Miscellaneous

COMM 205 - Lecture 23 - R8

Hasan Cavusoglu

2020

Agenda

- Converting data types
  - from numeric to character
  - from character to numeric
- Loading a dataset from a file
  - `readRDS()`
  - `read_csv()`
- Exporting data to a file
  - `saveRDS()`
  - `write_csv()`

Converting the data type from *numeric* to *character*

- `as.character()` is used to convert a *numerical* object into a *character* object.

as.character() Syntax

as.character(R_object)
where **R_object** can an atomic object or a vector.

- It works on an atomic object as well as vectors

```
a <- 3
new1 <- as.character(a)
```

- You can *verify* the type with either `typeof()` or `is.character()`.

```
> typeof(new1) #  asking the type of the object
[1] "character"
> is.character(new1)  # asking if the object is character
[1] TRUE
```

```
> new2 <- as.character(c(1,2,3))
> is.character(new2)
[1] TRUE
```

Example

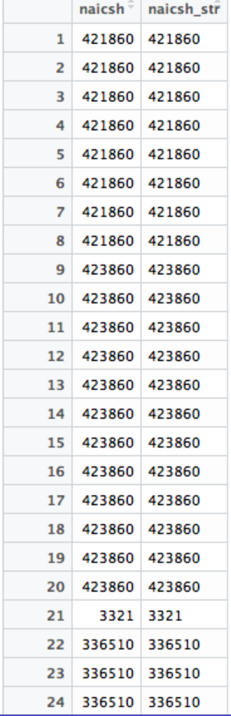
Question

Suppose we want to convert the variable **naicsh** (a numeric column) into a character colunmn. We want to preserve the original **naicsh**, and want R to create a separate column that contains the exact same values of **naicsh** for every single observation in our dataset, just with a different variable type. Let's call this new variable **naicsh_str**. Only keep those two columns.

Let's load `tidyverse` and our North American Stock Market 1994-2013 Dataset and name it as `companies`.

```
library(tidyverse)
```

```
df1 <- companies %>%
  mutate(naicsh_str = as.character(naicsh)) %>%
  select(naicsh, naicsh_str)
```



	naicsh	naicsh_str
1	421860	421860
2	421860	421860
3	421860	421860
4	421860	421860
5	421860	421860
6	421860	421860
7	421860	421860
8	421860	421860
9	423860	423860
10	423860	423860
11	423860	423860
12	423860	423860
13	423860	423860
14	423860	423860
15	423860	423860
16	423860	423860
17	423860	423860
18	423860	423860
19	423860	423860
20	423860	423860
21	3321	3321
22	336510	336510
23	336510	336510
24	336510	336510

- You can see that the left column is numeric and the right column is character.
- The numeric column is right-aligned while the character object is left-aligned (see row 21)

Converting the data type from character to numeric

- `as.numeric()` is used to convert a character object into numeric object.

as.numeric() Syntax

as.numeric(R_object)
where **R_object** can an atomic object or a vector.

- It works on an atomic object or a vector

```
james <- "007"
new3 <- as.numeric(james)
```

- You can *verify* the type with either `typeof()` or `is.numeric()`.

```
> typeof(new3)
[1] "double"
> is.numeric(new3)
[1] TRUE
```

```
> new4 <- as.numeric(c("1","2","3"))
> is.numeric(new4)
[1] TRUE
```

When character object does not contain numeric value

- Please note that if `character` object does not contain a number, `as.numeric()` will produce `NA`.
- Here is a simple illustration:

```
e <- c("a", "3")
new5 <- as.numeric(e)
```

```
## Warning: NAs introduced by coercion
```

- As you can see above, R warns you that `NAs introduced by coercion`. If you just type `new5` at the console, you should see `NA` introduced for the element for which R could not convert the value into a numeric value.

```
new5
```

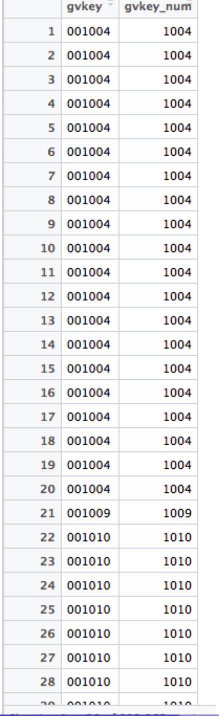
```
## [1] NA  3
```

Example

Question

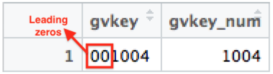
Suppose we want to convert the column **gvkey** (a **character** vector) into a **numeric** column We want to preserve the original **gvkey**, and want R to create a separate variable that contains the exact same values of **gvkey** for every single observations in our dataset, just with a different variable type. Let's call this new variable **gvkey_num**.

```r
df2 <- companies %>%
  mutate(gvkey_num = as.numeric(gvkey)) %>%
  select(gvkey, gvkey_num)
```



	gvkey	gvkey_num
1	001004	1004
2	001004	1004
3	001004	1004
4	001004	1004
5	001004	1004
6	001004	1004
7	001004	1004
8	001004	1004
9	001004	1004
10	001004	1004
11	001004	1004
12	001004	1004
13	001004	1004
14	001004	1004
15	001004	1004
16	001004	1004
17	001004	1004
18	001004	1004
19	001004	1004
20	001004	1004
21	001009	1009
22	001010	1010
23	001010	1010
24	001010	1010
25	001010	1010
26	001010	1010
27	001010	1010
28	001010	1010

- You verify that **gvkey** is a character variable, while **gvkey_num** is a **numeric** variable.
- Remember that **numeric** values are right-aligned.
- Note that **leading zeros** (i.e., any 0 digit that comes before the first nonzero digit in the character value) are lost.


Leading zeros

	gvkey	gvkey_num
1	001004	1004

Importing a dataset to R

- Data can be imported to R from various different file formats. We will cover two formats:
  - RDS
  - CSV

RDS files

- RDS is R's custom binary format.
- This is the format in which not only data but also the data types are preserved.
- Basic Syntax:

```r
readRDS("path to RDS file")
```

- You can also load RDS file via RStudio's GUI.

CSV files

- CSV stands for “comma-separated values”. A CSV file stores tabular data in plain text. Each line of the file is an observation whose values for different columns are separated by commas.
- We use `read_csv()` from `readr` which comes with `tidyverse`.
- Basic syntax:

```
read_csv("path to CSV file / url")
```

- You can also use Studio’s GUI.

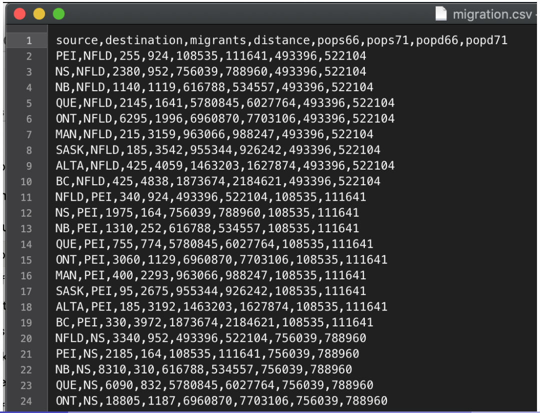
Load data from a csv file via RStudio’s GUI

- Click on the three dots on the Files Pane
- Locate the **folder** in which the file resides
- Click on the file > Import Dataset . . . > Import

Example

Question

Let's load data from a file **migration.csv** under my working directory and create a **migration** data frame. This is a csv version of *Canadian Interprovincial Migration Data* from **car** package. Each observation is source and destination provinces pair. Details of the dataset can be found here.



```
source,destination,migrants,distance,pops66,pops71,popd66,popd71
PEI,NFLD,255,924,108535,111641,493396,522104
NS,NFLD,2380,952,756039,788960,493396,522104
NB,NFLD,1140,1119,616788,534557,493396,522104
QUE,NFLD,2145,1641,5780845,6027764,493396,522104
ONT,NFLD,6295,1996,6960870,7703106,493396,522104
MAN,NFLD,215,3159,963066,988247,493396,522104
SASK,NFLD,185,3542,955344,926242,493396,522104
ALTA,NFLD,425,4059,1463203,1627874,493396,522104
BC,NFLD,425,4838,1873674,2184621,493396,522104
NFLD,PEI,340,924,493396,522104,108535,111641
NS,PEI,1975,164,756039,788960,108535,111641
NB,PEI,1310,252,616788,534557,108535,111641
QUE,PEI,755,774,5780845,6027764,108535,111641
ONT,PEI,3060,1129,6960870,7703106,108535,111641
MAN,PEI,400,2293,963066,988247,108535,111641
SASK,PEI,95,2675,955344,926242,108535,111641
ALTA,PEI,185,3192,1463203,1627874,108535,111641
BC,PEI,330,3972,1873674,2184621,108535,111641
NFLD,NS,3340,952,493396,522104,756039,788960
PEI,NS,2185,164,108535,111641,756039,788960
NB,NS,8310,310,616788,534557,756039,788960
QUE,NS,6090,832,5780845,6027764,756039,788960
ONT,NS,18805,1187,6960870,7703106,756039,788960
```

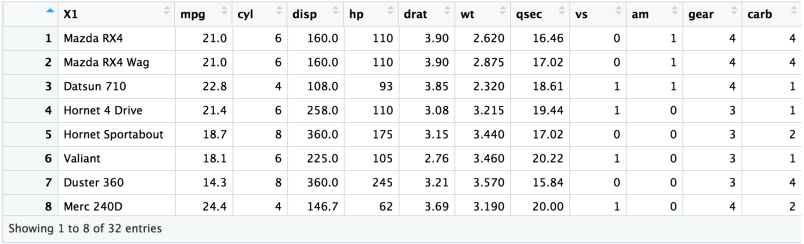
Reading from a URL

- You could also read a csv file on the Internet by passing its url address to `read_csv()`.

Question

mtcars is a dataset from 1974 Motor Trend US magazine. It can be access at https://raw.githubusercontent.com/vincentarelbundock/Rdatasets/master/csv/data
Read the csv file from the Internet and create **car_data** data frame.

```
car_data <-
  read_csv("https://raw.githubusercontent.com/vincentarelbundock/Rdatasets/mast
```



	X1	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
1	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
2	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
3	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
4	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
5	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
6	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
7	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
8	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2

Showing 1 to 8 of 32 entries

Exporting a dataset from R

Data can be exported from R into various different file formats.

RDS file

- You can save an R data object into an RDS file, R's custom binary format.
- Basic Syntax:

saveRDS(data object,"file name")

where **data object** is an R data object, such as data frame; **"file name"** file name.

Question

Suppose you are asked to save the **gvkey** and **tic** of the firms headquartered in **CAN** in **2010** in a file called **gvkey_tic_lookup.rds** (under your working directory).

```r
gvkey_tic_CAN_2010 <- companies %>%
  filter(loc == "CAN", fyear == 2010) %>%
  select(gvkey, tic)
saveRDS(gvkey_tic_CAN_2010,"gvkey_tic_lookup.rds")
```

You can confirm that the file has been created under the current working directory by navigating File Explorer in Windows/ Finder in Mac.

Exporting to a CSV file

CSV file

- **write_csv()** from **readr** package that comes with **tidyverse** can be used to write data to csv files.
- The basic syntax is

write_csv(data_name,"file name")
where **data_name** is a data frame and **filename** is a csv file name

Question

Suppose you want to save **gvkey_tic_CAN_2010** data frame in to a file called **gvkey_tic_lookup.csv** (will reside in your **current working directory**).

```
write_csv(gvkey_tic_CAN_2010,"gvkey_tic_lookup.csv")
```

You can confirm that the file has been created by navigating File Explorer in Windows/ Finder in Mac.

IN CLOSING

- Thank you for being such a **terrific** class!

Keep in touch

- Contact Info
  - email: cavusoglu@sauder.ubc.ca
  - phone: 604-822-8894
  - office: HA 379
- I would be happy to be connected to you in LinkedIn. Please send an invite:
  - Profile: https://www.linkedin.com/in/hasan-cavusoglu/

The End

Thanks for watching

See you in next time!