R15

Importing data

Covered in R15

- Overview
- Using datasets in packages
- Importing data frames from other programs
- Importing data from CSV files
- Importing data from Excel files and elsewhere

1 Overview

In the last chapter we looked at the data frame, which is going to be the standard object used for most data analysis.

In this chapter we look at importing data from various sources into R.

There are three fundamental ways of getting data into R:

- manually
- using datasets included in R's packages
- importing it from another program such as Excel.

We have already covered how to manually enter data into vectors or data frames in previous chapters. So this chapter will cover the other two methods.

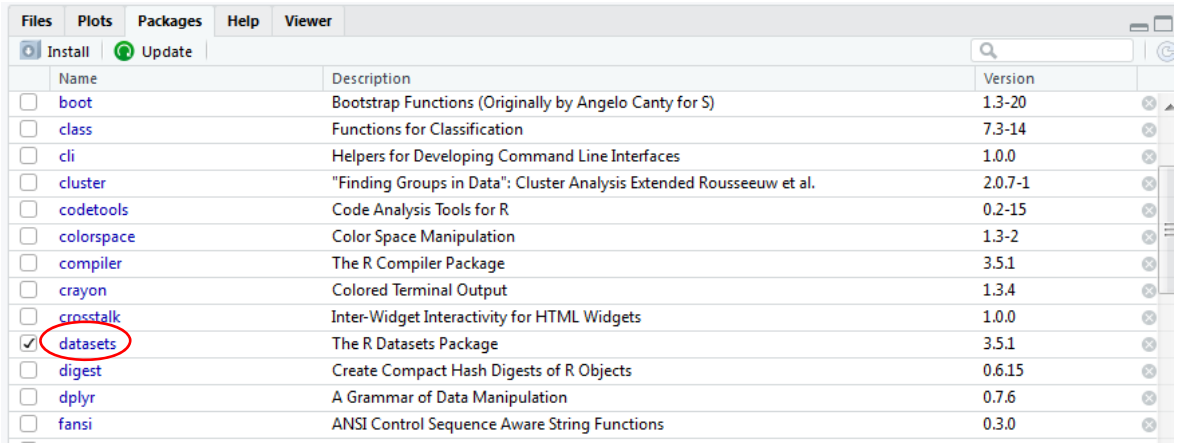
2 Using datasets in packages

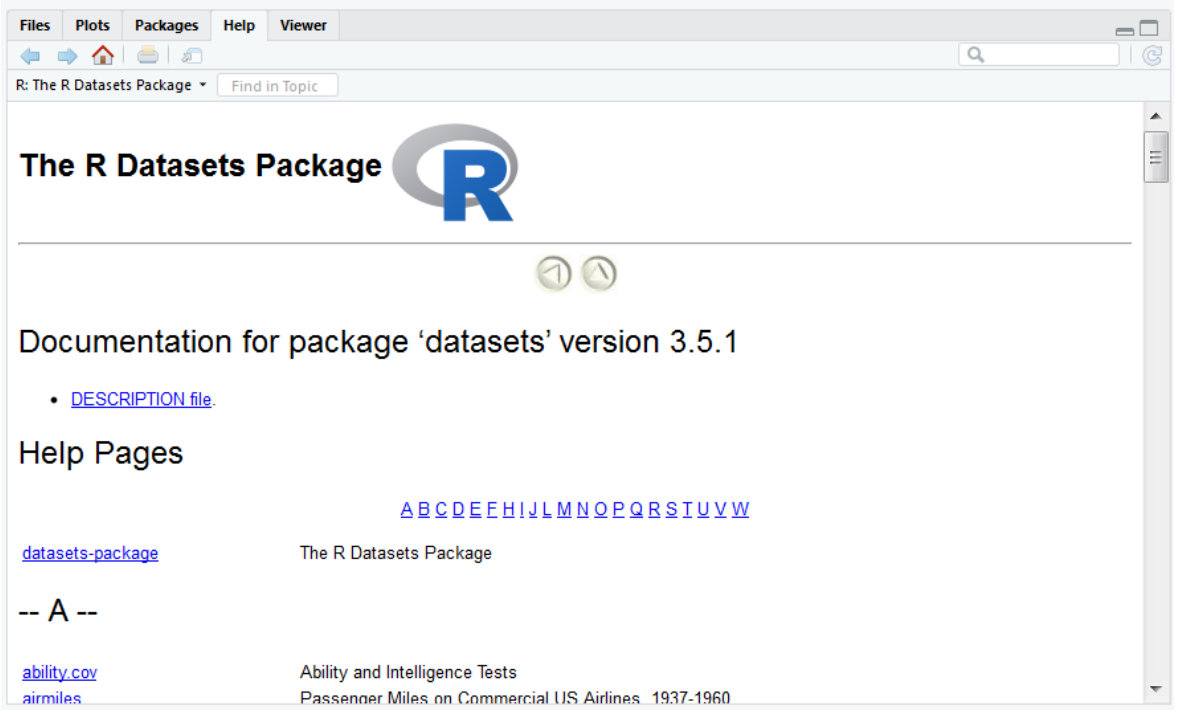
Inbuilt datasets

It may be helpful to review Chapter 8 before reading this section.

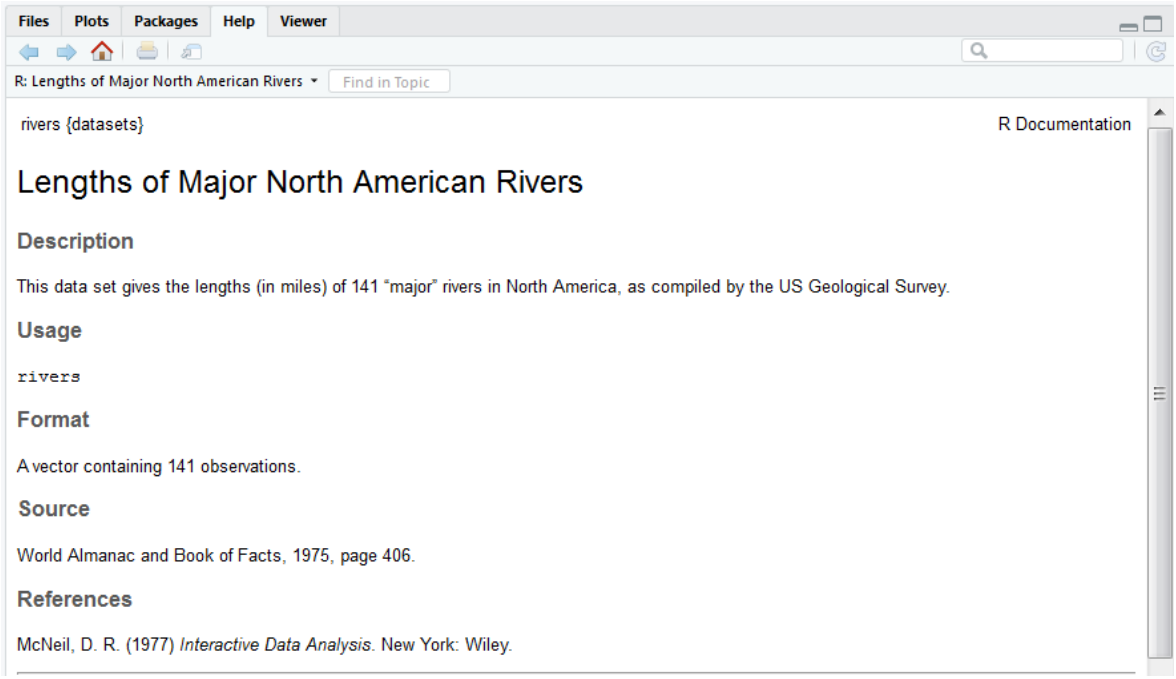
We're going to look at the datasets package which, unsurprisingly, contains a variety of datasets.

To find out more about the contents of this package we can click on its name in the Packages window.





Clicking on the links will take us to the help page on that specific dataset. For example, clicking on rivers gives:



Accessing inbuilt datasets

Since these inbuilt datasets are already loaded into R's workspace we can access them by simply typing their name:

```
> rivers
  [1]  735  320  325  392  524  450 1459  135  465  600  330  336  280  315  870
 [16]  906  202  329  290 1000  600  505 1450  840 1243  890  350  407  286  280
 [31]  525  720  390  250  327  230  265  850  210  630  260  230  360  730  600
 [46]  306  390  420  291  710  340  217  281  352  259  250  470  680  570  350
 [61]  300  560  900  625  332 2348 1171 3710 2315 2533  780  280  410  460  260
 [76]  255  431  350  760  618  338  981 1306  500  696  605  250  411 1054  735
 [91]  233  435  490  310  460  383  375 1270  545  445 1885  380  300  380  377
[106]  425  276  210  800  420  350  360  538 1100 1205  314  237  610  360  540
[121] 1038  424  310  300  444  301  268  620  215  652  900  525  246  360  529
[136]  500  720  270  430  671 1770
>
```

Here R displays the vector of 141 observations (lengths in miles of 141 major rivers in North America).

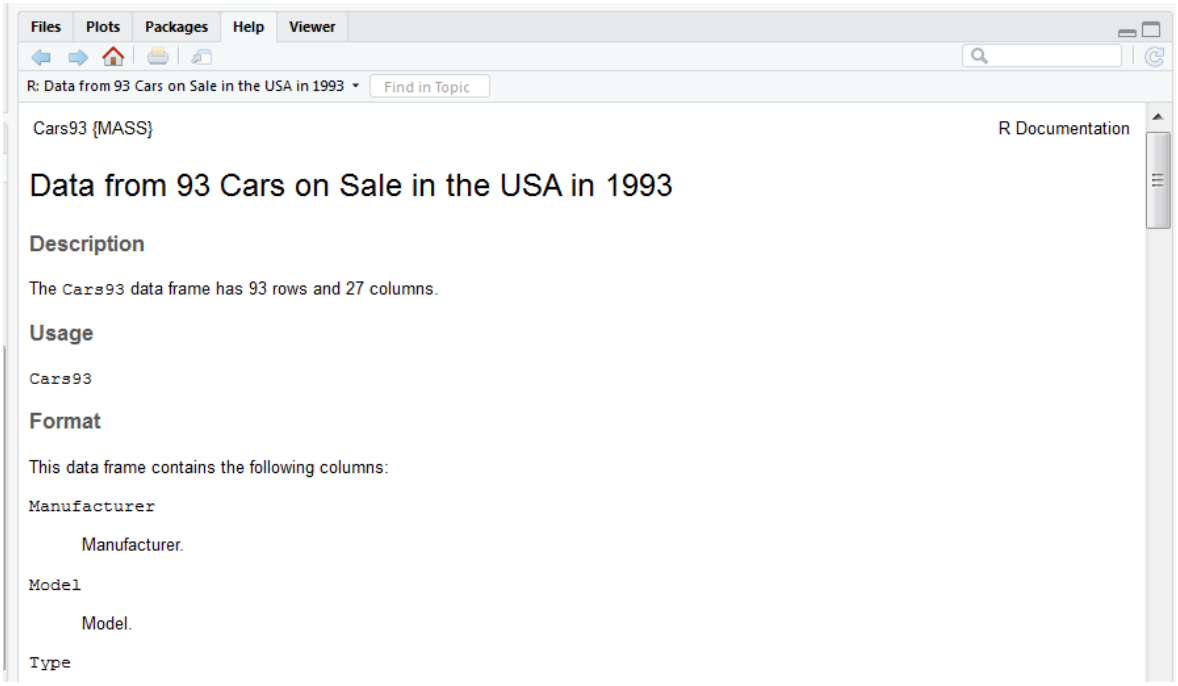
Datasets from other packages

Other packages often contain datasets.

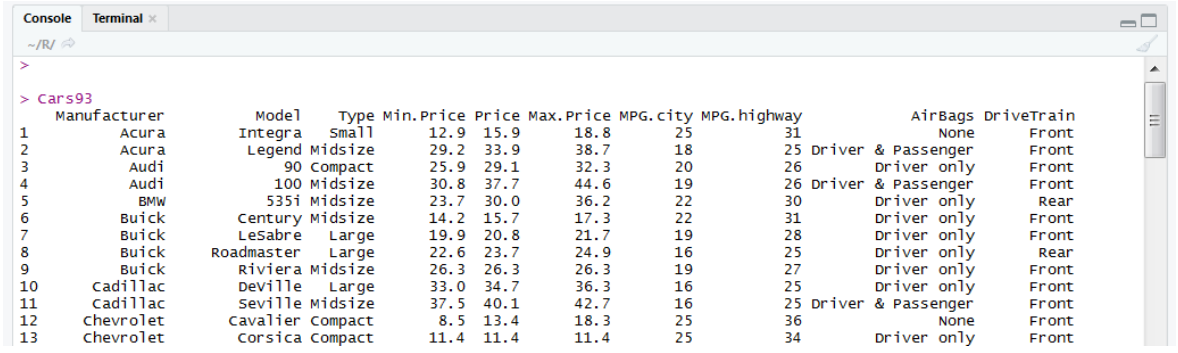
Let's load up the MASS package (using the tick box in the Packages window).

If we use library(help=MASS) or help(package="MASS") we can see that the MASS package contains lots of datasets (87 in fact).

Let's take a look at one of these, Cars93, which contains data from the different types of cars sold in the USA in 1993. From the help page we can see it is a data frame with 93 types of cars (rows) and 27 variables (columns):



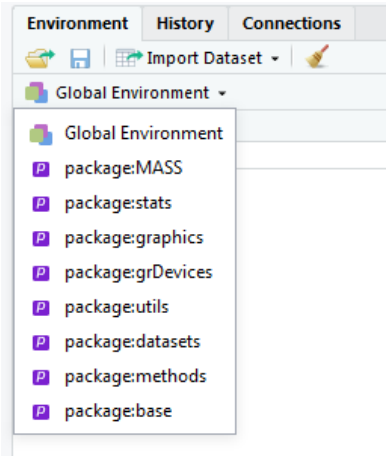
Once we've loaded the package into the workspace we can access the dataset simply by typing its name:



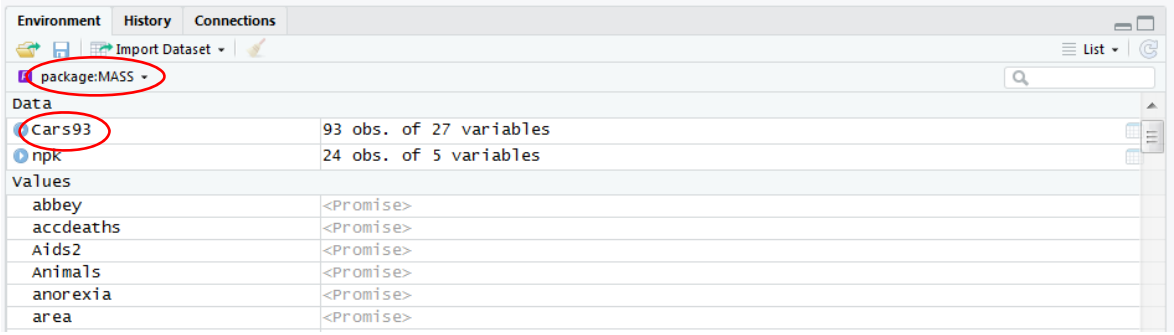
Because of the size of this data frame it is not possible to display it easily in the Console. Using the structure command can tell us about the different variables (columns):

```
Console  Terminal
~/R/
> str(Cars93)
'data.frame':   93 obs. of  27 variables:
 $ Manufacturer      : Factor w/ 32 levels "Acura","Audi",..: 1 1 2 2 3 4 4 4 4 5 ...
 $ Model             : Factor w/ 93 levels "100","190E","240",..: 49 56 9 1 6 24 54 74 73 35 ...
 $ Type              : Factor w/ 6 levels "Compact","Large",..: 4 3 1 3 3 3 2 2 3 2 ...
 $ Min.Price         : num  12.9 29.2 25.9 30.8 23.7 14.2 19.9 22.6 26.3 33 ...
 $ Price             : num  15.9 33.9 29.1 37.7 30 15.7 20.8 23.7 26.3 34.7 ...
 $ Max.Price         : num  18.8 38.7 32.3 44.6 36.2 17.3 21.7 24.9 26.3 36.3 ...
 $ MPG.city          : int  25 18 20 19 22 22 19 16 19 16 ...
 $ MPG.highway       : int  31 25 26 26 30 31 28 25 27 25 ...
 $ AirBags           : Factor w/ 3 levels "Driver & Passenger",..: 3 1 2 1 2 2 2 2 2 2 ...
 $ DriveTrain        : Factor w/ 3 levels "4WD","Front",..: 2 2 2 2 3 2 2 3 2 2 ...
 $ Cylinders         : Factor w/ 6 levels "3","4","5","6",..: 2 4 4 4 2 2 4 4 4 5 ...
 $ EngineSize        : num  1.8 3.2 2.8 2.8 3.5 2.2 3.8 5.7 3.8 4.9 ...
 $ Horsepower        : int  140 200 172 172 208 110 170 180 170 200 ...
```

It's easier to view the dataset by selecting it in the Environment window. First select the package by changing *Global Environment* to *packageMASS* using the dropdown menu:



Then select the dataset:



This will open up a new window displaying the dataset in a grid:


Documentation for package ‘MASS’ Cars93

Filter

	Manufacturer	Model	Type	Min.Price	Price	Max.Price	MPG.city	MPG.highway	AirBags	DriveTrain	Cylinders
1	Acura	Integra	Small	12.9	15.9	18.8	25	31	None	Front	4
2	Acura	Legend	Midsize	29.2	33.9	38.7	18	25	Driver & Passenger	Front	6
3	Audi	90	Compact	25.9	29.1	32.3	20	26	Driver only	Front	6
4	Audi	100	Midsize	30.8	37.7	44.6	19	26	Driver & Passenger	Front	6
5	BMW	535i	Midsize	23.7	30.0	36.2	22	30	Driver only	Rear	4
6	Buick	Century	Midsize	14.2	15.7	17.3	22	31	Driver only	Front	4
7	Buick	LeSabre	Large	19.9	20.8	21.7	19	28	Driver only	Front	6
8	Buick	Roadmaster	Large	22.6	23.7	24.9	16	25	Driver only	Rear	6

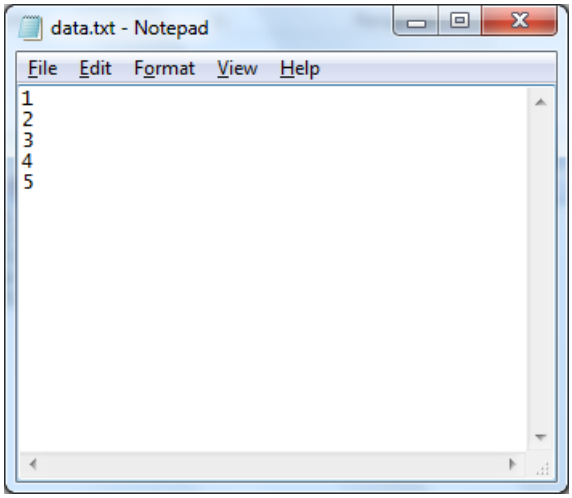
3 Importing data frames from other programs

We can import data from many sources, including:

- text documents
- CSV files
- Excel
- other statistical packages such as SAS or SPSS.

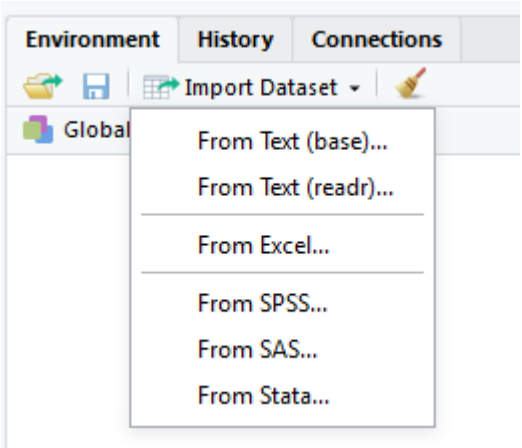
Importing data from text files

Let's create a simple text document. Open up notepad on your machine and enter the numbers 1 to 5 on separate lines. Note that you need to hit return after entering the last number:

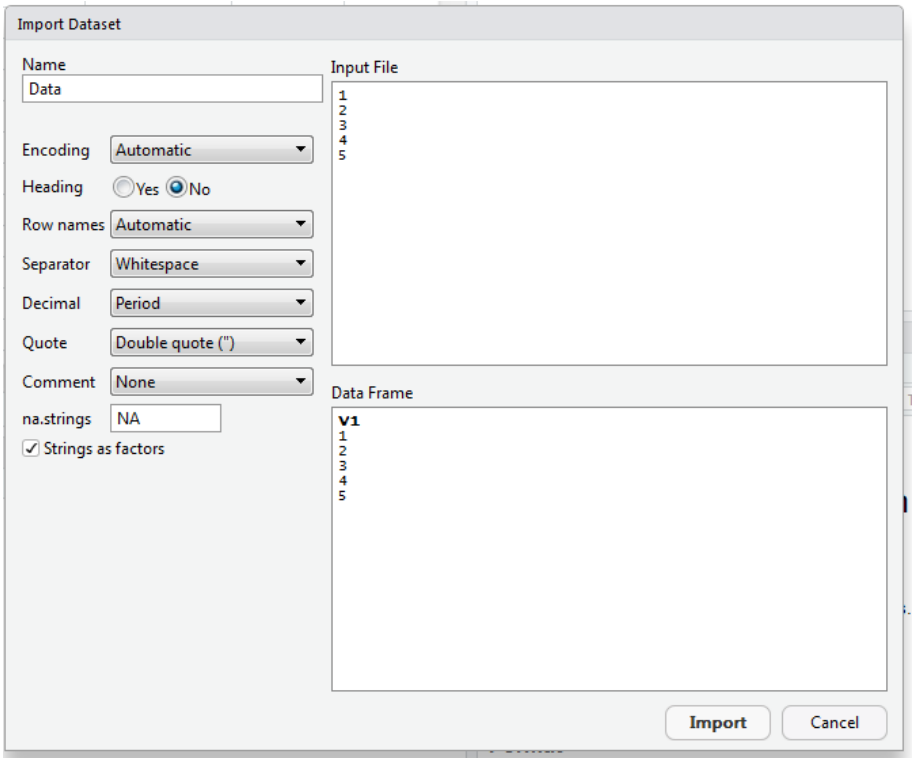


Save this file in your working directory as "Data.txt".

You can import datasets in RStudio using the Environment window. Click on *Import Dataset* and select the first option: *From Text (base)*.

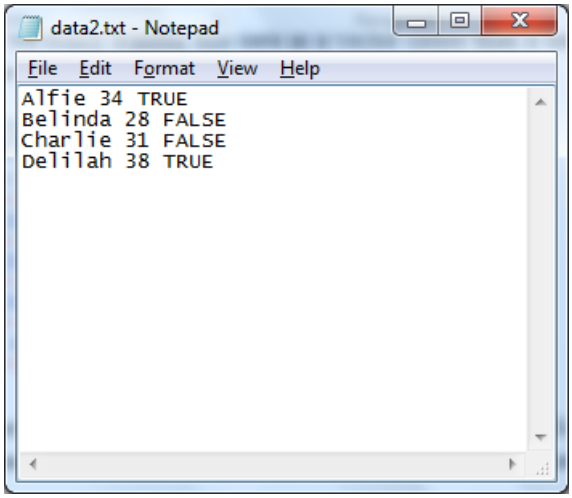


Open the text file you have just created and RStudio will open a window where you can select some options to use for the import (feel free to experiment with the options now or later).

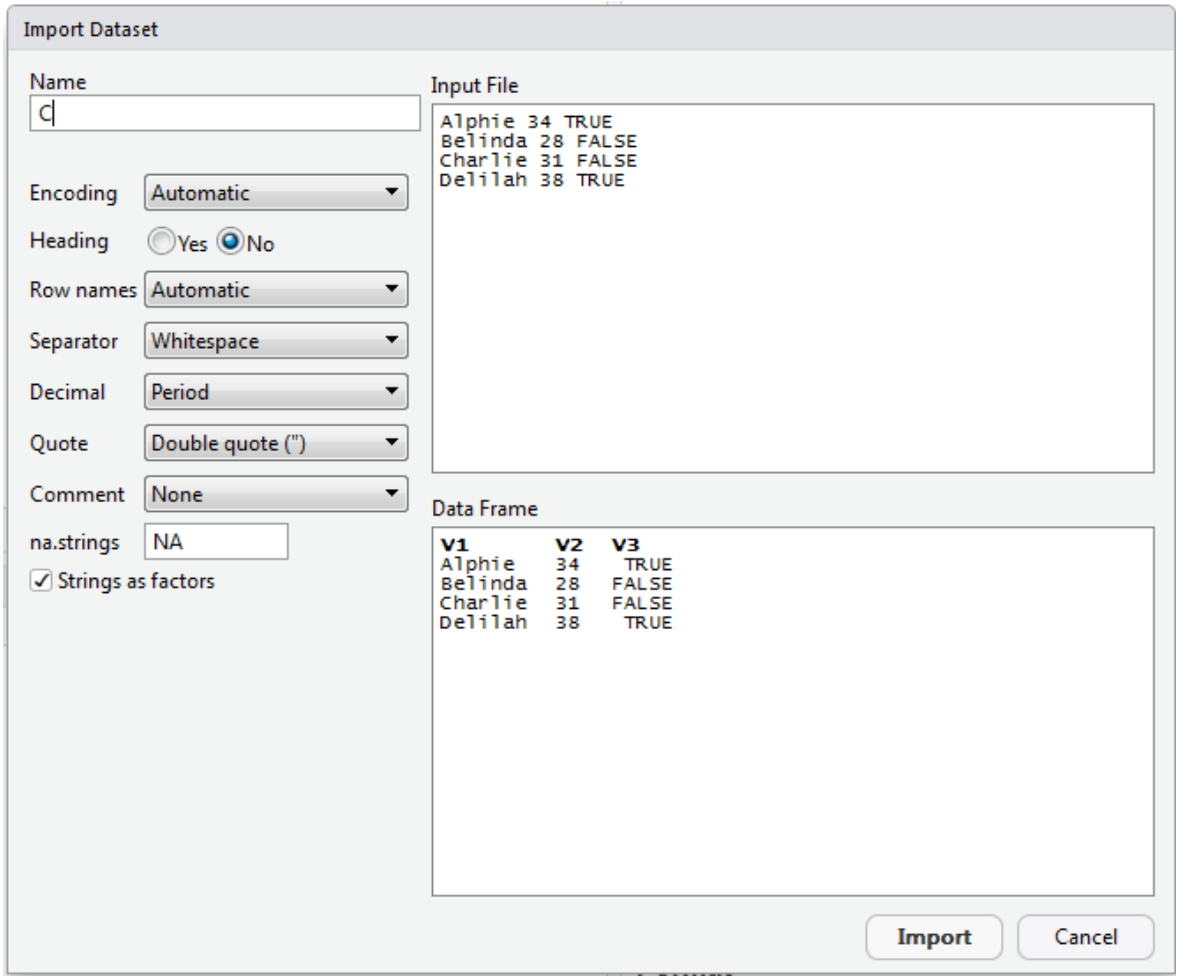


We can see that it has placed a V1 above the numbers. That's because R places the data by default into a data frame and has given the column the name "V1". When you click on *Import* RStudio will display the data in a window and you'll also see the object listed in the Environment window.

Let's now create a three-column dataset in notepad. We can separate the columns with spaces or tabs – both will work. Let's save this dataset as "data2":



We will now load this into object C (by changing the *Name* in the Import window):

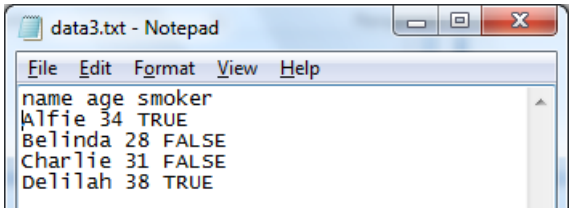


Again, we can see that it is placed in a data frame and by default it has column headings of V followed by the column number.

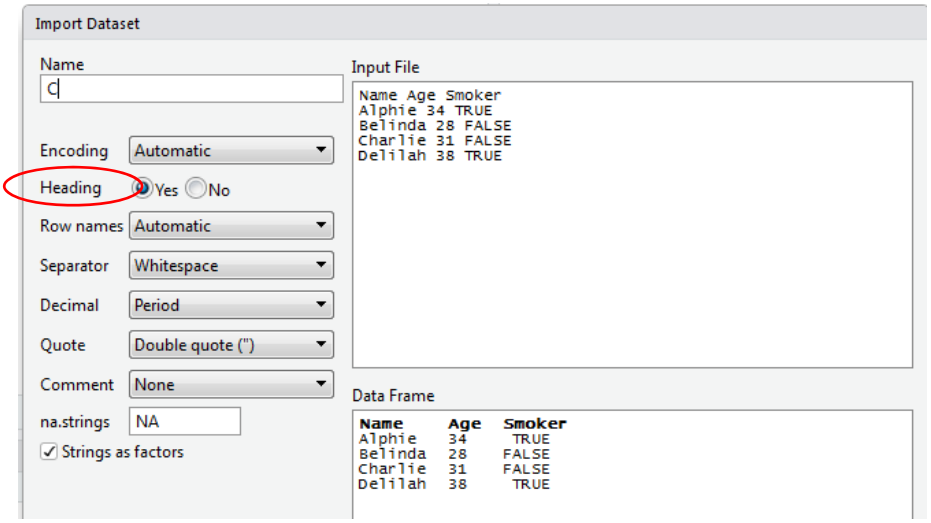
Suppose we want to have column headings “name”, “age” and “smoker”. One way to do this is to use the colnames (or names) function like we did in Chapter 14. For example:

```
colnames(C) <- c("name", "age", "smoker")
```

An alternative way is to add the column names in the original text document (calling this data3.txt):



When we reach the Import window we can use the *Heading* option.



By default the row names are "1", "2", "3", *etc*. We could change these using the rownames function like we did in Chapter 14. Alternatively, we could use the row names option in the window above and use the first column of data.

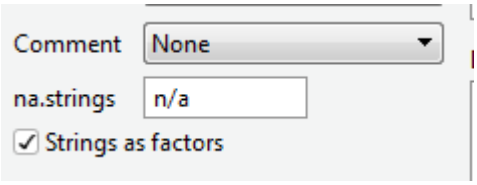
Let's look at the structure of this data frame:

```
> D <- read.table("data3.txt",header=TRUE)
> D
     name age smoker
1   Alfie  34   TRUE
2 Belinda  28  FALSE
3 Charlie  31  FALSE
4 Delilah  38   TRUE
> str(D)
'data.frame':   4 obs. of  3 variables:
 $ name  : Factor w/ 4 levels "Alfie","Belinda",..: 1 2 3 4
 $ age   : int  34 28 31 38
 $ smoker: logi  TRUE FALSE FALSE TRUE
>
```

Just like in the previous chapter, R assumes non-numeric data are factors unless we specify otherwise. We can do this by unchecking the *Strings as Factors* box in the Import window.

Missing data

Missing data is often an issue. R uses the logical value NA to tell its functions that the data is missing. However, our text file might use something else. In which case we need to tell R what it is. We do this via the na.strings option in the Import window. For example, we have specified "n/a" and R will replace any occurrences of this with NA:



If you keep an eye on the Console when you import data you will see that R is using the read.table (or possibly the read-delim) function. For example, the last instruction might have read:

```
Data5 <- read.table("~/R/Data3.txt", quote="\"", comment.char="", na.strings="n/a")
```

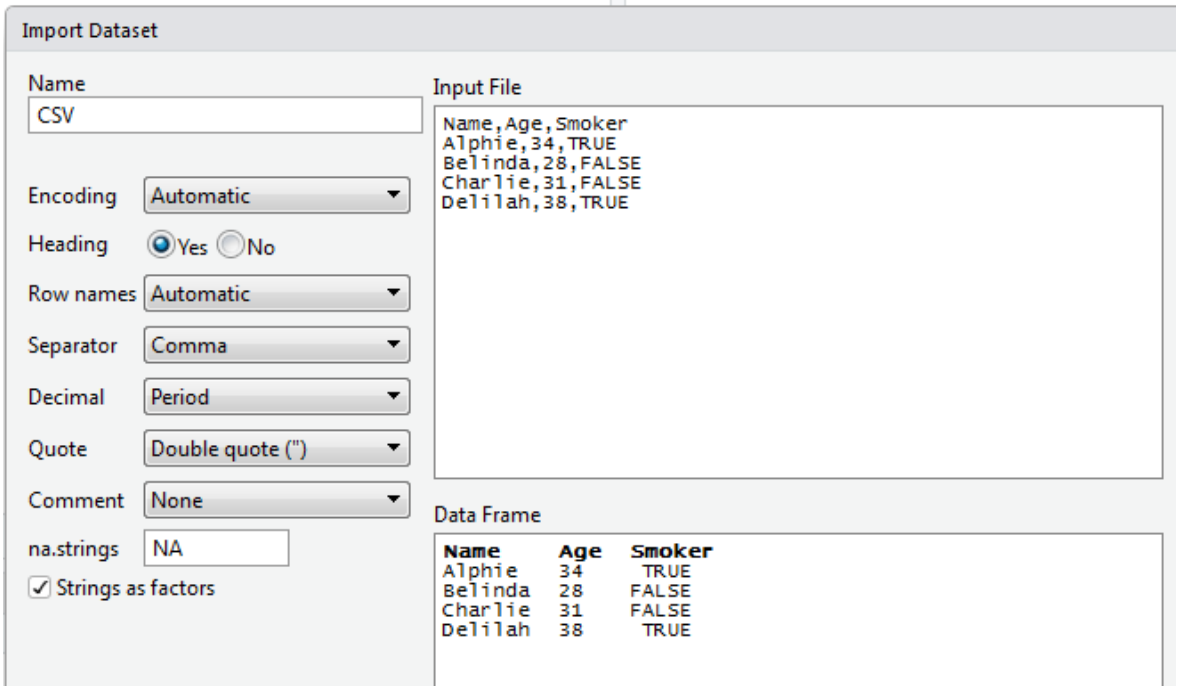
If the dataset contained more one way of indicating missing data, we could adjust this line of code to include the alternatives. For example, here we are looking for "n/a" and "-":

```
Data5 <- read.table("~/R/Data3.txt", quote="\"", comment.char="", na.strings= c("n/a", "-") )
```


4 Importing data from CSV files

CSV stands for Comma Separated Value and is to Excel what NotePad is to Word. Essentially it's a stripped down excel file that removes all the formatting and separates the values in the cells with, unsurprisingly, commas.

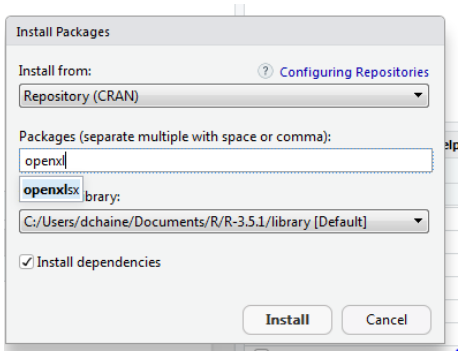
We can import CSV files in the same way as other text files. RStudio will usually recognise the file type and automatically change the *Separator* option to "Comma". For example:

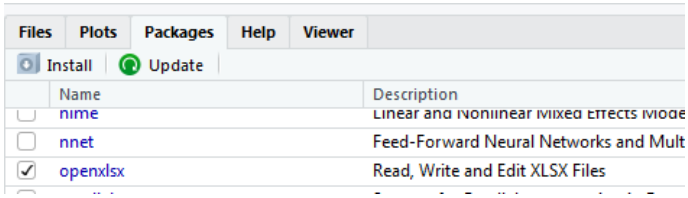


All the other features are the same as before.

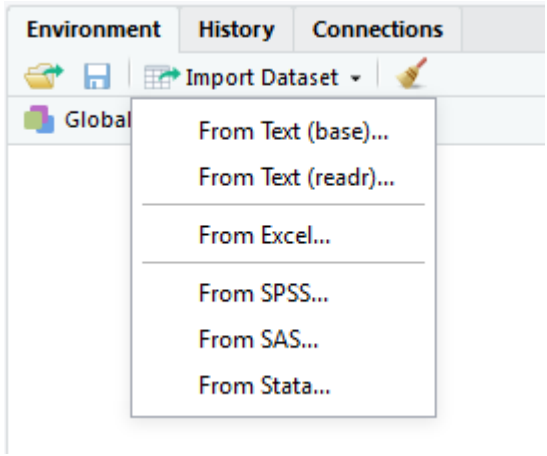
5 Importing data from Excel files and elsewhere

To use RStudio's install menu to install Excel files you first need to download and install a package called "openxlsx":





You can select *From Excel* from the import Dataset drop-down menu:



RStudio will then open a window from where you can open the Excel file and select a number of options. Experiment with the options so you can understand what they do.

Importing data from other statistical packages

It is also possible to import data from other statistical packages such as SPSS, SAS or Stata and you can see options for these in the drop-down menu.

You may be prompted to install or update some packages when you first use these options and if you receive an error then search the internet for some help to find out what you might need to install first.

6 Summary

Key terms

Import	Load data stored elsewhere into R
CSV	Comma Separated Value – a format often used to store data sets, with each field of a record separated by a comma

Menus

Import dataset	Located in the Environment window

Key commands

read.table	Imports data into R. View its help for more information on its options.arguments.

7 Have a go

You will only get proficient at R by practising.

1. Create a dataset in Notepad and import it into R.

2. Create a dataset in Excel, including column and row headings, and import it into R.

3. Alternatively, search the internet for some datasets and load one or two that interest you into R.