FOAR705 LEARNING JOURNAL R

MONA GHAI

NOVEMBER 2019

1. INTRODUCTION

I will be completing the ‘R for Social Scientists’ data carpentry exercises.

Objective: create a new project using R

Action: Opened R; Clicked on FILE menu; clicked NEW PROJECT; selected NEW DIRECTORY and then New Project. This new directory was named ‘data carpentry’; clicked Create Project; clicked ‘file’; then selected ‘new file’ and then ‘R Script’; clicked Save icon on toolbar and saved it as ‘R.Script’.

Errors: no errors.

Result: A new project is created for Data Carpentry Exercises.

DOWNLOADING THE DATA AND SETTING UP

Objective: To download the data necessary to do R data carpentry

Action: Type ‘dir.create(“data”)’ into the console; Type ‘dir.create(“data-output”)’ ; then type ‘dir.create(“fig-output”)’; Download dataset ‘SAFI-clean.csv’; and finally placed dataset in folder I created.

Result: the dataset needed for R data carpentry was successfully downloaded.

INSTALLING ADDITIONAL PACKAGES

Objective: Using R, install ‘tidyverse’ package.

Action: Clicked on package tab; clicked ‘install’ option; type ‘tidyverse’ in the text box; clicked ‘install’.

Result: ‘tidyverse’ package was successfully installed.

2. INTRODUCTION TO R

Objective: to show that changing the values of 2 variables ‘length’ and ‘breadth’ does not change the value of third variable ‘area’.

Action: Type length- 2.5’

Type width- 3.2’

Type area- length * width’

Area variable- 8

Changed value of length- 4

And value of width—5

Result: The value of ‘area’ variable remained the same.

EXERCISE 2:

Objective: To type ?round into console and to see what functions are similar, and how ‘digits’ parameter in round function can be used.

Action: Type ?round in console

Result: It opened the help tab, similar functions are ‘ceiling’, ‘floor’, ‘signif’. ‘Digits’ parameter in the round function is tells the number of decimal places.

VECTORS AND DATA TYPES

Exercise 1

Objective: to see what happens when we mix different atomic vectors: character, numeric, integer and logical, in a single vector. Next is to test the examples given and finally determine how many values in a ‘combined-logical’ vector are TRUE.

Action: Type ‘test-vectors -c(3, 6, “a”, “b”, TRUE)’; then type ‘typeof(test-vectors)’; type ‘num-chair’-c(1, 2, 3, “a”); type ‘class(num-chair), the result is ‘character’; type ‘num-logical- c(1, 2, 3, TRUE); type class(num-logical); result is ‘numeric’; type ‘char-logical-c(“a”, “b”, “c”, TRUE); type ‘class(char-logical)’; result is ‘character’; type ‘tricky-c(1, 2, 3, “4”)’; type ‘class(tricky)’; result is ‘character’; type ‘combine -logical-c(num-logical, char-logical)’.

ERRORS: NONE

RESULT: R converted all different types in each vector into one type. Vectors can be of one data type. R tries to convert the content of the vector to find a ‘common denominator’ which does not lose the information. Further, only one value is TRUE in ‘combined -logical’ vector.

MISSING DATA

Exercise

Objective: Using the given vectors, create a new vector with the NAs removed, then use the function ‘median()’ to calculate the median of the ‘rooms’ vector, and finally using R to figure out how many households in the set use more than 2 rooms for sleeping.

Action: Type ‘rooms <-c(1, 2, 1, 1, NA, 3, 1, 3, 2, 1, 1, 8, 3, 1, NA, 1)’

Type ‘rooms_no_na <- rooms[! Is.na(rooms)]

Type median(rooms, na . rm = TRUE)

Result is ‘[1] 1’

Type ‘rooms_above_2 < - rooms_no_na > 2]

Type ‘length(rooms_above_2)

Result is ‘[1] 4’

ERROR: None

RESULT: A new vector was created with NA’s removed. Median was calculated and the answer is 1. 4 households used more than 2 rooms for sleeping.

3. STARTING WITH DATA

PRESENTATION OF THE SAFI DATA.

Objective: to load the SAFI data in R

Action: Type ‘library(tidyverse)’

Type ‘interviews-read-csv(“data/SAFI-clean.csv” , na = “NULL”)’

Type ‘interviews’

Dataset appeared in the environment box

Cells icon within ‘interviews’ tab is clicked

The data is loaded as a table

Result: The data is loaded successfully.

INSPECTING THE DATA FRAMES

Objective: testing the functions that extract information from data frames in the section.

Action: ‘dim(interviews)’; type ‘nrow(interviews)’; type ‘ncol(interviews)’; type ‘head(interviews); type ‘tail(interviews)’; type ‘name(interviews)’; type ‘str(interviews)’;type ‘summary(interviews).

ERRORS: None

Result: first function showed the number of rows and columns in the object, in the second only number of rows are shown, third shows the number of columns, fourth shows the first six rows, the fifth shows the last six rows, sixth shows the column names, seventh shows the information about the class, length, content of all the columns and the last function the summary for each column.

INDEXING AND SUBSETTING DATA FRAMES

Objective: Create a data frame 'interviews_100' containing only the data in row 100 of the 'interviews' dataset, next pull out the last row in the data frame and create a new data frame titled 'interviews-last' from the row. Next, use 'nrow()' to extract the middle row of the data frame, and store the content in an object titled 'interviews-middle'. Lastly, the 'nrow' will be combined with the '-' notation to reproduce the bevaviour of 'head (interviews)', keeping just the first through 6th rows of the interviews dataset.

Action: type 'interviews-100 <-interviews[100,]'

Type 'n-rows <- nrow(interviews)'

Type 'interviews-last <-interviews[n-rows,]'

Type 'interviews-middle <-interviews[(n-rows / 2),]'

Type 'interviews-head <-interviews[-(7:n-rows),]'

ERRORS: None.

4. Introducing dplyr and tidyr

PIPES

Exercise

Objective: Using pipes, subset the ‘interviews’ data to include interviews where respondents were members of an irrigation association ‘memb_assoc’ and retain only the columns ‘affect_conflict’, ‘liv_content’ and ‘no-meals’.

Action: type ‘interviews %>%, type ‘filter(memb_assoc == “yes”) %>%, type ‘select(affect_conflicts, liv_count, no-meals).

ERRORS: None.

MUTATE

Exercise.

Objective: To create a new data frame from the ‘interviews’ data that meets the following criteria: contains only the ‘village’ column called ‘total meals’ containing the value that is equal to the total numbers of meals served in the household per day on average (‘no membrs’ times ‘no meals’). Only the rows where ‘total_meals’ is greater than 20 should be shown in the final data frame.

Action: Type ‘interviews_total_meals < - interviews %>%’

Type ‘mutate(total_meals = no-membrs * no_meals) %>%’

Type ‘filter(total_meals > 20) %>%’

Type ‘select(village, total_meals)’

ERRORS: None.

Result: got the desired result.

Exercise.

Objective: To find out how many households in the survey have an average of two meals per day? Three meals per day? Are there any other numbers of meals represented?

Action: type ‘interviews %>%

Type ‘count(no_meals)’

The outcome is that 52 households has 2 meals per day, and 79 had 3 meals per day.

Type ‘interviews %>%’

Type ‘group_by(village) %>%’

Type ‘summarize (

mean_no_membrs = mean(no_membrs) ,

min_no_membrs = min(no_membrs) ,

max_no_membrs = max(no_membrs) ,

n = n()

type closing ‘)’

This resulted in the mean, minimum and maximum number of household members for each village.

RESHAPING WITH GATHER AND SPREAD

Exercise.

Objective: Create a new data frame (named ‘interviews_months_lack_food’) that has one column for each month and records ‘TRUE’ or ‘FALSE’ for whether each interview respondent was lacking food in that month.

Action: Type ‘interviews_months_lack_food < - interviews %>%’

Type ‘separate_rows(months_lack_food, sep=”;”) %>%’

Type ‘mutate(months_lack_food_logical = TRUE) %>%’

Type ‘spread(key = months_lack_food, value = months_lack_food_logical, fill = FALSE).

ERORS: None.

Result: After this, the ‘interviews-spread’ data frame does not have column titled ‘respondent-wall-type’.

GATHERING

Exercise

Objective: gathering the columns names and transform them into two new variables, one representing the column names as values, and the other containing values previously associated with the columns names. To filter the data, then follow the steps.

Action: type ‘interviews_gather < - interviews_spread %>%’

Type gather(key = “respondent_wall_type” , value = “wall_type_logical”,

burntbricks : sunbricks) %>%

type ‘filter(wall_type_logical) %>%’

type ‘select(-wall_type_logical)’

ERRORS: None.

APPLYING ‘Spread()’ TO CLEAN THE DATA.

Exercise.

Objective: To create a new data frame (named ‘interviews_months_lack_food’) that has one column for each month and records ‘TRUE’ or ‘FALSE’ for whether each interview respondent was lacking food in that month and to determine how many months (on average) were respondents without food if they did belong to an irrigation association? What about if they didn’t?

Action: Type ‘interviews_months_lack_food < - interviews %>%’

Type ‘separate_rows(months_lack_food, sep=”;”) %>%’

Type ‘mutate(months_lack_food_logical = TRUE) %>%’

Type ‘spread(key = months_lack_food, value = months_lack_food_logical, fill = FALSE)’

Produced desired data frame.

Type ‘interviews_months_lack_food %>%’

Type ‘mutate(number_months = rowSums(select(. , Apr:Sept))) %>%’

Type ‘group_by(memb_assoc) %>%’

Type ‘summarize(mean_months = mean(number_months))’

Result: the respondents who were part of an irrigation association were 2.64 months on average without food and who were not a part of irrigation were 2.31 months on average without food.

ERRORS: None.

EXPORTING DATA

Exercise.

Objective: To create a version of the dataset where each of the columns includes only one data value to be used for next lesson.

Action: Type ‘interviews_plotting < - interviews %>%’

Type ‘separate_rows(items_owned, sep=”;”) %>%’

Type ‘spread(key = items_owned, value = items_owned_logical, fill = FALSE) %>%’

Type ‘rename(no_listed_items = ‘<NA>’) %>%’

Type ‘separate_rows(months_lack_food, sept=”;”) %>%’

Type ‘mutate(months_lack_food_logical = TRUE) %>%’

Type spread(key = months_lack_food rowSums(select(. , Apr:Set))) %>%’

Type ‘mutate(number_items = rowSums(select(. , bicycle:television)))’

Tried to save this data frame to the ‘data-output’ directory but got an error.

Result: able to create the version but unable to save it to the directory.

5. DATA VISUALIZATION WITH ggplot2

Exercise.

Objective: to build a ggplot with data from the previous lesson (‘interviews plotting’)

Action: Type ‘ggplot(data = interviews-plotting)’

Type ‘ggplot(data = interviews-plotting, aes(x = no-members, y = number-items))’

Type ‘ggplot(data = interviews-plotting, aes(x = no-members, y = number-items)) +’

Type ‘gcom-point()’

This gave a plot diagram.

RESULT: successfully created the plot using data from previous lesson.

BUILDING THE PLOTS ITERATIVELY

Objective: Continue with the process in the lesson, building the plots iteratively.

Action: Type ‘ggplot(data = interviews-plotting, aes(x = no-membrs, y = number-

items)) +’

Type ‘geom-point()’

Type ‘ggplot(data = interviews-plotting, aes(x = no-membrs, y = number-items)) +’

Type ‘geom-point(alpha = 0.5)’

Type ‘ggplot(data = interviews-plotting, aes(x = no-membrs, y = number-items)) +’

Type ‘geom-jitter(alpha = 0.5)’

Type ‘ggplot(data = interviews-plotting, aes(x = no-membrs, y = number-items)) +’

Type ‘geom-jitter(alpha = 0.5, color = ”blue”)’

Type ‘ggplot(data = interviews-plotting, aes(x = no-membrs, y = number-items)) +’

Type ‘geom-jitter(aes(color = village), alpha = 0.5)’

ERROR: None.

RESULT: successfully plotted an informative graph showing the data.

Exercise

Objective: To create a scatter a scatter plot of ‘rooms’ by ‘village’ with the different ‘respondent-wall-types’ shown in different colors. Determine whether this is a good way to show this data type.

Action: Type ‘ggplot(data = interviews-plotting, aes(x = village, y = rooms)) +’

Type ‘geom-jitter(aes(color = respondent-wall-type))’

ERRORS: None.

Result: successfully created the scatter plot. This is not a good way to show this kind of data as it is quite confusing, there is a difficulty in differentiating the various villages.

BOXPLOT

Objective: To show the use of Boxplot in order to show the distribution of rooms for the different wall types.

Action:

Type ‘ggplot(data = interviews-plotting, aes(x = respondent-wall-type,

y = rooms)) +’

Type ‘geom-boxplot()’

Type ‘ggplot(data = interviews-plotting, aes(x = respondent-wall-type, y = rooms)) +’

Type ‘geom-boxplot(alpha = 0) +’

Type ‘geom-jitter(alpha = 0.5, color = ”tomato”

ERRORS: NONE

EXERCISE.

Objective: To replace the boxplot with a violin plot, then to create a new boxplot for ‘liv-count’ for each wall type , overlaying the boxplot layer on a jitter layer to show the real measurements, after that adding color to the data points in the boxplot depending upon whether they are part of an irrigation association or not.

Action: type ‘ggplot(data = interviews-plotting, aes(x = respondent-wall-type, y = rooms)) +’

Type ‘geom-violin(alpha = 0) +’

Type ‘geom-jitter(alpha = 0.5, color = ”tomato”)’

Continuing with the next part, type ‘ggplot(data = interviews-plotting, aes(x = respondent-wall-type, y = liv-count)) +’

Type ‘geom-boxplot(alpha = 0) +’

Type ‘geom-jitter(alpha = 0.5)’

Finally, type ‘ggplot(data = interviews-plotting, aes(x = respondent-wall-type, y = liv-count)) +’

Type ‘geom-boxplot(alpha = 0) +’

Type ‘geom-jitter(aes(alpha = 0.5, color = memb-assoc))’

Error: None

Result: successfully completed the exercise.

BARPLOTS

Objective: To depict the process shown in this section.

Action: Type ‘ggplot(data = interviews-plotting, aes(x = respondent-wall-type)

Type ‘geom-bar()’

Type ‘ggplot(data = interviews-plotting, aes(x = respondent-wall-type))
+’

Type ‘geom-bar(aes(fill = village))’

Type ‘ggplot(data = interviews-plotting, aes(x = respondent-wall-type)) +’

Type ‘geom-bar(aes(fill = village), position = ”dodge”)’

Type ‘percent-wall-type ¡- interviews-plotting *%*’

Type ‘filter(respondent-wall-type != ”cement”) *%*’

Type ‘count(village, respondent-wall-type) *%*’

Type ‘group-by(village) *%*’

Type ‘mutate(percent = n / sum(n)) *%*’

Type ‘ungroup()’

Type ‘ggplot(percent-wall-type, aes(x = village, y = percent, fill = respondent-wall-type)) +’

Finally, type ‘geom-bar(stat = ”identity”, position = ”dodge”)’

ERRORS: None.

EXERCISE

Objective: To create a bar plot showing the proportion of respondents in each village who are or are not part of an irrigation association (memb_assoc) Include only respondents who answered that question in the calculations and plot. Further, which village had the lowest proportion of respondents in an irrigation association?

Action: Type ‘percent-memb-assoc <- interviews-plotting *%*’

Type ‘filter (! is.na(memb-assoc)) *%*’

Type ‘count(village, memb-assoc) *%*’

Type ‘group-by(village) *%*

Type ‘mutate(percent = n / sum(n)) *%*’

Type ‘ungroup()’

Type ‘ggplot(percent-memb-assoc, aes(x = village, y = percent,
fill = memb-assoc)) +’

Type ‘geom-bar(stat = ”identity”, position = ”dodge”)

ERRORS: None

Result: Successfully created the bar plot. Ruaca village has the lowest proportion of respondents in an irrigation association.

Adding Labels and Titles

Exercise.

Objective: To show the process shown in this section.

Action: Type ‘ggplot(percent-wall-type, aes(x = village, y = percent, fill = respondent-wall-type)) +’

Type ‘geom-bar(stat = ”identity”, position = ”dodge”) +’
Type ‘labs(title=”Proportion of wall type by village”,’

Type ‘x=”Wall Type”,’

Type ‘y=”Percent”)’

ERRORS: None

FACETING

Objective: to show the process depicted in this section.

Action: Type ‘ggplot(percent-wall-type, aes(x = respondent-wall-type, y = per-

cent)) +

Type ‘geom-bar(stat = ”identity”, position = ”dodge”) +’

Type ‘labs(title=”Proportion of wall type by village”,’

Typed ’x=”Wall Type”,’

Type ‘y=”Percent”) +’

Type ‘facet-wrap(* village)’

Type ‘ggplot(percent-wall-type, aes(x = respondent_wall_type, y = per-

cent)) +’

Type ‘geom_bar(stat = ”identity”, position = ”dodge”) +’

Type ‘labs(title=”Proportion of wall type by village”,’

Type ‘x=”Wall Type”,’

Type ‘y=”Percent”) +’

Type ‘facet_wrap(~ village) +

Type ‘theme_bw() +’

Type ‘theme(panel.grid = element_blank())’

Type ‘percent_items < - interviews-plotting *%*

Type ‘gather(items, items_owned_logical, bicycle:no_listed_items) *%*’

Type ‘filter(items-owned-logical) *%*’

Type ‘count(items, village) *%*

Type ‘mutate(people_in_village = case_when(village == ”Chirodzo”
~

39,’

Type ‘village == ”God” ~ 43,’

Type ‘village == ”Ruaca” ~ 49)) *%*’

Type ‘mutate(percent = n / people_in_village)’

Type ‘ggplot(percent-items, aes(x = village, y = percent)) +’

Type ‘geom_bar(stat = “identity”, position = “dodge”) +’

Type ‘facet_wrap(~ items) +

Type ‘theme-bw() +’

Type ‘theme(panel.grid = element-blank())’

ERRORS: None.

‘ggplot2’ themes

Exercise.

Objective: To experiment with 2 themes by building previous plot using each of the themes and to find which is the best.

ACTION: First, I experimented with the ’theme-dark’ theme

Type ‘ggplot(percent_items, aes(x = village, y = percent)) +’

Type_geom_bar(stat = ”identity”, position = ”dodge”) +’

Type_facet_wrap(~ items) +’

Type ‘theme-dark() +’

Type ‘theme(panel.grid = element_blank())’

Secondly, theme I tried was the ’theme_minimal’ theme

Type ‘ggplot(percent_items, aes(x = village, y = percent)) +’

Type ‘geom-bar(stat = ”identity”, position = ”dodge”) +’

Type ‘facet-wrap(~items) +’

Type ‘theme_minimal() +’

Type ‘theme(panel.grid = element_blank())’

ERRORS: None.

RESULT: ‘theme-dark’ is aesthetically better than ‘theme-minimal’

CUSTOMIZATION

Exercise

Objective: To improve one of the plots generated in this lesson, and save the file in the directory.

Action: Type ‘ggplot(percent_items, aes(x = village, y = percent)) +’

Type ‘geom-bar(stat = ”identity”, position = ”dodge”) +’

Type ‘facet-wrap(~ items) +’

Type ‘labs(title = ”Percent of respondents in each village \n who owned
each item”,’

Type ‘x = ”Village”,’

Type ‘y = ”Percent of Respondents”) +’

Type ‘theme_dark() +’

Type ‘theme(axis.text.x = element_text(colour = ”grey20”, size = 12,
angle = 45, hjust = 0.5, vjust = 0.5),’

Type ‘axis.text.y = element_text(colour = ”grey20”, size = 12),’

Type ‘text = element_text(size = 16))’

To save the graph in the directory, type ‘my-plot -
ggplot(percent_items, aes(x = village, y = percent)) +’

Type ‘geom-bar(stat = “identity”, position = “dodge”) +’

Type ‘facet-wrap(~ items) +’

Type ‘labs(title = “Percent of respondents in each village \n who
owned each item”,
Type ‘x = “Village”,’

Type ‘y = “Percent of Respondents”) +’

Type ‘theme-dark() +’

Type ‘theme(axis.text.x = element-text(colour = ”grey20”, size = 12,

angle = 45, hjust = 0.5, vjust = 0.5),’

Type ‘axis.text.y = element-text(colour = ”grey20”, size = 12),’

Type ‘text = element_text(size = 16),’

Type ‘plot.title = element_text(hjust = 0.5))’

Type ‘ggsave(”fig_output/data_carpentry graph.png”, my_plot, width = 15, height = 10)’

Error: None.

Result: Successfully completed R.

PROOF OF CONCEPT (IMPLIMENTATION)

The project entails making Mendeley (citation ref tool) communicate with hypothes.is(online tool). Earlier I was trying to use Zotero but then shifted to Mendeley as I found it easier to learn.

I successfully learnt how to use Mendeley for reference management. Earlier I was confused and thought that my task for technology deployment was concerned with learning how use Mendeley to tag my notes with keywords, manually editing the bibliographical metadata, adding annotations to online sources (learning Hypothes.is), automatically formatting notes (using LaTex).

For few days, I learnt the Mendeley software but after demonstrating it in class, Brian told that I was not on the right track as all these features are already in Mendeley. For the project I had to, ofcourse, learn Mendeley and Hypothes.is but I have to make both of them communicate with each other. Hypothes.is is a useful resource for annotating on the document.

I, then learnt Hypothes.is. I tried to learn the linking process but after lots of efforts was unable to do so.

DATA RECOVERY(DUPLICATI)

Earlier I thought of using IDrive but Shawn and Brian raised certain questions, and then I shifted to Duplicati**.**

I will use Duplicati as my disaster recovery plan.

Action: Went to https://www.duplicati.com and clicked blue button 'Download Duplicati 2.0 (beta)'.

1st November(1pm)

Met Brian to demonstrate the backup plan. But it wasn't complete. I demonstrated the setting up of the backup and he said me to show the backup next week.

Setting up a backup:

- Open Duplicati, on the left hand side of the menu select '+ Add Backup'. In the page that opened I selected 'configure a new backup', then clicked next
- In 'General backup settings', I gave 'name' Mona Backup, and a passphrase, repeated the passphrase, and clicked next
- In the 'storage type', I selected 'Google Drive', and then opened my google drive and copied the destination path from there and pasted it on the path on server, clicked 'AuthID' link to create AuthID, tested the connection, it worked, clicked next
- Source Data: I navigated to the folder I wanted to backup, I ticked the 'music' box and then clicked next
- From schedule I selected 1 pm daily for backup and from general options a drop down box appeared, I selected 'smart backup retention', I stored the passphrase safely
- Then again the home page appeared, where I could see my backup. Under that, I clicked 'Run Now'. The progress of backup could be seen in the progress bar. Clicked the down arrow next to backup name, under 'configuration', selected 'Export'. Further, I selected 'Encrypt file', entered passphrase. Then clicked Export and the file was downloaded.

8th Nov(1pm).

Met Brian again and successfully demonstrated the restoration of the file.

RESTORE A FILE FROM A BACKUP.

- I downloaded the configuration file. Opened duplicati and selected Restore in the menu on the left side. Selected ‘Restore from the configuration’, clicked next
- Clicked ‘choose file’, went to my download folder, selected the configuration file and clicked Choose. Entered the password and selected import. Backup location automatically was filled and I selected the test connection button. It said connection worked, clicked next
- In the next page , I selected Connect, the dropdown files page had a menu(Restore from) from which a date of backup could be selected. After selecting my file, clicked Continue.
- In the next page(Restore options) I selected ‘pick Location’ and chose a folder on my desktop., clicked ‘Restore’.

Result: successful in restoring a file, got message from duplicate. Demonstrated the complete process to Brian successfully.