

Package: tbnb (via r-universe)

July 21, 2026

Type Package

Title Threshold-Based and Iterative Threshold-Based Naive Bayes Classifier

Version 0.1.0

Description Implements the Threshold-Based Naive Bayes (Tb-NB) classifier and its iterative refinement (iTb-NB) for binary sentiment / text classification problems. The classifier computes a continuous log-likelihood ratio score per document and uses a data-driven decision threshold estimated via K-fold cross-validation on a user-selected criterion (accuracy, F1 score, Matthews correlation coefficient, balanced error, etc.). An optional iterative refinement procedure locally re-estimates the threshold in regions of class overlap using either Gaussian kernel density estimation or a Central Limit Theorem bootstrap approximation. The package exposes an idiomatic R formula + data.frame interface together with a 'quanteda'-based text preprocessing pipeline, supports user-supplied document-feature matrices, and includes an optional word-embedding extension that augments the Bag-of-Words with K nearest semantic neighbours of each token. The package additionally implements the p-value extension proposed by Romano (2025) for both document- and feature-level interpretability via `tbnb_pvalues()`. Methods are described in Romano, Contu, Mola, Conversano (2024) <[doi:10.1007/s11634-023-00536-8](https://doi.org/10.1007/s11634-023-00536-8)>, Romano, Zammarchi, Conversano (2024) <[doi:10.1007/s10260-023-00721-1](https://doi.org/10.1007/s10260-023-00721-1)>, and Romano (2025) <[doi:10.1007/978-3-031-96736-8_41](https://doi.org/10.1007/978-3-031-96736-8_41)>.

License GPL (>= 3)

Encoding UTF-8

Language en-GB

LazyData true

Depends R (>= 4.0)

Imports Matrix, methods, stats, grDevices, graphics, quanteda (>= 3.0.0)

Suggests text2vec, stopwords, SnowballC, cld2, cld3, dbscan,
viridisLite, testthat (>= 3.0.0), ggplot2

Config/testthat/edition 3

RoxygenNote 7.3.3

NeedsCompilation no

Author Maurizio Romano [aut, cre]

Maintainer Maurizio Romano <romano.maurizio@unica.it>

Repository <https://cran.r-universe.dev>

Date/Publication 2026-07-21 09:30:07 UTC

RemoteUrl <https://github.com/cran/tbnb>

RemoteRef HEAD

RemoteSha faa86269cf5f9a5b901b1d2fe9a0fab1d641af93

Contents

itbnb	2
plot.tbnb	6
predict.tbnb	7
summary.tbnb	8
tbnb_available_criteria	8
tbnb_categorize	9
tbnb_embedding	12
tbnb_hospitality_seeds	12
tbnb_iterate	13
tbnb_metrics	14
tbnb_optimize_threshold	14
tbnb_posthoc_plots	15
tbnb_preprocess	17
tbnb_pvalues	19
toy_reviews	21
Index	22

itbnb	<i>Fit a Threshold-Based Naive Bayes classifier (Tb-NB / iTb-NB)</i>
-------	--

Description

itbnb() fits the Threshold-Based Naive Bayes classifier proposed by Romano *et al.* (2024) and, optionally, its iterative refinement (iTb-NB). The method is designed for **binary classification of text documents** but accepts any binary Bag-of-Words-style feature matrix.

Usage

```
itbnb(...)

## S3 method for class 'formula'
itbnb(
  formula,
  data,
  subset,
  na.action,
  preprocess = tbnb_preprocess(),
  embedding = NULL,
  alpha = 1,
  fit_prior = TRUE,
  class_prior = NULL,
  optimize_threshold = TRUE,
  criterion = "balanced_error",
  tau_grid = "observed",
  n_tau = 50L,
  K = 5L,
  random_state = 42L,
  iterative = FALSE,
  iter_mode = c("kde", "clt"),
  p_iter = 0.2,
  s_iter = 20L,
  clt_n_boot = 500L,
  clt_sample_size = 30L,
  language_col = NULL,
  ...
)

## Default S3 method:
itbnb(
  x,
  y,
  preprocess = NULL,
  embedding = NULL,
  alpha = 1,
  fit_prior = TRUE,
  class_prior = NULL,
  optimize_threshold = TRUE,
  criterion = "balanced_error",
  tau_grid = "observed",
  n_tau = 50L,
  K = 5L,
  random_state = 42L,
  iterative = FALSE,
  iter_mode = c("kde", "clt"),
  p_iter = 0.2,
```

```

    s_iter = 20L,
    clt_n_boot = 500L,
    clt_sample_size = 30L,
    doc_languages = NULL,
    ...
)

```

Arguments

...	Currently unused.
formula	A formula of the form $y \sim \text{text_col}$. Only one RHS term is supported (the text column).
data	A <code>data.frame</code> containing the response and text columns.
subset	Optional logical / integer vector for row subsetting (only used by the formula interface).
na.action	What to do with NA values (formula interface).
preprocess	A configuration produced by <code>tbnb_preprocess()</code> . Ignored when using the matrix interface.
embedding	A configuration produced by <code>tbnb_embedding()</code> to enable the semantic-augmentation extension, or NULL (default).
alpha	Laplace smoothing parameter (default 1).
fit_prior	If TRUE, class priors are estimated from y ; otherwise <code>class_prior</code> must be supplied.
class_prior	Numeric vector of length 2 with class probabilities, used only when <code>fit_prior = FALSE</code> .
optimize_threshold	If TRUE (default), the decision threshold τ is optimised via K-fold cross-validation.
criterion	One of <code>tbnb_available_criteria()</code> ; the criterion used to pick the final τ .
tau_grid	Either "observed" (percentile-based grid), a numeric vector of candidate thresholds, or <code>list(low = , high =)</code> .
n_tau	Number of grid points when <code>tau_grid = "observed"</code> .
K	Number of CV folds.
random_state	Random seed for CV splits and CLT bootstrap.
iterative	If TRUE, applies the iterative refinement (iTb-NB) <i>after</i> the global threshold has been estimated.
iter_mode	"kde" or "clt" (see <code>tbnb_iterate()</code>).
p_iter, s_iter, clt_n_boot, clt_sample_size	Iterative refinement parameters; see <code>tbnb_iterate()</code> .
language_col	Optional name of a column in <code>data</code> containing the language of each document (ISO 639-1 code like "it"/"en" or a language name like "italian"/"english"). When supplied and <code>preprocess\$stem_language = "per_document"</code> , the per-document languages are taken from this column and no language detection is run (cld2 / cld3 are not invoked). Formula interface only.

x	Alternative to formula/data: a feature matrix.
y	Alternative to formula/data: the binary response vector.
doc_languages	Optional character vector of per-document language codes / names. Default and matrix interfaces only; same purpose as language_col.

Details

Two interfaces are provided:

1. Formula + data.frame (idiomatic R):

```
itbnb(sentiment ~ text, data = reviews, ...)
```

where the right-hand side names a single column of data that contains the raw text. The text is preprocessed internally with **quanteda** according to the `tbnb_preprocess()` configuration.

2. Matrix / dfm + response (advanced / pipeline use):

```
itbnb(x = dfm, y = sentiment, ...)
```

where x is an already-built binary document-feature matrix (any of: base matrix, Matrix sparse, quanteda::dfm, data.frame).

Value

An object of class "tbnb" with components: call, terms, xlevels, text_col, preprocess, embedding, feature_names, classes, class_prior, class_occurrences, feature_counts, log_pres, log_abs, alpha, threshold, criterion, optimizer, iterative, decisions.

References

Romano M., Contu G., Mola F., Conversano C. (2024). [doi:10.1007/s11634023005368](https://doi.org/10.1007/s11634023005368)

Romano M., Zammarchi G., Conversano C. (2024). [doi:10.1007/s10260023007211](https://doi.org/10.1007/s10260023007211)

Examples

```
# Toy synthetic dataset (formula interface)
reviews <- data.frame(
  sentiment = rep(c("pos", "neg"), each = 5),
  text = c(
    "great hotel friendly staff lovely view",
    "amazing room comfortable bed great breakfast",
    "wonderful stay clean rooms helpful staff",
    "fantastic location nice view comfortable",
    "lovely breakfast helpful staff great room",
    "dirty room rude staff terrible noise",
    "awful stay smelly room bad service",
    "noisy room rude reception dirty bathroom",
    "terrible breakfast cold food bad service",
    "horrible stay rude staff dirty noisy"
  ),
  stringsAsFactors = FALSE
)
```

```
fit <- itbnb(sentiment ~ text, data = reviews,
             preprocess = tbnb_preprocess(language = "english",
                                           min_count = 1, min_docfreq = 1),
             K = 2, n_tau = 10, iterative = FALSE)
predict(fit, newdata = reviews[1:3, ], type = "class")
summary(fit)
```

plot.tbnb

Plot the distribution of decision scores by class

Description

Plot the distribution of decision scores by class

Usage

```
## S3 method for class 'tbnb'
plot(x, newdata = NULL, y = NULL, breaks = 40, ...)
```

Arguments

x	A fitted tbnb model.
newdata	Optional data to score (default: scores cannot be recovered from the model alone, so newdata is required).
y	Optional response vector aligned with newdata, used to colour the score distribution by true class.
breaks	Histogram breaks.
...	Additional graphical parameters.

Value

Invisibly, the numeric vector of decision scores computed for newdata, or NULL when newdata is not supplied. The function is called for its side effect of drawing the score distribution together with the estimated threshold(s).

predict.tbnb	<i>Predictions from a fitted tbnb model</i>
--------------	---

Description

Predictions from a fitted tbnb model

Usage

```
## S3 method for class 'tbnb'
predict(
  object,
  newdata,
  type = c("class", "score", "prob", "raw"),
  threshold = NULL,
  iterative = NULL,
  doc_languages = NULL,
  ...
)
```

Arguments

object	A fitted model of class "tbnb".
newdata	Either a data.frame (when the model was fitted with the formula interface), a character vector of documents, or a feature matrix / dfm with the same vocabulary as the training data.
type	One of: • "class" – predicted class labels (factor with the original levels). • "score" – continuous log-likelihood ratio score. • "prob" – pseudo-probability obtained by applying a logistic squash to the (shifted) score ($1 / (1 + \exp(-(score - threshold)))$). • "raw" – integer label in $\{0, 1\}$.
threshold	Optional override for the decision threshold (default: the one stored in object\$threshold).
iterative	Optional logical to override object\$iterative at prediction time (e.g. for ablation: compare plain vs. iterative).
doc_languages	Optional character vector of per-document language codes / names for newdata. If NULL and the model was fitted with language_col, the values are taken from that column of newdata.
...	Currently unused.

Value

A vector of the requested type.

summary.tbnb

Summary of a fitted Tb-NB / iTb-NB model

Description

Summary of a fitted Tb-NB / iTb-NB model

Usage

```
## S3 method for class 'tbnb'
summary(
  object,
  pvalues = FALSE,
  pvalue_method = c("clt", "kde"),
  pvalue_alpha = 0.05,
  ...
)
```

Arguments

object	A fitted "tbnb" model.
pvalues	If TRUE, the summary additionally reports the significant features according to the p-value extension of Romano (2025) (see tbnb_pvalues()).
pvalue_method	Either "clt" (default) or "kde"; passed to tbnb_pvalues() when pvalues = TRUE.
pvalue_alpha	Significance level used to flag features.
...	Passed on.

Value

An object of class "tbnb_summary": a list collecting the class labels, the vocabulary size, the class priors, the Laplace smoothing parameter, the selection criterion and the estimated threshold, the most positive and most negative words, the cross-validated threshold-per-criterion table and, when pvalues = TRUE, the feature-level p-value table. A print method is provided.

tbnb_available_criteria

Names of the criteria supported by [itbnb\(\)](#) for threshold optimisation

Description

Names of the criteria supported by [itbnb\(\)](#) for threshold optimisation

Usage

```
tbnb_available_criteria()
```

Value

Character vector of criterion names.

tbnb_categorize	<i>Categorise the vocabulary of a fitted Tb-NB / iTb-NB model</i>
-----------------	---

Description

Produces a partition of the model vocabulary into a finite set of user-meaningful categories that can then be fed into [tbnb_posthoc_plots\(\)](#) for the post-hoc analyses described in Romano *et al.* (2024) — e.g. how the sentiment of *food*, *staff*, *cleanliness* etc. evolves over time, or differs across hotels.

Usage

```
tbnb_categorize(
  object,
  method = c("dbscan", "seed", "kmeans", "manual"),
  seeds = NULL,
  k = NULL,
  eps = NULL,
  minPts = 5L,
  n_categories_target = 10L,
  min_category_size = 5L,
  assignments = NULL,
  embedding = NULL,
  text_corpus = NULL,
  embedding_dim = 50L,
  embedding_iter = 15L,
  embedding_window = 5L,
  refine = FALSE,
  min_similarity = 0,
  features = NULL,
  hotel_reviews = FALSE,
  verbose = TRUE
)
```

Arguments

object	A fitted "tbnb" model.
method	One of "dbscan" (default, automatic K), "seed" (semi-supervised), "kmeans" (manual K), "manual". Overridden when hotel_reviews = TRUE (see below).

seeds	For method = "seed": a named list, with category names as names and seed words as character vectors (e.g. <code>list(food = c("squisita", "deliziosa"), staff = c("gentile", "cortese"))</code>).
k	For method = "kmeans": number of clusters.
eps, minPts	For method = "dbscan": see <code>dbscan::dbscan()</code> . If eps is NULL, it is auto-estimated by scanning the minPts-th nearest-neighbour distance distribution and picking the value that yields roughly <code>n_categories_target</code> non-noise clusters. Falls back to the 10th percentile if the scan fails.
n_categories_target	For method = "dbscan": target number of non-noise clusters used when eps is auto-estimated. Default 10.
min_category_size	Categories with fewer than <code>min_category_size</code> words are collapsed into "uncategorized" (post-clustering cleanup, applied to all methods). Default 5.
assignments	For method = "manual": a named character vector (names = words, values = category) or a 2-column data.frame.
embedding	Optional. Pre-trained word vectors as a numeric matrix with row names = tokens, or a <code>tbnb_embedding</code> object. Required by seed, kmeans and dbscan unless <code>text_corpus</code> is supplied for on-the-fly training.
text_corpus	Optional character vector of training texts used to train a GloVe embedding when embedding is missing. The texts are preprocessed using the configuration stored in the fitted model (so the vocabulary is consistent).
embedding_dim, embedding_iter, embedding_window	text2vec GloVe hyperparameters (used only when training on-the-fly).
refine	For method = "seed": if TRUE, iterate centroids k-means-style.
min_similarity	For method = "seed": words whose maximum cosine similarity to any seed centroid is below this threshold are sent to "uncategorized". Default 0.
features	Optional character vector to restrict categorisation to a subset of the model vocabulary. Default: all features in <code>object\$feature_names</code> .
hotel_reviews	Logical (default FALSE). If TRUE, the function forces method = "seed" and uses the 12 reference categories of Romano <i>et al.</i> (2024) for the Booking.com data — <i>cleaning, comfort, position, price-quality-rate, services, staff, wifi, bar, food, hotel, room, sleep-quality</i> — with bilingual IT+EN seed words (see <code>tbnb_hospitality_seeds()</code>). The 13th category from the paper, "other", is handled automatically as "uncategorized". When FALSE (default), behaviour is unchanged and the value of method is honoured (default: "dbscan"). Use this flag whenever the input is hospitality / hotel reviews.
verbose	Print progress messages.

Details

Four methods are supported:

- method = "seed" (semi-supervised). The user supplies a named list of *seed words* per category; each remaining vocabulary word is assigned to the category whose centroid (mean of seed vectors) is closest in the embedding space (cosine similarity). With `refine = TRUE` the

procedure iterates k-means-style by recomputing centroids from the current assignments until convergence.

- `method = "kmeans"` (unsupervised, `K` chosen). Standard k-means on the word vectors. Clusters are named `cluster_1`, ..., `cluster_K`.
- `method = "dbscan"` (unsupervised, `K` automatic). Density-based clustering with `eps` and `minPts`. Noise points are sent to the special category "uncategorized". Requires the **dbscan** package.
- `method = "manual"`. The user passes a named character vector (words -> category) or a `data.frame` with columns `word`, `category`. No clustering is performed.

For the three embedding-based methods, the user may supply a pre-trained embedding (`embedding =`); otherwise a GloVe model is trained on the fly on `text_corpus` using **text2vec**.

Value

An S3 object of class "tbnb_categorization" with components:

- `assignments` — `data.frame(word, category, log_odds, similarity)` (the similarity column is NA for manual / dbscan / kmeans).
- `categories` — character vector of category names.
- `method` — the method used.
- `centroids` — for seed / kmeans: matrix of category centroids.
- `embedding_used` — the embedding matrix actually used (or NULL).

References

Romano M., Contu G., Mola F., Conversano C. (2024). *Threshold-based Naive Bayes classifier*. *Advances in Data Analysis and Classification*, 18, 325–361. doi:[10.1007/s11634023005368](https://doi.org/10.1007/s11634023005368)

Examples

```
data(toy_reviews)
fit <- itbnb(sentiment ~ text, data = toy_reviews,
            preprocess = tbnb_preprocess(language = "english"))

# Manual categorisation (no embedding required)
cat_man <- tbnb_categorize(
  fit, method = "manual",
  assignments = c(staff = "service", room = "room",
                  breakfast = "food", view = "view"))
print(cat_man)
```

tbnb_embedding	Configure the word-embedding semantic augmentation of the BoW
----------------	---

Description

At fit / predict time, for each token w in a document, the top- k most similar tokens in the embedding space are added to the document with weight 1 (presence). This enriches the Bag-of-Words with synonyms / semantic neighbours, which can help mitigate sparsity issues and out-of-vocabulary problems while leaving the Tb-NB word-interpretability intact.

Usage

```
tbnb_embedding(vectors, k = 3L, min_similarity = 0.5, self_include = TRUE)
```

Arguments

vectors	Either: • a numeric matrix of shape [vocab x dim] with row names = tokens, or • a path to a text file in GloVe / word2vec format (one token per line: <token> <v1> <v2> ...).
k	Number of nearest semantic neighbours to add per token.
min_similarity	Minimum cosine similarity to keep a neighbour.
self_include	Whether to keep the original token itself among the neighbours (it would otherwise always be the top-1 with similarity 1). Should usually be TRUE.

Details

This is an *extension* not present in the original Tb-NB / iTb-NB papers. It is disabled by default and must be explicitly enabled by passing the returned configuration to the embedding argument of `itbnb()`.

Value

A `tbnb_embedding` configuration object.

tbnb_hospitality_seeds	Built-in seed words for hospitality / hotel reviews
------------------------	---

Description

Returns the 12 reference categories used in the Booking.com analysis of Romano *et al.* (2024) — *cleaning, comfort, position, price-quality-rate, services, staff, wifi, bar, food, hotel, room, sleep-quality* — each populated with bilingual Italian + English seed words. The 13th category from the paper, "other", is the residual one and is handled automatically as "uncategorized".

Usage

```
tbnb_hospitality_seeds()
```

Details

Used by `tbnb_categorize()` when `hotel_reviews = TRUE`.

Value

A named list of character vectors (one per category).

tbnb_iterate	<i>Iterative refinement of the decision threshold (iTb-NB)</i>
--------------	--

Description

Starting from a global threshold τ , the algorithm iteratively inspects the uncertainty region around it and proposes a refined local threshold by estimating the intersection of the two class-conditional score densities. The refinement is repeated until convergence, lack of data, or until the modes of the two densities become indistinguishable.

Usage

```
tbnb_iterate(
  scores,
  y,
  tau,
  p = 0.2,
  s = 20L,
  mode = c("kde", "clt"),
  clt_boot = 500L,
  clt_sample = 30L,
  epsilon = 0.001,
  max_iter = 50L
)
```

Arguments

scores	Numeric vector of Tb-NB scores.
y	Binary labels in $\{0, 1\}$.
tau	Initial global threshold.
p	Fraction of samples around τ (per class) defining the uncertainty window.
s	Minimum sample count required to keep iterating. In "kde" mode this is the total count in the window; in "clt" mode this is the per-class minimum.
mode	"kde" (Gaussian kernel density estimate) or "clt" (bootstrap normal approximation).

clt_boot, clt_sample CLT bootstrap settings, ignored when mode = "kde".

epsilon Convergence tolerance both on $|\tau_{new} - \tau|$ and on the mode-distance $|x_+^* - x_-^*|$.

max_iter Hard upper bound on iteration count (safety net).

Value

A data.frame of decisions, one row per refinement step, with columns: iteration, start, end, tau, x_max_pos, x_max_neg, direction ("r" if positive class mode is to the right of negative, "l" otherwise).

tbnb_metrics	<i>Compute all binary classification metrics from confusion counts</i>
--------------	--

Description

Compute all binary classification metrics from confusion counts

Usage

```
tbnb_metrics(TP, FP, TN, FN)
```

Arguments

TP, TN, FP, FN Confusion counts (scalars or vectors of equal length).

Value

A named list with the metrics listed in [tbnb_available_criteria\(\)](#).

tbnb_optimize_threshold	<i>Cross-validated threshold optimisation for a Tb-NB classifier</i>
-------------------------	--

Description

Given an already-fitted Tb-NB model and the training data used to fit it, explores a grid of candidate thresholds τ and reports the best value for each available criterion.

Usage

```
tbnb_optimize_threshold(
  X,
  y,
  alpha = 1,
  fit_prior = TRUE,
  class_prior = NULL,
  tau_grid = "observed",
  n_tau = 50L,
  K = 5L,
  random_state = 42L
)
```

Arguments

X	Binary sparse matrix (presence / absence of features).
y	Integer response in {0, 1}.
alpha	Laplace smoothing parameter.
fit_prior, class_prior	See itbnb() .
tau_grid	Either "observed" (percentile-based grid), a numeric vector of candidates, or <code>list(low = , high =)</code> .
n_tau	Grid size when tau_grid = "observed" or a range.
K	Number of CV folds.
random_state	Random seed.

Value

List with components:

- tau_grid – sorted grid of candidate thresholds;
- metric_mats – named list of K x n_tau matrices, one per metric;
- best_tau – named numeric vector of optimal τ per criterion.

tbnb_posthoc_plots	<i>Generate post-hoc analysis plots from a fitted Tb-NB / iTb-NB model</i>
--------------------	--

Description

Combines the per-word sentiment scores from a fitted Tb-NB / iTb-NB model with a categorisation produced by [tbnb_categorize\(\)](#) to generate the post-hoc plots described in Romano *et al.* (2024) — i.e. how each category contributes to overall sentiment, how sentiment evolves over time, which words are statistically significant within each category, etc.

Usage

```
tbnb_posthoc_plots(
  object,
  categorization,
  newdata,
  time_var = NULL,
  group_var = NULL,
  doc_languages = NULL,
  pvalues = NULL,
  which = c("category_score", "time_series", "feature_volcano", "category_distribution",
    "top_words_per_category"),
  top_n_words = 15L,
  exclude_words = NULL,
  exclude_top_quantile = NULL,
  exclude_carriers = FALSE,
  time_unit = c("month", "quarter", "year", "week"),
  palette = NULL,
  theme = "minimal"
)
```

Arguments

<code>object</code>	A fitted "tbnb" model.
<code>categorization</code>	A <code>tbnb_categorization</code> object produced by <code>tbnb_categorize()</code> .
<code>newdata</code>	Data on which to compute document-level scores (typically the training or test set). Either a <code>data.frame</code> (when the model was fit with the formula interface) or a character vector.
<code>time_var</code>	Name of a column of <code>newdata</code> containing the document timestamp (Date, POSIXct, or character coercible to Date). Required for the time-series plot.
<code>group_var</code>	Optional name of a column of <code>newdata</code> for stratification (e.g. "NomeStruttura", "provincia").
<code>doc_languages</code>	Optional per-document language vector (see <code>predict.tbnb()</code>).
<code>pvalues</code>	Optional output of <code>tbnb_pvalues()</code> (type = "feature") used to colour / annotate the volcano plot.
<code>which</code>	Subset of plots to compute; any of <code>c("category_score", "time_series", "feature_volcano", "category_distribution", "top_words_per_category")</code> .
<code>top_n_words</code>	Number of word labels shown in the volcano and the per-category top-words plot.
<code>exclude_words</code>	Optional character vector of words to drop from all post-hoc plots. Useful for sentiment carriers such as <i>bellissimo</i> , <i>consigliatissimo</i> , <i>orribile</i> that are useful for classification but uninformative for category-level analysis.
<code>exclude_top_quantile</code>	Numeric in $[0, 0.5)$. If provided, automatically drop the words whose $ \log\text{-odds} $ exceeds the $1 - \text{exclude_top_quantile}$ quantile of the per-feature absolute log-odds distribution (i.e. the global sentiment carriers). Typical value: 0.01 (drop the extreme 1%). Default NULL.

exclude_carriers	Logical. If TRUE, apply a built-in list of common Italian / English sentiment-carrier adjectives to drop. Combined (union) with exclude_words and exclude_top_quantile.
time_unit	How to bucket the timestamps for the time-series plot: "month", "quarter", "year", or "week".
palette	Optional vector of category colours (length equal to the number of categories). When NULL, a viridis-based palette is used.
theme	Either a ggplot2 theme object or the name of one of the built-in theme_* functions (without the theme_ prefix: e.g. "minimal", "bw", "classic").

Details

The returned list contains **ggplot2** objects so the user can further customise them, save them to disk via `ggplot2::ggsave()`, or arrange them in panels.

Value

A named list of ggplot objects (the keys are the entries of which).

References

Romano M., Contu G., Mola F., Conversano C. (2024). *Threshold-based Naive Bayes classifier*. *Advances in Data Analysis and Classification*, 18, 325–361. doi:[10.1007/s11634023005368](https://doi.org/10.1007/s11634023005368)

tbnb_preprocess	<i>Configure the text-preprocessing pipeline</i>
-----------------	--

Description

Helper used to build a configuration list to be passed to `itbnb()` via the preprocess argument. The pipeline is implemented on top of **quanteda** and produces a binary document-feature matrix that is fed to the Tb-NB core.

Usage

```
tbnb_preprocess(
  language = "english",
  lowercase = TRUE,
  remove_punct = TRUE,
  remove_numbers = TRUE,
  remove_symbols = TRUE,
  remove_url = TRUE,
  remove_stopwords = TRUE,
  extra_stopwords = NULL,
  stem = FALSE,
  stem_language = NULL,
  detect_engine = c("cld2", "cld3"),
```

```

    min_word_length = 2L,
    min_count = 1L,
    min_docfreq = 1L,
    min_words_per_doc = 3L,
    ngrams = 1L
)

```

Arguments

language	Character. One or more language names accepted by stopwords (e.g. "english", "italian", "spanish"). Use a vector for a multilingual corpus. Set to NULL to skip all language-specific steps.
lowercase	Logical. Convert text to lowercase before tokenisation.
remove_punct	Logical. Drop punctuation tokens.
remove_numbers	Logical. Drop numeric tokens.
remove_symbols	Logical. Drop symbol tokens.
remove_url	Logical. Drop URL tokens.
remove_stopwords	Logical. Drop stopwords for the language(s) in language.
extra_stopwords	Optional character vector with extra stopwords.
stem	Deprecated boolean. Use stem_language instead. When TRUE and stem_language is NULL, it is interpreted as stem_language = language[1].
stem_language	NULL (no stemming), a single language name (use that Snowball stemmer for all tokens), or "per_document" to detect the language of each document and stem accordingly.
detect_engine	Engine used when stem_language = "per_document": "cld2" (default, fast pure-C++) or "cld3" (small neural net). Requires the corresponding suggested package.
min_word_length	Minimum number of characters per token kept.
min_count, min_docfreq	Pruning of rare features. Tokens that appear fewer than min_count times in total, or in fewer than min_docfreq documents, are dropped.
min_words_per_doc	Minimum number of informative tokens (counted after stopwords removal and min_word_length pruning, but before stemming and n-gram expansion) for a document to enter the training set. Documents below this threshold are dropped from the training corpus (with a message). Default 3. Set to 0 or 1 to disable. The filter is applied at fit time only — predict() does not drop short documents.
ngrams	Integer vector with n-gram sizes to keep (default 1 = unigrams only). Use c(1, 2) for unigrams + bigrams.

Details

The function supports both monolingual and **multilingual** corpora: set language to a vector of language names (e.g. `c("italian", "english")`) and the union of the corresponding stopword lists will be removed. Stemming, which is language-specific, is controlled by `stem_language`:

- `stem_language = NULL` (default) — no stemming.
- `stem_language = "<lang>"` — apply the Snowball stemmer for that language to **every** token (a sensible default when one language dominates the corpus; tokens from the minority language are stemmed with the "wrong" rules but in practice this is rarely harmful).
- `stem_language = "per_document"` — detect the language of *each* document with **cld2** or **cld3** (see `detect_engine`) and stem each document with the appropriate Snowball stemmer. Only documents detected as one of the languages in `language` are stemmed; documents detected as another language (or undetected) are left as-is.

The legacy boolean `stem = TRUE` is still accepted for back-compat: it is interpreted as `stem_language = language[1]`.

Value

A named list of class "tbnb_preprocess" to be supplied to `itbnb()`.

tbnb_pvalues	<i>P-values for documents and features (Tb-NB / iTb-NB)</i>
--------------	---

Description

Implements the p-value extension introduced in Romano (2025) for the Tb-NB / iTb-NB classifier. Two flavours are exposed via the `type` argument:

Usage

```
tbnb_pvalues(
  object,
  type = c("document", "feature"),
  newdata = NULL,
  method = c("clt", "kde"),
  alpha = 0.05,
  n_boot = 500L,
  sample_size = 30L,
  two_sided = NULL
)
```

Arguments

object	A fitted "tbnb" model.
type	Either "document" (per-row p-values; requires newdata) or "feature" (per-token p-values; does not need newdata).
newdata	Required when type = "document". Same accepted formats as in <code>predict.tbnb()</code> .
method	Either "clt" (recommended, paper default) or "kde".
alpha	Significance level used to flag features in the significant column.
n_boot	Number of bootstrap resamples for method = "clt".
sample_size	Sample size of each bootstrap draw for method = "clt". Must be ≥ 30 for the CLT approximation to be valid.
two_sided	If TRUE (default for features), p-values are two-sided. For documents the convention is one-sided per class.

Details

- type = "document" — for each row of newdata, the per-class p-value of the observed log-likelihood ratio score $\Lambda(c_i)$ under the training-set score distribution of each class ($f^+(z)$ and $f^-(z)$). The two p-values are one-sided in the direction of class membership, so that *small* p_pos flags strong evidence *against* the "negative" hypothesis (i.e. evidence that the document is positive) and vice-versa.
- type = "feature" — for each token in the vocabulary, the two-sided p-value of its log-odds $\Lambda(w_j)$ against the distribution of log-odds across all features under the null hypothesis of "no discrimination" (vocabulary-wide reference distribution).

Two reference-distribution estimators are supported:

- method = "clt" — bootstrap means with Normal approximation (Central Limit Theorem). This is the procedure proposed in the paper and yields more stable results.
- method = "kde" — Gaussian kernel density estimate of the training distribution, with p-values computed by numerical integration via `stats::approx()`.

Value

An object of class `tbnb_pvalues` (a `data.frame` with attributes).

References

Romano, M. (2025). *Iterative Threshold-based Naive Bayes classifier: further interpretability with p-values*. In E. di Bella, V. Gioia, C. Lagazio & S. Zaccarin (Eds.), *Statistics for Innovation I: SIS 2025, Short Papers, Plenary, Specialized, and Solicited Sessions* (Chap. 41). Italian Statistical Society Series on Advances in Statistics. Springer, Cham. doi:10.1007/9783031967368_41

Examples

```
data(toy_reviews)
fit <- itbnb(sentiment ~ text, data = toy_reviews,
             preprocess = tbnb_preprocess(language = "english"))
```

```
# Document-level p-values
pv_doc <- tbnb_pvalues(fit, type = "document",
                      newdata = toy_reviews,
                      method = "clt")

head(pv_doc)

# Feature-level p-values
pv_feat <- tbnb_pvalues(fit, type = "feature", method = "clt")
head(pv_feat[order(pv_feat$p_value), ], 10)
```

toy_reviews	<i>Toy hotel reviews dataset</i>
-------------	----------------------------------

Description

A small, hand-crafted dataset of 200 short fictional hotel reviews labelled as positive or negative. It is meant exclusively for illustration and unit tests; the wording is intentionally simple so that the Tb-NB classifier behaves intuitively on it.

Usage

```
toy_reviews
```

Format

A data.frame with 200 rows and 2 columns:

sentiment Factor with two levels "neg" and "pos".

text Character. The (short) review text.

Source

Synthetic data generated by the package authors. Inspired by the Booking.com reviews used in Romano *et al.* (2024).

Index

* datasets

toy_reviews, [21](#)

dbscan::dbscan(), [10](#)

ggplot2::ggsave(), [17](#)

itbnb, [2](#)

itbnb(), [8](#), [12](#), [15](#), [17](#), [19](#)

plot.tbnb, [6](#)

predict.tbnb, [7](#)

predict.tbnb(), [16](#), [20](#)

summary.tbnb, [8](#)

tbnb_available_criteria, [8](#)

tbnb_available_criteria(), [4](#), [14](#)

tbnb_categorize, [9](#)

tbnb_categorize(), [13](#), [15](#), [16](#)

tbnb_embedding, [12](#)

tbnb_embedding(), [4](#)

tbnb_hospitality_seeds, [12](#)

tbnb_hospitality_seeds(), [10](#)

tbnb_iterate, [13](#)

tbnb_iterate(), [4](#)

tbnb_metrics, [14](#)

tbnb_optimize_threshold, [14](#)

tbnb_posthoc_plots, [15](#)

tbnb_posthoc_plots(), [9](#)

tbnb_preprocess, [17](#)

tbnb_preprocess(), [4](#), [5](#)

tbnb_pvalues, [19](#)

tbnb_pvalues(), [8](#), [16](#)

toy_reviews, [21](#)