052610d778a89c392e4bc2592d53e67cb402cec 2  
## 2 45a4258704885c8abc7da62a85e46143d1d54212 1  
## 3 9ae331c4acb585e03313e87bd43a75a2d914c631 2  
## 4 a38cd28b216de2b3e030c3f34c434fc9ef491fc7 2  
## 5 bd5af99f30a1de407cc23b2aae20b761624feeac 5

USING ARRANGE

1. Sort kids to find the kid who produced the most words on the 6th visit
2. Sort kids to find the kid who produced the least amount of words on the 1st visit.

# 1   
final\_data %>%   
 filter(VISIT == "6") %>%   
 arrange(desc(tokens\_CHI)) %>%   
 select(ID, tokens\_CHI) %>%   
 head(1)

## # A tibble: 1 x 2  
## ID tokens\_CHI  
## <chr> <dbl>  
## 1 e8e88ce3a62cb8f3e1209cf7c139f7faf676881f 1294

# 2  
final\_data %>%   
 filter(VISIT == "1") %>%   
 arrange(tokens\_CHI) %>%   
 head(1)

## # A tibble: 1 x 16  
## ID VISIT Diagnosis Ethnicity Gender Age ADOS MullenRaw ExpressiveLangR~  
## <chr> <fct> <chr> <chr> <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 4bb2~ 1 ASD White M 41.1 15 28 10  
## # ... with 7 more variables: Socialization <dbl>, MOT\_MLU <dbl>, CHI\_MLU <dbl>,  
## # types\_MOT <dbl>, types\_CHI <dbl>, tokens\_MOT <dbl>, tokens\_CHI <dbl>

USING SELECT

1. Make a subset of the data including only kids with ASD, mlu and word tokens
2. What happens if you include the name of a variable multiple times in a select() call?

# 1  
final\_data %>%   
 filter(Diagnosis == "ASD") %>%   
 select(ID, VISIT, CHI\_MLU, tokens\_CHI) %>%   
 head(3)

## # A tibble: 3 x 4  
## ID VISIT CHI\_MLU tokens\_CHI  
## <chr> <fct> <dbl> <dbl>  
## 1 d6afa93d3c2035a6d0b5fe880cb9ab02273e6eb3 1 2.28 461  
## 2 d6afa93d3c2035a6d0b5fe880cb9ab02273e6eb3 2 3.45 562  
## 3 d6afa93d3c2035a6d0b5fe880cb9ab02273e6eb3 3 3.12 983

# 2 - it is not included several times  
final\_data %>%   
 filter(Diagnosis == "ASD") %>%   
 select(ID, VISIT, CHI\_MLU, tokens\_CHI, ID) %>%   
 head(1)

## # A tibble: 1 x 4  
## ID VISIT CHI\_MLU tokens\_CHI  
## <chr> <fct> <dbl> <dbl>  
## 1 d6afa93d3c2035a6d0b5fe880cb9ab02273e6eb3 1 2.28 461

USING MUTATE, SUMMARISE and PIPES 1. Add a column to the data set that represents the mean number of words spoken during all visits. 2. Use the summarise function and pipes to add an column in the data set containing the mean amount of words produced by each trial across all visits. HINT: group by Child.ID 3. The solution to task above enables us to assess the average amount of words produced by each child. Why don’t we just use these average values to describe the language production of the children? What is the advantage of keeping all the data?

# 1 + 2 (don't see the difference?)  
final\_data %>%   
 group\_by(ID) %>%   
 summarise(mean\_CHI\_MLU = mean(CHI\_MLU, na.rm = T)) %>%   
 right\_join(final\_data) %>%   
 select(ID, VISIT, CHI\_MLU, mean\_CHI\_MLU) %>%   
 arrange(ID, VISIT) %>%   
 head(5)

## `summarise()` ungrouping output (override with `.groups` argument)

## Joining, by = "ID"

## # A tibble: 5 x 4  
## ID VISIT CHI\_MLU mean\_CHI\_MLU  
## <chr> <fct> <dbl> <dbl>  
## 1 0052610d778a89c392e4bc2592d53e67cb402cec 1 NA NaN   
## 2 0052610d778a89c392e4bc2592d53e67cb402cec 2 NA NaN   
## 3 087fb09b9b1ade8e554db444cadee9546d2a41dd 1 1.39 2.43  
## 4 087fb09b9b1ade8e554db444cadee9546d2a41dd 2 2.57 2.43  
## 5 087fb09b9b1ade8e554db444cadee9546d2a41dd 3 2.53 2.43