How can I use the dplyr package in R to create a new variable that is based on a condition, specifically if a certain column contains a certain string?

Authored by

stats writer

June 26, 2024

The dplyr package is a useful tool in R for manipulating data frames. One of its functions allows the user to create a new variable based on a condition, specifically if a certain column contains a certain string. This can be achieved by using the "mutate" function and specifying the condition in the form of a logical statement. The resulting new variable will be added to the data frame, making it easy to perform further analysis and visualization. Overall, the dplyr package offers a simple and efficient way to incorporate conditional logic into data manipulation tasks in R.

dplyr: Mutate Variable if Column Contains String

You can use the following basic syntax in to mutate a variable if a column contains a particular string:

library(dplyr)

df %>% mutate_at(vars(contains('starter')), ~ (scale(.) %>% as.vector))

This particular syntax applies the scale() function to each variable in the data frame that contains the string 'starter' in the column name.

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

Example: Mutate Variable if Column Contains String

Suppose we have the following data frame in R:

#create data frame

```
df <- data.frame(team=c('A', 'B', 'C', 'D', 'E', 'F'),
starter_points=c(22, 26, 25, 13, 15, 22),
starter_assists=c(4, 5, 10, 14, 12, 10),
bench_points=c(7, 7, 9, 14, 13, 10),
bench_assists=c(2, 5, 5, 4, 9, 14))

#view data frame
df
```

```
team starter_points starter_assists bench_points bench_assists
1 A 22 4 7 2
2 B 26 5 7 5
3 C 25 10 9 5
4 D 13 14 14 4
5 E 15 12 13 9
6 F 22 10 10 14
```

We can use the following syntax to apply the scale() function to each variable in the data frame that contains the string 'starter' in the column name.

```
library(dplyr)

#apply scale() function to each variable that contains
```

'starter' in the name

df %>% mutate_at(vars(contains('starter')), ~ (scale(.) %>% as.vector))

team starter_points starter_assists bench_points bench_assists

1 A 0.2819668 -1.3180158 7 2

2 B 1.0338784 -1.0629159 7 5

3 C 0.8459005 0.2125832 9 5

4 D -1.4098342 1.2329825 14 4

5 E -1.0338784 0.7227828 13 9

6 F 0.2819668 0.2125832 10 14

Using this syntax, we were able to apply the scale() function to scale each column that contained 'starter' such that their values now have a mean of 0 and standard deviation of 1.

Notice that the following columns were modified:

starter_pointsstarter_assists

All other columns remained unchanged.

Also note we can apply any function we'd like using this

syntax.

In the previous example, we chose to scale each column with the string 'starter' in the name.

However, we could do something simpler such as multiply the values by two for each column with 'starter' in the name:

```
library(dplyr)

#multiply values by two for each variable that contains 'starter' in the name
df %>% mutate_at(vars(contains('starter')), ~ (. * 2))
```

```
  team starter_points starter_assists bench_points bench_assists
1  A             44              8            7             2
2  B             52             10            7             5
3  C             50             20            9             5
4  D             26             28           14             4
5  E             30             24           13             9
6  F             44             20           10            14
```

The following tutorials explain how to perform other

common tasks in dplyr: