Fuzzy String Matching in R

(Using fuzzywuzzy, polyfuzz, and difflib)

If you like this video, please subscribe so that I can continue to make content like this.

String matching: more generally

company_name	location
Apple	Cupertino, California
Google	Mountain View, California
Amazon	Seattle, Washington

name_of_company	NASDAQ_close
Apple	$174.72
Google	$2,833.46
Amazon	$3,295.47

String matching: more generally

company_name	location
Apple	Cupertino, California
Google	Mountain View, California
Amazon	Seattle, Washington

name_of_company	NASDAQ_close
Apple	$174.72
Google	$2,833.46
Amazon	$3,295.47

String matching: more generally

company_name	location
Apple	Cupertino, California
Google	Mountain View, California
Amazon	Seattle, Washington

name_of_company	NASDAQ_close
Apple	$174.72
Google	$2,833.46
Amazon	$3,295.47

String matching: more generally

company_name	location	NASDAQ_close
Apple	Cupertino, California	$174.72
Google	Mountain View, California	$2,833.46
Amazon	Seattle, Washington	$3,295.47

dataset1 LEFT JOIN dataset2 on dataset1.company_name = dataset2.name_of_company

String matching: more generally

company_name	location
Apple	Cupertino, California
Google	Mountain View, California
Amazon	Seattle, Washington

name_of_company	NASDAQ_close
Appl	$174.72
GOOGLE	$2,833.46
Amazon.com, Inc	$3,295.47

String matching: more generally

company_name	location
Apple	Cupertino, California
Google	Mountain View, California
Amazon	Seattle, Washington

name_of_company	NASDAQ_close
Appl	$174.72
GOOGLE	$2,833.46
Amazon.com, Inc	$3,295.47

Enter...

“Fuzzy” string matching!

Often, we want to compare two strings that are referring to the same information, but the two strings are written slightly differently.

This may occur because of...

- Typos, or different spellings

company_name
Apple

name_of_company
Appl

This may occur because of...

- Differences in capitalizations

company_name
Google

name_of_company
GOOGLE

This may occur because of...

- Different usages of non-alphanumeric characters

company_name
Amazon

name_of_company
Amazon.com, Inc

In cases like these, we can use fuzzy string matching.

Fuzzy string matching

- Sometimes also known as “approximate” string matching
- We can compute a similarity score between two strings

company_name
Amazon

name_of_company
Amazon.com, Inc

Similarity Score: 0.5714286

Fuzzy string matching

- Sometimes also known as “approximate” string matching
- We can compute a similarity score between two strings
- The **higher** the score, the more similar the strings

company_name
Apple

name_of_company
Appl

Similarity Score: 0.8888889

Just like we can calculate how similar two numeric vectors are, we can determine how similar two strings are.

Levenshtein distance

- The Levenshtein distance is the minimum number of single-character edits (additions, substitutions, or deletions) required to get from one string to the other

The `fuzzywuzzy` Python package uses the Levenshtein distance (in some cases).

Gestalt Pattern Matching

- Gestalt Pattern Matching (aka Ratcliff/Obershelp Pattern Recognition) produces matches that look more “correct” to the human eye
- Depends on the **longest common substring** between the two strings

The `difflib` Python package uses Gestalt Pattern Matching. So does the `fuzzywuzzy` package (in some cases).

Choosing an algorithm

- There are many different fuzzy string matching algorithms (and more being developed all the time)
- Take into consideration computational time, availability of algorithms in programming languages, etc.

We'll be using a Kaggle dataset. Link in description and on my GitHub!

kaggle

Search

Sign In

Register

Create

Home

Competitions

Datasets

Code

Discussions

Courses

More

Fuzzy String Matching with Hotel Rooms

Python · Room Type

Notebook Data Logs Comments (0)

Data

room_type.csv (6.78 kB)

Detail Compact Column

2 of 2 columns

About this file

Input (6.78 kB)

Data Sources

Room Type

room_type.csv