# pipe operator in R-Simplify Your Code with %>%

BY FINNSTATS

pipe operator in R comes from the “magrittr” package. What does the pipe operator do?

The purpose is to lower improvement time and to enhance clarity and maintainability of code.

Packages in “tidyverse” load %>% pipe operator automatically, so usually no need to load “magrittr” explicitly.

The major task of this operator will forward a value, or the result of an expression, into the next function call/expression.

[Naive Bayes Classification in R » Prediction Model »](https://finnstats.com/index.php/2021/04/08/naive-bayes-classification-in-r/)

For instance, a function to filter data with the dplyr package can be written as

library(dplyr)

filter(data,variable==value)

or

data %>% filter(variable==value)

Above functions whole the equal task, but while we need to carry out more than one function we can get a clearer written code through using the %>% operator.

library(tidyverse)

Objective:

Suppose we want to look at the common engine displacement of automobiles with four cylinders grouped via way of means of to transmission from biggest to smallest.

In a common scenario, we can solve the above problem based on three methods.

First method with nested functions, a second option with using objects, and a third option with a %>% operator.

Here we are going discuss about nested function and %>% operator.

[Random Forest in R » Complete Tutorial »](https://finnstats.com/index.php/2021/04/13/random-forest-in-rrandom-forest-in-r/)

Method1:- Nested Function

arrange(

  summarize(

    group\_by(

      filter(mpg,cyl==4),trans

    ),avg\_displ=mean(displ)

  ),  desc(avg\_displ))

Ouput:-

trans      avg\_displ

  <chr>          <dbl>

1 auto(s4)        2.5

2 auto(s5)        2.4

3 auto(av)        2.25

4 auto(l4)        2.20

5 manual(m5)      2.15

6 auto(l3)        2.1

7 manual(m6)      2.07

8 auto(s6)        2

9 auto(l5)        1.9

In this case, it’s analyzing from the middle, and a few instances now no longer smooth to address those types of codes.

If you are using multiple objects (not explained here) for large data set leads to memory issues, so we can handle the same thorough %>%, operator.

[Decision Trees in R » Classification & Regression »](https://finnstats.com/index.php/2021/04/19/decision-trees-in-r/)

Method 2: %>% pipe operator in R

mpg %>%

  filter(cyl==4) %>%

  group\_by(trans) %>%

  summarize(avg\_displ=mean(displ)) %>%

              arrange(desc(avg\_displ))

Output:-

trans      avg\_displ

  <chr>          <dbl>

1 auto(s4)        2.5

2 auto(s5)        2.4

3 auto(av)        2.25

4 auto(l4)        2.20

5 manual(m5)      2.15

6 auto(l3)        2.1

7 manual(m6)      2.07

8 auto(s6)        2

9 auto(l5)        1.9

Conclusion

The %>% operator provides a cleaner, more readable and effiecient functions.

[Principal component analysis (PCA) in R »](https://finnstats.com/index.php/2021/05/07/pca/)