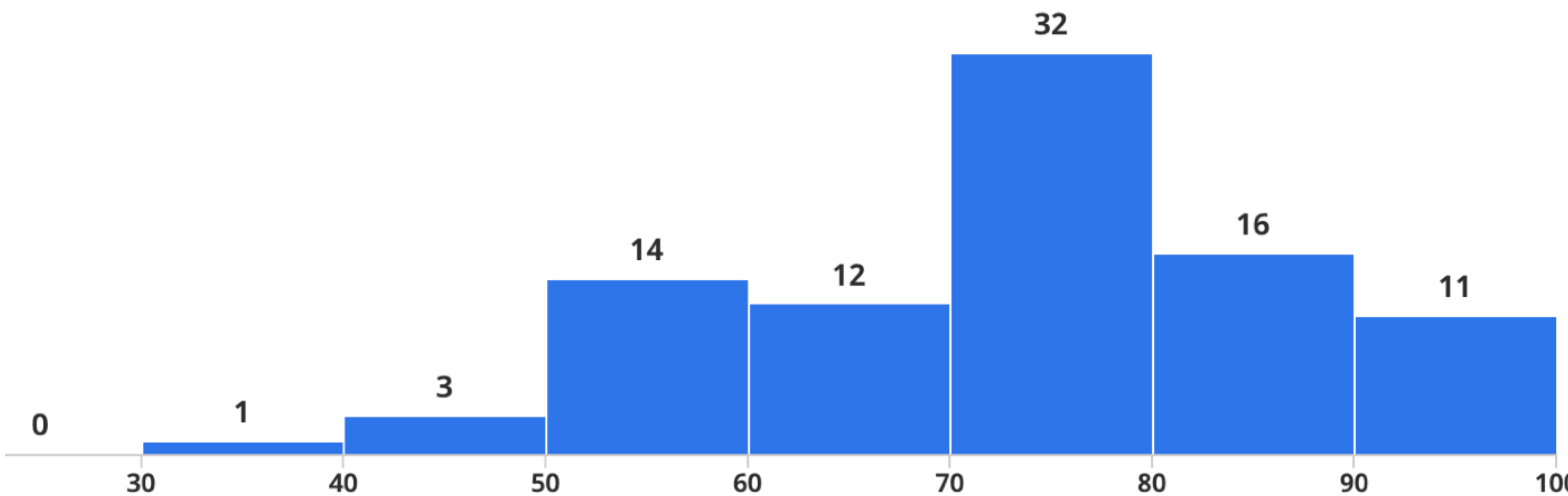
Carnegie Mellon University
Heinz College

95-865 Unstructured Data Analytics

Lecture 9: Wrap up clustering, topic modeling

Slides by George H. Chen

Quiz 1



Median	Maximum	Mean	Std Dev
73.5	98.0	72.42	13.74

These stats are typical of my quizzes (means are typically in the 60s/70s)

Remember: letter grades are assigned based on a curve

Solutions are in Canvas -> Files -> “Quiz 1 solutions.pdf”

Regrade requests (use Gradescope’s regrade request feature)
are due **Monday April 7, 11:59pm**
(for if you think there’s a genuine grading error)

More Things

- Each problem part/subpart is graded by a single grader
- It's possible that some parts/subparts may appear to be graded harsher than others as a result (e.g., some graders may be harsher)
- Very importantly, we emphasize fairness in grading
  - Two students who make the same amount of progress/same mistake(s) receive the same partial credit
- HW1 scores are also out on Gradescope — if you have any sort of regrade request, please use Gradescope's regrade request feature by no later than **Monday April 7, 11:59pm**

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!

Word

1 2 … *d*

Document

1
2
⋮
n

Term frequency (TF)

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

multiply TF by $\log \frac{1}{\mathbb{P}(\text{document mentions word } j)}$

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

Hack (additive smoothing): can add 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!)

Word

1 2 … *d*

Document

1
2
⋮
n

Term frequency (TF)

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

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

Default TF-IDF weighting in sklearn

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

An Alternative Feature Vector Representation for Text: TF-IDF

Demo

Clustering on Images

See the demo linked on the course webpage
(this is considered **required** reading material
so please do take a look sometime after class)

Last Remarks on Clustering

- We only saw two clustering methods (*k*-means, GMM)
- We only saw one general strategy to automatically choose # of clusters
    - You must specify a score function — no score function is perfect
- There are *lots* of clustering methods out there!
    - Many do not require specifying # of clusters (DP-means, DP-GMM, many variants of hierarchical clustering, DBSCAN, OPTICS, …)
- Ultimately, you have to decide on which clustering method and number of clusters make sense for your data
    - After you run a clustering algorithm, make visualizations to interpret the clusters *in the context of your application*!
    - Do not just blindly rely on numerical metrics (e.g., CH index)

Is clustering structure enough?

(Flashback) GMM with *k* Clusters

<u>Cluster 1</u>

Probability of generating a point from cluster 1 = π_1

Gaussian mean = μ_1

Gaussian covariance = Σ_1

$\ldots$

<u>Cluster *k*</u>

Probability of generating a point from cluster *k* = π_k

Gaussian mean = μ_k

d-dim.

Gaussian covariance = Σ_k

d-by-*d* matrices

How to generate points from this GMM:

1. Flip biased coin (side 1 has probability π_1, …, side *k* has probability π_k)

   Let *Z* be the side that we got (it is some value 1, …, *k*)

2. Sample 1 point from the Gaussian from cluster *Z*

Each data point has a single true cluster assignment *Z*
& is generated from the Gaussian for cluster *Z*

In reality, a data point could have "mixed" membership and belong to multiple "clusters"

For example, for news articles, possible topics could be *sports*, *medicine*, *movies*, or *finance*

A news article could be about *sports* and also about *finance*

How do we model this?

Topic Modeling:
Latent Dirichlet Allocation (LDA)

- A generative model

- Input: "document-word" matrix, and pre-specified # topics *k*

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

LDA Generative Model Example

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

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

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

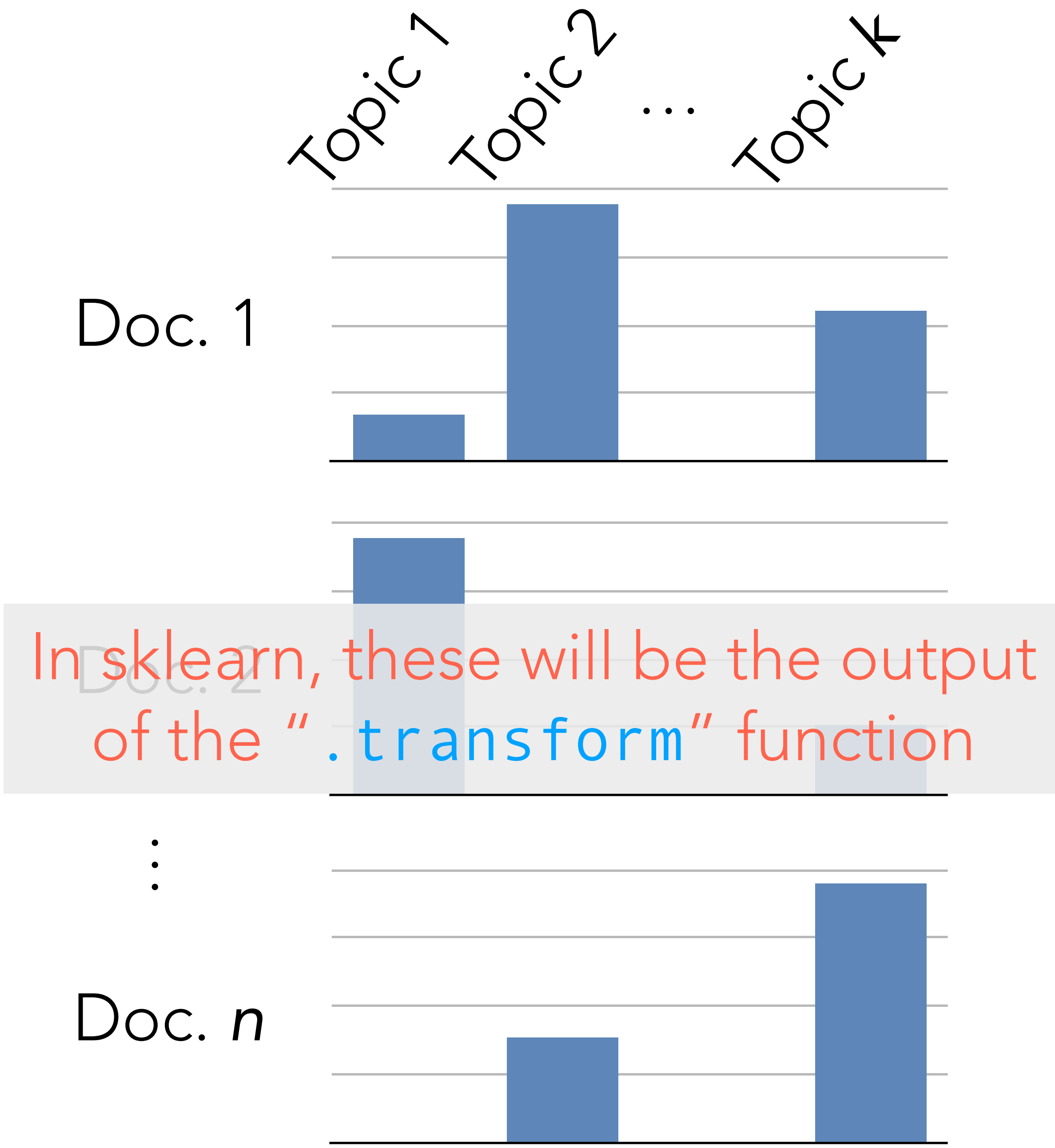
		Word			
		cold	hot	apple	pie
Topic	weather	0.3	0.7	0.0	0.0
	food	0.1	0.3	0.5	0.1

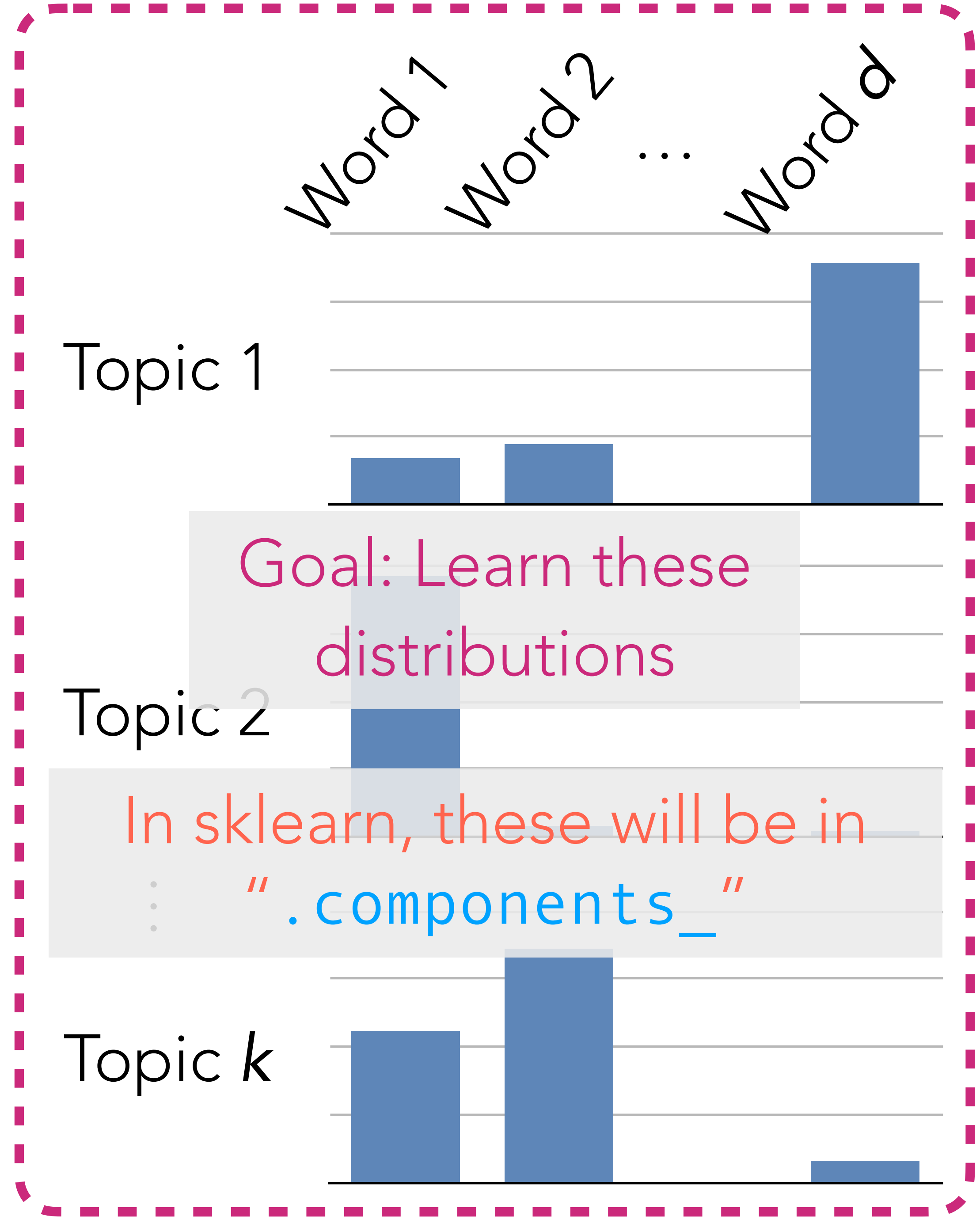
Each word in Bob's text is generated by:

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

"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:

1. Randomly choose a topic *Z* (use topic distribution for doc *i*)
2. Randomly choose a word (use word distribution for topic *Z*)

Topic Modeling: Latent Dirichlet Allocation (LDA)

- A generative model

- Input: "document-word" matrix, and pre-specified # topics *k*

		Word			
		1	2	…	*d*
Document	1	Either TF table or TF-IDF table			
	2				
	⋮				
	n				

- Output: the *k* topics' distributions over words

LDA

Demo