Python：Basic Slicing and Indexing

<http://docs.scipy.org/doc/numpy/reference/arrays.indexing.html>

LDA explanation

<http://pythonhosted.org/lda/api.html>

## lda.lda[¶](http://pythonhosted.org/lda/api.html" \l "module-lda.lda" \o "Permalink to this headline)

Latent Dirichlet allocation using collapsed Gibbs sampling

**class lda.lda.LDA(n\_topics, n\_iter=2000, alpha=0.1, eta=0.01, random\_state=None, refresh=10)**[¶](http://pythonhosted.org/lda/api.html#lda.lda.LDA)

Latent Dirichlet allocation using collapsed Gibbs sampling

|  |  |
| --- | --- |
| **Parameters:** | **n\_topics** : int  Number of topics  **n\_iter** : int, default 2000  Number of sampling iterations  **alpha** : float, default 0.1  Dirichlet parameter for distribution over topics  **eta** : float, default 0.01  Dirichlet parameter for distribution over words  **random\_state** : int or RandomState, optional  The generator used for the initial topics. |

**Examples**

>>> **import** numpy

>>> X **=** numpy**.**array([[1,1], [2, 1], [3, 1], [4, 1], [5, 8], [6, 1]])

>>> **import** lda

>>> model **=** lda**.**LDA(n\_topics**=**2, random\_state**=**0, n\_iter**=**100)

>>> model**.**fit(X)

LDA(alpha=...

>>> model**.**components\_

array([[ 0.85714286, 0.14285714],

[ 0.45 , 0.55 ]])

>>> model**.**loglikelihood()

-40.395...

**Attributes**

|  |  |
| --- | --- |
| *components\_* | (array, shape = [n\_topics, n\_features]) Point estimate of the topic-word distributions (Phi in literature) |
| *topic\_word\_* : | Alias for *components\_* |
| *nzw\_* | (array, shape = [n\_topics, n\_features]) Matrix of counts recording topic-word assignments in final iteration. |
| *ndz\_* | (array, shape = [n\_samples, n\_features]) Matrix of counts recording document-topic assignments in final iteration. |
| *doc\_topic\_* | (array, shape = [n\_samples, n\_features]) Point estimate of the document-topic distributions (Theta in literature) |
| *nz\_* | (array, shape = [n\_topics]) Array of topic assignment counts in final iteration. |

**fit(X, y=None)**[¶](http://pythonhosted.org/lda/api.html#lda.lda.LDA.fit)

Fit the model with X.

|  |  |
| --- | --- |
| **Parameters:** | **X: array-like, shape (n\_samples, n\_features)**  Training data, where n\_samples in the number of samples and n\_features is the number of features. Sparse matrix allowed. |
| **Returns:** | **self** : object  Returns the instance itself. |

**fit\_transform(X, y=None)**[¶](http://pythonhosted.org/lda/api.html" \l "lda.lda.LDA.fit_transform" \o "Permalink to this definition)

Apply dimensionality reduction on X

|  |  |
| --- | --- |
| **Parameters:** | **X** : array-like, shape (n\_samples, n\_features)  New data, where n\_samples in the number of samples and n\_features is the number of features. Sparse matrix allowed. |
| **Returns:** | **doc\_topic** : array-like, shape (n\_samples, n\_topics)  Point estimate of the document-topic distributions |

**loglikelihood()**[¶](http://pythonhosted.org/lda/api.html#lda.lda.LDA.loglikelihood)

Calculate complete log likelihood, log p(w,z)

Formula used is log p(w,z) = log p(w|z) + log p(z)

**transform(X, max\_iter=20, tol=1e-16)**[¶](http://pythonhosted.org/lda/api.html#lda.lda.LDA.transform)

Transform the data X according to previously fitted model

|  |  |
| --- | --- |
| **Parameters:** | **X** : array-like, shape (n\_samples, n\_features)  New data, where n\_samples in the number of samples and n\_features is the number of features.  **max\_iter** : int, optional  Maximum number of iterations in iterated-pseudocount estimation.  **tol: double, optional**  Tolerance value used in stopping condition. |
| **Returns:** | **doc\_topic** : array-like, shape (n\_samples, n\_topics)  Point estimate of the document-topic distributions |

## lda.utils[¶](http://pythonhosted.org/lda/api.html" \l "module-lda.utils" \o "Permalink to this headline)

**lda.utils.check\_random\_state(seed)**[¶](http://pythonhosted.org/lda/api.html#lda.utils.check_random_state)

**lda.utils.dtm2ldac(dtm, offset=0)**[¶](http://pythonhosted.org/lda/api.html#lda.utils.dtm2ldac)

Convert a document-term matrix into an LDA-C formatted file

|  |  |
| --- | --- |
| **Parameters:** | **dtm** : array of shape N,V |
| **Returns:** | **doclines** : iterable of LDA-C lines suitable for writing to file |

Notes

If a format similar to SVMLight is desired, offset of 1 may be used.

**lda.utils.ldac2dtm(stream, offset=0)**[¶](http://pythonhosted.org/lda/api.html#lda.utils.ldac2dtm)

Convert an LDA-C formatted file to a document-term array

|  |  |
| --- | --- |
| **Parameters:** | **stream: file object**  File yielding unicode strings in LDA-C format. |
| **Returns:** | **dtm** : array of shape N,V |

Notes

If a format similar to SVMLight is the source, an offset of 1 may be used.

**lda.utils.lists\_to\_matrix(WS, DS)**[¶](http://pythonhosted.org/lda/api.html#lda.utils.lists_to_matrix)

Convert array of word (or topic) and document indices to doc-term array

|  |  |
| --- | --- |
| **Parameters:** | **(WS, DS)** : tuple of two arrays  WS[k] contains the kth word in the corpus DS[k] contains the document index for the kth word |
| **Returns:** | **doc\_word** : array (D, V)  document-term array of counts |

**lda.utils.matrix\_to\_lists(doc\_word)**[¶](http://pythonhosted.org/lda/api.html#lda.utils.matrix_to_lists)

Convert a (sparse) matrix of counts into arrays of word and doc indices

|  |  |
| --- | --- |
| **Parameters:** | **doc\_word** : array or sparse matrix (D, V)  document-term matrix of counts |
| **Returns:** | **(WS, DS)** : tuple of two arrays  WS[k] contains the kth word in the corpus DS[k] contains the document index for the kth word |