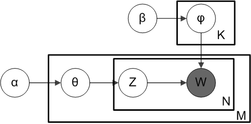
**LDA**

**Latent Dirichlet allocation (LDA) is a** [**statistical text model**](https://en.wikipedia.org/wiki/Generative_model)**, that allows sets of observable variables to be explained by  Latent(unobservable ) variables , where in the texts , the words is the observables variables and topics is the latent variables. , in this model, it is working based on having the values of prior per document topic distributions and prior per topic word distributions.LDA , has graphical model called** [plate notation](https://en.wikipedia.org/wiki/Plate_notation) .



the outer box represent the docuements , the inner boxes , represented the topics and words within the document.

M: is the number of documents

N: is the number of words

K: is is number of the topic

*α* is the parameter of the Dirichlet prior on the per-document topic distributions,

*β* is the parameter of the Dirichlet prior on the per-topic word distribution,

ᶿi is *the topic distribution for document i,*

ᶲk {\displaystyle \theta \_{i}} is the word distribution for topic *k*,

{\displaystyle \varphi \_{k}}

Zij{\displaystyle z\_{ij}} is the topic for the *j*th word in document *i*, and

Wij{\displaystyle w\_{ij}} is the specific word.

The {\displaystyle w\_{ij}}Wij  are the only [observable variables](https://en.wikipedia.org/wiki/Observable_variable), and the other variables are [latent variables](https://en.wikipedia.org/wiki/Latent_variable).

The generative process is as follows. Documents are represented as random mixtures over latent topics, where each topic is characterized by a distribution over words.

Explanation with the examples and how to implement:

Introduction

Suppose you have the following set of sentences:

* I like to eat broccoli and bananas.
* I ate a banana and spinach smoothie for breakfast.
* Chinchillas and kittens are cute.
* My sister adopted a kitten yesterday.
* Look at this cute hamster munching on a piece of broccoli.

What is latent Dirichlet allocation? It’s a way of automatically discovering **topics** that these sentences contain. For example, given these sentences and asked for 2 topics, LDA might produce something like

* **Sentences 1 and 2**: 100% Topic A
* **Sentences 3 and 4**: 100% Topic B
* **Sentence 5**: 60% Topic A, 40% Topic B
* **Topic A**: 30% broccoli, 15% bananas, 10% breakfast, 10% munching, … (at which point, you could interpret topic A to be about food)
* **Topic B**: 20% chinchillas, 20% kittens, 20% cute, 15% hamster, … (at which point, you could interpret topic B to be about cute animals)

The question, of course, is: how does LDA perform this discovery?

LDA Model

In more detail, LDA represents documents as **mixtures of topics** that spit out words with certain probabilities. It assumes that documents are produced in the following fashion: when writing each document, you

* Decide on the number of words N the document will have (say, according to a Poisson distribution).
* Choose a topic mixture for the document (according to a Dirichlet distribution over a fixed set of K topics). For example, assuming that we have the two food and cute animal topics above, you might choose the document to consist of 1/3 food and 2/3 cute animals.
* Generate each word w\_i in the document by:
  + First picking a topic (according to the multinomial distribution that you sampled above; for example, you might pick the food topic with 1/3 probability and the cute animals topic with 2/3 probability).
  + Using the topic to generate the word itself (according to the topic’s multinomial distribution). For example, if we selected the food topic, we might generate the word “broccoli” with 30% probability, “bananas” with 15% probability, and so on.

Assuming this generative model for a collection of documents, LDA then tries to backtrack from the documents to find a set of topics that are likely to have generated the collection.

Example

Let’s make an example. According to the above process, when generating some particular document D, you might

* Pick 5 to be the number of words in D.
* Decide that D will be 1/2 about food and 1/2 about cute animals.
* Pick the first word to come from the food topic, which then gives you the word “broccoli”.
* Pick the second word to come from the cute animals topic, which gives you “panda”.
* Pick the third word to come from the cute animals topic, giving you “adorable”.
* Pick the fourth word to come from the food topic, giving you “cherries”.
* Pick the fifth word to come from the food topic, giving you “eating”.

So the document generated under the LDA model will be “broccoli panda adorable cherries eating” (note that LDA is a bag-of-words model).

# Learning (how to implement)

So now suppose you have a set of documents. You’ve chosen some fixed number of K topics to discover, and want to use LDA to learn the topic representation of each document and the words associated to each topic. How do you do this? One way (known as collapsed Gibbs sampling) is the following:

* Go through each document, and randomly assign each word in the document to one of the K topics.
* Notice that this random assignment already gives you both topic representations of all the documents and word distributions of all the topics (albeit not very good ones).
* So to improve on them, for each document d…
  + Go through each word w in d…
    - And for each topic t, compute two things: 1) p(topic t | document d) = the proportion of words in document d that are currently assigned to topic t, and 2) p(word w | topic t) = the proportion of assignments to topic t over all documents that come from this word w. Reassign w a new topic, where we choose topic t with probability p(topic t | document d) \* p(word w | topic t) (according to our generative model, this is essentially the probability that topic t generated word w, so it makes sense that we resample the current word’s topic with this probability). (Also, I’m glossing over a couple of things here, in particular the use of priors/pseudocounts in these probabilities.)
    - In other words, in this step, we’re assuming that all topic assignments except for the current word in question are correct, and then updating the assignment of the current word using our model of how documents are generated.
* After repeating the previous step a large number of times, you’ll eventually reach a roughly steady state where your assignments are pretty good. So use these assignments to estimate the topic mixtures of each document (by counting the proportion of words assigned to each topic within that document) and the words associated to each topic (by counting the proportion of words assigned to each topic overall).