LDA(Latent Dirichlet allocation) 模型是一个生成模型，它刻画了一个语料(有很多的普通文档组成)每个文档各个位置的单词是怎么生成的，这有点类似于解数学题里的假设变量，求解变量的过程。

在BleiNJ03中是这样描绘该生成过程的：



解释一下这个生成过程：

步骤1~3只是描述了一篇文档是怎么生成的，对于多篇文档，按照步骤1~3依次生成每篇文档。

过程中涉及到的变量和参数解释：

：主题个数

：词典中单词的个数

：泊松分布的参数，是个标量

：生成文档的长度，标量

：Dirichlet分布的参数，维向量

：生成文档的主题分布(多项分布)，维向量，在后面表示的第个元素

：表示生成文档第个位置的单词，维向量，如果是单词，并且为整数，则的第个位置元素为1，其它元素均为0。后面会涉及到一个相关的变量，该变量是个标量，取值0或1，若是单词，则为1，否则为0，可以把它理解为的第个元素。

：表示生成文档第个位置的单词所属于的主题，维向量，如果是主题，并且为整数，则的第个位置元素为1，其它元素均为0。后面会涉及到一个相关的变量，该变量是个标量，取值0或1，若是主题，则为1，否则为0，可以把它理解为的第个元素。

：矩阵，行是主题，列是单词在词典中的索引。元素，即从主题生成单词的概率。应该是按行正规化，即每一行的元素加和为1。后面用表示的第行。(实际计算时按列正规化，不过不是太重要)

步骤1 ，该文档按照参数生成文档的长度。在BleiNJ03中，这样解释



在后面的模型中，会忽略变量。生成文档的长度取值实际文档的长度。

步骤2，从参数为的Dirichlet分布生成文档的主题分布，



步骤3，对于个位置，依次进行步骤(a)，(b)。

步骤(a)：从参数为的多项分布生成主题，



步骤(b)：从参数为的多项分布生成单词，



对于一篇文档，按照步骤1~3生成，，。是的矩阵，即所生成文档的所有主题的表示，为其第个元素。是的矩阵，即所生文档所有单词的表示，为其第个元素。

按照步骤1~3，把各个子步骤的概率相乘，我们得到该篇文档的生成概率。

 (1)

在公式(1)中，参数是和，即我们假设其是已知的，最终要求解它们。而，我们假设它们就是未知的，最终也不需要求解出具体的数值，我们称这种变量为隐含变量(latent variables)，隐含变量能够有效的刻画已知变量(这里是)之间的关系。

因为，从始至终我们都把它们当作未知量，所以要把它们积分出来，这样就得到一篇实际文档(假设单词都已知，就像本篇文档一样，虽然还没写完，为)的模型(LDA)概率：

一个语料(假设为,共篇文档)，每一篇文档都按照步骤1~3来生成，我们得到该实际语料的模型(LDA)概率：

 (2)

公式(2)也是LDA模型我们最终要优化的目标函数。

先总结一下LDA模型，

两个参数：，

两个隐含变量：，（I）

一个已知变量(或称观测变量)：

模型(由参数、变量、目标函数定义)我们知道是什么样子了，下面的问题是怎么求解该模型，即求解使公式(2)最大的参数值，。

对于这种带有隐含变量的模型，一般的求解方法是EM算法。下面简单看一下EM算法。假设一个模型已知变量是，参数是，隐含变量是。其似然函数是：



 (3)

上面不等式的原理是Jensen’s inequality。Jensen’s inequality是说对于concave function (这里是log函数)，

我们将最大化的问题转变为最大化其下界(lower bound)的问题。这里的下界并不是的最小值，而是另一个函数()，该函数的曲线始终在原函数()之下。就像把一块布扣在一个倒置的碗上一样，我们求不出布最高的位置的坐标(二维)，通过找出碗最高点的坐标来近似。

EM算法每次迭代分为两个子步骤：

(E step) 

(M step) 

E-step 我们假设维持不变，改变来使变大。M-step 我们假设不变，改变来使变大。可以证明，E-step和M-step都可以使变大或不变，但不会变小。具体证明过程可以参考(Probabilistic graphical models, Jordan or Koller的EM 算法部分)。我们还可以证明，如果的话可以使(3)不等式变为等式。

如果可以计算出话，我们可以省略E-step而直接使用M-step。

回到我们的LDA模型，如果可以求解出的话，就可以使优化问题变得相对简单一些(EM算法里常用方法是在E-step求解隐含变量的后验概率，很少有直接优化E-step本身的)。

但是：



其中



是计算不出来的。

所以使用另一个简单的分布来近似。通过公式(3)可以看出，使用近似虽然不能求解出（这里是）真实的最大值，但能够保证求解出它的一个下界的最大值。

这种方法称为变分方法，这是因为和都是中没有的，这种方法的效果是在目标函数中增加了新的参数。和都是文档级的参数(相比之下，和是语料级的参数)。是一个维向量，是生成该文档Dirichlet分布的参数。是一个二维矩阵，行是单词的位置（或者说文档一个单词位置）的索引，列是主题的索引，元素表示文档的第个位置的单词属于主题的概率。

 (4)

根据公式(3)中所示，对于一篇文档我们的优化目标由变成下面的函数：



 (5)

只是一种写法，该函数的参数有四个，(原有参数)和，(变分参数)。将公式(1)和(4)带入公式(5)中，变为下面的形式：

(6)

后五行分别是五个分解开的结果，拿第一个来看看是怎么分解的。





 (7)

上式中需要求解的一个量是,在BleiNJ03 A1 中，若

则

这里比较有意思的一个问题是，从参数为和参数为的Dirichlet分布我们都可以生成。那这里的究竟应该是还是呢?我们可以分析一下，因为



因为我们要对求积分，那就是说是变化的，会取尽所有合法值。说明，是随着变化的，而不是随着变化的。所以这里应该是，而不是。

所以可以继续公式(7)



其它四个可以用类似的过程分解,最终得到公式(6)。

再总结一下LDA模型，在（I）中，LDA模型的定义如下：

目标函数：



参数和变量：

两个参数：，

两个隐含变量：，

一个已知变量(或称观测变量)：

然而我们无法优化上述模型，所以采用了目标函数的一个下界来近似它，并且采用了变分的方法。这样我们的模型就变成了如下形式：

目标函数：





即我们的目标函数是

参数和变量：

四个参数：一般参数，和变分参数，

两个隐含变量：，（定义II）

一个已知变量(或称观测变量)：

虽然采用了变分方法，但是参数的求解过程依然采用EM算法。

E-Step ：

优化，通过优化，实现

因为，是文档级的参数，所以M-Step是逐篇文档进行。

对于一篇文档，目标函数如公式(6)所示，

中包含的部分是：



最后一项是拉格朗日因子，是个标量。

对求导数，结果如下：



令导数为0得：



中只包含的部分是

i





设置导数为0，得：



M-Step ：优化参数，(它们是语料级参数)



求解导数并设置为0，得：









不能准确地解出来，所以但其导数和曲率容易求解出来，一般采用牛顿方法进行优化。

(注：，和，的计算细节仍需要整理)



LDA(gibbs sampling version)



**LDA所涉及符号的解释(与变分版基本相同)**

|  |  |  |
| --- | --- | --- |
| **变量** | **类型** | **解释** |
| K | 整数 | 预设主题的个数，例如50 |
|  | 整数 | 词典中单词的个数(无重复) |
|  | 整数 | 文档的个数 |
|  | 整数 | 文档中单词的个数 |
| N | 整数 | 所有文档中单词的个数之和， |
| \alpha_{k=1 \dots K} | 正实数 | 主题在一篇文档中的先验权重；通常队于所有的主题都相同；一般是小于1，例如0.1，倾向于稀疏的主题分布，例如一篇文档只有少量的主题 |
| \boldsymbol\alpha | 正实数的维向量 | \alpha_k值的集合，通常作为一个向量 |
| \beta_{w=1 \dots V} | 正实数 | 单词*w*在一个主题中的先验权重；通常对于所有的单词都相同；一般是一个小于1的数，例如0.001，倾向于稀疏的主题分布，即每个主题有较少的单词 |
| \boldsymbol\beta | *V*维的正实数向量 | \beta_w值的集合，通常作为一个向量 |
| \phi_{k=1 \dots K,w=1 \dots V} | 概率（0和1之间的正实数） | 单词在主题k中的概率 |
| \boldsymbol\phi_{k=1 \dots K} | 维的概率向量，元素加和为1 | 单词在主题k中的分布 |
| \theta_{d=1 \dots M,k=1 \dots K} | 概率（0和1之间的正实数） | 文档d中单词属于主题k的概率 |
| \boldsymbol\theta_{d=1 \dots M} | 维的概率向量，元素加和为1 | 文档d中的主题分布 |
| z_{d=1 \dots M,w=1 \dots N_d} | 1和K之间的整数 | 单词w在文档d中的赋值 |
|  | N维整数向量 | 文档d中所有单词（可重复）的赋值 |
|  | 1和V之间的整数 | 文档d中所有单词w的赋值 |
| \mathbf{W} | N维整数向量 | 所有文档包含所有单词的赋值 |



这里我们要使用和的联合概率分布，所以把隐含变量和积分出来([collapsed Gibs sampling](http://en.wikipedia.org/wiki/Collapsed_Gibbs_sampling))：



和相互独立，分别处理：



不同文档相互独立，所以只考虑文档：





同样的道理：



合并和部分









因为是确定的,所以可以继续简化上式

