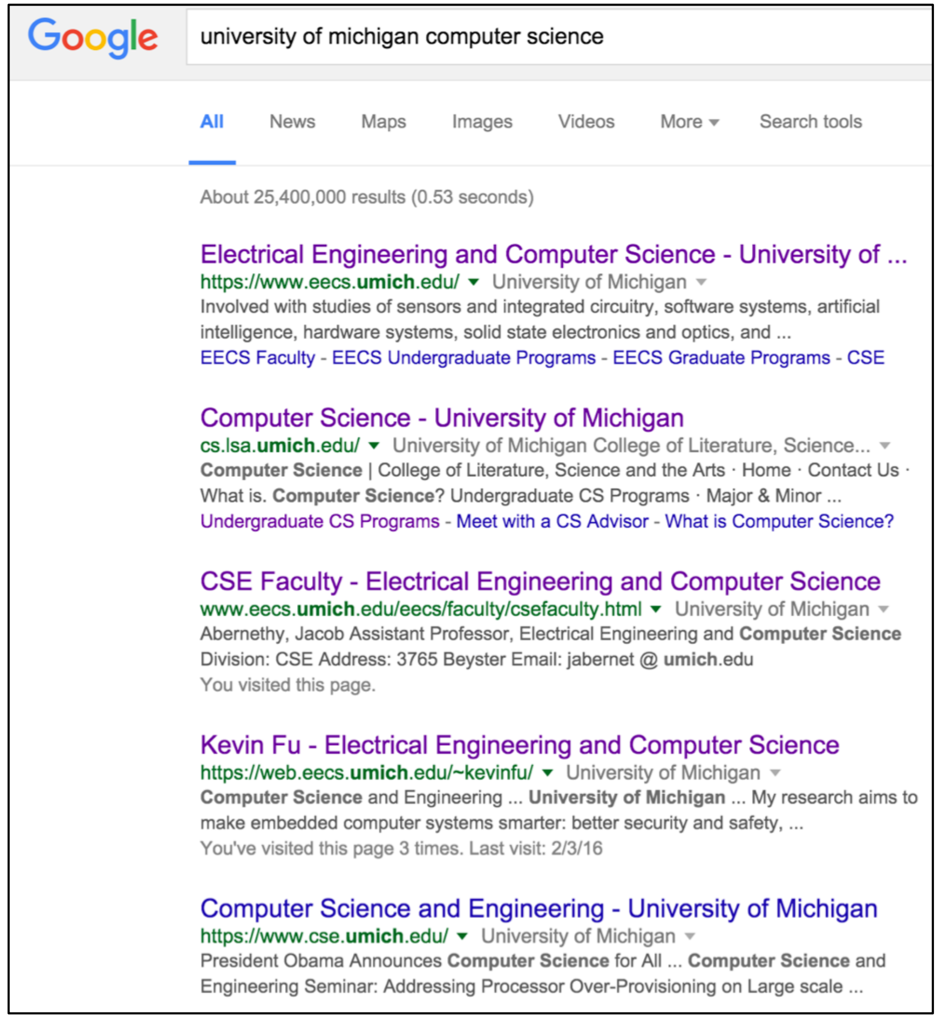
IR1: Introduction to Information Retrieval

Some slides due to Raghavan et al., via Dan Weld

Search Document Model

- 90% of users don't look beyond the first page of search results
- #1 result -> 33% clicks
- Top 3 results -> 61% clicks



Source: http://www.protofuse.com/blog/first-page-of-google-by-the-numbers/

Search Document Model

- Think of a web document as a tuple with several columns:
  - URL
  - Title
  - Page content
  - Unique docid
- Our goal: pick the best few tuples

Web Search and SQL

- We can think of web search like an SQL query

```
SELECT * FROM docs
WHERE docs.text LIKE 'userquery'
  AND docs.title LIKE 'userquery'
  AND //...
ORDER BY 'relevance'
```

- Where relevance is very complicated

Challenges

- Three challenges in web search:
  - Result relevance
  - Processing speed
  - Scaling to many documents
- We'll cover result relevance today and next lecture

How are pages ranked?

- Information retrieval is largely a study of how/why to prefer one page over another

- Boolean retrieval
- Vector-space model
- Cosine distance
- Assessing rank quality

Boolean Retrieval

- For each doc, two possible outcomes of query processing
  - TRUE or FALSE
  - “exact match” retrieval
  - Simplest form of ranking, used to be common
- Query specified with Boolean operators
  - AND, OR, NOT
  - Proximity operators (NEAR) also possible

Query

- Which plays of Shakespeare contain the words **Brutus** AND **Caesar** but NOT **Calpurnia**?
- Answer queries like this using a *term-document incidence*
- Basically, a table of Booleans

Term-document incidence

Which plays of Shakespeare contain the words **Brutus** AND **Caesar** but NOT **Calpurnia**?

	Tempest	Hamlet	Othello	Macbeth
Antony	0	0	0	1
Brutus	0	1	0	0
Caesar	0	1	1	1
Calpurnia	0	0	0	0
Cleopatra	0	0	0	0
mercy	1	1	1	1
worser	1	1	1	0

1 if **play** contains **word**,
0 otherwise

Beyond Term Search

- Phrases?
- Proximity: Gates NEAR Microsoft
  - Index should capture position info
- Subfields in documents:
  Find (**author**='Ullman') AND
  (**text** contains 'automata')

Ranking Search Results

- Boolean queries simply *include* or *exclude* a document from results
- That's fine with few hits
- Boolean is a good first pass, but we need to prefer some documents over others
- Another problem is that the user may not specify the critical term(s) correctly
  - Synonym
  - Spelling Error

Hit Counting

- We could simply measure the size of the overlap between the document and the query
  - How many query words "hit"?
- But what about:
  - Term frequency in document
  - Term scarcity in collection
  - Length of documents

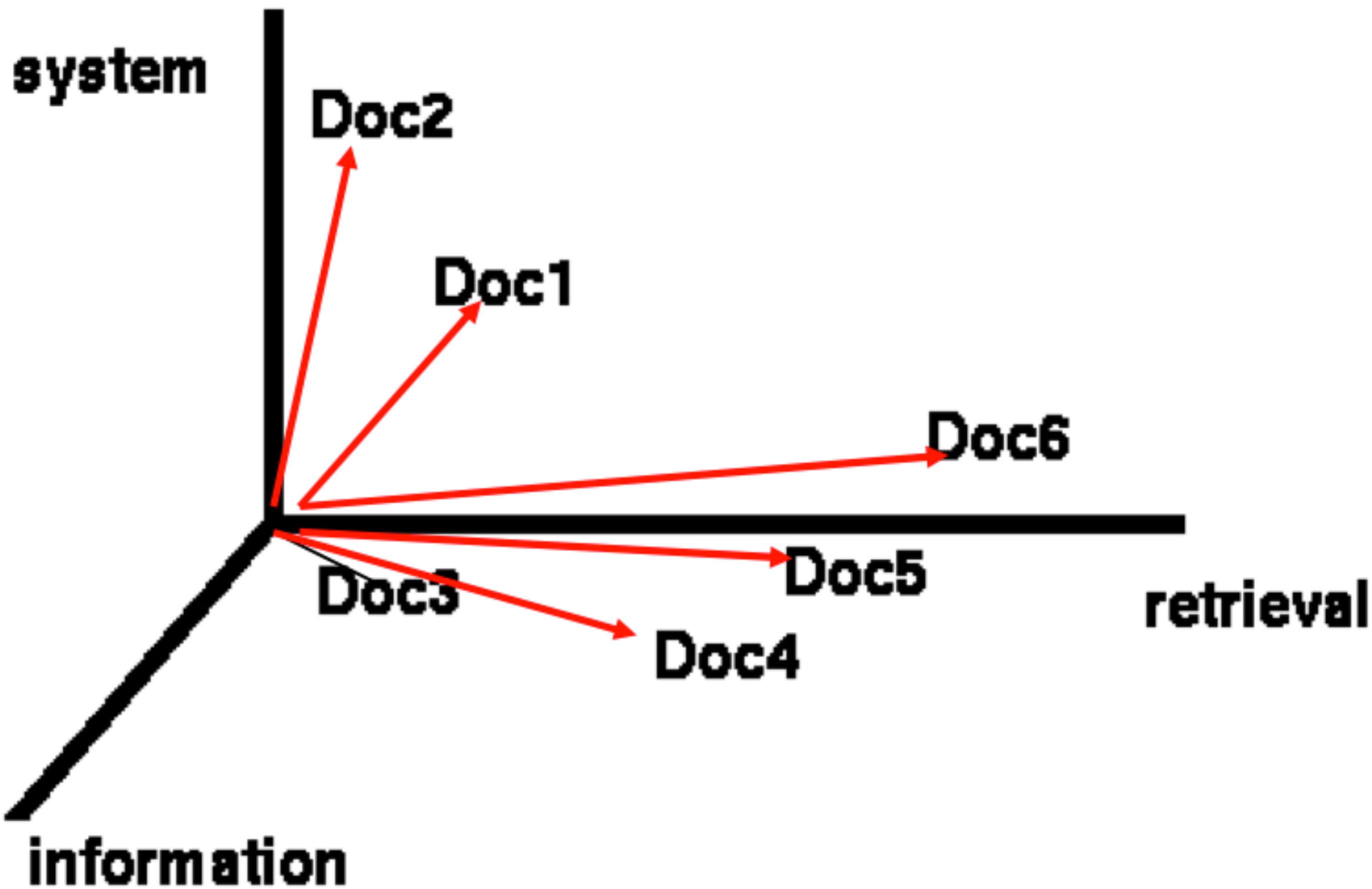
Documents as Vectors

- Each doc *j* can be viewed as a vector of term frequency *tf* values, one component for each term
- We thus have a *vector space*
  - A doc is a point in the space
- How many dimensions?
- Is this space sparse or dense? Why?

Documents as Vectors

- Each doc *j* can be viewed as a vector of term frequency *tf* values, one component for each term
- We thus have a *vector space*
  - A doc is a point in the space
- How many dimensions?
  - Dimension for every possible term (word)
- Is this space sparse or dense? Why?
  - Sparse. Most documents do not have most words.

Documents in 3D Space



- One assumption: documents that are "close together" in space are also close in meaning

Vector Space Query Model

1. Treat a query as a short document
2. Sort documents by increasing distance (decreasing similarity) to the query document
3. Easy to compute, as both query & doc are vectors

- First used in Salton's SMART system (1970). Now used by almost every information retrieval system

Vector Representation

- Docs and queries are vectors
- Position 1 corresponds to term 1
  Position *t* corresponds to term *t*
- Weight of term stored in each position

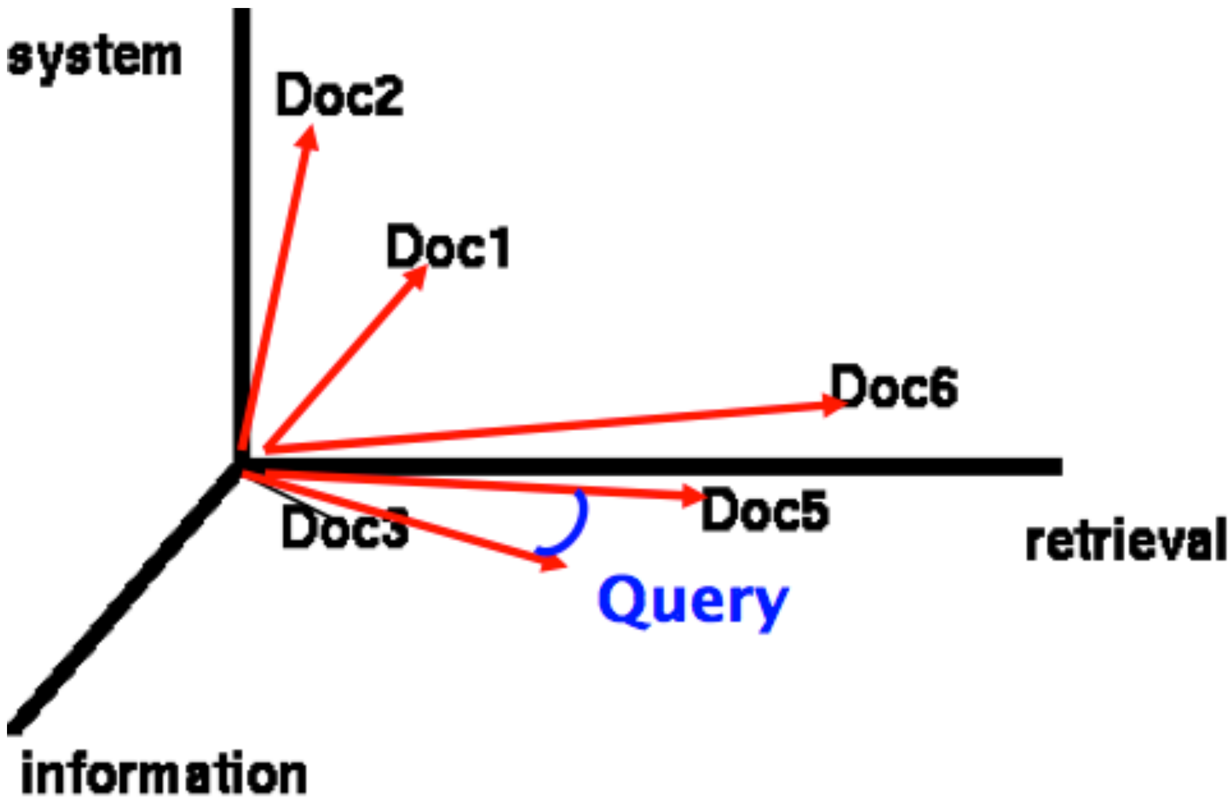
$$D_i = w_{d_{i1}}, w_{d_{i2}}, \ldots, w_{d_{it}}$$

$$Q = w_{q1}, w_{q2}, \ldots, w_{qt}$$

$$w = 0 \text{ if a term is absent}$$

Documents in 3D Space

- Term weights indicate length of document vector along a dimension



Computing Weights

- Which doc is a better match for the query "kangaroo", one with a single mention of "kangaroo", or a doc that mentions it 10 times?

- Find a few good examples and bad examples on the web for "kangaroo"
  - How many times does the word appear?

Computing Weights

- Which doc is a better match for the query "kangaroo", one with a single mention of "kangaroo", or a doc that mentions it 10 times?
- The doc that mentions it 10 times
- *Term frequency:* how many times the word appears in current document
- Higher is better

Computing Weights

- Which term is more indicative of document similarity, "Book" or "Rumpelstiltskin"?

Computing Weights

- Which term is more indicative of document similarity, "Book" or "Rumpelstiltskin"?
- Rumpelstiltskin
- *Document frequency:* how often a word appears in doc collection
- Lower is better

TF x IDF

- *Term-Frequency* x *Inverse-Document-Frequency*
  $\boldsymbol{W_{ik} = tf_{ik} * \log(N / n_k)}$
  - T_k = term k in document D_i
  - tf_{ik} = freq of term T_k in doc D_i
  - idf_k = inverse doc freq of term T_k in collection C
    $idf_k = \log(\frac{N}{n_k})$
  - N = total # docs in collection C
  - n_k = # docs in C that contain T_k

TF x IDF

- *Term-Frequency* x *Inverse-Document-Frequency*
  $\boldsymbol{W_{ik} = tf_{ik} * \log(N / n_k)}$
  - T_k = term k in document D_i
  - tf_{ik} = freq of term T_k in doc D_i
  - N = total # docs in collection C
  - n_k = # docs in C that contain T_k
- How would these affect the weight for a term T_k?
  - Large number of docs that contain T_k
  - Small number of docs that contain T_k
  - Large number of total documents
  - Small number of total documents

Inverse Document Frequency

- Inverse Document Frequency (IDF) provides high values for rare words, low values for common words

$$\log\left(\frac{10000}{10000}\right) = 0$$

$$\log\left(\frac{10000}{5000}\right) = 0.301$$

$$\log\left(\frac{10000}{20}\right) = 2.698$$

$$\log\left(\frac{10000}{1}\right) = 4$$

TF-IDF normalization

- Normalize term weights
  - Longer docs not given more weight
  - Force all values within [0,1]

$$w_{ik} = \frac{tf_{ik} \log(N / n_k)}{\sqrt{\sum_{k=1}^{t} (tf_{ik})^2 [\log(N / n_k)]^2}}$$

- Some references use non-normalized tf-idf
  - $w_{ik} = tf_{ik} \log(N/n_k)$

Vector space similarity

- Now, the similarity of two docs is:
  - Normalization done when computing term weights

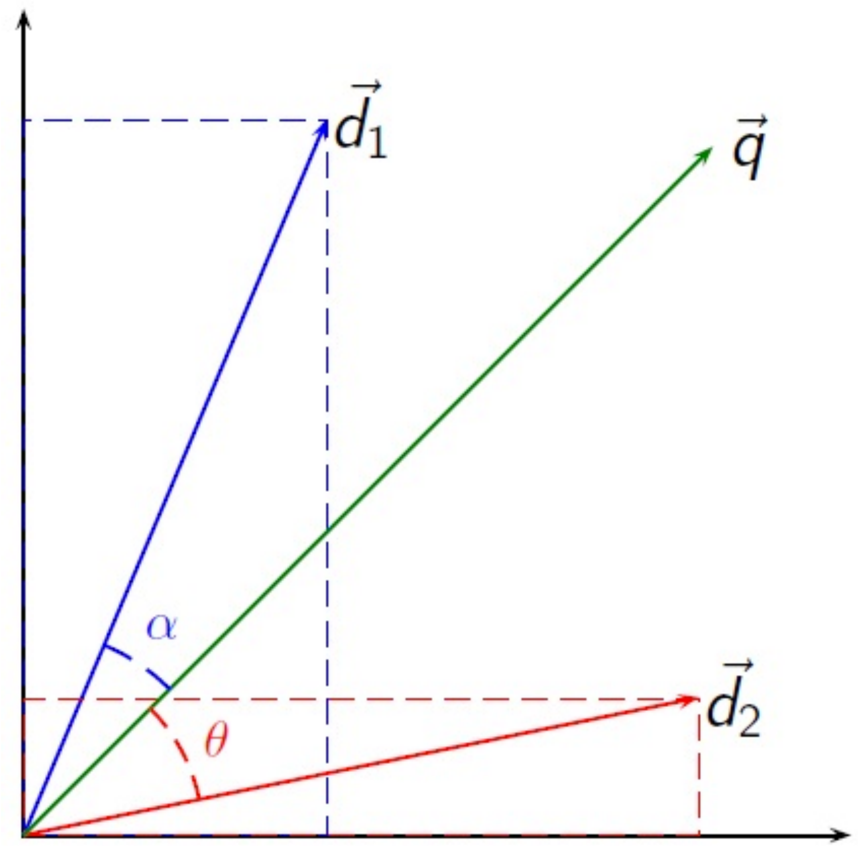
$$Sim(Di,Dj) = \sum_{k=1}^{t} w_{ik} * w_{jk}$$

  - Normalization not done when computing term weights

$$\mathrm{sim}(d_j, q) = \frac{\mathbf{d_j} \cdot \mathbf{q}}{\|\mathbf{d_j}\| \, \|\mathbf{q}\|} = \frac{\sum_{i=1}^{N} w_{i,j} w_{i,q}}{\sqrt{\sum_{i=1}^{N} w_{i,j}^2} \sqrt{\sum_{i=1}^{N} w_{i,q}^2}}$$

Vector space similarity

- Vector space similarity is also called the cosine, or normalized inner product
- Recall that cosine:
  - Depends on two adjacent vector lengths
  - =1 when angle is zero (points are identical)
  - Smaller when angle is greater



Computing a similarity score

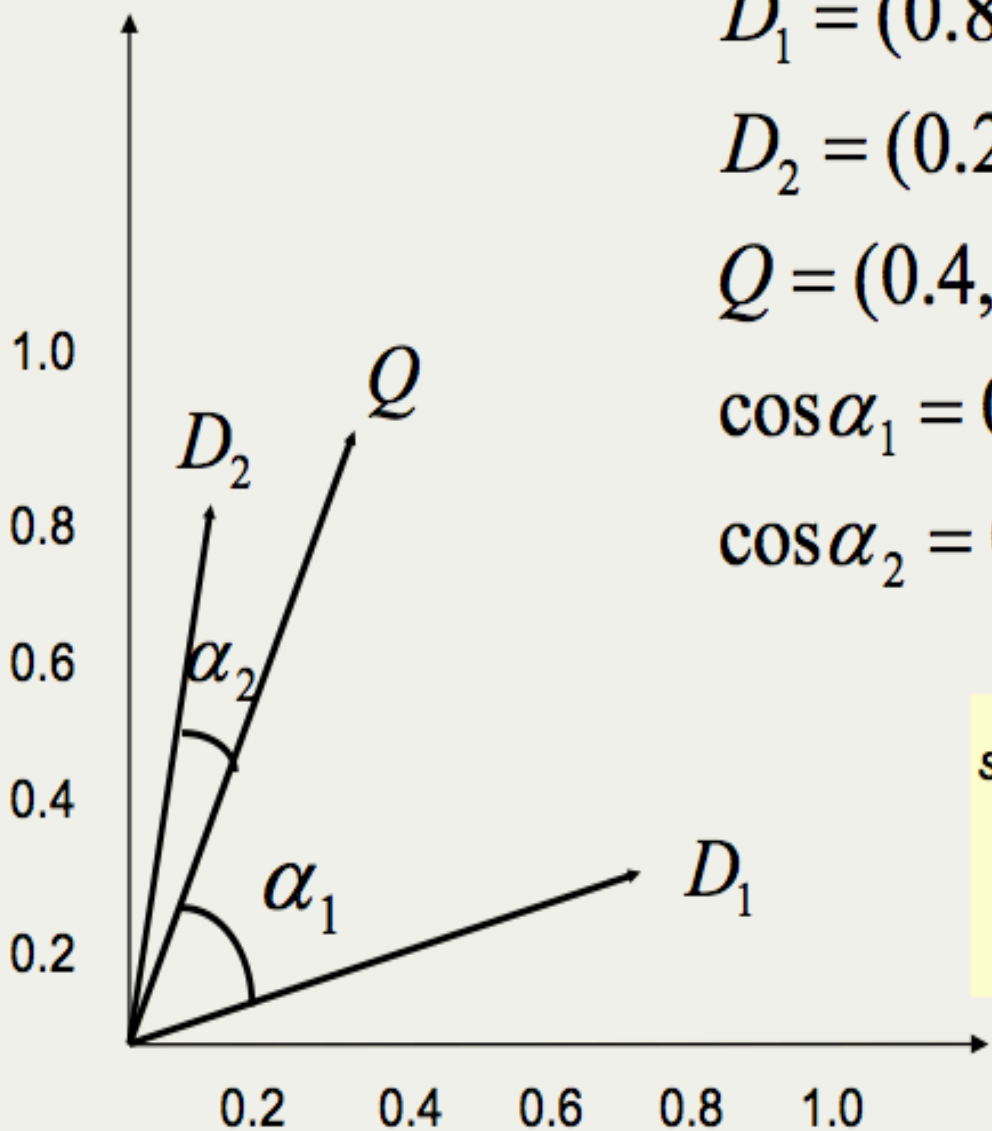
- Say we have query vector Q = (0.4,0.8)
- Also, document D_2 = (0.2,0.7)
- What is the result of the similarity computation?
- Assuming weights are *not* normalized

Computing a similarity score

- Say we have query vector Q = (0.4,0.8)
- Also, document D_2 = (0.2,0.7)
- What is the result of the similarity computation?
- Assuming weights are *not* normalized

$$sim(Q,D_2) = \frac{(0.4*0.2)+(0.8*0.7)}{\sqrt{[(0.4)^2+(0.8)^2]*[(0.2)^2+(0.7)^2]}}$$

$$= \frac{0.64}{\sqrt{0.42}} = 0.98$$



$$D_1 = (0.8, 0.3)$$

$$D_2 = (0.2, 0.7)$$

$$Q = (0.4, 0.8)$$

$$\cos\alpha_1 = 0.74$$

$$\cos\alpha_2 = 0.98$$

$$sim(Q, D_2) = \frac{(0.4*0.2) + (0.8*0.7)}{\sqrt{[(0.4)^2 + (0.8)^2]*[(0.2)^2 + (0.7)^2]}}$$

$$= \frac{0.64}{\sqrt{0.42}} = 0.98$$

Summary: Vector Spaces

- User’s query treated as short document
- Query is in same space as docs
- Easy to measure a doc’s distance to query
- Extension of Boolean retrieval

Assessing Quality

- You've built a ranker. How do you know if it's any good?
- Relevance has been studied for a long time
  - Many contributing factors
  - People disagree on what is relevant
- Retrieval/assessment models differ
  - Binary relevance vs sorted relevance
  - Query-relevance vs user-relevance

Assessing Quality

- Results from an experimental search engine
- Query: "Britney"
- URL 1: http://www.britneyspears.com

- URL 2: http://andrewdeorio.com

- URL 3: http://en.wikipedia.org/wiki/Britney_Spears

Assessing Quality

- Results from human "answer key"
- Query: "Britney"
- URL 1: http://www.britneyspears.com
- URL 2: http://en.wikipedia.org/wiki/Britney_Spears

  ...

- URL 90: http://andrewdeorio.com

Assessing Quality

- Results from an experimental search engine
- Query: "Britney"
- URL 1: http://www.britneyspears.com
  - Human answer: 1
- URL 2: http://andrewdeorio.com
  - Human answer: 90
- URL 3: http://en.wikipedia.org/wiki/Britney_Spears
  - Human answer: 2

Assessing Quality

- Results from an experimental search engine
  - Query: "Britney"
  - URLs: URL1, URL2, URL3, …
  - Rank: 1, 90, 2, ...
- Large hand-marked query/result tuples form the "answer key" for the ranker
- Text REtrieval Conference (TREC) is an annual conference, also publishes data
- Different tracks have included:
  - Blog track studies information-seeking
  - Chemical IR, Legal IR

Evaluating Search Ranking

- Precision/Recall Curves
- Kendall’s Tau

Positives and Negatives

- True positive
  - Relevant doc returned
- False positive
  - Irrelevant doc returned
- True negative
  - Irrelevant doc *not* returned
- False negative
  - Relevant doc *not* returned

Positives and Negatives

- Label these docs as TP, FP, TN, FN
  - Query = puppies
- Search results
  - Britney Spears (@britneyspears) Instagram photos
  - 10 Dog Breeds That Have The CUTEST Puppies
- Web pages not included in search results
  - Cats - Reddit
  - Puppy Bowl XI Highlights

Positives and Negatives

- Label these docs as TP, FP, TN, FN
  - Query = puppies
- Search results
  - Britney Spears (@britneyspears) Instagram photos **FP**
  - 10 Dog Breeds That Have The CUTEST Puppies **TP**
- Web pages not included in search results
  - Cats - Reddit **TN**
  - Puppy Bowl XI Highlights **FN**

Precision and recall

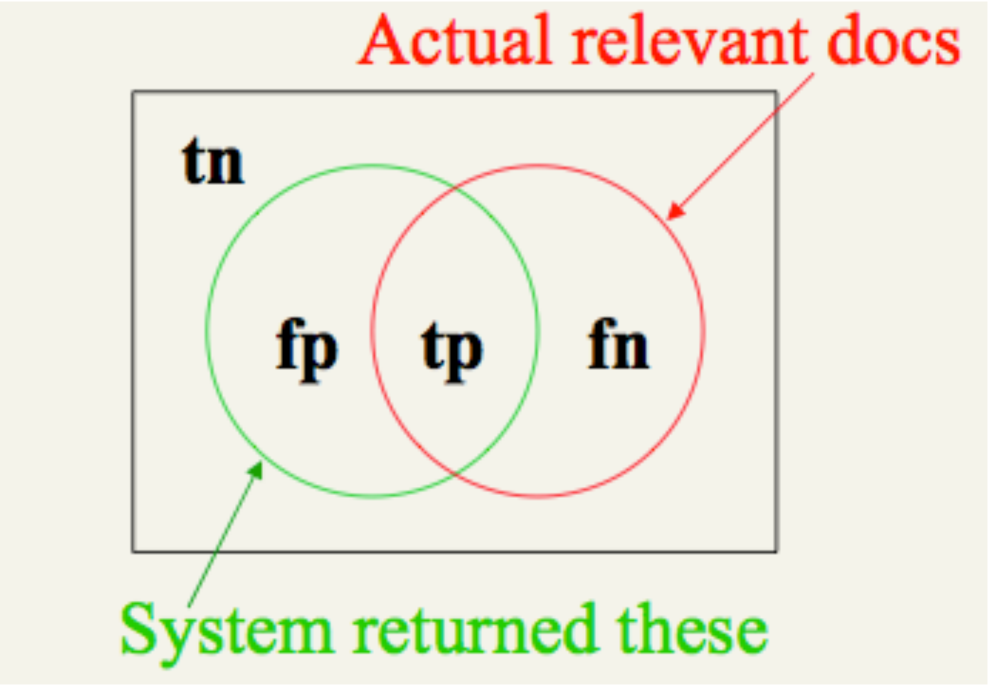
- Precision: fraction of retrieved docs that are relevant = relevant/retrieved
- Recall: fraction of relevant docs that are retrieved = retrieved/relevant

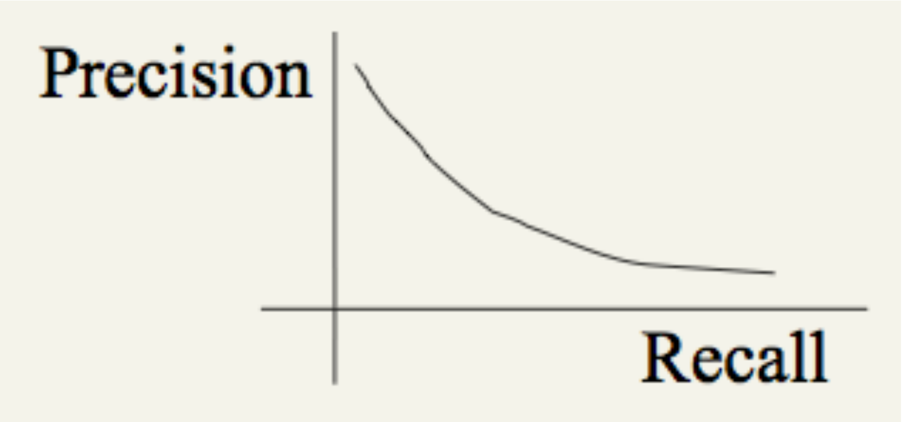
	Relevant	**Not Relevant**
Retrieved	TP	FP
Not retrieved	FN	TN

- Precision P = tp/(tp+fp)
- Recall R = tp/(tp+fn)

Precision and recall

- Precision:
  - % of selected items that are correct
- Recall:
  - % of correct items that are selected
- P/R curve shows tradeoff





Precision and recall

- Generally trade precision vs recall
  - How to get a system with high recall?
- Recall is a non-decreasing function of the # of docs retrieved
  - Precision **usually** decreases with more docs retrieved
- Drawbacks
  - Binary relevance
  - Need human judgments
  - Must average over large corpus
  - Alternatively, skewed by corpus/author selection

Exercise

- A search engine always returns all documents
- Do you expect high or low precision?
- Do you expect high or low recall?

Exercise

- A search engine always returns all documents

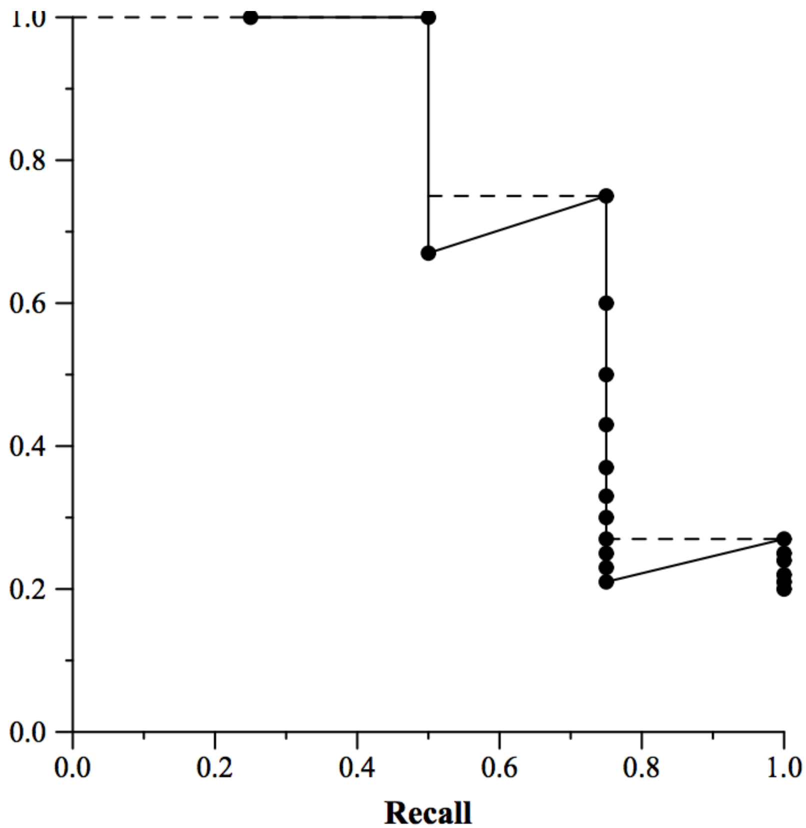
- Do you expect high or low precision?
  - Low. If all docs are returned, then many non-relevant docs are included, which will decrease the percentage of returned docs that are relevant.
- Do you expect high or low recall?
  - High. If all docs are returned, then all relevant docs must be returned.

Precision-recall curves

- A search engine will create a total ordering on all documents
- The top k are returned to the user
- We can calculate precision and recall for several values of k
- This creates a *precision-recall curve*

Precision-recall Example

- Collection of 20 documents
- Relevant docs are ranked 1, 2, 4, 15



Precision: % of selected items that are correct

Recall: % of correct items that are selected

http://trec.nist.gov/pubs/trec15/appendices/CE.MEASURES06.pdf

Take Ranking Into Account

- Precision at fixed recall
  - Precision of top k results, for k=1,10,50,…
  - Critical for Web Search
- Kendall's Tau for comparing sorts

Kendall's Tau

- Use a real ordering of documents, not just binary "relevant/not relevant"
- The correct document ordering is:
  - 1, 2, 3, 4
- Search Engine A outputs:
  - 1, 2, 4, 3
- Search Engine B outputs:
  - 4, 3, 1, 2
- Intuitively, A is better. How do we capture this numerically?

Measuring Rank Correlation

- Kendall's Tau has some nice properties:
  - If agreement between 2 ranks is perfect, then KT = 1
  - If disagreement is perfect,
    then KT = -1
  - If rankings are uncorrelated, then KT = 0 on average
- Intuition: Compute fraction of pairwise orderings that are consistent

Kendall's Tau

 # pairs that agree

 # pairs that disagree

$$\tau = \frac{n_c - n_d}{\frac{1}{2}n(n-1)}$$

total # pairs

- The non-normalized version is called Kendall's Tau Distance
- Also called *bubble-sort distance*

Try it out

- Correct ordering:
  - 1, 2, 3, 4
- Search Engine A:
  - 1, 2, 4, 3

$$\tau = \frac{5-1}{\frac{1}{2}4(4-1)} = \frac{4}{6} = 0.666$$

- Search Engine B:
  - 4, 3, 2, 1

Compute this one

Try it out

- Correct ordering:
  - 1, 2, 3, 4
- Search Engine A:
  - 1, 2, 4, 3

$$\tau = \frac{5-1}{\frac{1}{2}4(4-1)} = \frac{4}{6} = 0.666$$

- Search Engine B:
  - 4, 3, 2, 1

$$\tau = \frac{0-6}{\frac{1}{2}4(4-1)} = \frac{-6}{6} = -1$$

Mean Reciprocal Rank

- We have a query set Q, plus hand-marked search results
- "How close to the top of the search results is the 1st correct answer?"

$$meanreciprocalrank = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$$

- Strengths? Weaknesses?

Mean Reciprocal Rank

$$meanreciprocalrank = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$$

Windy City?	Toronto, Chicago, NYC
Tree City?	Ann Arbor, Madison, Capital City
Emerald City?	Vancouver, San Francisco, Seattle
MRR	

Mean Reciprocal Rank

$$meanreciprocalrank = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$$

Query	**Results**	**Rank**	**Reciprocal rank**
Windy City?	Toronto, Chicago, NYC	2	1/2
Tree City?	Ann Arbor, Madison, Capital City	1	1
Emerald City?	Vancouver, San Francisco, Seattle	3	1/3
MRR			**0.611**

Ranking Assessment

- Requires lots of hand-judged data
- Precision & Recall
  - Usually trade off each other
  - With a ranker, can generate PR curve
  - Requires relevant/not-relevant judgments
- Kendall's Tau
  - Measures correlation between two rankings
  - +1 if perfect agreement; -1 disagreement
  - Measure "fraction of pairs in agreement"