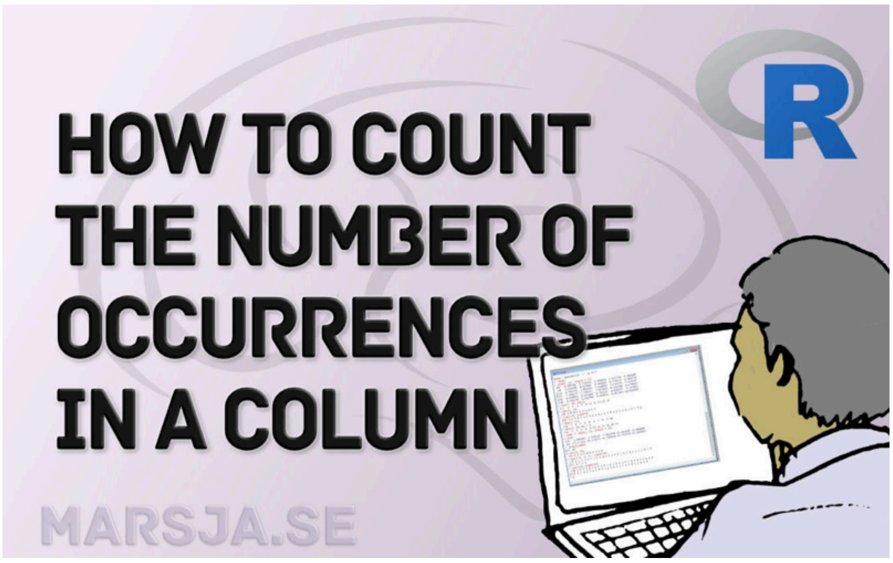
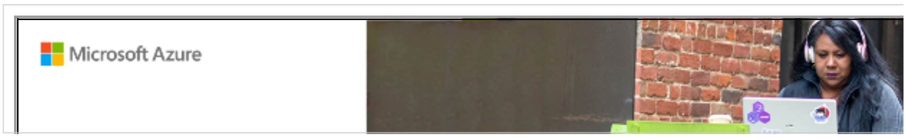
Search

R Count the Number of Occurrences in a Column using dplyr

by Erik Marsja | Apr 27, 2021 | Programming, R | 0 comments





In this R tutorial, you are going to learn how to count the number of occurrences in a column. Sometimes, before starting to analyze your data, it may be useful to know how many times a given value occurs in your variables. For example, when you have a limited set of possible values that you want to compare, In this case, you might want to know how many there are of each possible value before you carry out your analysis. Another example may be that you want to count the number of duplicate values in a column. Moreover, if we want to get an overview or information, let us say: how many men and women you have in your data set. In Psychological science. In this example, it is obligatory that you report the number of men and women in your research articles.

Posts

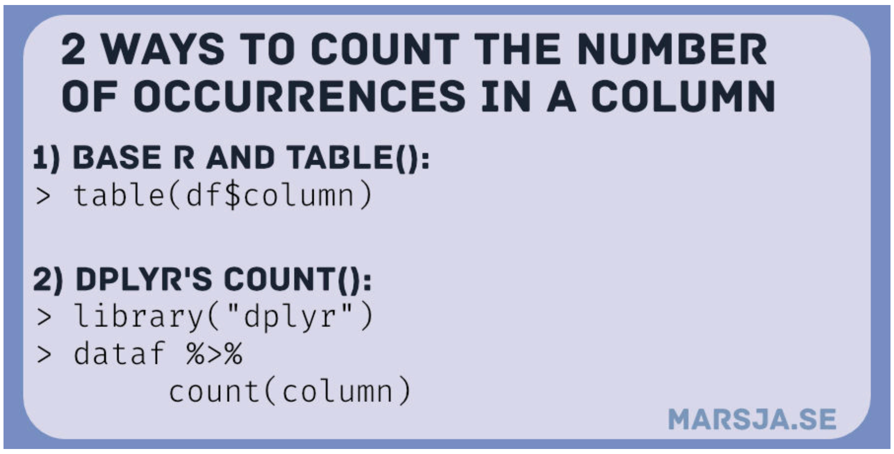
ow to Convert Matrix to dataframe in R with base functions & tibble

R Count the Number of Occurrences in a Column using dplyr

How to Do the Brown-Forsythe Test in R: A Step-By-Step Example

How to Concatenate Two Columns (or More) in R – stringr, tidyr

How to Calculate Five-Number Summary Statistics in R


2 WAYS TO COUNT THE NUMBER OF OCCURRENCES IN A COLUMN

1) BASE R AND TABLE():
```
> table(df$column)
```

2) DPLYR'S COUNT():
```
> library("dplyr")
> dataf %>%
      count(column)
```
MARSJA.SE

Table of Contents

Outline

In this post, you will learn how to use the R function table() to count the number of occurrences in a column. Moreover, we will also use the function count() from the package dplyr. First, we start by installing dplyr and then we import example data from a CSV file. Second, we will start looking at the table() function and how we can use it to count distinct occurrences. Here we will also have a look at how we can calculate the relative frequencies of factor levels.

Third, we will have a look at the count() function from dplyr and how to count the number of times a value appears in a column in R. Finally, we will also have a look at how we can calculate the proportion of factor/characters/values in a column.

In the next section, you are going to learn how to install dplyr. Of course, if you prefer to use table() you can jump to this section, directly.

Installing dplyr

As you may already be aware, it is quite easy to install R packages. Here's how you install dplyr using the install.packages() function:

```
install.packages("dplyr")
```

Note that dplyr is part of the Tidyverse package which can be installed. Installing the Tidyverse package will install a number of very handy and useful R packages. For example, we can use dplyr to remove columns, and remove duplicates in R. Moreover, we can use tibble to add a column to the dataframe in R. Finally, the package Haven can be used to read an SPSS file in R and to convert a matrix to a dataframe in R. For more examples, and R tutorials, see the end of the post.

Importing Example Data

Before learning how to use R to count the number of occurrences in a column, we need some data. For this tutorial, we will read data from a CSV file found online:

```
df <- read.csv('https://vincentarelbundock.github.io/Rdatasets/csv/carD
ata/Arrests.csv')
```

This data contains details of a person who has been arrested and in this tutorial we are going to have a look sex and age columns. First, the sex column classifies an individual's gender as male or female. Second, the age is, of course, referring to an individual in the datasets age. Let us have a quick look at the dataset:

structure of example data

Now, using the str() function we can see that we have 5226 observations across 9 columns. Moreover, we can se the data type of the 9 columns.

How to Count the Number of Occurrences in R using table()

Here's how to use the R function table() to count occurrences in a column:

```
table(df['sex'])
```

As you can see, we selected the column 'sex' using brackets (i.e. df['sex']) and used is the only parameter to the table() function. Here's the result:

Values in a column counted

Note it is also possible to use $ in R to select a single column. Now, as you can see in the image above, the function returns the count of all unique values in the given column ('sex' in our case) in descending order without any null values. By glancing at the above output see that there are more men than women in the dataset. In fact, the results show us that the vast majority are men.

Selecting a column using the $ operator.

Note, both of the examples above will remove missing values. This, of course, means that they will not be counted at all. In some cases, however, we may want to know how many missing values there are in a column as well. In the next section, we will therefore have a look at an argument that we can use (i.e., useNA) to count unique values and missing values, in a column. First, however, we are going to add 10 missing values to the column sex:

```
df_nan <- df
df_nan$sex[c(12, 24, 41, 44, 54, 66, 77, 79, 91, 101)] <- NaN
```

In the code above, we first used the column name (with the $ operator) and, then, used brackets to select rows. Finally, we used the NaN function to add the missing values to these rows that we selected. In the next section, we will count the occurrences including the 10 missing values that we just added to the dataframe.

How to Count the Number of Occurrences as well as Missing Values

Here's a code snippet that you can use to get the number of unique values in a column as well as how many missing values:

```
df_nan <- df
df_nan$sex[c(12, 24, 41, 44, 54, 66, 77, 79, 91, 101)] <- NaN
table(df_nan$sex, useNA = "ifany")
```

Now, as you can see in the code chunk above, we used the useNA argument. Here we added the character object "ifany" which will also count the missing values, if there are any. Here's the output:

Now, we already knew that we had 10 missing values in this column. Of course, when we are dealing with collected data we may not know this and, this, will let us know how many missing values there are in a specific column. In the next section, we will not count the number of times a value appears in a column in R. Next we will rather count the relative frequencies of unique values in a column.

Calculating the Relative Frequencies of the Unique Values

Another thing we can do, now, when we know how to count unique values in a column in R's dataframe is to calculate the relative frequencies of unique values. Here's how we can calculate the relative frequencies of men and women in the dataset:

```
table(df$sex)/length(df$sex)
```

In the code chunk above, we used the table() function as in the first example. We added something to get the relative frequencies of the factors (i.e., men and women). In the example, above, we used the length() function to get the total number of observations. We used this to calculate the relative frequency. This may be useful if we not only want to count the occurrences but want to know e.g. what percentage of the sample that are male and female.

How to Count the Number of Times a Value Appears in a Column in R with dplyr

Here's how we can use R to count the number of occurrences in a column using the package dplyr:

```
library(dplyr)

df %>%
  count(sex)
```

count the number of times a value appears in a column r using dplyr

In the example, above, we used the %>% operator which enables us to use the count() function to get this beautiful output. Now, as you can see when we are counting the number of times a value appears in a column in R using dplyr we get a different output compared to when using table(). For another great operator, see the post about how to use the %in% operator in R.

In the next section, we are going to count the relative frequencies of factor levels. Again, we will use dplyr but this time we will use group_by(), summarise(), and mutate().

Count the Relative Frequency of Factor Levels using dplyr

In this example, we are going to use three R functions (i.e., from the dplyr package). First, we use the piping operator, again, and then we group the data by a column. After we have grouped the data we count the unique occurrences in the column, we have selected. Finally, we are calculating the frequency of factor levels:

```
df %>%
  group_by(sex) %>%
  summarise(n = n()) %>%
  mutate(Freq = n/sum(n))
```

Using the code above, we get two columns. What we did, in the code chunk above, was to group the data by the column containing gender information. We then summarized the data. Using the n() function we got the number of observations of each value. Finally, we calculated a new variable, called "Freq". Here is were we calculate the frequencies. This gives us another nice output. Let us have a look at the output:

As you can see in the output, above, we get two columns. This is because we added a new column to the summarized data: the frequencies. Of course, counting a column, such as age, as we did in the previous example would not provide any useful information. In the next section, we will have a look at how to use the R package dplyr to count unique occurrences in a column.

There are 53 unique values of age data, a mean of 23.84 and a standard deviation of 8.31. Therefore, counting the unique values of the age column would produce a lot of headaches. In the next example, we will have a look at how we can count age but getting a readable output by binning. This is useful if we want to count e.g. even more continuous data.

How to create Bins when Counting Distinct Values

As previously mentioned, we can create bins and count the number of occurrences in each of these bins. Here's an example code in which we get 5 bins:

```
df %>%
  group_by(group = cut(age, breaks = seq(0, max(age), 11))) %>%
  summarise(n = n())
```

In the code chunk above, we used the group_by() function, again (of course, after the %>% operator). In this function, we also created the groups (i.e., the bins). Here we used the seq() function that can be used to generate a sequence of numbers in R. Finally, we used the summarise() function to get the number of occurrences in the column, binned. Here's the output:

For each bin, the range of age values is the same: 11 years. One contains ages from 11 to 22. The next bin contains ages from 22 to 33. However, we also see that there are a different number of persons in each age range. This enables us to see that most people, that are arrested are under the age of 22 Now this kind of makes sense, in this case, right?

Conclusion

In this post, you have learned how to use R to count the number of occurrences in a column. Specifically, you have learned how to count occurrences using the table() function and dplyr's count() function. Moreover, you have learned how to calculate the relative frequency of factor levels in a column. Furthermore, you have learned how to count the number of occurrences in different bins, as well.

R Tutorials

Here are a bunch of other tutorials you might find useful:

- How to Do the Brown-Forsythe Test in R: A Step-By-Step Example
- Select Columns in R by Name, Index, Letters, & Certain Words with dplyr
- How to Calculate Five-Number Summary Statistics in R
- How to Concatenate Two Columns (or More) in R – stringr, tidyr