Tutorial on loading and viewing data

Introduction

Data Science is all about data. In this tutorial we shall employ technology to load, summarise and visualise data. There are dozens of applications that can do this for you. But, rather than using someone else's software, we are going to use the R programming language to do this for us. In short, we are going to program.

Typically year 2 students would use the RStudio platform to program in R. This is a bit challenging for year 1 students. Instead, we shall use R in a simplified manner. This web page has blocks of code that you can edit. When you are done, you can run each block as a mini-program. The output of each mini-program will appear beneath each block.

As you might expect, R comes with many pre-existing functions that compute common things that we need to use all the time. The meaning of the pre-existing functions should be self-explanatory (though sometimes they are not.) Pre-existing functions are made available to R by installing 'packages' of R code.

If you want to summarise some data, the pre-existing function `summary()` is a good guess. This is from a package called **base**. Notice that functions always have braces at the end. There is where you would pass any parameters needed by the function. For the `summary()` function we need to give it the name of a variable that hold the values. As it happens, `summary()` also operates upon many other types of variable. Unlike some programming languages, variables in R are not normally a single value, they are, in fact, a list of values.

🎯 Goals

In this tutorial we are going to learn how to

- load a CSV file from your local disk
- deal with headers, categorical variable and missing values
- choose an existing dataset
- view its variables
- inspect the first n cases
- count duplicates cases
- do a generic plot of the dataset

using 

 Anatomy of a CSV file

A Comma Separated Variables (CSV) file is a file in which the cases are lines in a text file, and the values are separated by commas. In case, a variable holds some text that includes a comma, it is usual to enclose all text variables in speech marks.

An example of a CSV file contents is:

```
"Address","Postcode","Plot size (Ha)","House size (m2)","Number of bathrooms","Rateable
Value ($1000)","Purchased (YYYY/MM/DD)"
"54 Overtone Drive, West Melton",8093,0.97,645,3,450,"2018/04/04"
"18a Smith Street, Christchurch",8065,1.1,718,2,550,"2003/12/01"
"127 Church Road, Merivale",8054,0.58,549,3,698,"1994/03/19"
"56 Aztec Avenue, Avonhead",8081,0.64,890,2,712,1987
"Flat 2, 6 Dalton Ave, ChCh",8065,0.42,,1,510,"2019/08/15"
"O'Conner farm, RD6, Rakaia",8043,15.7,1204,3,1200,1956
"67 Johns Road, Addington, Christchurch",8065,0.83,876,2,766,"1996/07/25"
"Pines Hotel, Main road, Culverdon",8043,2.1,17040,24,3410,"1999/05/06"
```

Notice that the commas in the addresses (column 1) were not misinterpreted because of the surrounding speech marks.

Please download the file "houseData.csv" from LEARN (https://learn.canterbury.ac.nz/mod/resource/view.php?id=1851232) if you have not already.

 Loading a CSV file

Below is a dialogue that enables you to choose your previously downloaded file that now is local to your computer, probably in the **Downloads** folder. This web page is being generated on a cloud server in a server facility somewhere in the world. We do not want to look for files on that cloud server (in fact we are not allowed to!) We want to look for data on the local machine that you are currently control (even if that is a terminal-server session).

Click on "**Browse…**" and navigate to where the CSV file "houseData.csv" was saved on your local computer.

Once you have successfully selected your file, the file is uploaded from your computer to the cloud server. This takes an amount of time that depends on the size of the size. The file is now temporarily available on the cloud server as:

/tmp/RtmpxHPp78/53a7198eea926e58622ba6e9/0.csv

You need to click the "**Read File**" button below, to start the process of reading the file into an R dataset.

▶ Read File

The structure of the dataset is as follows:

```
'data.frame':   8 obs. of  9 variables:
 $ Address               : Factor w/ 8 levels "127 Church Road, Merivale",..: 3 2 1 4 6 7
 $ Postcode              : int  8093 8065 8054 8081 8065 8043 8065 8043
 $ Plot.size..Ha.        : num  0.97 1.1 0.58 0.64 0.42 15.7 0.83 2.1
 $ House.size...m2.      : int  645 718 549 890 NA 1204 876 17040
 $ Number.of.bathrooms   : int  3 2 3 2 1 3 2 24
 $ Rateable.Value...1000.: int  450 550 698 712 510 1200 766 3410
 $ Purchased..YYYY.MM.DD.: Factor w/ 8 levels "1956","1987",..: 7 6 3 2 8 1 4 5
 $ Exterior.colour       : Factor w/ 6 levels "","brown","green",..: 6 2 1 4 1 5 3 1
 $ Exterior.Cladding     : Factor w/ 4 levels "Block","Brick",..: 4 3 2 2 1 3 4 1
```

The first 12 cases are shown below:

Address	Postcode	Plot.size..Ha.	House.size...m2.	Number.of.bathrooms	Rate
54 Overtone Drive, West Melton	8093	0.97	645	3	
18a Smith Street, Christchurch	8065	1.10	718	2	
127 Church Road, Merivale	8054	0.58	549	3	
56 Aztec Avenue, Avonhead	8081	0.64	890	2	
Flat 2, 6 Dalton Ave, ChCh	8065	0.42	NA	1	
O'Conner farm, RD6, Rakaia	8043	15.70	1204	3	

Address	Postcode	Plot.size..Ha.	House.size...m2.	Number.of.bathrooms	Rate
67 Johns Road, Addington, Christchurch	8065	0.83	876	2	
Pines Hotel, Main road, Culverdon	8043	2.10	17040	24	

Now that you can see the data, you might need to adjust the way the data is loaded. We need to consider the following:

- Is the first row a list of variable names? This is quite commonly the case.
- Are the non-numeric variables to be converted into *factors*? *Factors* are categorical variables in R. We normally want to do this. We might choose not to is we are loading vast amounts of text.
- Are there values that need to be translated in missing values. Missing values are have the label **NA** in R.

▶ What does NA stand for?

Set the controls appropriately (Header = tick, Factors = tick, placeholders = all of the configured ones)

☑ Header

☑ Convert non-numeric to factor variables

Convert these placeholders to NA

NA - -- N/A

Now, go back to the "**Read File**" button and re-read the data correctly this time. Have a look at the output to ensure the data looks reasonable.

Doing this in code

In order to see how to make this happen using using R code a copy of the CSV file has previously been placed in a directory on the server called **images**, the snippet of code below is *almost* ready to run. You will need to supply missing values for the parameters that you used earlier. Set both of the missing values to TRUE.

R-code Start Over Run Code Submit Answer

```
dataset <- read.csv(file = "data/house-data.csv",
                    header = TRUE,
                    stringsAsFactors = TRUE,
                    na.strings = c("NA", "", "-", "--", " ", "N/A"))
str(dataset)
```

```
'data.frame':   8 obs. of  9 variables:
 $ Address               : Factor w/ 8 levels "127 Church Road, Merivale",..: 3 2 1 4 6
7 5 8
 $ Postcode              : int  8093 8065 8054 8081 8065 8043 8065 8043
 $ Plot.size..Ha.        : num  0.97 1.1 0.58 0.64 0.42 15.7 0.83 2.1
 $ House.size...m2.      : int  645 718 549 890 NA 1204 876 17040
 $ Number.of.bathrooms   : int  3 2 3 2 1 3 2 24
 $ Rateable.Value...1000.: int  450 550 698 712 510 1200 766 3410
 $ Purchased..YYYY.MM.DD.: Factor w/ 8 levels "1956","1987",..: 7 6 3 2 8 1 4 5
 $ Exterior.colour       : Factor w/ 5 levels "brown","green",..: 5 1 NA 3 NA 4 2 NA
 $ Exterior.Cladding     : Factor w/ 4 levels "Block","Brick",..: 4 3 2 2 1 3 4 1
```

Excellent!

Using an existing R dataset

In practice, it is usual to load your own data into R, otherwise you will not be able to investigate your own personal data set. There are many datasets already available to you without loading them from a file. Typically, these datasets are ones supplied as aids-to-learning rather than genuine data problems. Having said that, at one time these data sets would have been seen as challenging and the subject of debate and analysis amongst researchers. In this exercise we shall investigate an existing dataset.

Dataset from the carData package

To learn what datasets are available within the **carData** package, we issue a command that reveals what these datasets are called.
By the way, the term “Package” refers to some optional components (data and/or functions) that can be

loaded into our programming environment.

We begin by using the **data()** function. Just about everything we need to do in R, involves functions. All you need to do this time, is click the *Run Code* button.

R-code Start Over Run Code

```
1 data(package = "carData")
2
3
```

```
Ornstein                Interlocking Directorates Among Major Canadian
                        Firms
Pottery                 Chemical Composition of Pottery
Prestige                Prestige of Canadian Occupations
Quartet                 Four Regression Datasets
Robey                   Fertility and Contraception
Rossi                   Rossi et al.'s Criminal Recidivism Data
SLID                    Survey of Labour and Income Dynamics
Sahlins                 Agricultural Production in Mazulu Village
Salaries                Salaries for Professors
Soils                   Soil Compositions of Physical and Chemical
                        Characteristics
States                  Education and Related Statistics for the U.S.
                        States
TitanicSurvival         Survival of Passengers on the Titanic
Transact                Transaction data
UN                      National Statistics from the United Nations,
                        Mostly From 2009-2011
UN98                    United Nations Social Indicators Data 1998]
USPop                   Population of the United States
Vocab                   Vocabulary and Education
WVS                     World Values Surveys
WeightLoss              Weight Loss Data
Wells                   Well Switching in Bangladesh
Womenlf                 Canadian Women's Labour-Force Participation
Wong                    Post-Coma Recovery of IQ
Wool                    Wool data
```

Now we need to pick one of these to examine in detail. We shall choose the “Arrests” data. This dataset has the following characteristics:
Data on police treatment of individuals arrested in Toronto for simple possession of small quantities of marijuana. The data are part of a larger data set featured in a series of articles in the Toronto Star newspaper.

Meaning of variables:

- **released** Whether or not the arrestee was released with a summons; a factor with levels: No; Yes.
- **colour** The arrestee’s race; a factor with levels: Black; White.
- **year** 1997 through 2002; a numeric vector.
- **age** in years; a numeric vector.
- **sex** a factor with levels: Female; Male.

- **employed** a factor with levels: No; Yes.
- **citizen** a factor with levels: No; Yes.
- **checks** Number of police data bases (of previous arrests, previous convictions, parole status, etc. – 6 in all) on which the arrestee's name appeared; a numeric vector

Arrests data

The fifth dataset is the one called **Arrests**." We shall load this data into a programme variable of the same name as the dataset. This code has already been entered into the code box so all you need to do is click the *Run Code* button.

R-code | Start Over | Run Code

```
1 data("Arrests", package = "carData")
2
3
```

In order to see the structure of the data set, we use the `str()` function. Type "str(Arrests)" in the box below and use the *Run code* button.

R-code | Start Over | Hints | Run Code | Submit Answer

```
1 str(Arrests)
2
3
```

```
'data.frame':   5226 obs. of  8 variables:
 $ released: Factor w/ 2 levels "No","Yes": 2 1 2 1 2 2 2 2 2 2 ...
 $ colour  : Factor w/ 2 levels "Black","White": 2 1 2 1 1 1 2 2 1 2 ...
 $ year    : int  2002 1999 2000 2000 1999 1998 1999 1998 2000 2001 ...
 $ age     : int  21 17 24 46 27 16 40 34 23 30 ...
 $ sex     : Factor w/ 2 levels "Female","Male": 2 2 2 2 1 1 2 1 2 2 ...
 $ employed: Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 1 2 2 2 ...
 $ citizen : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 2 2 2 ...
 $ checks  : int  3 3 3 1 1 0 0 1 4 3 ...
```

Out of this world!

R code is case sensitive.
You must follow the instructions exactly.

Quiz

Which *Arrests* variables are categorical? Bear in mind that R uses the term *factor* for categorical variables

✓ released
✓ color
✗ year
✗ age
✓ sex
✓ employed
✓ citizen
✗ checks

Correct!

Which *Arrests* variables are quantitative? R uses the term *integer* or *numeric* for quantitaive variables

✗ released
✗ color
✓ year
✓ age
✗ sex
✗ employed
✗ citizen
✓ checks

Correct!

How many cases are there? R uses the term *observations* for cases

✗ 5192
✗ 8
✗ 2347
✓ 5226

Correct!

How many variables are there?

✗ 5226

✗ 7

✗ 9

✓ 8

Correct!

Data table Summary

An alternative approach is to utilise the *summarytools* package. This displays a lot more information in a browser friendly fashion. Notice that this code snippet uses %>% operators that chain the output of one command into the next command. The functions `dfSummary()` and `view()` are both from the *summarytools* package

Notice below, that the code uses `%>%` to chain the output of each command into the next command. The code below is ready to run. Press press the *Run Code* button.

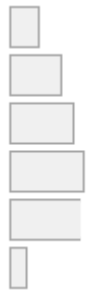
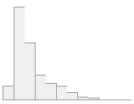
R-code Start Over Run Code

```
1 library(summarytools)
2 Arrests %>%
3   dfSummary(col.widths = c(10,80,150,120,120,180,220)) %>%
4   view(method = "render")
```

Data Frame Summary

Arrests

Dimensions: 5226 x 8
Duplicates: 2347

No	Variable	Stats / Values	Freqs (% of Valid)	Graph	Valid	Missing
1	released [factor]	1. No 2. Yes	892 (17.1%) 4334 (82.9%)		5226 (100.0%)	0 (0.0%)
2	colour [factor]	1. Black 2. White	1288 (24.6%) 3938 (75.4%)		5226 (100.0%)	0 (0.0%)
3	year [integer]	Mean (sd) : 1999.5 (1.4) min ≤ med ≤ max: 1997 ≤ 2000 ≤ 2002 IQR (CV) : 3 (0)	1997 : 492 (9.4%) 1998 : 877 (16.8%) 1999 : 1099 (21.0%) 2000 : 1270 (24.3%) 2001 : 1211 (23.2%) 2002 : 277 (5.3%)		5226 (100.0%)	0 (0.0%)
4	age [integer]	Mean (sd) : 23.8 (8.3) min ≤ med ≤ max: 12 ≤ 21 ≤ 66 IQR (CV) : 9 (0.3)	53 distinct values		5226 (100.0%)	0 (0.0%)
5	sex [factor]	1. Female 2. Male	443 (8.5%) 4783 (91.5%)		5226 (100.0%)	0 (0.0%)
6	employed [factor]	1. No 2. Yes	1115 (21.3%) 4111 (78.7%)		5226 (100.0%)	0 (0.0%)

No	Variable	Stats / Values	Freqs (% of Valid)	Graph	Valid	Missing
7	citizen [factor]	1. No 2. Yes	771 (14.8%) 4455 (85.2%)		5226 (100.0%)	0 (0.0%)
8	checks [integer]	Mean (sd) : 1.6 (1.5) min ≤ med ≤ max: 0 ≤ 1 ≤ 6 IQR (CV) : 3 (0.9)	0 : 1851 (35.4%) 1 : 854 (16.3%) 2 : 789 (15.1%) 3 : 953 (18.2%) 4 : 643 (12.3%) 5 : 127 (2.4%) 6 : 9 (0.2%)		5226 (100.0%)	0 (0.0%)

Generated by summarytools (https://github.com/dcomtois/summarytools) 1.0.1 (R (https://www.r-project.org/) version 4.2.1)
2022-07-20

Quiz

Which proportion of variable *released* are Yes?

✗ 0.171
✗ 0.754
✗ 0.246
✓ 0.829

Correct!

Which of *Arrests*' variables are continuous?

✗ All of them
✓ None of them
✗ Some of them

Correct!

Which of *Arrests*' variables are discrete?

✗ Some of them

✗ None of them

✓ All of them

Correct!

View the first n cases

To view a portion of the dataset, you can print the top n cases using the **head()** function. The function has a second parameter called *n* which determines the number of cases to reveal. Set this to 12. Use the *Hints* button if you are unsure.

R-code | Start Over | Hints | Run Code | Submit Answer

```
1 head(Arrests, n = 12)
2
3
```

```
   released colour year age    sex employed citizen checks
1       Yes  White 2002  21   Male      Yes     Yes      3
2        No  Black 1999  17   Male      Yes     Yes      3
3       Yes  White 2000  24   Male      Yes     Yes      3
4        No  Black 2000  46   Male      Yes     Yes      1
5       Yes  Black 1999  27 Female      Yes     Yes      1
6       Yes  Black 1998  16 Female      Yes     Yes      0
7       Yes  White 1999  40   Male       No     Yes      0
8       Yes  White 1998  34 Female      Yes     Yes      1
9       Yes  Black 2000  23   Male      Yes     Yes      4
10      Yes  White 2001  30   Male      Yes     Yes      3
11      Yes  White 1999  18   Male      Yes      No      0
12      Yes  White 2000  18   Male      Yes     Yes      3
```

Excellent!

View the 6-number-summary

The 6 number summary is the same as the 5-number summary with the addition of the **mean**. Only a quantitative variable can be handled this way. The categorical variables are summarised in terms of the category frequencies.

To view a 6-number summary of each variable of the dataset, you can use the **summary()** function. Use the *Hints* button if you are unsure.

R-code | Start Over | Hints | Run Code | Submit Answer

```
1 summary(Arrests)
```

```
 released     colour          year           age           sex
 No : 892   Black:1288   Min.   :1997   Min.   :12.00   Female: 443
 Yes:4334   White:3938   1st Qu.:1998   1st Qu.:18.00   Male  :4783
                         Median :2000   Median :21.00
                         Mean   :2000   Mean   :23.85
                         3rd Qu.:2001   3rd Qu.:27.00
                         Max.   :2002   Max.   :66.00
 employed   citizen        checks
 No :1115   No : 771   Min.   :0.000
 Yes:4111   Yes:4455   1st Qu.:0.000
                       Median :1.000
                       Mean   :1.636
                       3rd Qu.:3.000
                       Max.   :6.000
```

Smashing!

These 6-number-summaries give the same 6 statistics that the *dfSummary()* function produced. Because the format is simpler, this style of summary is a bit easier to comprehend.

Quiz

What is the value of the median *Age* variable?

✗ 12

✓ 21

✗ There is no median; the variable is categorical

✗ 27

✗ 23.85

✗ 18

Correct!

What is the value of the inter-quartile range (IQR) of the *checks* variable?

✓ 3

✗ 2

✗ 6

✗ 0.636

✗ There is no IQR; the variable is categorical

Correct!

Is the *checks* variable distribution skewed? You can deduce this from the 5/6-number-summary

✗ No

✗ There is no skew; the variable is categorical

✓ Yes - right skewed

✗ Yes - left skewed

Correct!

What is the cardinality of the *released* variable? How many categories does it have?

✗ 5226

✗ 2347

✗ There is no cardinality; the variable is quantitative

✓ 2

✗ 8

✗ 1

Correct!

What is the cardinality of the *year* variable? How many distinct values does it have?

✓ 6

✗ 2

✗ 2347

✗ 8

✗ There is no cardinality; the variable is quantitative

✗ 5226

Correct!

Duplicates

To count the number of duplicate cases, we can use a pair of functions and chain them together. **duplicated()** will determine if a case is seen before, and **sum()** will total these up to a count. Type these two function names into the the gaps in the code snippet below.

R-code | Start Over | Hint | Run Code | Submit Answer

```
1 Arrests %>%
2   duplicated() %>%
3   sum()
```

```
[1] 2347
```

What first-rate work!

This number of duplicated cases agrees with the *dfSummary* output. Go back and check if you are not sure where this was displayed earlier.

Is this a problem? Is it reasonable to expect duplicates? Has someone made a mistake in curating the dataset?

For data with no case identifiers and the variables being *all* discrete, duplicates can be expected. The cases are likely to be different individuals but occasionally resembling a earlier case. There is no cause for alarm.

Why do you think the case identifiers have not been supplied?

The cases are people. For reasons of privacy the names of the people have been deliberately removed.

Add a implied case identifier

Since the original case identifiers have been lost *and* we are confident that the cases are *not* accidental repetitions, we can assign an arbitrary label for the cases. We shall compose an identifier out of a concatenation of *"person" + "-" + row number*

R-code Start Over Run Code

```
Arrests %>%
  dplyr::mutate(id = paste("Person",1:nrow(.), sep="-")) %>%
  head(n=12)
```

```
   released colour year age    sex employed citizen checks        id
1       Yes  White 2002  21   Male      Yes     Yes      3  Person-1
2        No  Black 1999  17   Male      Yes     Yes      3  Person-2
3       Yes  White 2000  24   Male      Yes     Yes      3  Person-3
4        No  Black 2000  46   Male      Yes     Yes      1  Person-4
5       Yes  Black 1999  27 Female      Yes     Yes      1  Person-5
6       Yes  Black 1998  16 Female      Yes     Yes      0  Person-6
7       Yes  White 1999  40   Male       No     Yes      0  Person-7
8       Yes  White 1998  34 Female      Yes     Yes      1  Person-8
9       Yes  Black 2000  23   Male      Yes     Yes      4  Person-9
10      Yes  White 2001  30   Male      Yes     Yes      3 Person-10
11      Yes  White 1999  18   Male      Yes      No      0 Person-11
12      Yes  White 2000  18   Male      Yes     Yes      3 Person-12
```

MISSING Missing values

The code snippet below, that you must complete, will check whether each value of each variable is missing. The total count of missing values will be calculated.

To count the number of missing cases, we can use a pair of functions and chain them together. **is.na()** will determine if a case is missing, and **sum()** will total these up to a count. Type these two function names into the the gaps in the code snippet below.

R-code Start Over Hint Run Code Submit Answer

```
Arrests %>%
  is.na() %>%
  sum()
```

```
[1] 0
```

Spectacular job!

It would seem these are no missing values in this dataset. Most real-life datasets have missing data for all sorts of reasons.

Wrap-up

Using any of the previous outputs, answer the following quiz questions to show your understanding of what the data tells us.

Quiz

Based on the latest change we made to the *Arrests* data, which of the following statements are correct?

✓ The variable names are unique
✓ There is a variable that has the case-identifier role
✓ The case-identifiers are unique
✓ The data-types of values in each particular variable are all the same
✗ There are variables that are constant

Correct!

Is the *Arrests* data table well-structured?

✗ Oh no, its not
✓ Yes, it is

Correct!

How many variables does *Arrests* have at the end?

✗ 8 : like it started with
✓ 9

Correct!

How many cases does *Arrests* have at the end?

✗ 2347

✗ 2879

✓ 5226 : like it started with

Correct!

Code

In this tutorial you have been exposed to the following R functions:

Function	Package	Description
`read.csv()`	utils	read a CSV file
`str()`	base	structure
`data()`	base	load built-in data
`head()`	base	the first few cases
`duplicated()`	base	whether case is repeated
`summary()`	base	summarise
`dfSummary()`	summarytools	data frame summary

In case this on-line tutorial is not available in future, you may want to keep a PDF copy of this material for reference purposes.

Export as PDF

Topic not yet completed...