Package ‘genius’

May 28, 2020

Title  Easily Access Song Lyrics from Genius.com
Version  2.2.2
Description  Easily access song lyrics in a tidy way.
URL  https://github.com/josiahparry/genius
BugReports  https://github.com/josiahparry/genius/issues
Depends  R (>= 3.1.2)
Imports  dplyr (>= 0.7.0), rvest, stringr, tidyr, purrr, readr, tibble,
         tidytext, reshape2, rlang
License  MIT + file LICENSE
Encoding  UTF-8
LazyData  true
RoxygenNote  7.1.0
Suggests  knitr, rmarkdown, testthat
VignetteBuilder  knitr
NeedsCompilation  no
Author  Josiah Parry [aut, cre],
       Nathan Barr [aut, ctb],
       Chris Billingham [ctb],
       Evan Oppenheimer [ctb]
Maintainer  Josiah Parry <josiah.parry@gmail.com>
Repository  CRAN
Date/Publication  2020-05-28 16:10:02 UTC

R topics documented:

    add_genius ................................................. 2
    calc_self_sim ............................................ 3
    cleaning .................................................. 4
    genius_album ............................................. 4
add_genius

Description

This function is to be used to build on a data frame with artist and album/track information. To use the function with a data frame of mixed type (albums and tracks), create another column that specifies type. The type values are "album" and "lyrics".

Usage

```
add_genius(data, artist, title, type = c("album", "track", "lyrics"))
```

Arguments

data This is a dataframe with one column for the artist name, and the other column being either the track title or the album title.
artist This is the column which has artist title information
title This is the column that has either album titles, track titles, or both.
type This is a single value character string of either "album" or "track". This tells the function what kind of lyrics to pull. Alternatively, this can be a column with the value of "album" or "track" associated with each row. "lyric" can be used for backward compatibility.

Examples

```
# Albums only
artist_albums <- tibble::tribble(
  ~artist, ~album,
  "J. Cole", "KOD",
  "Sampha", "Process"
)
add_genius(artist_albums, artist, album, "album")
```
# Individual Tracks only

```r
calc_self_sim
artist_songs <- tibble::tribble(
  ~artist, ~track,
  "J. Cole", "Motiv8",
  "Andrew Bird", "Anonanimal"
)

calc_self_sim(artist_songs, artist, track, type = "track")
```

# Tracks and Albums

```r
calc_self_sim
mixed_type <- tibble::tribble(
  ~artist, ~album, ~type,
  "J. Cole", "KOD", "album",
  "Andrew Bird", "Proxy War", "track"
)

calc_self_sim(mixed_type, artist, album, type)
```

---

**calc_self_sim**  
*Calculate a self-similarity matrix*

**Description**

Calculate the self-similarity matrix for song lyrics.

**Usage**

```r
calc_self_sim(
  df, 
  lyric_col, 
  output = "tidy", 
  remove_stop_words = FALSE, 
  language = "en", 
  source = "snowball"
)
```

**Arguments**

- `df`  
  The data frame containing song lyrics. Usually from the output of `genius_lyrics()`.  
- `lyric_col`  
  The unquoted name of the column containing lyrics  
- `output`  
  Determine the type of output. Default is "tidy". Set to "matrix" for the raw matrix.
remove_stop_words
- Optional argument to remove stop words from self-similarity matrix.

language
- Language of stop words. See tidytext::get_stopwords().

source
- Stop words source. See tidytext::get_stopwords().

Examples

```r
## Not run:
bad_habits <- genius_lyrics("Alix", "Bad Habits")
self_sim <- calc_self_sim(bad_habits, lyric)
## End(Not run)
```

cleaning
- Function which produces a vector to be used in string cleaning from scraping. There are a lot of hard-coded values in here and will need to be adapted for the weird nuances.

Description
- Function which produces a vector to be used in string cleaning from scraping. There are a lot of hard-coded values in here and will need to be adapted for the weird nuances.

Usage
- cleaning()

---

**genius_album**

*Retrieve song lyrics for an album*

Description
- Obtain the lyrics to an album in a tidy format.

Usage
- genius_album(artist = NULL, album = NULL, info = "simple")

Arguments
- **artist**
  - The quoted name of the artist. Spelling matters, capitalization does not.
- **album**
  - The quoted name of the album. Spelling matters, capitalization does not.
- **info**
  - Return track level metadata. See details.
Details

The ‘info’ argument returns additional columns to the returned tibble: “simple” returns only the song lyrics. “title” returns the track title and lyrics. “artist” returns the lyrics and artist. “features” returns the lyrics, song elements, and element artists. “all” returns all of the above mentioned, plus appends the album name.

Examples

## Not run:
genius_album(artist = "Petal", album = "Comfort EP")
genius_album(artist = "Fit For A King", album = "Deathgrip", info = "all")

## End(Not run)

---

## genius_lyrics

Retrieve song lyrics from Genius.com

Description

Retrieve the lyrics of a song with supplied artist and song name.

Usage

    genius_lyrics(artist = NULL, song = NULL, info = "title")

Arguments

artist  The quoted name of the artist. Spelling matters, capitalization does not.
song    The quoted name of the song. Spelling matters, capitalization does not.
info    Default "title", returns the track title. Set to "simple" for only lyrics, "artist" for the lyrics and artist, "features" for song element and the artist of that element, "all" to return artist, track, line, lyric, element, and element artist.

Examples

    genius_lyrics(artist = "Margaret Glaspy", song = "Memory Street")
    genius_lyrics(artist = "Kendrick Lamar", song = "Money Trees")
    genius_lyrics("JMSN", "Drinkin")
**genius_tracklist**  
*Create a tracklist of an album*

**Description**

Creates a ‘tibble’ containing all track titles for a given artist and album. This function is used internally in `genius_album()`.

**Usage**

```r
genius_tracklist(artist = NULL, album = NULL)
```

**Arguments**

- **artist**  
  The quoted name of the artist. Spelling matters, capitalization does not.

- **album**  
  The quoted name of the album. Spelling matters, capitalization does not.

**Examples**

```r
genius_tracklist(artist = "Andrew Bird", album = "Noble Beast")
```

---

**genius_url**  
*Use Genius url to retrieve lyrics*

**Description**

This function is used inside of the `genius_lyrics()` function. Given a url to a song on Genius, this function returns a tibble where each row is one line. Pair this function with `gen_song_url()` for easier access to song lyrics.

**Usage**

```r
genius_url(url, info = "title")
```

**Arguments**

- **url**  
  The url of song lyrics on Genius

- **info**  
  Default "title", returns the track title. Set to "simple" for only lyrics, "artist" for the lyrics and artist, "features" for song element and the artist of that element, "all" to return artist, track, line, lyric, element, and element artist.
**gen_album_url**

**Examples**

```r
url <- gen_song_url(artist = "Kendrick Lamar", song = "HUMBLE")
genius_url(url)

genius_url("https://genius.com/Head-north-in-the-water-lyrics", info = "all")
```

---

**gen_album_url**  
**Create Genius Album url**

**Description**

Creates a string containing the url to an album tracklist on Genius.com. The function is used internally to `genius_tracklist()`.

**Usage**

```r
gen_album_url(artist = NULL, album = NULL)
```

**Arguments**

- **artist**  
  The quoted name of the artist. Spelling matters, capitalization does not.
- **album**  
  The quoted name of the album. Spelling matters, capitalization does not.

**Examples**

```r
gen_album_url(artist = "Pinegrove", album = "Cardinal")
```

---

**gen_song_url**  
**Create Genius url**

**Description**

Generates the url for a song given an artist and a song title. This function is used internally within the `genius_lyrics()` function.

**Usage**

```r
gen_song_url(artist = NULL, song = NULL)
```

**Arguments**

- **artist**  
  The quoted name of the artist. Spelling matters, capitalization does not.
- **song**  
  The quoted name of the song. Spelling matters, capitalization does not.
Examples

```python
gen_song_url(artist = "Kendrick Lamar", song = "HUMBLE")
gen_song_url("Margaret glaspy", "Memory Street")
```

---

**possible_album**

*Form of genius_album that can handle errors*

**Description**

Form of genius_album that can handle errors

**Usage**

`possible_album(...)`

**Arguments**

```python
...  # arguments that would be passed to `genius_album()`
```

---

**possible_lyrics**

*Form of genius_lyrics that can handle errors*

**Description**

Form of genius_lyrics that can handle errors

**Usage**

`possible_lyrics(...)`

**Arguments**

```python
...  # arguments that would be passed to `genius_lyrics()`
```
possible_url  
Form of genius_url that can handle errors

Description
Form of genius_url that can handle errors

Usage
possible_url(...)

Arguments
... arguments that would be passed to ‘genius_url()’

prep_info  
Prepares input strings for ‘gen_song_url()’

Description
Applies a number of regular expressions to prepare the input to match Genius url format

Usage
prep_info(input)

Arguments
input Either artist, song, or album, function input.
Index

add_genius, 2
calc_self_sim, 3
cleaning, 4

gen_album_url, 7
gen_song_url, 7
genius_album, 4
genius_lyrics, 5
genius_tracklist, 6
genius_url, 6

possible_album, 8
possible_lyrics, 8
possible_url, 9
prep_info, 9