FuzzyWuzzy is a great all-purpose library for fuzzy string matching. It has been developed by [SeatGeek](https://github.com/seatgeek/fuzzywuzzy" \t "_blank). As per their [blog](https://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/) is was developed to pull in event ticket from all over the Internet and displaying them on the same screen in order to compare them. In order to do this they built library of 'fuzzy' string matching. It uses Levenshtein Distance to calculate difference between sequences. This distance is the number of single character edits it would take to transform one string into another. Therefore, the smaller the edit distance, the more similar two strings are.

You can install it using :

pip via PyPl

pip install fuzzywuzzy

pip via Github

pip install git+git://github.com/seatgeek/fuzzywuzzy.git@0.18.0#egg=fuzzywuzzy

**Usage of FuzzyWuzzy**

Lets, import the main module:

from fuzzywuzzy import fuzz

from fuzzywuzzy import process

1. To get the similarity score between two strings, we can use methods **ratio()** and **partial\_ratio()**

In[] > fuzz.ratio("I am from Omaha", "I am from Omaha NE")

Out[] > 91

In[] > fuzz.partial\_ratio("I am from Omaha", "I am from Omaha NE")

Out[] > 100

It seems ratio method is confused by state name "NE", but actually both the string refer the same context, which is covered by partial\_ratio.

**More examples:**

fuzz.ratio("Python", "Data Wrangling with Python") ==> 38

fuzz.ratio("Python", "Wrangling with Python") ==> 44

fuzz.ratio("Python", "with Python") ==> 71

fuzz.ratio("Python", "python") ==> 86

fuzz.ratio("Python", "Python") ==> 100

2. **Token\_\*** function split the string into white-spaces, convert everything into lowercase, sorting alphabetically and ignore non-numeric non-alpha characters.

In[] > fuzz.token\_sort\_ratio('Gourav Verma', 'Gourav K. Verma')

Out[] > 92

In[] > fuzz.token\_set\_ratio('Gourav Verma', 'Gourav K. Verma')

Out[] > 100

In[] > fuzz.token\_sort\_ratio('Gourav K Verma', 'Gourav K. Verma')

Out[] > 100

In[] > fuzz.token\_set\_ratio('Gourav K Verma', 'Gourav K. Verma')

Out[] > 100

3. **Process :**It helps to extract the matching string based on certain conditions.

In[] > options = ["Atlanta Falcons", "New York Jets", "New York Giants", "Dallas Cowboys"]

In[] > process.extract("new york jets", options, limit=2)

Out[] > [('New York Jets', 100), ('New York Giants', 79)]

In[] > process.extractOne("cowboys", options)

Out[] > ('Dallas Cowboys', 90)

The fuzzywuzzy python library is a better way to do fuzzy string matching between record sets for small data sets. However, as the record sets grow large, the required processing time can grow beyond what is acceptable for the application. An alternative way is to use the NLP technique of tf-idf combined with k-NN to find matching strings using n-grams. This has the capacity to match data sets in a fraction of the time.  
   
   
***References:***

<https://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/>

<https://github.com/seatgeek/fuzzywuzzy>

<https://caserta.com/data-blog/string-matching-record-linkage-python-strategies/>