CHAPTER 3

R Datasets

OBJECTIVE

This chapter focuses on the 2 important tasks of bringing your data into R, and making it accessible to your scripts. You will learn various techniques for importing your data, exporting it, organizing it, saving it, and retrieving it. I will introduce the important concepts of project and working directories and how to use them to improve the portability of your R scripts. The importance of this chapter cannot be emphasized enough. Take time to read through it, especially if you are new to R.

Contents

3.1 Introduction

Bringing your data into R is one of the key tasks that you will often need to perform before your data analysis can begin. Therefore, the main objective of this chapter is to describe the best ways of creating R datasets. Section 2.5.1 of chapter 2 gave you a glimpse into the creation of R data frames. In this chapter however, you will learn much more about this important topic.

R is an old language, which is actually a dialect of another older language named S that I used extensively as a graduate student in the late eighties. The capability of this language is expanded nowadays through the development of packages, which are R modules created by independent developers. These packages must be installed, then loaded to your R session by you before they become usable. One key package that every R user needs is called `tidyverse`. Even for the purpose of creating R data frames I recommend using it. Therefore, I will first show how to install and load it before moving on to other topics.

The installation of the `tidyverse` is done entirely inside the RStudio Console pane (see Figure 2.5) as follows:

- Type `> intall.packages("tidyverse")` in the RStudio console as shown in Figure 3.1, and press `ENTER`. Once the installation is completed, the Console and Files windows will appear as shown in Figure 3.2. The `tidyverse` package must appear on the `Files` window, which is a confirmation that this package was successfully installed.

- Once installed, the `tidyverse` package will be usable only if it is loaded to your R session. To load `tidyverse`, type `> library(tidyverse)` in the RStudio console as shown in Figure 3.3 and press `ENTER`. You may receive a warning message as shown in this figure. This warning message will generally be harmless and can safely be ignored. If you do want to get rid of this warning, the solution would be to install the latest version of R as previously discussed.

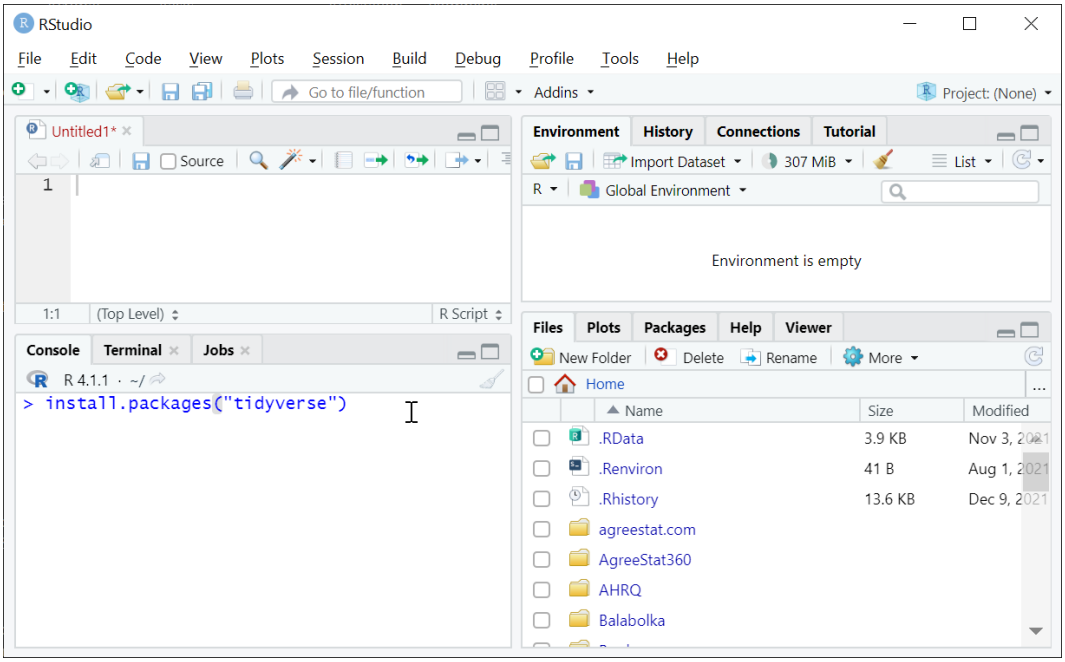

Figure 3.1: Installing the `tidyverse` package

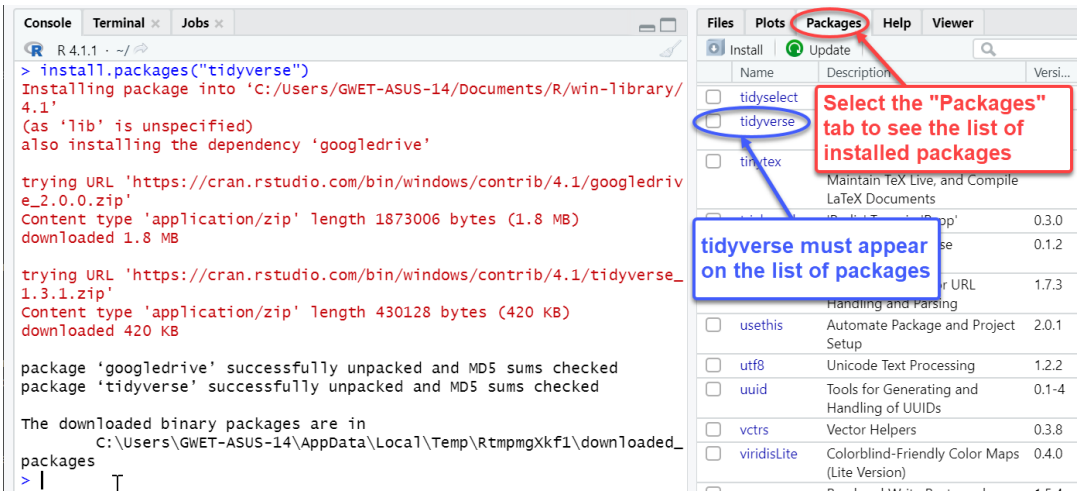

Figure 3.2: R console after `tidyverse` package has been installed

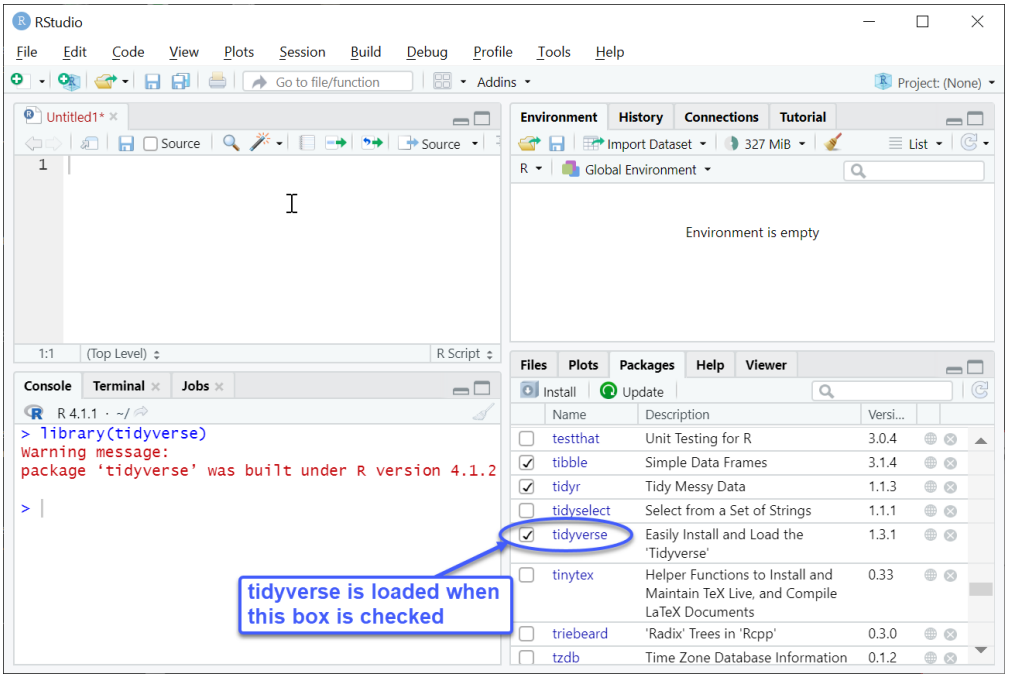

Figure 3.3: Loading the `tidyverse` package after installation

R datasets are generally called `data frames`. However, there exists a modern version of a data frame called `tibble`. It works with `tidyverse` and avoids many known issues associated with data frames. Therefore, I will primarily use tibbles from now on. Although there are some subtle differences between data frames and tibbles, which I will occasionally allude to, I will often use the terms "data frame" and "tibble" interchangeably. Each time I say R dataset or data frame, I will generally refer to a tibble. The terms data frame and tibble are good to know, since you often see them in several R reference books.

3.2 Manipulating Datasets in R

In this section, I am going to briefly discuss the use of datasets in R. You will learn how to bring your existing dataset to R and get it ready for analysis. You will also learn how to save your datasets and other R objects to the disk

in R formats for subsequent use. Exporting your R datasets to non-R formats is also an important issue that will be addressed. First, let us explore some R datasets that come with R, and are ready for use.

3.2.1 *R Built-In Datasets*

You may want to know that R comes with several built-in datasets that you can readily use. A list of these datasets can be obtained by calling the `data()` function from the RStudio console as shown in Figure 3.4. A file named `R data sets` and containing all built-in data frames will appear on the `Scripts` pane. One of these datasets is named `ChickWeight` and contains the weight versus age of chicks on different diets. If you want to print out this dataset, then type the command `> print(ChickWeight)` on the console, and R will print out the first 250 rows of data before displaying the message `[ reached 'max' / getOption("max.print") -- omitted 328 rows ]`. This message indicates that the default maximum number of observations that could be displayed on the console has been reached. This printing style is generally an indication that you are dealing with a traditional data frame. Since I want you to use tibbles instead, you need to coerce any traditional data frame to a tibble.

Coercing the data frame `ChickWeight` to a tibble can be done as shown in Figure 3.6. The command `chick.data <- as_tibble(ChickWeight)` creates the R object `chick.data`, which is the tibble version of the `ChickWeight` data frame[1]. When you print out the tibble `chick.data`, the outcome shown on the console presents the following features:

- The title in gray color says `# A tibble: 578 x 4`, meaning that your tibble has 578 rows and 7 columns.

- Following the title, are column labels or variable names of the dataset.

[1]R cannot find the `as_tibble()` function if you do not first load the `tidyverse` package as in Figure 3.3.

- Below each column label, is the data type in gray color. The data types associated with the variables `weight`, `Time`, `Chick` and `Diet` are `<dbl>`, `<dbl>`, `<ord>` and `<fct>`. `<dbl>` refers to double precision floating point numbers. `<ord>` indicates that the data is ordinal. It identifies records following a natural order. `<fct>` is the factor type. It takes a limited number of values, categorizes and stores them as levels.

- Only 10 records are displayed by default. This will help when the total number of records in the dataset is unknown. You can display more records. For example, `> print(chick.data,n=15)` will display 15 rows, not the default 10. If you want to print all rows in the `chick.data` tibble, then execute the command `> print(chick.data, n=Inf)`, where `Inf` stands for infinity.

- The last line of the printout in gray color says, `# ... with 568 more rows` and indicates that there are 568 more rows to display.

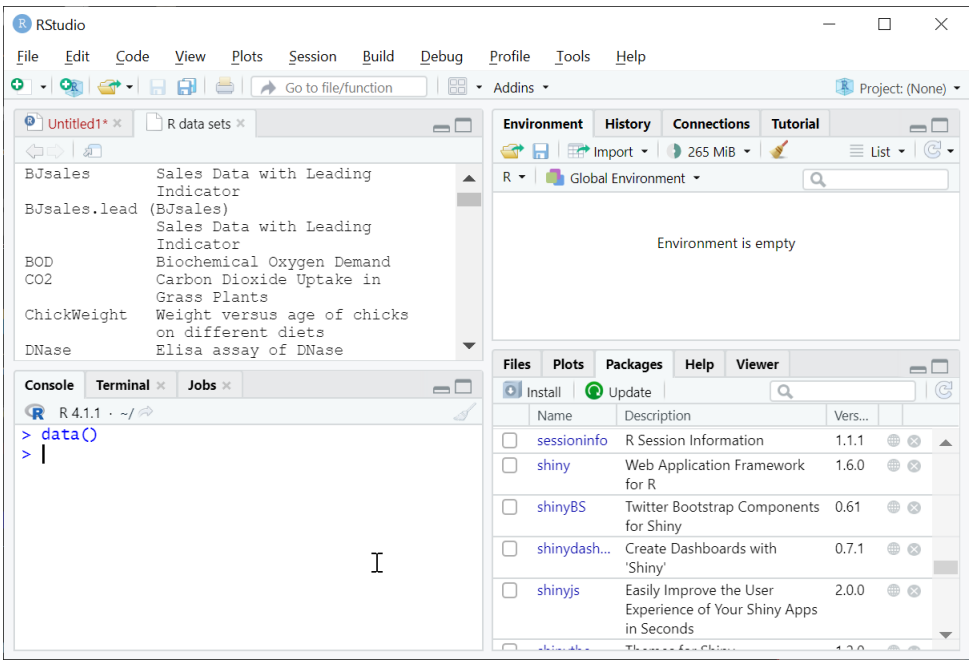

Figure 3.4: List of built-in R datasets

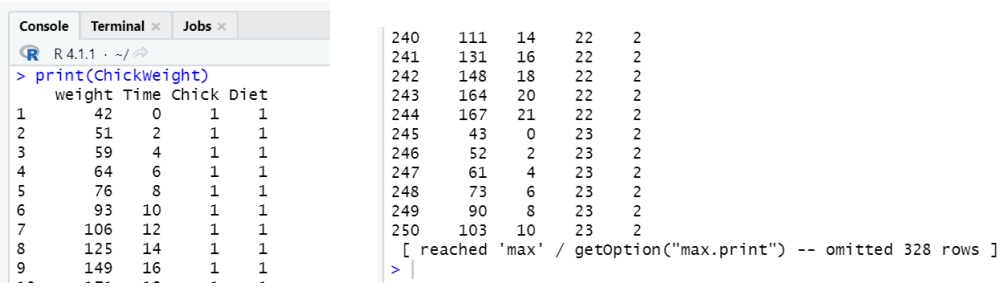

Figure 3.5: Printing of the ChickWeight data frame

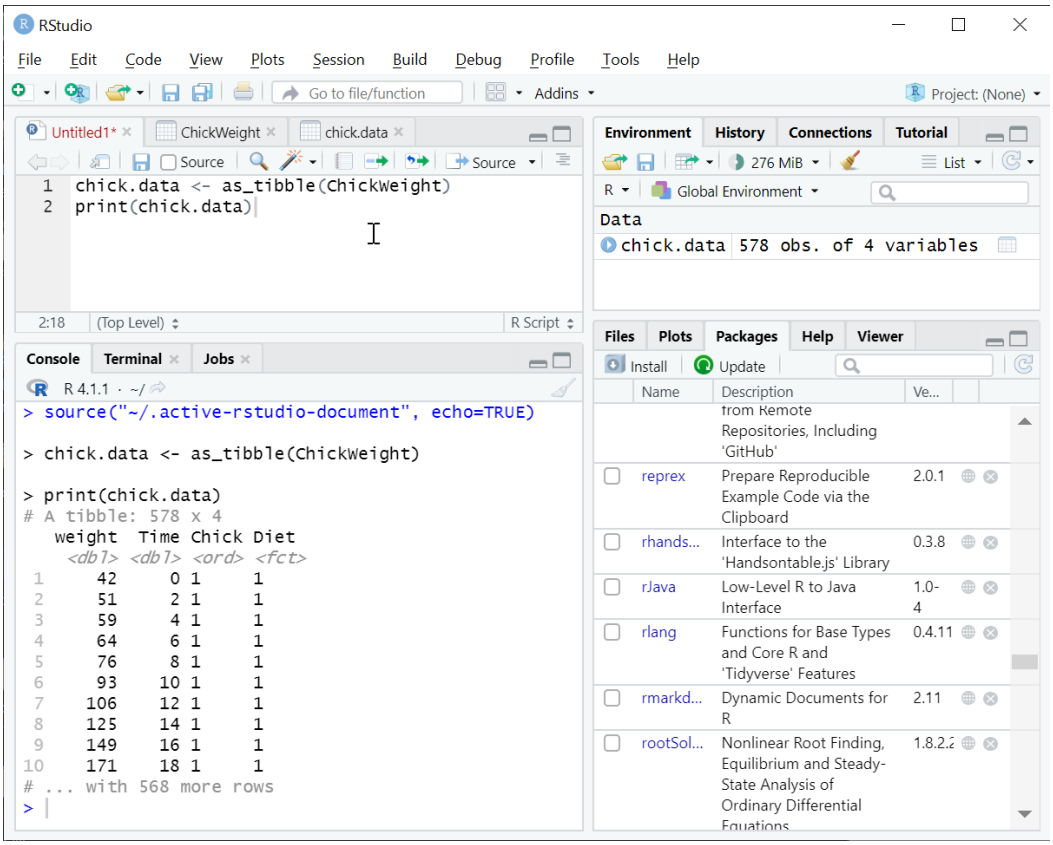

Figure 3.6: Coercing the ChickWeight data frame to a tibble

3.2.2 *Creating New Datasets in R*

Two approaches are typically used to get your data ready for analysis in R. One commonly-used approach is to import your data from an external source. This external data source can be a text or a `.csv` file, an Excel spreadsheet, ACCESS files, or any other database. A second and less common approach is to create your R dataset from scratch within your R script. This approach works best for small datasets, or for larger datasets that can be created programmatically, by executing lines of code.

CREATING TIBBLES BY IMPORTING TEXT FILES

My preferred method for creating a dataset in R from scratch is to first create it in Excel, then save it in the `CSV (Comma delimited)` format[2], before importing from within R. Consider the small Excel dataset of Figure 2.19 of chapter 2, called `employee.xlsx`[3]. I first saved it in the CSV format as `employee.csv`[4]. Now, you can import it from within R by following the menu items `File > Import Dataset > From Text(readr)` as shown in Figure 3.7 and by filling out the dialog form shown in Figure 3.8. Note that the last menu item to select is `From Text(readr)...` instead of `From Text(base)`. It is because `Text(readr)` creates a tibble in R, whereas `Text(base)` creates the traditional data frame that I do not recommend. Long-time R users who are used to traditional data frames, may continue using them. But newcomers to R have a better option in tibbles.

The word `readr` in the menu item `"From Text(readr)..."` is the name of the R package required to use this option. This package is already included in the larger package `tidyverse`. The word `"base"` in `"From Text(base)"` refers to base R, indicating that the use of this option requires no external package.

[2]This format saves your dataset as a smaller text file, and makes it easier to track changes overtime with version control techniques discussed in chapter 10

[3]Here is the download link:https://agreestat.com/books/xls2r/chap2/data/employee.xlsx

[4]Here is the download link: https://agreestat.com/books/xls2r/chap2/data/employee.csv