sklearn-rri Documentation

Release 0.1.0

Michal Ciesielczyk

Sep 30, 2017

Contents:

CHAPTER 1

sklearn_rri package

Submodules

sklearn_rri.rri module

Reflective Random Indexing (RRI) algorithm implementation.

class sklearn_rri.rri.**ReflectiveRandomIndexing**(*n_components=None*, *n_iter=3*, *seed='auto'*, *norm=True*, *dense_components=False*, *random_state=None*)

Bases: sklearn.base.BaseEstimator, sklearn.base.TransformerMixin

Dimensionality reduction using Reflective Random Indexing (RRI).

This transformer performs dimensionality reduction by means of RRI. It can work both with numpy.ndarray and scipy.sparse matrices efficiently.

Parameters

- **n_components** (*int, default = None*) – Desired dimensionality of output data. if n_components is not set all components are kept:

```
n_components == min(n_samples, n_features)
```

- **n_iter** (*int, default = 3*) – Number of iterations (aka reflections) to be performed.
- **seed** (*int, default = 'auto'*) – Random indexing seed value (number of non-zero values in every index vector). If seed = 'auto', the value is set to sqrt(n_features).
- **norm** (*bool, default = True*) – Indicates whether the context vectors should be normalized after every reflection step.
- **dense_components** (*bool, default = False*) – Indicates whether the estimated components matrix should be sparse (by default) or dense.

- **random_state** (*int or RandomState instance, default = None*) – If int, random_state is the seed used by the random number generator; If RandomState instance, random_state is the random number generator; If None, the random number generator is the RandomState instance used by *np.random*.

components_
array, shape (n_components, n_features) – Estimated components.

References

Reflective Random Indexing and Indirect Inference: A Scalable Method for Discovery of Implicit Connections, Trevor Cohen, Roger Schaneveldt, and Dominic Widdows, 2010. https://www.ncbi.nlm.nih.gov/pubmed/19761870

Examples

```
>>> from sklearn_rri import ReflectiveRandomIndexing
>>> from sklearn.random_projection import sparse_random_matrix
>>> X = sparse_random_matrix(100, 100, density=0.01, random_state=42)
>>> rri = ReflectiveRandomIndexing(50, random_state=42)
>>> rri.fit(X)
ReflectiveRandomIndexing(n_components=50, n_iter=3, norm=True,
         random_state=42, seed='auto')
>>> rri.transform(X)
<100x50 sparse matrix of type '<class 'numpy.float64'>'
        with 1154 stored elements in Compressed Sparse Row format>
```

fit(*X*, *y=None*)
Fit RRI model on training data X.

Parameters

- **X** (*{array-like, sparse matrix}, shape (n_samples, n_features)*) – Training data, where n_samples in the number of samples and n_features is the number of features.
- **y** (*(ignored)*) –

Returns **self** – Returns the transformer object.

Return type object

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** (*numpy array of shape [n_samples, n_features]*) – Training set.
- **y** (*numpy array of shape [n_samples]*) – Target values.

Returns **X_new** – Transformed array.

Return type numpy array of shape [n_samples, n_features_new]

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

Parameters **deep** (*boolean, optional*) – 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 mapping of string to any

inverse_transform(*X*)
Transform X back to its original space.

Returns an array X_original whose transform would be X.

Parameters **X** (*array-like, shape (n_samples, n_components)*) – New data, where n_samples in the number of samples and n_features is the number of features.

Returns **X_original** – Note that this is always a dense array.

Return type array, shape (n_samples, n_features)

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

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

Returns

Return type self

transform(*X*)
Perform dimensionality reduction on X.

Parameters **X** (*{array-like, sparse matrix}, shape (n_samples, n_features)*) – New data, where n_samples in the number of samples and n_features is the number of features.

Returns **X_new** – Reduced version of X. This will always be a dense array.

Return type array, shape (n_samples, n_components)

Module contents

scikit-learn compatible classifier based on Reflective Random Indexing.

Installation

Latest from the source:

```
git clone https://github.com/cmick/sklearn-rri.git
cd sklearn-rri
python setup.py install
```

Using PyPI:

```
pip install sklearn-rri
```

Usage example

```
>>> from sklearn_rri import ReflectiveRandomIndexing
>>> from sklearn.random_projection import sparse_random_matrix
>>> X = sparse_random_matrix(100, 100, density=0.01, random_state=42)
>>> rri = ReflectiveRandomIndexing(50, random_state=42)
>>> rri.fit(X)
ReflectiveRandomIndexing(n_components=50, n_iter=3, norm=True,
        random_state=42, seed='auto')
>>> rri.transform(X)
<100x50 sparse matrix of type '<class 'numpy.float64'>'
        with 1154 stored elements in Compressed Sparse Row format>
```

References

Reflective Random Indexing and Indirect Inference: A Scalable Method for Discovery of Implicit Connections, Trevor Cohen, Roger Schaneveldt, and Dominic Widdows, 2010. https://www.ncbi.nlm.nih.gov/pubmed/19761870

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

S

Index