How to Easily Find Rows in One DataFrame That Are Not in Another

Authored by

stats writer

November 22, 2025

If you want to get all the rows in one dataframe that are not in another dataframe, you can use the `pd.concat()` method to combine the two dataframes and then use the `.drop_duplicates(keep=False)` method to drop any duplicate rows, leaving only the rows that were not in the other dataframe. This is a useful technique for finding rows that are unique to one dataframe and not in the other.

You can use the following basic syntax to get the rows in one pandas DataFrame which are not in another DataFrame:

#merge two DataFrames and create indicator column
df_all = df1.merge(df2.drop_duplicates(), on=,
how='left', indicator=True)

#create DataFrame with rows that exist in first DataFrame only
df1_only = df_all == 'left_only']

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

Example: Get Rows in Pandas DataFrame Which Are Not in Another DataFrame

Suppose we have the following two pandas DataFrames:

import pandas as pd

#create first DataFrame
df1 = pd.DataFrame({'team' : ,
'points' : })

print(df1)

team points
0 A 12
1 B 15
2 C 22
3 D 29
4 E 24

#create second DataFrame
df2 = pd.DataFrame({'team' : ,

'points' : })

print(df2)

team points
0 A 12
1 D 29
2 F 15
3 G 19
4 H 10

We can use the following syntax to merge the two DataFrames and create an indicator column to indicate which rows belong in each DataFrame:

#merge two DataFrames and create indicator column
df_all = df1.merge(df2.drop_duplicates(), on=,
how='left', indicator=True)

#view result
print(df_all)

We can then use the following syntax to only get the rows in the first DataFrame that are not in the second DataFrame:

#create DataFrame with rows that exist in first DataFrame only
df1_only = df_all == 'left_only']

#view DataFrame
print(df1_only)

team points _merge
1 B 15 left_only
2 C 22 left_only
4 E 24 left_only

Lastly, we can drop the _**merge** column if we'd like:

#drop '_merge' column
df1_only = df1_only.drop('_merge', axis=1)

#view DataFrame

```
print(df1_only)
```

```
team points
1 B 15
2 C 22
4 E 24
```

The result is a DataFrame in which all of the rows exist in the first DataFrame but not in the second DataFrame.