Package ‘text’

December 14, 2020

Type Package

Title Analyses of Text using Natural Language Processing and Machine Learning

Version 0.9.10

Description Transforms text variables to word embeddings; where the word embeddings are used to statistically test the mean difference between set of texts, compute semantic similarity scores between texts, predict numerical variables, and visually statistically significant words according to various dimensions etc. For more information see <https://www.r-text.org>.

License GPL-3

URL https://r-text.org/, https://github.com/OscarKjell/text/

BugReports https://github.com/OscarKjell/text/issues/

Encoding UTF-8

Archs x64

SystemRequirements Python (>= 3.6.0)

LazyData true

BuildVignettes true

Imports dplyr, tokenizers, tibble, stringr, tidyr, ggplot2, ggrepel, cowplot, rlang, purrr, magrittr, recipes, rsample, reticulate, tune, workflows, yardstick, future, furrr

RoxygenNote 7.1.0

Suggests knitr, markdown, testthat, rio, glmnet, randomForest, covr, xml2, ranger

VignetteBuilder knitr

Depends R (>= 4.00)

Config/reticulate list( packages = list( list(package = "torch", pip = TRUE), list(package = "transformers", version = "3.3.1", pip = TRUE), list(package = "nltk", pip = TRUE), list(package = "numpy", pip = TRUE) ) )

NeedsCompilation no
centrality_data_harmony

Example data for plotting a Semantic Centrality Plot.

Description

The dataset is a shortened version of the data sets of Study 1 from Kjell, et al., 2016.
Usage

centrality_data_harmony

Format

A data frame with 2,146 and 4 variables:

- **words** unique words
- **n** overall word frequency
- **central_cosine** cosine semantic similarity to the aggregated word embedding
- **n_percent** frequency in percent

Source


---

**DP_projections_HILS_SWLS_100**

*Data for plotting a Dot Product Projection Plot.*

---

Description

Tibble is the output from textProjection. The dataset is a shortened version of the data sets of Study 3-5 from Kjell, Kjell, Garcia and Sikström 2018.

Usage

DP_projections_HILS_SWLS_100

Format

A data frame with 583 rows and 12 variables:

- **words** unique words
- **dot.x** dot product projection on the x-axes
- **p_values_dot.x** p-value for the word in relation to the x-axes
- **n_g1.x** frequency of the word in group 1 on the x-axes variable
- **n_g2.x** frequency of the word in group 2 on the x-axes variable
- **dot.y** dot product projection on the y-axes
- **p_values_dot.y** p-value for the word in relation to the y-axes
- **n_g1.y** frequency of the word in group 1 on the y-axes variable
- **n_g2.y** frequency of the word in group 2 on the x-axes variable
- **n** overall word frequency
- **n.percent** frequency in percent
- **N_participant_responses** number of participants (as this is needed in the analyses)
Source
https://psyarxiv.com/er6t7/

embeddings_from_huggingface2
Word embeddings from textEmbedLayersOutput function

Description
The dataset is a shortened version of the data sets of Study 3-5 from Kjell, Kjell, Garcia and Sikström 2018.

Usage
eMBEDDINGS_FROM_HUGGINGFACE2

Format
A list with word embeddings for harmony words for only contexts. BERT-base embeddings based on mean aggregation of layer 1 and 2.

tokens words
layer_number layer of the transformer model
Dim1:Dim768 Word embeddings dimensions

Source
https://psyarxiv.com/er6t7/

Language_based_assessment_data_3_100
Example text and numeric data.

Description
The dataset is a shortened version of the data sets of Study 3-5 from Kjell, Kjell, Garcia and Sikström 2018.

Usage
Language_based_assessment_data_3_100
Format
A data frame with 100 rows and 4 variables:

- **harmonywords**: Word responses from the harmony in life word question
- **hilstotal**: total score of the Harmony In Life Scale
- **swlstotal**: total score of the Satisfaction With Life Scale

Source
https://psyarxiv.com/er6t7/

Text and numeric data for 10 participants.

Description
The dataset is a shortened version of the data sets of Study 3-5 from Kjell et al., (2018; https://psyarxiv.com/er6t7/).

Usage
Language_based_assessment_data_8

Format
A data frame with 40 participants and 8 variables:

- **harmonywords**: descriptive words where respondents describe their harmony in life
- **satisfactionwords**: descriptive words where respondents describe their satisfaction with life
- **harmonyttexts**: text where respondents describe their harmony in life
- **satisfactiontexts**: text where respondents describe their satisfaction with life
- **hilstotal**: total score of the Harmony In Life Scale
- **swlstotal**: total score of the Satisfaction With Life Scale
- **age**: respondents age in years
- **gender**: respondents gender 1=male, 2=female

Source
https://psyarxiv.com/er6t7/
PC_projections_satisfactionwords_40

Example data for plotting a Principle Component Projection Plot.

Description
The dataset is a shortened version of the data sets of Study 1 from Kjell, et al., 2016.

Usage
PC_projections_satisfactionwords_40

Format
A data frame.

- **words**: unique words
- **n**: overall word frequency
- **Dim_PC1**: Principle component value for dimension 1
- **Dim_PC2**: Principle component value for dimension 2

Source

textCentrality

Compute cosine semantic similarity score between single words' word embeddings and the aggregated word embedding of all words.

Description
Compute cosine semantic similarity score between single words' word embeddings and the aggregated word embedding of all words.

Usage
textCentrality(
  words,
  wordembeddings,
  single_wordembeddings = single_wordembeddings_df,
  aggregation = "mean",
  min_freq_words_test = 0
)
textCentralityPlot

Arguments

words Word or text variable to be plotted.
wordembeddings Word embeddings from textEmbed for the words to be plotted (i.e., the aggregated word embeddings for the "words" variable).
single_wordembeddings Word embeddings from textEmbed for individual words (i.e., the decontextualized word embeddings).
aggregation Method to aggregate the word embeddings (default = "mean"; see also "min", "max" or "[CLS]").
min_freq_words_test Option to select words that have at least occurred a specified number of times (default = 0); when creating the semantic similarity scores within cosine similarity.

Value

A dataframe with variables (e.g., including semantic similarity, frequencies) for the individual words that are used for the plotting in the textCentralityPlot function.

See Also

see textCentralityPlot textProjection

Examples

wordembeddings <- wordembeddings4
data <- Language_based_assessment_data_8
df_for_plotting <- textCentrality(
data$harmonywords,
wordembeddings$harmonywords,
wordembeddings$singlewords_we
)
df_for_plotting

### textCentralityPlot

Plot words according to cosine semantic similarity to the aggregated word embedding.

Description

Plot words according to cosine semantic similarity to the aggregated word embedding.
Usage

textCentralityPlot(
  word_data,
  min_freq_words_test = 1,
  plot_n_word_extreme = 10,
  plot_n_word_frequency = 10,
  plot_n_words_middle = 10,
  titles_color = "#61605e",
  x_axes = "central_cosine",
  title_top = "Semantic Centrality Plot",
  x_axes_label = "Semantic Centrality",
  scale_x_axes_lim = NULL,
  scale_y_axes_lim = NULL,
  word_font = NULL,
  centrality_color_codes = c("#EAEAEA", "#85DB8E", "#398CF9", "#9e9d9d"),
  word_size_range = c(3, 8),
  position_jitter_hight = 0,
  position_jitter_width = 0.03,
  point_size = 0.5,
  arrow_transparency = 0.1,
  points_without_words_size = 0.5,
  points_without_words_alpha = 0.5,
  legend_title = "SC",
  legend_x_axes_label = "x",
  legend_x_position = 0.02,
  legend_y_position = 0.02,
  legend_h_size = 0.2,
  legend_w_size = 0.2,
  legend_title_size = 7,
  legend_number_size = 2,
  seed = 1007
)

Arguments

word_data Tibble from textPlotData.

min_freq_words_test Select words to significance test that have occurred at least min_freq_words_test (default = 1).

plot_n_word_extreme Number of words per dimension to plot with extreme Supervised Bicentroid Projection value. (i.e., even if not significant; duplicates are removed).

plot_n_word_frequency Number of words to plot according to their frequency. (i.e., even if not significant).

plot_n_words_middle Number of words to plot that are in the middle in Supervised Bicentroid Projection score (i.e., even if not significant; duplicates are removed).
titles_color  
Color for all the titles (default: 
"#61605e").

x_axes  
Variable to be plotted on the x-axes (default is "central_cosine").

title_top  
Title (default " ").

x_axes_label  
Label on the x-axes.

scale_x_axes_lim  
Length of the x-axes (default: NULL, which uses c(min(word_data$central_cosine)-0.05, max(word_data$central_cosine)+0.05); change this by e.g., try c(-5, 5)).

scale_y_axes_lim  
Length of the y-axes (default: NULL, which uses c(-1, 1); change e.g., by trying c(-5, 5)).

word_font  
Type of font (default: NULL).

centrality_color_codes  
Colors of the words selected as plot_n_word_extreme (minimum values), plot_n_words_middle, plot_n_word_extreme (maximum values) and plot_n_word_frequency; the default is c("#EAEAEA", "#85DB8E", 
"#398CF9", "#000000"), respectively.

word_size_range  
Vector with minimum and maximum font size (default: c(3, 8)).

position_jitter_height  
Jitter height (default: .0).

position_jitter_width  
Jitter width (default: .03).

point_size  
Size of the points indicating the words’ position (default: 0.5).

arrow_transparency  
Transparency of the lines between each word and point (default: 0.1).

points_without_words_size  
Size of the points not linked to a word (default is to not show the point; , i.e., 0).

points_without_words_alpha  
Transparency of the points that are not linked to a word (default is to not show it; i.e., 0).

legend_title  
Title of the color legend (default: "(SCP)").

legend_x_axes_label  
Label on the color legend (default: "(x)").

legend_x_position  
Position on the x coordinates of the color legend (default: 0.02).

legend_y_position  
Position on the y coordinates of the color legend (default: 0.05).

legend_h_size  
Height of the color legend (default 0.15).

legend_w_size  
Width of the color legend (default 0.15).

legend_title_size  
Font size of the title (default = 7).

legend_number_size  
Font size of the values in the legend (default = 2).

seed  
Set different seed.
Value

A 1-dimensional word plot based on cosine similarity to the aggregated word embedding, as well as tibble with processed data used to plot.

See Also

see textCentrality and textProjection

Examples

```r
# The test-data included in the package is called: centrality_data_harmony
names(centrality_data_harmony)
# Plot
centrality_plot <- textCentralityPlot(
  word_data = centrality_data_harmony,
  min_freq_words_test = 10,
  plot_n_word_extreme = 10,
  plot_n_word_frequency = 10,
  plot_n_words_middle = 10,
  titles_color = "#61605e",
  x_axes = "central_cosine",
  #
  # title_top = "Semantic Centrality Plot",
  # x_axes_label = "Semantic Centrality",
  #
  # word_font = NULL,
  # centrality_color_codes = c("#EAEAEA","#85DB8E","#398CF9","#000000"),
  # word_size_range = c(3, 8),
  # point_size = 0.5,
  # arrow_transparency = 0.1,
  # points_without_words_size = 0.5,
  # points_without_words_alpha = 0.5,
  # )
  # centrality_plot
```

---

textEmbed

Extract layers and aggregate them to word embeddings, for all character variables in a given dataframe.

Description

Extract layers and aggregate them to word embeddings, for all character variables in a given dataframe.

Usage

```r
textEmbed(
  x,
  model = "bert-base-uncased",
  layers = 11:12,
)```
contexts = TRUE,
context_layers = layers,
context_aggregation_layers = "concatenate",
context_aggregation_tokens = "mean",
context_tokens_select = NULL,
context_tokens_deselect = NULL,
decontexts = TRUE,
decontext_layers = layers,
decontext_aggregation_layers = "concatenate",
decontext_aggregation_tokens = "mean",
decontext_tokens_select = NULL,
decontext_tokens_deselect = NULL
)

Arguments

x
A character variable or a tibble/dataframe with at least one character variable.

model

layers
Specify the layers that should be extracted (default 11:12). It is more efficient to only extract the layers that you need (e.g., 12). Layer 0 is the decontextualized input layer (i.e., not comprising hidden states) and thus advised to not use. These layers can then be aggregated in the textEmbedLayerAggregation function. If you want all layers then use 'all'.

contexts
Provide word embeddings based on word contexts (standard method; default = TRUE).

context_layers
Specify the layers that should be aggregated (default the number of layers extracted above). Layer 0 is the decontextualized input layer (i.e., not comprising hidden states) and thus advised not to be used.

context_aggregation_layers
Method to aggregate the contextualized layers (e.g., "mean", "min" or "max, which takes the minimum, maximum or mean, respectively, across each column; or "concatenate", which links together each word embedding layer to one long row.

context_aggregation_tokens
Method to aggregate the contextualized tokens (e.g., "mean", "min" or "max, which takes the minimum, maximum or mean, respectively, across each column; or "concatenate", which links together each word embedding layer to one long row.

context_tokens_select
Option to select word embeddings linked to specific tokens such as [CLS] and [SEP] for the context embeddings.

context_tokens_deselect
Option to deselect embeddings linked to specific tokens such as [CLS] and [SEP] for the context embeddings.
decontexts  Provide word embeddings of single words as input (embeddings, e.g., used for plotting; default = TRUE).

decontext_layers  Layers to aggregate for the decontext embeddings the number of layers extracted above.

decontext_aggregation_layers  Method to aggregate the decontextualized layers (e.g., "mean", "min" or "max, which takes the minimum, maximum or mean, respectively, across each column; or "concatenate", which links together each word embedding layer to one long row.

decontext_aggregation_tokens  Method to aggregate the decontextualized tokens (e.g., "mean", "min" or "max, which takes the minimum, maximum or mean, respectively, across each column; or "concatenate", which links together each word embedding layer to one long row.

decontext_tokens_select  Option to select embeddings linked to specific tokens such as [CLS] and [SEP] for the decontext embeddings.

decontext_tokens_deselect  Option to deselect embeddings linked to specific tokens such as [CLS] and [SEP] for the decontext embeddings.

Value

A tibble with tokens, a column for layer identifier and word embeddings. Note that layer 0 is the input embedding to the transformer

See Also

see textEmbedLayerAggregation and textEmbedLayersOutput

Examples

x <- Language_based_assessment_data_8[1:2, 1:2]
# Example 1
wordembeddings <- textEmbed(x, layers = 9:11, context_layers = 11, decontext_layers = 9)
# Show information that have been saved with the embeddings about how they were constructed
comment(wordembeddings$satisfactionwords)
comment(wordembeddings$singlewords_we)
comment(wordembeddings)
# Example 2
wordembeddings <- textEmbed(x, layers = "all", context_layers = "all", decontext_layers = "all")
textEmbedLayerAggregation

Select and aggregate layers of hidden states to form a word embeddings.

Description

Select and aggregate layers of hidden states to form a word embeddings.

Usage

textEmbedLayerAggregation(
  word_embeddings_layers,
  layers = 11:12,
  aggregate_layers = "concatenate",
  aggregate_tokens = "mean",
  tokens_select = NULL,
  tokens_deselect = NULL
)

Arguments

word_embeddings_layers
  Layers outputted from textEmbedLayersOutput.

layers
  The numbers of the layers to be aggregated (e.g., c(11:12) to aggregate the eleventh and twelfth). Note that layer 0 is the input embedding to the transformer, and should normally not be used. Selecting 'all' thus removes layer 0.

aggregate_layers
  Method to carry out the aggregation among the layers for each word/token, including "min", "max" and "mean" which takes the minimum, maximum or mean across each column; or "concatenate", which links together each layer of the word embedding to one long row. Default is "concatenate"

aggregate_tokens
  Method to carry out the aggregation among the word embeddings for the words/tokens, including "min", "max" and "mean" which takes the minimum, maximum or mean across each column; or "concatenate", which links together each layer of the word embedding to one long row.

tokens_select
  Option to only select embeddings linked to specific tokens such as "[CLS]" and "[SEP]" (default NULL).

tokens_deselect
  Option to deselect embeddings linked to specific tokens such as "[CLS]" and "[SEP]" (default NULL).
textEmbedLayersOutput

Value

A tibble with word embeddings. Note that layer 0 is the input embedding to the transformer, which is normally not used.

See Also

see textEmbedLayersOutput and textEmbed

Examples

```r
embeddings_layers <- textEmbedLayersOutput(Language_based_assessment_data_8$harmonywords[1], layers = 11)
wordembeddings <- textEmbedLayerAggregation(embeddings_layers$context, layers = 11)
```

textEmbedLayersOutput  Extract layers of hidden states (word embeddings) for all character variables in a given dataframe.

Description

Extract layers of hidden states (word embeddings) for all character variables in a given dataframe.

Usage

```r
textEmbedLayersOutput(
  x, 
  contexts = TRUE, 
  decontexts = TRUE, 
  model = "bert-base-uncased", 
  layers = 11, 
  return_tokens = TRUE
)
```

Arguments

- **x**: A character variable or a tibble/dataframe with at least one character variable.
- **contexts**: Provide word embeddings based on word contexts (standard method; default = TRUE).
- **decontexts**: Provide word embeddings of single words as input (embeddings used for plotting; default = TRUE).
layers Specify the layers that should be extracted (default 11). It is more efficient to only extract the layers that you need (e.g., 11). You can also extract several (e.g., 11:12), or all by setting this parameter to "all". Layer 0 is the decontextualized input layer (i.e., not comprising hidden states) and thus should normally not be used. These layers can then be aggregated in the textEmbedLayerAggregation function.

return_tokens If TRUE, provide the tokens used in the specified transformer model.

Value
A tibble with tokens, column specifying layer and word embeddings. Note that layer 0 is the input embedding to the transformer, and should normally not be used.

See Also
see textEmbedLayerAggregation and textEmbed

Examples

```r
x <- Language_based_assessment_data_8[1:2, 1:2]
word_embeddings_with_layers <- textEmbedLayersOutput(x, layers = 11:12)
```

textEmbedStatic Applies word embeddings from a given decontextualized static space (such as from Latent Semantic Analyses) to all character variables

Description
Applies word embeddings from a given decontextualized static space (such as from Latent Semantic Analyses) to all character variables

Usage
textEmbedStatic(df, space, tk_df = "null", aggregate = "mean")

Arguments
df dataframe that at least contains one character column.
space decontextualized/static space (from textSpace, which is not included in the current text package).
tk_df default "null"; option to use either the "tk" of "df" space (if using textSpace, which has not been implemented yet).
aggregate method to aggregate semantic representation when there are more than a single word. (default is "mean"; see also "min" and "max")
Value

A list with tibbles for each character variable. Each tibble comprises a column with the text, followed by columns representing the semantic representations of the text. The tibbles are called the same as the original variable.

See Also

see `textEmbed`

---

**textPCA**

*Compute 2 PCA dimensions of the word embeddings for individual words.*

Description

Compute 2 PCA dimensions of the word embeddings for individual words.

Usage

textPCA(words, single_wordembeddings = single_wordembeddings_df, seed = 1010)

Arguments

- **words** Word or text variable to be plotted.
- **single_wordembeddings** Word embeddings from textEmbed for individual words (i.e., decontextualized embeddings).
- **seed** Set different seed.

Value

A dataframe with words, their frequency and two PCA dimensions from the wordembeddings for the individual words that is used for the plotting in the textPCAPlot function.

See Also

see `textPCAPlot`

Examples

```r
# Data
df_for_plotting2d <- textPCA(
  words = Language_based_assessment_data_8$harmonywords,
  single_wordembeddings = wordembeddings4$singlewords_we
)
df_for_plotting2d
```
textPCAPlot

Plot words according to 2-D plot from 2 PCA components.

Description

Plot words according to 2-D plot from 2 PCA components.

Usage

textPCAPlot(
  word_data,
  min_freq_words_test = 1,
  plot_n_word_extreme = 5,
  plot_n_word_frequency = 5,
  plot_n_words_middle = 5,
  titles_color = "#61605e",
  title_top = "Principal Component (PC) Plot",
  x_axes_label = "PC1",
  y_axes_label = "PC2",
  scale_x_axes_lim = NULL,
  scale_y_axes_lim = NULL,
  word_font = NULL,
  bivariate_color_codes = c("#398CF9", "#60A1F7", "#5dc688", "#e07f6a", "#EAEAEA",
    "#40DD52", "#FF0000", "#EA7467", "#85DB8E"),
  word_size_range = c(3, 8),
  position_jitter_height = 0,
  position_jitter_width = 0.03,
  point_size = 0.5,
  arrow_transparency = 0.1,
  points_without_words_size = 0.2,
  points_without_words_alpha = 0.2,
  legend_title = "PC",
  legend_x_axes_label = "PC1",
  legend_y_axes_label = "PC2",
  legend_x_position = 0.02,
  legend_y_position = 0.02,
  legend_h_size = 0.2,
  legend_w_size = 0.2,
  legend_title_size = 7,
  legend_number_size = 2,
  seed = 1002
)

Arguments

word_data Dataframe from textPCA
min_freq_words_test
Select words to significance test that have occurred at least min_freq_words_test (default = 1).

plot_n_word_extreme
Number of words that are extreme on Supervised Bicentroid Projection per dimension. (i.e., even if not significant; per dimensions, where duplicates are removed).

plot_n_word_frequency
Number of words based on being most frequent. (i.e., even if not significant).

plot_n_words_middle
Number of words plotted that are in the middle in Supervised Bicentroid Projection score (i.e., even if not significant; per dimensions, where duplicates are removed).

titles_color
Color for all the titles (default: ":61605e")

title_top
Title (default ":")

x_axes_label
Label on the x-axes.

y_axes_label
Label on the y-axes.

scale_x_axes_lim
Manually set the length of the x-axes (default = NULL, which uses ggplot2::scale_x_continuous(limits = scale_x_axes_lim); change e.g., by trying c(-5, 5)).

scale_y_axes_lim
Manually set the length of the y-axes (default = NULL; which uses ggplot2::scale_y_continuous(limits = scale_y_axes_lim); change e.g., by trying c(-5, 5)).

word_font
Font type (default: NULL).

bivariate_color_codes
The different colors of the words (default: c("#398CF9", ":60A1F7", ":5dc688", ":e07f6a", ":EAEAEA", ":40DD52", ":FF0000", ":EA7467", ":85DB8E")).

word_size_range
Vector with minimum and maximum font size (default: c(3, 8)).

position_jitter_height
Jitter height (default: .0).

position_jitter_width
Jitter width (default: .03).

point_size
Size of the points indicating the words’ position (default: 0.5).

arrow_transparency
Transparency of the lines between each word and point (default: 0.1).

points_without_words_size
Size of the points not linked with a words (default is to not show it, i.e., 0).

points_without_words_alpha
Transparency of the points not linked with a words (default is to not show it, i.e., 0).

legend_title
Title on the color legend (default: "(PCA)").

legend_x_axes_label
Label on the color legend (default: "(x)").
textPredict

Predict scores or classification from, e.g., textTrain.

Description

Predict scores or classification from, e.g., textTrain.

Usage

textPredict(model_info, new_data, type = NULL, ...)

Examples

# The test-data included in the package is called: DP_projections_HILS_SWLS_100

# Supervised Bicentroid Projection Plot
principle_component_plot_projection <- textPCAPlot(PC_projections_satisfactionwords_40)
principle_component_plot_projection

names(DP_projections_HILS_SWLS_100)

Legend

- **legend_y_axes_label**: Label on the color legend (default: "(y)").
- **legend_x_position**: Position on the x coordinates of the color legend (default: 0.02).
- **legend_y_position**: Position on the y coordinates of the color legend (default: 0.05).
- **legend_h_size**: Height of the color legend (default 0.15).
- **legend_w_size**: Width of the color legend (default 0.15).
- **legend_title_size**: Font size (default: 7).
- **legend_number_size**: Font size of the values in the legend (default: 2).

See Also

see textPCA
textProjection

Arguments

- **model_info**: Model info (e.g., saved output from textTrain, textTrainRegression or textRandomForest).
- **new_data**: Word embeddings from new data to be predicted from.
- **type**: Type of prediction; e.g., "prob", "class"
- **...**: From predict

Value

Predicted scores from word embeddings.

See Also

see `textTrain` textTrainLists textTrainRandomForest textSimilarityTest

Examples

```r
wordembeddings <- wordembeddings4
ratings_data <- Language_based_assessment_data_8

textProjection(words, wordembeddings, single_wordembeddings = single_wordembeddings_df, x, y = NULL, pca = NULL, aggregation = "mean", split = "quartile", word_weight_power = 1, min_freq_words_test = 0, Npermutations = 10000, n_per_split = 50000, seed = 1003)
```

Description

Compute Supervised Bicentroid Projection and related variables for plotting words.

Usage

```r
textProjection(  
  words,  
  wordembeddings,  
  single_wordembeddings = single_wordembeddings_df,  
  x,  
  y = NULL,  
  pca = NULL,  
  aggregation = "mean",  
  split = "quartile",  
  word_weight_power = 1,  
  min_freq_words_test = 0,  
  Npermutations = 10000,  
  n_per_split = 50000,  
  seed = 1003)  
)```
Arguments

words: Word or text variable to be plotted.
wordembeddings: Word embeddings from textEmbed for the words to be plotted (i.e., the aggregated word embeddings for the "words" parameter).
single_wordembeddings: Word embeddings from textEmbed for individual words (i.e., decontextualized embeddings).
x: Numeric variable that the words should be plotted according to on the x-axes.
y: Numeric variable that the words should be plotted according to on the y-axes (y=NULL).
pca: Number of PCA dimensions applied to the word embeddings in the beginning of the function. A number below 1 takes out % of variance; An integer specify number of components to extract. (default is NULL as this setting has not yet been evaluated).
aggregation: Method to aggregate the word embeddings (default = "mean"; see also "min", "max", and "[CLS]").
split: Method to split the axes (default = "quartile" involving selecting lower and upper quartile; see also "mean"). However, if the variable is only containing two different values (i.e., being dichotomous) mean split is used.
word_weight_power: Compute the power of the frequency of the words and multiply the word embeddings with this in the computation of aggregated word embeddings for group low (1) and group high (2). This increases the weight of more frequent words.
min_freq_words_test: Option to select words that have occurred a specified number of times (default = 0); when creating the Supervised Bicentroid Projection line (i.e., single words receive Supervised Bicentroid Projection and p-value).
Npermutations: Number of permutations in the creation of the null distribution.
n_per_split: A setting to split Npermutations to avoid reaching computer memory limits; the higher the faster, but too high may lead to abortion.
seed: Set different seed.

Value

A dataframe with variables (e.g., including Supervised Bicentroid Projection, frequencies, p-values) for the individual words that is used for the plotting in the textProjectionPlot function.

Examples

# Data
wordembeddings <- wordembeddings4
raw_data <- Language_based_assessment_data_8
# Pre-processing data for plotting
df_for_plotting <- textProjection(
  words = raw_data$harmonywords,
wordembeddings = wordembeddings$harmonywords,
single_wordembeddings = wordembeddings$singlewords_we,
x = raw_data$hilstotal,
split = "mean",
Npermutations = 10,
n_per_split = 1
)
df_for_plotting
#' @seealso see \code{\link{textProjectionPlot}}

---

**textProjectionPlot**

*Plot words according to Supervised Bicentroid Projection.*

### Description

Plot words according to Supervised Bicentroid Projection.

### Usage

```r
textProjectionPlot(
  word_data,
  k_n_words_to_test = FALSE,
  min_freq_words_test = 1,
  min_freq_words_plot = 1,
  plot_n_words_square = 3,
  plot_n_words_p = 5,
  plot_n_word_extreme = 5,
  plot_n_word_frequency = 5,
  plot_n_words_middle = 5,
  titles_color = "#61605e",
  y_axes = FALSE,
  p_alpha = 0.05,
  p_adjust_method = "none",
  title_top = "Supervised Bicentroid Projection",
  x_axes_label = "Supervised Bicentroid Projection (SBP)",
  y_axes_label = "Supervised Bicentroid Projection (SBP)",
  scale_x_axes_lim = NULL,
  scale_y_axes_lim = NULL,
  word_font = NULL,
  bivariate_color_codes = c("#398CF9", "#60A1F7", "#5dc688", "#e07f6a", "#EAEAEA",
                           "#40DD52", "#FF0000", "#EA7467", "#85DB8E"),
  word_size_range = c(3, 8),
  position_jitter_hight = 0,
  position_jitter_width = 0.03,
  point_size = 0.5,
  arrow_transparency = 0.1,
  points_without_words_size = 0.2,
  points_without_words_alpha = 0.2,
)```
legend_title = "DPP",
legend_x_axes_label = "x",
legend_y_axes_label = "y",
legend_x_position = 0.02,
legend_y_position = 0.02,
legend_h_size = 0.2,
legend_w_size = 0.2,
legend_title_size = 7,
legend_number_size = 2,
seed = 1005
}

Arguments

word_data Dataframe from textProjection

k_n_words_to_test
Select the k most frequent words to significance test (k = sqrt(100*N); N = number of participant responses). Default = TRUE.

min_freq_words_test
Select words to significance test that have occurred at least min_freq_words_test times. (default = 1).

min_freq_words_plot
Select words to plot that has occurred at least min_freq_words_plot times.

plot_n_words_square
Select number of significant words in each square of the figure to plot. The significant words, in each square is selected according to most frequent words.

plot_n_words_p
Number of significant words to plot on each(positive and negative) side of the x-axes and y-axes, (where duplicates are removed); selects first according to lowest p-value and then according to frequency. Hence, on a two dimensional plot it is possible that plot_n_words_p = 1 yield 4 words.

plot_n_word_extreme
Number of words that are extreme on Supervised Bicentroid Projection per dimension. (i.e., even if not significant; per dimensions, where duplicates are removed).

plot_n_word_frequency
Number of words based on being most frequent. (i.e., even if not significant).

plot_n_words_middle
Number of words plotted that are in the middle in Supervised Bicentroid Projection score (i.e., even if not significant; per dimensions, where duplicates are removed).

titles_color Color for all the titles (default: "#61605e")

y_axes If TRUE, also plotting on the y-axes (default is FALSE). Also plotting on y-axes produces a two dimension 2-dimensional plot, but the textProjection function has to have had a variable on the y-axes.

p_alpha Alpha (default = .05).
**p_adjust_method**
Method to adjust/correct p-values for multiple comparisons (default = "holm"; see also "none", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr").

**title_top**
Title (default "")

**x_axes_label**
Label on the x-axes.

**y_axes_label**
Label on the y-axes.

**scale_x_axes_lim**
Manually set the length of the x-axes (default = NULL, which uses ggplot2::scale_x_continuous(limits = scale_x_axes_lim); change e.g., by trying c(-5, 5)).

**scale_y_axes_lim**
Manually set the length of the y-axes (default = NULL; which uses ggplot2::scale_y_continuous(limits = scale_y_axes_lim); change e.g., by trying c(-5, 5)).

**word_font**
Font type (default: NULL).

**bivariate_color_codes**
The different colors of the words. Note that, at the moment, two squares should not have the exact same colour-code because the numbers within the squares of the legend will then be aggregated (and show the same, incorrect value). (default: c("#398CF9", "#60A1F7", "#5dc688", "#e07f6a", "#EAEAEA", "#40DD52", "#FF0000", "#EA7467", "#85DB8E").)

**word_size_range**
Vector with minimum and maximum font size (default: c(3, 8)).

**position_jitter_height**
Jitter height (default: 0).

**position_jitter_width**
Jitter width (default: 0.03).

**point_size**
Size of the points indicating the words’ position (default: 0.5).

**arrow_transparency**
Transparency of the lines between each word and point (default: 0.1).

**points_without_words_size**
Size of the points not linked with a words (default is to not show it, i.e., 0).

**points_without_words_alpha**
Transparency of the points not linked with a words (default is to not show it, i.e., 0).

**legend_title**
Title on the color legend (default: "(SBP)").

**legend_x_axes_label**
Label on the color legend (default: "(x)").

**legend_y_axes_label**
Label on the color legend (default: "(y)").

**legend_x_position**
Position on the x coordinates of the color legend (default: 0.02).

**legend_y_position**
Position on the y coordinates of the color legend (default: 0.05).

**legend_h_size**
Height of the color legend (default 0.15).

**legend_w_size**
Width of the color legend (default 0.15).
textSimilarity

Compute the cosine semantic similarity between two text variables.

Description
Compute the cosine semantic similarity between two text variables.

Usage

textSimilarity(x, y)
textSimilarityNorm

Arguments

x  Word embeddings from textEmbed.
y  Word embeddings from textEmbed.

Value

A vector comprising cosine semantic similarity scores.

See Also

see textSimilarityNorm and textSimilarityTest

Examples

library(dplyr)
wordembeddings <- wordembeddings4
similarity_scores <- textSimilarity(wordembeddings$harmonytext, wordembeddings$satisfactiontext)
comment(similarity_scores)

textSimilarityNorm  Compute the semantic similarity between a text variable and a word norm (i.e., a text represented by one word embedding that represent a construct).

Description

Compute the semantic similarity between a text variable and a word norm (i.e., a text represented by one word embedding that represent a construct).

Usage

textSimilarityNorm(x, y)

Arguments

x  Word embeddings from textEmbed (with several rows of text).
y  Word embedding from textEmbed (from only one text).

Value

A vector comprising cosine semantic similarity scores.

See Also

see textSimilarity and textSimilarityTest
Examples

```r
## Not run:
library(dplyr)
library(tibble)
harmonynorm <- c("harmony peace ")
satisfactionnorm <- c("satisfaction achievement")

norms <- tibble::tibble(harmonynorm, satisfactionnorm)
wordembeddings <- wordembeddings4
wordembeddings_wordnorm <- textEmbed(norms)
similarity_scores <- textSimilarityNorm(
  wordembeddings$harmonytext,
  wordembeddings_wordnorm$harmonynorm
)

## End(Not run)
```

textSimilarityTest

Test whether there is a significant difference in meaning between two sets of texts (i.e., between their word embeddings).

Description

Test whether there is a significant difference in meaning between two sets of texts (i.e., between their word embeddings).

Usage

```r
textSimilarityTest( 
  x, 
  y, 
  Npermutations = 10000, 
  method = "paired", 
  alternative = c("two_sided", "less", "greater"), 
  output.permutations = TRUE, 
  N_cluster_nodes = 1, 
  seed = 1001 
)
```

Arguments

- `x`: Set of word embeddings from textEmbed.
- `y`: Set of word embeddings from textEmbed.
- `Npermutations`: Number of permutations (default 1000).
- `method`: Compute a "paired" or an "unpaired" test.
- `alternative`: Use a two or one-sided test (select one of: "two_sided", "less", "greater").
output.permutations  
If TRUE, returns permuted values in output.

N_cluster_nodes  
Number of cluster nodes to use (more makes computation faster; see parallel package).

seed  
Set different seed.

Value  
A list with a p-value, cosine_estimate and permuted values if output.permutations=TRUE.

Examples  
x <- wordembeddings4$harmonywords
y <- wordembeddings4$satisfactionwords
textSimilarityTest(x, y, method = "paired", Npermutations = 10, N_cluster_nodes = 1, alternative = "two_sided")

---

**textTrain**  
*Train word embeddings to a numeric (ridge regression) or categorical (random forest) variable.*

**Description**  
Train word embeddings to a numeric (ridge regression) or categorical (random forest) variable.

**Usage**  
textTrain(x, y, force_train_method = "automatic", ...)

**Arguments**  
x  
Word embeddings from textEmbed (or textEmbedLayerAggregation). Can analyze several variables at the same time; but if training to several outcomes at the same time use a tibble within the list as input rather than just a tibble input (i.e., keep the name of the wordembedding).

y  
Numeric variable to predict. Can be several; although then make sure to have them within a tibble (this is required even if it is only one outcome but several word embeddings variables).

force_train_method  
default is "automatic", so if y is a factor random_forest is used, and if y is numeric ridge regression is used. This can be overridden using "regression" or "random_forest".
Arguments from textTrainRegression or textTrainRandomForest the textTrain function.

Value

A correlation between predicted and observed values; as well as a tibble of predicted values.

See Also

textTrainRegression textTrainRandomForest textTrainLists textSimilarityTest

Examples

```r
## Not run:
wordembeddings <- wordembeddings4
ratings_data <- Language_based_assessment_data_8
results <- textTrain(
  wordembeddings$harmonytext,
  ratings_data$hilstotal
)
## End(Not run)
```

textTrainLists

Individually trains word embeddings from several text variables to several numeric or categorical variables. It is possible to have word embeddings from one text variable and several numeric/categorical variables; or vice versa, word embeddings from several text variables to one numeric/categorical variable. It is not possible to mix numeric and categorical variables.

Description

Individually trains word embeddings from several text variables to several numeric or categorical variables. It is possible to have word embeddings from one text variable and several numeric/categorical variables; or vice versa, word embeddings from several text variables to one numeric/categorical variable. It is not possible to mix numeric and categorical variables.

Usage

textTrainLists(
  x, y,
  force_train_method = "automatic",
  save_output = "all",
  method_cor = "pearson",
  model = "regression",
  eval_measure = "rmse",
)
Arguments

x  
Word embeddings from textEmbed (or textEmbedLayerAggregation).

y  
Tibble with several numeric or categorical variables to predict. Please note that you cannot mix numeric and categorical variables.

force_train_method  
default is automatic; see also "regression" and "random_forest".

save_output  
Option not to save all output; default "all". see also "only_results" and "only_results_predictions".

method_cor  
"pearson",

model  
type of model to use in regression; default is "regression"; see also "logistic".  
(To set different random forest algorithms see extremely_randomised_splitrule parameter in textTrainRandomForest)

eval_measure  
Type of evaluative measure to assess models on.

p_adjust_method  
Method to adjust/correct p-values for multiple comparisons (default = "holm"; see also "none", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr").

Value

Correlations between predicted and observed values.

See Also

see textTrain textTrainRegression textTrainRandomForest

Examples

```r
## Not run:
wordembeddings <- wordembeddings4[1:2]
results <- textTrainLists(
  wordembeddings,
  ratings_data
)
results
comment(results)
```

## End(Not run)
textTrainRandomForest  Train word embeddings to a categorical variable using random forest.

Description
Train word embeddings to a categorical variable using random forest.

Usage
textTrainRandomForest(
  x,
  y,
  cv_method = "validation_split",
  outside_folds = 10,
  outside_strata_y = "y",
  outside_breaks = 4,
  inside_folds = 3/4,
  inside_strata_y = "y",
  inside_breaks = 4,
  mode_rf = "classification",
  preprocess_step_center = FALSE,
  preprocess_scale_center = FALSE,
  preprocess_PCA = NA,
  extremely_randomised_splitrule = "extratrees",
  mtry = c(1, 10, 20, 40),
  min_n = c(1, 10, 20, 40),
  trees = c(1000),
  eval_measure = "bal_accuracy",
  model_description = "Consider writing a description of your model here",
  multi_cores = "multi_cores_sys_default",
  save_output = "all",
  seed = 2020,
  ...
)

Arguments
x  Word embeddings from textEmbed.
y  Categorical variable to predict.
cv_method  Cross-validation method to use within a pipeline of nested outer and inner loops of folds (see nested_cv in rsample). Default is using cv_folds in the outside folds and "validation_split" using rsample::validation_split in the inner loop to achieve a development and assessment set (note that for validation_split the inside_folds should be a proportion, e.g., inside_folds = 3/4); whereas "cv_folds" uses rsample::vfold_cv to achieve n-folds in both the outer and inner loops.
outside_folds  Number of folds for the outer folds (default = 10).
outside_strata_y Variable to stratify according (default "y"; can also set to NULL).
outside_breaks The number of bins wanted to stratify a numeric stratification variable in the outer cross-validation loop.
inside_folds Number of folds for the inner folds (default = 3/4).
inside_strata_y Variable to stratify according (default "y"; can also set to NULL).
inside_breaks The number of bins wanted to stratify a numeric stratification variable in the inner cross-validation loop.
mode_rf Default is "classification" ("regression" is not supported yet).
preprocess_step_center normalizes dimensions to have a mean of zero; default is set to TRUE. For more info see (step_center in recipes).
preprocess_scale_center normalizes dimensions to have a standard deviation of one. For more info see (step_scale in recipes).
preprocess_PCA Pre-processing threshold for PCA. Can select amount of variance to retain (e.g., .90 or as a grid c(0.80, 0.90)); or number of components to select (e.g., 10). Default is "min_halving", which is a function that selects the number of PCA components based on number of participants and feature (word embedding dimensions) in the data. The formula is: preprocess_PCA = round(max(min(number_features/2), number_participants/2), min(50, number_features))).
extremely_randomised_splitrule default: "extratrees", which thus implement a random forest; can also select: NULL, "gini" or "hellinger"; if these are selected your mtry settings will be overridden (see Geurts et al. (2006) Extremely randomized trees for details; and see the ranger r-package for details on implementations).
mtry hyper parameter that may be tuned; default: c(1, 20, 40),
min_n hyper parameter that may be tuned; default: c(1, 20, 40)
trees Number of trees to use (default 1000).
eval_measure Measure to evaluate the models in order to select the best hyperparameters default "roc_auc"; see also "accuracy", "bal_accuracy", "sens", "spec", "precision", "kappa", "f_measure".
model_description Text to describe your model (optional; good when sharing the model with others).
multi_cores If TRUE it enables the use of multiple cores if the computer system allows for it (i.e., only on unix, not windows). Hence it makes the analyses considerably faster to run. Default is "multi_cores_sys_default", where it automatically uses TRUE for Mac and Linux and FALSE for Windows.
save_output Option not to save all output; default "all". see also "only_results" and "only_results_predictions".
seed Set different seed.
... For example settings in yardstick::accuracy to set event_level (e.g., event_level = "second").
textTrainRegression

Value

A list with roc_curve_data, roc_curve_plot, truth and predictions, preprocessing_recipe, final_model, model_description chisq and fishers test as well as evaluation measures, e.g., including accuracy, f_meas and roc_auc (for details on these measures see the yardstick r-package documentation).

See Also

see textTrainLists textSimilarityTest

Examples

results <- textTrainRandomForest(
        wordembeddings$harmonywords,
        as.factor(Language_based_assessment_data_8$gender),
        trees = c(1000, 1500),
        mtry = c(1), # this is short because of testing
        min_n = c(1), # this is short because of testing
        multi_cores = FALSE # This is FALSE due to CRAN testing and Windows machines.
    )

textTrainRegression  Train word embeddings to a numeric variable.

Description

Train word embeddings to a numeric variable.

Usage

textTrainRegression(
    x,
    y,
    cv_method = "validation_split",
    outside_folds = 10,
    outside_strata_y = "y",
    outside_breaks = 4,
    inside_folds = 3/4,
    inside_strata_y = "y",
    inside_breaks = 4,
    model = "regression",
    eval_measure = "default",
    preprocess_step_center = TRUE,
    preprocess_step_scale = TRUE,
    preprocess_PCA = NA,
    penalty = 10^seq(-16, 16),
    mixture = c(0),
)
first_n_predictors = NA,
impute_missing = FALSE,
method_cor = "pearson",
model_description = "Consider writing a description of your model here",
multi_cores = "multi_cores_sys_default",
save_output = "all",
seed = 2020,
...)

Arguments

x         Word embeddings from textEmbed (or textEmbedLayerAggregation).
y         Numeric variable to predict.
cv_method   Cross-validation method to use within a pipeline of nested outer and inner loops of folds (see nested_cv in rsample). Default is using cv_folds in the outside folds and "validation_split" using rsample::validation_split in the inner loop to achieve a development and assessment set (note that for validation_split the inside_folds should be a proportion, e.g., inside_folds = 3/4); whereas "cv_folds" uses rsample::vfold_cv to achieve n-folds in both the outer and inner loops.
outside_folds   Number of folds for the outer folds (default = 10).
outside_strata_y Variable to stratify according (default y; can set to NULL).
outside_breaks   The number of bins wanted to stratify a numeric stratification variable in the outer cross-validation loop.
inside_folds    The proportion of data to be used for modeling/analysis; (default proportion = 3/4). For more information see validation_split in rsample.
inside_strata_y Variable to stratify according (default y; can set to NULL).
inside_breaks   The number of bins wanted to stratify a numeric stratification variable in the inner cross-validation loop.
model        Type of model. Default is "regression"; see also "logistic" for classification.
eval_measure Type of evaluative measure to select models from. Default = "rmse" for regression and "bal_accuracy" for logistic. For regression use "rsq" or "rmse"; and for classification use "accuracy", "bal_accuracy", "sens", "spec", "precision", "kappa", "f_measure", or "roc_auc",(for more details see the yardstick package).
preprocess_step_center normalizes dimensions to have a mean of zero; default is set to TRUE. For more info see (step_center in recipes).
preprocess_step_scale  normalize dimensions to have a standard deviation of one. For more info see (step_scale in recipes).
preprocess_PCA Pre-processing threshold for PCA (to skip this step set it to NA). Can select amount of variance to retain (e.g., .90 or as a grid c(0.80, 0.90)); or number of components to select (e.g., 10). Default is "min_halving", which is a function
that selects the number of PCA components based on number of participants and feature (word embedding dimensions) in the data. The formula is: preprocess_PCA = round(max(min(number_features/2), number_participants/2), min(50, number_features))).

penalty
hyper parameter that is tuned

mixture
hyper parameter that is tuned default = 0 (hence a pure ridge regression).

first_n_predictors
by default this setting is turned off (i.e., NA). To use this method, set it to the highest number of predictors you want to test. Then the X first dimensions are used in training, using a sequence from Kjell et al., 2019 paper in Psychological Methods. Adding 1, then multiplying by 1.3 and finally rounding to the nearest integer (e.g., 1, 3, 5, 8). This option is currently only possible for one embedding at the time.

impute_missing
default FALSE (can be set to TRUE if something else than wordembeddings are trained).

method_cor
Type of correlation used in evaluation (default "pearson"; can set to "spearman" or "kendall").

model_description
Text to describe your model (optional; good when sharing the model with others).

multi_cores
If TRUE it enables the use of multiple cores if the computer system allows for it (i.e., only on unix, not windows). Hence it makes the analyses considerably faster to run. Default is "multi_cores_sys_default", where it automatically uses TRUE for Mac and Linux and FALSE for Windows.

save_output
Option not to save all output; default "all". see also "only_results" and "only_results_predictions".

seed
Set different seed.

... For example settings in yardstick::accuracy to set event_level (e.g., event_level = "second").

Value
A (one-sided) correlation test between predicted and observed values; tibble of predicted values, as well as information about the model (preprocessing_recipe, final_model and model_description).

See Also
see textEmbedLayerAggregation textTrainLists textTrainRandomForest textSimilarityTest

Examples

```r
results <- textTrainRegression(
  wordembeddings4$harmonytext,
  Language_based_assessment_data_8$hilstotal,
  multi_cores = FALSE # This is FALSE due to CRAN testing and Windows machines.
)
```
Description
The dataset is a shortened version of the data sets of Study 3-5 from Kjell, Kjell, Garcia and Sikström 2018.

Usage
wordembeddings4

Format
A list with word embeddings for harmony words, satisfaction words, harmony text, satisfaction text and decontextualized word embeddings. BERT-base embeddings based on mean aggregation of layer 11 and 12.

words words
n word frequency
Dim1:Dim768 Word embeddings dimensions

Source
https://psyarxiv.com/er6t7/
Index

* datasets
  centrality_data_harmony, 2
  DP_projections_HILS_SWLS_100, 3
  embeddings_from_huggingface2, 4
  Language_based_assessment_data_3_100, 4
  Language_based_assessment_data_8, 5
  PC_projections_satisfactionwords_40, 6
  wordembeddings4, 36

centrality_data_harmony, 2
DP_projections_HILS_SWLS_100, 3
embeddings_from_huggingface2, 4
Language_based_assessment_data_3_100, 4
Language_based_assessment_data_8, 5
PC_projections_satisfactionwords_40, 6
wordembeddings4, 36
textCentrality, 6, 10
textCentralityPlot, 7, 7
textEmbed, 10, 14–16
textEmbedLayerAggregation, 12, 13, 15, 35
textEmbedLayersOutput, 12, 14, 14
textEmbedStatic, 15
textPCA, 16, 19
textPCAPlot, 16, 17
textPredict, 19
textProjection, 7, 10, 20, 25
textProjectionPlot, 22
textSimilarity, 25, 26
textSimilarityNorm, 26, 26
textSimilarityTest, 20, 26, 27, 29, 33, 35
textTrain, 20, 28, 30
textTrainLists, 20, 29, 29, 33, 35
textTrainRandomForest, 20, 29, 30, 31, 35
textTrainRegression, 29, 30, 33
wordembeddings4, 36

37