What are the steps to specify dtypes when importing a CSV file in Pandas?

Authored by

stats writer

June 25, 2024

When importing a CSV file in Pandas, there are several steps that you can follow to specify the data types (dtypes) of the columns in the data frame. Firstly, you can use the "dtype" parameter in the "read_csv" function to explicitly specify the data types for each column. Alternatively, you can use the "converters" parameter to define custom functions to convert the data types. Another option is to use the "dtype" parameter in the "to_numeric" function to convert the data types after importing the data frame. Additionally, you can use the "astype" function to convert the data types of specific columns in the data frame. These steps allow for the precise specification of data types when importing a CSV file in Pandas.

Pandas: Specify dtypes when Importing CSV File

You can use the following basic syntax to specify the dtype of each column in a DataFrame when importing a CSV file into pandas:

df = pd.read_csv('my_data.csv',
dtype = {'col1': str, 'col2': float, 'col3': int})

The dtype argument specifies the data type that each column should have when importing the CSV file into a pandas DataFrame.

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

Example: Specify dtypes when Importing CSV File into Pandas

Suppose we have the following CSV file called basketball_data.csv:

```
team,points,rebounds
A, 22, 10
B, 14, 9
C, 29, 6
D, 30, 2
```


If we import the CSV file using the read_csv() function, pandas will attempt to identify the data type for each column automatically:

```
import pandas as pd

#import CSV file
df = pd.read_csv('basketball_data.csv')

#view resulting DataFrame
print(df)
```

A 22 10

0 B 14 9

1 C 29 6

2 D 30 2

3 E 22 9

4 F 31 10

#view data type of each column

print(df.dtypes)

team object

points int64

rebounds int64

dtype: object

From the output we can see that the columns in the DataFrame have the following data types:

team: objectpoints: int64rebounds: int64

However, we can use the dtype argument within the read_csv() function to specify the data types that each column should have:

import pandas as pd

#import CSV file and specify dtype of each column

df = pd.read_csv('basketball_data.csv',

dtype = {'team': str, 'points': float, 'rebounds': int}))

#view resulting DataFrame
print(df)

A 22 10
0 B 14 9
1 C 29 6
2 D 30 2
3 E 22 9
4 F 31 10

#view data type of each column
print(df.dtypes)

team object
points float64
rebounds int32
dtype: object

From the output we can see that the columns in the DataFrame have the following data types:

team: objectpoints: float64rebounds: int32

These data types match the ones that we specified

using the dtype argument.

Note that in this example, we specified the dtype for each column in the DataFrame.

Note: You can find the complete documentation for the pandas read_csv() function .

The following tutorials explain how to perform other common tasks in pandas: