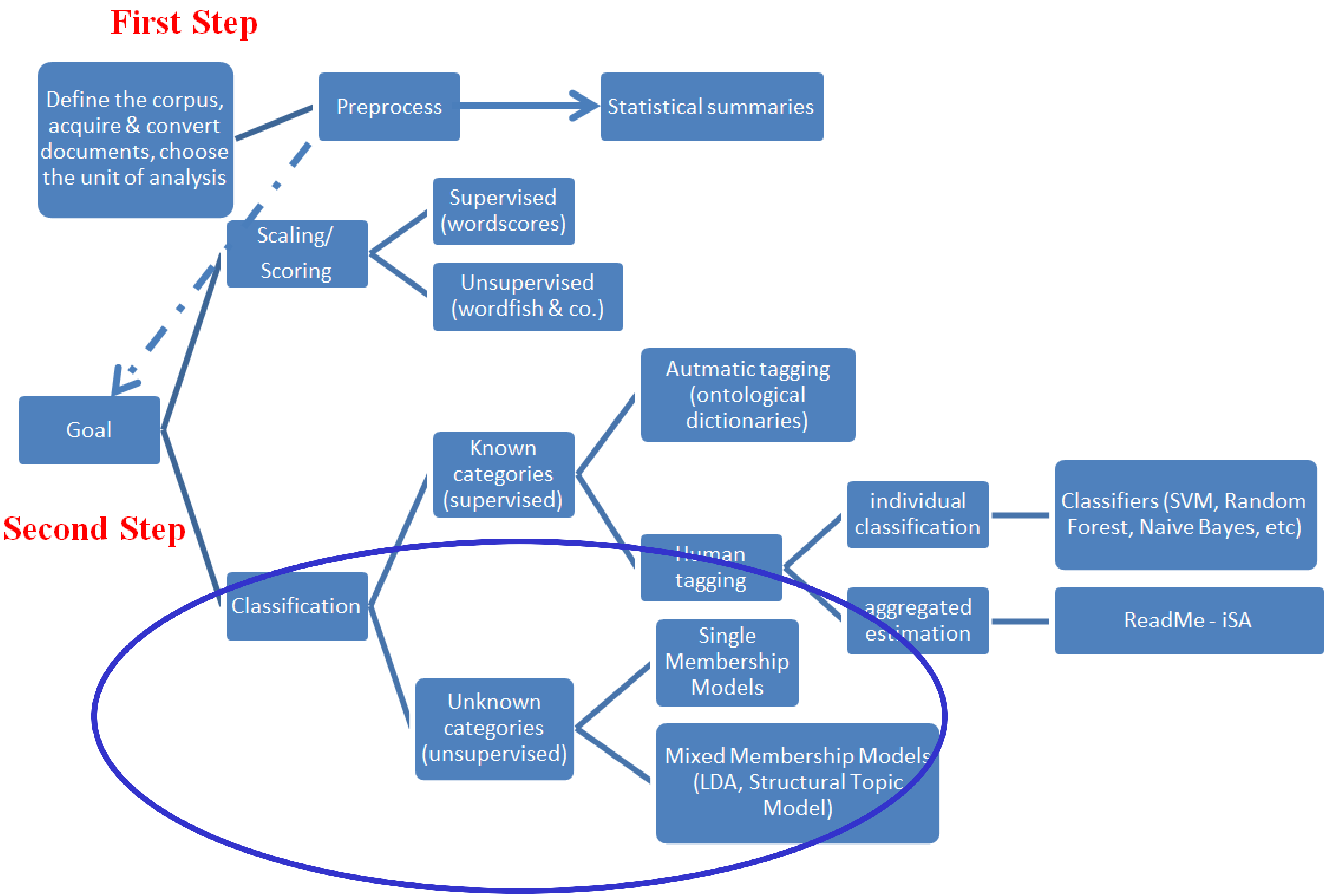
Applied Scaling & Classification Techniques in Political Science

Lecture 4

Unsupervised classification methods: the Topic Model

Our Course Map



Reference

- ✓ Roberts, Margaret E., Brandon M. Stewart, Dustin Tingley, Christopher Luca, Jetson Leder-Luis, Shana Kushner Gadarian, Bethany Albertson, David G. Rand. 2014. Structural Topic Models for Open-Ended Survey Response, *American Journal of Political Science*, 58(4), 1064-1082
- ✓ Roberts, Margaret E., Brandon M. Stewart, Dustin Tingley. 2014. STM: R Package for Structural Topic Models, *Journal of Statistical Software*, https://cran.r-project.org/web/packages/stm/vignettes/stmVignette.pdf

Classification methods

Scaling methods differs from classification methods in that scaling aims to estimate a **position** on a latent dimension, while classification aims to estimate a **text's membership in a class**…more in details…

Classification methods organize texts into a set of **known** (or **unknown**) categories

Classification methods

Sometimes researchers **know the categories** beforehand

In this case, the challenge is to attribute a semantic meaning to each text in a corpus given a **precoded set of words (or texts) that have been already assigned to some categories** (this is why such way of classification is called “**supervised**”)

This step is also called *tagging*, and tagging may occur through *automatic* (via a **dictionary** for example) or *human* coding

Machine learning algorithms, as we will see, can be considered as supervised classification methods

Classification methods

Classification methods can however also be used to **discover new ways** of organizing texts

Unsupervised classification methods are a class of methods that “**learn**” underlying features of text without explicitly imposing categories of interest (as it happens with supervised methods)

They use modeling assumptions and properties of the texts to estimate a **set of categories** and simultaneously **assign documents (or parts of documents)** to those categories

Therefore such models *infer* rather than *assume* the content of the categories under study

Back to validation

Because text analysis methods are necessarily incorrect models of language (remember!), the output always necessitates careful validation

For **supervised classification methods**, this requires demonstrating that the classification from machines replicates hand-coding

For **unsupervised classification and scaling methods**, this requires validating that the measures produced correspond with the concepts claimed

Classification methods

Supervised and **unsupervised methods** are different models with different objectives

If there are **predetermined categories** and documents that need to be placed in those categories, then use a supervised learning method!

If, however, researchers approach a problem without a **predetermined categorization scheme**, unsupervised methods can be useful. Supervised methods will never contribute a new coding scheme by definition!

Classification methods

Far from being competitors, supervised and unsupervised methods can *then* be productively viewed as **complementary methods**, particularly for new projects

For example, the categories of interest in a new corpus can be unclear or could benefit from extensive exploration of the data. In this case, unsupervised methods provide insights into classifications that would be difficult to obtain without guidance

Once the unsupervised method is fit, supervised learning methods can be used to validate or generalize the findings

Classification methods

Among the unsupervised classification methods, we can have…

Single membership models: these technique are designed to partition the set of documents into k non overlapping groups (clusters) in a way that maximizes the difference between groups and minimizes the difference within them, i.e., into homogenous groups according to some notion of **distance** among them

Classification methods

More in details…

….given a **dissimilarity measure** *d* among the documents (for example, Euclidean distance in a multi-dimensional features space), *clustering* algorithms proceed by grouping (*agglomerative* methods) or splitting (*dissociative* methods) subsequently the whole set of data according to *d*

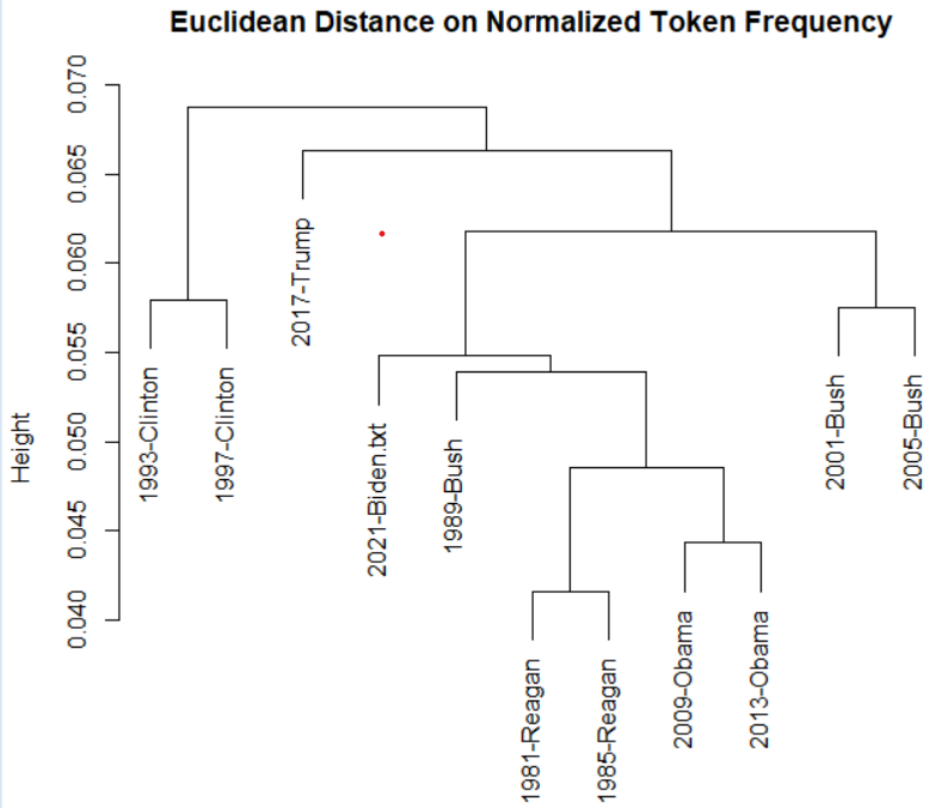
If this procedure is sequential, the method is called *hierarchical*

Classification methods

An example from the Inaugural Speeches by US Presidents corpus post 1980

The hierarchical agglomerative cluster algorithm works as follows:

1) Put each document in its own cluster
2) Identify the closest two clusters by considering (in this case) their squared Euclidean distance in the features space and combine them into one cluster
3) Then each new aggregation occurs either forming a new group of two units, or aggregating a unit to the closets group already formed or aggregating two distinct groups till all the documents are in a single cluster

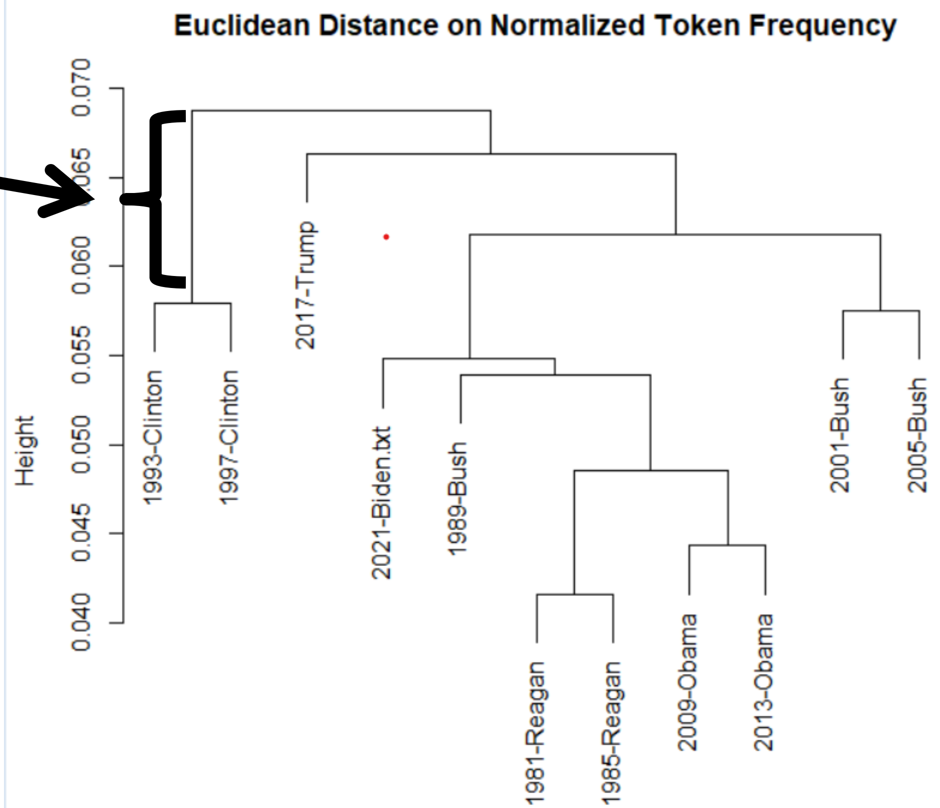


Classification methods

An example from the Inaugural Speeches by US Presidents corpus

In the **dendrogram**, long vertical lines indicate more distinct separation between the groups, while short vertical bars show observations that are all close to each other

The script to replicate this figure is available on-line!



Classification methods

Given the lack of input from the researcher, the estimates produced require a stronger effort of validation from the researcher, in the sense of interpreting whether these clusters are truly meaningful or not

There is no standard procedure to validate results: both qualitative and quantitative tests may be performed

One possibility in the previous examples could be comparing the estimates with existing literature and experts' judgement

Classification methods

The main limit of the **single membership model** approach is that it operates from an assumption that each document must belong to a single category and that categories do not overlap

This setting could result as too restrictive when classifying more complex documents, such as political speeches

In this case, each politicians' speech is likely to deal with a **variety of categories**

Classification methods

Mixed membership models (aka, **topic models**) assume precisely that each document is a mixture of categories (**topics**), meaning that a single document can be composed of multiple categories

This is reasonable after all!

Classification methods

To understand topic models, we need first of all starting with a better understanding of what we mean by “**topic**”

Substantively, topics are **distinct concepts**

In congressional speech, one topic may convey attention to America’s involvement in Afghanistan, with a **high probability attached to words** like troop, war, Taliban, and Afghanistan

A second topic may discuss the health-care debate, regularly using words like health, care, reform, and insurance

Statistically, a topic is defined as a (multinomial) **distribution over the words in the vocabulary of the corpus**

Classification methods

How to estimate a topic (which, remember, is **learned & discovered** rather than **assumed** by the researcher)?

We can observe **only documents and words**, **not topics** – the latter are part of the hidden (or latent) structure of documents

Still, our aim is to infer precisely the latent topic structure given the words and document

For solving this riddle, models use **the patterns of words co-occurrence within and across documents**

Classification methods

To this aim, we can for example taking advantage of the Latent Dirichlet Allocation (LDA) model. Why LDA?

Latent: topics that document consists of are unknown, but they are believed to be present as the text is generated based on those topics

Dirichlet: Dirichlet distribution is the multivariate generalisation of the Beta distribution. In the context of topic modeling, the Dirichlet is the distribution of topics in documents and distribution of words in the topic

Allocation: once we have the Dirichlet distribution, we will allocate topics to the documents and words of the document to topics

Classification methods

The basic assumption behind LDA is that each of the documents in a corpus consists of a **mixture of topics** (by “mixture” in this context we mean a set of positive values that sum to one), with **each word** within a given document belonging to **exactly one topic** (however, if a word **appears twice** in a document, each word may be assigned to different topics)

LDA *also* assume that any given topic will have a **high probability of generating certain words** and a **low probability** of generating other words as it is normally the case with real-world documents

Classification methods

As a result, each document can be represented as a **vector of proportions** that denote what **fraction of the words belongs to each topic**

Documents, then, are a **probability distribution over topics**

In this sense, a whole document may be “classified” into a given topic, but more accurately portions of documents are classified into topics across the entire corpus

Note the difference! In single membership models, on the contrary, **each document is restricted to only one topic (i.e., group)**, so all words within it are generated from the same distribution

Classification methods

How LDA works?

LDA “recreates” the documents in the corpus by adjusting the relative importance of topics in documents and words in topics **iteratively**, that is…

…given a corpus, LDA **backtracks** and tries to figure out what topics (and which words in each topic) would create the documents included in the corpus in the first place!

Classification methods

Let’s suppose you have *N* documents in your corpus and the total number of words (features) in your DfM is *W*

For example N=2 and W=5

	fish	eat	vegetables	milk	kitten
D1	2	2	1	0	0
D2	2	0	0	1	2

Classification methods

No **human input is required** to fit the topics besides a document-feature matrix, with *one critical exception*: the **number of topics** must be decided in advance

In fitting and interpreting topic models, therefore, a core concern is choosing the **“correct” number of topics**. There are statistical measures in this respect that you can take advantage of, but a better measure is often the **interpretability** of the topics as we will discuss (be back on this later on)

Suppose you select K (i.e., # topics) = 2

The assumed **data generating process** for each document in our corpus is as follows

Classification methods

1. Choose θ_i~DIRICHLET(α)

where:

θ_i=topic distribution for document *i*

α=parameter of Dirichlet prior on distribution of topics over docs that governs what the distribution of topics is for all the documents in the corpus looks like. A low value of **alpha** will assign fewer topics to each document whereas a high value of alpha will have the opposite effect

θ_i is a **topic mixture** drawn for the document *d* according to a Dirichlet distribution over the fixed set of K topics. If K=2, for example, θ_{ik} can be something like 0.3 for topic 1, i.e., 30% of the words in document *i* refers to topic 1; and 0.7 for topic 2, i.e., 70% of the words in document *i* refers to topic 2

As a result of this first draw, we create a new matrix

Classification methods

In matrices: LDA splits the original DfM of our corpus into two lower dimensions matrices (an example with *K*=2, d=2 and w=4)

	w1	w2	w3	w4
d1	0	2	3	1
d2	2	0	2	4

	k1	k2
d1	??	??
d2	??	??

N = total number of documents (i)
K = total number of topics (k)
M = the vocabulary size (words: m)

θ = **document-topics matrix** with dimension (N, K) where θ_{ik} corresponds to the probability that document i belongs to topic k

Of course the sum of the topic proportions across all topics for a given document is 1

Instead of ?? we have in real world-case some values

Classification methods

2. Choose $\beta_k \sim$DIRICHLET(δ)

where:

β_k=word distribution for topic *k* over all the documents (i.e., the probability of a word occurring in a given topic)

δ= parameter of Dirichlet prior on distribution of words over topics that governs what the distribution of words in each topic looks like. A low value of **delta** will use fewer words to model a topic whereas a high value will use more words, thus making topics more similar between them

As a result of this this draw, we create a second new matrix

Classification methods

In matrices: LDA splits the original DfM of our corpus into two lower dimensions matrices (an example with *K*=2, d=2 and w=4)

	w1	w2	w3	w4
d1	0	2	3	1
d2	2	0	2	4

N = total number of documents (i)
K = total number of topics (k)
M = the vocabulary size (words: m)

	w1	w2	w3	w4
k1	??	??	??	??
k2	??	??	??	??

β= **topic-terms matrix** with dimension (K, M) where β_{kw} corresponds to the probability that word m belongs to topic k

Instead of ?? we have in real world-case some values

Of course the sum of the topic probabilities for a word, across all topics, is 1

Classification methods

In matrices: LDA splits the original DfM of our corpus into two lower dimensions matrices (an example with *K*=2, d=2 and w=4)

	w1	w2	w3	w4
d1	0	2	3	1
d2	2	0	2	4

	k1	k2
d1	??	??
d2	??	??

N = total number of documents (i)
K = total number of topics (k)
M = the vocabulary size (words: m)

	w1	w2	w3	w4
k1	??	??	??	??
k2	??	??	??	??

θ = **document-topics matrix**

β= **topic-terms**

Classification methods

3. Choose a topic z ~Multinomial(θ_i)

In words: randomly choose a topic from the distribution of topics in document *i* based on their assigned values. In the previous example, let's say we choose Topic 1. Then…

- Choose a word w_i~Multinomial$\left(\beta_{i,k=z}\right)$

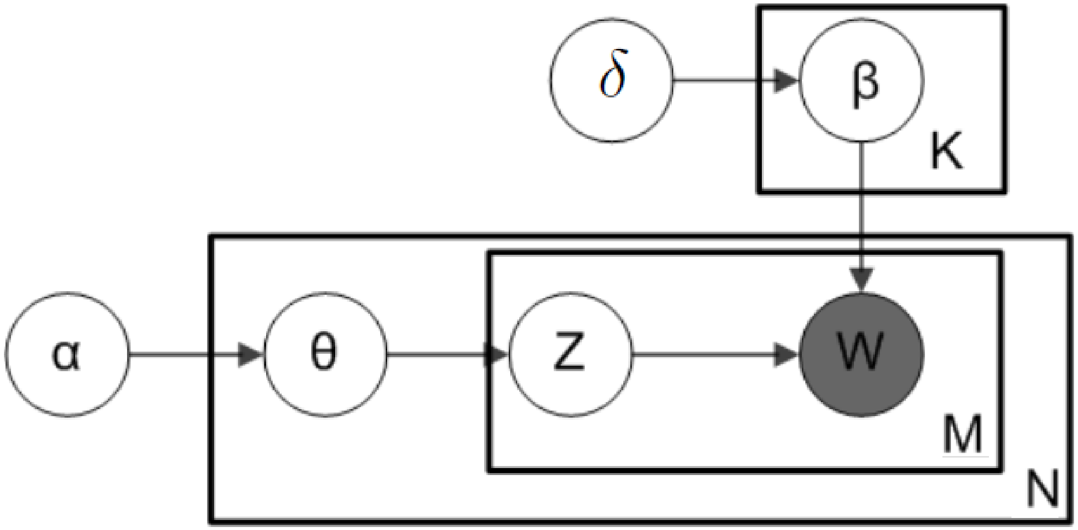
In words: based on the distribution of words for the chosen topic, go through document *i* and assign word *w* in the document to topic *z*

- Repeat this step for each word *w* in document *I*

That is: suppose you extract Topic 1 for document *i*, and that Topic 1 according to step 1 (i.e., θ_i) has assigned 20% of words in document *i*; then you extract randomly from the β_1 a word, and you look through document *i* for that word; you keep doing it till you assign 20% of words of document *i* to Topic 1; then you extract the next Topic for document *i*, etc.

Classification methods

Plate notation for a Topic Model



Where: N = total numer of documents; M = total number of words in each document; θ = document-topics matrix; β= topic-terms matrix; z is the topic for the n-th word in document *i*; w is the specific word (the only thing we can observe in the corpus)

Classification methods

Of course, if our initial guess of the values for the *document-topics matrix* and to*pic-terms matrix* is incorrect, then the actual data that we observe **will be very unlikely** under our assumed values and data generating process

Classification methods

For example, let's say we have the following document D1 :

"*Donald Trump has won the 2016 US Presidential Elections in a surprising way*"

…and let's say we assign to D1 high values (i.e., weights) to topic T1 which has high values (i.e., weights) for words like war, military, Iraq etc.

From this we can infer that given our assumption of how data is generated, it is very unlikely that T1 belongs to D1 or these words belongs to T1

Therefore, what to do? We have to maximize the likelihood of our data *given* the two previous matrices (*document-topics matrix* and to*pic-terms matrix*)

Classification methods

To identify the correct values/weights LDA uses a process known as *Gibbs sampling*

Gibbs sampling is an algorithm for successively sampling conditional distributions of variables, whose distribution over states converges to the true distribution in the long run

It works by performing a random walk in such a way that reflects the characteristics of a desired distribution (in our case, the Dirichlet one). The starting point of the walk is chosen at random

How does the Gibbs sampling work in our Topic Model scenario?

Classification methods

After having defined the number of topics *K* to discover, we run a first assignment (i.e., we complete the three steps of the LDA above once)

This first (random) assignment already gives you both topic representations of all the documents (θ_{ik}) and word distributions (β_{kw}) of all the topics

Classification methods

Let's go back to our example with 2 documents:

	Document X		Document Y
	Fish		Fish
	Fish		Fish
	Eat		Milk
	Eat		Kitten
	Vegetables		Kitten

We selected at the beginning K=2

Classification methods

Step 1 to Step 3 – **first random assignment**

	fish	eat	vegetables	milk	kitten
D1	2	2	1	0	0
D2	2	0	0	1	2

	K1	K2
D1	1	0
D2	0.4	0.6

Document-topics matrix (first assignment) - Step 1

	fish	eat	vegetables	milk	kitten
K1	0.429	0.286	0.143	0.143	0.143
K2	0.333	0	0	0	0.666

Topic-terms matrix (first assignment) – Step 2

Step 3 – suppose you draw K2 for D2. You know from Step 1 that 60% of words should be devoted to K2 – i.e., 3 out of 5 words from D2, and 40% to K1. Then you randomly assign the words included in D2 to K2, knowing the % of Step 2. For example….

Classification methods

Step 1 to Step 3 – **first random assignment**

	fish	eat	vegetables	milk	kitten
D1	2	2	1	0	0
D2	2	0	0	1	2

	K1	K2
D1	1	0
D2	0.4	0.6

Document-topics matrix (first assignment)

	fish	eat	vegetables	milk	kitten
K1	0.429	0.286	0.143	0.143	0.143
K2	0.333	0	0	0	0.666

Topic-terms matrix (first assignment)

	fish	eat	vegetables	milk	kitten
D1	2 (K1)	2 (K1)	1 (K1)	0	0
D2	2 (1 to K1 & 1 to K2)	0	0	1 (K1)	2 (K2)

Classification methods

Of course, the values obtained via the first assignment are not necessarily very good ones

So to improve on them, **both values are updated**. How?

We will slowly change the values as reported in our two new matrices and get to an answer that maximizes the likelihood of the data that we have

We will do this *on word by word basis* by changing the topic assignment of one single word at the time

Classification methods

When doing it, **we are assuming that all topic assignments** except for the current word in question, **are correct** (i.e., we assume that we don’t know the topic assignment of the given word but **we do know the assignment of all other words in the text**)…

…and then we **update** the assignment of the current word using our model of how documents are generated (i.e., we try to infer what topic will be assigned to this word)

Classification methods

More in details, for each document *i*…

….go through each word *m* in *i*…

...and for each topic *k*, compute two things:

1) p(topic k | document i) = the proportion of words in document i that are currently assigned to topic k, i.e., **how prevalent are topics in the document?**
2) p(word m | topic k) = the proportion of assignments to topic k over all documents that come from this word m, i.e., **how prevalent is that word across topics?**

What we mean by that? Let’s go back to our previous example

Classification methods

These are the values according to our first random assignment

	fish	eat	vegetables	milk	kitten
D1	2	2	1	0	0
D2	2	0	0	1	2

	K1	K2
D1	1	0
D2	0.4	0.6

Document-topics matrix (first assignment)

	fish	eat	vegetables	milk	kitten
K1	0.429	0.286	0.143	0.143	0.143
K2	0.333	0	0	0	0.666

Topic-terms matrix (first assignment)

	fish	eat	vegetables	milk	kitten
D1	2 (K1)	2 (K1)	1 (K1)	0	0
D2	2 (1 to K1 & 1 to K2)	0	0	1 (K1)	2 (K2)

Classification methods

That is….

	Document X		Document Y
K1	Fish	K2	Fish
K1	Fish	K1	Fish
K1	Eat	K1	Milk
K1	Eat	K2	Kitten
K1	Vegetables	K2	Kitten

Classification methods

Imagine now that we are now checking the possible **new topic assignment** for the word “fish” in Doc Y. **Assuming that all topic assignments** except for the current word in question, **are correct,** changing the topic assignment of word “fish” in Doc Y from topic K2 to topic K1, is going to improve the model or not?

	Document X		Document Y
K1	Fish	???	Fish
K1	Fish	K1	Fish
K1	Eat	K1	Milk
K1	Eat	K2	Kitten
K1	Vegetables	K2	Kitten

Classification methods

To answer this question we need to compare therefore two conditional probabilities:

*p(topic K1 | document Y)*p(word Fish | topic K1)*

with

*p(topic K2 | document Y)*p(word Fish | topic K2)*

If the former probability is larger than the second, then we will assign word Fish to topic K1; otherwise we will keep it in topic K2

According to our generative model, this is essentially the **probability that topic k generated word m** (in our case: the probability that topic K1 – or topic K2 – generated the word Fish)

Classification methods

How prevalent are topics in the document? i.e., p(topic k | document i)? Since the words in Doc Y are assigned to Topic K1 and Topic K2 in a 50-50 ratio, the remaining "fish" word seems equally likely to be about either topic

	Document X		Document Y
K1	Fish	???	Fish
K1	Fish	K1	Fish
K1	Eat	K1	Milk
K1	Eat	K2	Kitten
K1	Vegetables	K2	Kitten

Classification methods

How prevalent is that word across topics? i.e., p(word m | topic k)? The "fish" words across both documents appears nearly half of the time in Topic K1 words (3/7), but 0% among Topic K2 words

	Document X		Document Y
K1	**Fish**	???	Fish
K1	**Fish**	K1	**Fish**
K1	Eat	K1	Milk
K1	Eat	K2	Kitten
K1	Vegetables	K2	Kitten

Classification methods

As a conclusion from the two criteria (i.e., by *multiplying* the two previous probabilities), we would move the “fish” word of Doc Y to Topic K1

In fact: p(topic K1 | document Y)*p(word Fish | topic K1) > p(topic K2 | document Y)*p(word Fish | topic K2)

That is, 0.5*.43>0.5*0!

Of course, thanks to this change, the initial values in the Document-topics matrix and in the Topic-terms matrix will change accordingly compared to the first assignment

Classification methods

Going back to our example

	Document X		Document Y
	Fish		Fish
	Fish		Fish
	Eat		Milk
	Eat		Kitten
	Vegetables		Kitten

	fish	eat	vegetables	milk	kitten
D1	2	2	1	0	0
D2	2	0	0	1	2

	K1	K2
D1	?	?
D2	?	?

Document-topics matrix

	fish	eat	vegetables	milk	kitten
K1	?	?	?	?	?
K2	?	?	?	?	?

Topic-terms matrix

Classification methods

INITIAL ASSIGNMENT

	fish	eat	vegetables	milk	kitten
D1	2	2	1	0	0
D2	2	0	0	1	2

	K1	K2
D1	1	0
D2	0.4	0.6

Document-topics matrix (first assignment)

	fish	eat	vegetables	milk	kitten
K1	0.429	0.286	0.143	0.143	0.143
K2	0.333	0	0	0	0.666

Topic-terms matrix (first assignment)

	fish	eat	vegetables	milk	kitten
D1	2 (K1)	2 (K1)	1 (K1)	0	0
D2	2 (1 to K1 & 1 to K2)	0	0	1 (K1)	2 (K2)

Classification methods

UPDATED ASSIGNMENT

	fish	eat	vegetables	milk	kitten
D1	2	2	1	0	0
D2	2	0	0	1	2

	fish	eat	vegetables	milk	kitten
D1	2 (K1)	2 (K1)	1 (K1)	0	0
D2	2 (K2)	0	0	1 (K1)	2 (K2)

In the initial assignment it was different!

	K1	K2
D1	1	0
D2	0.6	0.4

Document-topics matrix

	fish	eat	vegetables	milk	kitten
K1	0.5	0.25	0.125	0.125	0
K2	0	0	0	0	1

Topic-terms matrix

Classification methods

By following this procedure, we (eventually) reassign any given *m* to a new topic, where topic *k* is chosen with probability p(topic k | document i) * p(word m | topic k)

After repeating the previous step a **large number of times**, you'll eventually reach a roughly steady state where your assignments (the document topic and topic term distributions) are pretty good

This is the **convergence point** of the Gibbs sampling algorithm

Classification methods

Once the convergent point is reached, use the obtained assignments to estimate the:

1. **Document-topic proportions** (by counting the proportion of words assigned to each topic *within* that document)
2. **Topic-word proportions** (by counting the proportion of words assigned to each topic overall, i.e., *across documents*)

Classification methods

The quantities of interest from a Topic Model:

QOI: Document-Topic Proportions

- Level of Analysis: Document
- Part of the Model: θ
- Description: Proportion of words in a given document about each topic.
- Example Use: Can be used to identify the documents that devote the highest or lowest proportion of words to a particular topic. Those with the highest proportion of words are often called "exemplar" documents and can be used to validate that the topic has the meaning the analyst assigns to it.

Classification methods

The quantities of interest from a Topic Model:

QOI: Topic-Word Proportions

- Level of Analysis: Corpus
- Part of the Model: κ, β
- Description: Probability of observing each word in the vocabulary under a given topic.
- Example Use: The top 10 most probable words under a given topic are often used as a summary of the topic's content and help inform the user-generated label.

Classification methods

Which are the main challenges of a topic model?

First of all we need to give an answer to the following question: *How many topics?*

The analyst **must choose the number of topics**. There is no “right” answer to this choice

Therefore, the choice will be dependent both on the nature of the documents under study and the goals of the analysis

Classification methods

Largely, the answer will be related to the **semantic meaning** of the topics extracted

The researcher is indeed tasked with selecting a number of topics and confirming that those recovered are **substantively meaningful**

For example, if you extract 15 topics, and you are able to give a clear and unambiguous interpretation of those topics, then 15 is a good number for you!

That is, choose K based on "substantive fit" (as well as according to your main research interest! If you are mainly interest in detecting the change over time of the topic "immigration" in your corpus, when you are able to "discover" such topic in an unambiguous way among the K topics you extracted, stop there!)

Classification methods

Examining the terms with highest probabilities of belonging to each topic and reading the documents with highest probabilities of belonging to it gives the researcher a sense of the **substantive meaning** of each topic

	fish	eat	vegetables	milk	kitten
K1	0.5	0.25	0.125	0.125	0
K2	0	0	0	0	1

In our example: K1 is related to “food” and K2 to “animals”?

Classification methods

Given that is practically impossible to guess the exact number of topics in the corpus (although, **empirically**, **tests** have been introduced in the literature - and we will see them), a good practice is beginning with a **wider number of topics** rather than a potentially too narrow one

Then a researcher should settled on a specification of *K* lower that the initial one when she found that at higher specifications, substantively-meaningful topics were being divided up in ways that were less amenable to testing her hypotheses

In practice the precise choice of topics contains a degree of **arbitrariness**, and often to recover interpretable topics, some extra ones are also generated that are not readily interpretable

Classification methods

But therefore, finding a "*correct number of topics*" is mainly related to our ability to clearly understand the semantic meaning of each single topic extracted

And this is the **second main challenge of a topic model**!

But which are the main qualities of a semantically interpretable topic?

Classification methods

A **semantically interpretable topic** has two qualities:

(a) it is *coherent/cohesive* in the sense that high-probability words for the topic tend to co-occur (i.e., *do top words of one topic tend to co-occur across documents*?)

Therefore semantic coherence is a property of the “within topics”

Classification methods

Semantic coherence **however** only addresses whether a topic is internally consistent (i.e., it checks if we are evaluating a well-defined concept)

It **does not penalize topics that are alike**

This could be a problem!

Classification methods

A **semantically interpretable topic** has two qualities

(b) it is *exclusive* in the sense that the top words for that topic are unlikely to appear within top words of other topics (i.e., *are the top words of one topic different from the top words of other topics*?): if words with high probability under topic *k* have low probabilities under other topics, then we say that topic *k* is exclusive

Therefore semantic exclusivity is a property of the "between topics"

Classification methods

A topic that is both *cohesive and exclusive* is more likely to be **semantically useful**

We will discuss in the lab-session how looking for precisely semantically useful topics also help us in our quest of the «correct number of topics»

Classification methods

A non-technical resume

Topic models provide a parametric model describing the relationship between **clusters of co-occurring words representing “topics”** and their relationship to documents which contain them in **relative proportions**

By estimating the parameters of this model, it is possible to **recover these topics** (and the words that they comprise) and to estimate the degree to which documents pertain to each topic

The **estimated topics are unlabelled**, so a human must assign these labels by interpreting the content of the words most highly associated with each topic, perhaps assisted by contextual information

STM and R

install.packages("ldatuning", repos='http://cran.us.r-project.org')

install.packages("topicomodels", repos='http://cran.us.r-project.org')

install.packages("lubridate", repos='http://cran.us.r-project.org')

install.packages("topicdoc", repos='http://cran.us.r-project.org')