

Package ‘sdam’

October 29, 2020

Type Package

Title Digital tools for the SDAM project at Aarhus University

Description

This package provides tools for performing analyses within Social Dynamics and complexity in the Ancient Mediterranean (SDAM), which is a research group based at Aarhus University.

Version 0.3.0

Date 2020-10-29

Depends R (>= 3.6.0)

Suggests multigraph, httr, rjson

Author Antonio Rivero Ostoic [aut, cre],
Adela Sobotkova [ctb],
Vojtech Kase [ctb],
Petra Hermankova [ctb]

Maintainer Antonio Rivero Ostoic <jaro@cas.au.dk>

License CC BY 4.0

R topics documented:

sdam-package	1
EDH	2
edhw	3
get.edh	5
get.edhw	9
request	10
simil	12

Index	13
--------------	-----------

sdam-package	<i>Digital tools for the SDAM project at Aarhus University</i>
--------------	--

Description

This package provides tools for performing analyses within Social Dynamics and complexity in the Ancient Mediterranean (SDAM), which is a research group based at Aarhus University.

Details

Package: `sdam`
 Type: `Package`
 Version: `0.3.0`
 Date: `29 October 2020`
 License: `CC BY-SA 4.0`

Currently, it is possible with the `sdam` package to access data from the Epigraphic Database Heidelberg API with `get.edh()`, and the wrapper functions `get.edhw()` `edhw()` as well. Most of the data is available in the dataset attached to the package, called `EDH`, and which can be manipulated by using the `edhw()` convenient function.

Besides, the `request()` function allows performing different types of HTTP requests from a cloud repository like DEiC'S <https://sciencedata.dk> or another customized URL address.

Similarity by simple matching among column vectors is achieved by the `simil()` function in order to make analyses of relations between assemblages and artifacts.

Note that this latter function still under development.

Author(s)

Author: Antonio Rivero Ostoic [aut, cre], Adela Sobotkova [ctb], Vojtech Kase [ctb], Petra Hermankova [ctb]

Maintainer: Antonio Rivero Ostoic <jaro@cas.au.dk>

See Also

[multigraph](#)

EDH

Epigraphic Database Heidelberg Data Set

Description

This is a data set retrieved from the Epigraphic Database Heidelberg API repository.

Usage

```
data("EDH")
```

Format

A list object of 82464 items (until 20-11-2019) with at least one of the following 46 (or more) names in the EDH list:

```
"ID", "commentary", "fotos", "country", "depth", "diplomatic_text", "edh_geography_uri",
"findspot", "findspot_ancient", "findspot_modern", "geography", "height", "id", "language",
"last_update", "letter_size", "literature", "material", "military", "modern_region",
"not_after", "not_before", "people" (which is a list with: "person_id", "nomen", "cognomen",
"praenomen", "name", "gender", "status", "tribus", "origo", "occupation", "age: years",
"age: months", "age: days"), "present_location", "province_label", "religion",
"responsible_individual", "social_economic_legal_history", "transcription", "trismegistos_uri",
"type_of_inscription", "type_of_monument", "uri", "width", "work_status", and "year_of_find".
```

Source

<https://edh-www.adw.uni-heidelberg.de/data/api>

See Also

[get.edh](#), [get.edhw](#), [edhw](#)

edhw	<i>Wrapper function for manipulation of EDH dataset</i>
------	---

Description

A function to obtain variable data and perform transformations on the Epigraphic Database Heidelberg EDH dataset.

Usage

```
edhw(vars, x = NULL, as = c("list", "df"), type = c("long", "wide", "narrow"), split,
      select, addID, limit, id, na.rm, ...)
```

Arguments

<code>vars</code>	Choose variables of interest from <code>x</code> . If <code>x=NULL</code> , the entire EDH dataset is taken. (optional, vector)
<code>x</code>	A list object name with fragments of the EDH dataset (optional)
<code>as</code>	Format to return the output. Currently either as a "list" or a data frame "df" object.
<code>type</code>	Type format of data frame. Currently either "long" or "wide" ("narrow" not yet implemented)
<code>split</code>	Divide the data into groups by id? (logical and optional)
<code>select</code>	select 'people' variables? (logical and optional, for data frame "long" only)
<code>addID</code>	Add identification to the output? (optional and logical)
<code>limit</code>	Limit the returned output. Ignored if <code>id</code> is specified (optional, integer or vector)
<code>id</code>	Select only the <code>hd_nr</code> id(s) (optional, integer or character)
<code>na.rm</code>	Remove entries with NA data? (logical and optional)
<code>...</code>	Optional arguments if needed.

Details

This is a convenient function to "extract" *variables* from the EDH dataset attached to this package. However, the input in `x` can be fragments of the EDH dataset or from the the Epigraphic Database Heidelberg API obtained by functions `get.edh()` or `get.edhw()` with the `rjson` format. When `x` is explicited, it must be at least a list object with a comparable structure to the EDH dataset.

Through `vars` argument and return the output either as a list with `list` or a data frame with `df`, and when argument `vars` is missing, then all entries in `x` are taken.

By default, a list object is returned, with or without an ID identification provided by the `addID` argument. When the input list is converted into a data frame, the ordering of the variables is given

alphabetically. If desired, it is also possible to remove missing data from the output by activating `na.rm` and work with complete cases.

Arguments `id` and `limit` serve to reduce the returned output either to some Epigraphic Database number or numbers, which are specified by `hd_nr`, or else by limiting the amount of the returned output. `limit` here is like the `limit` argument of function `get.edh()`, but in this case the offset can be specified as a sequence. While "`limit`" is a faster way to get to entries in the EDH dataset, argument `id` is for referring to precisely one or more `hd_nrs` in the Epigraphic Database Heidelberg API.

Component "`people`" is a separated list in the EDH dataset, and it should be considered as a separate case from the rest of the variables. In the case that the output is a data frame, the default output is a 'long' type table; that is records can appear in different rows and each variable is assigned into a single column, and with this option is possible to select "`people`" variables.

By setting "`wide`" in type, it is possible to place the different people from a single entry column by column in the data frame and each record has a single row. Finally, argument `split` allows dividing the data in the data frame into groups by '`id`', which corresponds to the HD number of inscription in the EDH dataset.

Value

A list or a data frame, depending on the input.

Note

In case that the output chosen is a dataframe, then component "`people`" should be chosen alone.

Author(s)

Antonio Rivero Ostoic

References

<https://edh-www.adw.uni-heidelberg.de/data/api>

See Also

[get.edh](#), [get.edhw](#)

Examples

```
## Not run:
## load data set
data(EDH)

## make a list for three variables in 'EDH' (default)
edhw(vars=c("type_of_inscription", "not_after", "not_before") )

## select 'gender' from 'people' in 'EDH' (default)
edhw(vars=c("people", "not_after", "not_before"), select="gender" )
## End(Not run)
```

get.edh

*Get data from the Epigraphic Database Heidelberg API***Description**

A function to obtain data from the Epigraphic Database Heidelberg API repository.

Usage

```
get.edh(search = c("inscriptions", "geography"),
        url = "https://edh-www.adw.uni-heidelberg.de/data/api",
        hd_nr, province, country, findspot_modern, findspot_ancient,
        year_not_before, year_not_after, tm_nr, transcription, type,
        bbox, findspot, pleiades_id, geonames_id, offset, limit,
        maxlimit = 4000, addID, printQ)
```

Arguments

search	Whether the search is on inscriptions <i>or</i> on geography.
url	Open data repository API
hd_nr	HD number of inscription
province	Ancient Roman province name
country	Actual country name
findspot_modern	Actual location name findspot
findspot_ancient	Ancient location name findspot
year_not_before	Year, not before (integer, BC years are negative)
year_not_after	Year, not after (integer, BC years are negative)
tm_nr	Trismegistos' database number (?)
transcription	Automatic leading and trailing truncation (brackets are ignored)
type	Type of inscription (case insensitive)
bbox	Bounding box with character format bbox = "minLong,minLat,maxLong,maxLat"
findspot	Level of village, street etc. (add leading and/or trailing)
pleiades_id	Pleiades identifier of a place (integer)
geonames_id	Geonames identifier of a place (integer)
offset	Clause to specify which row to start from retrieving data (optional and integer)
limit	Clause to limit the number of results (optional and integer)
maxlimit	Maximum limit of the query (integer, default 4000)
addID	Add identification to the output? (optional and logical)
printQ	Also print query? (optional and logical)

Details

Since with the `inscriptions` option the `id` "component" of the output list is not with a numeric format, then the function adds an ID at the beginning of the list with the identifier with a numerical format.

Notice that `hd_nr` is not the same as `ID` nor `id`.

Use function `get.edhw` in case you want to grab several items.

Entries in `country` are abbreviated country names where the inscription was located. A list with the of valid values for countries from the EDH API are

"ad"	Andorra	"gr"	Greece	"pl"	Poland
"al"	Albania	"hr"	Croatia	"pt"	Portugal
"am"	Armenia	"hu"	Hungary	"rks"	Kosovo
"at"	Austria	"il"	Israel	"ro"	Romania
"az"	Azerbaijan	"iq"	Iraq	"rs"	Serbia
"ba"	Bosnia and Herzegovina	"it"	Italy	"ru"	Russia
"be"	Belgium	"jo"	Jordan	"sa"	Saudi Arabia
"bg"	Bulgaria	"kg"	Kyrgyzstan	"sd"	Sudan
"ch"	Switzerland	"kz"	Kazakhstan	"se"	Sweden
"cy"	Cyprus	"lb"	Lebanon	"si"	Slovenia
"cz"	Czech Republic	"li"	Liechtenstein	"sk"	Slovakia
"de"	Germany	"lu"	Luxembourg	"sm"	San Marino
"dk"	Denmark	"ly"	Libyan Arab Jamahiriya	"sy"	Syrian Arab Republic
"dz"	Algeria	"ma"	Morocco	"tj"	Tajikistan
"eg"	Egypt	"mc"	Monaco	"tn"	Tunisia
"es"	Spain	"md"	Moldova	"tr"	Turkey
"fr"	France	"me"	Montenegro	"ua"	Ukraine
"gb"	United Kingdom	"mk"	Macedonia	"uz"	Uzbekistan
"ge"	Georgia	"mt"	Malta	"va"	Vatican City State
"gi"	Gibraltar	"nl"	Netherlands	"ye"	Yemen

And for the ancient Roman provinces the valid values are

"Ach"	Achaia	"Cor"	Corsica	"Mes"	Mesopotamia
"Aeg"	Aegyptus	"Cre"	Creta	"MoI"	Moesia inferior
"Aem"	Aemilia (Regio VIII)	"Cyp"	Cyprus	"MoS"	Moesia superior
"Afr"	Africa Proconsularis	"Cyr"	Cyrene	"Nar"	Narbonensis
"AlC"	Alpes Cottiae	"Dac"	Dacia	"Nor"	Noricum
"AlG"	Alpes Graiae	"Dal"	Dalmatia	"Num"	Numidia
"AlM"	Alpes Maritimae	"Epi"	Epirus	"PaI"	Pannonia inferior
"AlP"	Alpes Poeninae	"Etr"	Etruria (Regio VII)	"PaS"	Pannonia superior
"ApC"	Apulia et Calabria (Regio II)	"Gal"	Galatia	"Pic"	Picenum (Regio V)
"Aqu"	Aquitania	"GeI"	Germania inferior	"Rae"	Raetia
"Ara"	Arabia	"GeS"	Germania superior	"ReB"	Regnum Bospori
"Arm"	Armenia	"HiC"	Hispania citerior	"Rom"	Roma
"Asi"	Asia	"Inc"	Provincia incerta	"Sam"	Samnium (Regio IV)
"Ass"	Assyria	"Iud"	Iudaea	"Sar"	Sardinia
"Bae"	Baetica	"LaC"	Latium et Campania (Regio I)	"Sic"	Sicilia, Melita
"Bar"	Barbaricum	"Lig"	Liguria (Regio IX)	"Syr"	Syria
"Bel"	Belgica	"Lug"	Lugdunensis	"Thr"	Thracia
"BiP"	Bithynia et Pontus	"Lus"	Lusitania	"Tra"	Transpadana (Regio XI)

"BrL"	Bruttium et Lucania (Regio III)	"LyP"	Lycia et Pamphylia	"Tri"	Tripolitania
"Bri"	Britannia	"MaC"	Mauretania Caesariensis	"Umb"	Umbria (Regio VI)
"Cap"	Cappadocia	"MaT"	Mauretania Tingitana	"Val"	Valeria
"Cil"	Cilicia	"Mak"	Macedonia	"VeH"	Venetia et Histria (Regio X)

Value

A list object with at least one the following items:

"ID" (Optional), only if addID is set to TRUE.

"commentary"

"fotos"

"country"

"depth"

"diplomatic_text"

"edh_geography_uri"

"findspot"

"findspot_ancient"

"findspot_modern"

"geography"

"height"

"id"

"language"

"last_update"

"letter_size"

"literature"

"material"

"military"

"modern_region"

"not_after"

"not_before"

"people" This item is another list with at least one the following items:

"person_id"

"nomen"

"cognomen"

"praenomen"

"name"

```

        "gender"
        "status"
        "tribus"
        "origo"
        "occupation"
        "age: years"
        "age: months"
        "age: days"

    "present_location"

    "religion"
    "province_label"

    "responsible_individual"

    "social_economic_legal_history"

    "transcription"

    "trismegistos_uri"

    "type_of_inscription"

    "type_of_monument"

    "uri"
    "width"
    "work_status"
    "year_of_find"

```

And also the query is printed if specified by printQ.

Note

The other two search options from the [EDH] database [API], which are "photos" and "bibliography" may be implemented in the future.

Author(s)

Antonio Rivero Ostoic

References

<https://edh-www.adw.uni-heidelberg.de/data/api>
<https://edh-www.adw.uni-heidelberg.de/data/api/terms/country>
<https://edh-www.adw.uni-heidelberg.de/data/api/terms/province>

See Also[get.edhw](#), [simil](#)**Examples**

```
## get inscriptions from EDH API data
## Not run:
get.edh(findspot_modern="madrid")
## End(**Not run**)
```

`get.edhw`*Wrapper to get data from the Epigraphic Database Heidelberg API*

Description

A wrapper function to obtain data from the Epigraphic Database Heidelberg repository.

Usage

```
get.edhw(hd_nr, ...)
```

Arguments

hd_nr	HD number of inscriptions
...	Additional arguments

Details

This is a wrapper function to obtain a sample data from the Epigraphic Database Heidelberg API repository.

Value

A list of lists object with the items described in [get.edh](#).

Note

Large samples can take a lot of time.

Author(s)

Antonio Rivero Ostoic

References

<https://edh-www.adw.uni-heidelberg.de/data/api>

See Also[get.edh](#), [simil](#)

Examples

```
## get 10 records from EDH API data
## Not run:
get.edhw(hd_nr=1:10)
## End(**Not run**)
```

request

Perform an HTTP request

Description

A function to perform an HTTP request

Usage

```
request(file, URL = "https://sciencedata.dk", method = c("GET", "POST", "PUT", "DELETE"),
        anonymous = FALSE, cred = NULL, path = "/files", subdomain = NULL, force = FALSE,
        rm.file, ...)
```

Arguments

file	the request file
URL	protocol and domain of the url
method	the http <i>verb</i> for the object
anonymous	unauthenticated user? (logical)
cred	authentication credentials (vector with username and password)
path	path to add to the url (optional)
subdomain	subdomain to add to the url (optional)
force	force remote file overwriting? (optional)
rm.file	remove file in local machine? (optional and logical)
...	extra parameters if required

Details

request is basically a HTTP request, first aimed to interact with DEiC's (Danish e-Infrastructure Cooperation) <https://sciencedata.dk>. However, it is possible to specify the URL path and subdomain if necessary.

There are two types of folders in DEiC's <https://sciencedata.dk> that are *personal* and *shared* folders and both requires authentication with credentials.

The *path* to the shared folders where the files are located must be specified with the path argument. However, for personal folders is the file argument that includes the path information. Many times, DEiC's <https://sciencedata.dk> places the data on a *subdomain*, and for some methods like PUT it is required to specify the subdomain as well.

When a file already exists on the remote server, there is a prompt question for overwriting the file when the PUT method is invoked, and by activating argument force we can prevent confirmation and replace the file.

In case that accessing the server requires basic authentication, then package "[tcltk](#)" may be needed as well to input the credentials with a widget prompt, and there is the cred argument for performing a basic authentication without a prompt. Public folders can be accessed as anonymous user.

Value

Depends on the method, an action on the server site. A *Response* message is returned when the method is PUT with the url and items Date, Status, Content-Type.

Method POST is not currently supported at *sciencedata.dk*.

Note

Aliases for this function are `sddk()` and `SDDK()`.

Author(s)

Antonio Rivero Ostoic

See Also

<https://sciencedata.dk>

https://mplex.github.io/cedhar/Sciencedata_dk.html

Examples

```
## get a public file from remote server as anonymous user
## Not run:
request("filename.extension", method="GET", anonymous=TRUE)
## End(Not run)

## put a file in remote server
## Not run:
sddk("filename.extension", method="PUT")
## End(Not run)

## put an existing file in remote server and force overwriting
## Not run:
sddk("filename.extension", method="PUT", force=TRUE)
## End(Not run)

## put an existing file in remote server and remove file from local machine
## Not run:
sddk("filename.extension", method="PUT", rm.file=TRUE)
## End(Not run)

## remove a file in remote server
## Not run:
SDDK("filename.extension", method="DELETE")
## End(Not run)
```

simil	<i>Similarity between (column) vectors</i>
-------	--

Description

A function to compute the Similarity between vectors, which can arise from columns in a data frame or list entries.

Usage

```
simil(x, att, null, uniq, diag.incl)
```

Arguments

x	A list or a data frame
att	Column(s) in x representing attributes (vector)
null	Include NA or NULLs? (optional and logical)
uniq	remove duplicates? (optional and logical)
diag.incl	include entries in matrix diagonal? (optional and logical)

Details

At this point, the ID column in the input represents the labels of the nodes. In case that an ID column does not exists, then the first column is taken provided that there are not duplicated entry names.

Value

A valued matrix with similarities among units by simple matching.

Note

Other similarity measures will be added in the near future.

Author(s)

Antonio Rivero Ostoic

See Also

[get.edh](#)

Examples

```
# TBD
```

Index

- *Topic **IO**
 - `get.edh`, [5](#)
 - `get.edhw`, [9](#)
 - `request`, [10](#)
- *Topic **datasets**
 - EDH, [2](#)
- *Topic **data**
 - `edhw`, [3](#)
- *Topic **manip**
 - `edhw`, [3](#)
- *Topic **metrics**
 - `simil`, [12](#)
- *Topic **package**
 - `sdam-package`, [1](#)
- EDH, [2](#)
- `edhw`, [3](#), [3](#)
- `get.edh`, [3](#), [4](#), [5](#), [9](#), [12](#)
- `get.edhw`, [3](#), [4](#), [6](#), [9](#), [9](#)
- `multigraph`, [2](#)
- `request`, [10](#)
- `sdam (sdam-package)`, [1](#)
- `sdam-package`, [1](#)
- `simil`, [9](#), [12](#)
- SSDK (`request`), [10](#)
- `ssdk (request)`, [10](#)
- `tcltk`, [10](#)