

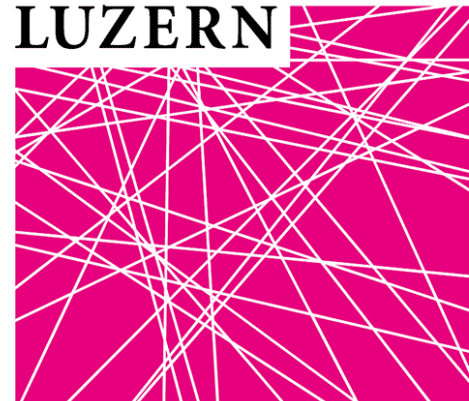
# *Big Data Analytics*

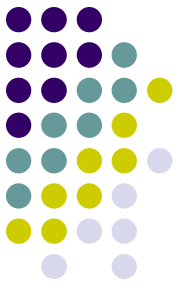
Lecture 2/A

Unsupervised classification methods:  
the Topic Model



UNIVERSITÄT  
LUZERN





# Reference

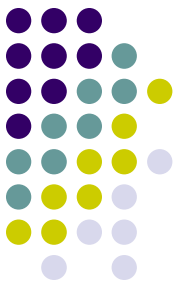
- ✓ Roberts, Margaret E., Brandon M. Stewart, Dustin Tingley, Christopher Luca, Jetson Leder-Luis, Shana Kushner Gadarian, Bethany Albertson, David G. Rand. 2014. Structural Topic Models for Open-Ended Survey Response, *American Journal of Political Science*, 58(4), 1064-1082

# Classification methods



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





# Classification methods

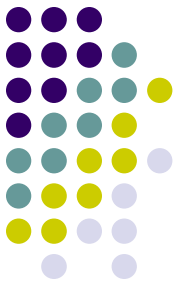
Sometimes researchers **know the categories** beforehand

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* (via **Machine Learning algorithms**)

In both cases, supervised classification algorithms require some kind of **a-priori information** by the researcher to produce estimates

# Classification methods



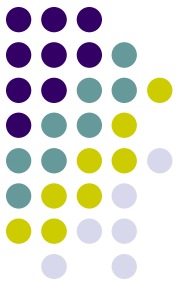
Classification methods can however also be used to **discover new ways** of organizing texts (this is why such way of classification is called “**unsupervised**”)

**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

# 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, on the contrary, 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



However remember: far from being competitors, supervised and unsupervised methods can also 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

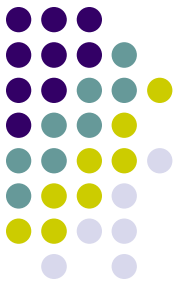
Among the unsupervised classification methods, we can have...

**Single membership models** (such as clustering algorithms): in this case, each document is assigned entirely to a single latent category

This assumption could however result as too restrictive (and not that much reasonable) in many instances

Quite often, in fact, authors in a given text deal with a **variety of categories**

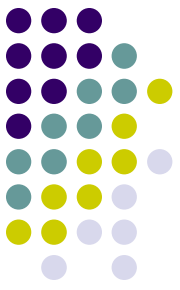




# Classification methods

**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

Given their higher flexibility, we are going to focus precisely on this latter class of unsupervised classification models



# 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**



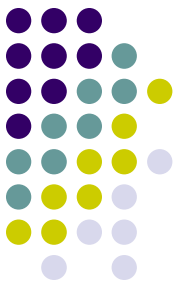
# Classification methods

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**. But how exactly?



# Classification methods

One possibility in this respect is to take 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 generalization of the Beta distribution. In the context of topic modeling, the Dirichlet refers to the *distribution of topics in documents* and the *distribution of words in the topic*

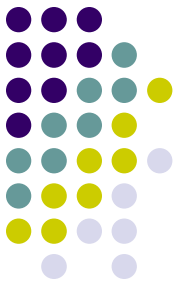
*Allocation:* once we have the Dirichlet distribution, we will allocate topics to the documents and words of the document to topics



# 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



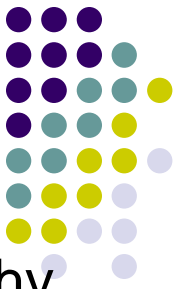
# Classification methods

Suppose you have the following set of sentences:

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

LDA is a way of automatically discovering topics that these sentences contain

# Classification methods



For example, given these sentences and asked for 2 topics (why 2? More on this later on), 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...

Topic B: 20% chinchillas, 20% kittens, 20% cute, 15%...

You could infer that topic A is a topic about food, and topic B is a topic about cute animals for example

But LDA does not explicitly identify topics in this manner! All it can do is tell you the probability that specific words are associated with the topic. Then it's up to you to give a substantial interpretation to these topics. **VALIDATION** step!



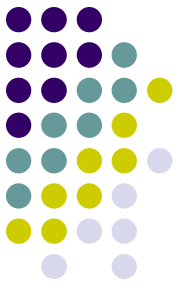


# Classification methods

But how LDA works in practice?

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  $N=2$  (document X and document Y) and  $W=5$

	fish	eat	vegetables	milk	kitten
X	2	2	1	0	0
Y	2	0	0	1	2



# Classification methods

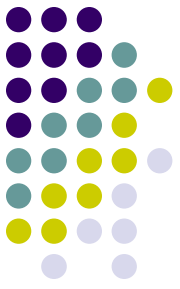
No **human input is required** to fit the topics besides a document-feature matrix, with *one critical exception*: the **number of topics** must be decided in advance

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  $\theta_i \sim \text{DIRICHLET}(\alpha)$

where:

$\theta_i$ =topic distribution for document  $i$  extracted according to a Dirichlet distribution

$\alpha$ =parameter of the Dirichlet distribution, i.e., the prior on the distribution of topics over docs. A low value of **alpha** will assign fewer topics to each document whereas a high value of alpha will have the opposite effect

$\theta_i$  is a **topic mixture** drawn for the document  $d$  over the fixed set of  $K$  topics. If  $K=2$ , for example,  $\theta_{ik}$  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  $K=2$ ,  $N=2$  and  $W=4$ )

	w1	w2	w3	w4
X	0	2	3	1
Y	2	0	2	4



	k1	k2
X	??	??
Y	??	??

$N$  = total number of documents ( $i$ )  
 $K$  = total number of topics ( $k$ )  
 $W$  = the vocabulary size (words:  $w$ )

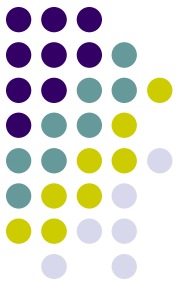


$\theta$  = **document-topics matrix** with dimension ( $N$ ,  $K$ ) where  $\theta_{ik}$  corresponds to the probability that document  $i$  belongs to topic  $k$

By definition the sum of the topic proportions across all topics for a given document is 1

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

# Classification methods



2. Choose  $\beta_k \sim \text{DIRICHLET}(\delta)$

where:

$\beta_k$  = word distribution for topic  $k$  over all the documents (i.e., the probability of a word occurring in a given topic) extracted according to a Dirichlet distribution

$\delta$  = parameter of the Dirichlet distribution, i.e., the prior on the distribution of words in each topic. 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

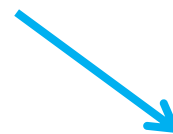
# Classification methods



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

	w1	w2	w3	w4
X	0	2	3	1
Y	2	0	2	4

$N$  = total number of documents (i)  
 $K$  = total number of topics (k)  
 $W$  = the vocabulary size (words: w)



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

$\beta$  = **topic-terms matrix** with dimension  $(K, W)$  where  $\beta_{kw}$  corresponds to the probability that word  $w$  belongs to topic  $k$

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

By definition the sum of the topic probabilities, across all words, is 1

# Classification methods



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

	w1	w2	w3	w4
X	0	2	3	1
Y	2	0	2	4

	k1	k2
X	??	??
Y	??	??

$N$  = total number of documents (i)  
 $K$  = total number of topics (k)  
 $W$  = the vocabulary size (words: w)

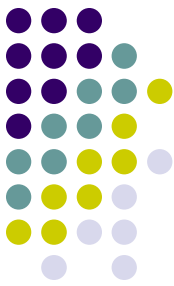
	w1	w2	w3	w4
k1	??	??	??	??
k2	??	??	??	??

$\theta$  = document-topics matrix

$\beta$  = topic-terms matrix



# Classification methods



3. Choose a topic  $z \sim \text{Multinomial}(\theta_i)$

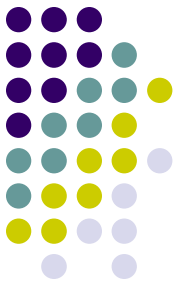
*In words:* randomly choose a topic from the distribution of topics in document  $i$  based on their assigned values. In the previous example, let's say we choose Topic 1. Then...

- Choose a word  $w_i \sim \text{Multinomial}(\beta_{i,k=z})$

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

- Repeat this step for each word  $w$  that forms document  $i$

That is: suppose you extract Topic 1 for document  $i$ , and that Topic 1 according to step 1 (i.e.,  $\theta_i$ ) has assigned 30% of words in document  $i$ ; and suppose that in document  $i$  you have 10 words. Then you extract randomly from the  $\beta_1$  a word, and you assign that word to document  $i$ ; you keep doing it till you assign 30% of words from  $\beta_1$  to document  $i$  (i.e., 3 words); then you extract the next Topic for document  $i$ , etc.



# 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

Suppose we select at the beginning  $K=2$

# Classification methods



## Step 1 to Step 3 – first random assignment

	fish	eat	vegetables	milk	kitten
X	2	2	1	0	0
Y	2	0	0	1	2



	K1	K2
X	0.6	0.4
Y	0.4	0.6

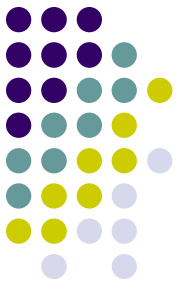
	fish	eat	vegetables	milk	kitten
K1	0.2	0.4	0.1	0.2	0.1
K2	0.3	0.1	0.1	0.1	0.4

Document-topics matrix (first assignment) - Step 1

Topic-terms matrix (first assignment) – Step 2



Step 3 – suppose you draw K1 for document X. You know from Step 1 that 60% of words should be devoted to K1 – i.e., 3 out of 5 words from X, and 40% to K2. We can now generate the document X under the LDA model defined above, and the same thing for document Y



# Classification methods

That is...when generating the document  $X$  (consisting of 5 words), LDA does the following:

Decide that  $X$  will be .6 about  $K1$  (i.e., 3 out of 5 words) and .4 about  $K2$  (i.e., 2 out of 5 words)

Pick the first word (out of 5) to come from the beta distribution of  $K1$  topic, which then gives you the word “eat”

Pick the second word to come from  $K1$ , which gives you “eat”

Pick the third word to come from  $K1$ , giving you “milk”

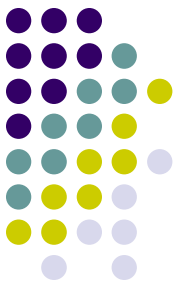
Pick the fourth word to come from  $K2$ , giving you “fish”

Pick the fifth word to come from  $K2$ , giving you “kitten”

So the document  $X$  generated under the LDA model will be “eat eat milk fish kitten” (in a bag-of-words approach)

LDA then does the same for document  $Y$  as well

# Classification methods



Step 1 to Step 3 – first random assignment

	fish	eat	vegetables	milk	kitten
X	2	2	1	0	0
Y	2	0	0	1	2



	K1	K2
X	0.6	0.4
Y	0.4	0.6

	fish	eat	vegetables	milk	kitten
K1	0.2	0.4	0.1	0.2	0.1
K2	0.3	0.1	0.1	0.1	0.4

Document-topics matrix (first assignment)

Topic-terms matrix (first assignment)



	fish	eat	vegetables	milk	kitten
X	1 (K2)	2 (K1)	0	1 (K1)	1 (K2)
Y	1 (K2) & 1 (K1)	0	1 (K1)	0	2 (K2)



# Classification methods

This first (random) assignment (by completing the three steps of the LDA above once) already gives you both topic representations of all the documents ( $\theta_j$ ) and word distributions ( $\beta_k$ ) of all the topics

That is, if our initial guess of the values for the *document-topics matrix* and *topic-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

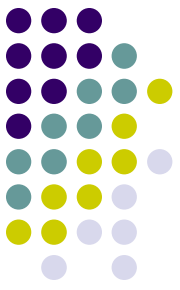
For example, let's say we have the following document D1 :

*“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

# Classification methods



Moreover, note that in our example we have assigned via the LDA to document  $X$  the word “milk” that is not included in the original document  $X$ ; and the same thing applies to the word “vegetables” for the document  $Y$

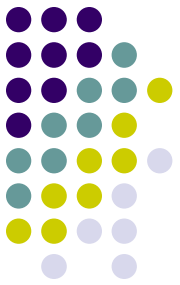
These are our “errors” (that we want to “minimize” by maximizing the likelihood of our data given the *document-topics matrix* and *topic-terms matrix*) in our attempt to backtrack the original documents from our step 1-3

But how to do that?

We have to **maximize the likelihood** of our data *given* the two previous matrices (*document-topics matrix* and *topic-terms matrix*)



# Classification methods



We will update the values in both the *document-topics matrix* and *topic-terms matrix*

But how?

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

And 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**)...

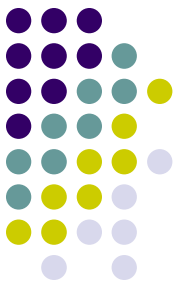
...and then we **update** the assignment of the current word using our model of how documents are generated (i.e., we try to infer what topic will be assigned to this word)



# Classification methods

To identify the correct values/weights LDA uses in particular 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



# Classification methods

More in details, for each document  $i...$

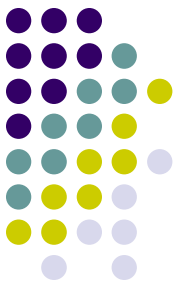
....go through each word  $m$  in  $i...$

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

- 1)  $p(\text{topic } k \mid \text{document } i) =$  the proportion of words in document  $i$  that are currently assigned to topic  $k$ , i.e., **how prevalent are topics in the document?**
- 2)  $p(\text{word } m \mid \text{topic } k) =$  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
X	2	2	1	0	0
Y	2	0	0	1	2



	K1	K2
X	0.6	0.4
Y	0.4	0.6

	fish	eat	vegetables	milk	kitten
K1	0.2	0.4	0.1	0.2	0.1
K2	0.3	0.1	0.1	0.1	0.4

Document-topics matrix (first assignment)

Topic-terms matrix (first assignment)



	fish	eat	vegetables	milk	kitten
X	1 (K2)	2 (K1)	0	1 (K1)	1 (K2)
Y	1 (K2) & 1 (K1)	0	1 (K1)	0	2 (K2)



# Classification methods

That is....

Document X	Topic	Document Y	Topic
fish	K2	fish	K1
eat	K1	fish	K2
eat	K1	vegetables	K1
milk	K1	kitten	K2
kitten	K2	kitten	K2



# 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 K1 to topic K2, is going to improve the model or not?

Document X	Topic	Document Y	Topic
fish	K2	fish	???
eat	K1	fish	K2
eat	K1	vegetables	K1
milk	K1	kitten	K2
kitten	K2	kitten	K2



# Classification methods

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

$$p(\text{topic } K1 \mid \text{document } Y) * p(\text{word } \textit{Fish} \mid \text{topic } K1)$$

with

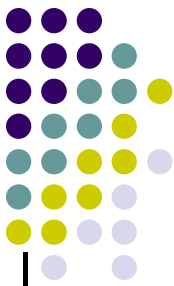
$$p(\text{topic } K2 \mid \text{document } Y) * p(\text{word } \textit{Fish} \mid \text{topic } K2)$$

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 *w*** (in our case: the probability that topic *K1* – or topic *K2* – generated the word *Fish*)



# Classification methods

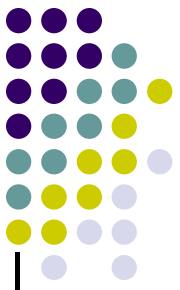


*How prevalent are topics in the document? i.e.,  $p(\text{topic } k \mid \text{document } i)$ ?*

Since the remaining words in Doc Y are assigned to Topic K2 and Topic K1 in a 3 to 1 ratio, the remaining “fish” word seems more likely to be about topic K2

Document X	Topic	Document Y	Topic
fish	K2	fish	???
eat	K1	fish	K2
eat	K1	vegetables	K1
milk	K1	kitten	K2
kitten	K2	kitten	K2

# Classification methods



*How prevalent is that word across topics? i.e.,  $p(\text{word } m \mid \text{topic } k)$ ?*

The “fish” words across both documents appears nearly half of the time in Topic K2 words (2/5), but 0% among Topic K1 words

Document X	Topic	Document Y	Topic
fish	K2	fish	???
eat	K1	fish	K2
eat	K1	vegetables	K1
milk	K1	kitten	K2
kitten	K2	kitten	K2



# Classification methods

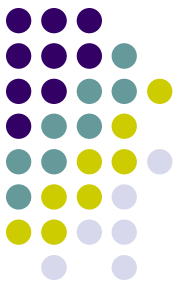
**As a conclusion from the two criteria** (i.e., by *multiplying* the two previous probabilities), we would move the “fish” word of Doc Y to Topic K2

In fact:  $p(\text{topic K2} \mid \text{document Y}) * p(\text{word Fish} \mid \text{topic K2}) > p(\text{topic K1} \mid \text{document Y}) * p(\text{word Fish} \mid \text{topic K1})$

That is,  $0.75 * 0.4 > 0.25 * 0!$

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: **INITIAL ASSIGNMENT**

	fish	eat	vegetables	milk	kitten
X	2	2	1	0	0
Y	2	0	0	1	2



	K1	K2
X	0.6	0.4
Y	0.4	0.6

	fish	eat	vegetables	milk	kitten
K1	0.2	0.4	0.1	0.2	0.1
K2	0.3	0.1	0.1	0.1	0.4

Document-topics matrix (first assignment)

Topic-terms matrix (first assignment)



	fish	eat	vegetables	milk	kitten
X	1 (K2)	2 (K1)	0	1 (K1)	1 (K2)
Y	1 (K2) & 1 (K1)	0	1 (K1)	0	2 (K2)

# Classification methods



## UPDATED ASSIGNMENT

	fish	eat	vegetables	milk	kitten
X	2	2	1	0	0
Y	2	0	0	1	2



	fish	eat	vegetables	milk	kitten
X	1 (K2)	2 (K1)	0	1 (K1)	1 (K2)
Y	2 (K2)	0	1 (K1)	0	2 (K2)



In the initial assignment it was different!



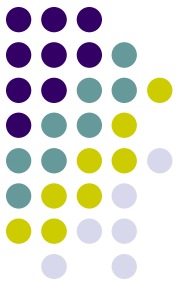
	K1	K2
X	0.6	0.4
Y	0.2	0.8

Document-topics matrix

	fish	eat	vegetables	milk	kitten
K1	0.0	0.5	0.25	0.25	0
K2	0.5	0	0	0	0.5

Topic-terms matrix

# Classification methods

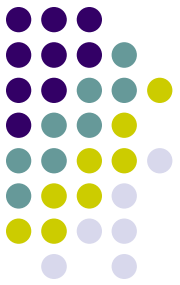


You keep doing that for each word in each of your document

After you have done it once, you have ended your first iteration of the LDA

By doing it, you have updated (and improved) the distribution of thetas for each single document as well as the vocabulary for each topic (i.e., the distribution of betas)

Now you can go back to steps 1-3 to recreate our documents according to LDA and the two new matrices (second iteration)...and so on...



# Classification methods

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 and you cannot improve anymore the **likelihood** of your data *given* the two previous matrices (*document-topics matrix* and *topic-terms matrix*)

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



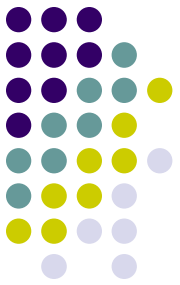
# Classification methods

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), i.e., the thetas!
2. **Topic-word proportions** (by counting the proportion of words assigned to each topic overall, i.e., *across documents*), i.e., the betas!



# 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

# Classification methods



## A non-technical resume

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

## Back to validation!

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

Not an easy exercise! How to do that? More on this in a moment...



# 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

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

# Classification methods



© MARK ANDERSON, WWW.ANDERTOONS.COM



"Trust me on this."

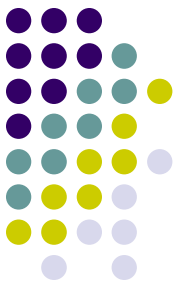
# Classification methods



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** (you have to validate them!!!)

# Classification methods

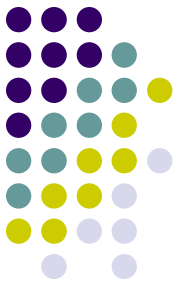


For example, if you extract 15 topics, and you are able to give a clear and unambiguous interpretation of those topics, then 15 is a good number for you!

That is, always choose  $K$  based on “**substantive fit**”...

...as well as according to your main research interest! If you are mainly interested 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	0.5	0.25	0.25	0
K2	0.5	0	0	0	0.5

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

# Classification methods

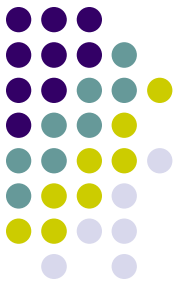


Given that it is practically impossible to guess the exact number of topics in the corpus (although, **empirically, tests** have been introduced in the literature - and we will see them), a good practice is beginning with a **wider number of topics** rather than a potentially too narrow one

Then a researcher should settle on a specification of  $K$  lower than 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



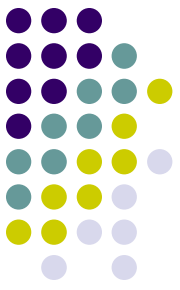


# Classification methods

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?*)

That is, we would like that when documents discuss about a given topic, they use more or less the same vocabulary

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



# Classification methods

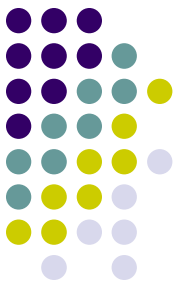
Semantic coherence **however** only addresses whether a topic is internally consistent (i.e., it checks if we are evaluating a well-defined concept)

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

For example, we could have two topics (topic 1 and 2), each of them with a high level of internal coherence (i.e., documents discussing about topic 1 and topic 2 use more or less always the same vocabulary in our corpus)...

....however such vocabulary could also be very similar across topic 1 and 2!

Therefore...



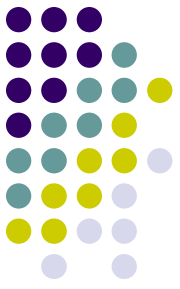
# Classification methods

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

That is, we would like that each topic is characterized by its own distinct vocabulary

Therefore semantic exclusivity is a property of the “between topics”



# Classification methods

In the previous example, the *lack of exclusiveness* between topics 1 and 2 would be a good sign that these two topics should be reduced to a single one (by reducing the number of topics you extract from the analysis!)

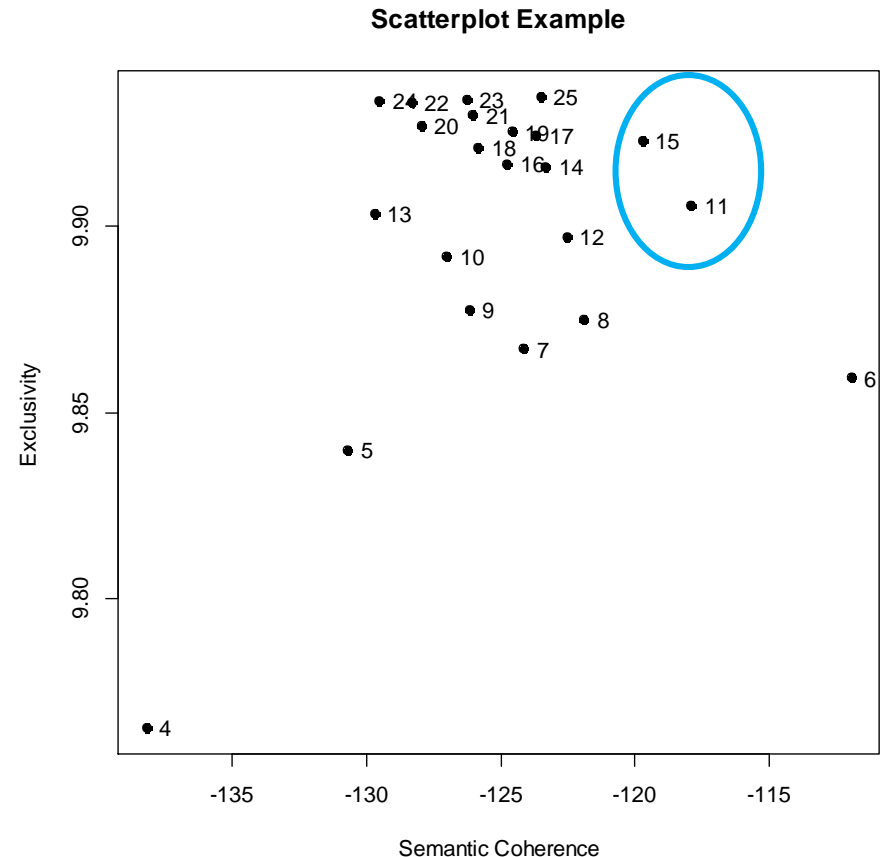
A topic that is both *cohesive and exclusive* is more likely to be **semantically useful**



# Classification methods

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

Basically, we should focus on those models that lie on the *semantic coherence-exclusivity 'frontier'*, that is, where no model strictly dominates another in terms of semantic coherence and exclusivity (i.e., models with average scores towards the upper right side of the plot)



# Classification methods



Alternative metrics for evaluating topic models are:

*Log-likelihood*: it measures how plausible model parameters of your Topic Model are, given the data you analyze. The *higher* the log-likelihood, the *better* the model

A similar metric is the *perplexity*: it is a statistical measure of how well a probability model predicts a sample

Typically the *Perplexity* is estimated on documents that *have not been used* for training a topic model. In this case we also talk about “*held-out log-likelihood*”



# Classification methods

More in details, perplexity assesses a topic model's ability to predict the words that appear in new documents (i.e., the held-out documents!) after having been trained on a training set

That is, do the words in the held-out set appear together as they would according to the topic-terms matrix probabilities as learned in the training-stage?

The *lower* the perplexity, the *better* the model





# Classification methods

Now suppose that you have found the highest (lowest) value of the log-likelihood (perplexity) at  $K=20$ . Do you stop there?

Once again no! This is not the end of your journey! You still have to validate the topics! And if you are not able, then, change the # topic!

Note that Perplexity is not always strongly correlated to human judgment

Chang, Jonathan, Jordan Boyd-Graber, Sean Gerrish, Chong Wang and David M. Blei (2009). Reading Tea Leaves: How Humans Interpret Topic Models. *NIPS*

Therefore, better focusing on coherence and exclusivity as above!