

tidyr包



大纲

- 第一部分 tidyrr简介与安装
- 第二部分 数据长短格式转换
- 第三部分 数据合并
- 第四部分 数据分割
- 第五部分 缺失值处理
- 第六部分 应用实例

第一部分 tidyr简介与安装

- ❖ **tidyr**用于数据处理，可以实现数据长格式和宽格式之间的相互转换，这里所指的长格式数据就是一个观测对象由多行组成，而宽数据格式则是一个观测仅由一行组成。除此之外，**tidyr**还可以对数据进行拆分和合并，同时也能够对缺失值进行简单的处理。
- ❖ 安装：

```
> install.packages("tidyr")
```
- ❖

```
library(tidyr)
```

Tibbles - an enhanced data frame



The **tibble** package provides a new S3 class for storing tabular data, the tibble. Tibbles inherit the data frame class, but improve three behaviors:

- **Subsetting** - [always returns a new tibble, [[and \$ always return a vector.
- **No partial matching** - You must use full column names when subsetting
- **Display** - When you print a tibble, R provides a concise view of the data that fits on one screen

A tibble: 234 x 6

manufacturer	model	displ	year	cases	pop
audi	a4	1.8	1999	0.7K	19M
audi	a4	2.0	2000	2K	20M
audi	a4	2.8	1999	37K	172M
audi	a4	3.0	2000	80K	174M
audi	a4	3.6	1999	212K	1T
audi	a4	4.0	2000	213K	1T

tibble display

country	year	cases	pop
A	1999	0.7K	19M
A	2000	2K	20M
B	1999	37K	172M
B	2000	80K	174M
C	1999	212K	1T
C	2000	213K	1T

- Control the default appearance with options:
 - `options(tibble.print_max = n, tibble.print_min = m, tibble.width = Inf)`
- View full data set with `View()` or `glimpse()`
- Revert to data frame with `as.data.frame()`

CONSTRUCT A TIBBLE IN TWO WAYS

tibble(...)
Construct by columns.
`tibble(x=1:3, y=c("a", "b", "c"))`

tibble(...)
Construct by rows.
`tibble(~x, ~y, <int> <chr>)`

x	y
1	"a"
2	"b"
3	"c"

`as_tibble(x, ...)` Convert data frame to tibble.
`enframe(x, name = "name", value = "value")`
 Convert named vector to a tibble
`is_tibble(x)` Test whether x is a tibble.

Tidy Data with tidyr

Tidy data is a way to organize tabular data. It provides a consistent data structure across packages.

A table is tidy if:

Each **variable** is in its own **column**

Each **observation**, or **case**, is in its own **row**

Tidy data:

Makes variables easy to access as vectors

Preserves cases during vectorized operations

Reshape Data - change the layout of values in a table

Use `gather()` and `spread()` to reorganize the values of a table into a new layout.

gather(data, key, value, ..., na.rm = FALSE, convert = FALSE, factor_key = FALSE)
spread(data, key, value, fill = NA, convert = FALSE, drop = TRUE, sep = NULL)

`gather()` moves column names into a key column, gathering the column values into a single value column.
`spread()` moves the unique values of a key column into the column names, spreading the values of a value column across the new columns.

table4a

country	1999	2000
A	0.7K	2K
B	37K	80K
C	212K	213K

`gather(table4a, `1999`, `2000`, key = "year", value = "cases")`

table2

country	year	type	count
A	1999	cases	0.7K
A	1999	pop	19M
A	2000	cases	2K
A	2000	pop	20M
B	1999	cases	37K
B	1999	pop	172M
B	2000	cases	80K
B	2000	pop	174M
C	1999	cases	212K
C	1999	pop	1T
C	2000	cases	213K
C	2000	pop	1T

`spread(table2, type, count)`

Split Cells



Use these functions to split or combine cells into individual, isolated values.

separate(data, col, into, sep = "[^:alnum:]", remove = TRUE, convert = FALSE, extra = "warn", fill = "warn", ...)

Separate each cell in a column to make several columns.

table3

country	year	rate
A	1999	0.7K/19M
A	2000	2K/20M
B	1999	37K/172M
B	2000	80K/174M
C	1999	212K/1T
C	2000	213K/1T

`separate(table3, rate, sep = "/", into = c("cases", "pop"))`

country	year	cases	pop
A	1999	0.7K	19M
A	2000	2K	20M
B	1999	37K	172M
B	2000	80K	174M
C	1999	212K	1T
C	2000	213K	1T

separate_rows(data, ..., sep = "[^:alnum:]", convert = FALSE)
 Separate each cell in a column to make several rows.

table3

country	year	rate
A	1999	0.7K/19M
A	2000	2K/20M
B	1999	37K/172M
B	2000	80K/174M
C	1999	212K/1T
C	2000	213K/1T

`separate_rows(table3, rate, sep = "/")`

country	year	rate
A	1999	0.7K
A	2000	19M
A	2000	2K
B	1999	37K
B	1999	172M
B	2000	80K
B	2000	174M
C	1999	1T
C	1999	212K
C	2000	213K
C	2000	1T

unite(data, col, ..., sep = "_", remove = TRUE)

Collapse cells across several columns to make a single column.

table5

country	century	year
Afghan	19	99
Afghan	20	00
Brazil	19	99
Brazil	20	00
China	19	99
China	20	00

`unite(table5, century, year, col = "year", remove = TRUE)`

country	year
Afghan	1999
Afghan	2000
Brazil	1999
Brazil	2000
China	1999
China	2000

Handle Missing Values

drop_na(data, ...)
Drop rows containing NA's in ... columns.

fill(data, ..., .direction = c("down", "up"))
Fill in NA's in ... columns with most recent non-NA values.

replace_na(data, replace = list(), ...)
Replace NA's by column.

`drop_na(x, x2)`

x	x2
A	1
B	NA
C	NA
D	3
E	NA

`fill(x, x2)`

x	x2
A	1
B	NA
C	NA
D	3
E	NA

`replace_na(x, list(x2 = 2))`

x	x2
A	1
B	NA
C	NA
D	3
E	NA

Expand Tables - quickly create tables with combinations of values

complete(data, ..., fill = list())
Adds to the data missing combinations of the values of the variables listed in ...
`complete(mtcars, cyl, gear, carb)`

expand(data, ...)
Create new tibble with all possible combinations of the values of the variables listed in ...
`expand(mtcars, cyl, gear, carb)`



第二部分 数据长短格式转换

tidyr中的gather函数类似于reshape2中的melt函数，可实现将宽格式数据转换为长数据格式。

gather(data, key, value, ..., na.rm = FALSE, convert = FALSE, factor_key = FALSE)

data: 为需要转换的长形data.frame

key: 设置需要扩宽的类别变量

value: 设置需要扩宽的变量的度量值

fill: 对于缺失值，可将fill的值赋值给被转型后的缺失值

convert: 为TRUE时会自动在新列上使用type.convert函数，其中as.is = TRUE，默认值为FALSE

drop: 为FALSE保留factor的level，使用fill的值填充missing的值

sep: 为默认值NULL时，新列名使用key中的值，非NULL时，新列名为<key_name><sep><key_value>

第二部分 数据长短格式转换

	GeneId	Sample1	Sample2	Sample3
1	gene1	1	2.0	0.3
2	gene2	4	5.0	6.0
3	gene3	7	0.8	9.0
4	gene4	10	11.0	12.0

	GeneId	sample_name	expression
1	gene1	Sample1	1.0
5	gene1	Sample2	2.0
9	gene1	Sample3	0.3
2	gene2	Sample1	4.0
6	gene2	Sample2	5.0
10	gene2	Sample3	6.0
3	gene3	Sample1	7.0
7	gene3	Sample2	0.8
11	gene3	Sample3	9.0
4	gene4	Sample1	10.0
8	gene4	Sample2	11.0
12	gene4	Sample3	12.0

- ❖ # gather (data=数据框名, key="key名", value="value名", 要转换的列1, 列2, 列3)
- ❖ > gene_exp_tidy <- gather(data = gene_exp, key = "sample_name", value = "expression", Sample1, Sample2, Sample3)
- ❖ # 在指定要转换的列时, 也可不用列名, 直接指定列的编号即可

第二部分 数据长短格式转换

tidyr中的spread函数类似于reshape2中的cast函数，可实现将长格式数据转换为宽数据格式。

spread(data, key, value, fill = NA, convert = FALSE, drop = TRUE, sep = NULL)

data: 为需要转换的长形data.frame

key: 设置需要扩宽的类别变量

value: 设置需要扩宽的变量的度量值

fill: 对于缺失值，可将fill的值赋值给被转型后的缺失值

convert: 为TRUE时会自动在新列上使用type.convert函数，其中as.is = TRUE，默认值为FALSE

drop: 为FALSE保留factor的level，使用fill的值填充missing的值

sep: 为默认值NULL时，新列名使用key中的值，非NULL时，新列名为<key_name><sep><key_value>

第二部分 数据长短格式转换

	GeneId	Sample1	Sample2	Sample3
1	gene1	1	2.0	0.3
2	gene2	4	5.0	6.0
3	gene3	7	0.8	9.0
4	gene4	10	11.0	12.0

	GeneId	sample_name	expression
1	gene1	Sample1	1.0
5	gene1	Sample2	2.0
9	gene1	Sample3	0.3
2	gene2	Sample1	4.0
6	gene2	Sample2	5.0
10	gene2	Sample3	6.0
3	gene3	Sample1	7.0
7	gene3	Sample2	0.8
11	gene3	Sample3	9.0
4	gene4	Sample1	10.0
8	gene4	Sample2	11.0
12	gene4	Sample3	12.0

❖ `> spread(data = gene_exp_tidy, key = "sample_name", value = "expression")`

第三部分 数据合并

tidyr中的unite函数可将多列按指定分隔符合并为一系列。

- `unite(data, col, ..., sep = "_", remove = TRUE)`
- `data`: 为数据框
- `col`: 被组合的新列名称
- `...`: 指定哪些列需要被组合, 可用于选择两列之间的所有列`col1:coln`, 排除列`-coln`
- `sep`: 组合列之间的连接符, 默认为下划线
- `remove`: 是否删除被组合的列

第三部分 数据合并

```
> data
```

	date	hour	min	second	event
1:	2016-11-01	7	30	29	u
2:	2016-11-02	9	43	36	a
3:	2016-11-03	13	58	60	l
4:	2016-11-04	20	22	11	q
5:	2016-11-05	5	44	47	p
6:	2016-11-06	18	52	37	k
7:	2016-11-07	19	12	43	r
8:	2016-11-08	12	35	6	i
9:	2016-11-09	11	7	38	e

```
> dataNew %>%unite(datetime, datehour, min, second, sep = ':')
```

	datetime	event
1:	2016-11-01 7:30:29	u
2:	2016-11-02 9:43:36	a
3:	2016-11-03 13:58:60	l
4:	2016-11-04 20:22:11	q
5:	2016-11-05 5:44:47	p
6:	2016-11-06 18:52:37	k
7:	2016-11-07 19:12:43	r
8:	2016-11-08 12:35:6	i
9:	2016-11-09 11:7:38	e
10:	2016-11-10 1:14:21	b
11:	2016-11-11 3:20:42	w
12:	2016-11-12 14:1:32	t
13:	2016-11-13 23:19:52	h
14:	2016-11-14 21:41:26	s
15:	2016-11-15 8:16:25	o

把date, hour, min和second列合并为新列datetime

R中的日期时间格式为"Year-Month-Day-

Hour:Min:Second"

```
dataNew <- data %>%unite(datehour, date, hour, sep = '')
```

```
%>%unite(datetime, datehour, min, second, sep = ':')
```

第四部分 数据分割

- `tidyr`中的`separate`函数可将一列按分隔符分割为多列，类似于`reshape2`中的`colsplit`函数，常用于日期时间类型数据的组合和拆分
- `separate(data, col, into, sep = "[^[:alnum:]]+", remove = TRUE, convert = FALSE, extra = "warn", fill = "warn", ...)`
- `data`: 为数据框
- `col`: 需要被拆分的列
- `into`: 新建的列名，为字符串向量
- `sep`: 被拆分列的分隔符
- `remove`: 是否删除被分割的列
- `convert`: 为`TRUE`时会自动在新列上使用`type.convert`函数，其中`as.is = TRUE`，默认值为`FALSE`
- `fill`: 当分割成的列少于`length(into)`时，`"warn"`（默认值）发出警告并从右侧填充缺失值，`"right"`直接从右侧填充缺失值，`"left"`直接从左侧填充缺失值

第四部分 数据分割

```
> dataNew %>% unite(datetime, datehour, min, second, sep = ':')
```

```
datetime event
1: 2016-11-01 7:30:29 u
2: 2016-11-02 9:43:36 a
3: 2016-11-03 13:58:60 l
4: 2016-11-04 20:22:11 q
5: 2016-11-05 5:44:47 p
6: 2016-11-06 18:52:37 k
7: 2016-11-07 19:12:43 r
8: 2016-11-08 12:35:6 i
9: 2016-11-09 11:7:38 e
10: 2016-11-10 1:14:21 b
11: 2016-11-11 3:20:42 w
12: 2016-11-12 14:1:32 t
13: 2016-11-13 23:19:52 h
14: 2016-11-14 21:41:26 s
15: 2016-11-15 8:16:25 o
```

```
> data
```

```
date hour min second event
1: 2016-11-01 7 30 29 u
2: 2016-11-02 9 43 36 a
3: 2016-11-03 13 58 60 l
4: 2016-11-04 20 22 11 q
5: 2016-11-05 5 44 47 p
6: 2016-11-06 18 52 37 k
7: 2016-11-07 19 12 43 r
8: 2016-11-08 12 35 6 i
9: 2016-11-09 11 7 38 e
```

可以用separate函数将数据恢复到刚创建的时候

首先，将datetime分为date列和time列

然后，将time列分为hour, min, second列

```
data1 <- dataNew %>% separate(datetime, c('date', 'time'), sep = ' ') %>% separate(time, c('hour', 'min', 'second'), sep = ':')
```

第五部分 缺失值处理

1. 使用给定值替换每列的缺失值

```
replace_na(data, replace = list(), ...)
```

2. 以前一个值填充缺失值，默认自上向下填充

```
fill(data, ..., .direction = c("down", "up"))
```

3. 删除包含缺失值的行

```
drop_na(data, ...)
```

4. 转换隐式的缺失值为显式的

```
complete(data, ..., fill = list())
```

tidyr包



谢谢