How can I use the aggregate() function in R to calculate a summary statistic for a data frame, while also retaining rows with missing values (NA)?

Authored by

stats writer

June 24, 2024

RECOMMENDED CITATION

stats writer (2024). *How can I use the aggregate() function in R to calculate a summary statistic for a data frame, while also retaining rows with missing values (NA)?*. PSYCHOLOGICAL SCALES. Retrieved from https://scales.arabpsychology.com/?p=151402

The aggregate() function in R allows for the calculation of summary statistics for a data frame while also retaining rows with missing values (NA). This function takes in a data frame and allows the user to specify which columns to group by and which summary statistic to calculate. The resulting output will include the aggregated values for the specified columns, while also retaining any rows with missing values. This is useful for analyzing data sets that may contain missing values, as it allows for a more comprehensive summary of the data.

R: Use aggregate() and Not Drop Rows with NA

You can use the aggregate() function in R to calculate summary statistics for variables in a data frame.

By default, if the aggregate() function encounters a row in a data frame with one or more NA values, it will simply drop the row when performing calculations.

This can cause unintended consequences when performing calculations.

To avoid this behavior, you can use the argument na.action=NULL within the aggregate() function.

The following example shows how to use this argument in practice.

Example: Use aggegate() in R and Do Not Drop Rows with NA

Suppose we have the following data frame in R that shows the points and assists for basketball players on

various teams:

```
#create data frame
df <- data.frame(team=c('A', 'A', 'A', 'A', 'A', 'B', 'B', 'B', 'C', 'C'),
points=c(5, 9, 12, 14, 14, 13, 10, 6, 15, 18),
assists=c(NA, 4, 4, 5, 3, 6, 8, 4, 10, NA))

#view data frame
df

team points assists
1 A 5 NA
2 A 9 4
3 A 12 4
4 A 14 5
5 A 14 3
6 B 13 6
7 B 10 8
8 B 6 4
9 C 15 10
10 C 18 NA
```

Now suppose that we attempt to use the aggregate() function to calculate the sum of points and assists,

grouped by team:

#attempt to calculate sum of points and assists, grouped by team
aggregate(. ~ team, data=df, FUN=sum, na.rm=TRUE)

team points assists
1 A 49 16
2 B 29 18
3 C 15 10

The output appears to show us the sum of points and assists by team, but the rows with NA values were actually dropped when performing these calculations.

We can confirm this by viewing the original data frame and seeing that team C has two values in the points column:

1518

Thus, team C should have a sum of points of 33, but the output only shows 15.

This is because the row with a points value of 18 has a value of NA in the assists column, which means this

row was actually not used when calculating the sum of points for team C.

To ensure that rows with NA values are not dropped when performing calculations, we must use the argument na.action=NULL as follows:

#calculate sum of points and assists, grouped by team (don't drop NA rows)
aggregate(. ~ team, data=df, FUN=sum, na.rm=TRUE, na.action=NULL)

team points assists
1 A 54 16
2 B 29 18
3 C 33 10

Note: The argument na.rm=TRUE specifies that NA values should be *ignored* when performing a calculation in a specific column.