CS5560 Knowledge Discovery and Management

Problem Set 5

July 3 (T), 2017

Name: Mudunuri Sri Sai Sarat Chandra Varma
Class ID: 14

1. LDA

Read the following articles to learn more about LDA

- https://algobeans.com/2015/06/21/laymans-explanation-of-topic-modeling-with-lda-2/
- http://engineering.intenthq.com/2015/02/automatic-topic-modelling-with-lda/

Consider the topics discovered from Yale Law Journal. (Here the number of topics was set to be 20.) Topics about subjects like about discrimination and contract law.

Figure 3. A topic model fit to the *Yale Law Journal*. Here, there are 20 topics (the top eight are plotted). Each topic is illustrated with its top-most frequent words. Each word's position along the *x*-axis denotes its specificity to the documents. For example "estate" in the first topic is more specific than "tax."

a. Describe the overall process to generate such topics from the corpus.

Answer:

Latent Dirichlet allocation (LDA) is a technique that automatically discovers topics that these documents contain.

This process is done in three steps:

Step 1:

You tell the algorithm how many topics you think there are. You can either use an informed estimate (e.g. results from a previous analysis), or simply trial-and-error. In trying different estimates, you may pick the one that generates topics to your desired level of interpretability, or the one yielding the highest statistical certainty (i.e. log likelihood). In our example above, the number of topics might be inferred just by eyeballing the documents.

Step 2:

The algorithm will assign every word to a temporary topic**.** Topic assignments are temporary as they will be updated in Step 3. Temporary topics are assigned to each word in a semi-random manner (according to a Dirichlet distribution, to be exact). This also means that if a word appears twice, each word may be assigned to different topics. Note that in analyzing actual documents, function words (e.g. "the", "and", "my") are removed and not assigned to any topics.

Step 3 (iterative):

The algorithm will check and update topic assignments, looping through each word in every document. For each word, its topic assignment is updated based on two criteria:

- How prevalent is that word across topics?
- How prevalent are topics in the document?

To understand how these two criteria work, imagine that we are now checking the topic assignment for the word "fish" in Doc Y:

	Document X		Document Y
F	Fish	?	Fish
F	Fish	F	Fish
F	Eat	F	Milk
F	Eat	P	Kitten
F	Vegetables	P	Kitten

- How prevalent is that word across topics? Since "fish" words across both documents comprise nearly half of remaining Topic F words but 0% of remaining Topic P words, a "fish" word picked at random would more likely be about Topic F.

	Document X		Document Y
F	**Fish**	?	Fish
F	**Fish**	**F**	**Fish**
F	Eat	F	Milk
F	Eat	P	Kitten
F	Vegetables	P	Kitten

- How prevalent are topics in the document? Since the words in Doc Y are assigned to Topic F and Topic P in a 50-50 ratio, the remaining "fish" word seems equally likely to be about either topic.

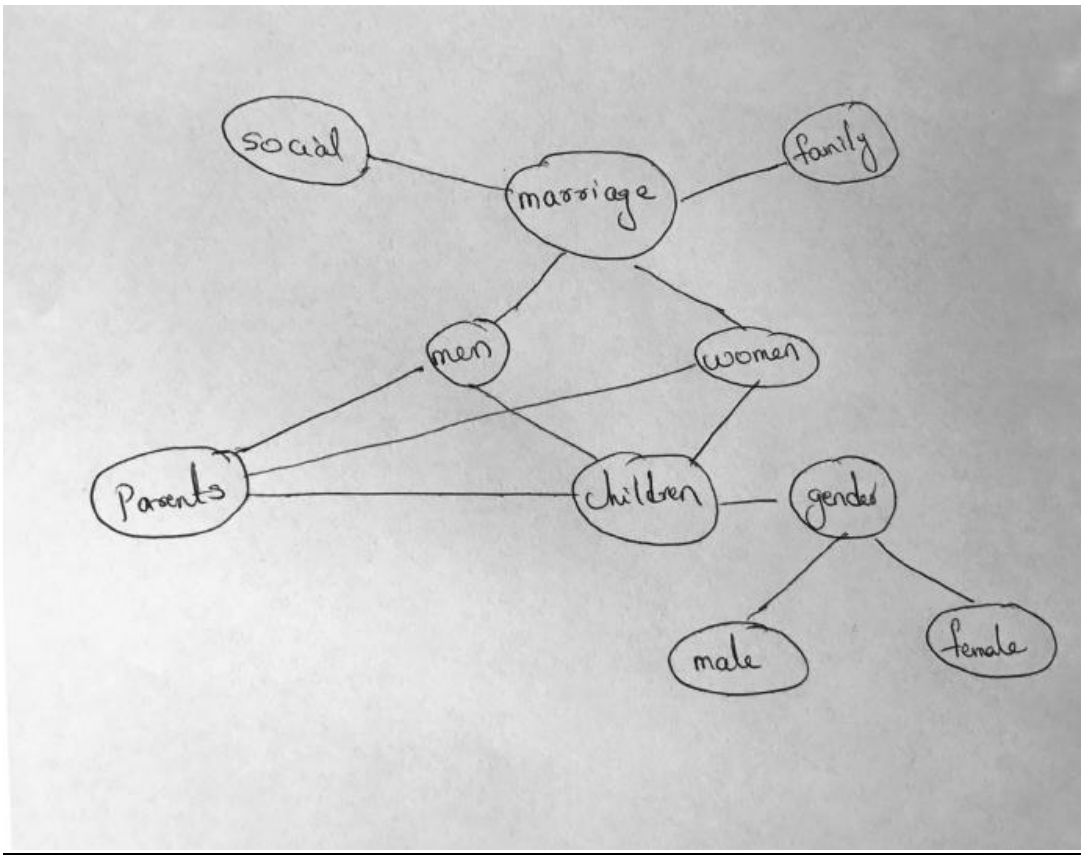
	Document X		Document Y
F	Fish	?	Fish
F	Fish	F	Fish
F	Eat	F	Milk
F	Eat	P	Kitten
F	Vegetables	P	Kitten

Weighing conclusions from the two criteria, we would assign the "fish" word of Doc Y to Topic F. Doc Y might then be a document on what to feed kittens.

The process of checking topic assignment is repeated for each word in every document, cycling through the entire collection of documents multiple times. This iterative updating is the key feature of LDA that generates a final solution with coherent topics.

b. Draw a knowledge graph for Topic 3 in Yale Law Journal (The First Figure).

Answer:



c. Each topic is illustrated with its topmost frequent words. Each word's position along the x-axis denotes its specificity to the documents. For example "estate" in the first topic is more specific than "tax." (the second figure). Describe how to determine the generality or specificity of the terms in a topic**.**

Answer:

When we deal with a huge amount of data on a daily basis. Much of this is textual as we try to get an intimate understanding of articles, web pages, blog posts and opinions. Fortunately, **text mining**, a branch of data science, gives us a hand.

In this post we're going to describe how topics can be automatically assigned to text documents; this process is named, unsurprisingly, **topic-modelling**. It works like fuzzy (or soft) clustering since it's not a strict categorisation of the document to its dominant topic.

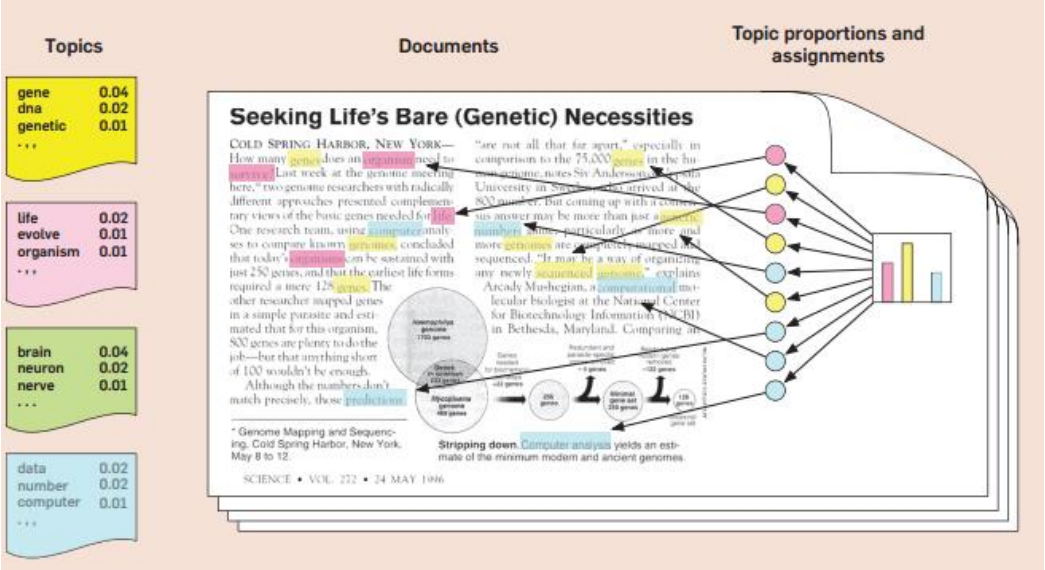
Let's start with an example: the optimal topic-modelling outcome for Shakespeare's Romeo & Juliet would be a composition of topics of circa 50% tragedy and 50% romance. Surely, topics like social networks, football and indian cuisine don't appear in the play, so their weights would be all 0%.

One of the most advanced algorithms for doing topic-modelling is **Latent Dirichlet Allocation** (or **LDA**). This is a probabilistic model developed by Blei, Ng and Jordan in 2003. LDA is an iterative algorithm which requires only three parameters to run: when they're chosen properly, its accuracy is pretty high. Unfortunately, one of the required parameters is the number of topics: exactly as happens with K-means, this requires a deep a-priori knowledge of the dataset.

A good measure to evaluate the performance of LDA is **perplexity**. This measure is taken from information theory and measures how well a probability distribution predicts an observed sample. To evaluate the LDA model, one document is taken and split in two. The first half is fed into LDA to compute the topics composition; from that composition, then, the word distribution is estimated. This distribution is then compared with the word distribution of the second half of the document. a a measure of distance is extracted. Thanks to this measure, in practice, perplexity is often used to select the best number of topics of the LDA model.

Under the hood, LDA models both the topics-per-document and the topic-per-word distribution as Dirichlet distributions (that's why it appears in its name). By using a Markov Chain Monte Carlo (MCMC) method to sample and approximate the underlying Markov Chain stationary distribution (called **Gibbs sampling**), the whole process is iterative, pretty fast, convergent and accurate

d. Describe the inference algorithm that was used in LDA.



Answer:

LDA models each document as a mixture over topics. We assume there are K latent topics, each being a multinomial distribution over a vocabulary of size W. For document j, we first draw a mixing proportion θj = {θjk} over K topics from a symmetric Dirichlet with parameter α. For the ith word in the document, a topic zij is drawn with topic k chosen with probability θjk, then word xij is drawn from the zij th topic, with xij taking on value w with probability ϕkw. Finally, a symmetric Dirichlet prior with parameter β is placed on the topic parameters ϕk = {ϕkw}. The full joint distribution over all parameters and variables is:

p(x, z, θ, φ|α, β) = Y D j=1 Γ(Kα) Γ(α)K QK k=1 θ α−1+njk· jk Y K k=1 Γ(W β) Γ(β)W QW w=1 φ β−1+n·kw kw (

where njkw = #{i : xij = w, zij = k}, and dot means the corresponding index is summed out: n·kw = P j njkw, and njk· = P w njkw. Given the observed words x = {xij} the task of Bayesian inference is to compute the posterior distribution over the latent topic indices z = {zij}, the mixing proportions θ = {θj} and the topic parameters ϕ = {ϕk}. There are three current approaches, variational Bayes (VB) [2], expectation propagation [7] and collapsed Gibbs sampling [5]. We review the VB and collapsed Gibbs sampling methods here as they are the most popular methods and to motivate our new algorithm which combines advantages of both.

2. K-means clustering vs. LDA

Read the K-means clustering for text clustering from https://www.experfy.com/blog/k-means-clustering-in-text-data

(a) Describe the steps how the following 10 documents have moved into 3 different clusters using clustered using k-means (K=3).

Document/Term Matrix

Documents	Online	Festival	Book	Flight	Delhi
D1	1	0	1	0	1
D2	2	1	2	1	1
D3	0	0	1	1	1
D4	1	2	0	2	0
D5	3	1	0	0	0
D6	0	1	1	1	2
D7	2	0	1	2	1
D8	1	1	0	1	0
D9	1	0	2	0	0
D10	0	1	1	1	1

Distance Matrix

	Distance from 3 clusters				
Documents	D2	D5	D7	Min. Distance	Movement
D1	2.0	2.6	2.2	2.0	D2
D2	0.0	2.6	1.7	0.0	
D3	2.4	3.6	2.2	2.2	D7
D4	2.8	3.0	2.6	2.6	D7
D5	2.6	0.0	2.8	0.0	
D6	2.4	3.9	2.6	2.4	D2
D7	1.7	2.8	0.0	0.0	
D8	2.6	2.0	2.8	2.0	D5
D9	2.0	3.0	2.6	2.0	D2
D10	2.2	3.5	2.4	2.2	D2

Answer:

- The first step is to pull the social media mentions for a particular timeframe using social media listening tools (Radian 6, Sysmos, Synthesio etc.). You would need to build query/add keywords to pull the data from social Media Listening tools.
- The next step is data cleansing. This is the most important part as social media comments do not have any specific format. People use locals/slangs etc. on social media to express their emotions, so it's important to be able to see through them and understand the underlying sentiment.
- Remove punctuations, numbers, stopwords (R has specific stopword library but you can also create your own list of stopwords). Also, remove duplicate rows or URLs from the social media mentions.
- The next step is to create corpus vector of all the words.
- Once you have created the corpus vector of words, the next step is to create a document term matrix.

Let's visualize the problem with one example. Let's assume that there are 10 documents/mentions and 5 unique words post data cleansing. Below is the document term matrix for this dataset. It shows for how many times one word has appeared in the document. For example, in document 1 (D1), the words *online, book* and *Delhi* have each been mentioned once.

Document/Term Matrix

Documents	Online	Festival	Book	Flight	Delhi
D1	1	0	1	0	1
D2	2	1	2	1	1
D3	0	0	1	1	1
D4	1	2	0	2	0
D5	3	1	0	0	0
D6	0	1	1	1	2
D7	2	0	1	2	1
D8	1	1	0	1	0
D9	1	0	2	0	0
D10	0	1	1	1	1

- Let's assume that we want to create K=3 clusters. First, three seeds should be chosen. Suppose, D2, D5 & D7 are chosen as initial three seeds.

- The next step is to calculate the Euclidean distance of other documents from D2, D5 & D7.
- Assuming: U=Online, V= Festival, X=Book, Y=Flight, Z=Delhi. Then the Euclidean distance between D1 & D2 would be:

  ((U1-U2)^2 + (W1-W2)^2+(X1-X2)^2+ (Y1-Y2)^2+(Z1-Z2)^2)^0.5

Distance Matrix

	Distance from 3 clusters				
Documents	D2	D5	D7	Min. Distance	Movement
D1	2.0	2.6	2.2	2.0	D2
D2	0.0	2.6	1.7	0.0	
D3	2.4	3.6	2.2	2.2	D7
D4	2.8	3.0	2.6	2.6	D7
D5	2.6	0.0	2.8	0.0	
D6	2.4	3.9	2.6	2.4	D2
D7	1.7	2.8	0.0	0.0	
D8	2.6	2.0	2.8	2.0	D5
D9	2.0	3.0	2.6	2.0	D2
D10	2.2	3.5	2.4	2.2	D2

Clusters	**# of Observations**
D2	5
D5	2
D7	3

- Hence, 10 documents have moved into 3 different clusters. Instead of Centroids, Medoids are formed and again distances are re-calculated to ensure that the documents who are closer to a medoid is assigned to the same cluster.
- Medoids are used to build the story for each cluster.

But there is still one important question remaining: How do you choose the optimal number of clusters?

One approach would be to use the Elbow method to choose the optimal number of clusters. This is based on plotting the cost function for various number of clusters and identifying the breakpoints. If adding more clusters is not significantly reducing the variance within the cluster, one should stop adding more clusters. Although this method cannot give you the optimal number of clusters as an exact point, it can give you an optimal range.

(b) Describe the difference (pro and con) of k-means clustering and the LDA topic discovery model.

Answer:

Time Complexity

K-means is linear in the number of data objects i.e. O(n)O(n), where *n* is the number of data objects. The time complexity of most of the hierarchical clustering algorithms is quadratic i.e. O(n2)O(n2). Therefore, for the same amount of data, hierarchical clustering will take quadratic amount of time. Imagine clustering 1 million records?

Shape of Clusters

K-means works well when the shape of clusters are hyper-spherical (or circular in 2 dimensions). If the natural clusters occurring in the dataset are non-spherical then probably K-means is not a good choice.

Repeatability

K-means starts with a random choice of cluster centers, therefore it may yield different clustering results on different runs of the algorithm. Thus, the results may not be repeatable and lack consistency. However, with hierarchical clustering, you will most definitely get the same clustering results.

Off course, K-means clustering requires prior knowledge of K (or number of clusters), whereas in hierarchical clustering you can stop at whatever level (or clusters) you wish.