Package ‘chorrrds’

June 30, 2020

Title Music Chords Extraction

Type Package

Version 0.1.9.5

Description Extracts music chords from the 'CifraClub' website <https://www.cifraclub.com.br/>.
The package also has functions for cleaning the extracted data and
feature extraction.

Depends R (>= 2.10)

Suggests ggplot2, knitr, network, covr, testthat

URL `https://github.com/r-music/chorrrds`

BugReports `https://github.com/r-music/chorrrds/issues`

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.0.0

Imports stringr, dplyr, xml2, rvest, magrittr, purrr, forcats, rlang

NeedsCompilation no

Author Bruna Wundervald [aut, cre] (<https://orcid.org/0000-0001-8163-220X>),
Matthew Leonawicz [ctb] (<https://orcid.org/0000-0001-9452-2771>),
Luca Carbone [ctb] (<https://orcid.org/0000-0003-1688-9468>)

Maintainer Bruna Wundervald <`brunadaviesw@gmail.com`>

Repository CRAN

Date/Publication 2020-06-30 17:30:02 UTC

R topics documented:

`all` *all*

Description

All data available.

Usage

```
all
```

Format

A data frame with 6 variables:

`date` integer.The date of the album which contains the music.

`music` factor. The name of the music.

`popul` integer. The popularity of the music.

`chord` factor. The chord names of each music, by order of occurrence in the music.

`key` factor. The key for each music.

`artist` factor. The name of the artist

`chords_ngram` *chords_ngram*

Description

Builds chords ngrams for a chord datdaset.

Usage

```
chords_ngram(data, n = 2)
```

Arguments

`data`	dataframe. The chords dataset to exract the features from.
`n`	numeric. The number of grams. The default is 2 (bigram).

Value

A chords dataset added with the chords ngram.

Examples

```
{

  songs <- chorrrds::get_songs("tim-maia")
  chords <- get_chords(songs$url[4])
  chords_ngram(chords)

}
```

`clean` *clean*

Description

Clean data when there is some excessive long text on a column.

Usage

```
clean(data, column = "chord", long = 15, message = TRUE)
```

Arguments

`data`	a data.frame.
`column`	string. The column by which we want to make the cleaning.
`long`	numeric. The longest string we wish exists on our
`message`	logical. Should the function print how many lines were removed?

Value

A database, with the text cleaning done.

Examples

```
{
## Not run:
data("caetano")
  clean(data  = caetano, column = "chord", long = 15, message = TRUE)

## End(Not run)
}
```

---

create_dat *create_dat*

Description

Break song by verse with chords and corresponding lyrics.

Usage

```
create_dat(artist, track)
```

Arguments

`artist`	character. The artist's name.
`track`	character. The song's title.

Value

An object of type 'data.frame' with the song chords and lyrics is retuned. The object is to be later used in the 'create_net()' function to get accurate connections between chords and words.

Examples

```
{

  create_dat("The Weeknd", "Acquainted")

}
```

`create_net` *create_net*

Description

Match music lyrics with the corresponding chords.

Usage

```
create_net(chords_dat)
```

Arguments

`chords_dat`	data frame. A data frame as produced by the ‘create_dat()‘ function with chords in the first column and lyrics in the second column.

Value

An object of type ‘tibble‘ with the song chords and lyrics is retuned. Each chord is linked to the words that are sung when that chord is played.

Examples

```
{

  chords_dat <- create_dat("The Weeknd", "Acquainted")
  create_net(chords_dat)

}
```

`deg_maj` *deg_maj*

Description

Accessory data with the chords present in each scale, with its respective degrees, for the minor cases.

Usage

```
deg_maj
```

Format

An object of class `data.frame` with 7 rows and 18 columns.

`deg_min` *deg_min*

Description

Accessory data with the chords present in each scale, with its respective degrees, for the minor cases.

Usage

```
deg_min
```

Format

An object of class `data.frame` with 7 rows and 16 columns.

`dist` *dist*

Description

A simple measure of the chords distances in the circle of fifths.

Usage

```
dist
```

Format

A data frame with 3 variables:

`prox` factor. The chord.

`dist` numeric. The distance from C in the circle of fifths.

`order` integer. The order in the circle of fifths.

`eqv` *eqv*

Description

Accessory data for the recognition of equivalent keys, including major and minor relatives.

Usage

```
eqv
```

Format

A data frame with 3 variables:

`key` factor. Keys ordered by the circle of fifths.

`minor.rel` factor. Relative minors of the key in the previous column.

`rep` num. A number indicating if the key scale is equivalent to some other; repeated numbers indicate equivalent keys.

`feature_extraction` *feature_extraction*

Description

Extracts features from a chords dataset.

Usage

```
feature_extraction(data)
```

Arguments

`data`	dataframe. The chords dataset to exract the features from.

Value

A dataframe with the chords set added with logical features (1 or 0), to indicate if each chord is:

Examples

```
{

  songs <- get_songs("tim-maia")
  chords <- get_chords(songs$url[4])
  feature_extraction(chords)

}
```

`genre` *genre*

Description

Accessory data with the genre for each artist in the package.

Usage

```
genre
```

Format

An object of class `data.frame` with 106 rows and 2 columns.

`get_chords` *get_chords*

Description

Extracts music chords from an artist.

Usage

```
get_chords(song_url, nf = FALSE)
```

Arguments

`song_url`	The song URLs to be used for the chords collection. Can be either a character vector or straightforwardly the result of the ‘get_songs()‘ function.
`nf`	logical. If the chords of a song are not found, should we return this information in the final result?

Value

An object of type ‘tibble‘ with the chords sequences, key, song names and name of the artist.

Examples

```
{

  songs <- get_songs("tim-maia")
  get_chords(songs$url[2])

}
```

get_songs *get_songs*

Description

Get songs names and URLs for an artist.

Usage

```
get_songs(artist)
```

Arguments

artist	character. The artist's name.

Value

If the artist (or band) is found, an object of type 'tibble' with the song names, URLs and artist is retuned. The URLs are to be later used in the 'get_chords()' function.

Examples

```
{

  get_songs("jorge")
  get_songs("los-hermanos")

}
```

search_data *search_data*

Description

Search artists in the available package database.

Usage

```
search_data(name)
```

Arguments

name	character. The searched artist's name.

Value

If a database with the corresponding searched name is found, it's name is returned. If not, nothing is returned.

Examples

```
{
   search_data("chico")

}
```

simplify_chords *simplify_chords*

Description

Simplifies music chords extracted with the chords package, eliminating chords extensions, such as 4th, 5th, 6th, 7th, 9th, sus. It leaves the chords in the simplest format possible.

Usage

```
simplify_chords(data)
```

Arguments

`data`	character. The chords to be simplified.

Value

The dataset with a new column called "chord_simplified" with the simplified version of the chords.

Examples

```
{

  songs <- get_songs("tim-maia")
  chords <- get_chords(songs$url[2])
  simplify_chords(chords)

}
```

Index