

# Package ‘BERTopic’

January 12, 2026

**Type** Package

**Title** Topic Modeling with 'BERTopic' in R

**Version** 0.1.0

**Description** Interface to the Python package 'BERTopic' for transformer-based topic modeling. Provides R wrappers to fit BERTopic models, transform new documents, update and reduce topics, extract topic- and document-level information, and generate interactive visualizations. Python backends and dependencies are managed via the 'reticulate' package.

**License** MIT + file LICENSE

**Encoding** UTF-8

**Depends** R (>= 3.5)

**Imports** reticulate,  
rlang,  
tibble,  
utils

**Suggests** Matrix,  
htmltools,  
testthat (>= 3.1.0)

**LazyData** true

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.3

**Config/testthat/edition** 3

**URL** <https://github.com/Feng-Ji-Lab/BERTopic>

**BugReports** <https://github.com/Feng-Ji-Lab/BERTopic/issues>

**Language** en-US

## Contents

as.data.frame.bertopic_r . . . . .	2
bertopic_as_document_topic_matrix . . . . .	3
bertopic_available . . . . .	3
bertopic_find_topics . . . . .	4
bertopic_fit . . . . .	4
bertopic_get_document_info . . . . .	5

bertopic_get_representative_docs . . . . .	6
bertopic_has_embedding_model . . . . .	6
bertopic_load . . . . .	7
bertopic_reduce_topics . . . . .	7
bertopic_save . . . . .	8
bertopic_self_check . . . . .	8
bertopic_session_info . . . . .	9
bertopic_set_embedding_model . . . . .	10
bertopic_set_topic_labels . . . . .	10
bertopic_topics . . . . .	11
bertopic_topics_over_time . . . . .	11
bertopic_topic_terms . . . . .	12
bertopic_transform . . . . .	12
bertopic_update_topics . . . . .	13
bertopic_visualize_barchart . . . . .	13
bertopic_visualize_distribution . . . . .	14
bertopic_visualize_documents . . . . .	15
bertopic_visualize_heatmap . . . . .	15
bertopic_visualize_hierarchical_documents . . . . .	16
bertopic_visualize_hierarchy . . . . .	17
bertopic_visualize_term_rank . . . . .	17
bertopic_visualize_topics . . . . .	18
bertopic_visualize_topics_over_time . . . . .	18
bertopic_visualize_topics_per_class . . . . .	19
coef.bertopic_r . . . . .	20
fortify.bertopic_r . . . . .	20
install_py_deps . . . . .	21
install_py_deps_conda . . . . .	21
install_py_deps_venv . . . . .	22
predict.bertopic_r . . . . .	23
print.bertopic_r . . . . .	23
set_bertopic_seed . . . . .	24
sms_spam . . . . .	24
summary.bertopic_r . . . . .	25
use_bertopic . . . . .	25
use_bertopic_condaenv . . . . .	26
use_bertopic_virtualenv . . . . .	26

## Index 28

---

as.data.frame.bertopic\_r  
*Coerce to data.frame*

---

### Description

Coerce to data.frame

### Usage

```
## S3 method for class 'bertopic_r'
as.data.frame(x, ...)
```

**Arguments**

x	A "bertopic_r" model.
...	Unused.

**Value**

A data.frame equal to `bertopic_topics()`.

---

bertopic\_as\_document\_topic\_matrix

*Coerce to a document-topic probability matrix*


---

**Description**

Extract the document-topic probabilities as a matrix. If probabilities were not computed during fitting, returns NULL (with a warning).

**Usage**

```
bertopic_as_document_topic_matrix(model, sparse = TRUE, prefix = TRUE)
```

**Arguments**

model	A "bertopic_r" model object.
sparse	Logical; if TRUE and Matrix is available, returns a sparse matrix.
prefix	Logical; if TRUE, prefix columns as topic ids.

**Value**

A matrix or sparse Matrix of size n\_docs x n\_topics, or NULL.

---

bertopic\_available

*Is Python + BERTopic available?*


---

**Description**

Checks whether the active Python (as initialized by **reticulate**) can import the key modules needed for BERTopic.

**Usage**

```
bertopic_available()
```

**Value**

Logical scalar.

**Examples**

```
## Not run:
bertopic_available()

## End(Not run)
```

---

bertopic_find_topics	<i>Find nearest topics for a query string</i>
----------------------	---

---

**Description**

Use `BERTopic.find_topics()` to retrieve the closest topics for a query string. Augments topic IDs/scores with topic labels when available.

**Usage**

```
bertopic_find_topics(model, query_text, top_n = 5L)
```

**Arguments**

model	A "bertopic_r" model.
query_text	A length-1 character query.
top_n	Number of nearest topics to return.

**Value**

A tibble with columns topic, score, and label.

---

bertopic_fit	<i>Fit BERTopic from R</i>
--------------	----------------------------

---

**Description**

A high-level wrapper around Python 'BERTopic'. Python dependencies are checked at runtime.

**Usage**

```
bertopic_fit(text, embeddings = NULL, ...)
```

**Arguments**

text	Character vector of documents.
embeddings	Optional numeric matrix (n_docs x dim). If supplied, passed through to Python.
...	Additional arguments forwarded to <code>bertopic.BERTopic(...)</code> .

**Value**

An S3 object of class "bertopic\_r" containing:

- .py: the underlying Python model (reticulate object)
- topics: integer vector of topic assignments
- probs: numeric matrix/data frame of topic probabilities (if available)

**Examples**

```
## Not run:
if (reticulate::py_module_available("bertopic")) {
  m <- bertopic_fit(c("a doc", "another doc"))
  print(class(m))
}

## End(Not run)
```

---

bertopic\_get\_document\_info

*Document-level information*

---

**Description**

Retrieve document-level information for the provided documents.

**Usage**

```
bertopic_get_document_info(model, docs)
```

**Arguments**

model	A "bertopic_r" model.
docs	Character vector of documents to query (required).

**Value**

A tibble with document-level information.

---

`bertopic_get_representative_docs`*Representative documents for a topic*

---

**Description**

Retrieve representative documents for a given topic using `BERTopic.get_representative_docs()`. Falls back across signature variants.

**Usage**

```
bertopic_get_representative_docs(model, topic_id, top_n = 5L)
```

**Arguments**

<code>model</code>	A "bertopic_r" model.
<code>topic_id</code>	Integer topic id.
<code>top_n</code>	Number of representative documents to return.

**Value**

A tibble with columns rank and document. If scores are available in the current BERTopic version, a score column is included.

---

`bertopic_has_embedding_model`*Does the model have a usable embedding model?*

---

**Description**

Does the model have a usable embedding model?

**Usage**

```
bertopic_has_embedding_model(model)
```

**Arguments**

<code>model</code>	A "bertopic_r" model.
--------------------	-----------------------

**Value**

Logical; TRUE if `embedding_model` is present and not None.

---

bertopic_load	<i>Load a BERTopic model</i>
---------------	------------------------------

---

**Description**

Load a BERTopic model from disk that was saved with `bertopic_save()`.

**Usage**

```
bertopic_load(path)
```

**Arguments**

path	Path used in <code>bertopic_save()</code> (file or directory).
------	--

**Value**

A "bertopic\_r" object with the loaded Python model.

---

bertopic_reduce_topics	<i>Reduce/merge topics</i>
------------------------	----------------------------

---

**Description**

Wrapper over Python `reduce_topics`, compatible with multiple signatures.

**Usage**

```
bertopic_reduce_topics(  
    model,  
    nr_topics = "auto",  
    representation_model = NULL,  
    docs = NULL  
)
```

**Arguments**

model	A "bertopic_r" model.
nr_topics	Target number (integer) or "auto".
representation_model	Optional Python representation model.
docs	Optional character vector of training docs (used if required by backend).

**Value**

The input model (invisibly).

---

bertopic_save	<i>Save a BERTopic model</i>
---------------	------------------------------

---

### Description

Save a fitted BERTopic model to disk. Depending on the serialization method, this may produce either a single file (e.g., \*.pkl / \*.pt / \*.safetensors) or a directory bundle. The function does not pre-create the target path; it only ensures the parent directory exists and lets BERTopic decide the layout.

### Usage

```
bertopic_save(
  model,
  path,
  serialization = c("pickle", "safetensors", "pt"),
  save_embedding_model = FALSE,
  overwrite = FALSE
)
```

### Arguments

model	A "bertopic_r" model.
path	Destination path (file or directory, as required by BERTopic).
serialization	One of "pickle", "safetensors", or "pt". Default "pickle".
save_embedding_model	Logical; whether to include the embedding model. Default FALSE.
overwrite	Logical; if TRUE and the target exists, it will be replaced.

### Value

Invisibly returns the normalized path.

---

bertopic_self_check	<i>Quick self-check for the BERTopic R interface</i>
---------------------	--

---

### Description

Runs a quick end-to-end smoke test:

- Report Python path/version.
- Verify that bertopic is importable and report its version.
- Minimal round trip: fit -> transform -> save -> load.

### Usage

```
bertopic_self_check()
```



**Value**

A named list with fields:

**python\_ok** Logical.

**bertopic\_ok** Logical.

**roundtrip\_ok** Logical.

**details** Character vector of diagnostic messages.

**Examples**

```
## Not run:
bertopic_self_check()

## End(Not run)
```

---

bertopic\_session\_info *Summarize Python/BERTopic session info*

---

**Description**

Summarize Python/BERTopic session info

**Usage**

```
bertopic_session_info()
```

**Value**

A named list containing paths, versions, and module availability:

**python** Path of the active Python.

**libpython** Path to libpython, if any.

**version** Python version string.

**numpy** Whether NumPy is available.

**numpy\_version** NumPy version string (if available).

**modules** A data.frame with availability for key modules.

**Examples**

```
## Not run:
bertopic_session_info()

## End(Not run)
```

---

bertopic\_set\_embedding\_model

*Replace or set the embedding model*


---

### Description

Set a new embedding model on a fitted BERTopic instance. This enables transform() after loading when the embedding model was not saved.

### Usage

```
bertopic_set_embedding_model(model, embedding_model)
```

### Arguments

model                    A "bertopic\_r" model.

embedding\_model        Either a character identifier (e.g., "all-MiniLM-L6-v2") or a Python embedding model object (e.g., a SentenceTransformer instance).

### Value

The input model (invisibly).

---

bertopic\_set\_topic\_labels

*Relabel topics*


---

### Description

Set custom labels for topics. Accepts a named character vector or a data.frame with columns topic and label.

### Usage

```
bertopic_set_topic_labels(model, labels)
```

### Arguments

model                    A "bertopic\_r" model.

labels                   A named character vector (names are topic ids) or a data.frame.

### Value

The input model (invisibly).

---

bertopic_topics	<i>Get topic info as a tibble</i>
-----------------	-----------------------------------

---

**Description**

Get topic info as a tibble

**Usage**

```
bertopic_topics(model)
```

**Arguments**

model	A "bertopic_r" object returned by <a href="#">bertopic_fit()</a> .
-------	--

**Value**

A tibble with topic-level information from Python `get_topic_info()`.

---

bertopic_topics_over_time	<i>Compute topics over time</i>
---------------------------	---------------------------------

---

**Description**

Wrapper for Python `BERTopic.topics_over_time()`. Returns a tibble and attaches the original Python dataframe in the `"_py"` attribute for use in visualization.

**Usage**

```
bertopic_topics_over_time(
  model,
  docs,
  timestamps,
  nr_bins = NULL,
  datetime_format = NULL
)
```

**Arguments**

model	A "bertopic_r" model.
docs	Character vector of documents.
timestamps	A vector of timestamps (Date, POSIXt, or character).
nr_bins	Optional number of temporal bins.
datetime_format	Optional strftime-style format if timestamps are strings.

**Value**

A tibble with topics-over-time data; attribute `"_py"` stores the original Python dataframe.

---

bertopic_topic_terms	<i>Get top terms for a topic</i>
----------------------	----------------------------------

---

**Description**

Get top terms for a topic

**Usage**

```
bertopic_topic_terms(model, topic_id, top_n = 10L)
```

**Arguments**

model	A "bertopic_r" model
topic_id	Integer topic id
top_n	Number of top terms to return

**Value**

A tibble with columns term and weight

---

bertopic_transform	<i>Transform new documents with a fitted BERTopic model</i>
--------------------	---

---

**Description**

Transform new documents with a fitted BERTopic model

**Usage**

```
bertopic_transform(model, new_text, embeddings = NULL)
```

**Arguments**

model	A "bertopic_r" model from <a href="#">bertopic_fit()</a> .
new_text	Character vector of new documents.
embeddings	Optional numeric matrix for new documents.

**Value**

A list with topics and probs for the new documents.

---

`bertopic_update_topics`*Update topic representations*

---

**Description**

Call Python `BERTopic.update_topics()` to recompute topic representations.

**Usage**

```
bertopic_update_topics(model, text)
```

**Arguments**

<code>model</code>	A "bertopic_r" model.
<code>text</code>	Character vector of training documents used in fit.

**Value**

The input model (invisibly), updated in place on the Python side.

---

`bertopic_visualize_barchart`*Visualize a topic barchart*

---

**Description**

Visualize a topic barchart

**Usage**

```
bertopic_visualize_barchart(model, topic_id = NULL, file = NULL)
```

**Arguments**

<code>model</code>	A "bertopic_r" model.
<code>topic_id</code>	Integer topic id. If NULL, a set of top topics is shown.
<code>file</code>	Optional HTML output path.

---

bertopic\_visualize\_distribution

*Visualize topic probability distribution*


---

## Description

Wrapper around Python `BERTopic.visualize_distribution()`. This function takes a single document's topic probability vector (e.g., one row from `probs`) and returns an interactive Plotly figure as HTML or writes it to disk.

## Usage

```
bertopic_visualize_distribution(
    model,
    probs,
    min_probability = NULL,
    custom_labels = FALSE,
    title = NULL,
    width = NULL,
    height = NULL,
    file = NULL
)
```

## Arguments

<code>model</code>	A "bertopic_r" model.
<code>probs</code>	Numeric vector of topic probabilities for a single document.
<code>min_probability</code>	Optional numeric scalar. If provided, only probabilities greater than this value are visualized (forwarded to <code>min_probability</code> in Python).
<code>custom_labels</code>	Logical or character scalar. If logical, whether to use custom topic labels as set via <code>set_topic_labels()</code> . If character, selects labels from other aspects (e.g., "Aspect1").
<code>title</code>	Optional character plot title.
<code>width, height</code>	Optional integer figure width/height in pixels.
<code>file</code>	Optional HTML output path. If NULL, an <code>htmltools::HTML</code> object is returned.

## Value

If `file` is NULL, an `htmltools::HTML` object. Otherwise, the normalized file path is returned invisibly.

---

bertopic\_visualize\_documents  
*Visualize embedded documents*

---

### Description

Visualize embedded documents

### Usage

```
bertopic_visualize_documents(model, docs = NULL, file = NULL)
```

### Arguments

model	A "bertopic_r" model.
docs	Optional character vector of documents to visualize.
file	Optional HTML output path.

---

bertopic\_visualize\_heatmap  
*Visualize topic similarity heatmap*

---

### Description

Visualize topic similarity heatmap

### Usage

```
bertopic_visualize_heatmap(model, file = NULL)
```

### Arguments

model	A "bertopic_r" model.
file	Optional HTML output path.

---

bertopic\_visualize\_hierarchical\_documents

*Visualize hierarchical documents and topics*


---

## Description

Wrapper around Python `BERTopic.visualize_hierarchical_documents()`. This function visualizes documents and their topics in 2D at different levels of a hierarchical topic structure.

## Usage

```
bertopic_visualize_hierarchical_documents(
    model,
    docs,
    hierarchical_topics,
    topics = NULL,
    embeddings = NULL,
    reduced_embeddings = NULL,
    sample = NULL,
    hide_annotations = FALSE,
    hide_document_hover = TRUE,
    nr_levels = 10L,
    level_scale = c("linear", "log"),
    custom_labels = FALSE,
    title = NULL,
    width = NULL,
    height = NULL,
    file = NULL
)
```

## Arguments

<code>model</code>	A "bertopic_r" model.
<code>docs</code>	Character vector of documents used in <code>fit</code> / <code>fit_transform</code> .
<code>hierarchical_topics</code>	A data frame or Python object as returned by <code>BERTopic.hierarchical_topics(docs, ...)</code> .
<code>topics</code>	Optional integer vector of topic IDs to visualize.
<code>embeddings</code>	Optional numeric matrix of document embeddings.
<code>reduced_embeddings</code>	Optional numeric matrix of 2D reduced embeddings.
<code>sample</code>	Optional numeric (0–1) or integer controlling subsampling of documents per topic (forwarded to Python).
<code>hide_annotations</code>	Logical; if TRUE, hide cluster labels in the plot.
<code>hide_document_hover</code>	Logical; if TRUE, hide document text on hover to speed up rendering.
<code>nr_levels</code>	Integer; number of hierarchy levels to display.



level_scale	Character, either "linear" or "log", controlling how hierarchy distances are scaled across levels.
custom_labels	Logical or character scalar controlling label behavior (forwarded to Python).
title	Optional character plot title.
width, height	Optional integer figure width/height in pixels.
file	Optional HTML output path. If NULL, an <code>htmltools::HTML</code> object is returned.

**Value**

If file is NULL, an `htmltools::HTML` object. Otherwise, the normalized file path is returned invisibly.

---

bertopic\_visualize\_hierarchy  
*Visualize hierarchical clustering of topics*

---

**Description**

Visualize hierarchical clustering of topics

**Usage**

```
bertopic_visualize_hierarchy(model, file = NULL)
```

**Arguments**

model	A "bertopic_r" model.
file	Optional HTML output path.

---

bertopic\_visualize\_term\_rank  
*Visualize term rank evolution*

---

**Description**

Visualize term rank evolution

**Usage**

```
bertopic_visualize_term_rank(model, file = NULL)
```

**Arguments**

model	A "bertopic_r" model.
file	Optional HTML output path.

---

```
bertopic_visualize_topics
```

*Visualize topic map*

---

### Description

Visualize topic map

### Usage

```
bertopic_visualize_topics(model, file = NULL)
```

### Arguments

model	A "bertopic_r" model.
file	Optional HTML output path. If NULL, returns <code>htmltools::HTML</code> .

---

```
bertopic_visualize_topics_over_time
```

*Visualize topics over time*

---

### Description

Visualize topics over time

### Usage

```
bertopic_visualize_topics_over_time(
  model,
  topics_over_time,
  top_n = 10L,
  file = NULL
)
```

### Arguments

model	A "bertopic_r" model.
topics_over_time	A tibble returned by <a href="#">bertopic_topics_over_time()</a> , or a Python dataframe compatible with <code>visualize_topics_over_time()</code> .
top_n	Number of topics to display.
file	Optional HTML output path.

---

```
bertopic_visualize_topics_per_class
    Visualize topics per class
```

---

## Description

Wrapper around Python `BERTopic.visualize_topics_per_class()`. This visualizes how topics are distributed across a set of classes, using the output of Python `topics_per_class(docs, classes)`.

## Usage

```
bertopic_visualize_topics_per_class(
    model,
    topics_per_class,
    top_n_topics = 10L,
    topics = NULL,
    normalize_frequency = FALSE,
    custom_labels = FALSE,
    title = NULL,
    width = NULL,
    height = NULL,
    file = NULL
)
```

## Arguments

<code>model</code>	A "bertopic_r" model.
<code>topics_per_class</code>	A data frame or Python object as returned by <code>BERTopic.topics_per_class(docs, classes)</code> .
<code>top_n_topics</code>	Integer; number of most frequent topics to display.
<code>topics</code>	Optional integer vector of topic IDs to include.
<code>normalize_frequency</code>	Logical; whether to normalize each topic's frequency within classes.
<code>custom_labels</code>	Logical or character scalar controlling label behavior (forwarded to Python).
<code>title</code>	Optional character plot title.
<code>width, height</code>	Optional integer figure width/height in pixels.
<code>file</code>	Optional HTML output path. If NULL, an <code>htmltools::HTML</code> object is returned.

## Value

If `file` is NULL, an `htmltools::HTML` object. Otherwise, the normalized file path is returned invisibly.

---

coef.bertopic_r	<i>Coefficients (top terms) for BERTopic</i>
-----------------	--

---

**Description**

Coefficients (top terms) for BERTopic

**Usage**

```
## S3 method for class 'bertopic_r'  
coef(object, top_n = 10L, ...)
```

**Arguments**

object	A "bertopic_r" model.
top_n	Number of terms per topic.
...	Unused.

**Value**

A data.frame with columns topic, term, weight.

---

fortify.bertopic_r	<i>Fortify method for ggplot2</i>
--------------------	-----------------------------------

---

**Description**

Fortify method for ggplot2

**Usage**

```
fortify.bertopic_r(model, data, ...)
```

**Arguments**

model	A "bertopic_r" model.
data	Ignored.
...	Unused.

**Value**

A data.frame of document-topic assignments.

---

install_py_deps	<i>Install Python dependencies for BERTopic (auto route)</i>
-----------------	--

---

**Description**

Tries Conda first (recommended). If Conda is unavailable, falls back to virtualenv. On success, prints which route was used.

**Usage**

```
install_py_deps(
    envname = "r-bertopic",
    python_version = "3.10",
    python = NULL,
    reinstall = FALSE,
    validate = TRUE,
    verbose = TRUE
)
```

**Arguments**

envname	Character. Environment name (both routes). Default "r-bertopic".
python_version	Character. Python version for Conda route, e.g. "3.10".
python	Optional path to python for virtualenv route.
reinstall	Logical. Recreate the environment if it exists (route-specific).
validate	Logical. Attempt to validate imports if reticulate is not already initialized to another Python.
verbose	Logical. Print progress.

**Value**

Invisibly, the path to the selected Python interpreter.

---

install_py_deps_conda	<i>Install Python dependencies for BERTopic (Conda route)</i>
-----------------------	---

---

**Description**

Creates (or reuses) a Conda environment with a pinned Python toolchain, installs the scientific stack + PyTorch (CPU) + sentence-transformers, then installs bertopic==0.16.0 via pip. Optionally validates imports.

**Usage**

```
install_py_deps_conda(
    envname = "r-bertopic",
    python_version = "3.10",
    reinstall = FALSE,
    validate = TRUE,
    verbose = TRUE
)
```

**Arguments**

envname	Character. Conda environment name. Default "r-bertopic".
python_version	Character. Python version to use, e.g. "3.10".
reinstall	Logical. If TRUE, delete any existing env and recreate.
validate	Logical. If TRUE, bind and validate imports (will skip if reticulate is already initialized to another Python).
verbose	Logical. Print progress messages.

**Value**

Invisibly returns the path to the Python executable inside the env.

**Examples**

```
## Not run:
install_py_deps_conda(envname = "r-bertopic", python_version = "3.10")

## End(Not run)
```

---

install\_py\_deps\_venv    *Install Python dependencies for BERTopic (virtualenv route)*

---

**Description**

Creates (or reuses) a virtualenv and installs bertopic==0.16.0 plus required dependencies via pip. Optionally validates imports.

**Usage**

```
install_py_deps_venv(
  envname = "r-bertopic",
  python = NULL,
  reinstall = FALSE,
  validate = TRUE,
  verbose = TRUE
)
```

**Arguments**

envname	Character. Virtualenv name. Default "r-bertopic".
python	Character. Path to a Python executable to create the venv with. If NULL, tries to find python / python3 on PATH.
reinstall	Logical. If TRUE, delete existing venv and recreate.
validate	Logical. If TRUE, bind and validate imports (will skip if reticulate is already initialized to another Python).
verbose	Logical. Print progress messages.

**Value**

Invisibly returns the path to the Python executable inside the venv.

**Examples**

```
## Not run:
install_py_deps_venv(envname = "r-bertopic")

## End(Not run)
```

---

predict.bertopic_r	<i>Predict method for BERTopic models</i>
--------------------	---

---

**Description**

Predict method for BERTopic models

**Usage**

```
## S3 method for class 'bertopic_r'
predict(
  object,
  newdata,
  type = c("both", "class", "prob"),
  embeddings = NULL,
  ...
)
```

**Arguments**

object	A "bertopic_r" model.
newdata	Character vector of new documents.
type	One of "class", "prob", or "both".
embeddings	Optional numeric matrix of embeddings.
...	Reserved for future arguments.

**Value**

Depending on type, an integer vector, a matrix/data frame, or a list.

---

print.bertopic_r	<i>Print method for bertopic_r</i>
------------------	------------------------------------

---

**Description**

Print method for bertopic\_r

**Usage**

```
## S3 method for class 'bertopic_r'
print(x, ...)
```

**Arguments**

x	A "bertopic_r" object.
...	Unused.

---

set_bertopic_seed	<i>Set random seed for R and Python backends</i>
-------------------	--

---

**Description**

Set random seed for R and Python backends

**Usage**

```
set_bertopic_seed(seed)
```

**Arguments**

seed	Integer seed
------	--------------

---

sms_spam	<i>SMS Spam Collection (UCI) - subset for examples</i>
----------	--

---

**Description**

A cleaned subset of the UCI SMS Spam Collection, suitable for quick examples and tests in this package. Each row is an SMS message labeled as "ham" or "spam".

**Usage**

```
sms_spam
```

**Format**

A data frame with two columns:

**label** Character, either "ham" or "spam".

**text** Character, the SMS message content (UTF-8).

**Note**

This dataset is included for educational/demo purposes. If you use it in publications, please cite the original authors and the UCI repository page.

**Source**

UCI Machine Learning Repository: SMS Spam Collection. Dataset page: <https://archive.ics.uci.edu/dataset/228/sms+spam+collection> Original citation: Almeida, T.A., Hidalgo, J.M.G., & Yamakami, A. (2011). Contributions to the Study of SMS Spam Filtering: New Collection and Results.



**Examples**

```
data(sms_spam)
head(sms_spam)
```

---

summary.bertopic_r	<i>Summary for BERTopic models</i>
--------------------	------------------------------------

---

**Description**

Summary for BERTopic models

**Usage**

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

**Arguments**

object	A "bertopic_r" model.
...	Unused.

**Value**

Invisibly returns a named list of summary fields.

---

use_bertopic	<i>Bind current R session to the BERTopic environment (auto route)</i>
--------------	--

---

**Description**

If a Conda env with the given name exists, prefer Conda; otherwise try a virtualenv with the same name. Stops if neither exists.

**Usage**

```
use_bertopic(envname = "r-bertopic")
```

**Arguments**

envname	Character. Environment name. Default "r-bertopic".
---------	--

**Value**

Invisibly, the Python executable path.

---

use\_bertopic\_condaenv *Bind current R session to a BERTopic Conda environment*

---

### Description

Sets RETICULATE\_PYTHON to the environment's Python and initializes **reticulate**. If **reticulate** is already initialized to a different Python, this stops with an informative error.

### Usage

```
use_bertopic_condaenv(envname = "r-bertopic", required = TRUE)
```

### Arguments

envname	Character. Conda env name (default "r-bertopic").
required	Logical. Kept for API symmetry; unused.

### Value

Invisibly returns the Python executable path in the env.

### Examples

```
## Not run:
use_bertopic_condaenv("r-bertopic")

## End(Not run)
```

---

use\_bertopic\_virtualenv  
*Bind current R session to a BERTopic virtualenv*

---

### Description

Sets RETICULATE\_PYTHON to the Python inside the given virtualenv and initializes **reticulate**. If **reticulate** is already initialized to a different Python, this stops with an informative error.

### Usage

```
use_bertopic_virtualenv(envname = "r-bertopic", required = TRUE)
```

### Arguments

envname	Character. Virtualenv name (default "r-bertopic").
required	Logical. Kept for API symmetry; unused.

### Value

Invisibly returns the Python executable path in the venv.

**Examples**

```
## Not run:  
use_bertopic_virtualenv("r-bertopic")  
  
## End(Not run)
```

# Index

**\* datasets**  
    sms\_spam, [24](#)

as.data.frame.bertopic\_r, [2](#)

bertopic\_as\_document\_topic\_matrix, [3](#)  
bertopic\_available, [3](#)  
bertopic\_find\_topics, [4](#)  
bertopic\_fit, [4](#)  
bertopic\_fit(), [11](#), [12](#)  
bertopic\_get\_document\_info, [5](#)  
bertopic\_get\_representative\_docs, [6](#)  
bertopic\_has\_embedding\_model, [6](#)  
bertopic\_load, [7](#)  
bertopic\_reduce\_topics, [7](#)  
bertopic\_save, [8](#)  
bertopic\_save(), [7](#)  
bertopic\_self\_check, [8](#)  
bertopic\_session\_info, [9](#)  
bertopic\_set\_embedding\_model, [10](#)  
bertopic\_set\_topic\_labels, [10](#)  
bertopic\_topic\_terms, [12](#)  
bertopic\_topics, [11](#)  
bertopic\_topics(), [3](#)  
bertopic\_topics\_over\_time, [11](#)  
bertopic\_topics\_over\_time(), [18](#)  
bertopic\_transform, [12](#)  
bertopic\_update\_topics, [13](#)  
bertopic\_visualize\_barchart, [13](#)  
bertopic\_visualize\_distribution, [14](#)  
bertopic\_visualize\_documents, [15](#)  
bertopic\_visualize\_heatmap, [15](#)  
bertopic\_visualize\_hierarchical\_documents,  
    [16](#)  
bertopic\_visualize\_hierarchy, [17](#)  
bertopic\_visualize\_term\_rank, [17](#)  
bertopic\_visualize\_topics, [18](#)  
bertopic\_visualize\_topics\_over\_time,  
    [18](#)  
bertopic\_visualize\_topics\_per\_class,  
    [19](#)

coef.bertopic\_r, [20](#)

fortify.bertopic\_r, [20](#)

install\_py\_deps, [21](#)  
install\_py\_deps\_conda, [21](#)  
install\_py\_deps\_venv, [22](#)

predict.bertopic\_r, [23](#)  
print.bertopic\_r, [23](#)

set\_bertopic\_seed, [24](#)  
sms\_spam, [24](#)  
summary.bertopic\_r, [25](#)

use\_bertopic, [25](#)  
use\_bertopic\_condaenv, [26](#)  
use\_bertopic\_virtualenv, [26](#)