Psy/Educ 6600: Chapter 1 - Section C

Importing Data & Variable Manipulation

Your Name

Spring 2019

Contents

PREPARATION

Packages

```
library(tidyverse)
```

```
## -- Attaching packages -------------

## v ggplot2 3.3.0     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   0.8.5
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ----------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(readxl)
```

SECTION C

Question C-0: Import the data from an Excel file

Import Data File

Read in the Data from the Excel file (*Ihno_dataset.xls*) and store the data frame as the names/label `df_0`.

> Use the `read_excel()` function from the `readxl` package with the argument being the name of the excel file (with the *.xls*) in quotes. Preceed the fuction with the dataset name and the assignment opperator (`df_0 <-`).

NOTE: If you correctly read in the dataset and save it under the assigned name/label, then you will NOT produce any output, but you WILL see the dataset show up in panel with the tab called 'Environment'.

Print Full Data

Print out the dataset.

> Just type the name you stored the data as above (`df_0`).

List Variable Names

List out the **variable names**.

> Use the function `names()` from base **R**, with the only argument being the name you stored the data as above (`df_0`).

Count Dimentions

How many row (participants) and columns (variables) are in this dataset?

> Use the function `dim()` from base **R**, with the only argument being the name you stored the data as above (`df_0`).

Print Brief Data

Note the **class** of each variable, shown in pointed brackets after each variable name (`< >`).

> Use the function `glimpse()` from the `tibble` package, with the only argument being the name you stored the data as above (`df_0`).

Question C-1: Declare and label categorical variables

Revise the original data frame (`df_0`) to create **factor** versions of the categorical variables, including the text labels that correspond to each level. Store the new data frame as the names/label `df_1`.

- `gender` participant's gender
  - 1 = Female
  - 2 = Male
- `major` participant's major
  - 1 = Psychology
  - 2 = Premed
  - 3 = Biology
  - 4 = Sociology
  - 5 = Economics
- `reason` participant's reason for taking the course
  - 1 = Program requirement
  - 2 = Personal interest
  - 3 = Advisor recommendation
- `exp_cond` experimental condition participant was randomized to
  - 1 = Easy
  - 2 = Moderate
  - 3 = Difficult
  - 4 = Impossible
- `coffee`
  - 0 = Not a regular coffee drinker
  - 1 = Regularly drinks coffee

Declare Factors

Starting with the dataset you imported from Excel (stored as `df_0`) and convert all the variable names to lowercase with a piped step. Then add five more steps, also connected with pipes (`%>%`), and preceed everything with the new data frame's name and the assignment opperator (`df_1 <-`).

> Use the `rename_all()` function from the `dplyr` package in the first step. The only argument should be the command `tolower` (no quotes).
>
> Create each new varaible with the `mutate()` function from the `dplyr` package. The arguments within the parentheses should be of the form: `new_var_name = ...`. Create the new variable names to be the same as the original variable names with an additional capital "F" for factor at the end (eg. `genderF`).
>
> Set each new variable equal to an existing variable name enumerated with the `factor()` function from base **R**. This function will need three arguments: (1) the original variable's name, (2) a concatinated list of the *bare* numberic `levels` the variable is currently stored as, and (3) a concatinated list of the *quoted* text `labels` you the numeric levels pertain to.

NOTE: If you correctly save the new dataset under the assigned name/label, then you will NOT produce any output, but you WILL see the new dataset show up in panel with the tab called 'Environment'.

Print Brief Data

View the FIVE new varaibles at the end of the data frame. Also check the **class** of each variable; shown in pointed brackets after each variable name (`< >`).

> Use the function `glimpse()` from the `tibble` package, with the only argument being the name you stored the data as above (`df_0`).

Note: Now all the variable names are strictly in lower cases, other than the capital F's.

Question C-2: Create a new variable = `mathquiz` + 50

Create New Variable

Create a new dataset (`df_2`) by starting with the previous dataset (`df_1`) and adding a step connected by a pipe: `df_2 <- df_1 %>% ....`

> Create a new varaible with the `mutate()` function from the `dplyr` package. The arguments within the parentheses should be of the form: `new_var_name = ....`
>
> Use the addition (`+`) symbol to add 50 points to every `mathquiz` value and name the new varaible `mathquiz_p50`.

NOTE: If you correctly save the new dataset under the assigned name/label, then you will NOT produce any output, but you WILL see the new dataset show up in panel with the tab called 'Environment'.

Print Data Subset

Print the first few lines of the dataset, but only include the participant's identification number and the math quiz scores, both the original and the new version, to verify you created the new variable correctly.

> Use the `select()` function from the `dplyr` package with the vairable names, seperated with commas, you wish to keep as the arguments (no quotes needed).
>
> Use the `head()` function base **R** to print only the first few (default is `n = 6`).

Note: start with `df_2` and chain the steps together with pipes (%>%).

Question C-3: Create a new bariable = `hr_base` / 60

Create New Variable

Create a new dataset (`df_3`) by starting with the previous dataset (`df_2`) and adding a step connected by a pipe: `df_3 <- df_2 %>% ....`

> Create a new varaible with the `mutate()` function from the `dplyr` package. The arguments within the parentheses should be of the form: `new_var_name = ....`
>
> Use the division (`/`) symbol to divide the beats per minute by sixty to convert the every `hr_base` value and name the new varaible `hr_base_bps`.

NOTE: If you correctly save the new dataset under the assigned name/label, then you will NOT produce any output, but you WILL see the new dataset show up in panel with the tab called 'Environment'.

Print Data Subset

Print the last TEN few lines of the dataset, but only include the participant's identification number and the heat rates, both the original and the new version, to verify you created the new variable correctly.

> Use the `select()` function from the `dplyr` package with the vairable names, seperated with commas, you wish to keep as the arguments (no quotes needed).
>
> Use the `tail()` function base **R**, but include an argument to inclues to ten rows (`n = 10`).

Note: start with `df_2` and chain the steps together with pipes (`%>%`).

Question C-4: Create two new versions of the stat quiz score

USE PARENTHESES TO SPECIFY ORDER OF OPERATIONS CAREFULLY!

Create 1st New Variable: Add two, then multiply by ten

Create a new dataset (`df_4a`) by starting with the previous dataset (`df_3`) and adding a step connected by a pipe: `df_4a <- df_3 %>% ....`

> Create a new varaible with the `mutate()` function from the `dplyr` package. The arguments within the parentheses should be of the form: `new_var_name = ....`
>
> Use the addition (`+`) and multiplication (`*`) symbols, as well as parentheses, to convert every `statquiz` value and name the new varaible `statquiz_4a`.

NOTE: If you correctly save the new dataset under the assigned name/label, then you will NOT produce any output, but you WILL see the new dataset show up in panel with the tab called 'Environment'.

Create 2nd New Variable: multiply by ten, then add two

Create a new dataset (`df_4b`) by starting with the previous dataset (`df_4a`) and adding a step connected by a pipe: `df_4b <- df_4a %>% ....`

> Create a new varaible with the `mutate()` function from the `dplyr` package. The arguments within the parentheses should be of the form: `new_var_name = ....`
>
> Use the addition (`+`) and multiplication (`*`) symbols, as well as parentheses, to convert every `statquiz` value and name the new varaible `statquiz_4b`.

NOTE: If you correctly save the new dataset under the assigned name/label, then you will NOT produce any output, but you WILL see the new dataset show up in panel with the tab called 'Environment'.

Print Data Subset

Print the first few lines of the dataset, but only include the participant's identification number and the three stat quiz scores, the original and both the new versions, to verify you created the new variables correctly.

> Use the `select()` function from the `dplyr` package with the vairable names, seperated with commas, you wish to keep as the arguments (no quotes needed).
>
> Use the `head()` function base **R** to print only the first few (default is `n = 6`).

Note: start with `df_4b` and chain the steps together with pipes (`%>%`).

Question C-5a: Create a new variable that is the SUM of the 3 anxiety measures

Create New Variable

Create a new dataset (`df_5a`) by starting with the previous dataset (`df_4b`) and adding a step connected by a pipe: `df_5a <- df_4b %>% ....`

> Create TWO new varaibles, both with the `mutate()` function from the `dplyr` package. The arguments within the parentheses should be of the form: `new_var_name = ....`
>
> For the first new variable, use addition symbols (`+`) add the three original anxiety scores and name the new varaible `anx_plus`.
>
> For the second new variable, use the `sum()` function from base **R** with the thee original axiety variable names as the arguments, seperated with commas. Name this new variable `anx_sum`.

NOTE: If you correctly save the new dataset under the assigned name/label, then you will NOT produce any output, but you WILL see the new dataset show up in panel with the tab called 'Environment'.

Print Data Subset

Print the first few lines of the dataset, but only include the participant's identification number, the three anxiety scores, and the two new combined scores, to verify you created the new variables correctly.

> Use the `select()` function from the `dplyr` package with the vairable names, seperated with commas, you wish to keep as the arguments (no quotes needed).
>
> Use the `head()` function base **R** to print only the first few (default is `n = 6`).

Note: start with `df_5a` and chain the steps together with pipes (%>%).

Question C-5b: Create a new variable that is the AVERAGE of the 3 heart rates

Create New Variable

Create a new dataset (`df_5b`) by starting with the previous dataset (`df_5a`) and adding a step connected by a pipe: `df_5b <- df_5a %>% ....`

> Create TWO new varaibles, both with the `mutate()` function from the `dplyr` package. The arguments within the parentheses should be of the form: `new_var_name = ....`
>
> For the first new variable, use addition symbols (`+`) add the three original heart rates and then surrond the sum with parentheses followed by teh division (`/`) by three. Name the new varaible `hr_avg`.
>
> For the second new variable, use the `mean()` function from base **R** with the thee original heart rate variable names as the arguments, seperated with commas. Name this new variable `hr_mean`. You will also need to include two additional arguments after the varaible names (`, na.rm = TRUE, trim = 0`).

NOTE: If you correctly save the new dataset under the assigned name/label, then you will NOT produce any output, but you WILL see the new dataset show up in panel with the tab called 'Environment'.

Print Data Subset

Print the first few lines of the dataset, but only include the participant's identification number, the three heart rates, and the two new combined scores, to verify you created the new variables correctly.

> Use the `select()` function from the `dplyr` package with the vairable names, seperated with commas, you wish to keep as the arguments (no quotes needed).
>
> Use the `head()` function base **R** to print only the first few (default is `n = 6`).

Note: start with `df_5b` and chain the steps together with pipes (%>%).

Question C-6: Create a new variable that is the difference between `statquiz` and `Exp_sqz`

Create New Variable

Create a new dataset (`df_6`) by starting with the previous dataset (`df_5b`) and adding a step connected by a pipe: `df_6 <- df_5b %>% ....`

> Create a new varaible with the `mutate()` function from the `dplyr` package. The arguments within the parentheses should be of the form: `new_var_name = ....`
>
> Use the subtraction (`-`) symbol calculate the difference and name the new varaible `statdiff`.

NOTE: If you correctly save the new dataset under the assigned name/label, then you will NOT produce any output, but you WILL see the new dataset show up in panel with the tab called 'Environment'.

Print Data Subset

Print the first few lines of the dataset, but only include the participant's identification number, the two stat quiz scores, and the new difference score, to verify you created the new variable correctly.

> Use the `select()` function from the `dplyr` package with the vairable names, seperated with commas, you wish to keep as the arguments (no quotes needed).
>
> Use the `head()` function base **R** to print only the first few (default is `n = 6`).

Note: start with `df_6` and chain the steps together with pipes (%>%).

Put it all together

Combine all the varaible manipulation steps into a single code chunk, starting with reading in the excel data file. Name this data frame `ihno_clean`.

For variables that were created multiple ways (5a, 5b), choose one way only.

Note: This will be one of the first code chunks in every remaining homework assignment.