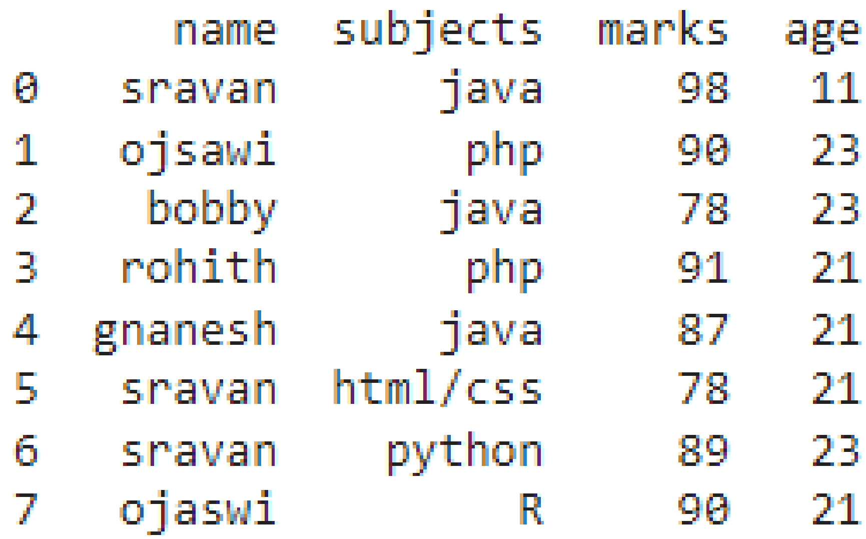
How to Count Occurrences of Specific Value in Pandas Column?

sravankumar_171fa07058

Read Discuss Courses Practice Video

In this article, we will discuss how to count occurrences of a specific column value in the pandas column.

Dataset in use:



```
      name  subjects  marks  age
0   sravan      java     98   11
1   ojsawi       php     90   23
2    bobby      java     78   23
3   rohith       php     91   21
4  gnanesh      java     87   21
5   sravan  html/css     78   21
6   sravan    python     89   23
7   ojaswi         R     90   21
```

We can count by using the value_counts() method. This function is used to count the values present in the entire dataframe and also count values in a particular column.

Syntax:

```
data['column_name'].value_counts()[value]
```

where

- data is the input dataframe
- value is the string/integer value present in the column to be counted
- column_name is the column in the dataframe

Example: To count occurrences of a specific value

Python3

```
# import pandas module
import pandas as pd

# create a dataframe
# with 5 rows and 4 columns
data = pd.DataFrame({
```

```
                 'html/css', 'python', 'R'],
    'marks': [98, 90, 78, 91, 87, 78, 89, 90],
    'age': [11, 23, 23, 21, 21, 21, 23, 21]
})

# count values in name column
print(data['name'].value_counts()['sravan'])

# count values in subjects column
print(data['subjects'].value_counts()['php'])

# count values in marks column
print(data['marks'].value_counts()[89])
```

Output:

```
3
2
1
```

If we want to count all values in a particular column, then we do not need to mention the value.

Syntax:

```
data['column_name'].value_counts()
```

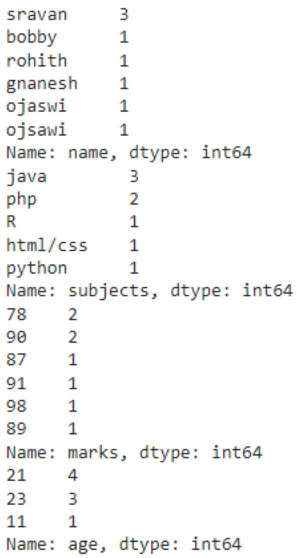
Example: To count the occurrence of a value in a particular column

Python3

```
# import pandas module
import pandas as pd

# create a dataframe
# with 5 rows and 4 columns
data = pd.DataFrame({
    'name': ['sravan', 'ojsawi', 'bobby',  'rohith',
             'gnanesh', 'sravan', 'sravan', 'ojaswi'],
    'subjects': ['java', 'php', 'java', 'php', 'java',
                 'html/css', 'python', 'R'],
    'marks': [98, 90, 78, 91, 87, 78, 89, 90],
    'age': [11, 23, 23, 21, 21, 21, 23, 21]
})
```

```
# count all values in subjects column
print(data['subjects'].value_counts())

# count all values in marks column
print(data['marks'].value_counts())

# count all values in age column
print(data['age'].value_counts())
```

Output:



```
sravan     3
bobby      1
rohith     1
gnanesh    1
ojaswi     1
ojsawi     1
Name: name, dtype: int64
java        3
php         2
R           1
html/css    1
python      1
Name: subjects, dtype: int64
78    2
90    2
87    1
91    1
98    1
89    1
Name: marks, dtype: int64
21    4
23    3
11    1
Name: age, dtype: int64
```

If we want to get the results in order (like ascending and descending order), we have to specify the parameter

Syntax:

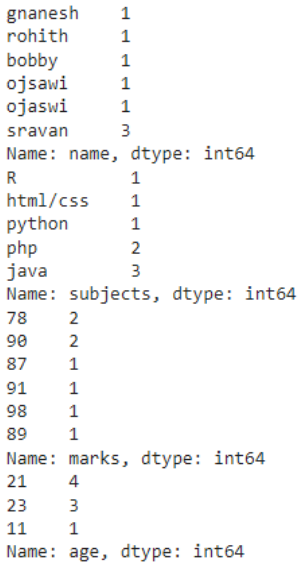
> ***Ascending order:***
>
> *data['column_name'].value_counts(ascending=True)*
>
> ***Descending Order:***
>
> *data['column_name'].value_counts(ascending=False)*

Example: To get results in an ordered fashion

Python3

```
# create a dataframe
# with 5 rows and 4 columns
data = pd.DataFrame({
    'name': ['sravan', 'ojsawi', 'bobby',  'rohith',
             'gnanesh', 'sravan', 'sravan', 'ojaswi'],
    'subjects': ['java', 'php', 'java', 'php', 'java',
                 'html/css', 'python', 'R'],
    'marks': [98, 90, 78, 91, 87, 78, 89, 90],
    'age': [11, 23, 23, 21, 21, 21, 23, 21]
})

# count all values in name column in ascending order
print(data['name'].value_counts(ascending=True))

# count all values in subjects column in ascending order
print(data['subjects'].value_counts(ascending=True))

# count all values in marks column in descending order
print(data['marks'].value_counts(ascending=False))

# count all values in age column in descending order
print(data['age'].value_counts(ascending=False))
```

Output:



```
gnanesh    1
rohith     1
bobby      1
ojsawi     1
ojaswi     1
sravan     3
Name: name, dtype: int64
R           1
html/css    1
python      1
php         2
java        3
Name: subjects, dtype: int64
78    2
90    2
87    1
91    1
98    1
89    1
Name: marks, dtype: int64
21    4
23    3
11    1
Name: age, dtype: int64
```

Dealing with missing values

Here we can count the occurrence with or without NA values. By using dropna parameter to include NA values if set to True, it will not count NA if set to False.

Syntax:

Include NA values:

data[‘column_name’].value_counts(dropna=True)

Exclude NA Values:

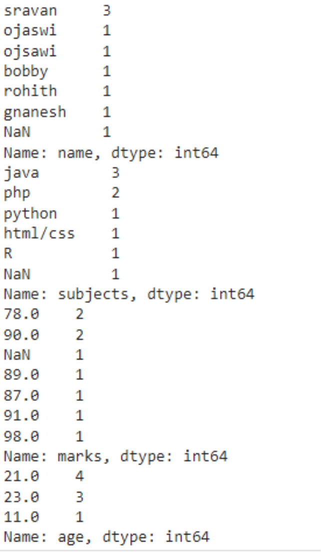
data[‘column_name’].value_counts(dropna=False)

Example: Dealing with missing values

Python3

```
# import pandas module
import pandas as pd

#import numpy
import numpy

# create a dataframe
# with 5 rows and 4 columns
data = pd.DataFrame({
    'name': ['sravan', 'ojsawi', 'bobby',  'rohith', 'gnanesh',
             'sravan', 'sravan', 'ojaswi', numpy.nan],
    'subjects': ['java', 'php', 'java', 'php', 'java', 'html/css',
                 'python', 'R', numpy.nan],
    'marks': [98, 90, 78, 91, 87, 78, 89, 90, numpy.nan],
    'age': [11, 23, 23, 21, 21, 21, 23, 21, numpy.nan]
})

# count all values in name column including NA
print(data['name'].value_counts(dropna=False))

# count all values in subjects column including NA
print(data['subjects'].value_counts(dropna=False))

# count all values in marks column excluding NA
print(data['marks'].value_counts(dropna=False))

# count all values in age column excluding NA
print(data['age'].value_counts(dropna=True))
```

Output:



```
sravan     3
ojaswi     1
ojsawi     1
bobby      1
rohith     1
gnanesh    1
NaN        1
Name: name, dtype: int64
java        3
php         2
python      1
html/css    1
R           1
NaN         1
Name: subjects, dtype: int64
78.0    2
90.0    2
NaN     1
89.0    1
87.0    1
91.0    1
98.0    1
Name: marks, dtype: int64
21.0    4
23.0    3
11.0    1
Name: age, dtype: int64
```

Count values with relative frequencies

We are going to add normalize parameter to get the relative frequencies of the repeated data. It is set to True.

Syntax:

> *data['column_name'].value_counts(normalize=True)*

Example: Count values with relative frequencies

Python3

```
# import pandas module
import pandas as pd

# create a dataframe
# with 5 rows and 4 columns
data = pd.DataFrame({
    'name': ['sravan', 'ojsawi', 'bobby',  'rohith',
             'gnanesh', 'sravan', 'sravan', 'ojaswi'],
    'subjects': ['java', 'php', 'java', 'php', 'java',
                 'html/css', 'python', 'R'],
    'marks': [98, 90, 78, 91, 87, 78, 89, 90],
    'age': [11, 23, 23, 21, 21, 21, 23, 21]
})
```

Output:

```
sravan     0.375
ojaswi     0.125
ojsawi     0.125
bobby      0.125
rohith     0.125
gnanesh    0.125
Name: name, dtype: float64
```

Get details

If we want to get the details like count, mean, std, min, 25%, 50%,75%, max, then we have to use describe() method.

Syntax:

```
data['column_name'].describe()
```

Example: Get details

Python3

```
# import pandas module
import pandas as pd

# create a dataframe
# with 5 rows and 4 columns
data = pd.DataFrame({
    'name': ['sravan', 'ojsawi', 'bobby',  'rohith',
             'gnanesh', 'sravan', 'sravan', 'ojaswi'],
    'subjects': ['java', 'php', 'java', 'php', 'java',
                 'html/css', 'python', 'R'],
    'marks': [98, 90, 78, 91, 87, 78, 89, 90],
    'age': [11, 23, 23, 21, 21, 21, 23, 21]
})

# get about age
print(data['age'].describe())
```

Output:

```
count     8.000000
mean     20.500000
std       3.964125
min      11.000000
25%      21.000000
50%      21.000000
75%      23.000000
max      23.000000
Name: age, dtype: float64
```

Using size() with groupby()

Here this will return the count of all occurrences in a particular column.

Syntax:

```
data.groupby('column_name').size()
```

Example: Count of all occurrences in a particular column

Python3

```
# import pandas module
import pandas as pd

# create a dataframe
# with 5 rows and 4 columns
data = pd.DataFrame({
    'name': ['sravan', 'ojsawi', 'bobby',  'rohith',
             'gnanesh', 'sravan', 'sravan', 'ojaswi'],
    'subjects': ['java', 'php', 'java', 'php', 'java',
                 'html/css', 'python', 'R'],
    'marks': [98, 90, 78, 91, 87, 78, 89, 90],
    'age': [11, 23, 23, 21, 21, 21, 23, 21]
})

# get the size of name
print(data.groupby('name').size())
```

Output:

```
gnanesh    1
ojaswi     1
ojsawi     1
rohith     1
sravan     3
dtype: int64
```

Using count() with groupby()

Here this will return the count of all occurrences in a particular column across all columns.

Syntax:

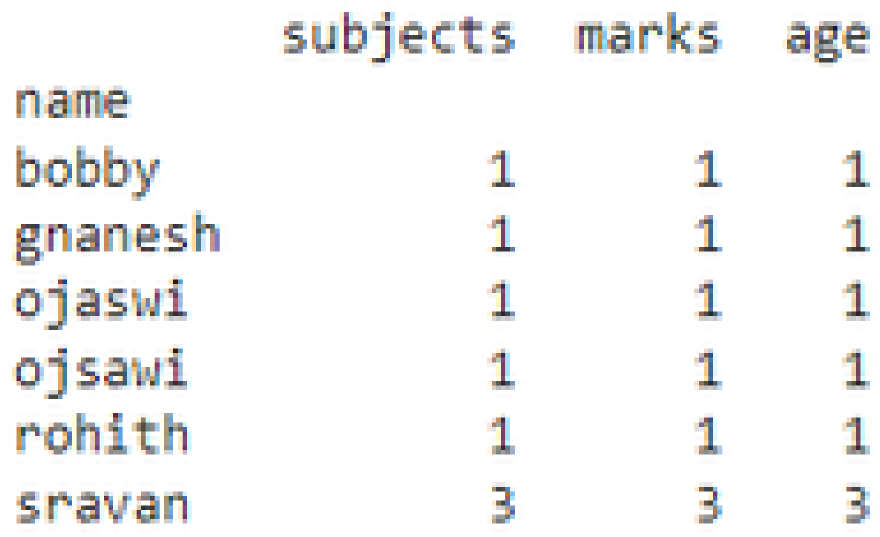
```
data.groupby('column_name').count()
```

Example: Count of all occurrences in a particular column

Python3

```
# import pandas module
import pandas as pd

# create a dataframe
# with 5 rows and 4 columns
data = pd.DataFrame({
    'name': ['sravan', 'ojsawi', 'bobby',  'rohith',
             'gnanesh', 'sravan', 'sravan', 'ojaswi'],
    'subjects': ['java', 'php', 'java', 'php', 'java',
                 'html/css', 'python', 'R'],
    'marks': [98, 90, 78, 91, 87, 78, 89, 90],
    'age': [11, 23, 23, 21, 21, 21, 23, 21]
})

# get the count of name across all columns
print(data.groupby('name').count())
```

Output:



```
         subjects  marks  age
name
bobby           1      1    1
gnanesh         1      1    1
ojaswi          1      1    1
ojsawi          1      1    1
rohith          1      1    1
sravan          3      3    3
```

Using bins

If we want to get the count in a particular range of values, then the bins parameter is applied. We can specify the number of ranges(bins).

Syntax:

```
(data['column_name'].value_counts(bins)
```

where,

- data is the input dataframe
- column_name is the column to get bins
- bins is the total number of bins to be specified

Example: Get count in particular range of values

Python3

```
# import pandas module
import pandas as pd

# create a dataframe
# with 5 rows and 4 columns
data = pd.DataFrame({
    'name': ['sravan', 'ojsawi', 'bobby',  'rohith',
             'gnanesh', 'sravan', 'sravan', 'ojaswi'],
    'subjects': ['java', 'php', 'java', 'php', 'java',
                 'html/css', 'python', 'R'],
})

# get count of  age column with  6 bins
print(data['age'].value_counts(bins=6))

# get count of  age column with  4 bins
print(data['age'].value_counts(bins=4))
```

Output:

```
(19.0, 21.0]      4
(21.0, 23.0]      3
(10.987, 13.0]    1
(17.0, 19.0]      0
(15.0, 17.0]      0
(13.0, 15.0]      0
Name: age, dtype: int64
(20.0, 23.0]      7
(10.987, 14.0]    1
(17.0, 20.0]      0
(14.0, 17.0]      0
Name: age, dtype: int64
```

Using apply()

If we want to get a count of all columns across all columns, then we have to use apply() function. In that we will use value_counts() method.

Syntax:

```
data.apply(pd.value_counts)
```

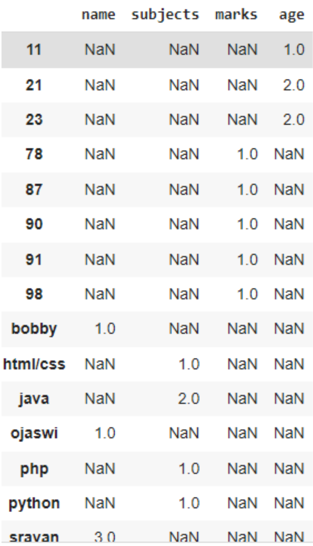
Example: Get count of all columns across all columns

Python3

```
# import pandas module
import pandas as pd

# create a dataframe
# with 5 rows and 4 columns
```

```
    'marks': [98, 90, 78, 91, 87],
    'age': [11, 23, 23, 21, 21]
})

# get all count
data.apply(pd.value_counts)
```

Output:



	name	subjects	marks	age
11	NaN	NaN	NaN	1.0
21	NaN	NaN	NaN	2.0
23	NaN	NaN	NaN	2.0
78	NaN	NaN	1.0	NaN
87	NaN	NaN	1.0	NaN
90	NaN	NaN	1.0	NaN
91	NaN	NaN	1.0	NaN
98	NaN	NaN	1.0	NaN
bobby	1.0	NaN	NaN	NaN
html/css	NaN	1.0	NaN	NaN
java	NaN	2.0	NaN	NaN
ojaswi	1.0	NaN	NaN	NaN
php	NaN	1.0	NaN	NaN
python	NaN	1.0	NaN	NaN
sravan	3.0	NaN	NaN	NaN

Last Updated : 23 Dec, 2021

2

Similar Reads

1. How to Count the NaN Occurrences in a Column in Pandas Dataframe?
2. Pandas GroupBy - Count occurrences in column
3. Insert a given column at a specific position in a Pandas DataFrame
4. Rename specific column(s) in Pandas
5. How to import excel file and find a specific column using Pandas?
6. How to merge two csv files by specific column using Pandas in Python?
7. Get column index from column name of a given Pandas DataFrame

8. Create a Pandas DataFrame from a Numpy array and specify the index column and column headers
9. Python - Scaling numbers column by column with Pandas
10. How to Drop Rows that Contain a Specific Value in Pandas?

Related Tutorials

1. Flask Tutorial
2. Natural Language Processing (NLP) Tutorial
3. Data Science for Beginners
4. Introduction to Heap - Data Structure and Algorithm Tutorials
5. Introduction to Segment Trees - Data Structure and Algorithm Tutorials

Previous Next

Article Contributed By :

sravankumar_171fa07058
@sravankumar_171fa07058

Vote for difficulty

Easy Normal Medium Hard Expert

Improved By : anikakapoor

Article Tags : pandas-dataframe-program, Picked, Python pandas-dataFrame, Python-pandas, Python

Practice Tags : python

Improve Article

Report Issue

A-143, 9th Floor, Sovereign Corporate Tower,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

Company

About Us

Careers

In Media

Contact Us

Terms and Conditions

Privacy Policy

Copyright Policy

Third-Party Copyright Notices

Advertise with us

Languages

Python

Java

C++

GoLang

SQL

R Language

Android Tutorial

Data Structures

Array

String

Linked List

Stack

Queue

Tree

Graph

Algorithms

Sorting

Searching

Greedy

Dynamic Programming

Pattern Searching

Recursion

Backtracking

Web Development

HTML

Write & Earn

Write an Article

Bootstrap

ReactJS

AngularJS

NodeJS

Write Interview Experience

Internships

Video Internship

Computer Science

GATE CS Notes

Operating Systems

Computer Network

Database Management System

Software Engineering

Digital Logic Design

Engineering Maths

Data Science & ML

Data Science With Python

Data Science For Beginner

Machine Learning Tutorial

Maths For Machine Learning

Pandas Tutorial

NumPy Tutorial

NLP Tutorial

Interview Corner

Company Preparation

Preparation for SDE

Company Interview Corner

Experienced Interview

Internship Interview

Competitive Programming

Aptitude

Python

Python Tutorial

Python Programming Examples

Django Tutorial

Python Projects

Python Tkinter

OpenCV Python Tutorial

GfG School

CBSE Notes for Class 8

CBSE Notes for Class 9

CBSE Notes for Class 10

CBSE Notes for Class 11

CBSE Notes for Class 12

English Grammar

UPSC/SSC/BANKING

SSC CGL Syllabus

SBI PO Syllabus

IBPS PO Syllabus

UPSC Ethics Notes

UPSC Economics Notes

UPSC History Notes