Package ‘wordmap’

January 7, 2025

Type Package

Title Feature Extraction and Document Classification with Noisy Labels

Version 0.9.2

Maintainer Kohei Watanabe <watanabe.kohei@gmail.com>

Description Extract features and classify documents with noisy labels given by document-meta data or keyword matching Watanabe & Zhou (2020) <doi:10.1177/0894439320907027>.

License MIT + file LICENSE

URL https://github.com/koheiw/wordmap

BugReports https://github.com/koheiw/wordmap/issues

LazyData TRUE

Encoding UTF-8

Depends R (>= 3.5), methods

Imports utils, Matrix, quanteda (>= 2.1), stringi, ggplot2, ggrepel

Suggests spelling, testthat

Language en-US

RoxygenNote 7.3.2

NeedsCompilation no

Author Kohei Watanabe [aut, cre, cph]

Repository CRAN

Date/Publication 2025-01-07 22:10:02 UTC

Contents

`accuracy` *Evaluate classification accuracy in precision and recall*

Description

`accuracy()` counts the number of true positive, false positive, true negative, and false negative cases for each predicted class and calculates precision, recall and F1 score based on these counts. `summary()` calculates micro-average precision and recall, and macro-average precision and recall based on the output of `accuracy()`.

Usage

```
accuracy(x, y)

## S3 method for class 'textmodel_wordmap_accuracy'
summary(object, ...)
```

Arguments

`x`	vector of predicted classes.
`y`	vector of true classes.
`object`	output of `accuracy()`.
`...`	not used.

Value

`accuracy()` returns a data.frame with following columns:

`tp`	the number of true positive cases.
`fp`	the number of false positive cases.
`tn`	the number of true negative cases.
`fn`	the number of false negative cases.
`precision`	$tp/(tp + fp)$.
`recall`	$tp/(tp + fn)$.
`f1`	the harmonic mean of precision and recall.

`summary()` returns a named numeric vector with the following elements:

`p`	micro-average precision.
`r`	micro-average recall
`P`	macro-average precision.
`R`	macro-average recall.

Examples

```
class_pred <- c('US', 'GB', 'US', 'CN', 'JP', 'FR', 'CN') # prediction
class_true <- c('US', 'FR', 'US', 'CN', 'KP', 'EG', 'US') # true class
acc <- accuracy(class_pred, class_true)
print(acc)
summary(acc)
```

`afe` *Compute Average Feature Entropy (AFE)*

Description

`afe()` computes Average Feature Entropy (AFE), which measures randomness of occurrences of features in labelled documents (Watanabe & Zhou, 2020). In creating seed dictionaries, AFE can be used to avoid adding seed words that would decrease classification accuracy.

Usage

```
afe(x, y, smooth = 1)
```

Arguments

`x`	a dfm for features.
`y`	a dfm for labels.
`smooth`	a numeric value for smoothing to include all the features.

Value

Returns a single numeric value.

References

Watanabe, Kohei & Zhou, Yuan (2020). "Theory-Driven Analysis of Large Corpora: Semisupervised Topic Classification of the UN Speeches". doi:10.1177/0894439320907027. *Social Science Computer Review*.

as.dictionary.textmodel_wordmap

Create lexicon from a Wordmap model

Description

`as.list()` returns features with the largest coefficients as a list of character vector. `as.dictionary()` returns a quanteda::dictionary object that can be use for dictionary analysis.

Usage

```
## S3 method for class 'textmodel_wordmap'
as.dictionary(x, separator = NULL, ...)

## S3 method for class 'textmodel_wordmap'
as.list(x, ...)
```

Arguments

`x`	a model fitted by `textmodel_wordmap()`.
`separator`	the character in between multi-word dictionary values. If `NULL`, `x$concatenator` will be used.
`...`	passed to coef.textmodel_wordmap

Value

Returns a list or a quanteda::dictionary object.

coef.textmodel_wordmap

Extract coefficients from a Wordmap model

Description

`coef()` extracts top `n` features with largest coefficients for each class.

Usage

```
## S3 method for class 'textmodel_wordmap'
coef(object, n = 10, select = NULL, ...)

## S3 method for class 'textmodel_wordmap'
coefficients(object, n = 10, select = NULL, ...)
```

Arguments

`object`	a model fitted by `textmodel_wordmap()`.
`n`	the number of coefficients to extract.
`select`	returns the coefficients for the selected class; specify by the names of rows in `object$model`.
`...`	not used.

Value

Returns a list of named numeric vectors sorted in descending order.

`data_corpus_ungd2017` *UN General Debate speeches from 2017*

Description

A corpus of 196 speeches from the 2017 UN General Debate (Mikhaylov and Baturo, 2017). The economic data for 2017 (GDP and GDP per capita) are downloaded from the World Bank website.

Usage

```
data_corpus_ungd2017
```

Format

The corpus includes the following document variables:

country_iso ISO3c country code, e.g. "AFG" for Afghanistan

un_session UN session, a numeric identifier (in this case, 72)

year 4-digit year (2017).

country country name, in English.

continent continent of the country, one of: Africa, Americas, Asia, Europe, Oceania. Note that the speech delivered on behalf of the European Union is coded as "Europe".

gdp GDP in $US for 2017, from the World Bank. Contains missing values for 9 countries.

gdp_per_capita GDP per capita in $US for 2017, derived from the World Bank. Contains missing values for 9 countries.

Source

Mikhaylov, M., Baturo, A., & Dasandi, N. (2017). "United Nations General Debate Corpus". doi:10.7910/DVN/0TJX8Y. Harvard Dataverse, V4.

References

Baturo, A., Dasandi, N., & Mikhaylov, S. (2017). "Understanding State Preferences With Text As Data: Introducing the UN General Debate Corpus". doi:10.1177/2053168017712821. *Research and Politics*.

`data_dictionary_topic` *Seed topic dictionary*

Description

A dictionary with seed words for size common topics at the United Nations General Assembly (Watanabe and Zhou, 2020).

Usage

```
data_dictionary_topic
```

Format

An object of class `dictionary2` of length 6.

Author(s)

Kohei Watanabe <`watanabe.kohei@gmail.com`>

References

Watanabe, Kohei & Zhou, Yuan (2020). "Theory-Driven Analysis of Large Corpora: Semisupervised Topic Classification of the UN Speeches". doi:10.1177/0894439320907027. *Social Science Computer Review*.

`predict.textmodel_wordmap` *Predict the most likely class of documents*

Description

Predict document class using fitted Wordmap models.

Usage

```
## S3 method for class 'textmodel_wordmap'
predict(
  object,
  newdata = NULL,
  confidence = FALSE,
  rank = 1L,
  type = c("top", "all"),
  rescale = FALSE,
  min_conf = -Inf,
  min_n = 0L,
  ...
)
```

Arguments

`object`	a model fitted by `textmodel_wordmap()`.
`newdata`	a dfm on which prediction will be made.
`confidence`	if `TRUE`, it returns likelihood ratio scores.
`rank`	rank of the class to be predicted. Only used when `type = "top"`.
`type`	if `top`, returns the most likely class specified by `rank`; otherwise return a matrix of likelihood ratio scores for all possible classes.
`rescale`	if `TRUE`, likelihood ratio scores are normalized using `scale()`. This affects both types of results.
`min_conf`	returns `NA` when confidence is lower than this value.
`min_n`	set the minimum number of polarity words in documents.
`...`	not used.

Value

Returns predicted classes as a vector. If `confidence = TRUE`, it returns a list of two vectors:

`class`	predicted classes of documents.
`confidence.fit`	the confidence of predictions.

`textmodel_wordmap` *A model for multinomial feature extraction and document classification*

Description

Wordmap is a model for multinomial feature extraction and document classification. Its naive Bayesian algorithm allows users to train the model on a large corpus with noisy labels given by document meta-data or keyword matching.

Usage

```
textmodel_wordmap(
  x,
  y,
  label = c("all", "max"),
  smooth = 0.01,
  boolean = FALSE,
  drop_label = TRUE,
  entropy = c("none", "global", "local", "average"),
  residual = FALSE,
  verbose = quanteda_options("verbose"),
  ...
)
```

Arguments

`x`	a dfm or fcm created by `quanteda::dfm()`
`y`	a dfm or a sparse matrix that record class membership of the documents. It can be created applying `quanteda::dfm_lookup()` to `x`.
`label`	if "max", uses only labels for the maximum value in each row of `y`.
`smooth`	the amount of smoothing in computing coefficients. When `smooth = 0.01`, 1% of the mean frequency of words in each class is added to smooth likelihood ratios.
`boolean`	if `TRUE`, only consider presence or absence of features in each document to limit the impact of words repeated in few documents.
`drop_label`	if `TRUE`, drops empty columns of `y` and ignore their labels.
`entropy`	the scheme to compute the entropy to regularize likelihood ratios. The entropy of features are computed over labels if `global` or over documents with the same labels if `local`. Local entropy is averaged if `average`. See the details.
`residual`	if `TRUE`, a residual class is added to `y`. It is named "other" but can be changed via `base::options(wordmap_residual_name)`.
`verbose`	if `TRUE`, shows progress of training.
`...`	additional arguments passed to internal functions.

Details

Wordmap learns association between words in `x` and classes in `y` based on likelihood ratios. The large likelihood ratios tend to concentrate to a small number of features but the entropy of their frequencies over labels or documents helps to disperse the distribution.

A residual class is created internally by adding a new column to `y`. The column is given 1 if the other values in the same row are all zero (i.e. `rowSums(y) == 0`); otherwise 0. It is useful when users cannot create an exhaustive dictionary that covers all the categories.

Value

Returns a fitted textmodel_wordmap object with the following elements:

`model`	a matrix that records the association between classes and features.
`data`	the original input of `x`.
`feature`	the feature set in `x`
`class`	the class labels in `y`.
`concatenator`	the concatenator in `x`.
`entropy`	the scheme to compute entropy weights.
`boolean`	the use of the Boolean transformation of `x`.
`call`	the command used to execute the function.
`version`	the version of the wordmap package.

References

Watanabe, Kohei (2018). "Newsmap: semi-supervised approach to geographical news classification". doi.org/10.1080/21670811.2017.1293487, *Digital Journalism*.

Watanabe, Kohei & Zhou, Yuan (2020). "Theory-Driven Analysis of Large Corpora: Semisupervised Topic Classification of the UN Speeches". doi:10.1177/0894439320907027. *Social Science Computer Review*.

Examples

```
require(quanteda)

# split into sentences
corp <- corpus_reshape(data_corpus_ungd2017)

# tokenize
toks <- tokens(corp, remove_punct = TRUE) %>%
   tokens_remove(stopwords("en"))

# apply seed dictionary
toks_dict <- tokens_lookup(toks, data_dictionary_topic)

# form dfm
dfmt_feat <- dfm(toks)
dfmt_dict <- dfm(toks_dict)

# fit wordmap model
map <- textmodel_wordmap(dfmt_feat, dfmt_dict)
coef(map)
predict(map)
```

`textplot_terms` *Plot coefficients of words*

Description

Plot coefficients of words

Usage

```
textplot_terms(
  x,
  highlighted = NULL,
  max_highlighted = 50,
  max_words = 1000,
  ...
)
```

Arguments

`x`	a fitted textmodel_wordmap object.
`highlighted`	quanteda::pattern to select words to highlight. If a quanteda::dictionary is passed, words in the top-level categories are highlighted in different colors.
`max_highlighted`	the maximum number of words to highlight. When `highlighted = NULL`, words to highlight are randomly selected proportionally to `coef ^ 2`.
`max_words`	the maximum number of words to plot. Words are randomly sampled to keep the number below the limit.
`...`	passed to underlying functions. See the Details.

Details

Users can customize the plots through `...`, which is passed to `ggplot2::geom_text()` and `ggrepel::geom_text_repel()`. The colors are specified internally but users can override the settings by appending `ggplot2::scale_colour_manual()` or `ggplot2::scale_colour_brewer()`. The legend title can also be modified using `ggplot2::labs()`.

Index