TMC 204
Statistical Data Analysis with R
Unit 4
Manipulating Objects Part 3

Presented By : Aditya Joshi
Asst. Professor
Department of Computer Application
Graphic Era Deemed to be University
07-04-2020

Sorting Data Frames

Lets suppose we have data Frame which is inbuilt in CRAN

mtcars

> mtcars

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1

we might able to sort the data frame into an appropriate order. We will be using the **order( )** function to accomplish this.

sort dataframe by column

```
> mtcars[order(mtcars$gear),]
                     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
```

This will sort your data from Smallest to largest by gear

Get top N Results in data frame

```
> mtcars[order(-mtcars$gear),][1:5,]
                mpg cyl  disp  hp drat    wt qsec vs am gear carb
Porsche 914-2  26.0   4 120.3  91 4.43 2.140 16.7  0  1    5    2
Lotus Europa   30.4   4  95.1 113 3.77 1.513 16.9  1  1    5    2
Ford Pantera L 15.8   8 351.0 264 4.22 3.170 14.5  0  1    5    4
Ferrari Dino   19.7   6 145.0 175 3.62 2.770 15.5  0  1    5    6
Maserati Bora  15.0   8 301.0 335 3.54 3.570 14.6  0  1    5    8
```

This will give you top 5 results sorted by **gear** which is having maximum value

Sorting data frame by Multiple Factors or columns or sorting by multiple variables

if we wanted to sort the entire list by the good horse power cars and gear will be lowest to highest?

> mtcars[order(mtcars$gear, -mtcars$hp),] **- negative sign is for descending order**

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2

Merge Data Frames in R : Full and Partial Match

we have data from multiple sources. To perform an analysis, we need to merge two dataframes together with one or more common key variables.

Full match

A full match returns values that have a counterpart in the destination table. The values that are not match won't be return in the new data frame. The partial match, however, return the missing values as NA.

merge(x, y, by.x = x, by.y = y)

Arguments:

-x: The origin data frame

-y: The data frame to merge

-by.x: The column used for merging in x data frame. Column x to merge on

-by.y: The column used for merging in y data frame. Column y to merge on

Example:

Create First Dataset with variables

surname
Nationality

Create Second Dataset with variables

surname
movies
The common key variable is surname. We can merge both data and check if the dimensionality is 7x3.

We add stringsAsFactors=FALSE in the data frame because we don't want R to convert string as factor, we want the variable to be treated as character.

```
> producers <- data.frame( surname =
c("Joshi","Scorsese","Hitchcock","Tarantino","Polanski"),nationality =
c("IN","US","UK","US","Poland"),stringsAsFactors=FALSE)
> movies <- data.frame(
+     surname = c("Joshi",
+                 "Scorsese",
+                 "Hitchcock",
+                 "Hitchcock",
+                 "Spielberg",
+                 "Tarantino",
+                 "Polanski"),
+     title = c("Super 30",
+               "Taxi Driver",
+               "Psycho",
+               "North by Northwest",
+               "Catch Me If You Can",
+               "Reservoir Dogs","Chinatown"),
+     stringsAsFactors=FALSE)
```

```
m1 <- merge(producers, movies, by.x = "surname")
> m1
    surname nationality             title
1 Hitchcock          UK            Psycho
2 Hitchcock          UK North by Northwest
3     Joshi          IN          Super 30
4  Polanski      Poland         Chinatown
5  Scorsese          US       Taxi Driver
6 Tarantino          US     Reservoir Dogs
```

Let's merge data frames when the common key variables have different names.

We change surname to name in the movies data frame. We use the function identical(x1, x2) to check if both dataframes are identical.

#change name of movies dataframe

```
colnames(movies)[colnames(movies) == 'surname'] <- 'name’
```

 # Merge with different key value

```
m2 <- merge(producers, movies, by.x = "surname", by.y = "name")
head(m2)
surname nationality             title
1 Hitchcock          UK             Psycho
2 Hitchcock          UK North by Northwest
3     Joshi          IN           Super 30
4  Polanski      Poland          Chinatown
5  Scorsese          US       Taxi Driver
6 Tarantino          US     Reservoir Dogs
```

 # Check if data are identical

```
identical(m1, m2)
 [1] TRUE
```

This shows that merge operation is performed even if the column names are different.

Partial match

In the **full matching**, the dataframe returns **only** rows found in both x and y data frame. With **partial merging**, it is possible to keep the rows with no matching rows in the other data frame. These rows will have NA in those columns that are usually filled with values from y. We can do that by setting all.x= TRUE.

For instance, we can add a new producer, Nautiyal, in the producer data frame without the movie references in movies data frame. If we set all.x= FALSE, R will join only the matching values in both data set. In our case, the producer Nautiyal will not be join to the merge because it is missing from one dataset.

Let's see the dimension of each output when we specify all.x= TRUE and when we don't.

 # Create a new producer

```
add_producer <-  c('Nautiyal', 'IN')
```

 # Append it to the ` producer` dataframe

```
producers <- rbind(producers, add_producer)
```

 # Use a partial merge

```
m3 <-merge(producers, movies, by.x = "surname", by.y = "name", all.x = TRUE)
> m3
   surname nationality            title
1 Hitchcock          UK           Psycho
2 Hitchcock          UK North by Northwest
3     Joshi          IN         Super 30
4  Nautiyal          IN             <NA>
5  Polanski      Poland         Chinatown
6  Scorsese          US       Taxi Driver
7 Tarantino          US    Reservoir Dogs
```

```
# Compare the dimension of each data frame
> dim(m1)
[1] 6 3
> dim(m2)
[1] 6 3
> dim(m3)
[1] 7 3
```

As we can see, the dimension of the new data frame 7x3 compared with 6x3 for m1 and m2.

Sources : Beginning R By Dr. Mark Gardner

And internet searches