

# Package ‘levitate’

October 1, 2023

**Type** Package

**Title** Fuzzy String Comparison

**Version** 0.2.0

**Description** Provides string similarity calculations inspired by the Python 'thefuzz' package. Compare strings by edit distance, similarity ratio, best matching substring, ordered token matching and set-based token matching. A range of edit distance measures are available thanks to the 'stringdist' package.

**License** GPL-3

**URL** <https://github.com/lewinfox/levitate/>,  
<https://www.lewinfox.com/levitate/>

**BugReports** <https://github.com/lewinfox/levitate/issues>

**Depends** R (>= 2.10)

**Imports** rlang, stringdist

**Suggests** glue, knitr, pkgdown, rmarkdown, styler, testthat

**VignetteBuilder** knitr

**Encoding** UTF-8

**Language** en-GB

**LazyData** true

**RoxygenNote** 7.2.3

**NeedsCompilation** no

**Author** Lewin Appleton-Fox [aut, cre, cph]

**Maintainer** Lewin Appleton-Fox <lewin.a.f@gmail.com>

**Repository** CRAN

**Date/Publication** 2023-10-01 00:00:02 UTC

## R topics documented:

lev_best_match . . . . .	2
lev_distance . . . . .	3
lev_partial_ratio . . . . .	4
lev_ratio . . . . .	5
lev_score_multiple . . . . .	6
lev_token_set_ratio . . . . .	6
lev_token_sort_ratio . . . . .	8
lev_weighted_token_ratio . . . . .	9
lev_weighted_token_set_ratio . . . . .	10
lev_weighted_token_sort_ratio . . . . .	10

<b>Index</b>	<b>12</b>
--------------	-----------

---

lev_best_match	<i>Get the best matched string from a list of candidates</i>
----------------	--

---

### Description

Given an input string and multiple candidates, return the candidate with the best score as calculated by `.fn`.

### Usage

```
lev_best_match(input, candidates, .fn = lev_ratio, ..., decreasing = TRUE)
```

### Arguments

<code>input</code>	A single string
<code>candidates</code>	One or more candidate strings to score
<code>.fn</code>	The scoring function to use, as a string or function object. Defaults to <code>lev_ratio()</code> .
<code>...</code>	Additional arguments to pass to <code>.fn</code> .
<code>decreasing</code>	If TRUE (the default), the candidate with the highest score is ranked first. If using a comparison <code>.fn</code> that computes <i>distance</i> rather than similarity, or if you want the worst match to be returned first, set this to FALSE.

### Value

A string

### See Also

[lev\\_score\\_multiple\(\)](#)

### Examples

```
lev_best_match("bilbo", c("frodo", "gandalf", "legolas"))
```

---

lev_distance	<i>String distance metrics</i>
--------------	--------------------------------

---

### Description

Uses `stringdist::stringdistmatrix()` to compute a range of [string distance metrics](#).

### Usage

```
lev_distance(a, b, pairwise = TRUE, useNames = TRUE, ...)
```

### Arguments

a, b	The input strings
pairwise	Boolean. If TRUE, only the pairwise distances between a and b will be computed, rather than the combinations of all elements.
useNames	Boolean. Use input vectors as row and column names?
...	Additional arguments to be passed to <code>stringdist::stringdistmatrix()</code> or <code>stringdist::stringsimmatrix()</code> .

### Value

A numeric scalar, vector or matrix depending on the length of the inputs. See "Details".

### Details

This is a thin wrapper around `stringdist::stringdistmatrix()` and mainly exists to coerce the output into the simplest possible format (via `lev_simplify_matrix()`).

The function will return the simplest possible data structure permitted by the length of the inputs a and b. This will be a scalar if a and b are length 1, a vector if either (but not both) is length > 1, and a matrix otherwise.

### Other options

In addition to `useNames` `stringdist::stringdistmatrix()` provides a range of options to control the matching, which can be passed using `...`. Refer to the `stringdist` documentation for more information.

### Examples

```
lev_distance("Bilbo", "Frodo")

lev_distance("Bilbo", c("Frodo", "Merry"))

lev_distance("Bilbo", c("Frodo", "Merry"), useNames = FALSE)

lev_distance(c("Bilbo", "Gandalf"), c("Frodo", "Merry"))
```

---

lev_partial_ratio	<i>Ratio of the best-matching substring</i>
-------------------	---

---

### Description

Find the best `lev_ratio()` between substrings.

### Usage

```
lev_partial_ratio(a, b, pairwise = TRUE, useNames = TRUE, ...)
```

### Arguments

<code>a, b</code>	The input strings
<code>pairwise</code>	Boolean. If TRUE, only the pairwise distances between a and b will be computed, rather than the combinations of all elements.
<code>useNames</code>	Boolean. Use input vectors as row and column names?
<code>...</code>	Additional arguments to be passed to <code>stringdist::stringdistmatrix()</code> or <code>stringdist::stringsimmatrix()</code> .

### Value

A numeric scalar, vector or matrix depending on the length of the inputs.

### Details

If string a has length `len_a` and is shorter than string b, this function finds the highest `lev_ratio()` of all the `len_a`-long substrings of b (and vice versa).

### Examples

```
lev_ratio("Bruce Springsteen", "Bruce Springsteen and the E Street Band")

# Here the two "Bruce Springsteen" strings will match perfectly.
lev_partial_ratio("Bruce Springsteen", "Bruce Springsteen and the E Street Band")
```

---

lev_ratio	<i>String similarity ratio</i>
-----------	--------------------------------

---

### Description

String similarity ratio

### Usage

```
lev_ratio(a, b, pairwise = TRUE, useNames = TRUE, ...)
```

### Arguments

a, b	The input strings
pairwise	Boolean. If TRUE, only the pairwise distances between a and b will be computed, rather than the combinations of all elements.
useNames	Boolean. Use input vectors as row and column names?
...	Additional arguments to be passed to <a href="#">stringdist::stringdistmatrix()</a> or <a href="#">stringdist::stringsimmatrix()</a> .

### Value

A numeric scalar, vector or matrix depending on the length of the inputs.

### Details

This is a thin wrapper around [stringdist::stringsimmatrix\(\)](#) and mainly exists to coerce the output into the simplest possible format (via [lev\\_simplify\\_matrix\(\)](#)).

The function will return the simplest possible data structure permitted by the length of the inputs a and b. This will be a scalar if a and b are length 1, a vector if either (but not both) is length > 1, and a matrix otherwise.

### Examples

```
lev_ratio("Bilbo", "Frodo")  
lev_ratio("Bilbo", c("Frodo", "Merry"))  
lev_ratio("Bilbo", c("Frodo", "Merry"), useNames = FALSE)  
lev_ratio(c("Bilbo", "Gandalf"), c("Frodo", "Merry"))
```

---

lev\_score\_multiple     *Score multiple candidate strings against a single input*

---

### Description

Given a single input string and multiple candidates, compute scores for each candidate.

### Usage

```
lev_score_multiple(input, candidates, .fn = lev_ratio, ..., decreasing = TRUE)
```

### Arguments

input	A single string
candidates	One or more candidate strings to score
.fn	The scoring function to use, as a string or function object. Defaults to <a href="#">lev_ratio()</a> .
...	Additional arguments to pass to .fn.
decreasing	If TRUE (the default), the candidate with the highest score is ranked first. If using a comparison .fn that computes <i>distance</i> rather than similarity, or if you want the worst match to be returned first, set this to FALSE.

### Value

A list where the keys are candidates and the values are the scores. The list is sorted according to the decreasing parameter, so by default higher scores are first.

### See Also

[lev\\_best\\_match\(\)](#)

### Examples

```
lev_score_multiple("bilbo", c("frodo", "gandalf", "legolas"))
```

---

lev\_token\_set\_ratio     *Matching based on common tokens*

---

### Description

Compare strings based on shared tokens.

### Usage

```
lev_token_set_ratio(a, b, pairwise = TRUE, useNames = TRUE, ...)
```

## Arguments

a, b	The input strings
pairwise	Boolean. If TRUE, only the pairwise distances between a and b will be computed, rather than the combinations of all elements.
useNames	Boolean. Use input vectors as row and column names?
...	Additional arguments to be passed to <code>stringdist::stringdistmatrix()</code> or <code>stringdist::stringsimmatrix()</code> .

## Value

A numeric scalar, vector or matrix depending on the length of the inputs.

## Details

Similar to `lev_token_sort_ratio()` this function breaks the input down into tokens. It then identifies any common tokens between strings and creates three new strings:

```
x <- {common_tokens}
y <- {common_tokens}{remaining_unique_tokens_from_string_a}
z <- {common_tokens}{remaining_unique_tokens_from_string_b}
```

and performs three pairwise `lev_ratio()` calculations between them (x vs y, y vs z and x vs z). The highest of those three ratios is returned.

## See Also

[lev\\_token\\_sort\\_ratio\(\)](#)

## Examples

```
x <- "the quick brown fox jumps over the lazy dog"
y <- "my lazy dog was jumped over by a quick brown fox"

lev_ratio(x, y)

lev_token_sort_ratio(x, y)

lev_token_set_ratio(x, y)
```

---

lev\_token\_sort\_ratio *Ordered token matching*

---

### Description

Compares strings by tokenising them, sorting the tokens alphabetically and then computing the [lev\\_ratio\(\)](#) of the result. This means that the order of words is irrelevant which can be helpful in some circumstances.

### Usage

```
lev_token_sort_ratio(a, b, pairwise = TRUE, useNames = TRUE, ...)
```

### Arguments

a, b	The input strings
pairwise	Boolean. If TRUE, only the pairwise distances between a and b will be computed, rather than the combinations of all elements.
useNames	Boolean. Use input vectors as row and column names?
...	Additional arguments to be passed to <a href="#">stringdist::stringdistmatrix()</a> or <a href="#">stringdist::stringsimmatrix()</a> .

### Value

A numeric scalar, vector or matrix depending on the length of the inputs.

### See Also

[lev\\_token\\_set\\_ratio\(\)](#)

### Examples

```
x <- "Episode IV - Star Wars: A New Hope"
y <- "Star Wars Episode IV - New Hope"

# Because the order of words is different the simple approach gives a low match ratio.
lev_ratio(x, y)

# The sorted token approach ignores word order.
lev_token_sort_ratio(x, y)
```



---

`lev_weighted_token_ratio`*Weighted token similarity measure*

---

### Description

Computes similarity but allows you to assign weights to specific tokens. This is useful, for example, when you have a frequently-occurring string that doesn't contain useful information. See examples.

### Usage

```
lev_weighted_token_ratio(a, b, weights = list(), ...)
```

### Arguments

<code>a, b</code>	The input strings
<code>weights</code>	List of token weights. For example, <code>weights = list(foo = 0.9, bar = 0.1)</code> . Any tokens omitted from <code>weights</code> will be given a weight of 1.
<code>...</code>	Additional arguments to be passed to <code>stringdist::stringdistmatrix()</code> or <code>stringdist::stringsimmatrix()</code> .

### Value

A float

### Details

The algorithm used here is as follows:

- Tokenise the input strings
- Compute the edit distance between each pair of tokens
- Compute the maximum edit distance between each pair of tokens
- Apply any weights from the `weights` argument
- Return  $1 - (\text{sum}(\text{weighted\_edit\_distances}) / \text{sum}(\text{weighted\_max\_edit\_distance}))$

### See Also

Other weighted token functions: `lev_weighted_token_set_ratio()`, `lev_weighted_token_sort_ratio()`

### Examples

```
lev_weighted_token_ratio("jim ltd", "tim ltd")
```

```
lev_weighted_token_ratio("tim ltd", "jim ltd", weights = list(ltd = 0.1))
```

---

lev\_weighted\_token\_set\_ratio

*Weighted version of lev\_token\_set\_ratio()*

---

### Description

Weighted version of lev\_token\_set\_ratio()

### Usage

```
lev_weighted_token_set_ratio(a, b, weights = list(), ...)
```

### Arguments

a, b	The input strings
weights	List of token weights. For example, <code>weights = list(foo = 0.9, bar = 0.1)</code> . Any tokens omitted from weights will be given a weight of 1.
...	Additional arguments to be passed to <code>stringdist::stringdistmatrix()</code> or <code>stringdist::stringsimmatrix()</code> .

### Value

Float

### See Also

[lev\\_token\\_set\\_ratio\(\)](#)

Other weighted token functions: [lev\\_weighted\\_token\\_ratio\(\)](#), [lev\\_weighted\\_token\\_sort\\_ratio\(\)](#)

---

lev\_weighted\_token\_sort\_ratio

*Weighted version of lev\_token\_sort\_ratio()*

---

### Description

This function tokenises inputs, sorts tokens and computes similarities for each pair of tokens. Similarity scores are weighted based on the weights argument, and a total similarity score is returned in the same manner as [lev\\_weighted\\_token\\_ratio\(\)](#).

### Usage

```
lev_weighted_token_sort_ratio(a, b, weights = list(), ...)
```

**Arguments**

a, b	The input strings
weights	List of token weights. For example, <code>weights = list(foo = 0.9, bar = 0.1)</code> . Any tokens omitted from weights will be given a weight of 1.
...	Additional arguments to be passed to <code>stringdist::stringdistmatrix()</code> or <code>stringdist::stringsimmatrix()</code> .

**Value**

Float

**See Also**

[lev\\_token\\_sort\\_ratio\(\)](#)

Other weighted token functions: [lev\\_weighted\\_token\\_ratio\(\)](#), [lev\\_weighted\\_token\\_set\\_ratio\(\)](#)

# Index

## \* **weighted token functions**

- lev\_weighted\_token\_ratio, 9
- lev\_weighted\_token\_set\_ratio, 10
- lev\_weighted\_token\_sort\_ratio, 10

- lev\_best\_match, 2
- lev\_best\_match(), 6
- lev\_distance, 3
- lev\_partial\_ratio, 4
- lev\_ratio, 5
- lev\_ratio(), 2, 4, 6–8
- lev\_score\_multiple, 6
- lev\_score\_multiple(), 2
- lev\_simplify\_matrix(), 3, 5
- lev\_token\_set\_ratio, 6
- lev\_token\_set\_ratio(), 8, 10
- lev\_token\_sort\_ratio, 8
- lev\_token\_sort\_ratio(), 7, 11
- lev\_weighted\_token\_ratio, 9, 10, 11
- lev\_weighted\_token\_ratio(), 10
- lev\_weighted\_token\_set\_ratio, 9, 10, 11
- lev\_weighted\_token\_sort\_ratio, 9, 10, 10

- string distance metrics, 3
- stringdist::stringdistmatrix(), 3–5, 7–11
- stringdist::stringsimmatrix(), 3–5, 7–11