15-826: Multimedia (Databases) and Data Mining

Lecture #14: Text - part III:

Vector space model and clustering

C. Faloutsos

Must-Read Material

- MM Textbook, Chapter 6

Outline

Goal: 'Find similar / interesting things'

- Intro to DB
- Indexing - similarity search
- Data Mining

Indexing - Detailed outline

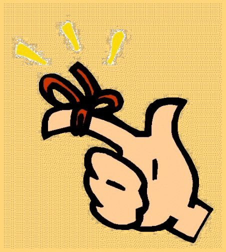
- primary key indexing
- secondary key / multi-key indexing
- spatial access methods
- fractals
- text
- multimedia
- ...

Text - Detailed outline

- text
  - problem
  - full text scanning
  - inversion
  - signature files
  - clustering
  - information filtering and LSI

Problem

- How to find doc’s with “data mining”?



Conclusion

- How to find doc's with "data mining"?
- A1: full text scanning
  - A1.1: string editing distance
- A2: inversion
  - Elias Codes
- (A3: signature files – 'Bloom filters')
- A4: vector space model + clustering

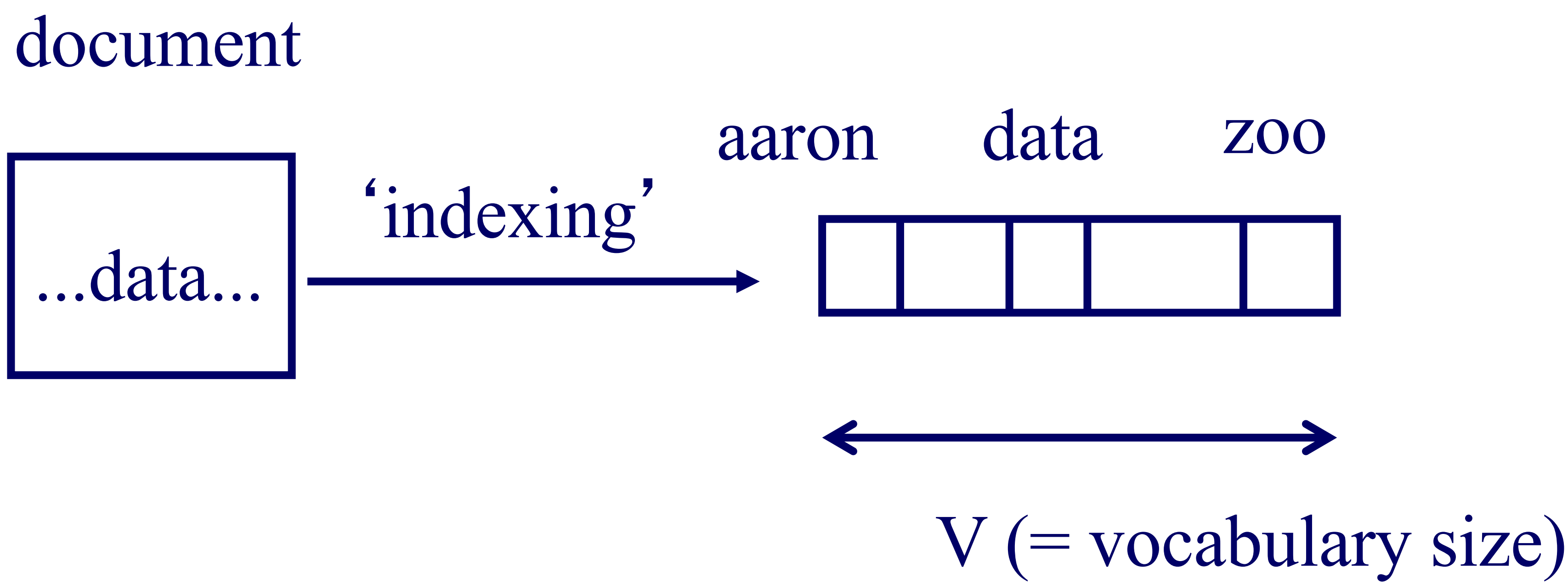
Vector Space Model and Clustering

- keyword queries (vs Boolean)
- each document: -> vector (HOW?)
- each query: -> vector
- search for ‘similar’ vectors

Carnegie Mellon

Vector Space Model and Clustering

- main idea:



Vector Space Model and Clustering

Then, group nearby vectors together

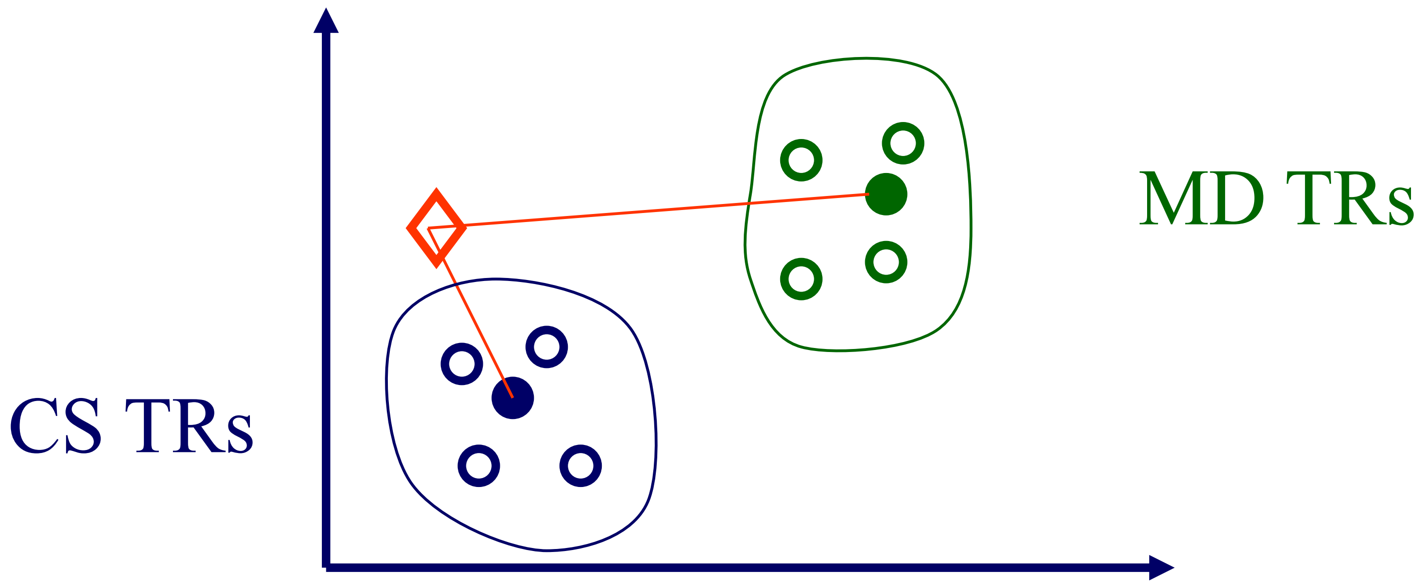
- Q1: cluster search?
- Q2: cluster generation?

Two significant contributions

- ranked output
- relevance feedback

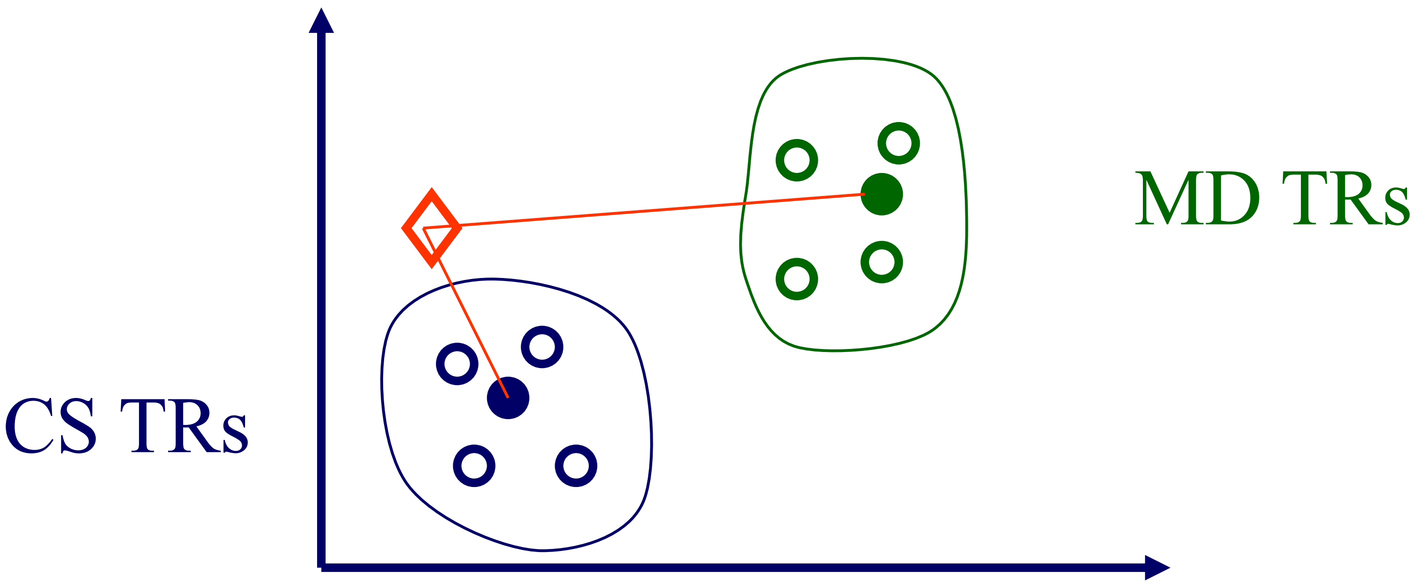
Vector Space Model and Clustering

- cluster search: visit the (k) closest superclusters; continue recursively



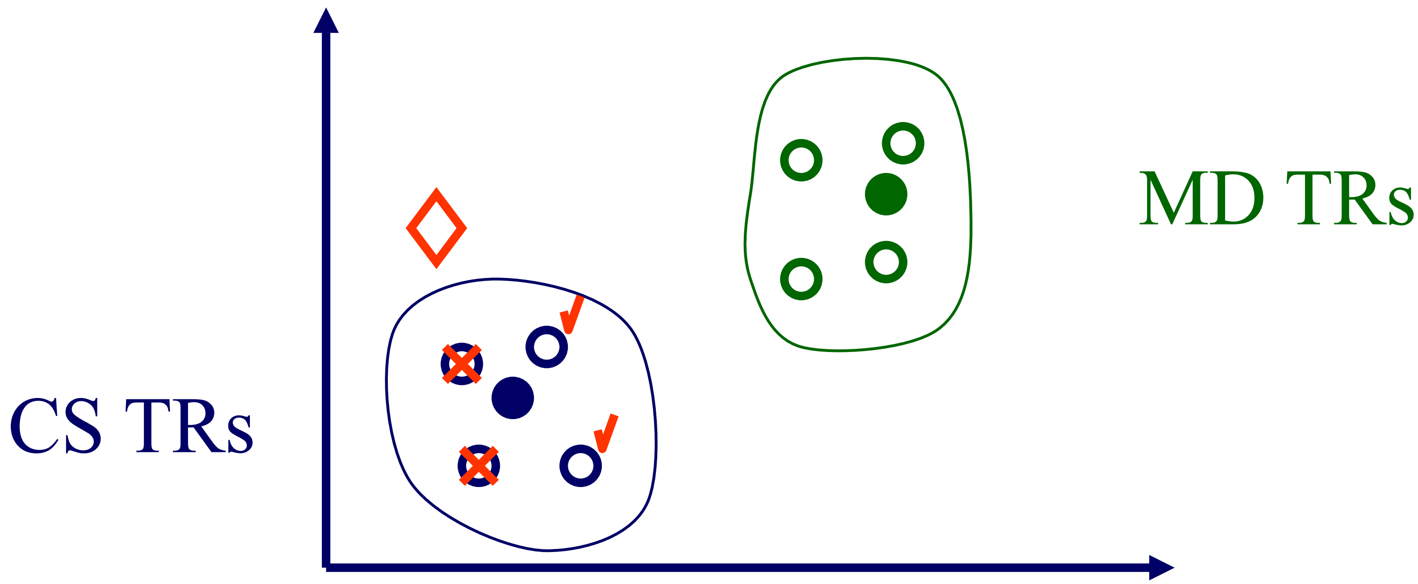
Vector Space Model and Clustering

- ranked output: easy!



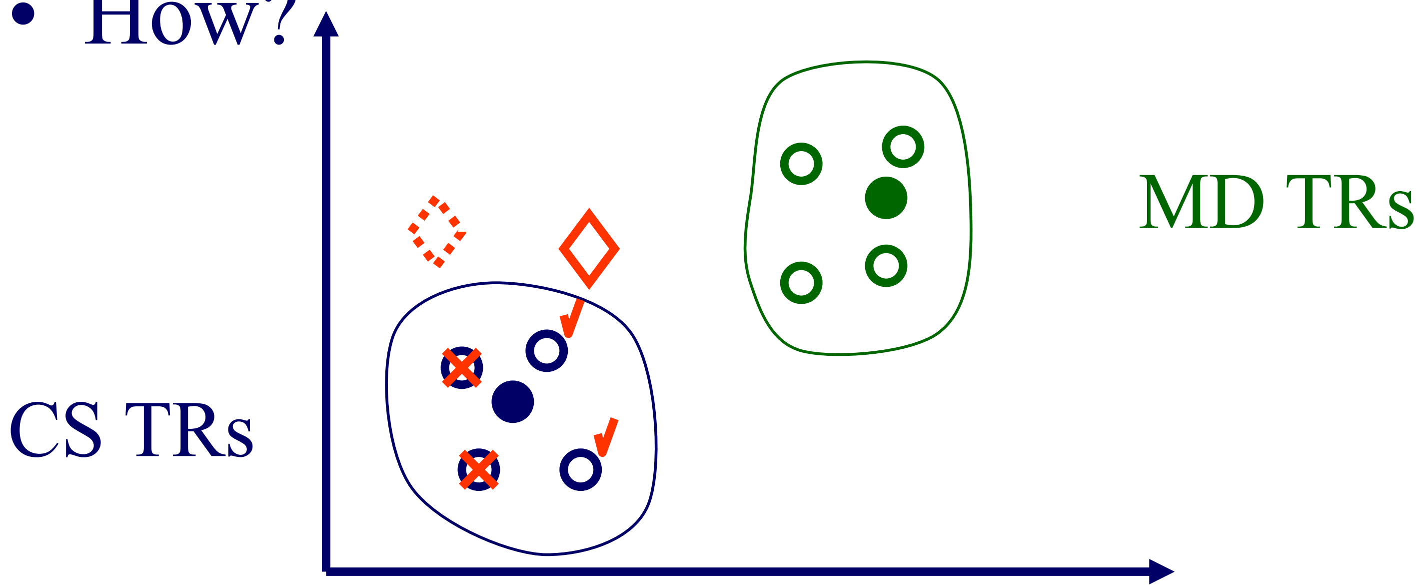
Vector Space Model and Clustering

- relevance feedback (brilliant idea) [Roccio' 73]



Vector Space Model and Clustering

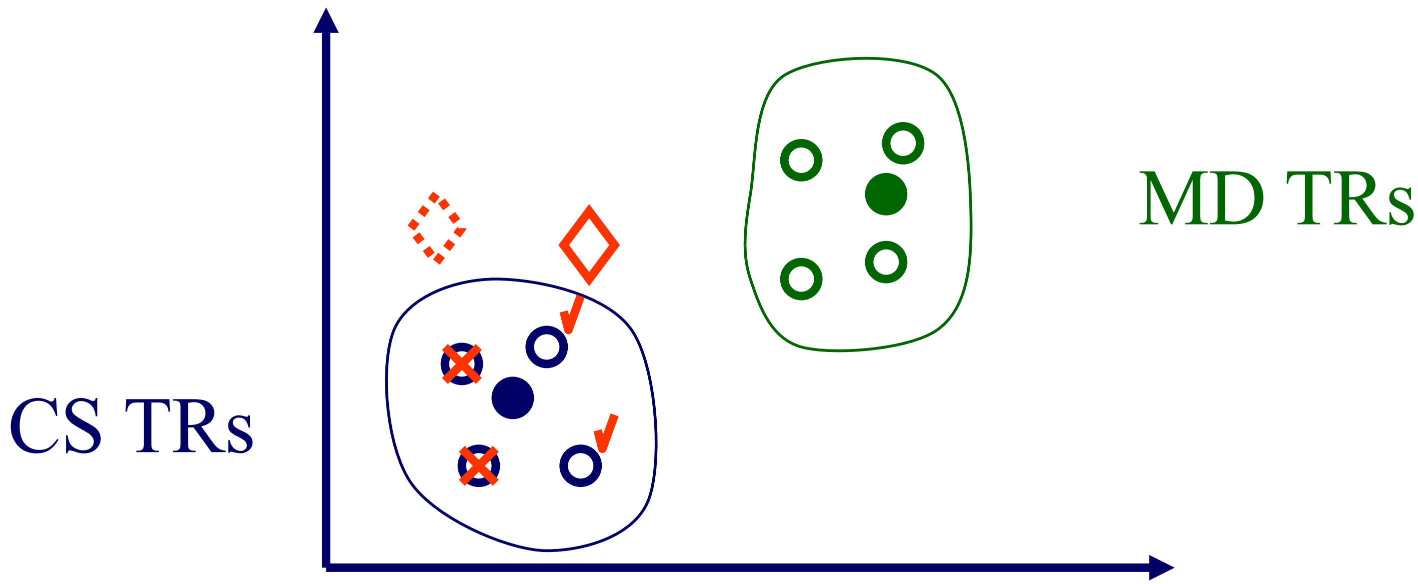
- relevance feedback (brilliant idea) [Roccio' 73]
- How?



Carnegie Mellon

Vector Space Model and Clustering

- How? A: by adding the 'good' vectors and subtracting the 'bad' ones

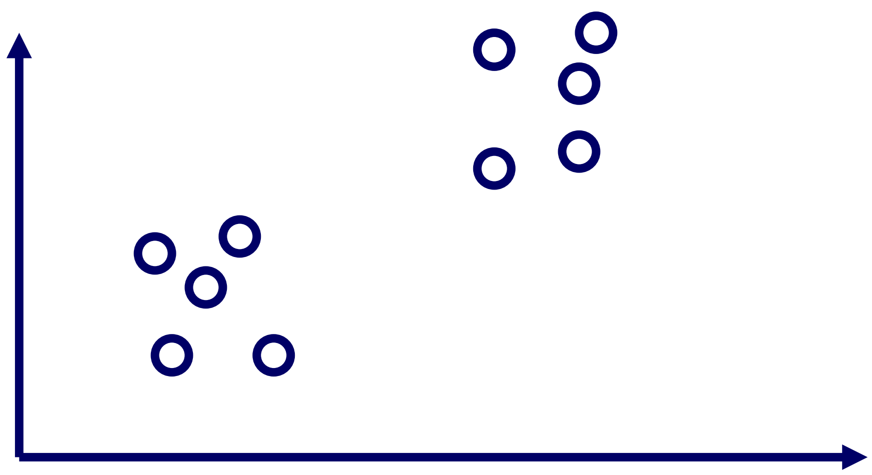


Outline - detailed

- main idea
- cluster search
- cluster generation
- evaluation

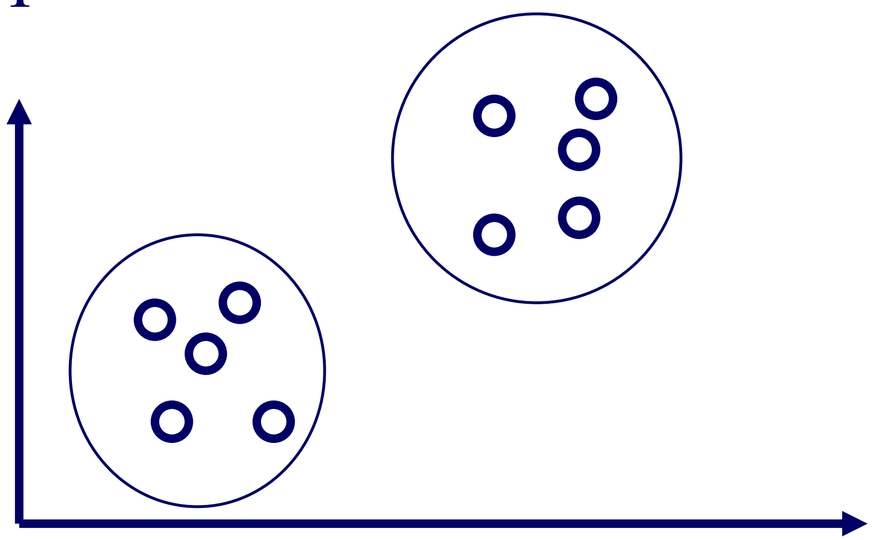
Cluster generation

- Problem:
  - given N points in V dimensions,
  - group them



Cluster generation

- Problem:
  - given N points in V dimensions,
  - group them



Cluster generation

We need

- Q1: document-to-document similarity
- Q2: document-to-cluster similarity

Cluster generation

Q1: document-to-document similarity

(recall: ‘bag of words’ representation)

- D1: { ‘data’, ‘retrieval’, ‘system’ }
- D2: { ‘lung’, ‘pulmonary’, ‘system’ }
- distance/similarity functions?

Cluster generation

A1: # of words in common

A2: .... normalized by the vocabulary sizes

A3: .... etc

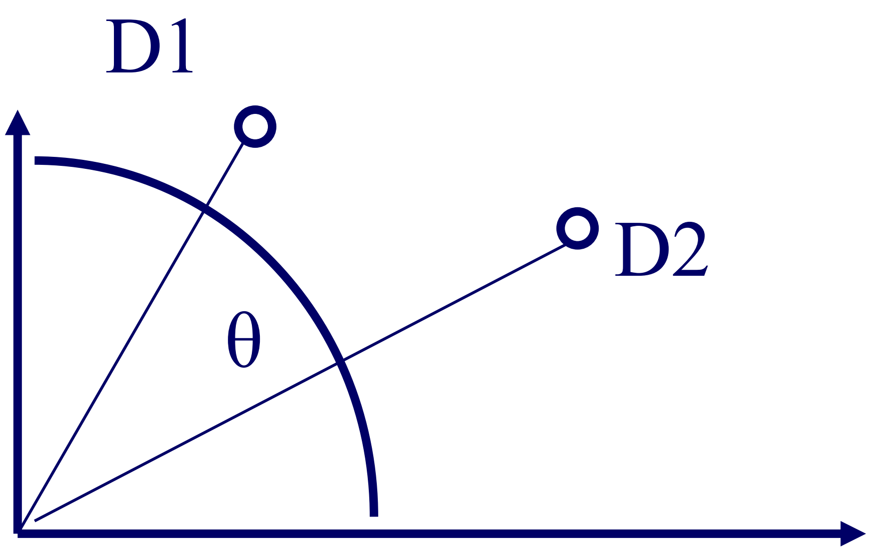
About the same performance - prevailing one: cosine similarity

Cluster generation

cosine similarity:

$$\text{similarity}(D1, D2) = \cos(\theta) =$$

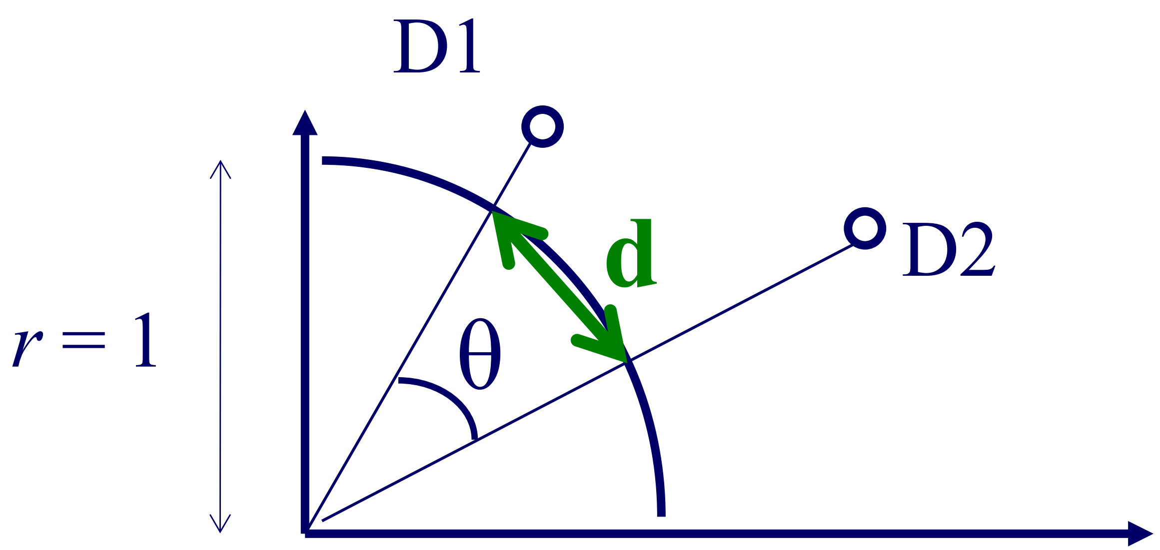
$$\text{sum}(v_{1,i} * v_{2,i}) / \text{len}(v_1) / \text{len}(v_2)$$



Cluster generation

cosine similarity - observations:

- related to the Euclidean distance
- weights $v_{i,j}$: according to tf/idf

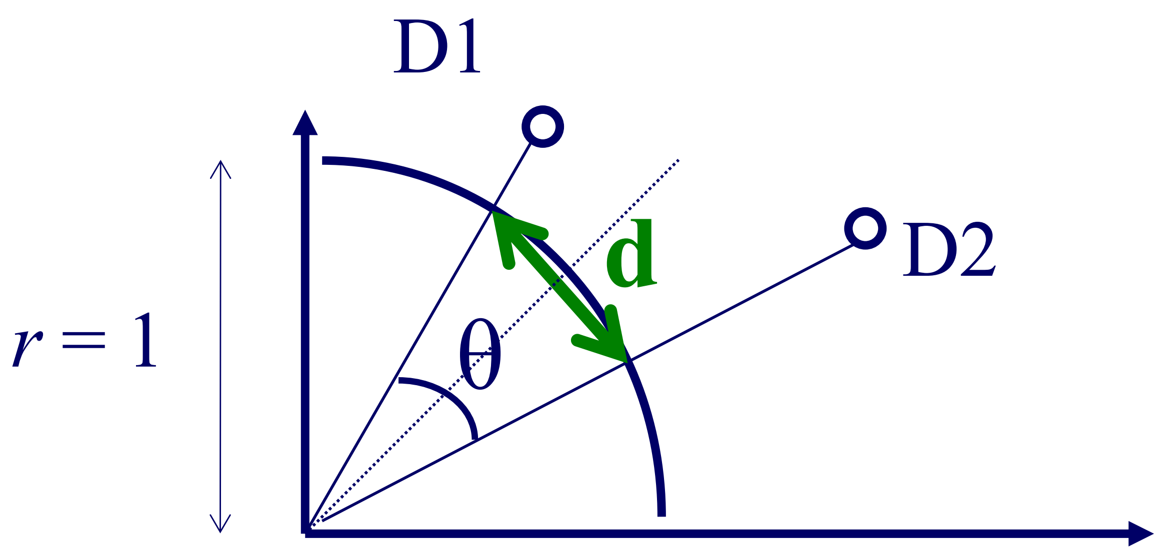


DETAILS

Cluster generation

cosine similarity - observations:

- related to the Euclidean distance
- weights $v_{i,j}$: according to tf/idf



$$d = 2 * \sin(\theta/2)$$

$$d^2 = 2 * (1 - \cos(\theta))$$

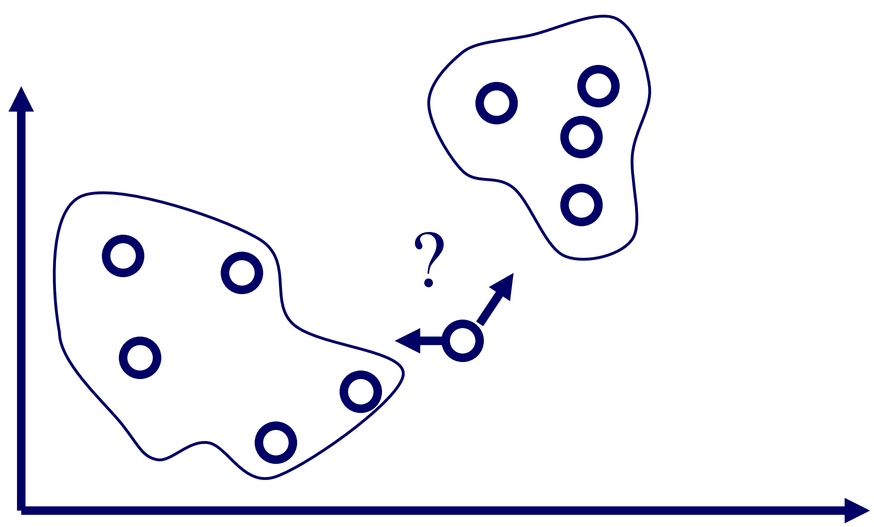
Cluster generation

- tf (‘term frequency’)
  - high, if the term appears very often in this document.
- idf (‘inverse document frequency’)
  - penalizes ‘common’ words, that appear in almost every document

Cluster generation

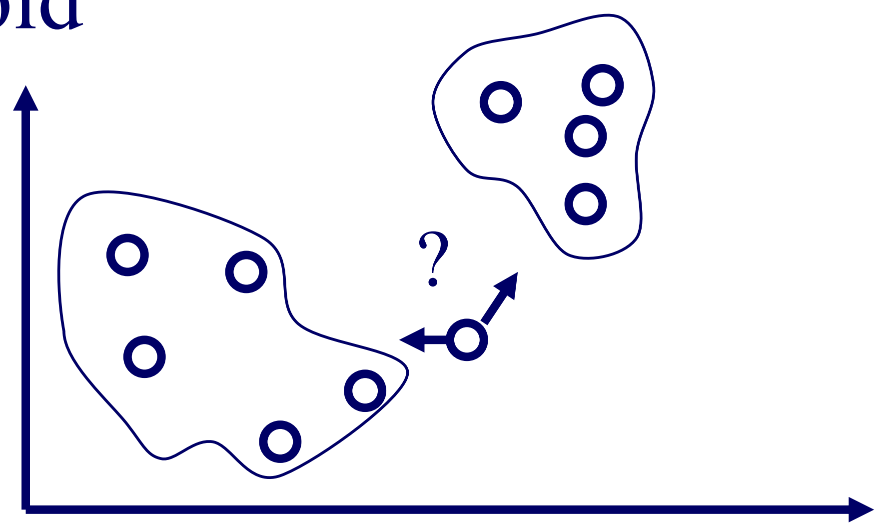
We need

- Q1: document-to-document similarity
- Q2: document-to-cluster similarity



Cluster generation

- A1: min distance (‘single-link’)
- A2: max distance (‘all-link’)
- A3: avg distance
- A4: distance to centroid



Cluster generation

- A1: min distance (‘single-link’)
  - leads to elongated clusters
- A2: max distance (‘all-link’)
  - many, small, tight clusters
- A3: avg distance
  - in between the above
- A4: distance to centroid
  - fast to compute

Cluster generation

We have

- document-to-document similarity
- document-to-cluster similarity

Q: How to group documents into ‘natural’ clusters

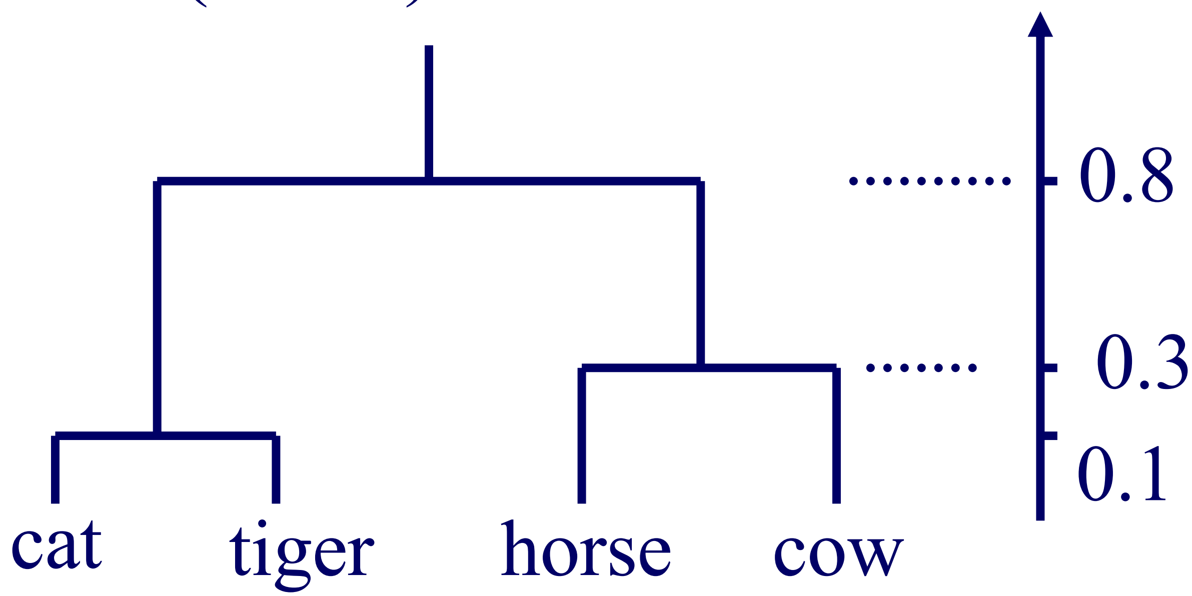
Cluster generation

A: *many-many* algorithms - in two groups [VanRijsbergen]:

- theoretically sound (O(N^2))
  - independent of the insertion order
- iterative (O(N), O(N log(N))

Cluster generation - ‘sound’ methods

- Approach#1: dendrograms - create a hierarchy (bottom up or top-down) - choose a cut-off (how?) and cut



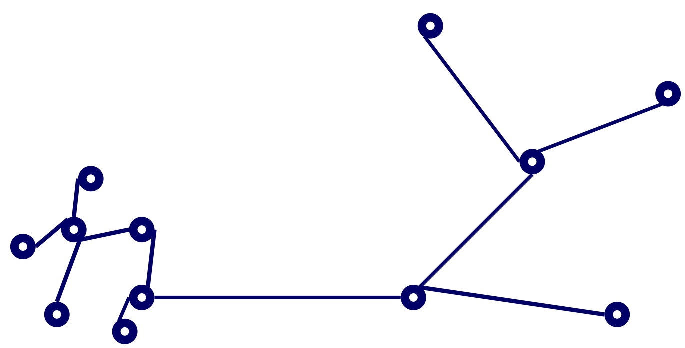
Cluster generation - ‘sound’ methods

- Approach#2: min. some statistical criterion (eg., sum of squares from cluster centers)
  - like ‘k-means’
  - but how to decide ‘k’ ?

Carnegie Mellon

Cluster generation - ‘sound’ methods

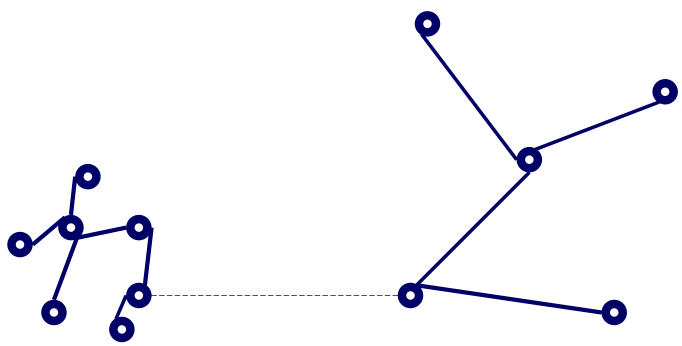
- Approach#3: Graph theoretic [Zahn]:
  - build MST;
  - delete edges longer than 3* std of the local average



Carnegie Mellon

Cluster generation - 'sound' methods

- Result:



- why '3' ?
- variations
- Complexity?

Cluster generation - ‘iterative’ methods

general outline:

- Choose ‘seeds’ (how?)
- assign each vector to its closest seed (possibly adjusting cluster centroid)
- possibly, re-assign some vectors to improve clusters

Fast and practical, but ‘unpredictable’

Cluster generation - 'iterative' methods

general outline:

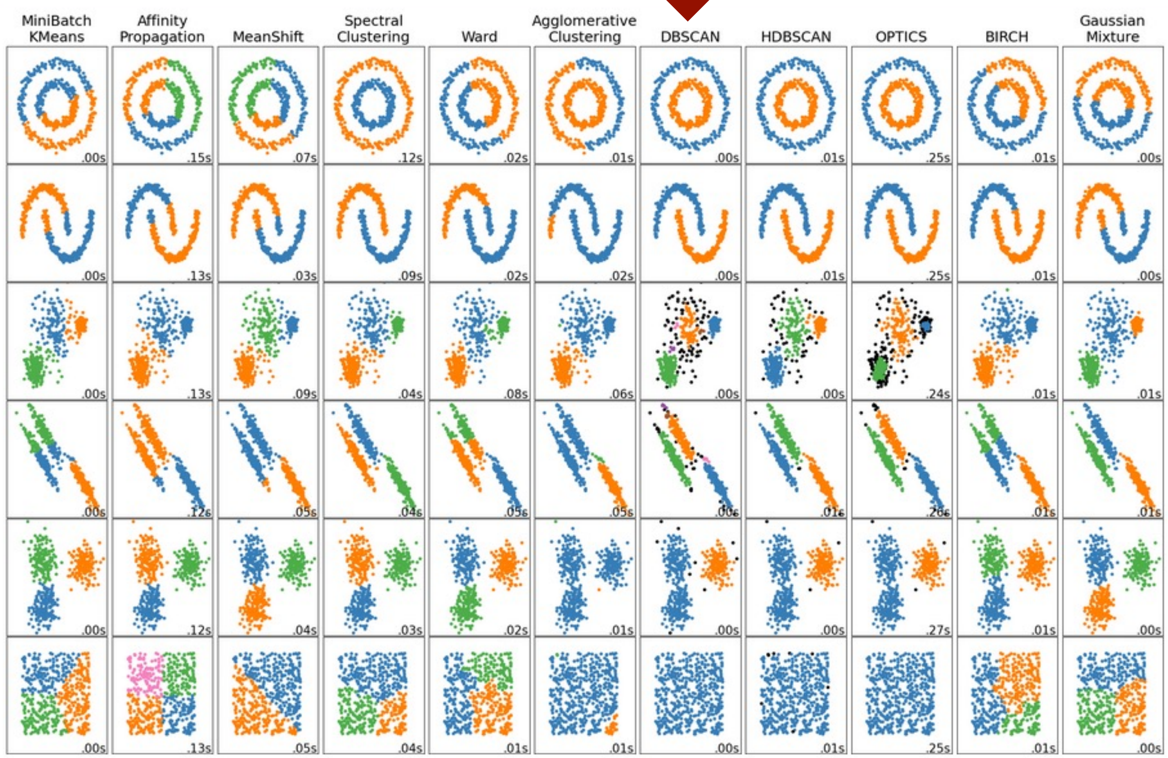
- Choose 'seeds' (how?)
- assign each vector to its closest seed (possibly adjusting cluster centroid)
- possibly, re-assign some vectors to improve clusters

Fast and practical, but 'unpredictable'

Cluster generation

one way to estimate # of clusters *k*: the 'cover coefficient' [Can+] ~ SVD

Visual overview



A comparison of the clustering algorithms in scikit-learn

DBSCAN

~single-link

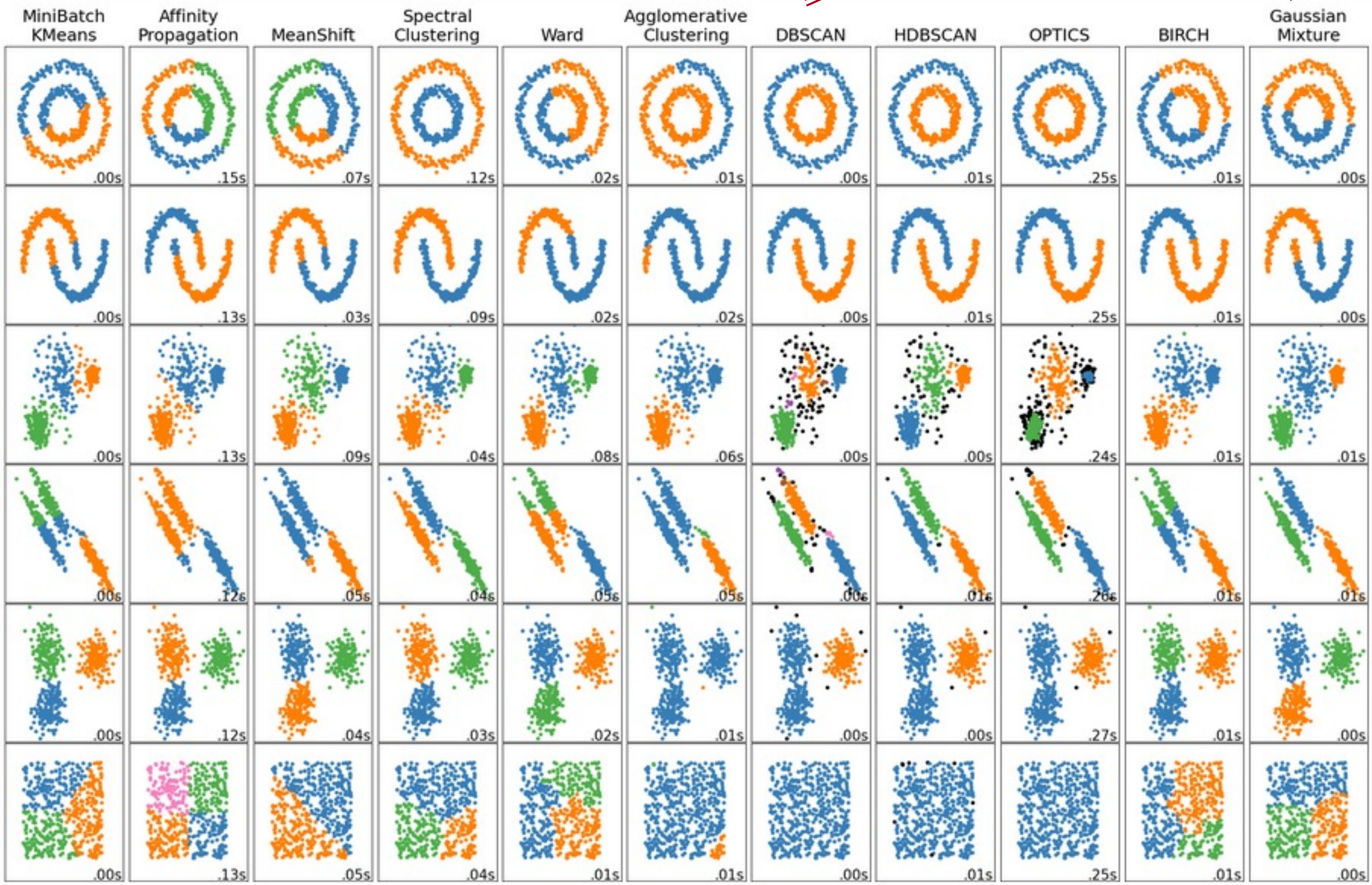
https://scikit-learn.org/stable/modules/clustering.html

~k-means

spectral

DBSCAN

Gaussian mixture



A comparison of the clustering algorithms in scikit-learn

Outline - detailed

- main idea
- cluster search
- cluster generation
- evaluation

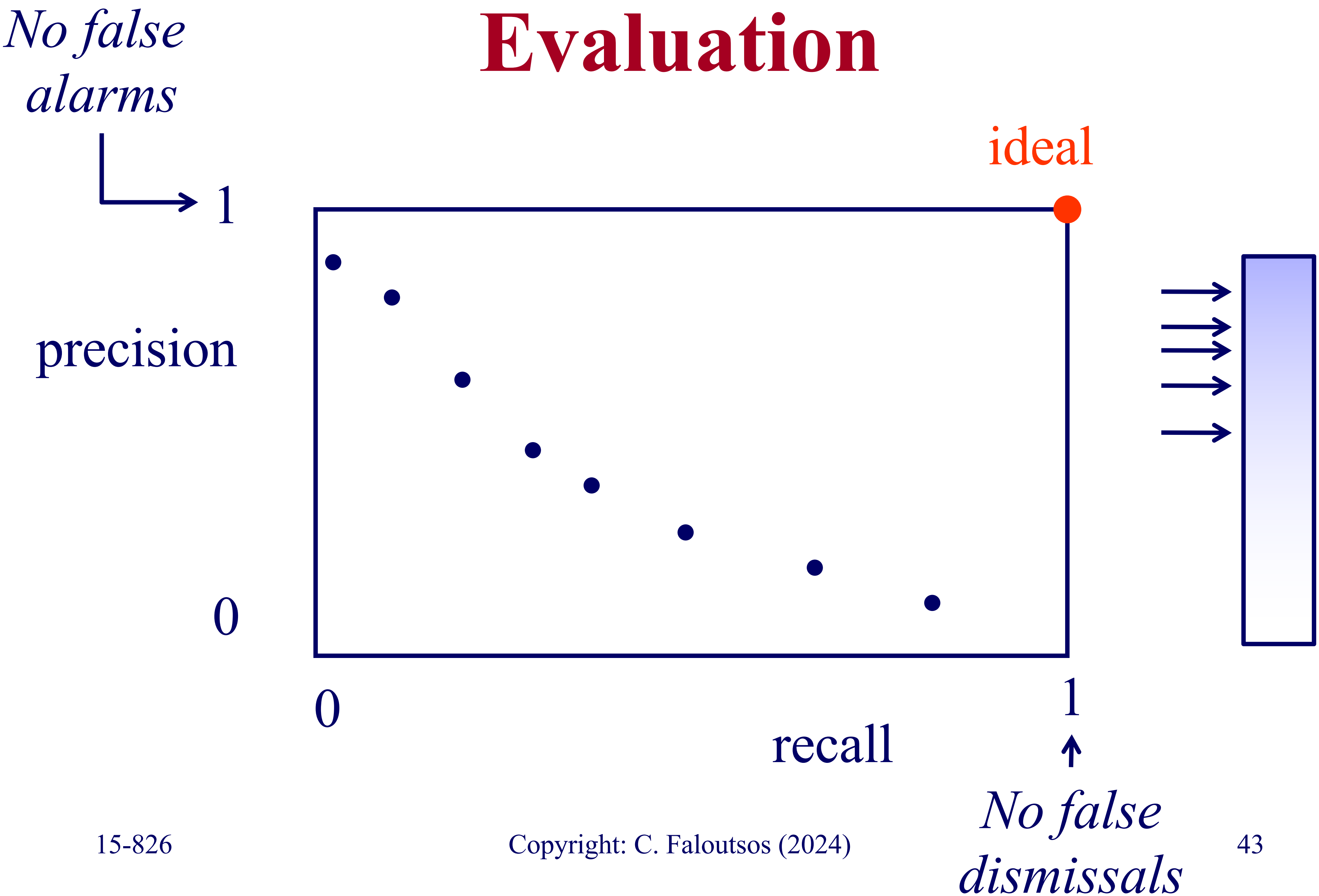
Evaluation

- Q: how to measure 'goodness' of one distance function vs another?
- A: ground truth (by humans) and
  - 'precision' and 'recall'

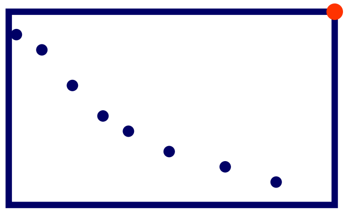
Evaluation

- precision = (retrieved & relevant) / retrieved
  - 100% precision -> no false alarms
- recall = (retrieved & relevant)/ relevant
  - 100% recall -> no false dismissals

Evaluation


No false
alarms
1
precision
0
ideal
0
1
recall
No false
dismissals

Evaluation

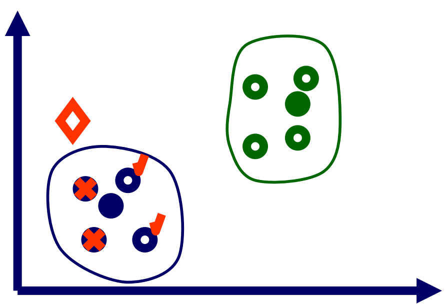


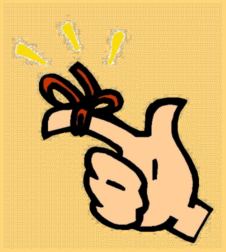
- compressing such a curve into a single number:
  - 11-point average precision
  - etc

Carnegie Mellon

Conclusions – main ideas

- 'bag of words' idea + keyword queries
- Cosine similarity
- Ranked output
- Relevance feedback





Conclusion

- How to find doc’s with “data mining”?
- A1: full text scanning
  - A1.1: string editing distance
- A2: inversion
  - Elias Codes
- (A3: signature files – ’Bloom filters’)
- A4: vector space model + clustering

References

- *Modern Information Retrieval* R. Baeza-Yates, Acm Press, Berthier Ribeiro-Neto, February 1999
- Can, F. and E. A. Ozkarahan (Dec. 1990). "Concepts and Effectiveness of the Cover-Coefficient-Based Clustering Methodology for Text Databases." ACM TODS 15(4): 483-517.
- Noreault, T., M. McGill, et al. (1983). A Performance Evaluation of Similarity Measures, Document Term Weighting Schemes and Representation in a Boolean Environment. Information Retrieval Research, Butterworths.

References

- Rocchio, J. J. (1971). Relevance Feedback in Information Retrieval. The SMART Retrieval System - Experiments in Automatic Document Processing. G. Salton. Englewood Cliffs, New Jersey, Prentice-Hall Inc.
- Salton, G. (1971). The SMART Retrieval System - Experiments in Automatic Document Processing. Englewood Cliffs, New Jersey, Prentice-Hall Inc.

References

- Salton, G. and M. J. McGill (1983). Introduction to Modern Information Retrieval, McGraw-Hill.
- Van-Rijsbergen, C. J. (1979). Information Retrieval. London, England, Butterworths.
- Zahn, C. T. (Jan. 1971). "Graph-Theoretical Methods for Detecting and Describing Gestalt Clusters." IEEE Trans. on Computers C-20(1): 68-86.