Package 'wactor'

December 18, 2019

Title Word Factor Vectors

Version 0.0.1

Description A user-friendly factor-like interface for converting strings of text into numeric vectors and rectangular data structures.

Encoding UTF-8

LazyData true

Imports xgboost, tokenizers, text2vec, R6, utils, tibble, ggplot2, stats, Matrix

URL https://github.com/mkearney/wactor

BugReports https://github.com/mkearney/wactor/issues

RoxygenNote 7.0.2

License MIT + file LICENSE

Suggests testthat (>= 2.1.0), covr

NeedsCompilation no

Author Michael W. Kearney [aut, cre] (<https://orcid.org/0000-0002-0730-4694>), Lingshu Hu [ctb] (<https://orcid.org/0000-0003-0304-882X>)

Maintainer Michael W. Kearney <kearneymw@missouri.edu>

Repository CRAN

Date/Publication 2019-12-18 15:30:02 UTC

R topics documented:

as_wactor *As wactor*

Description

Convert data into object of type 'wactor'

Usage

```
as_wactor(.x, ...)
```

Arguments

.x	Input text vector
...	Other args passed to Wactr$new(...)

Value

An object of type wactor

dtm *Document term frequency*

Description

Converts character vector into document term matrix (dtm)

Usage

```
dtm(object, .x = NULL)
```

Arguments

object	Input object containing dictionary (column), e.g., wactor
.x	Text from which the document term matrix will be created

Value

A c-style matrix

Examples

```
## create wactor
w <- wactor(letters)

## use wactor to create dtm of same vector
dtm(w, letters)

## using the initial data is the default; so you don't actually have to
## respecify it
dtm(w)

## use wactor to create dtm on new vector
dtm(w, c("a", "e", "i", "o", "u"))

## apply directly to character vector
dtm(letters)
```

---

split_test_train *Split into test and train data sets*

Description

Randomly partition input into a list of `train` and `test` data sets

Usage

```
split_test_train(.data, .p = 0.8, ...)
```

Arguments

`.data`	Input data. If atomic (numeric, integer, character, etc.), the input is first converted to a data frame with a column name of "x."
`.p`	Proportion of data that should be used for the `train` data set output. The default value is 0.80, meaning the `train` output will include roughly 80 pct. of the input cases while the `test` output will include roughly 20 oct..
`...`	Optional. The response (outcome) variable. Uses tidy evaluation (quotes are not necessary). This is only relevant if the identified variable is categorical–i.e., character, factor, logical–in which case it is used to ensure a uniform distribution for the `train` output data set. If a value is supplied, uniformity in response level observations is prioritized over the `.p` (train proportion) value.

Value

A list with `train` and `test` tibbles (data.frames)

Examples

```
## example data frame
d <- data.frame(
  x = rnorm(100),
  y = rnorm(100),
  z = c(rep("a", 80), rep("b", 20))
)

## split using defaults
split_test_train(d)

## split 0.60/0.40
split_test_train(d, 0.60)

## split with equal response level obs
split_test_train(d, 0.80, label = z)

## apply to atomic data
split_test_train(letters)
```

---

tfidf *Term frequency inverse document frequency*

Description

Converts character vector into a term frequency inverse document frequency (TFIDF) matrix

Usage

```
tfidf(object, .x = NULL)
```

Arguments

object	Input object containing dictionary (column), e.g., wactor
.x	Text from which the tfidf matrix will be created

Value

A c-style matrix

Examples

```
## create wactor
w <- wactor(letters)

## use wactor to create tfidf of same vector
```

```
tfidf(w, letters)

## using the initial data is the default; so you don't actually have to
## respecify it
tfidf(w)

## use wactor to create tfidf on new vector
tfidf(w, c("a", "e", "i", "o", "u"))

## apply directly to character vector
tfidf(letters)
```

`wactor` *Create wactor*

Description

Create an object of type 'wactor'

Usage

```
wactor(.x, ...)
```

Arguments

`.x`	Input text vector
`...`	Other args passed to Wactr$new(...)

Value

An object of type wactor

Examples

```
## create
w <- wactor(c("a", "a", "a", "b", "b", "c"))

## summarize
summary(w)

## plot
plot(w)

## predict
predict(w)

## use on NEW data
dtm(w, letters[1:5])
```

```
## dtm() is the same as predict()
predict(w, letters[1:5])

## works if you specify 'newdata' too
predict(w, newdata = letters[1:5])
```

`Wactr` *A wactor object*

Description

A factor-like class for word vectors

Methods

Public methods:

- `Wactr$new()`
- `Wactr$clone()`

Method `new()`:

Usage:

```
Wactr$new(
  text = character(),
  tokenizer = NULL,
  max_words = 1000,
  doc_prop_max = 1,
  doc_prop_min = 0
)
```

Arguments:

`max_words` Maximum number of words in vocabulary

`doc_prop_max` Maximum proportion of docs for terms in dinctionary

`doc_prop_min` Minimum proportion of docs for terms in dictionary.

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
Wactr$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

xgb_mat *xgb matrix*

Description

Simple wrapper for creating a xgboost matrix

Usage

```
xgb_mat(x, ..., y = NULL, split = NULL)
```

Arguments

x	Input data
...	Other data to cbind
y	Label vector
split	Optional number between 0-1 indicating the desired split between train and test

Value

A xgb.Dmatrix

Examples

```
xgb_mat(data.frame(x = rnorm(20), y = rnorm(20)))
```

Index