How can I use the scale() function in R and what are some examples of its application?

Authored by

stats writer

July 1, 2024

The scale() function in R is a powerful tool that allows for standardization and normalization of data. It is primarily used to transform numerical data into a specific range or distribution, making it easier to compare and analyze. This function can be applied to various types of data, including vectors, matrices, and data frames. It has a wide range of applications in data analysis, such as preprocessing, feature engineering, and machine learning.

To use the scale() function, the user can specify the data to be scaled and the desired range or distribution using the parameters "center" and "scale". The center parameter allows for shifting the data to a specific mean, while the scale parameter allows for scaling the data to a desired standard deviation. The result of the scale() function is a scaled version of the original data, which can then be used for further analysis.

Some examples of the application of the scale() function include standardizing numerical data before performing regression analysis, normalizing data before clustering, and scaling data before feeding it into a neural network. It is also commonly used in data visualization to ensure that the data is displayed accurately and in a meaningful way.

In summary, the scale() function in R is a versatile tool that allows for the transformation of data into a desired range or distribution, making it a valuable asset in data analysis and visualization. Its application can greatly enhance the accuracy and effectiveness of data analysis and aid in making informed decisions.

Use the scale() Function in R (With Examples)

The scale() function in R can be used to scale the values in a vector, matrix, or data frame.

This function uses the following basic syntax:

scale(x, center = TRUE, scale = TRUE)

where:

x: Name of the object to scalecenter: Whether to

subtract the mean when scaling. Default is TRUE.scale: Whether to divide by the standard deviation when scaling. Default is TRUE.

This function uses the following formula to calculate scaled values:

xscaled = (xoriginal - x?) / s

where:

xoriginal: The original x-valuex?: The sample means: The sample standard deviation

This is also known as *standardizing* data, which simply converts each original value into a .

The following examples show how to use this function in practice.

Example 1: Scale the Values in a Vector

Suppose we have the following vector of values in R:

```
#define vector of values
x <- c(1, 2, 3, 4, 5, 6, 7, 8, 9)
```

#view mean and standard deviation of values
mean(x)

5

sd(x)

2.738613

The following code shows how to scale the values in the vector using the scale() function:

#scale the values of x
x_scaled <- scale(x)

#view scaled values
x_scaled

-1.4605935
-1.0954451
-0.7302967
-0.3651484
0.0000000
0.3651484
0.7302967
1.0954451

1.4605935

Here is how each scaled value was calculated:

Value 1: (1 - 5) / 2.738613 = -1.46Value 2: (2 - 5) / 2.738613 = -1.09Value 3: (3 - 5) / 2.738613 = -0.73

And so on.

Note that if we specified scale=FALSE then the function would not have divided by the standard deviation when performing the scaling:

```
#scale the values of x but don't divide by standard deviation
x_scaled <- scale(x, scale = FALSE)

#view scaled values
x_scaled
```

-4

-3

-2

-1

0

1

2

3

4

Here is how each scaled value was calculated:

Value 1: 1 - 5 = -4Value 2: 2 - 5 = -3Value 3: 3 - 5 = -2

And so on.

Example 2: Scale the Column Values in a Data Frame

More often than not, we use the scale() function when we want to scale the values in multiple columns of a data frame such that each column has a mean of 0 and a standard deviation of 1.

For example, suppose we have the following data frame in R:

```
#create data frame
df <- data.frame(x=c(1, 2, 3, 4, 5, 6, 7, 8, 9),
y=c(10, 20, 30, 40, 50, 60, 70, 80, 90))

#view data frame
df
```

x y
1 1 10
2 2 20
3 3 30
4 4 40
5 5 50
6 6 60
7 7 70
8 8 80
9 9 90

Notice that the range of values for the y variable is much larger than the range of values for the x variable.

We can use the scale() function to scale the values in both columns such that the scaled values of x and y both have a mean of 0 and a standard deviation of 1:

```
#scale values in each column of data frame
df_scaled <- scale(df)

#view scaled data frame
df_scaled
```

x y

-1.4605935 -1.4605935

-1.0954451 -1.0954451

-0.7302967 -0.7302967

-0.3651484 -0.3651484

0.0000000 0.0000000

0.3651484 0.3651484

0.7302967 0.7302967

1.0954451 1.0954451

1.4605935 1.4605935

Both the x column and the y column now have a mean of 0 and a standard deviation of 1.

Additional Resources