How can I find unique values in a Pandas dataframe while ignoring NaN values?

Authored by

stats writer

June 24, 2024

To find unique values in a Pandas dataframe while ignoring NaN values, one can use the "dropna()" function to remove all rows and columns containing NaN values. Then, the "unique()" function can be applied to the remaining dataframe to obtain a list of unique values. This method ensures that only non-NaN values are considered when searching for unique values in the dataframe.

Pandas: Find Unique Values and Ignore NaN

You can define the following custom function to find unique values in pandas and ignore NaN values:

```
def unique_no_nan(x):
return x.dropna().unique()
```

This function will return a pandas Series that contains each unique value except for NaN values.

The following examples show how to use this function in different scenarios with the following pandas DataFrame:

```
import pandas as pd
import numpy as np

#create DataFrame
df = pd.DataFrame({'team': ,
'points': })
```

#view DataFrame
print(df)

team points
0 Mavs 95.0
1 Mavs 95.0
2 Mavs 100.0
3 Celtics 113.0
4 Celtics 100.0
5 Celtics NaN

Example 1: Find Unique Values in Pandas Column and Ignore NaN Values

Suppose we use the pandas unique() function to display all of the unique values in the points column of the DataFrame:

#display unique values in 'points' column
df.unique()

array()

Notice that the unique() function includes nan in the results by default.

However, suppose we instead use our custom function unique_no_nan() to display the unique values in the points column:

```
#display unique values in 'points' column and ignore NaN
unique_no_nan(df)
```

```
array()
```

Our function returns each unique value in the points column, not including NaN.

Example 2: Find Unique Values in Pandas Groupby and Ignore NaN Values

Suppose we use the pandas groupby() and agg() functions to display all of the unique values in the points column, grouped by the team column:

```
#display unique values in 'points' column grouped by team
df.groupby('team').agg()
```

```
unique
team
```

Celtics
Mavs

Notice that the unique() function includes nan in the results by default.

However, suppose we instead use our custom function unique_no_nan() to display the unique values in the points column, grouped by the team column:

```
#display unique values in 'points' column grouped by team and ignore NaN
df.groupby('team').apply(lambda x: unique_no_nan(x))
```

```
team
Celtics
Mavs
Name: points, dtype: object
```

Our function returns each unique value in the points column for each team, not including NaN values.

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