Package 'LLMing'

August 27, 2026

Title Large Language Model (LLM) Tools for Psychological Text Analysis

Version 1.3.0

Maintainer Lindley Slipetz <ddj6tu@virginia.edu>

Description A collection of large language model (LLM) text analysis methods designed with psychological data in mind. Currently, LLMing (aka ``lemming'') includes a text anomaly detection method based on the angle-based subspace approach described by Zhang, Lin, and Karim (2015) and a text generation method. <doi:10.1016/j.ress.2015.05.025>.

License MIT + file LICENSE

Encoding UTF-8

Imports Rdpack, quanteda, stopwords, stringi, dbscan, pracma, stats, quanteda, stopwords, stringi, utils, caret, word2vec, keras3

SystemRequirements Python (>= 3.10) with packages: torch, transformers, pandas, numpy

RdMacros Rdpack

URL https://github.com/sliplr19/LLMing

BugReports https://github.com/sliplr19/LLMing/issues

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Lindley Slipetz [aut, cre],
Teague Henry [aut],
Siqi Sun [ctb]

Depends R (>= 4.1.0)

Repository CRAN

Date/Publication 2026-08-27 02:30:02 UTC

Contents

`clean_texts` *Clean texts before embeddings*

Description

Converted to lowercase with numbers, punctuations, slashes, extra whitespace, and stopwords removed. Stemming is also applied

Usage

```
clean_texts(dat, text_col)
```

Arguments

`dat`	Dataframe containing a text column
`text_col`	Name of text column in dat

Value

Vector of clean text

construct_validity *Evaluate Construct Validity of Text Embeddings*

Description

Evaluates the construct validity of text embeddings by testing how well embeddings predict a continuous construct score in held-out data using an artificial neural network.

Usage

```
construct_validity(
  dat,
  severity_col,
  text_col,
  seed = 973,
  p = 0.8,
  embed_method = c("Qwen", "NV", "e5", "word2vec"),
  batch_size = NULL,
  python_path = NULL,
  cache_dir = NULL,
  model = NULL,
  python_script = NULL,
  clean = TRUE,
  word2vec_dim = 50L,
  word2vec_iter = 20L,
  word2vec_window = 5L,
  word2vec_threads = 1L,
  verbose = TRUE
)
```

Arguments

dat	Dataframe containing the text column.
severity_col	Name of the column that contains scores
text_col	Name of the text column in dat.
seed	Seed for train/test split
p	Proportion in training set
embed_method	Embedding method to use: "Qwen", "NV", "e5", or "word2vec".
batch_size	Positive integer batch size. If NULL, method-specific defaults are used.
python_path	Path to a Python executable. Ignored for word2vec.
cache_dir	Hugging Face cache directory. Ignored for word2vec.
model	Optional Hugging Face model ID. If NULL, uses the package default for the selected embedding method.
python_script	Optional path to the Python.

clean	Logical. If TRUE, apply the package's clean_texts() function before embedding.
word2vec_dim	Embedding dimension for word2vec.
word2vec_iter	Number of word2vec training iterations.
word2vec_window	Context window for word2vec.
word2vec_threads	Number of threads for word2vec.
verbose	Logical. Print Python subprocess output.

Details

The function splits observations into training and test sets, generates text embeddings, standardizes the embeddings using training-set statistics, fits a neural network regression model, and predicts construct scores in the held-out test set.

Value

The test dataframe with an added column for predicted scores

---

embed *Generate document embeddings*

Description

Generate one embedding vector per row of a data frame using Qwen, NV-Embed, E5, or a locally trained word2vec model.

Usage

```
embed(
  dat,
  embed_method = c("Qwen", "NV", "e5", "word2vec"),
  text_col,
  batch_size = NULL,
  python_path = NULL,
  cache_dir = NULL,
  model = NULL,
  python_script = "embed.py",
  clean = TRUE,
  word2vec_dim = 50L,
  word2vec_iter = 20L,
  word2vec_window = 5L,
  word2vec_threads = 1L,
  verbose = TRUE
)
```

Arguments

dat	Dataframe containing the text column.
embed_method	Embedding method: "Qwen", "NV", "e5", or "word2vec".
text_col	Name of the text column in dat.
batch_size	Positive integer batch size. If NULL, method-specific defaults are used.
python_path	Path to a Python executable. If NULL, searches for "python3" or "python". Ignored for word2vec.
cache_dir	Hugging Face cache directory. If NULL, uses the package user cache directory. Ignored for word2vec.
model	Optional Hugging Face model ID. If NULL, uses the package default for the selected embedding method.
python_script	Path to Python embedding script. Defaults to package script.
clean	Logical. If TRUE, apply the package's clean_texts() function before embedding.
word2vec_dim	Embedding dimension for word2vec.
word2vec_iter	Number of word2vec training iterations.
word2vec_window	Context window for word2vec.
word2vec_threads	Number of threads for word2vec.
verbose	Logical. Print Python subprocess output.

Value

A dataframe with one row per input row and one column per embedding dimension.

G_thres *Thresholding of pCOS dataframe*

Description

Converts each column of a pCOS score matrix into binary indicators

Usage

```
G_thres(pCOS_mat, theta)
```

Arguments

pCOS_mat	Dataframe of pCOS values
theta	Numeric threshold

Value

A matrix of 0s and 1s of which cells meet the threshold

Examples

```
z_dat <- data.frame("A" = rnorm(500,0,1), "B" = rnorm(500,0,1), "C" = rnorm(500,0,1))
snn <- sim_SNN(z_dat, 10, 5)
vec_snn <- vector_SNN(z_dat, snn)
pCOSdat <- pCOS(z_dat, vec_snn)
G <- G_thres(pCOSdat, theta = 0.1)
```

---

normahalo *Local outlier score*

Description

Computes a normalized Mahalanobis distance score. Only features with nonzero scores in S receive nonzero Mahalanobis scores.

Usage

```
normahalo(z, rs, S)
```

Arguments

z	Dataframe of z scores
rs	List of reference sets
S	Dataframe of numeric values

Value

A dataframe of local outlier scores

---

pCOS *pCOS scores for every row of dataframe*

Description

Applies pCOS_row() to corresponding rows of two data frames, returning one pCOS value per row.

Usage

```
pCOS(z_dat, vec_SNN)
```

Arguments

z_dat	Numeric dataframe, typically z-scores
vec_SNN	Numeric dataframe, typically the output of vector_SNN

Value

A dataframe with same dimensions as z_dat

---

pCOS_row *Pairwise cosine-style row score*

Description

Given two numeric vectors, computes an average cosine-based similarity.

Usage

```
pCOS_row(z, v_SNN)
```

Arguments

z	Numeric vector
v_SNN	Numeric vector, same size as z

Value

A numeric vector

---

rep_set *The vectors of the shared nearest neighbors*

Description

Creates a list of the vectors of the top shared nearest neighbors for each row of the z dataframe

Usage

```
rep_set(z, snn)
```

Arguments

z	Dataframe of values of reference set
snn	Dataframe of shared nearest neighbors indices

Value

A list of dataframes where each row of the dataframe is the vector representation of a given shared nearest neighbor

sim_SNN *Compute shared nearest neighbors*

Description

Builds a shared nearest neighbors matrix and, for each row (observation), returns the indices of the top neighbors with the largest SNN overlap counts

Usage

```
sim_SNN(z_dat, k, tops)
```

Arguments

z_dat	A dataframe with numeric columns
k	An integer representing number of nearest neighbors
tops	An integer representing how many of shared nearest neighbors to return

Value

A dataframe of top rows with shared nearest neighbors

textanomaly *Text anomaly score*

Description

Text anomaly detection method adapted from (Zhang et al. 2015).

Usage

```
textanomaly(dat, k, tops, theta, text_method)
```

Arguments

dat	A dataframe with text data, one text per row
k	An integer representing number of nearest neighbors
tops	An integer representing how many of shared nearest neighbors to return
theta	Numeric threshold
text_method	Character scalar specifying the embedding method. One of "E5", "Qwen3", "NV-Embed", "BERT", or "GloVe"

Value

Dataframe of local outlier score

References

Zhang L, Lin J, Karim R (2015). "An angle-based subspace anomaly detection approach to high-dimensional data: With an application to industrial fault detection." *Reliability Engineering & System Safety*, **142**, 482–497. ISSN 0951-8320, doi:10.1016/j.ress.2015.05.025.

`text_datagen` *Generate Synthetic Text Using an Ollama Language Model*

Description

Generates synthetic text using a locally available Ollama language model. Generation conditions are supplied in an input CSV, and few-shot examples are supplied in a separate example CSV. Prompt content, severity instructions, model settings, and generation settings can be customized.

Usage

```
text_datagen(
  prompt_info_csv,
  examples_csv,
  output_csv,
  model,
  system_prompt,
  prompt_template,
  items,
  severity_instructions,
  python_script = "text_datagen.py",
  python_path = "python",
  severity_min = 10,
  severity_max = 90,
  label_order = c("minimum", "moderate", "severe"),
  label_order_double = c("minimum", "moderate", "severe", "severe", "moderate",
    "minimum"),
  batch_size = 2L,
  max_retries = 2L,
  temperature = 0.7,
  top_p = 0.9,
  repeat_penalty = 1.05,
  num_predict = 1200L,
  num_ctx = 8192L,
  min_words = 80L,
  max_words = 400L,
  require_single_paragraph = TRUE,
  verbose = TRUE
)
```

Arguments

`prompt_info_csv`
: Character string. Path to the CSV containing the generation conditions. The file must contain a `severity` column and may contain `num` and `seed` columns. `num` controls the number of examples selected from each example category, and `seed` controls random example selection.

`examples_csv`
: Character string. Path to the CSV containing few-shot text examples. The file must contain `label` and `text` columns.

`output_csv`
: Character string. Path where the CSV containing the generated text will be written.

`model`
: Character string. Name of the Ollama model used for text generation, for example `"llama3:8b"`.

`system_prompt`
: Character string. System-level instruction supplied to the language model.

`prompt_template`
: Character string. Template used to construct the user prompt. The Python worker should replace `{severity}`, `{severity_instructions}`, `{items}`, and `{example_section}` with the corresponding information for each generation.

`items`
: Character string. Questionnaire items, construct description, scoring information, or other measurement information supplied to the language model.

`severity_instructions`
: A data.frame with columns `min`, `max`, and `instructions`. Each row defines the instructions associated with a range of severity values.

`python_script`
: Character. Path to the Python script used to generate the synthetic text.

`python_path`
: Character string. Python executable used to run the package's Python generation script. Defaults to `"python"`.

`severity_min`
: Numeric. Minimum valid severity value. Defaults to 10.

`severity_max`
: Numeric. Maximum valid severity value. Defaults to 90.

`label_order`
: Character vector. Order in which example categories are selected when `num = 1`.

`label_order_double`
: Character vector. Order in which example categories are selected when `num = 2`.

`batch_size`
: Integer. Batch-size argument supplied to the Python generation script. Defaults to 2.

`max_retries`
: Integer. Number of retries permitted after an invalid or failed model response. Defaults to 2.

`temperature`
: Numeric. Ollama sampling temperature. Higher values generally produce more variable output. Defaults to 0.7.

`top_p`
: Numeric. Nucleus-sampling probability passed to Ollama. Defaults to 0.9.

`repeat_penalty`
: Numeric. Repetition penalty passed to Ollama. Defaults to 1.05.

`num_predict`
: Integer. Maximum number of tokens that Ollama may generate for each response. Defaults to 1200.

`num_ctx`
: Integer. Context-window size supplied to Ollama. Defaults to 8192.

`min_words` Integer. Minimum number of words required for a generated response to be considered valid. Defaults to 80.

`max_words` Integer. Maximum number of words permitted for a generated response to be considered valid. Defaults to 400.

`require_single_paragraph` Logical. If `TRUE`, responses containing multiple paragraphs are considered invalid. Defaults to `TRUE`.

`verbose` Logical. If `TRUE`, output from the Python subprocess is printed to the R console. Defaults to `TRUE`.

Details

The Python generation script `text_datagen.py` is included with the package and is located automatically. Users do not need to provide the path to the Python script.

Value

A data.frame containing the original generation-condition columns and a `response` column containing the generated text.

`vector_SNN` *Aggregate dataframe into mean feature vectors Aggregrate dataframe into mean feature vectors*

Description

For each row of the SNN index matrix, this function takes the rows of reference dataframe, z, and computes their column means, yielding one mean vector per observation.

Usage

```
vector_SNN(z, snn)
```

Arguments

`z` Numeric dataframe

`snn` Dataframe of shared nearest neighbors indices

Value

Dataframe of same dimensions as z

z_score *Z-score on columns*

Description

Z-score on columns

Usage

```
z_score(dat)
```

Arguments

dat	A dataframe with numeric cells

Value

A dataframe with numeric cells with the same dimensions as dat

Index