Reminder: Quiz 1 this Friday
A4/B4–3:10pm-4:30pm, Z4—6:30pm-7:50pm

- **Join on Zoom with your video on; we will take attendance**
- The quiz is **80** minutes long
- **You produce a Jupyter notebook and submit it on Canvas**
- Open notes, open internet, **closed to collaboration**
- You are responsible for making sure your laptop has a compute environment set up appropriately and has working internet
  - Check that you can run all lecture demos without issues
- Late exams will *not* be accepted
  - The Canvas submission system will close after 80 minutes; **we will only grade what is submitted to Canvas via the assignment submission system (NOT Canvas mailbox)**

More on Quiz 1

Coverage:

- Up to and including dimensionality reduction/manifold learning
- Clustering and topic modeling are *not* on Quiz 1; there is no need to study these for the quiz

A note on academic integrity:

- The top of the quiz will have a statement you will agree to regarding academic integrity; **if you do not sign it, we will not grade your quiz — no exceptions**

Violations to academic integrity will result in an F in the course =(

We want the quiz to be as fair as possible to all students. We realize that given the pandemic situation, there's only so much we can do.

At least you can take the quiz remotely
hopefully some place comfortable!

Like the beach

Reminder: Your Quiz 1 review session is tonight at 9pm-10:20pm

Carnegie Mellon University
Heinz College

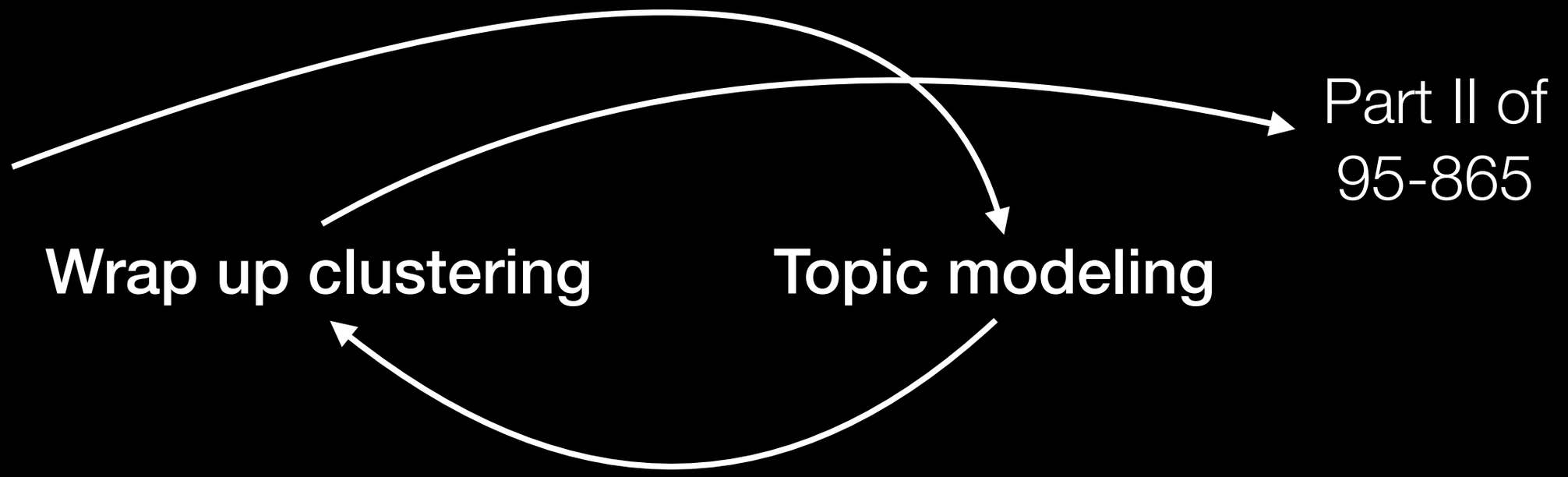
Unstructured Data Analysis

Lecture 9: Topic modeling

George Chen

The usual timeline for 95-865

→ **Wrap up clustering** → **Topic modeling** → Part II of 95-865



How I'll cover things this mini
(so that you have everything you need for HW2)

Is clustering structure enough?

- Fundamentally, clustering models assume each point comes from a single cluster

- In reality, a data point could have “mixed” membership and belong to multiple clusters

Topic Modeling

Text

Each document is part of multiple topics

Each topic consists of a bunch of regularly co-occurring words
(example topics: “sports”, “medicine”, “movies”, “finance”)

Movie recommendation

Each user is part of multiple “clusters”/topics

Each cluster/topic consists of a bunch of movies
(example clusters: “sci-fi epics”, “cheesy rom-coms”)

Health care

Each patient’s health records explained by multiple “topics”

Each topic consists of co-occurring “events”
(example topics: “heart condition”, “severe pancreatitis”)

Topic Modeling

Text

Each document is part of multiple topics

Each topic consists of a bunch of regularly co-occurring words
(examp[illegible]nce")

In all of these examples:

- Each data point (a feature vector) is part of multiple topics
- Each topic corresponds to specific feature values in the feature vector likely appearing

(e[illegible]s")

Health care

Each patient's health records explained by multiple "topics"

Each topic consists of co-occurring "events"
(example topics: "heart condition", "severe pancreatitis")

Latent Dirichlet Allocation (LDA)

- Easy to describe in terms of text (but works for not just text)
- A generative model
- Input: "document-word" matrix, and pre-specified # topics *k*

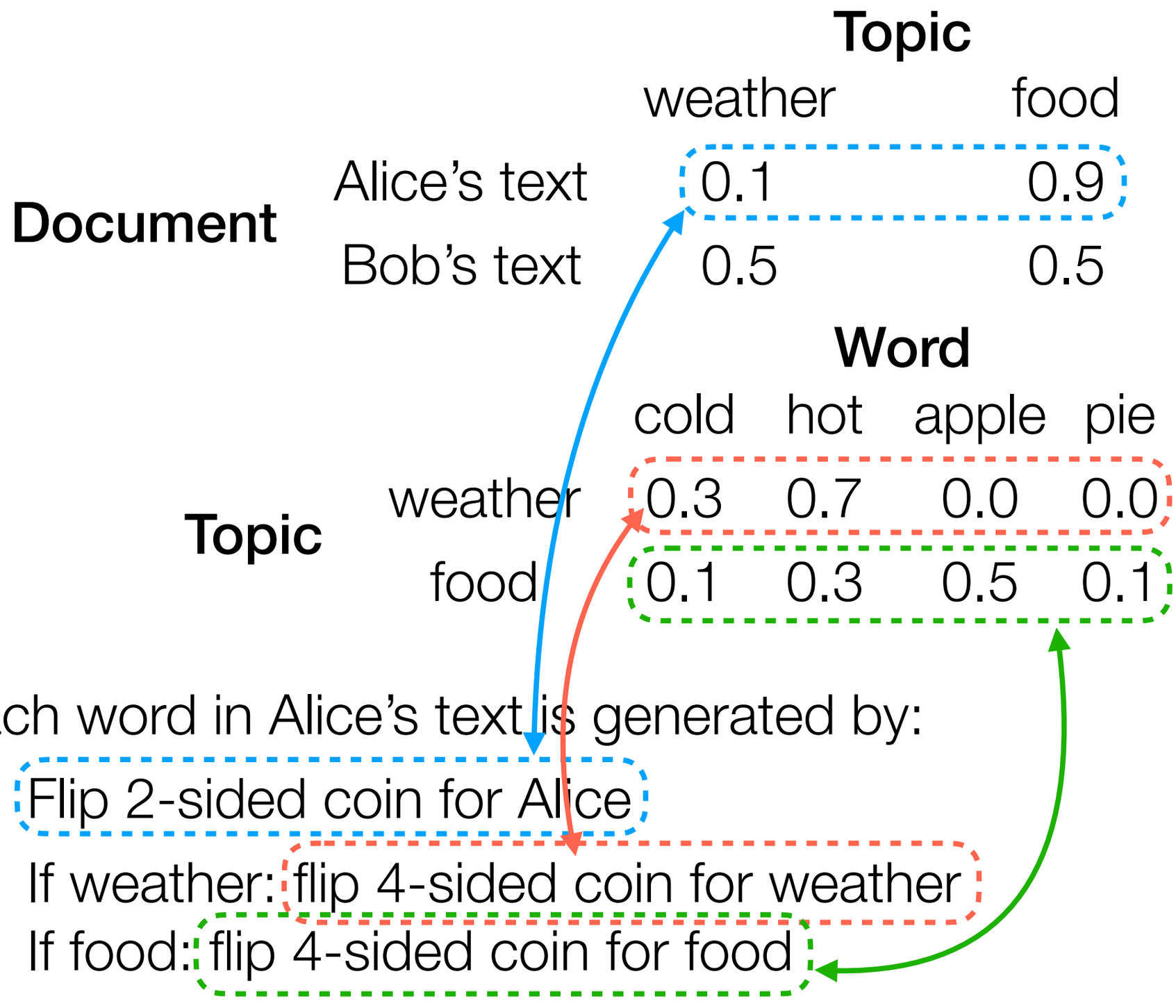
		Word			
		1	2	…	*d*
Document	1				
	2				
	⋮				
	n				

Each row is a feature vector representing a raw counts histogram!

i-th row, *j*-th column: # times word *j* appears in doc *i*

- Output: what the *k* topics are (details on this shortly)

LDA Generative Model Example



Each word in Alice's text is generated by:

1. Flip 2-sided coin for Alice
2. If weather: flip 4-sided coin for weather
   If food: flip 4-sided coin for food

LDA Generative Model Example

Topic

Document	weather	food
Alice's text	0.1	0.9
Bob's text	0.5	0.5

Word

Topic	cold	hot	apple	pie
weather	0.3	0.7	0.0	0.0
food	0.1	0.3	0.5	0.1

Each word in Bob's text is generated by:

1. Flip 2-sided coin for Bob
2. If weather: flip 4-sided coin for weather
   If food: flip 4-sided coin for food

LDA Generative Model Example

Topic

Document	weather	food
Alice's text	0.1	0.9
Bob's text	0.5	0.5

Word

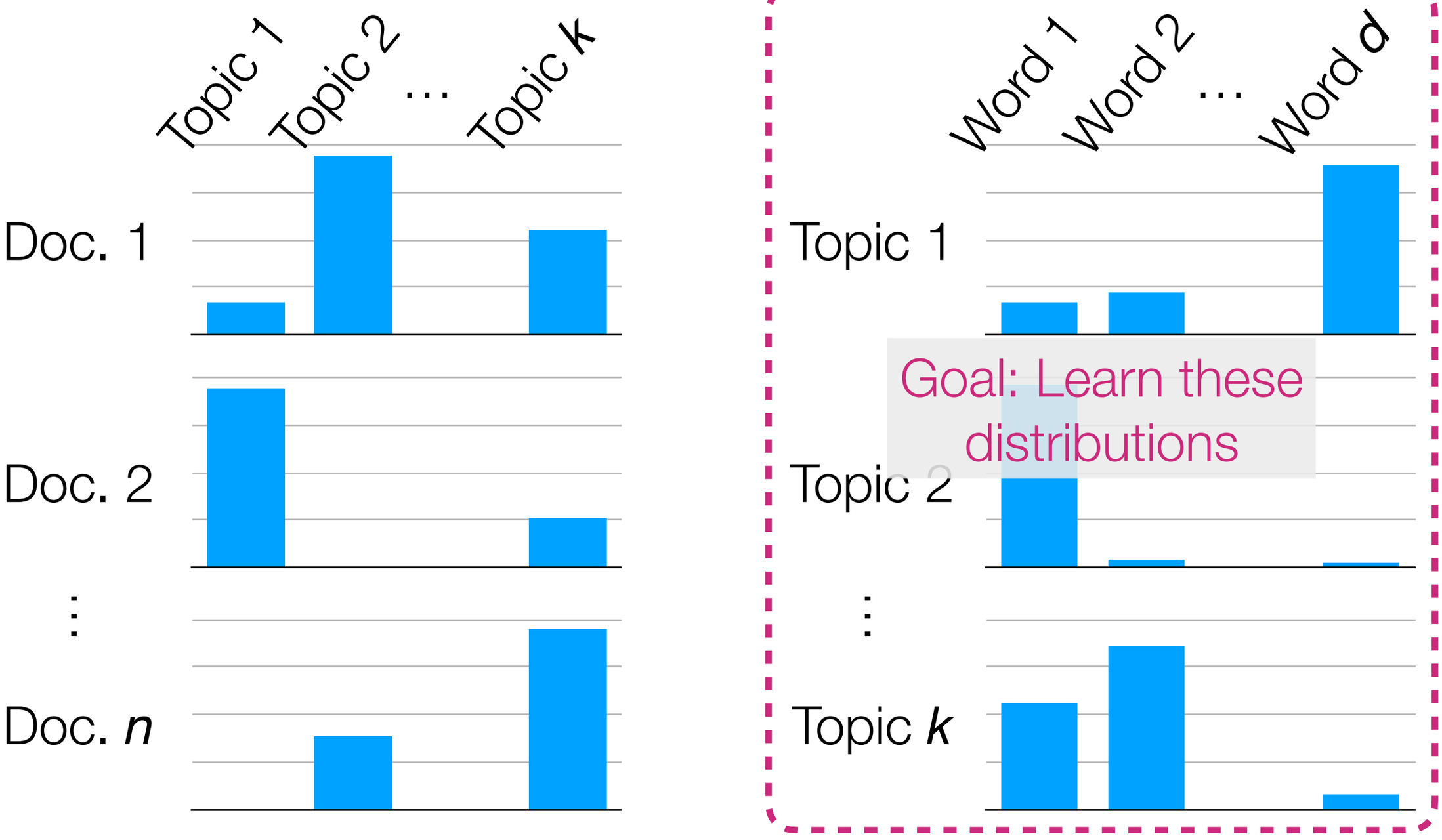
Topic	cold	hot	apple	pie
weather	0.3	0.7	0.0	0.0
food	0.1	0.3	0.5	0.1

Each word in doc *i* is generated by:

1. Flip 2-sided coin for doc *i*
2. If weather: flip 4-sided coin for weather
   If food: flip 4-sided coin for food

"Learning the topics" means figuring out these 4-sided coin probabilities

LDA Generative Model



LDA models each word in document *i* to be generated as:

- Randomly choose a topic *Z* (use topic distribution for doc *i*)
- Randomly choose a word (use word distribution for topic *Z*)

LDA

- Easy to describe in terms of text (but works for not just text)
- A generative model
- Input: “document-word” matrix, and pre-specified # topics *k*

		Word			
		1	2	…	*d*
Document	1				
	2				
	⋮				
	n				

Each row is a feature vector representing a raw counts histogram!

i-th row, *j*-th column: # times word *j* appears in doc *i*

- Output: the *k* topics’ distribution of words

LDA

Demo

An Alternative Feature Vector Representation for Text: TF-IDF

Intuition: words that appear in more documents are likely less useful (same intuition as stop words!) — let's *downweight* these words!

Term frequency (TF)

i-th row, *j*-th column: # times word *j* appears in doc *i*

multiply TF by $\log \frac{1}{\mathbb{P}(\text{document has word } j)} = \log \frac{1}{\left(\frac{\#\text{ documents that have word } j}{n}\right)}$

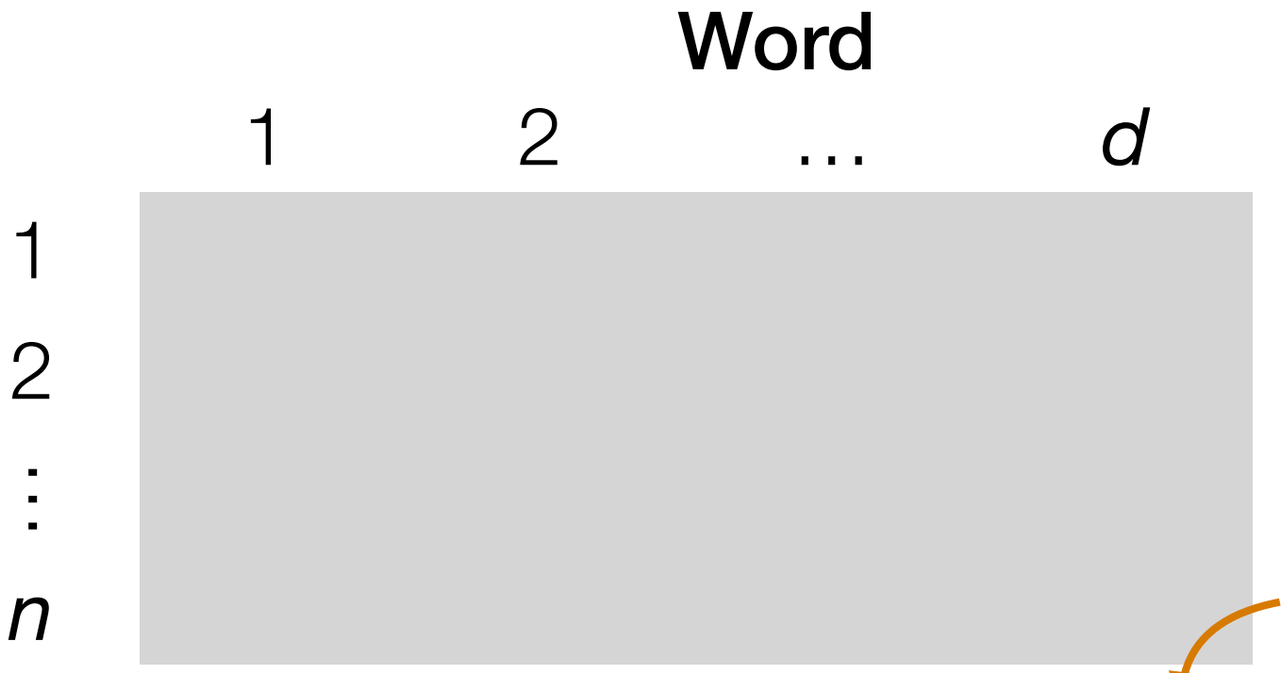
$= \log \frac{n}{\#\text{ documents that have word } j}$

Hack: avoid dividing by 0 by adding 1 to numerator & 1 to denominator

An Alternative Feature Vector Representation for Text: TF-IDF

Intuition: words that appear in more documents are likely less useful (same intuition as stop words!) — let's *downweight* these words!

There are many TF-IDF variants! (Lots of hacks!)



Term frequency (TF)

i-th row, *j*-th column: # times word *j* appears in doc *i*

$$\times \left[1 + \log \frac{1 + n}{1 + \#\text{ documents that have word } j}\right]$$

Default TF-IDF weighting in sklearn

sklearn's default behavior further normalizes each row to have Euclidean norm 1

TF-IDF is in your HW2

(usage is similar to `CountVectorizer` from the demo)