What is the process for grouping and summarizing data in R?

Authored by

stats writer

April 18, 2024

The process for grouping and summarizing data in R involves several steps. First, the data must be imported into R using the appropriate function. Next, the data can be grouped based on a specific variable or set of variables using the "group_by" function. This allows for grouping the data into categories or subsets. Then, summary functions such as "summarize" or "aggregate" can be used to calculate summary statistics for each group. These summary statistics can include measures like mean, median, standard deviation, and more. Finally, the summarized data can be visualized using various plotting functions in R. This process allows for efficient and effective analysis of large datasets and can provide valuable insights into the data.

The Complete Guide: Group & Summarize Data in R

Two of the most common tasks that you'll perform in data analysis are grouping and summarizing data.

Fortunately the dplyr package in R allows you to quickly group and summarize data.

This tutorial provides a quick guide to getting started with dplyr.

Install & Load the dplyr Package

Before you can use the functions in the dplyr package, you must first load the package:

#install dplyr (if not already installed)
install.packages('dplyr')

#load dplyr

library(dplyr)

Next, we'll illustrate several examples of how to use the functions in dplyr to group and summarize data using the built-in R dataset called mtcars:

#obtain rows and columns of *mtcars*
dim(mtcars)

32 11

#view first six rows of *mtcars*head(mtcars)

mpg cyl disp hp drat wt qsec vs am gear carb
Mazda RX4 21.0 6 160 110 3.90 2.620 16.46 0 1 4 4
Mazda RX4 Wag 21.0 6 160 110 3.90 2.875 17.02 0 1 4 4
Datsun 710 22.8 4 108 93 3.85 2.320 18.61 1 1 4 1
Hornet 4 Drive 21.4 6 258 110 3.08 3.215 19.44 1 0 3 1
Hornet Sportabout 18.7 8 360 175 3.15 3.440 17.02 0 0 3 2
Valiant 18.1 6 225 105 2.76 3.460 20.22 1 0 3 1

The basic syntax that we'll use to group and summarize data is as follows:

```
data %>%
group_by(col_name) %>%
summarize(summary_name = summary_function)
```

Note: The functions summarize() and summarise() are equivalent.

Example 1: Find Mean & Median by Group

The following code shows how to calculate measures of central tendency by group including the mean and the median:

```
#find mean mpg by cylinder
mtcars %>%
group_by(cyl) %>%
summarize(mean_mpg = mean(mpg, na.rm = TRUE))

# A tibble: 3 x 2
cyl mean_mpg

1 4 26.7
2 6 19.7
3 8 15.1

#find median mpg by cylinder
```

```
mtcars %>%
group_by(cyl) %>%
summarize(median_mpg = median(mpg, na.rm = TRUE))

# A tibble: 3 x 2
cyl median_mpg

1 4 26
2 6 19.7
3 8 15.2
```

Example 2: Find Measures of Spread by Group

The following code shows how to calculate measures of dispersion by group including the standard deviation, interquartile range, and median absolute deviation:

```
#find sd, IQR, and mad by cylinder
mtcars %>%
group_by(cyl) %>%
summarize(sd_mpg = sd(mpg, na.rm = TRUE),
iqr_mpg = IQR(mpg, na.rm = TRUE),
mad_mpg = mad(mpg, na.rm = TRUE))

# A tibble: 3 x 4
```

cyl sd_mpg iqr_mpg mad_mpg

1 4 4.51 7.60 6.52
2 6 1.45 2.35 1.93
3 8 2.56 1.85 1.56

Example 3: Find Count by Group

The following code shows how to find the count and the unique count by group in R:

```
#find row count and unique row count by cylinder
mtcars %>%
group_by(cyl) %>%
summarize(count_mpg = n(),
u_count_mpg = n_distinct(mpg))
```

```
# A tibble: 3 x 3
cyl count_mpg u_count_mpg
```

1 4 11 9
2 6 7 6
3 8 14 12

Example 4: Find Percentile by Group

The following code shows how to find the 90th percentile of values for mpg by cylinder group:

```
#find 90th percentile of mpg for each cylinder group
mtcars %>%
group_by(cyl) %>%
summarize(quant90 = quantile(mpg, probs = .9))

# A tibble: 3 x 2
cyl quant90

1 4 32.4
2 6 21.2
3 8 18.3
```

You can find the complete documentation for the dplyr package along with helpful visualize cheat sheets here.

Other useful functions that you can use along with group_by() and summarize() include functions for filtering data frame rows and arranging rows in certain orders.