Steps for creating the corpus :

1. A collection of documents
2. Remove all commons word and words that occurs once in the corpora
3. Split on whitespace and lowercase the words: this mimics the Deerwester et al’s original article on LSA
4. A document is represented by the features extracted from it.
5. Here they use the bag-of-words model
   1. Each document is represented by the features extracted from it
   2. Each vector element represents a question answer pair
6. eg : how many times does the word “system” appear in the doc?
7. The mapping between the questions & ids is called dictionary
8. gensim.corpora.dictionary.Dictionary : sweeps across the text, collecting word counts and relevant statistics
9. At the end we have the distinct set of words in the corpora
10. Each document will be represented by an N-D array.
11. In order to convert tokenized documents into vectors : doc2bow : counts the number of occurrences of each distinct word, converts and returns as a sparse vector
12. Corpus resides in memory, as a plain python list.

Topic Transformation :

In this tutorial, I will show how to transform documents from one vector representation into another. This process serves two goals:

1. To bring out hidden structure in the corpus, discover relationships between words and use them to describe the documents in a new and (hopefully) more semantic way.
2. To make the document representation more compact. This both improves efficiency (new representation consumes less resources) and efficacy (marginal data trends are ignored, noise-reduction).

Creating a transformation:

1. The transformations are standard Python objects, typically initialized by means of a ***training corpus***
2. Transformations always convert between two specific vector spaces. The same vector space (= the same set of feature ids) must be used for training as well as for subsequent vector transformations. Failure to use the same input feature space, such as applying a different string preprocessing, using different feature ids, or using bag-of-words input vectors where TfIdf vectors are expected, will result in feature mismatch during transformation calls and consequently in either garbage output and/or runtime exceptions.

Transforming vectors

1. From now on, tfidf is treated as a read-only object that can be used to convert any vector from the old representation (bag-of-words integer counts) to the new representation (TfIdf real-valued weights):