Mastering R by Examples

Avan Suinesiaputra

Table of contents

About

This is a cookbook recipes for R using data directly. There is no structure in this book. All chapters are independent and can be executed directly. Use these recipe to quickly learn to solving data analytics with R.

1 Reading CSV file

R built-in function to load a CSV file is the `read.csv()` function, but I prefer to use `read_csv()` function from `tidyverse` package. It's more versatile and you have more controls over correctness of the data.

```
library(tidyverse)
```

This example demonstrates how to read a CSV file, correcting some column data types, and creating new categorical variables.

The data is taken from echocardiographic exams, but we only select some variables and cases to reduce the dimension for the sake of the explanation of this example.

1.1 Without correction

```
read_csv('../NEDA/Version_03062024_SurvivalAllCases/Aortopathy_ECHO_Progression_03_VALID.c
         show_col_types = FALSE) %>%
  select(StudyID, PatientID, Sex, Age_At_Echo, Examination_Date, Outcome, SoV, STJ, AscAo)
  sample_n(10000) %>%
  glimpse()
```

```
Rows: 10,000
Columns: 9
$ StudyID          <dbl> 779077, 1670072, 2029178, 917043, 833855, 1717473, 19~
$ PatientID        <dbl> 455277, 385496, 784046, 229541, 531403, 452658, 71343~
$ Sex              <chr> "Female", "Male", "Male", "Female", "Male", "Male", "~
$ Age_At_Echo      <dbl> 43, 42, 44, 79, 65, 79, 85, 24, 19, 70, 65, 68, 90, 8~
$ Examination_Date <date> 2006-10-11, 2016-06-13, 2009-03-31, 2007-08-13, 2015~
$ Outcome          <chr> "Alive", "Alive", "Alive", "Dead", "Alive", "Alive", ~
$ SoV              <dbl> 3.021, 3.577, 3.100, 3.000, 4.324, 3.600, 4.307, 2.70~
$ STJ              <dbl> NA, 3.20, NA, 2.90, 3.90, 3.00, NA, NA, 2.30, 2.08, N~
$ AscAo            <dbl> NA, NA, NA, 3.6, 4.1, NA, NA, NA, NA, NA, NA, NA, NA,~
```

1.2 With corrections

and some new variables added

```r
dt <- read_csv('../NEDA/Version_03062024_SurvivalAllCases/Aortopathy_ECHO_Progression_03_V
         show_col_types = FALSE,
         col_types = cols(
           PatientID = col_character(),
           StudyID = col_character(),
           Examination_Date = col_date("%Y-%m-%d"),
           Sex = col_factor(),
           Outcome = col_factor())) %>%
  # select few variables for the sake of simplicity
  select(StudyID, PatientID, Sex, Age_At_Echo, Examination_Date, Outcome, SoV, STJ, AscAo)
  # just take randomly 10,000 rows for this demonstration
  sample_n(10000) %>%
  mutate(
    # we want to create a new variable Age that consists of range of ages
    Age = case_when(
      Age_At_Echo < 40 ~ "< 40",
      Age_At_Echo < 50 ~ "40-50",
      Age_At_Echo < 60 ~ "50-60",
      Age_At_Echo < 70 ~ "60-70",
      Age_At_Echo >= 70 ~ "  70"
    ) %>% factor(levels=c("< 40", "40-50", "50-60", "60-70", "  70"))
  ) %>%
  # another computation is to create a new variable called Aorta_Size
  # which takes the maximum value between SoV, STJ, and Asc_Ao values
  # for each scan, then categorise it to 4 groups of severities
  rowwise() %>%
  mutate(
    AortaSize = max(SoV, STJ, AscAo, na.rm=TRUE),
    AortaSize_cat = case_when(
      AortaSize <= 4.0 ~ "Normal",
      AortaSize <= 4.5 ~ "Mild",
      AortaSize <= 5.0 ~ "Moderate",
      AortaSize <= 9.0 ~ "Severe",
      .default = NA) %>% factor(levels = c("Normal", "Mild", "Moderate", "Severe"))) %>%
  ungroup()

# show the structure
glimpse(dt)
```

```
Rows: 10,000
Columns: 12
$ StudyID          <chr> "1805420", "1638511", "1594965", "1441877", "229036",~
$ PatientID        <chr> "299519", "520816", "591275", "482605", "384916", "74~
$ Sex              <fct> Female, Male, Female, Male, Male, Male, Male, Female,~
$ Age_At_Echo      <dbl> 88, 81, 60, 68, 88, 71, 65, 63, 67, 54, 86, 71, 53, 7~
$ Examination_Date <date> 2009-09-21, 2012-04-05, 2011-05-19, 2009-03-24, 2016~
$ Outcome          <fct> Dead, Alive, Dead, Alive, Dead, Alive, Alive, Alive, ~
$ SoV              <dbl> 3.700, 3.450, 2.700, 3.533, 2.892, 3.800, 4.300, 2.67~
$ STJ              <dbl> NA, 3.3, 2.0, NA, NA, NA, 3.4, NA, NA, NA, NA, 4.1, N~
$ AscAo            <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
$ Age              <fct>   70,   70, 60-70, 60-70,   70,   70, 60-70, 60-70, 6~
$ AortaSize        <dbl> 3.700, 3.450, 2.700, 3.533, 2.892, 3.800, 4.300, 2.67~
$ AortaSize_cat    <fct> Normal, Normal, Normal, Normal, Normal, Normal, Mild,~
```

Explanations

1. Correcting data types

The argument of

```
col_types = cols(
  PatientID = col_character(),
  StudyID = col_character(),
  Examination_Date = col_date("%Y-%m-%d"),
  Sex = col_factor(),
  Outcome = col_factor()
)
```

forces `read_csv()` to use specific data types for specific columns (see: `cols` specification).

2. Create a new variable with `mutate`

The statement

```
mutate(
  Age = case_when(
    Age_At_Echo < 40 ~ "< 40",
    Age_At_Echo < 50 ~ "40-50",
    Age_At_Echo < 60 ~ "50-60",
    Age_At_Echo < 70 ~ "60-70",
    Age_At_Echo >= 70 ~ "  70"
    ) %>% factor(levels=c("< 40", "40-50", "50-60", "60-70", "  70"))
```

```
)
```

creates a new column `Age` as a factor that shows a range of ages between `<40, 40-50, 50-60, 60-70,  70` years old.

There is also another `mutate` statement to create a new column `AortaSize` and `AortaSize_cat` based on the maximum value between `STJ`, `SoV` and `AscAo` measurements. I separated this creation from the above because we need to specify R to calculate the maximum value row-wise instead of column-wise. Hence the `rowwise()` function preceded.

See more about `mutate`, `case_when`, and `rowwise` functions.

1.3 Table summary

Let's summarise our data to compare all patients based on their survival: *dead* or *alive*.

```
library('gtsummary')
```

Note that the data may contain multiple scans for a patient. Thus, we will search the earliest scan first for each patient for the comparison.

```
dt %>%
  # analyse examination date per patient
  group_by(PatientID) %>%
  mutate(
    Earliest_Date = min(Examination_Date)
  ) %>%
  # release the grouping and now filter the earliest date only
  ungroup() %>%
  filter(Examination_Date == Earliest_Date) %>%
  # this should filter out multiple scan
  # we can safely give the data to tbl_summary function
  tbl_summary(
    by = Outcome,
    include = c(Sex, Age, SoV, STJ, AscAo, AortaSize, AortaSize_cat),
    missing = "no"
  ) %>%
  add_p() %>%
  separate_p_footnotes()
```

```
Table printed with `knitr::kable()`, not {gt}. Learn why at
https://www.danieldsjoberg.com/gtsummary/articles/rmarkdown.html
To suppress this message, include `message = FALSE` in code chunk header.
```

Characteristic	**Alive**, N = 7,050	**Dead**, N = 2,830	**p-value**
Sex			<0.001
Male	3,725 (53%)	1,613 (57%)	
Female	3,325 (47%)	1,217 (43%)	
Age			<0.001
< 40	1,174 (17%)	86 (3.0%)	
40-50	873 (12%)	94 (3.3%)	
50-60	1,347 (19%)	237 (8.4%)	
60-70	1,605 (23%)	532 (19%)	
70	2,051 (29%)	1,881 (66%)	
SoV	3.29 (2.96, 3.60)	3.37 (3.05, 3.70)	<0.001
STJ	3.20 (2.70, 3.60)	3.40 (2.90, 3.80)	<0.001
AscAo	3.30 (3.00, 3.60)	3.50 (3.20, 3.80)	<0.001
AortaSize	3.30 (3.00, 3.60)	3.40 (3.09, 3.70)	<0.001
AortaSize_cat			<0.001
Normal	6,569 (93%)	2,539 (90%)	
Mild	406 (5.8%)	249 (8.8%)	
Moderate	64 (0.9%)	37 (1.3%)	
Severe	11 (0.2%)	5 (0.2%)	

2 Different ways to summarise

```
library(tidyverse)
```

Let's use `mtcars` data, but first we need to convert two numeric variables into factor:

```
# fix some numeric variables into factor
dt <- mutate( mtcars,
  vs = factor(vs, levels=c(0, 1), labels=c("V-shaped", "straight")),
  am = factor(am, levels=c(0, 1), labels=c("automatic", "manual"))
)

glimpse(dt)
```

```
Rows: 32
Columns: 11
$ mpg  <dbl> 21.0, 21.0, 22.8, 21.4, 18.7, 18.1, 14.3, 24.4, 22.8, 19.2, 17.8,~
$ cyl  <dbl> 6, 6, 4, 6, 8, 6, 8, 4, 4, 6, 6, 8, 8, 8, 8, 8, 8, 4, 4, 4, 4, 8,~
$ disp <dbl> 160.0, 160.0, 108.0, 258.0, 360.0, 225.0, 360.0, 146.7, 140.8, 16~
$ hp   <dbl> 110, 110, 93, 110, 175, 105, 245, 62, 95, 123, 123, 180, 180, 180~
$ drat <dbl> 3.90, 3.90, 3.85, 3.08, 3.15, 2.76, 3.21, 3.69, 3.92, 3.92, 3.92,~
$ wt   <dbl> 2.620, 2.875, 2.320, 3.215, 3.440, 3.460, 3.570, 3.190, 3.150, 3.~
$ qsec <dbl> 16.46, 17.02, 18.61, 19.44, 17.02, 20.22, 15.84, 20.00, 22.90, 18~
$ vs   <fct> V-shaped, V-shaped, straight, straight, V-shaped, straight, V-sha~
$ am   <fct> manual, manual, manual, automatic, automatic, automatic, automati~
$ gear <dbl> 4, 4, 4, 3, 3, 3, 3, 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 4, 4, 4, 3, 3,~
$ carb <dbl> 4, 4, 1, 1, 2, 1, 4, 2, 2, 4, 4, 3, 3, 3, 4, 4, 4, 1, 2, 1, 1, 2,~
```

2.1 desc_table

```
library(desctable)
```

The library `desctable` provides `desc_table()` function to calculate main descriptive statistics. The output is a new dataframe. You can also change the output using `desc_output()` function.

Numeric variables

```
desc_table(dt %>% select(-c(am, vs))) %>% desc_output('pander')
```

	Min	Q1	Med	Mean	Q3	Max	sd	IQR
mpg	10	15	19	20	23	34	6	7.4
cyl	4	4	6	6.2	8	8	1.8	4
disp	71	121	196	231	326	472	124	205
hp	52	96	123	147	180	335	69	84
drat	2.8	3.1	3.7	3.6	3.9	4.9	0.53	0.84
wt	1.5	2.6	3.3	3.2	3.6	5.4	0.98	1
qsec	14	17	18	18	19	23	1.8	2
gear	3	3	4	3.7	4	5	0.74	1
carb	1	2	2	2.8	4	8	1.6	2

Categorical variables

```
desc_table(dt %>% select(c(am, vs))) %>% desc_output("pander")
```

	N	%
am	32	
automatic	19	59
manual	13	41
vs	32	
V-shaped	18	56
straight	14	44

See more: https://cran.r-project.org/web/packages/desctable/vignettes/desctable.html

2.2 skim

```
library(skimr)
```

`skim()` from `skimr` package provides a complete summary separated between numeric and categorical variables. Interestingly, histogram bars are shown in the last column for numeric variables.

```
skim(dt)
```

Table 2.3: Data summary

Name	dt
Number of rows	32
Number of columns	11
Column type frequency:	
factor	2
numeric	9
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
vs	0	1	FALSE	2	V-s: 18, str: 14
am	0	1	FALSE	2	aut: 19, man: 13

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
mpg	0	1	20.09	6.03	10.40	15.43	19.20	22.80	33.90	
cyl	0	1	6.19	1.79	4.00	4.00	6.00	8.00	8.00	
disp	0	1	230.72	123.94	71.10	120.83	196.30	326.00	472.00	
hp	0	1	146.69	68.56	52.00	96.50	123.00	180.00	335.00	
drat	0	1	3.60	0.53	2.76	3.08	3.70	3.92	4.93	
wt	0	1	3.22	0.98	1.51	2.58	3.33	3.61	5.42	
qsec	0	1	17.85	1.79	14.50	16.89	17.71	18.90	22.90	
gear	0	1	3.69	0.74	3.00	3.00	4.00	4.00	5.00	
carb	0	1	2.81	1.62	1.00	2.00	2.00	4.00	8.00	

See more: https://cran.r-project.org/web/packages/skimr/vignettes/skimr.html

2.3 tableone

```
library(tableone)
```

Table 1 is a common name used in biomedical research paper that describes the patient demographics. A package called `tableone` aims to ease the production of this table, and we can use this package to summarise our data.

```
CreateTableOne(data=dt, strata="am")
```

```
                   Stratified by am
                    automatic         manual          p      test
  n                     19               13
  mpg (mean (SD))    17.15 (3.83)     24.39 (6.17)   <0.001
  cyl (mean (SD))     6.95 (1.54)      5.08 (1.55)    0.002
  disp (mean (SD))  290.38 (110.17) 143.53 (87.20)   <0.001
  hp (mean (SD))    160.26 (53.91)  126.85 (84.06)    0.180
  drat (mean (SD))    3.29 (0.39)      4.05 (0.36)   <0.001
  wt (mean (SD))      3.77 (0.78)      2.41 (0.62)   <0.001
  qsec (mean (SD))   18.18 (1.75)     17.36 (1.79)    0.206
  vs = straight (%)      7 (36.8)         7 ( 53.8)   0.556
  am = manual (%)        0 ( 0.0)        13 (100.0)  <0.001
  gear (mean (SD))    3.21 (0.42)      4.38 (0.51)   <0.001
  carb (mean (SD))    2.74 (1.15)      2.92 (2.18)    0.754
```

2.4 tbl_summary

```
library(gtsummary)
```

The `gtsummary` package provides a rich collection of functions for summarising tables and results from different statistical analyses, e.g. regression, survival analysis, etc.. The simple one, `tbl_summary()`, can generate a beautiful summary table, ready for publication.

```
dt %>%
  tbl_summary(
    by = am,
    label = c(
      mpg ~ "Miles/gallon (US)",
```

```r
      cyl ~ "Number of cylinders",
      disp ~ "Displacement (cu.in)",
      hp ~ "Gross horsepower",
      drat ~ "Rear axle ratio",
      wt ~ "Weight (1,000 lbs)",
      qsec ~"Quarter mile time",
      vs ~ "Engine type",
      gear ~ "Number of forward gears",
      carb ~ "Number of carburators"
    ),
    statistic = c(all_continuous() ~ "{mean} ± {sd}")
  ) %>%
  add_p() %>%
  separate_p_footnotes() %>%
  add_overall(last = TRUE) %>%
  modify_spanning_header(c("stat_1", "stat_2") ~ "**Transmission**")
```

Characteristic	**automatic**, N = 19	**manual**, N = 13	**p-value**	**Overall**, N = 32
Miles/gallon (US)	17.1 ± 3.8	24.4 ± 6.2	0.002	20.1 ± 6.0
Number of cylinders			0.009	
4	3 (16%)	8 (62%)		11 (34%)
6	4 (21%)	3 (23%)		7 (22%)
8	12 (63%)	2 (15%)		14 (44%)
Displacement (cu.in)	290 ± 110	144 ± 87	<0.001	231 ± 124
Gross horsepower	160 ± 54	127 ± 84	0.046	147 ± 69
Rear axle ratio	3.29 ± 0.39	4.05 ± 0.36	<0.001	3.60 ± 0.53
Weight (1,000 lbs)	3.77 ± 0.78	2.41 ± 0.62	<0.001	3.22 ± 0.98
Quarter mile time	18.18 ± 1.75	17.36 ± 1.79	0.3	17.85 ± 1.79
Engine type			0.3	
V-shaped	12 (63%)	6 (46%)		18 (56%)
straight	7 (37%)	7 (54%)		14 (44%)
Number of forward gears			<0.001	
3	15 (79%)	0 (0%)		15 (47%)
4	4 (21%)	8 (62%)		12 (38%)
5	0 (0%)	5 (38%)		5 (16%)
Number of carburators			0.3	
1	3 (16%)	4 (31%)		7 (22%)

Characteristic	**automatic**, N = 19	**manual**, N = 13	**p-value**	**Overall**, N = 32
2	6 (32%)	4 (31%)		10 (31%)
3	3 (16%)	0 (0%)		3 (9.4%)
4	7 (37%)	3 (23%)		10 (31%)
6	0 (0%)	1 (7.7%)		1 (3.1%)
8	0 (0%)	1 (7.7%)		1 (3.1%)

See more: https://www.danieldsjoberg.com/gtsummary/index.html

References