topicpy

Release 0.2.0

May 19, 2022

Contents

CHAPTER 1

Installation

Install topicpy by running:

```
pip install topicpy
```

This package consists of multiple modules:

- converter: handels gene name conversions
- geneontology: uses GSEA to perform gene ontologies
- gtex: handle GTEx data
- hsbmpy: handle ouput of hierarchical stochastic block model
- hypergeom: perform hypergeometric tests
- lda: Run sklearn LatenDirichletAllocation with some more parameters
- tableanalyser: study distributions of a RNA-Seq data
- TCGA_files: handle TCGA metadata

CHAPTER 2

Resources

- Online documentation: https://topicpy.readthedocs.io/en/latest/
- Paper: in preparation

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

3.1 converter

topicpy.converter.**converter**
alias of *topicpy.converter.converter*

topicpy.converter.**convert_list_to_sybmols**(*ensgs: list*) → list

it converts a list of ENSG to gene names

Parameters **ensgs** – list of ENSG

convert_list_to_sybmols(["ENSG00000159763"])

topicpy.converter.**convert_list_to_ensg**(*symbols: list*) → list

it converts a list of gene names to ENSG identifiers

Parameters **symbols** – list of gene symbols

convert_list_to_sybmols(["PIP"])

3.2 geneontology

topicpy.geneontology.**geneontology**
alias of *topicpy.geneontology.geneontology*

topicpy.geneontology.**topic_analysis**(*directory*, *l*, *algorithm='topsbm'*, *background=None*, *verbose=True*, *save_Pvalues=True*)

Parameters

- **directory** – where to find files
- **l** – level of the analisys
- **algorithm** – name of folder and files containing topics table (e.g. path/to/topsbm/topsbm_level_3_topics.csv)
- **background** – List of background genes
- **verbose** – verbosity
- **save_Pvalues** – save data for P-values plot

topicpy.geneontology.**save_plot_Pvalues**(*df_topics*, *l*, *directory*, *algorithm*)

Parameters

- **df_topics** – Topics DataFrame
- **l** – level of the analysis
- **directory** –

topicpy.geneontology.**get_ontology_df**(*topic*, *cutoff=0.05*, *threshhold=0.5*, *gene_sets=['GO_Molecular_Function_2018', 'GO_Biological_Process_2018', 'GO_Cellular_Component_2018', 'Human_Phenotype_Ontology', 'GTEx_Tissue_Sample_Gene_Expression_Profiles_up', 'GTEx_Tissue_Sample_Gene_Expression_Profiles_down', 'Tissue_Protein_Expression_from_Human_Proteome_Map', 'KEGG_2019_Human', 'NCI-60_Cancer_Cell_Lines']*, *background=None*) → pandas.core.frame.DataFrame

Parameters

- **topic** – list of genes
- **background** – enrichment test background
- **cutoff** – Enrichments cutoff
- **threshhold** – threshold on Adjusted P-value

Returns DataFrame with terms and P-vals

3.3 gtex

topicpy.gtex.**gtex**
alias of *topicpy.gtex.gtex*

3.4 hsbmpy

class topicpy.hsbmpy.**painter**

topicpy.hsbmpy.**hsbmpy**
alias of *topicpy.hsbmpy.hsbmpy*

topicpy.hsbmpy.**clusteranalysis**(*directory*, *labels*, *algorithm='topsbm'*) → None
Perform analyses of an algorithm output

Parameters

- **directory** – where to search the data
- **labels** – ground truth label to search. This should be in a file called directory/files.dat
- **algorithm** – name of the folder in which data are stored

3.5 hypergeom

topicpy.hypergeom.**hypergeom**

topicpy.hypergeom.**parameters_for_hypergeometric**(*list_1: pandas.core.series.Series*, *list_2: pandas.core.series.Series*) -> (<class 'float'>, <class 'float'>, <class 'float'>, <class 'float'>, (<class 'list'>, <class 'list'>))

Parameters

- **list_1** – series
- **list_2** – series

lists of elements

Returns

- x num of successes
- M population size
- k successes in population
- N sample size
- (list_1, list_2) tuple of original lists

Example:

l1 = pd.Series(index=["ENSG00000000123", "ENSG00000000456", "ENSG00000000789", "ENSG00000000XXX"], data=["c1", "c1", "c1", "c2"], dtype=str) l2 = pd.Series(index=["ENSG00000000123", "ENSG00000000456", "ENSG00000000789"], data=["c1", "c1", "c1"], dtype=str) x, M, k, N, _ = parameters_for_hypergeometric(l1, l2)

```
>>> x
    c1
c1   3
c2   0
>>> M
3
>>> k
{'c1': 3}
>>> N
{'c1': 3, 'c2': 1}
```

topicpy.hypergeom.**build_map**(*num_successes*, *population_size*, *pop_successes*, *sample_sizes*, *lists*, *last_name=None*)

topicpy.hypergeom.**plot_map**(*df_cmap*, *first_name='topsbm'*, *last_name='lda'*, **args*, ***kwargs*)

3.6 lda

class topicpy.lda.**lda**(*learning_method='online'*, *max_doc_update_iter=5*, *max_iter=5*, *topic_word_prior=1*, *doc_topic_prior=1*, *random_state=42*, ***params*)

_approx_bound(*X*, *doc_topic_distr*, *sub_sampling*)

Estimate the variational bound.

Estimate the variational bound over "all documents" using only the documents passed in as X. Since log-likelihood of each word cannot be computed directly, we use this bound to estimate it.

Parameters

- **X** (*{array-like, sparse matrix} of shape (n_samples, n_features)*) – Document word matrix.
- **doc_topic_distr** (*ndarray of shape (n_samples, n_components)*) – Document topic distribution. In the literature, this is called gamma.
- **sub_sampling** (*bool, default=False*) – Compensate for subsampling of documents. It is used in calculate bound in online learning.

Returns score

Return type float

_check_feature_names(*X*, ***, *reset*)

Set or check the *feature_names_in_* attribute.

New in version 1.0.

Parameters

- **X** (*{ndarray, dataframe} of shape (n_samples, n_features)*) – The input samples.
- **reset** (*bool*) – Whether to reset the *feature_names_in_* attribute. If False, the input will be checked for consistency with feature names of data provided when reset was last True. .. note:

```
It is recommended to call `reset=True` in `fit` and in the first
call to `partial_fit`. All other methods that validate `X`
should set `reset=False`.
```

_check_n_features(*X*, *reset*)

Set the *n_features_in_* attribute, or check against it.

Parameters

- **X** (*{ndarray, sparse matrix} of shape (n_samples, n_features)*) – The input samples.
- **reset** (*bool*) – If True, the *n_features_in_* attribute is set to *X.shape[1]*. If False and the attribute exists, then check that it is equal to *X.shape[1]*. If False and the attribute does *not* exist, then the check is skipped. .. note:

```
It is recommended to call reset=True in `fit` and in the first
call to `partial_fit`. All other methods that validate `X`
should set `reset=False`.
```

_check_non_neg_array(*X*, *reset_n_features*, *whom*)
check X format

check X format and make sure no negative value in X.

Parameters **X** (*array-like or sparse matrix*) –

_check_params()
Check model parameters.

_e_step(*X*, *cal_sstats*, *random_init*, *parallel=None*)
E-step in EM update.

Parameters

- **X** (*{array-like, sparse matrix} of shape (n_samples, n_features)*) – Document word matrix.
- **cal_sstats** (*bool*) – Parameter that indicate whether to calculate sufficient statistics or not. Set `cal_sstats` to True when we need to run M-step.
- **random_init** (*bool*) – Parameter that indicate whether to initialize document topic distribution randomly in the E-step. Set it to True in training steps.
- **parallel** (*joblib.Parallel, default=None*) – Pre-initialized instance of joblib.Parallel.

Returns *doc_topic_distr* is unnormalized topic distribution for each document. In the literature, this is called *gamma*. *suff_stats* is expected sufficient statistics for the M-step. When *cal_sstats* == *False*, it will be None.

Return type (doc_topic_distr, suff_stats)

_em_step(*X*, *total_samples*, *batch_update*, *parallel=None*)
EM update for 1 iteration.

update *_component* by batch VB or online VB.

Parameters

- **X** (*{array-like, sparse matrix} of shape (n_samples, n_features)*) – Document word matrix.
- **total_samples** (*int*) – Total number of documents. It is only used when batch_update is *False*.
- **batch_update** (*bool*) – Parameter that controls updating method. *True* for batch learning, *False* for online learning.
- **parallel** (*joblib.Parallel, default=None*) – Pre-initialized instance of joblib.Parallel

Returns **doc_topic_distr** – Unnormalized document topic distribution.

Return type ndarray of shape (n_samples, n_components)

classmethod _get_param_names()
Get parameter names for the estimator

_init_latent_vars(*n_features*)
Initialize latent variables.

_perplexity_precomp_distr(*X*, *doc_topic_distr=None*, *sub_sampling=False*)
Calculate approximate perplexity for data X with ability to accept precomputed doc_topic_distr

Perplexity is defined as exp(-1. * log-likelihood per word)

Parameters

- **X** (*{array-like, sparse matrix} of shape (n_samples, n_features)*) – Document word matrix.
- **doc_topic_distr** (*ndarray of shape (n_samples, n_components), default=None*) – Document topic distribution. If it is None, it will be generated by applying transform on X.

Returns **score** – Perplexity score.

Return type float

_repr_html_
HTML representation of estimator.

This is redundant with the logic of *_repr_mimebundle_*. The latter should be favorted in the long term, *_repr_html_* is only implemented for consumers who do not interpret *_repr_mimbundle_*.

_repr_html_inner()
This function is returned by the @property *_repr_html_* to make *hasattr(estimator, "_repr_html_") return 'True* or *False* depending on *get_config()["display"]*.

_repr_mimebundle_(***kwargs*)
Mime bundle used by jupyter kernels to display estimator

_unnormalized_transform(*X*)
Transform data X according to fitted model.

Parameters **X** (*{array-like, sparse matrix} of shape (n_samples, n_features)*) – Document word matrix.

Returns **doc_topic_distr** – Document topic distribution for X.

Return type ndarray of shape (n_samples, n_components)

_validate_data(*X='no_validation'*, *y='no_validation'*, *reset=True*, *validate_separately=False*, ***check_params*)
Validate input data and set or check the *n_features_in_* attribute.

Parameters

- **X** (*{array-like, sparse matrix, dataframe} of shape (n_samples, n_features), default='no validation'*) – The input samples. If *'no_validation'*, no validation is performed on *X*. This is useful for meta-estimator which can delegate input validation to their underlying estimator(s). In that case *y* must be passed and the only accepted *check_params* are *multi_output* and *y_numeric*.
- **y** (*array-like of shape (n_samples,), default='no_validation'*) – The targets.
  - If *None*, *check_array* is called on *X*. If the estimator's requires_y tag is True, then an error will be raised.
  - If *'no_validation'*, *check_array* is called on *X* and the estimator's requires_y tag is ignored. This is a default placeholder and is never meant to be explicitly set. In that case *X* must be passed.
  - Otherwise, only *y* with *_check_y* or both *X* and *y* are checked with either *check_array* or *check_X_y* depending on *validate_separately*.

- **reset** (*bool, default=True*) – Whether to reset the *n_features_in_* attribute. If False, the input will be checked for consistency with data provided when reset was last True. .. note:

```
It is recommended to call reset=True in `fit` and in the first
call to `partial_fit`. All other methods that validate `X`
should set `reset=False`.
```

- **validate_separately** (*False or tuple of dicts, default=False*) – Only used if y is not None. If False, call validate_X_y(). Else, it must be a tuple of kwargs to be used for calling check_array() on X and y respectively.
- ****check_params** (*kwargs*) – Parameters passed to `sklearn.utils.check_array()` or `sklearn.utils.check_X_y()`. Ignored if validate_separately is not False.

Returns **out** – The validated input. A tuple is returned if both *X* and *y* are validated.

Return type {ndarray, sparse matrix} or tuple of these

fit (*X*, *y=None*)

Learn model for the data X with variational Bayes method.

When *learning_method* is 'online', use mini-batch update. Otherwise, use batch update.

Parameters

- **X** (*{array-like, sparse matrix} of shape (n_samples, n_features)*) – Document word matrix.
- **y** (*Ignored*) – Not used, present here for API consistency by convention.

Returns Fitted estimator.

Return type self

fit_transform (*X*, *y=None*, ***fit_params*)

Fit to data, then transform it.

Fits transformer to *X* and *y* with optional parameters *fit_params* and returns a transformed version of *X*.

Parameters

- **X** (*array-like of shape (n_samples, n_features)*) – Input samples.
- **y** (*array-like of shape (n_samples,) or (n_samples, n_outputs), default=None*) – Target values (None for unsupervised transformations).
- ****fit_params** (*dict*) – Additional fit parameters.

Returns **X_new** – Transformed array.

Return type ndarray array of shape (n_samples, n_features_new)

full_analysis (*directory*, *xl*, *tl=None*, *label='primary_site'*, *logarithmise=False*, *round_data=False*, **args*, ***kwargs*) → None

Parameters

- **df** –
- **directory** –
- **xl** –
- **tl** –

- **kwargs** – argouments to LatentDirichletAllocation().fit_transform

get_params(*deep=True*)
Get parameters for this estimator.

Parameters **deep** (*bool, default=True*) – If True, will return the parameters for this estimator and contained subobjects that are estimators.

Returns **params** – Parameter names mapped to their values.

Return type dict

partial_fit(*X*, *y=None*)
Online VB with Mini-Batch update.

Parameters

- **X** (*{array-like, sparse matrix} of shape (n_samples, n_features)*) – Document word matrix.
- **y** (*Ignored*) – Not used, present here for API consistency by convention.

Returns Partially fitted estimator.

Return type self

perplexity(*X*, *sub_sampling=False*)
Calculate approximate perplexity for data X.

Perplexity is defined as exp(-1. * log-likelihood per word)

Changed in version 0.19: *doc_topic_distr* argument has been deprecated and is ignored because user no longer has access to unnormalized distribution

Parameters

- **X** (*{array-like, sparse matrix} of shape (n_samples, n_features)*) – Document word matrix.
- **sub_sampling** (*bool*) – Do sub-sampling or not.

Returns **score** – Perplexity score.

Return type float

score(*X*, *y=None*)
Calculate approximate log-likelihood as score.

Parameters

- **X** (*{array-like, sparse matrix} of shape (n_samples, n_features)*) – Document word matrix.
- **y** (*Ignored*) – Not used, present here for API consistency by convention.

Returns **score** – Use approximate bound as score.

Return type float

set_params(***params*)
Set the parameters of this estimator.

The method works on simple estimators as well as on nested objects (such as `Pipeline`). The latter have parameters of the form `<component>__<parameter>` so that it's possible to update each component of a nested object.

Parameters ****params** (*dict*) – Estimator parameters.

Returns **self** – Estimator instance.

Return type estimator instance

transform(*X*)

Transform data X according to the fitted model.

Changed in version 0.18: *doc_topic_distr* is now normalized

Parameters **X** (*{array-like, sparse matrix} of shape (n_samples, n_features)*) – Document word matrix.

Returns **doc_topic_distr** – Document topic distribution for X.

Return type ndarray of shape (n_samples, n_components)

3.7 tableanalyser

topicpy.tableanalyser.**tableanalyser**

alias of *topicpy.tableanalyser.tableanalyser*

topicpy.tableanalyser.**scalinglawsandoverexpressed**(*working_dir*, *normalisation_str='counts'*, *method='sampling'*, *how_many_sigmas=3*, *distance=10*)

3.8 tacos_plot

topicpy.tacos_plot.**tacos_plot**

alias of *topicpy.tacos_plot.tacos_plot*

topicpy.tacos_plot.**scatterdense**(*x*, *y*, *ax=None*, *nbins=80*, *colorbar=False*, *c_title='density'*, ***kwargs*) → None

Plot a scatter plot with density

Parameters

- **x** – list of first variable
- **y** – list of second variable
- **ax** – matplotlib.pyplot.Axes add here or create a new one
- **nbins** – number of bins
- **colorbar** – plot colorbar
- **c_title** – color bar title
- ****kargs** – arguments to be passed to matplotlib.pyplot.scatter

3.9 TCGA_files

topicpy.TCGA_files.**TCGA_files**

alias of *topicpy.TCGA_files.TCGA_files*

topicpy.TCGA_files.**get_tcga_tissue**(*sample*)
Get primary_site of tcga sample

Parameters **sample** (*str*) – sample id

topicpy.TCGA_files.**queryFiles**(*files*)
Get infor for a list of files

Parameters **files** (*list*) – list of TCGA-ids

topicpy.TCGA_files.**queryFile**(*idFile*)
Get information for file

Parameters **idFile** (*str*) – file TCGA-id

Python Module Index

t

Index

Symbols

B

C

F

G

H

L

P

Q

S

T