LDA TUTORIAL

1. Overview

LDA (Latent Dirchlet Allocation) is an unsupervised machine learning technique that is often used in NLP (natural language processing) to determine things like what topic a document is about, or what documents are most similar. But first we are going to do a detour and explain some background material.

2. BackGround

2.1. **Conjugate Prior.**

The term conjugate prior will come up several times so it's good to know what a conjugate prior is. Say you have a conditional probability distribution $p(x|\theta)$. What happens if you multiply it by $p(\theta)$. In general, you can't say. But if $p(\theta)$ is the conjugate prior to the likelihood function, i.e., $p(x|\theta)$, then the posterior distribution $p(\theta|x)$, that results from $\frac{p(x|\theta)p(\theta)}{\text{Normalization Constant}}$ is in the same family as $p(\theta)$. Think of this as saying the posterior distribution "looks like" the prior.

2.1.1. *Example: Simple Multi-Arm Bandit.*

You have a row of slot machines. Each machine$_i$ has a probability p_i of dispensing a dollar or not. You want to play these slot machines as to maximize your reward. The p_i's are independent; that is, the order which you pull levers is not significant, and they don't change.

(TODO: FILL OUT EXAMPLE)

2.2. **Binomial, Multinomial, and Dirchlet Distribution.**

The binomial, multinomial, and dirchlet distribution are all connected. Given that LDA has dirchlet in its title; it's a safe bet that we will come across the dirchlet distribution. Since they're all connected, it's helpful to start with the former and move up to the latter. The multinomial distribution is a generalization of the the binomial, and the Dirchlet can be thought of as way of specifying a distribution over multinomial distributions.

But first, let's review the binomial distribution.

2.2.1. *Binomial.*

The most straightforward way of understanding the binomial distribution is to think of coin flips. The binomial distribution answers the questions what is the probability of getting s successes in a total of n trials. From Wikipedia, the probability mass function (aka the probability that there are s successes) is defined as:

$$P(X = k)^1 = \binom{n}{k} p^k (1-p)^{n-k}$$

.

Notice that we are limited to binary events: something happened or it didn't (like a coin flip). More technically, binomial only concerns itself with probability of k successes of n Bernoulli trials. The $\binom{n}{k}$ part is there because there may be several ways to get k successes.

But what happens if we want to get the probability that a particular side of a dice comes up? We have moved beyond to the non-binary world.

Enter the multinomial distribution.

2.2.2. *Multinomial.* Per Wikipedia, the probability

$$Pr(X_1 = x_1 \dots X_k = x_k) = \frac{n!}{x_1! \dots x_k!} p_1^{x_1} \dots p_k^{x_k}$$

.

For the time being, we can just ignore $\frac{n!}{x_1! \dots x_k!}$. The above formula is just a generalization of the binomial. To help see this, notice that that since this is a probability distribution, it must sum to 1; so $p_k = 1 - \sum_{i=1}^{k-1} p_i$. Likewise, $x_k = N - \sum_{i=1}^{k-1} x_i$. This is similar to the binomial, where the failures are defined in terms of the $N - k$,where k is number of successes, and the probability of failure is $1 - p$.

How to interpret the multinomial? Think of a die, and each p_i as the probability that side i comes up. This is just like in the binomial distribution, where p defined the probability that a particular side comes up (say, heads). So the probability that side i comes up is just p_i. The probability that you see 5 side is, is just p_i^5.

Technically, we need to add in the combinatorics junk: $\frac{n!}{x_1! \dots x_k!}$. Why? Well, often we don't need to because it is a fixed constant; so if you are doing, say, MLE you can just ignore it. Or if you are just trying to find the maximum p_i, you don't need it. However, if the numerical value *itself*, not the relative values of the p_i, is important, you need the combinatorics junk. By the way, this combinatorics junk is called the multinomial coefficient.

But for the sake of understanding LDA, just think of the multinomial distribution as a distribution that when you sample from it gives you a k length vector where all entries are non-negative and they sum up to 1.

2.2.3. *Extra Credit on the Combinatorics junk.* Why do we need it? Recall from combinatorics that $\frac{n!}{x_1! \dots x_k!}$ are the number of different permutations of n things and x_i are the number of indistinguishable of kind i. This is just a generalization the binomial theorem's $\binom{n}{k}$. That is, the multinomial coefficient is the number of ways of taking n trials and getting x_i successes for i objects. TODO: EXLPLAIN WHY TALK ABOUT GAMMA FUNCTION

[1] k is the same as s

3. LDA: Overview

3.1. **Storytelling.** LDA is a generative model that tells a story, a story about how the documents in our dataset were. created. Let K be the number of topics, V the number of terms in our dataset. For simplicity, let each topic K_i have V elements. Here is the generative story of how your corpus came to be, as told by LDA:

(1) For each document:
   (a) Choose distribution over topics according to some distribution (English: choose a distribution that says how important each topic is for document d.)
   (b) for each word (feature) in the document:
      (i) Draw a topic k_i from the K topics.
      (ii) Draw a word (feature) w_i from k_ith topic.

Obviously, this is not how documents are written in real-life. But this "story" might still be of use. First and foremost, this generative story captures the fact that documents contain multiple topics. Moreover, it captures that documents contain different proportions of topics. For example, a document may contain a lot of words from the topic "health" and only a few from "weather", or it may contain a lot of words from both topics. Also, we have not made any assumption about the word-order in a document. This is called the *bag-of-words assumption.*

Our goal will be to learn K topics automatically.

3.1.1. *Side Note: Topics.* You may be wondering, "what are topics?" Topics in LDA have a semi-intuitive understanding. Topics are distributions over words; that is, the probability of drawing a word under different topics will (typically) be different. Let topic 0 be 'sports' so one would expect that the probability of drawing the word 'athlete' to be higher than in the topic 'technology'.

3.1.2. *LDA: Generative Model Restated.*

(1) For each topic: $1 \dots k$
   (a) Let ϕ^k be the distribution of words for topic k. That is, draw $\phi^k \sim Dir(\beta)$
(2) For each document:
   (a) Draw a topic distribution according to a Dirchlet distribution; that draw topic $\theta \sim Dir(\alpha)$
   (b)
   (c) for each word (feature) in the document:
      (i) Draw a topic $k_i \sim multinomial(\theta)$
      (ii) Draw a word (feature) w_i from k_ith topic; i.e., sample from multinomial distribution, where each entry in the multinomial distribution is the probability of selecting a word under topic k_i. That is, draw w_i from ϕ^{k_i}.

4. Inference in LDA

4.1. **What we know and what we don't.** Currently, we have a generative story of how documents are "written". It's not plausible, but it does seem to capture some useful facts about documents: they have different topics, they contain one more topics, and the choose of topics greatly influences the choice of words.

Unfortunately, we *don't know* the topics. More accurately, we don't know the distribution of topics for any of the documents, the word distribution under a topic, or the number of topics. The latter can be "solved". The number of topics is chosen by guessing and checking combined with any prior knowledge of the corpus. The rest we have to learn. We need to learn them from the words in the corpus. The words and what documents contain what words are the only things we do know.

4.2. **The Inference Problem.** The generative process gives us the following joint distribution:

$$p(\mathbf{w}, \mathbf{z}, \theta, \phi | \alpha, \beta).$$

If you know the graphical model of LDA, then you know then it can be decomposed into:

$$p(\phi|\beta)p(\theta|\alpha)p(\mathbf{z}|\theta)p(\mathbf{w}|\phi_{\mathrm{z}}).$$

What are these variables? $\mathbf{z}, \phi, \theta$ are called *latent variables.* z_i is the topic assignment for word w_i. ϕ is the distribution of words in the topics. That is, ϕ^k is the distribution of words in topic k. θ is the distribution of topics for each document; that is ϕ_d is the distribution of topics for document d.

Another way of thinking of these variables is that θ_d is a representation of document d in "topic-space", z_i says what topic generated w_i, and ϕ^k gives us all the probabilities of a word given topic k. In fact, we can think of ϕ as (large) matrix s.t. $\phi_{i,j}$ gives us the probability of $p(w_i|z_j)$.

These latent variables are the very things we are interesting in learning. However, the only thing we know are the words and what documents contain what words. In other words, what we want to learn is

$$p(\theta, \mathbf{z}, \phi | \mathbf{w}, \alpha, \beta)$$

It's instructive to write out the relationship between the above formulas:

$$\begin{aligned} p(\theta, \mathbf{z}, \phi | \mathbf{w}, \alpha, \beta) &= \frac{p(\mathbf{w}, \mathbf{z}, \theta, \phi | \alpha, \beta)}{p(\mathbf{w}|\alpha, \beta)} \\ &= \frac{p(\phi|\beta)p(\theta|\alpha)p(\mathbf{z}|\theta)\mathbf{p}(\mathbf{w}|\phi_\mathbf{z})}{p(\mathbf{w}|\alpha, \beta)} \end{aligned}$$

What can we do with this? **Well, with a little thought one can see that the *$z's$* are all we really need.** In other words, ϕ and θ can be calculated from the z's. If I know the what words were assigned what topics (and what words go with what documents), I can calculate the probability a document has a topic ($\theta_{d,z}$) and I can calculate $\phi_{w,z}$, the probability of a word under a topic.

How can we get rid of θ and ϕ? Integrate them out.

$$\begin{aligned} \int\int \frac{p(\phi|\beta)p(\theta|\alpha)p(\mathbf{z}|\theta)\mathbf{p}(\mathbf{w}|\phi_z)\mathbf{d}\phi\mathbf{d}\theta}{p(\mathbf{w}|\alpha, \beta)} &= \frac{\int p(w|\phi_z)p(\phi|\beta)d\phi \int p(\mathbf{z}|\theta)p(\theta|\alpha)d\theta}{p(\mathbf{w}|\alpha, \beta)} \\ &= (Z^{-1})(p(w|\mathbf{z}, \beta)p(\mathbf{z}|\alpha)) \\ &= (Z^{-1})p(w, \mathbf{z}|\alpha, \beta) \\ &= p(\mathbf{z}|\alpha, \beta, \mathbf{w}) \end{aligned}$$

(Z^{-1}) is a normalization constant. Here it is $p(\mathbf{w}|\alpha, \beta)$.

The million dollar question: how do we figure out: $p(\mathbf{z}|\alpha, \beta, \mathbf{w})$?

4.3. **Answering the million dollar question.**

4.3.1. *Plug In.* We know what some of the distributions are, like the $p(\theta|\beta)$ is a Dirichlet distribution. So let us plug and (hopefully) chug.

(1) $p(w|\phi_{z_i})$ has a multinomial distribution. So we can write it as:
$p(w|\phi_{z_i}) = \prod_i^N \phi_{z_i, w_i} = \prod_k \prod_V \phi_{k,v}^{s_{k,v}}$, where $s_{k,v}$ is a matrix where entry k, v gives the number of times w_v has been assigned z_k (topic k).

(2) $p(\phi|\beta)$ has a Dirichlet distribution. Recall that ϕ is really a 2d matrix. So we can write $p(\phi|\beta) = \prod_k \frac{1}{B(\beta)} \prod_v \phi_{k,v}^{\beta-1}$.

(3) $p(\mathbf{z}|\theta)$ has a multinomial distribution. So it can be written as
$p(\mathbf{z}|\theta) = \prod_d \prod_k \theta_{d,k}^{r_{d,k}}$

(4) $p(\theta|\alpha)$ has a Dirichlet distribution. Thus, $p(\theta|\alpha) = \prod_d \frac{1}{B(\beta+s)} \prod_k \theta_{d,k}^{\alpha-1}$.

Now we can plug and chug. As you will see, conjugate priors will become very useful.

$$p(\mathbf{w}|\mathbf{z},\beta) = \int p(w|\phi_z))p(\phi|\beta)d\phi = \int \prod_k \frac{1}{B(\beta)} \prod_w \phi_{k,w}^{s_{k,w}} \phi_{k,w}^{\beta-1} d\phi_k \tag{1}$$

$$= \int \prod_k \frac{1}{B(\beta)} \prod_w \phi_{k,w}^{s_{k,w}+\beta-1} d\phi_k \tag{2}$$

$$= \prod_k \frac{B(s_{k,\cdot}+\beta)}{B(\beta)} \tag{3}$$

The last step can be seen via

$$\frac{1}{B(\beta+s_{k,\cdot})} \int \prod_w \phi^{s_{k,w}+\beta-1} d\phi_k = 1 \tag{4}$$

Using the same arguments, you get:

$$p(\mathbf{z}|\alpha) = \int p(\mathbf{z}|\theta)p(\theta|\alpha)d\theta_d = \prod_d \frac{B(\tau^{d,\cdot}+\alpha)}{B(\alpha)} \tag{5}$$

Thus,

$$(Z^{-1})p(\mathbf{w},\ \mathbf{z}|\alpha,\beta) = (Z^{-1}) \prod_k \frac{B(s_k+\beta)}{B(\beta)} \prod_d \frac{B(\tau^{d,\cdot}+\alpha)}{B(\alpha)} \tag{6}$$

Alas, it seems we are almost dk,one. But what about Z^{-1}, the normalization constant? The normalization constant is intractable. We have to sum out all the latent variables, and there are no tricks like conjugate priors to help us out. There are several different ways of seeing this is intractable. One is to note that, due to the graphical model, we could split up the joint distribution with the ϕ's and those with θ's. This is what allowed us to separate the the integral into 2 integrals $d\phi$ and $d\theta$. Then conjugacy took over and we could efficiently compute $p(w, \mathbf{z}|\alpha, \beta)$. However, no such tricks exist for our normalization constant. Why? We cannot totally separate ϕ and $\mathbf{z}$. More accurately, w depends on both $\mathbf{z}$ and ϕ. So we cannot do any "splitting up" like we did to compute $p(\mathbf{w},z|\alpha\beta)$. We will use a technique called Gibbs sampling to try and get around this.

5. Gibbs Sampling

Gibbs sampling falls under MCMC methods, which is another way of saying it uses sampling to approximate a distribution (for us, it is $p(\mathbf{z}|\alpha,\beta,\mathbf{w})$), but you can compute a conditional distribution (for us, it's $p(z_i|z_0 \dots z_{i-1} \dots z_{i+1} \dots z_k, \alpha, \beta, \mathbf{w})$. For what follows, $\mathbf{z}_{-i}$ means all topic assignments except for z_i, where z_i is some integer from 1 to k.

$$p(z_i|z_0 \dots z_{i-1} \dots z_{i+1} \dots z_k, \alpha, \beta, \mathbf{w}) = \frac{p(\mathbf{z}, \mathbf{w}|\alpha, \beta)}{p(z_{-i}, \mathbf{w}|\alpha, \beta)} \tag{7}$$

$$= \frac{p(\mathbf{w}|\mathbf{z}, \beta)(p(\mathbf{z}|\alpha))}{p(\mathbf{w}_{-i}|\mathbf{z}_i, \beta)(p(w_i)) * p(\mathbf{z}_{-i}|\alpha)} \tag{8}$$

$$\propto \prod_k \frac{B(s_{k,\cdot} + \beta)}{B(s_{k,\cdot}^{-i} + \beta)} \prod_d \frac{B(\tau_{d,\cdot} + \alpha)}{B(\tau_{d,\cdot}^{-i} + \alpha)} \tag{9}$$

To continue on, let's just focus on 1 document and one topic.

$$\frac{B(s_{k,\cdot} + \beta)}{B(s_{k,\cdot}^{-i} + \beta)} \frac{B(\tau_{d,\cdot} + \alpha)}{B(\tau_{d,\cdot}^{-i} + \alpha)} = \frac{\tau_{d,k}^{-i} + \alpha}{\sum_{k'} \tau_{d,k'}^{-i} + \alpha} \frac{s_{w,k}^{-i} + \beta}{\sum_{w'} s_{w',k}^{-i} + \beta} \tag{10}$$

$$\propto (\tau_{d,k}^{-i} + \alpha) \frac{s_{w,k}^{-i} + \beta}{\sum_{w'} s_{w',k}^{-i} + \beta} \tag{11}$$

In words, the probability that w_i in document d is assigned topic $z = j$ is proportional to the number of times d has been assigned j times the probability that a word w_i occurring given a topic. The latter is computed by taking the number of times a w_i has been assigned topic j divided by the number of words in topic j. If you are familiar with naive babes, this is very similar to saying that we are taking the probability of a word occurring given a class times the probability of a topic/label.

6. Algorithm

Algorithm Pseudocode, pg 10.
See Gist for sampling Done in Numpy,
If link doesn't work: it's https://gist.github.com/jsuit/7170ab35faf0ea7fd8b2

6.0.2. *Why?* Why does this work? I didn't find it obvious at first that this would work. You start by randomly initializing everything. And then you sample according to some formula (see 11) . After some thought, this works because you are moving the probability mass around according to two things: (a) the probability a word w_i gets assigned to k and (b) how many times the other words in document d get assigned to k. So (a) lets us take into account what other documents are doing (aka what topics they have). So if d_i has words $w_i, \dots w_n$ and document d_j has words $w_i, \dots, w_p$, then what topics get assigned $w_i \dots w_p$ influence the probability of w_i receiving $topic_k$, and thus also influence the probability d_i and d_j have the same topic distribution. (b) takes into account that what topics been assigned in the current topic.