How can I perform fuzzy matching in Pandas, and can you provide an example?

Authored by

stats writer

June 29, 2024

Fuzzy matching in Pandas is a method used to identify and link similar data points within a dataset. It allows for variations in spelling, punctuation, and other discrepancies, making it useful for data sets with potential human error or inconsistencies. This can be achieved by using the built-in function "str.contains()" and specifying a fuzzy matching algorithm such as Levenshtein distance or Jaro-Winkler distance. An example of this can be seen in the following code:

df.str.contains('search_term', regex=True, case=False, na=False, flags=re.IGNORECASE)]

This code will return all rows in the "column_name" column that contain the specified "search_term" with a fuzzy match. Overall, fuzzy matching in Pandas can help improve data accuracy and efficiency in data analysis and management.

Perform Fuzzy Matching in Pandas (With Example)

Often you may want to join together two datasets in pandas based on imperfectly matching strings. This is called fuzzy matching.

The easiest way to perform fuzzy matching in pandas is to use the get_close_matches() function from the difflib package.

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

Example: Fuzzy Matching in Pandas

Suppose we have the following two pandas DataFrames that contain information about various basketball teams:

```
import pandas as pd

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

df2 = pd.DataFrame({'team': ,
'assists': })

#view DataFrames
print(df1)

team points
0 Mavericks 99
1 Nets 90
2 Warriors 104
3 Heat 117
4 Lakers 100

print(df2)

team assists
0 Mavricks 22
1 Warrors 29
2 Heat 17
3 Netts 40
```

4 Lakes 32

Now suppose that we would like to merge the two DataFrames based on the team column.

Since the team names are slightly different between the two DataFrames, we must use fuzzy matching to find which team names most closely match.

We can use the get_close_matches() function from the difflib package to do so:

```
import difflib

#create duplicate column to retain team name from df2
df2 = df2

#convert team name in df2 to team name it most closely matches in df1
df2 = df2.apply(lambda x: difflib.get_close_matches(x, df1))

#merge the DataFrames into one
df3 = df1.merge(df2)

#view final DataFrame
```

print(df3)

team points assists team_match
0 Mavericks 99 22 Mavricks
1 Nets 90 40 Netts
2 Warriors 104 29 Warrors
3 Heat 117 17 Heat
4 Lakers 100 32 Lakes

The result is one data frame that contains each of the five team names from the first DataFrame along with the team that most closely matches from the second DataFrame.

The team_match column shows the team name from the second DataFrame that most closely matched the team name from the first DataFrame.

Note #1: By default, get_close_matches() returns the three closest matches. However, by using the at the end of the lambda function we were able to only return the closest team name match.

Note #2: You can find the complete documentation for the get_close_matches() function .

Additional Resources

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