Clustering and Similarity: Retrieving Documents

Emily Fox & Carlos Guestrin

Machine Learning Specialization

University of Washington

Retrieving documents of interest

Document retrieval

- Currently reading article you like

Document retrieval

- Currently reading article you like
- **Goal:** Want to find similar article

Document retrieval

Challenges

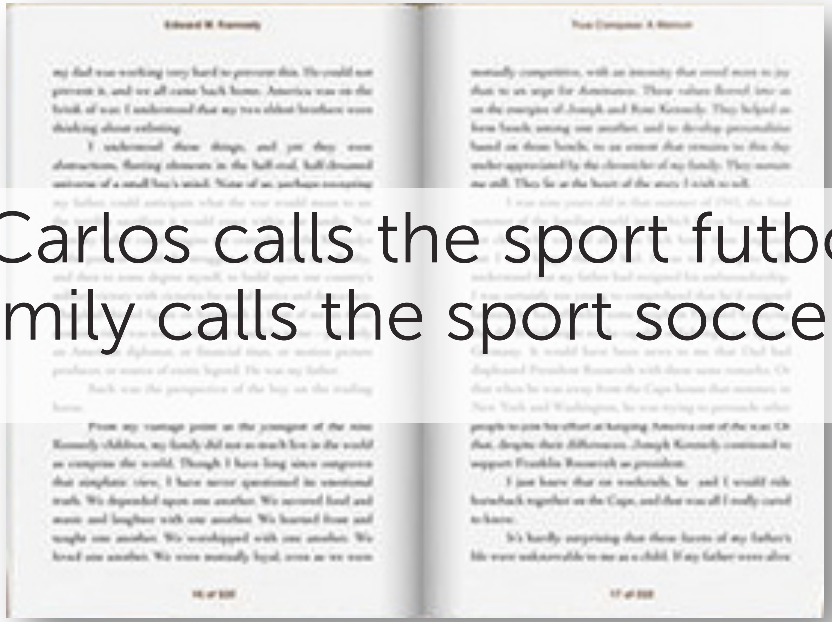
- How do we measure similarity?
- How do we search over articles?

Word count representation for measuring similarity

Word count document representation

- Bag of words model
  - Ignore order of words
  - Count # of instances of each word in vocabulary

0	1	2	0	0	2	2	0	1	0	0	1	1	....
	carlos	the		tree	calls	sport	cat	futbol	dog		soccer	Emily	



"Carlos calls the sport futbol. Emily calls the sport soccer."

Measuring similarity

1	0	0	0	5	3	0	0	1	0	0	0	0

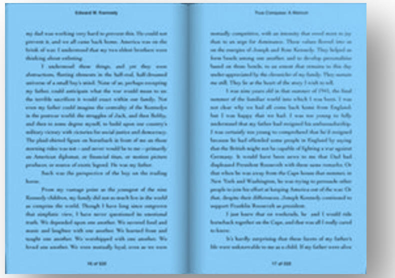
3	0	0	0	2	0	0	1	0	1	0	0	0

1*3

+

5*2

= 13

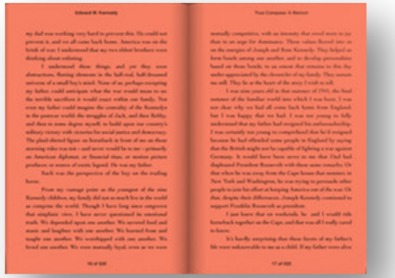


Measuring similarity

1	0	0	0	5	3	0	0	1	0	0	0	0

$\longrightarrow$ **0**

0	0	1	0	0	0	9	0	0	6	0	4	0



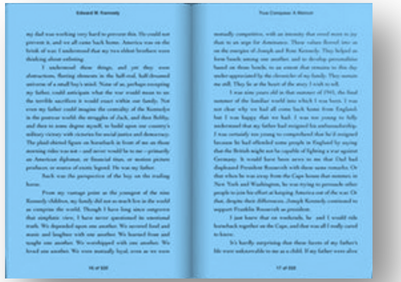
Issues with word counts – Doc length

1	0	0	0	5	3	0	0	1	0	0	0	0

2	0	0	0	10	6	0	0	2	0	0	0	0

3	0	0	0	2	0	0	1	0	1	0	0	0

Similarity = 13



6	0	0	0	4	0	0	2	0	2	0	0	0

Similarity = 52

Solution = normalize

1	0	0	0	5	3	0	0	1	0	0	0	0

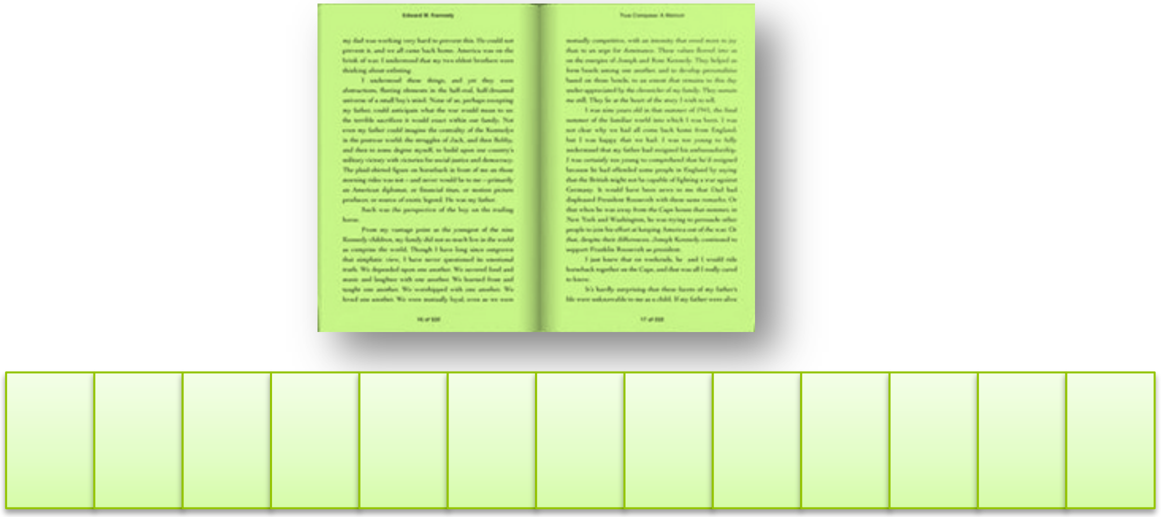
$$\sqrt{(1^2 + 5^2 + 3^2 + 1^2)}$$

Compute the norm of the word count vector.

1/6	0	0	0	5/6	3/6	0	0	1/6	0	0	0	0

Prioritizing important words with tf-idf

Issues with word counts – Rare words



Common words in doc: “the”, “player”, “field”, “goal”

Dominate rare words like: “futbol”, “Messi”

Document frequency

- What characterizes a rare word?
  - Appears infrequently in the corpus

Up-weight rare words

- Emphasize words appearing in few docs
  - Equivalently, discount word *w* based on
    # of docs containing *w* in corpus

Important words

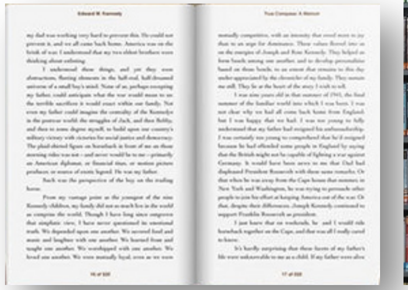
- Do we want only rare words to dominate???

- What characterizes an important word?
  - Appears frequently in document (common locally)
  - Appears rarely in corpus (rare globally)

- Trade off between local frequency and global rarity

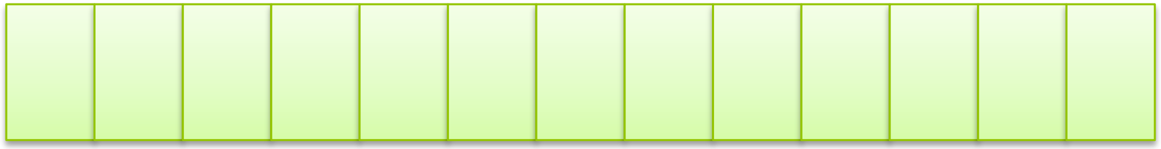
TF-IDF document representation

- Term frequency – inverse document frequency (tf-idf)

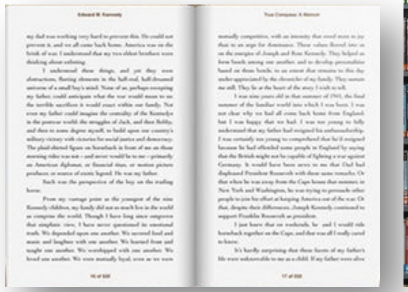



TF-IDF document representation

- Term frequency – inverse document frequency (tf-idf)
- Term frequency

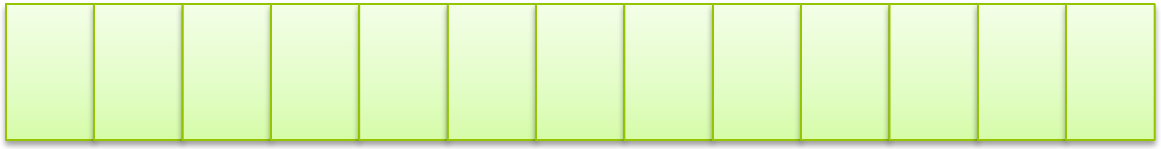


- Same as word counts



TF-IDF document representation

- Term frequency – inverse document frequency (tf-idf)
- Term frequency

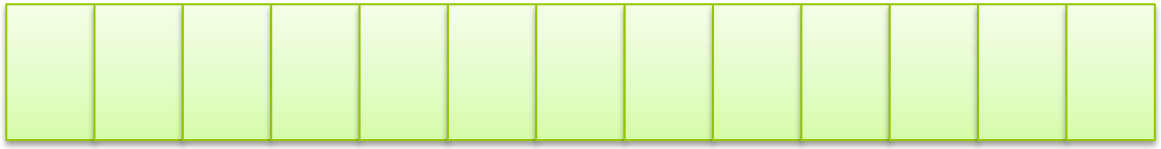


- Inverse document frequency

$$\log \frac{\#\ \text{docs}}{1 + \#\ \text{docs using word}}$$

TF-IDF document representation

- Term frequency – inverse document frequency (tf-idf)
- Term frequency



- Inverse document frequency

$$\log \frac{\#\ \text{docs}}{1 + \#\ \text{docs using word}}$$

word in many docs

$$\log \frac{\text{large}\#}{1+\text{large}\#} \approx \log 1 = 0$$

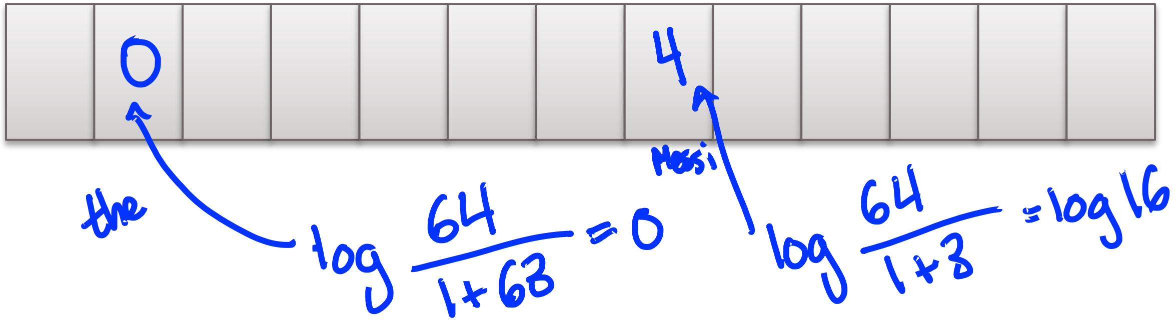
rare word

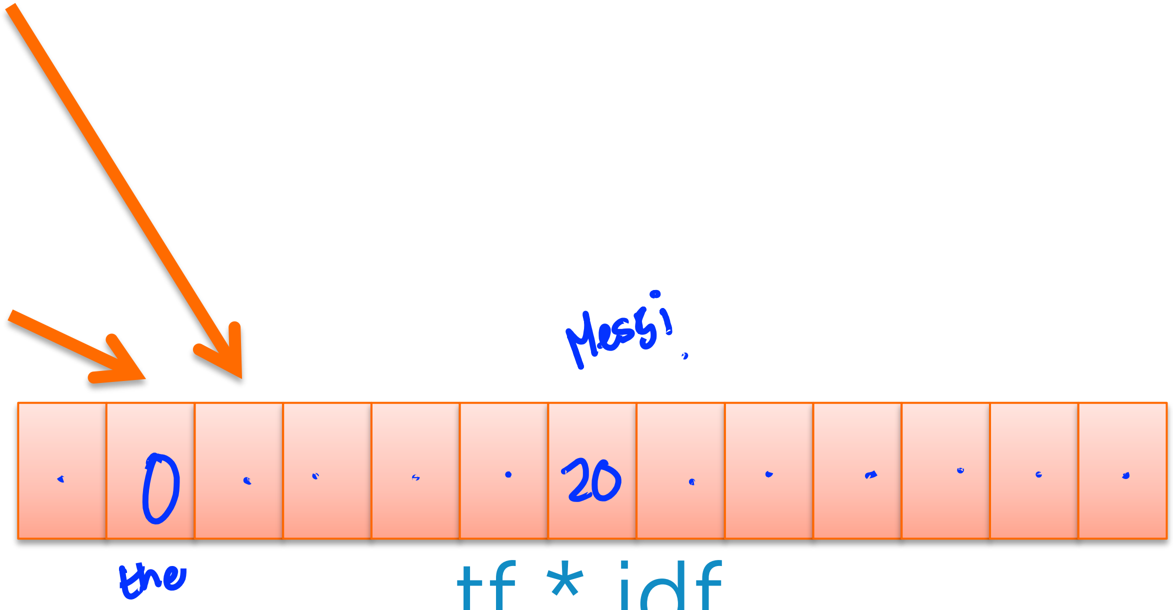
$$\log \frac{\text{large}\#}{1+\text{small}\#} \rightarrow \text{large}\#$$

TF-IDF document representation

- Term frequency – inverse document frequency (tf-idf)
- Term frequency

- Inverse document frequency

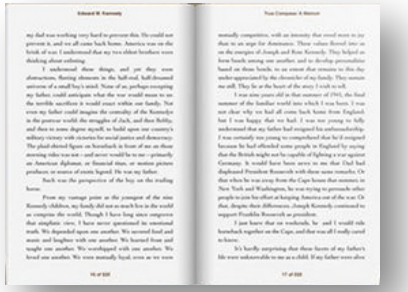
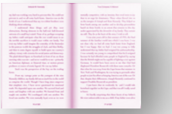
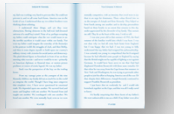
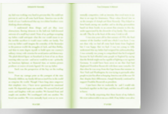




tf * idf

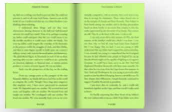
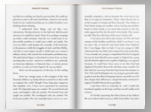
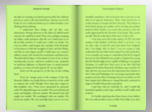
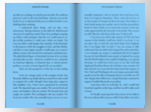
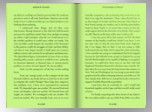
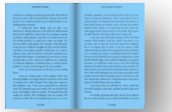
Retrieving similar documents

Nearest neighbor search

- Query article: 
- Corpus: 
- **Specify:** Distance metric
- **Output:** Set of most similar articles   

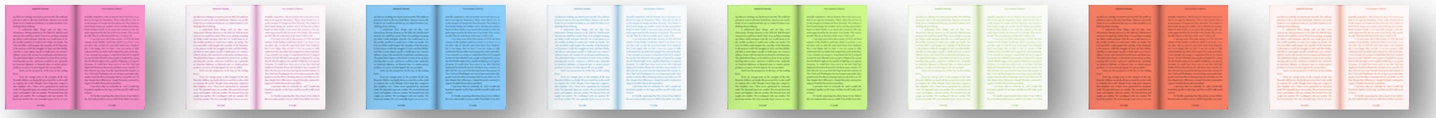
1 – Nearest neighbor

- **Input:** Query article 
- **Output:** *Most* similar article

- Algorithm:
  - Search over each article  in corpus
    - Compute **s = similarity(**  **,**  **)**
    - If **s > Best_s**, record  **=**  and set **Best_s = s**
  - Return 

k – Nearest neighbor

- **Input:** Query article 
- **Output:** *List of k* similar articles



Instead of keeping just the most related artical, keep a priority queue of the top k articles found so far.

Clustering documents

Structure documents by topic

- Discover groups (*clusters*) of related articles




SPORTS

BBC WORLD NEWS



WORLD NEWS

What if some of the labels are known?

- Training set of labeled docs

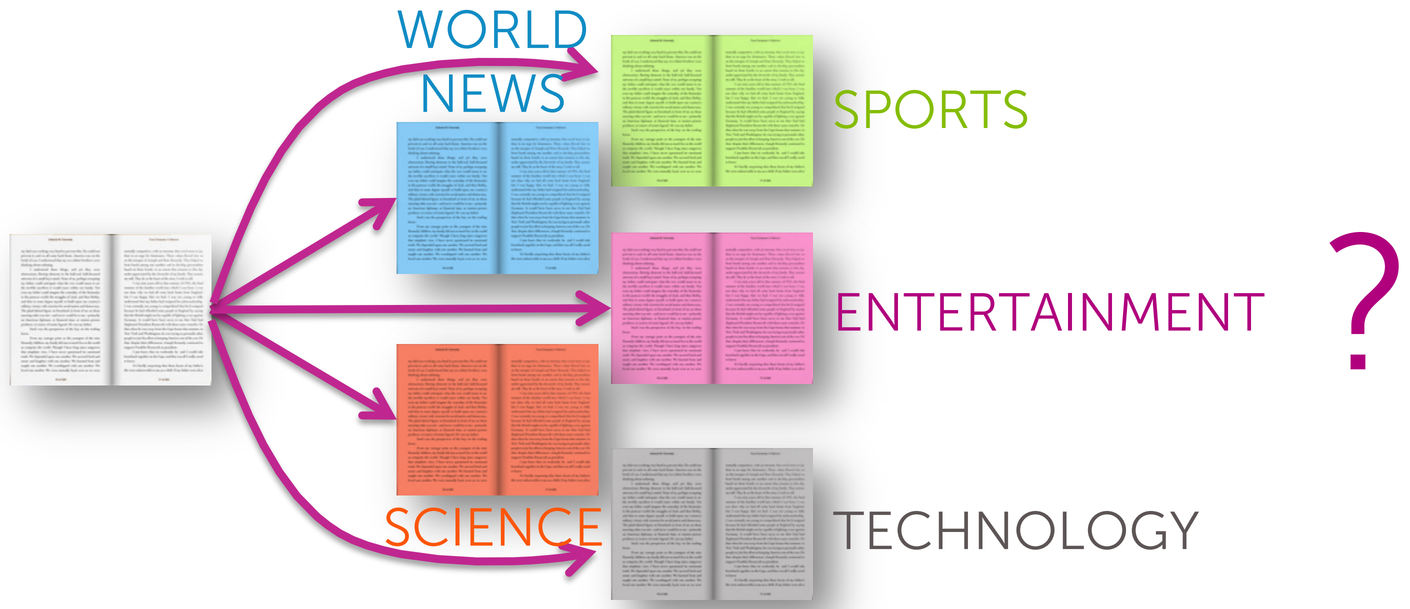
SPORTS

WORLD NEWS

ENTERTAINMENT

SCIENCE

Multiclass classification problem



Example of
supervised learning

Clustering

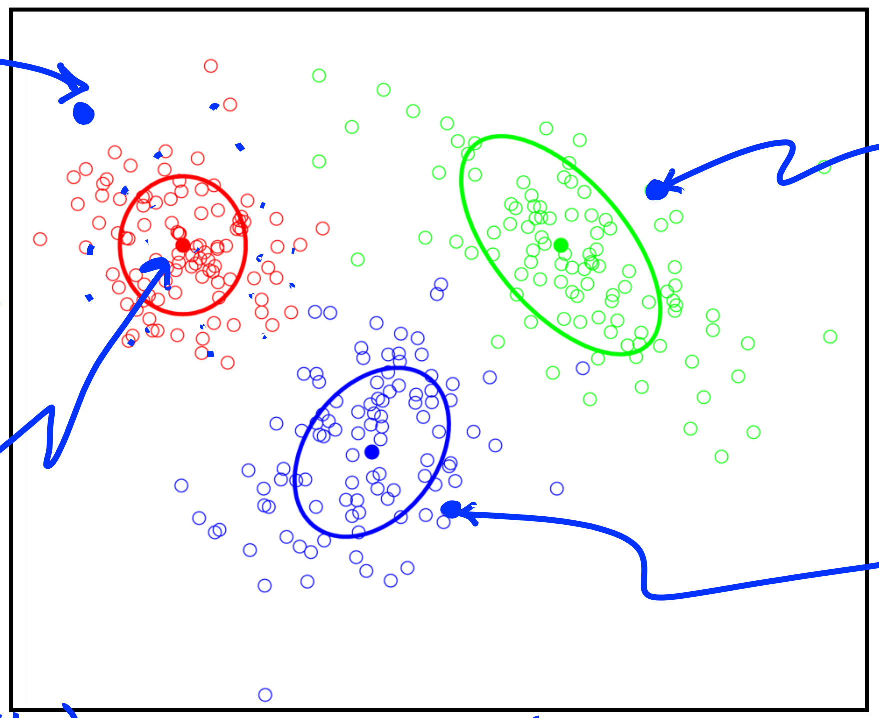
- No labels provided
- Want to uncover cluster structure

word count vectors, in this
example only two words

word1 word2

① ② ③

- **Input:** docs as vectors
- **Output:** cluster labels



word 2

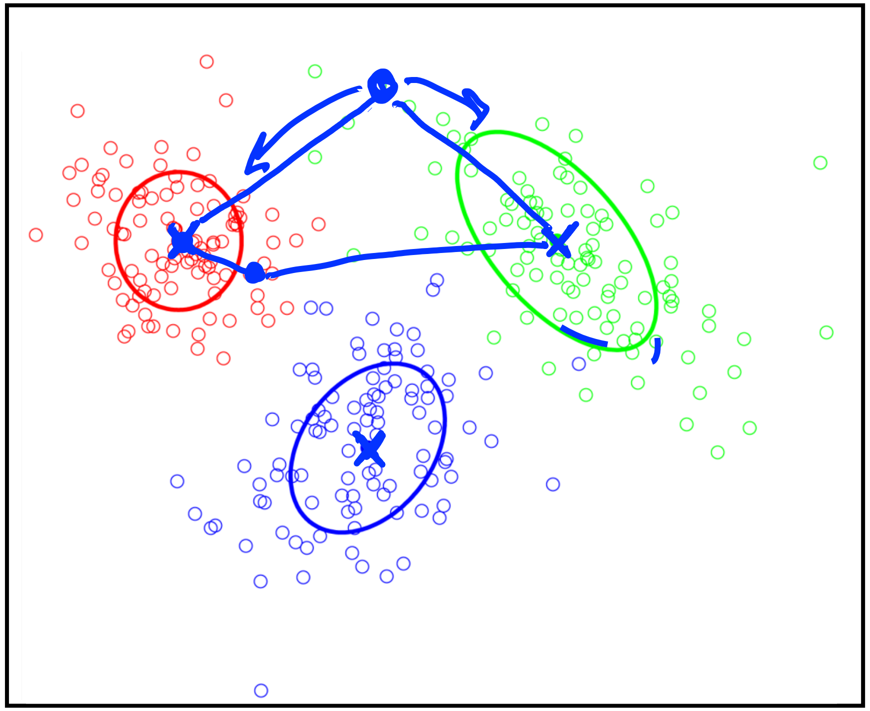
word 1

cluster 1
"sports"
(label provided postfacto)

An **unsupervised learning** task

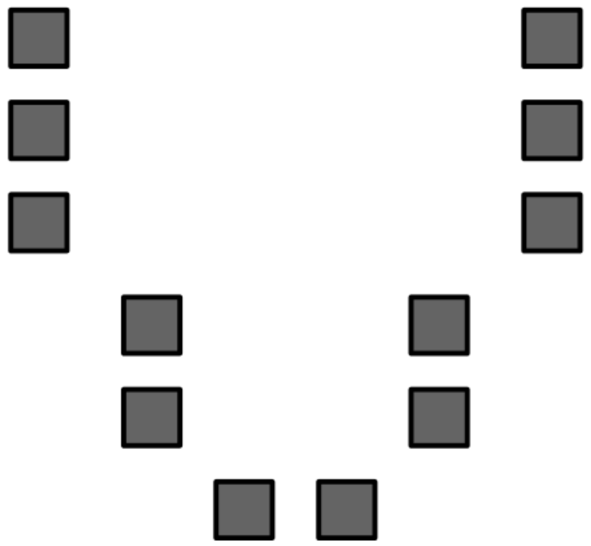
What defines a cluster?

- Cluster defined by center & shape/spread
- Assign observation (doc) to cluster (topic label)
  - Score under cluster is higher than others
    score every observation based on cluster center/shape
  - Often, just more similar to assigned cluster center than other cluster centers



k-means a clustering algorithm

- Assume
  - Similarity metric = distance to cluster center (smaller better)

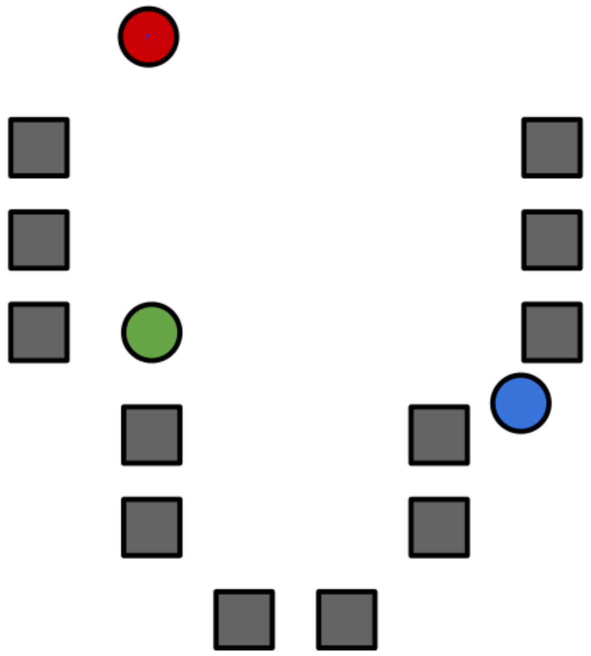


DATA to CLUSTER

k-means algorithm

0. Initialize cluster centers

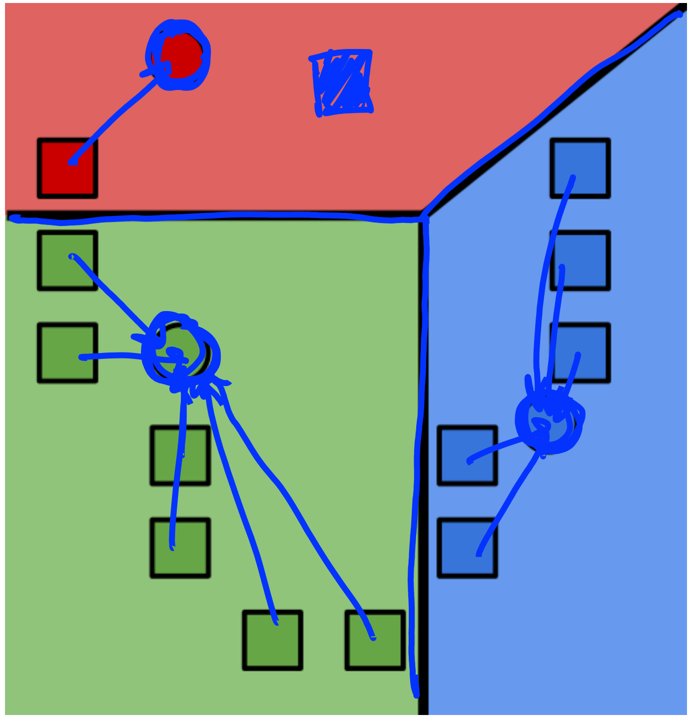
- specify the number of clusters ahead of time
- k clusters, look at means of clusters, just the cluster centers, when assigning points to different clusters.
- many ways to initialize where to put cluster centers, in this example just randomly initialize 3 centers.



k-means algorithm

0. Initialize cluster centers
1. Assign observations to closest cluster center

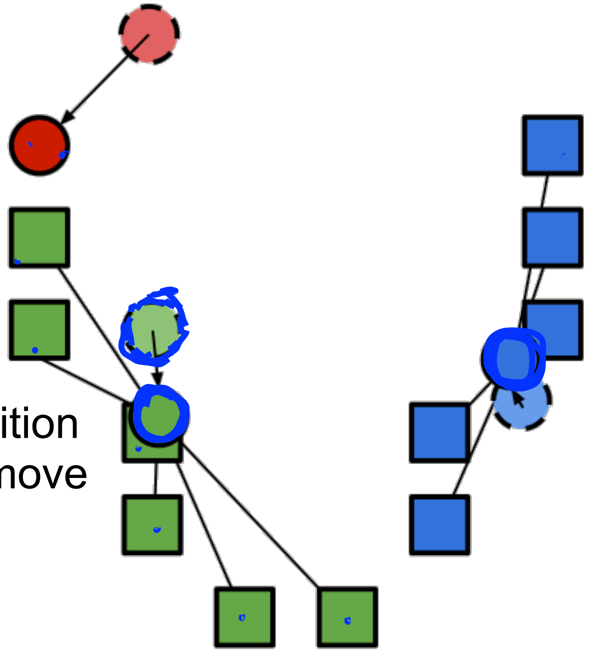
A way to do this is something called Voronoi tessellation.
- define regions, which represent areas where any observation might get



k-means algorithm

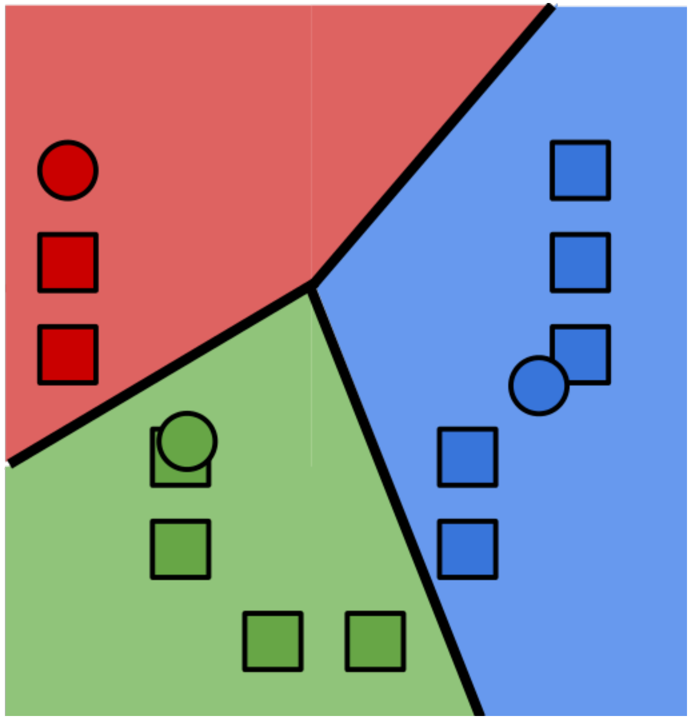
0. Initialize cluster centers
1. Assign observations to closest cluster center
2. Revise cluster centers as mean of assigned observations

cluster centers not really representing structure underlying data -> iterate this process, update definition of cluster center based on assigned observations (move to center of mass of all assigned observations).



k-means algorithm

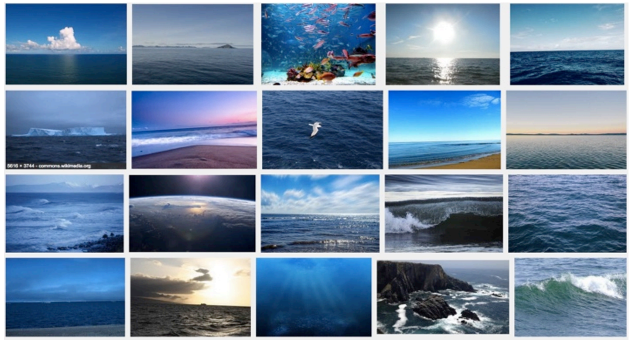
0. Initialize cluster centers
1. Assign observations to closest cluster center
2. Revise cluster centers as mean of assigned observations
3. Repeat 1.+2. until convergence

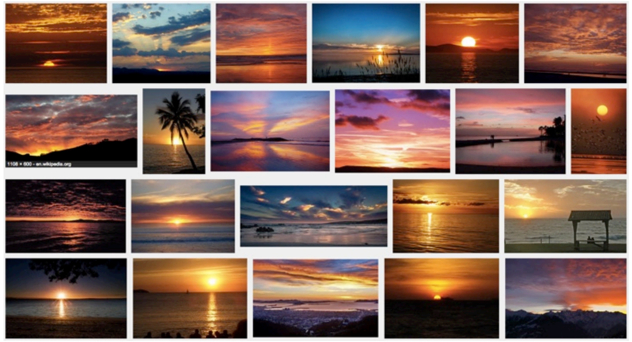


Other examples

Clustering images

- For search, group as:
  - Ocean
  - Pink flower
  - Dog
  - Sunset
  - Clouds
  - …

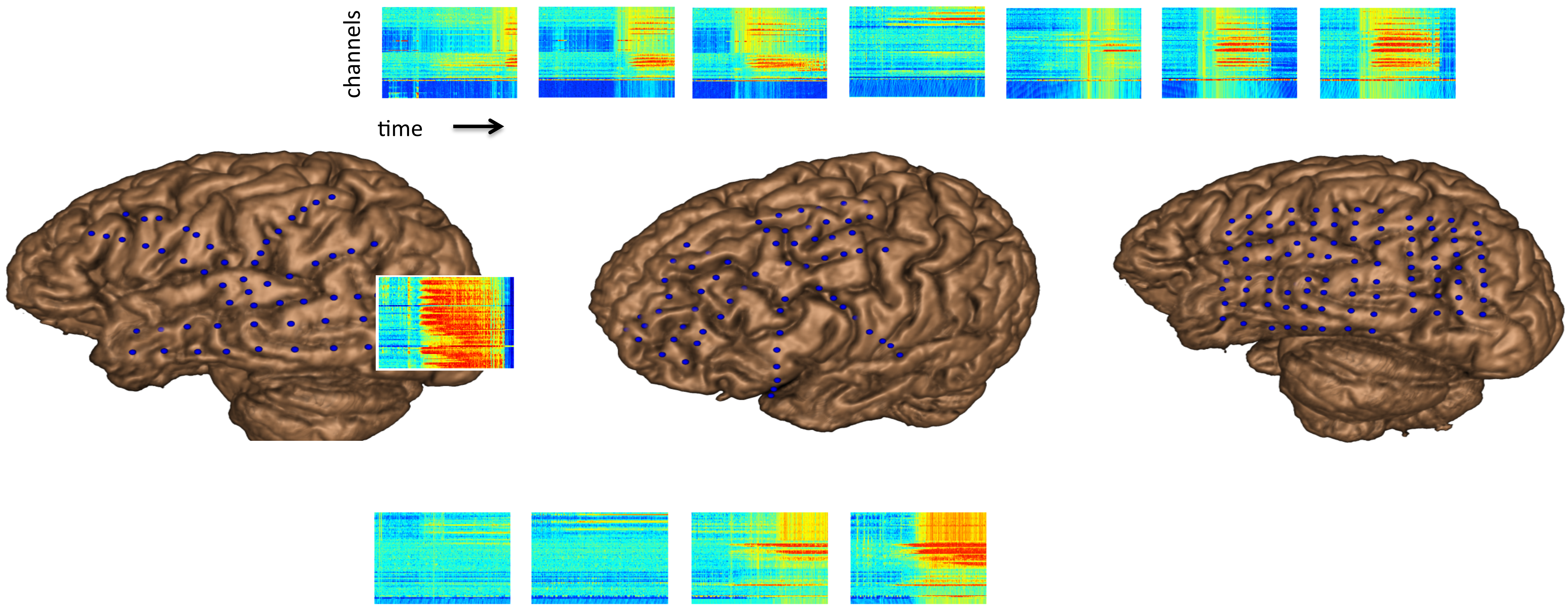




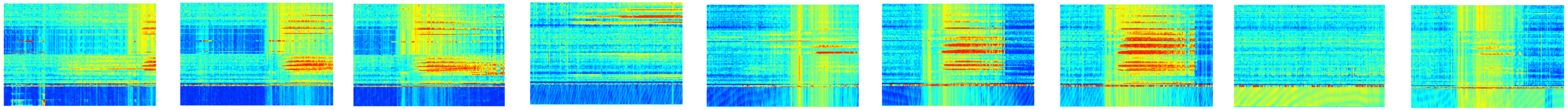
Grouping patients by medical condition

- Better characterize subpopulations and diseases

Example: Patients and seizures are diverse



Cluster seizures by observed time courses



Products on Amazon

- Discover product categories from purchase histories

~~"furniture"~~
"baby"

- Or discovering groups of **users**

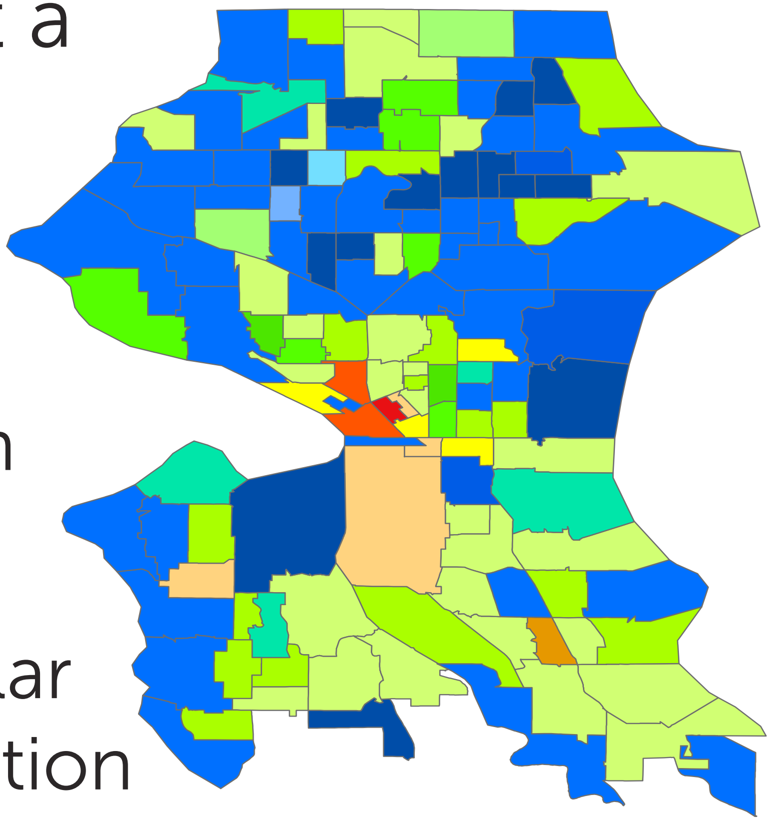
Structuring web search results

- Search terms can have multiple meanings
- Example: "cardinal"

- Use clustering to structure output

Discovering similar neighborhoods

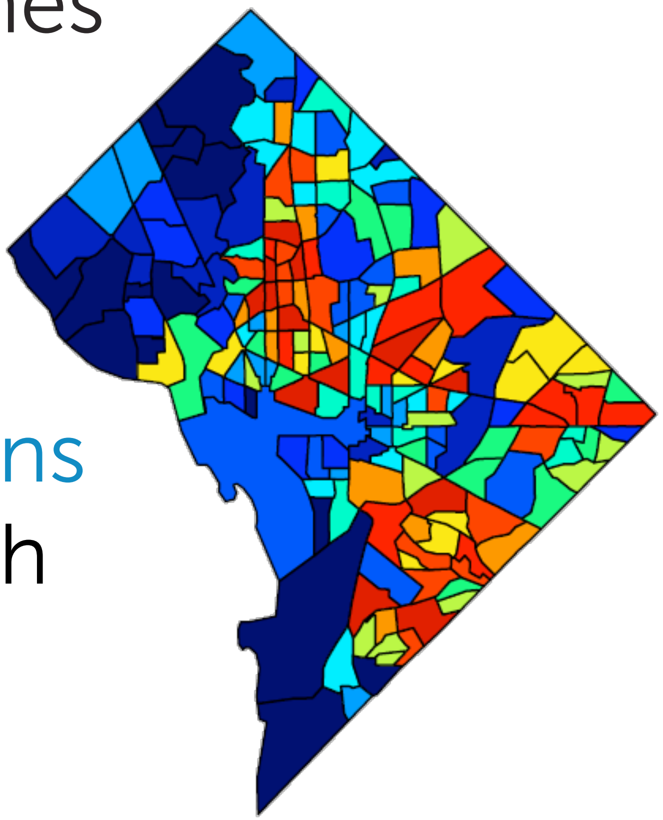
- **Task 1:** Estimate price at a small regional level
- Challenge:
  - Only a few (or no!) sales in each region per month
- Solution:
  - Cluster regions with similar trends and share information within a cluster



City of Seattle

Discovering similar neighborhoods

- **Task 2:** Forecast violent crimes to better task police
- Again, cluster regions and share information!
- Leads to improved predictions compared to examining each region independently



Washington, DC

Summary for clustering and similarity

What you can do now…

- Describe ways to represent a document (e.g., raw word counts, tf-idf,…)
- Measure the similarity between two documents
- Discuss issues related to using raw word counts
  - Normalize counts to adjust for document length
  - Emphasize important words using tf-idf
- Implement a nearest neighbor search for document retrieval
- Describe the input (unlabeled observations) and output (labels) of a clustering algorithm
- Determine whether a task is supervised or unsupervised
- Cluster documents using k-means (algorithmic details to come…)
- Describe other applications of clustering