Inspire…Educate…Transform.

TEXT MINING - DAY 1

Dr. Manoj Duse
Mar 9, 2019

Agenda

- Introduction to Text Mining
- Indexing & Search
- Inverted Index
- BOW
- N-grams
- Text to “vector space models”
- TD-IDF
- SVD [may be]

What is Text Mining ?

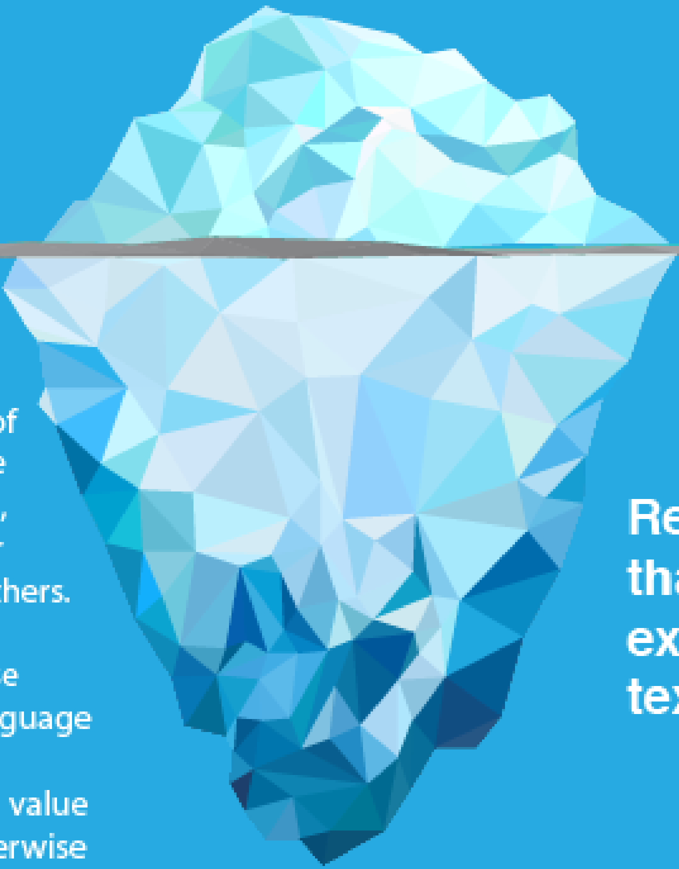
TEXT MINING, ROUGHLY EQUIVALENT TO TEXT ANALYTICS, IS THE PROCESS OF DERIVING HIGH QUALITY INFORMATION (INSIGHTS) FROM TEXT

WHY TEXT ANALYTICS ???

Unstructured VS Structured Data

20%

of unstructured data is utilized for company use.



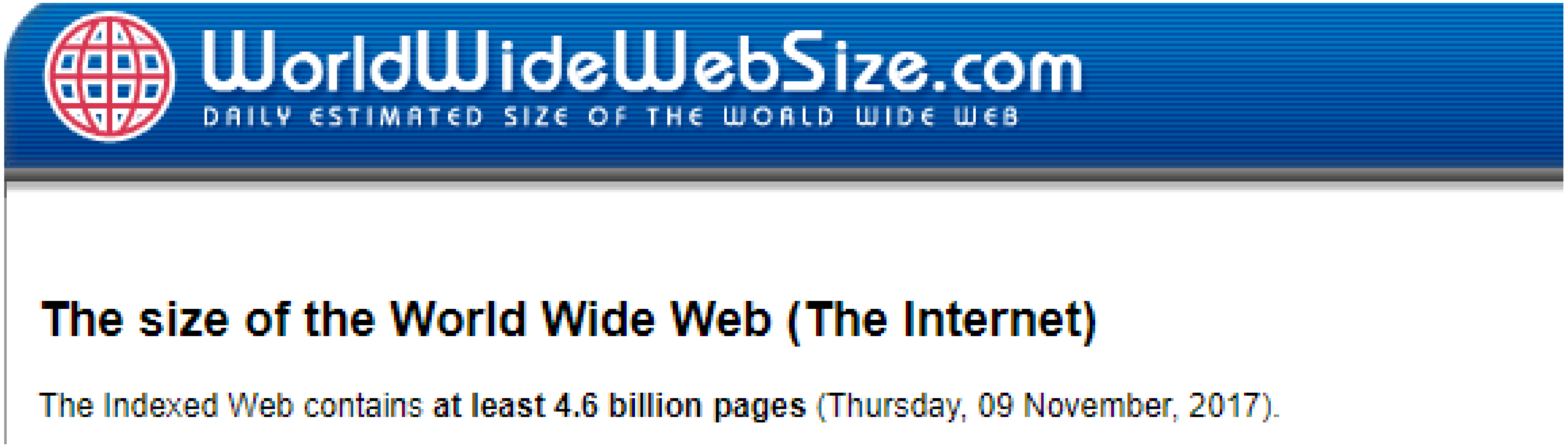
Text mining has become an important part of data analytics, attempting to determine the context surrounding the wealth of information that individuals leave across chats, social posts, reviews, surveys, call transcripts, customer emails, and news feeds among others. Through the advent of powerful, artificially intelligent tools that use machine learning and natural language processing for textual analysis, businesses are gaining additional value from information that would otherwise go unused.

80%

Research has shown that nearly 80% of data exists as unstructured textual data.

Stratifyd.com

The largest application on the planet

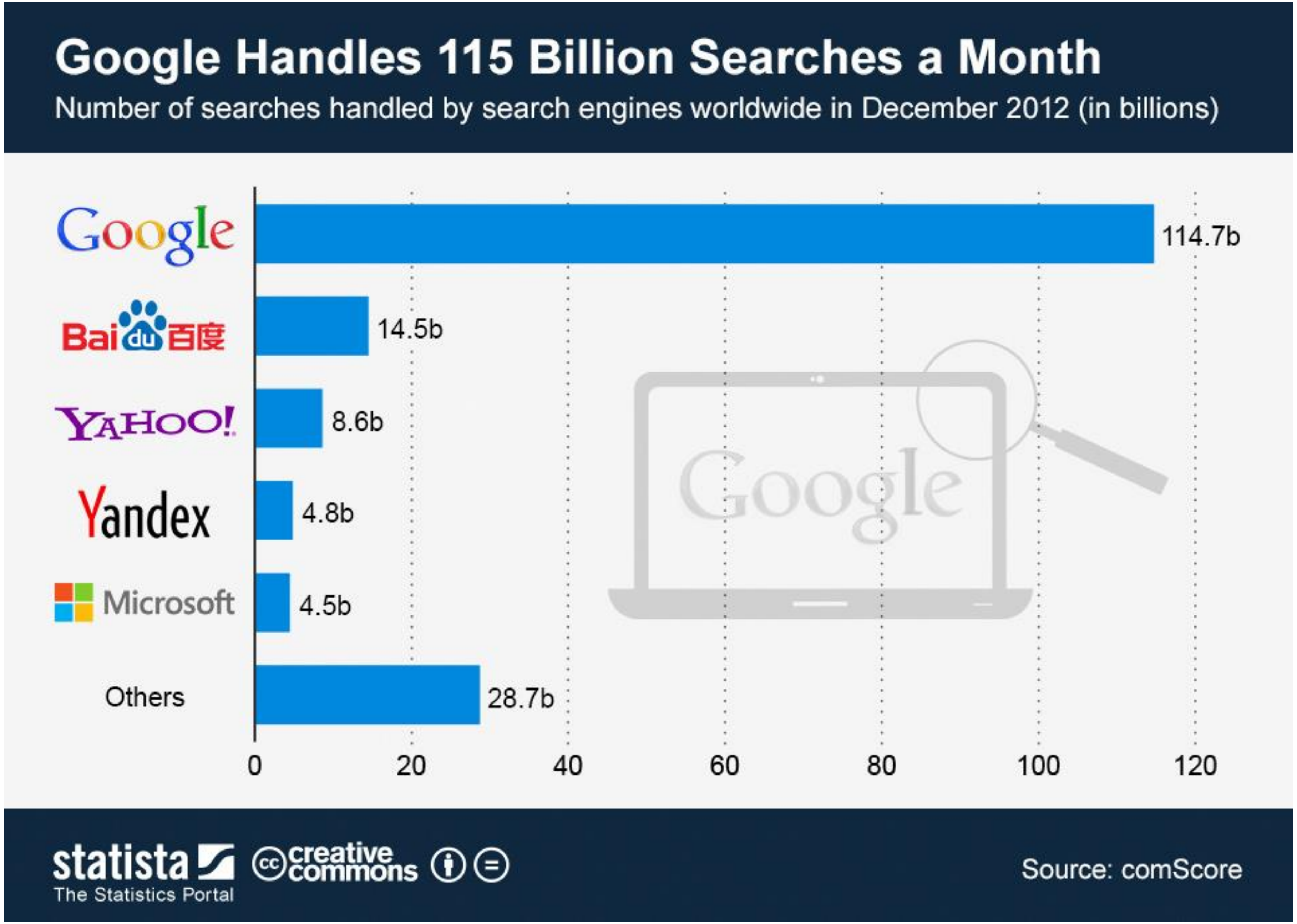

WorldWideWebSize.com

DAILY ESTIMATED SIZE OF THE WORLD WIDE WEB

The size of the World Wide Web (The Internet)

The Indexed Web contains **at least 4.6 billion pages** (Thursday, 09 November, 2017).

http://www.worldwidewebsize.com/



https://www.statista.com/chart/898/number-of-searches-handled-by-search-engines-worldwide/

Growing as we speak….

Internet Live Stats [a site that measures Internet statistic] estimates that every second there are at least :

7000	Tweets sent,
1140	Tumblr posts posted online,
733	Photos posted on Instagram,
2207	Skype calls,
55k	Google searches,
127k	YouTube videos viewed, and

over 2 million emails sent.

THERE IS MUCH MORE TO TEXT MINING THAN SEARCH

Why is this spam?

From: "" <takworlld@hotmail.com>
Subject: real estate is the only way... gem oalvgkay

Anyone can buy real estate with no money down

Stop paying rent TODAY !

There is no need to spend hundreds or even thousands for similar courses

I am 22 years old and I have already purchased 6 properties using the methods outlined in this truly INCREDIBLE ebook.

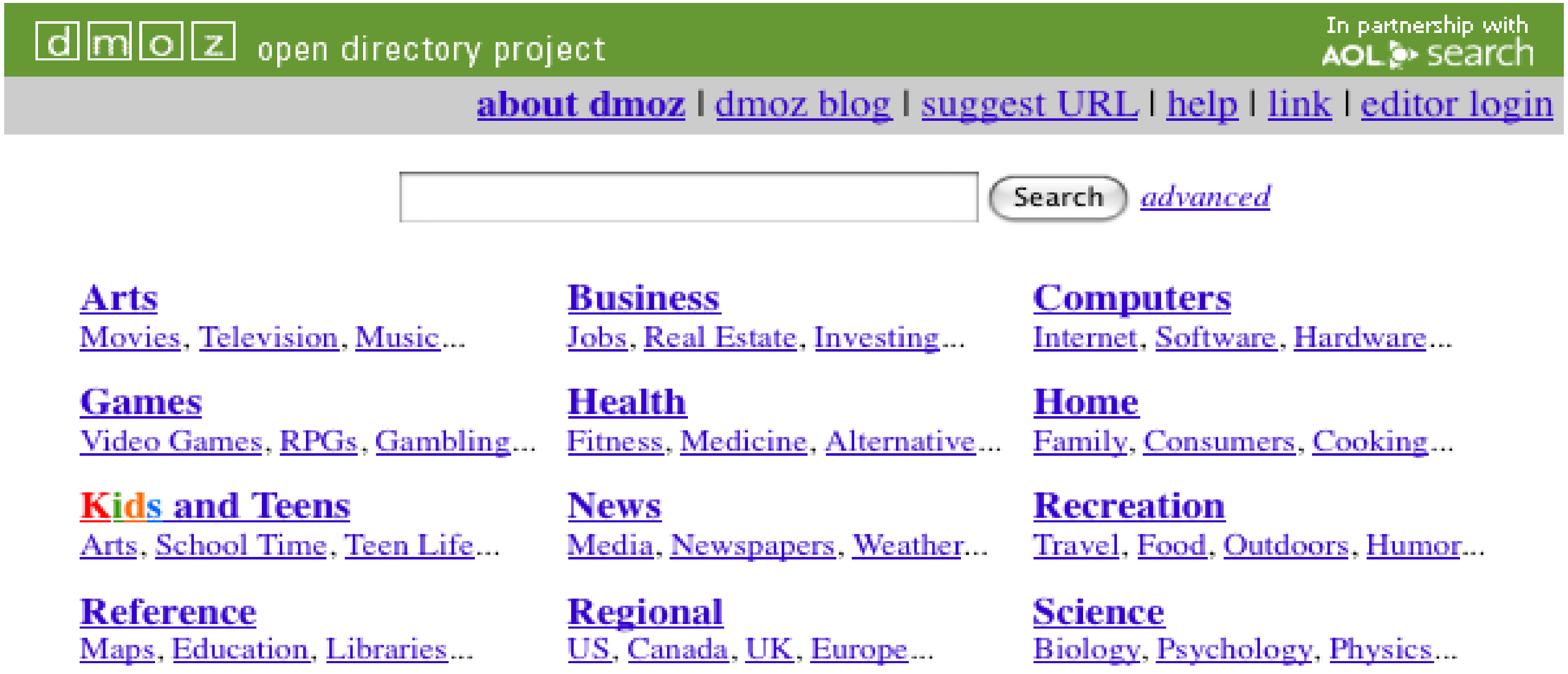
Change your life NOW !

=================================
Click Below to order:
http://www.wholesaledaily.com/sales/nmd.htm
=================================

Is this positive or negative sentiment?

"I bought an iPhone a few days ago. It was such a nice phone. The touch screen was really cool. The voice quality was clear too. Although the battery life was not long, that is ok for me. However, my mother was mad with me as I did not tell her before I bought the phone. She also thought the phone was too expensive, and wanted me to return it to the shop. …"

Categorization


dmoz open directory project

In partnership with AOL search

about dmoz | dmoz blog | suggest URL | help | link | editor login

Arts
Movies, Television, Music...

Business
Jobs, Real Estate, Investing...

Computers
Internet, Software, Hardware...

Games
Video Games, RPGs, Gambling...

Health
Fitness, Medicine, Alternative...

Home
Family, Consumers, Cooking...

Kids and Teens
Arts, School Time, Teen Life...

News
Media, Newspapers, Weather...

Recreation
Travel, Food, Outdoors, Humor...

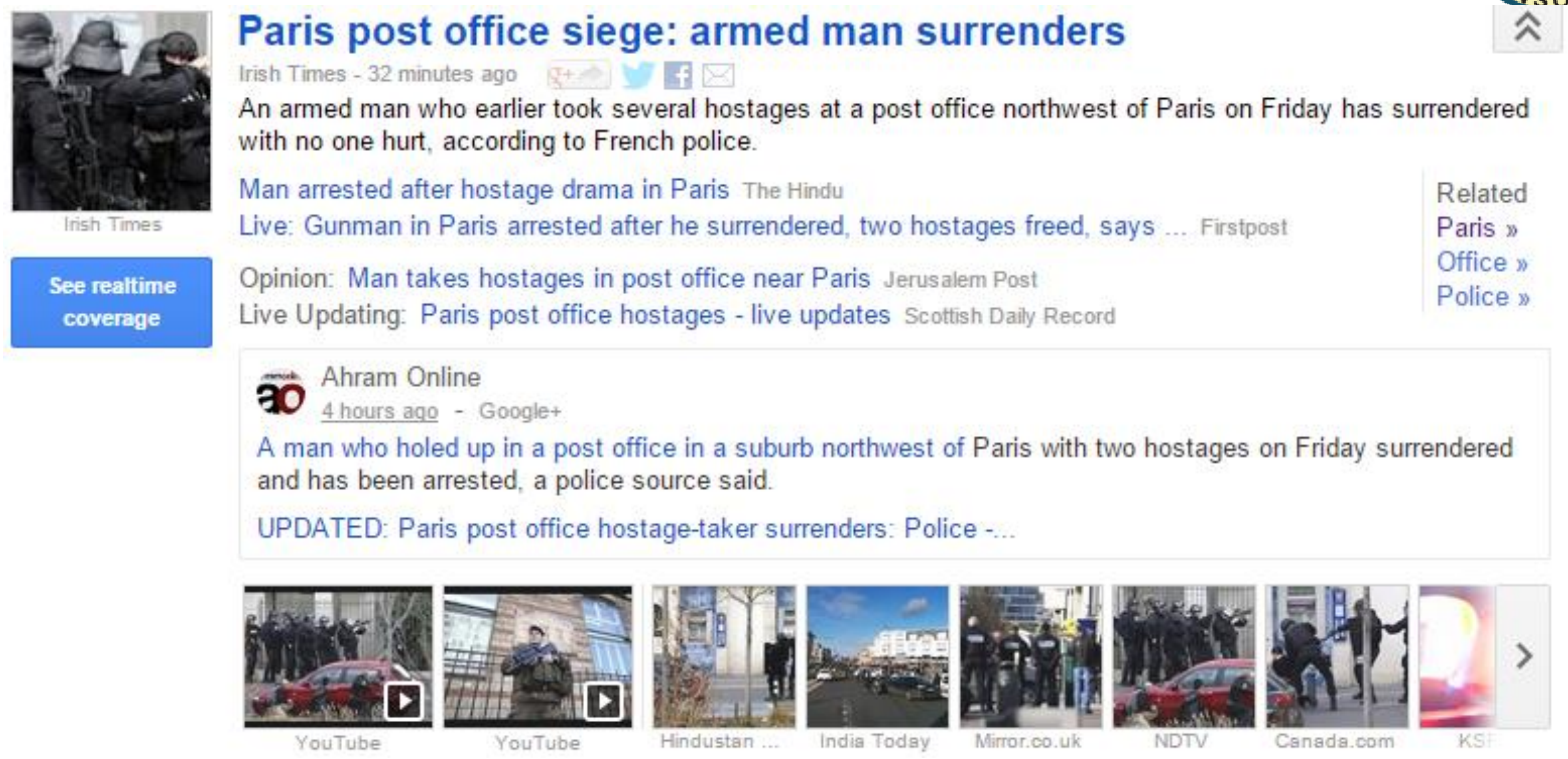
Reference
Maps, Education, Libraries...

Regional
US, Canada, UK, Europe...

Science
Biology, Psychology, Physics...

http://www.dmoz.org

Clustering



Paris post office siege: armed man surrenders

Irish Times - 32 minutes ago

An armed man who earlier took several hostages at a post office northwest of Paris on Friday has surrendered with no one hurt, according to French police.

Man arrested after hostage drama in Paris The Hindu

Live: Gunman in Paris arrested after he surrendered, two hostages freed, says ... Firstpost

Opinion: Man takes hostages in post office near Paris Jerusalem Post

Live Updating: Paris post office hostages - live updates Scottish Daily Record

Related
Paris »
Office »
Police »

Irish Times

See realtime coverage

Ahram Online
4 hours ago - Google+

A man who holed up in a post office in a suburb northwest of Paris with two hostages on Friday surrendered and has been arrested, a police source said.

UPDATED: Paris post office hostage-taker surrenders: Police -...

This company that makes Listerine & Sudafed was actually started by 3 brothers, not just 2.

What is Johnson & Johnson?

IBM Watson aces Jeopardy!

https://www.youtube.com/watch?v=WFR3lOm_xhE

Examples of Language Processing Applications

Text Mining	Language Processing
Spam filtering	Siri (Apple) and Google Now
Document Classification	Language Understanding
Date/time event detection	Spelling Correction
Information Extraction	Statistical Language Modeling
(Web) Search engines	Website translation (Google)
Information Retrieval	Machine Translation
Watson in Jeopardy! (IBM)	"Clippy" Assistant (Microsoft)
Question Answering	Dialog System
Twitter brand monitoring	Finding similar items (Amazon)
Sentiment Analysis (Stat. NLP)	Recommender System

Philosophy

- Statistical analysis of words
- Convert the text into a structured form
- Use statistical or data mining algorithms

TEXT RETRIEVAL: ALGORITHMS

Step 1: Preprocessing

- Implement the preprocessing functions:
  - Tokenization
  - Stop word removal
  - Stemming
  - etc.

- <u>Input</u>: Documents that are read one by one from the collection

- <u>Output</u>: Tokens to be added to the index
  - No punctuation, no stop-words, stemmed

Step 2: Indexing

- Build an inverted index, with an entry for each word in the vocabulary
- Input: Tokens obtained from the preprocessing module
- Output: An inverted index for fast access

Step 3: Retrieval

- Use inverted index to find the limited subset of documents that contain at least one of the query words.
- Incrementally compute cosine similarity of each indexed document as query words are processed one by one.
- To accumulate a total score for each retrieved document, store retrieved documents, where the document id is the key, and the partial accumulated score is the value.

- Input: Query and Inverted Index
- Output: Similarity values between query and documents

Step 4: Ranking

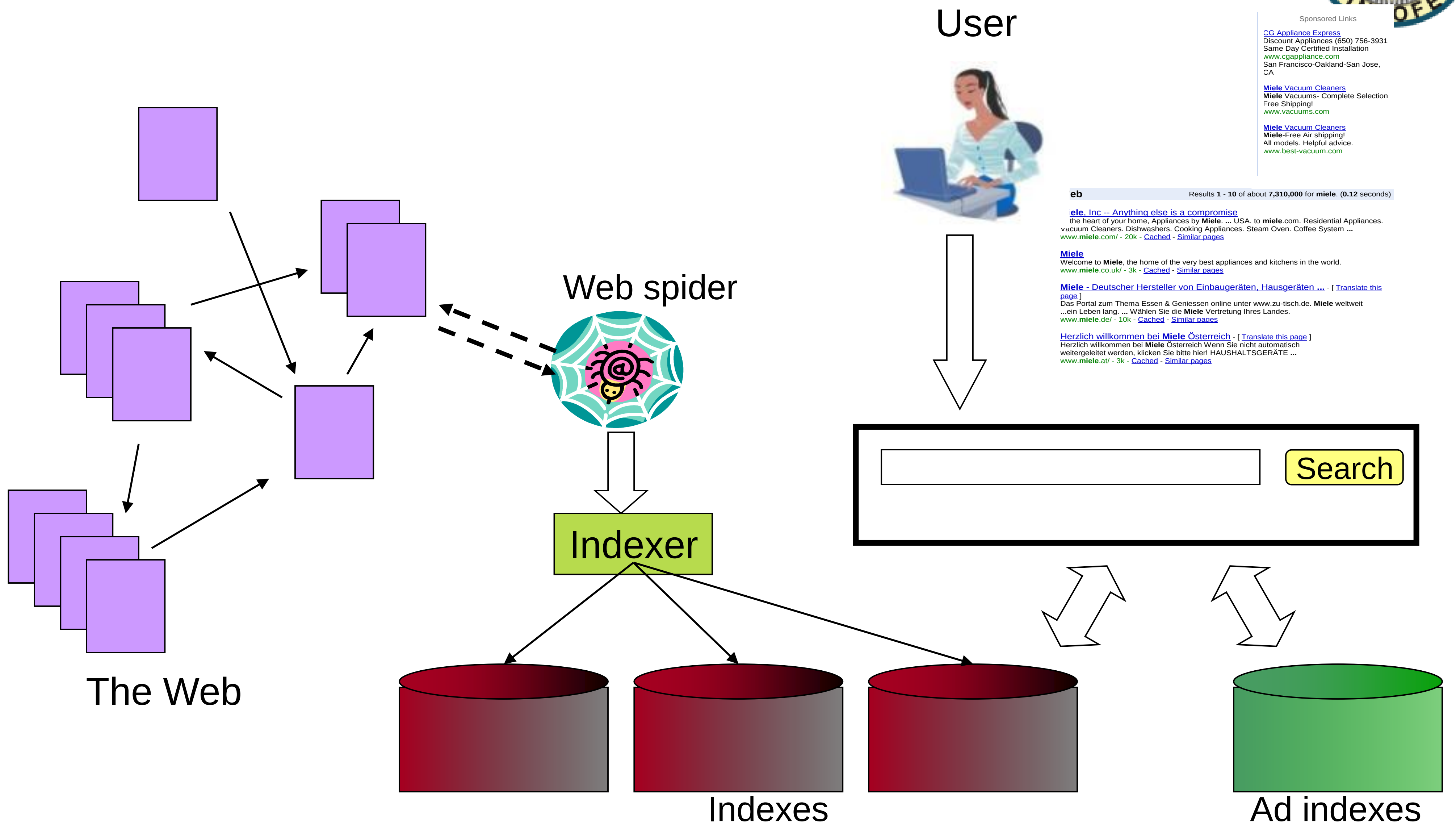
- Sort the retrieved documents based on the value of cosine similarity

- Return the documents in descending order of relevance

- Input: Similarity values between query and documents
- Output: Ranked list of documents [most relevant first]

STEP 0: CRAWLING

Web search basics



Sponsored Links

CG Appliance Express
Discount Appliances (650) 756-3931
Same Day Certified Installation
www.cgappliance.com
San Francisco-Oakland-San Jose, CA

Miele Vacuum Cleaners
Miele Vacuums- Complete Selection
Free Shipping!
www.vacuums.com

Miele Vacuum Cleaners
Miele-Free Air shipping!
All models. Helpful advice.
www.best-vacuum.com

eb Results **1** - **10** of about **7,310,000** for **miele**. (**0.12** seconds)

iele, Inc -- Anything else is a compromise
the heart of your home, Appliances by **Miele**. **...** USA. to **miele**.com. Residential Appliances.
vacuum Cleaners. Dishwashers. Cooking Appliances. Steam Oven. Coffee System **...**
www.**miele**.com/ - 20k - Cached - Similar pages

Miele
Welcome to **Miele**, the home of the very best appliances and kitchens in the world.
www.**miele**.co.uk/ - 3k - Cached - Similar pages

Miele - Deutscher Hersteller von Einbaugeräten, Hausgeräten ... - [Translate this page]
Das Portal zum Thema Essen & Geniessen online unter www.zu-tisch.de. **Miele** weltweit
...ein Leben lang. **...** Wählen Sie die **Miele** Vertretung Ihres Landes.
www.**miele**.de/ - 10k - Cached - Similar pages

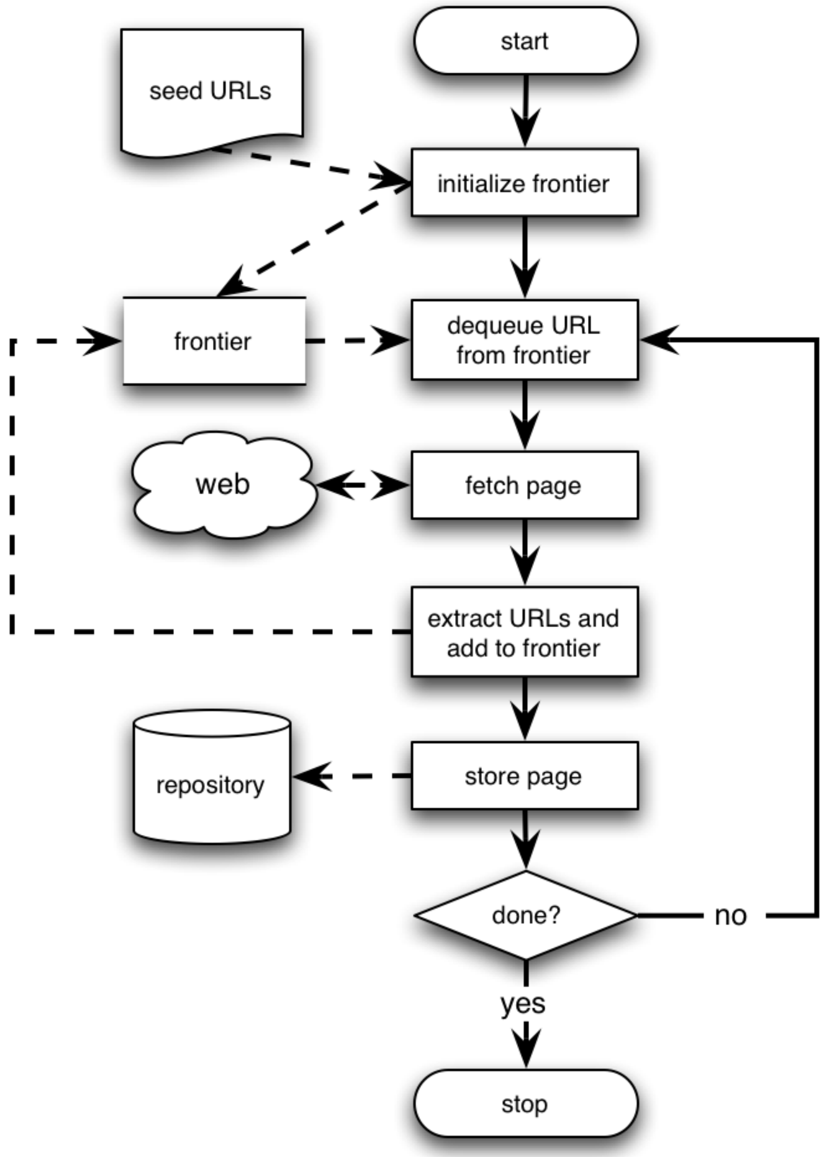
Herzlich willkommen bei **Miele** Österreich - [Translate this page]
Herzlich willkommen bei **Miele** Österreich Wenn Sie nicht automatisch
weitergeleitet werden, klicken Sie bitte hier! HAUSHALTSGERÄTE **...**
www.**miele**.at/ - 3k - Cached - Similar pages

Many names

- Crawler
- Spider
- Robot (or bot)
- Web agent
- Wanderer, worm, …
- And famous instances: googlebot, scooter, slurp, msnbot, …

Crawlers: Introduction

- Web crawler, spider
- Definition:
  - A Web crawler is a computer program that browses the World Wide Web in a methodical, automated manner. (Wikipedia)
- Utilities:
  - Gather pages from the Web.
  - Support a search engine, perform data mining and so on.
- Object:
  - Text, video, image, audio…
  - Link structure



Basic crawlers

- This is a sequential crawler
- Seeds can be any list of starting URLs
- Order of page visits is determined by frontier data structure
- Stop criterion can be anything

Features of a crawler

- Must provide:
  - Robustness: spider traps
    - Infinitely deep directory structures: http://foo.com/bar/foo/bar/foo/...
    - Pages filled a large number of characters.
  - Politeness: which pages can be crawled, and which cannot
    - robots exclusion protocol: robots.txt
    - http://blog.sohu.com/robots.txt
      - User-agent: *
      - Disallow: /manage/

Resilient

Show Respect

- Should be:
  - Distributed
  - Scalable
  - Performance and efficiency
  - Quality
  - Freshness
  - Extensible

www.apple.com/robots.txt

```
# robots.txt for http://www.apple.com/

User-agent: *
Disallow:
```

All crawlers…

…can go anywhere!

www.microsoft.com/robots.txt

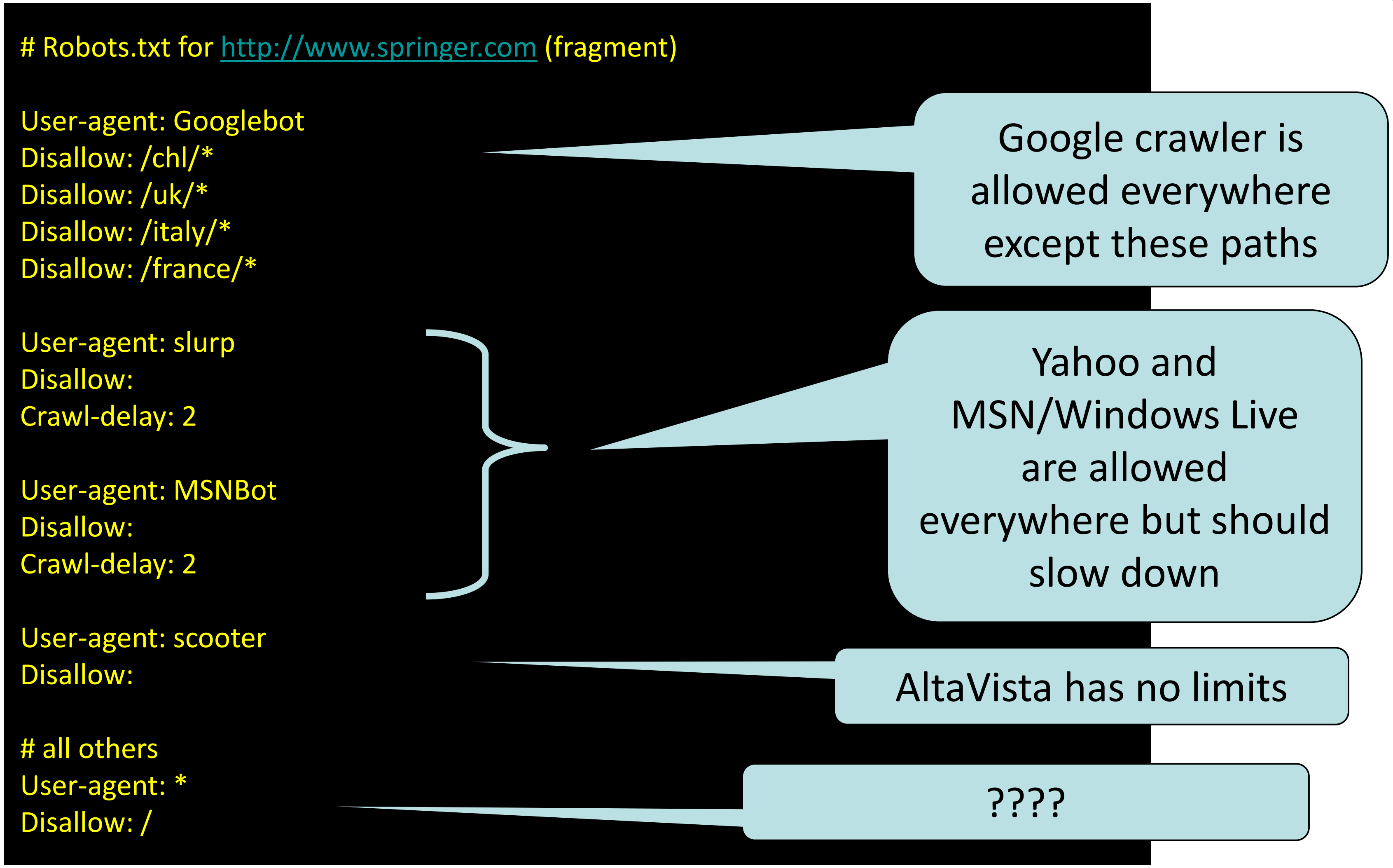
```
# Robots.txt file for http://www.microsoft.com

User-agent: *
Disallow: /canada/Library/mnp/2/aspx/
Disallow: /communities/bin.aspx
Disallow: /communities/eventdetails.mspx
Disallow: /communities/blogs/PortalResults.mspx
Disallow: /communities/rss.aspx
Disallow: /downloads/Browse.aspx
Disallow: /downloads/info.aspx
Disallow: /france/formation/centres/planning.asp
Disallow: /france/mnp_utility.mspx
Disallow: /germany/library/images/mnp/
Disallow: /germany/mnp_utility.mspx
Disallow: /ie/ie40/
Disallow: /info/customerror.htm
Disallow: /info/smart404.asp
Disallow: /intlkb/
Disallow: /isapi/
#etc…
```

All crawlers…

…are not allowed in these paths…

www.springer.com/robots.txt



```
# Robots.txt for http://www.springer.com (fragment)

User-agent: Googlebot
Disallow: /chl/*
Disallow: /uk/*
Disallow: /italy/*
Disallow: /france/*

User-agent: slurp
Disallow:
Crawl-delay: 2

User-agent: MSNBot
Disallow:
Crawl-delay: 2

User-agent: scooter
Disallow:

# all others
User-agent: *
Disallow: /
```

Google crawler is allowed everywhere except these paths

Yahoo and MSN/Windows Live are allowed everywhere but should slow down

AltaVista has no limits

????

Web Data Mining by Bing Liu

Crawlers (continued)

- Other issues:
  - Housekeeping tasks:
    - Log crawl progress statistics: URLs crawled, frontier size, etc. (Every few seconds)
    - Check-pointing: a snapshot of the crawler’s state (the URL frontier) is committed to disk. (Every few hours)
  - Priority of URLs in URL frontier:
    - Change rate
    - Quality
  - Restrictions not respected, then you are in trouble:
    - Blocked ☹

- Coverage
  - New pages get added all the time
  - Can the crawler find every page?
- Freshness
  - Pages change over time, get removed, etc.
  - How frequently can a crawler revisit ?
- Trade-off!
  - Focus on most “important” pages (crawler bias)?
  - “Importance” is subjective

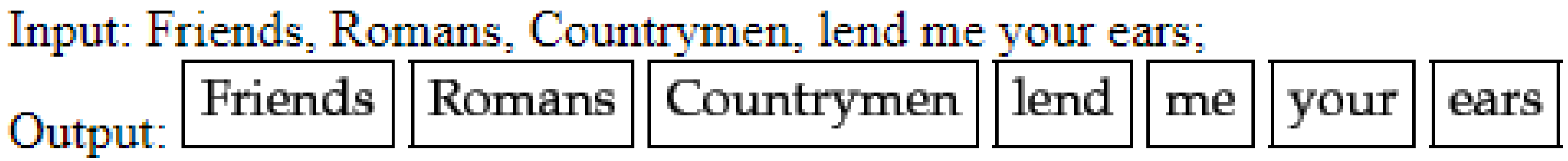
STEP 1: PREPROCESSING

Tokenization

- Analyze text into a sequence of discrete tokens (words).
- Sometimes punctuation (“e-mail”, “a.out”), numbers (“1999”), and case (“Congress” vs. “congress”) can be a meaningful part of a token.
  - However, frequently they are not.
- Simplest approach: ignore all numbers and punctuation and use only case-insensitive unbroken strings of alphabetic characters as tokens.
- More careful approach:
  - Separate ? ! ; : “ ‘ [ ] ( ) < >
  - Care with .
  - Care with other punctuation marks
  - New Delhi ?

Tokenization (continued)

- Example:



Input: Friends, Romans, Countrymen, lend me your ears;

Output: Friends | Romans | Countrymen | lend | me | your | ears

- Downloadable tool:
  - Word Splitter
    http://l2r.cs.uiuc.edu/~cogcomp/atool.php?tkey=WS

Case Folding

- Reduce all letters to lower case
  - exception: upper case in mid-sentence
    - *e.g.,* ***General Motors***
    - ***Fed*** vs. ***fed***
    - ***SAIL*** vs. ***sail***

Tokenizing HTML

- Should text in HTML commands not typically seen by the user be included as tokens?
  - Words appearing in URLs.
  - Words appearing in “meta text” of images.
- Simplest approach is to exclude all HTML tag information (between “<“ and “>”) from tokenization. But could lose critical information.

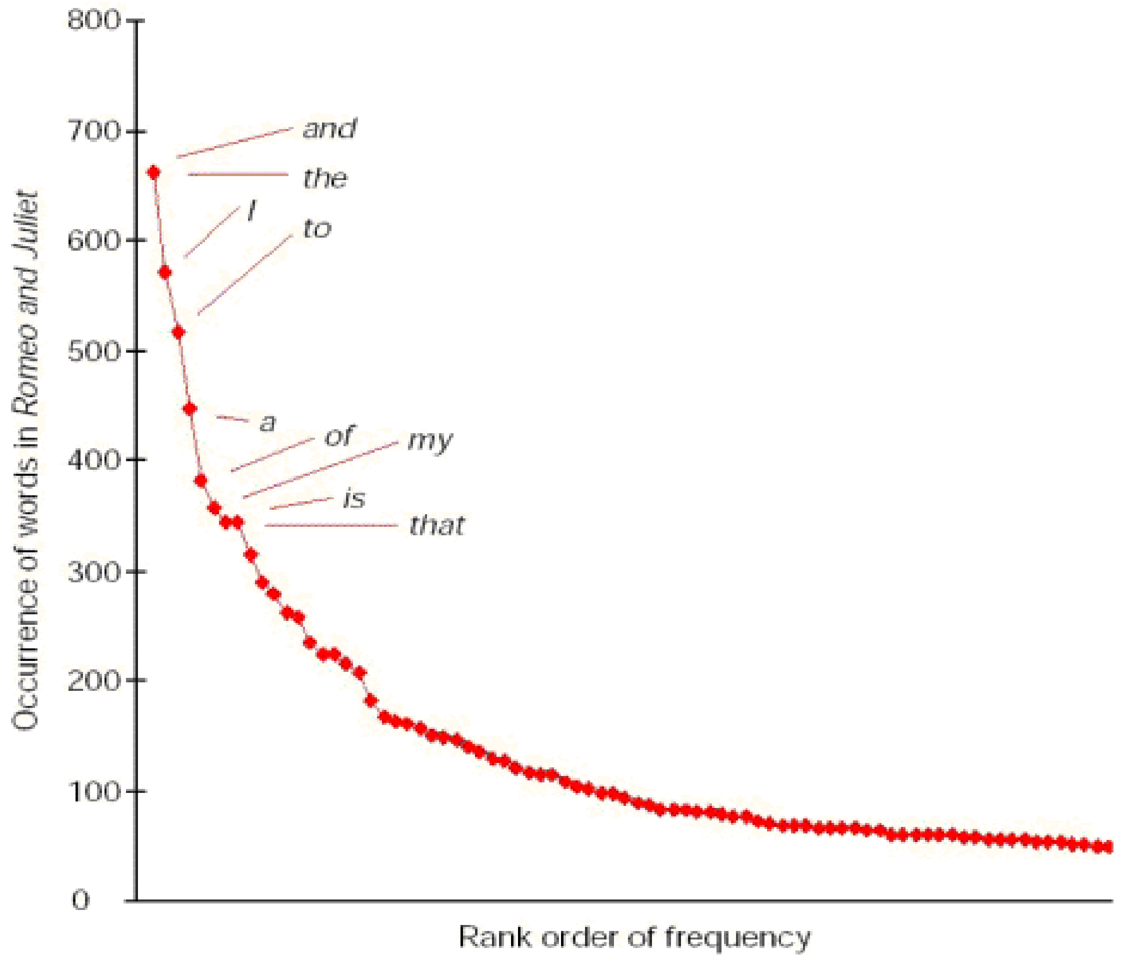
Stop-words

- What are these? - Some extremely common words which would appear to be of little value in helping select documents matching a user need are excluded from the vocabulary entirely.

- The general strategy for determining a stop list is to sort the terms by *collection frequency* (the total number of times each term appears in the document collection), and then to take the most frequent terms as a *stop list* , the members of which are then discarded during indexing.

- Why do this : Using a stop list significantly reduces the number of postings that a system has to store;

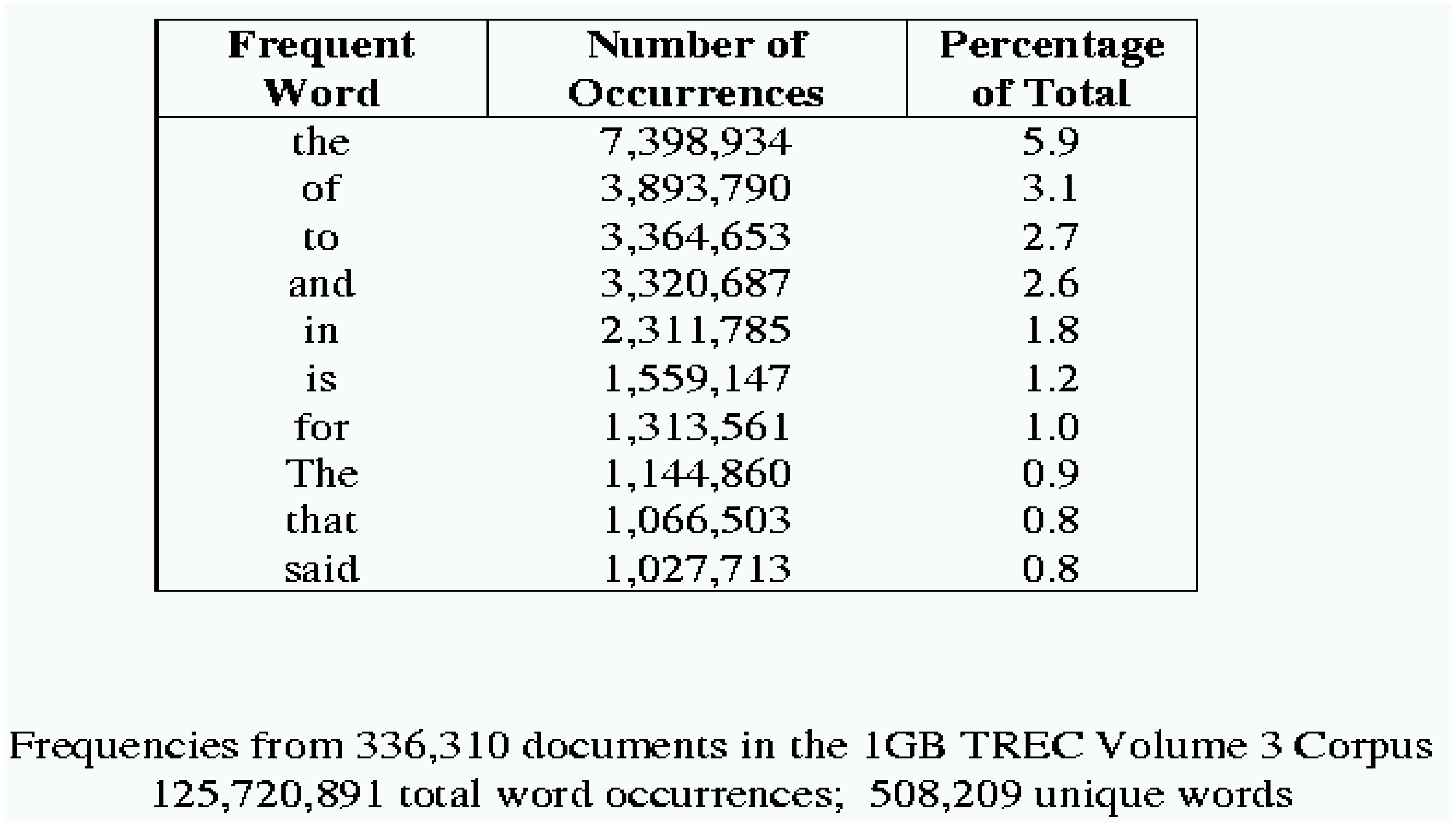
Words are not all equal



- A few words are very common.
  - 2 most frequent words (e.g. “the”, “of”) can account for about 10% of word occurrences.
- Most words are very rare.
  - Half the words in a corpus appear only once, called *hapax legomena* (Greek for “read only once”)
- Called a “heavy tailed” distribution, since most of the probability mass is in the “tail”

Sample Word Frequency Data

(from B. Croft, UMass)



Frequent Word	Number of Occurrences	Percentage of Total
the	7,398,934	5.9
of	3,893,790	3.1
to	3,364,653	2.7
and	3,320,687	2.6
in	2,311,785	1.8
is	1,559,147	1.2
for	1,313,561	1.0
The	1,144,860	0.9
that	1,066,503	0.8
said	1,027,713	0.8

Frequencies from 336,310 documents in the 1GB TREC Volume 3 Corpus
125,720,891 total word occurrences; 508,209 unique words

Stopwords

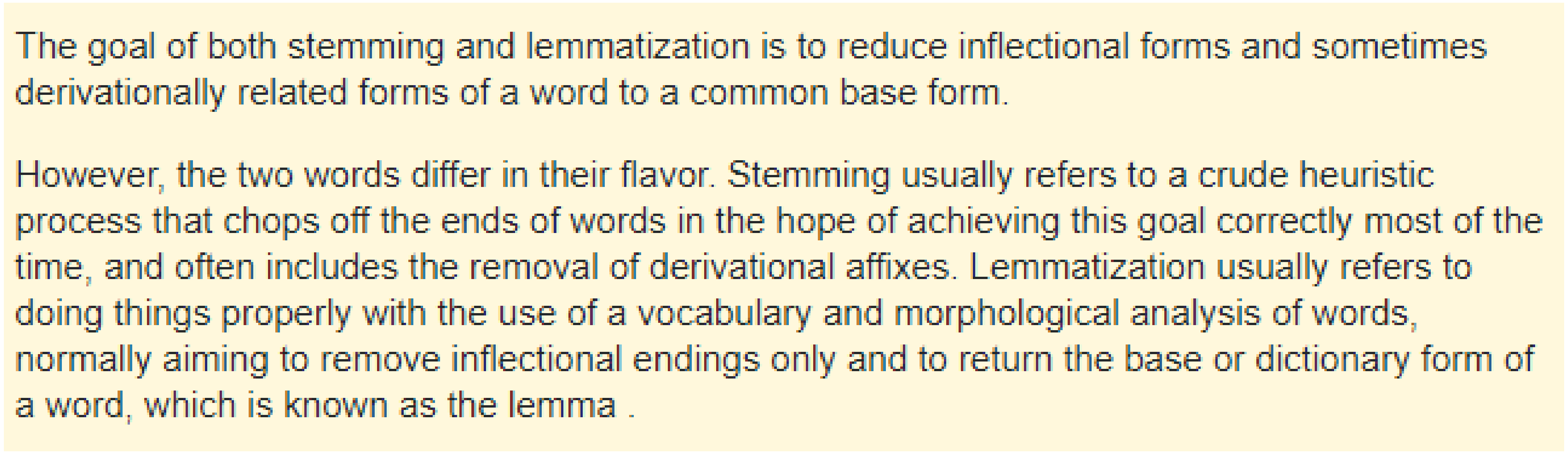
- Stop words are language & domain dependent

- How to determine a list of stopwords?
  - For English? – may use existing lists of stopwords
    - E.g. SMART’s common word list
    - WordNet stopword list
    - http://www.ranks.nl/resources/stopwords.html
    - http://ir.dcs.gla.ac.uk/resources/linguistic_utils/stop_words
  - For Spanish? Bulgarian? Hindi?

- Good News: Stopwords will account for a large fraction of text so eliminating them greatly reduces storage costs.

- Bad News: For most words, gathering sufficient data for meaningful statistical analysis is difficult since they are extremely rare.

Stemming and Lemmatization



The goal of both stemming and lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form.

However, the two words differ in their flavor. Stemming usually refers to a crude heuristic process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes. Lemmatization usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma .

Lemmatization

- Reduce inflectional/variant forms to base form
- Direct impact on VOCABULARY size
  - *am, are, is* $\rightarrow$ *be*
  - *car, cars, car's, cars'* $\rightarrow$ *car*

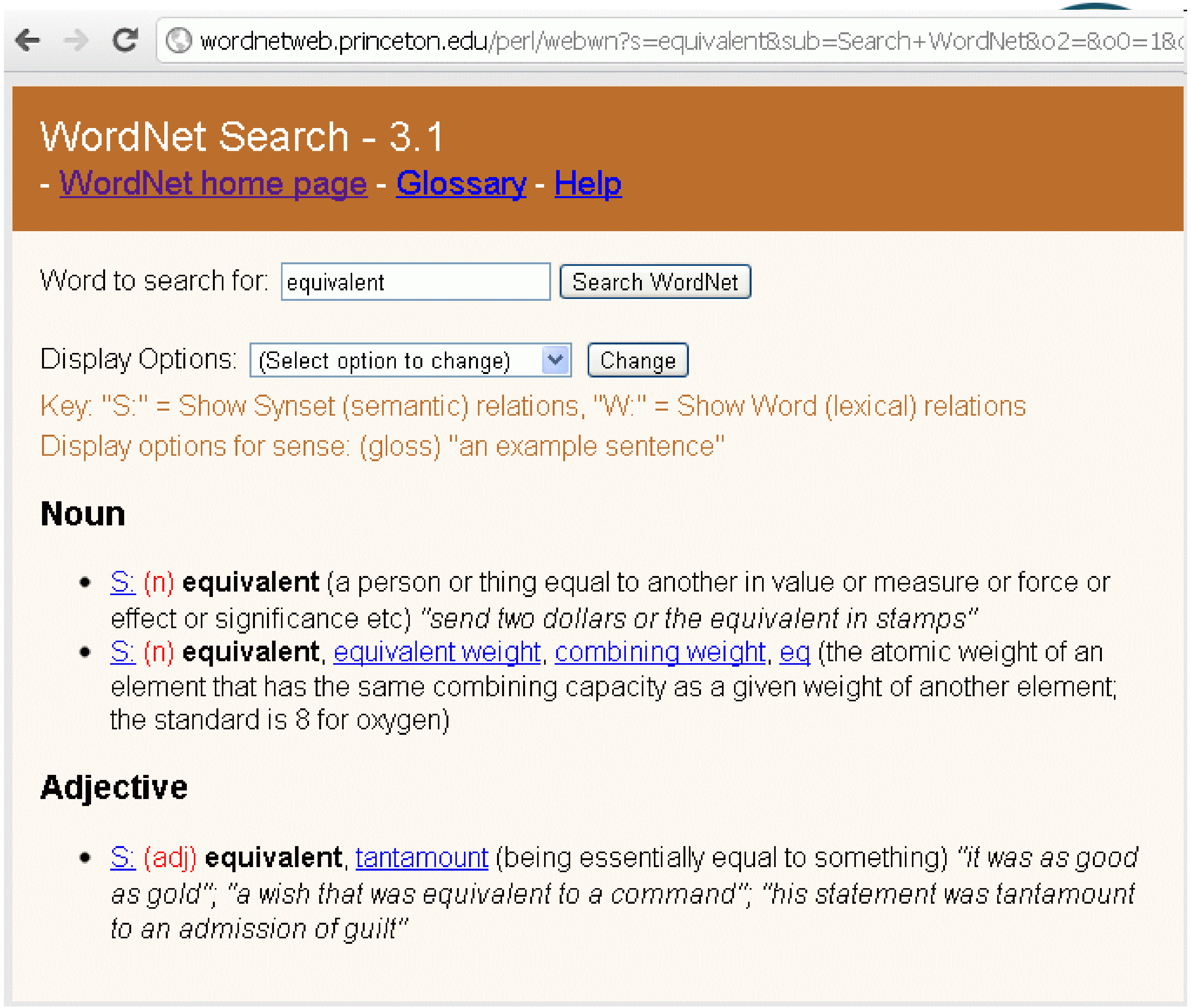
- *the boy's cars are different colors* $\rightarrow$ *the boy car be different color*

- How to do this?
  - Need a list of grammatical rules + a list of irregular words
  - Children $\rightarrow$ child, spoken $\rightarrow$ speak …
  - Practical implementation: use WordNet's morphstr function

WordNet

WordNet® is a large, <u>free</u> lexical database of English.

Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept.



wordnetweb.princeton.edu/perl/webwn?s=equivalent&sub=Search+WordNet&o2=&o0=1&

WordNet Search - 3.1

- WordNet home page - Glossary - Help

Word to search for: equivalent [Search WordNet]

Display Options: (Select option to change) [Change]

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (gloss) "an example sentence"

Noun

- S: (n) **equivalent** (a person or thing equal to another in value or measure or force or effect or significance etc) *"send two dollars or the equivalent in stamps"*
- S: (n) **equivalent**, equivalent weight, combining weight, eq (the atomic weight of an element that has the same combining capacity as a given weight of another element; the standard is 8 for oxygen)

Adjective

- S: (adj) **equivalent**, tantamount (being essentially equal to something) *"it was as good as gold"; "a wish that was equivalent to a command"; "his statement was tantamount to an admission of guilt"*

Stemming

- Reduce tokens to “root” form of words to recognize morphological variation.
  - “computer”, “computational”, “computation” all reduced to same token “comput”
- Correct morphological analysis is language specific and can be complex.
- Stemming “blindly” strips off known affixes (prefixes and suffixes) in an iterative fashion.

Porter Stemmer

http://tartarus.org/~martin/PorterStemmer/

Example (dtSearch): Stemming rules consist of a series of lines like this:

3+ies -> Y
4+ing ->

The first rule would convert any word with three or more letters followed by ies to the same initial letters followed by y. ***Applies* would turn into *apply***.

The second rule would remove the *ing* from any word with four or more letters followed by *ing*. ***Fishing* would turn into *fish*, but *sing* would not change**.

In general, a rule consists of: a minimum number of letters (not including the suffix), a + sign, a suffix to be removed, an arrow (->) and the replacement for the suffix, if any.

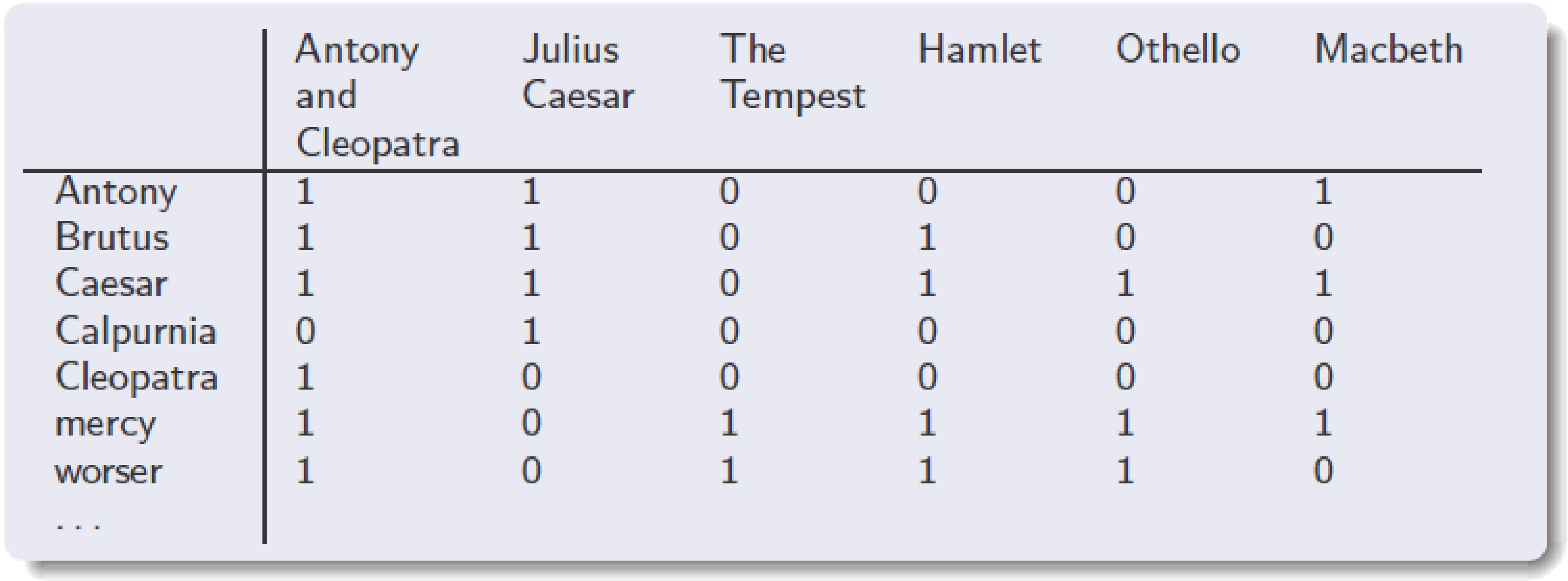
Porter Stemmer

- Simple procedure for removing known affixes in English without using a dictionary.
- Can produce unusual stems that are not English words:
  - “computer”, “computational”, “computation” all reduced to same token “comput”
- May conflate (reduce to the same token) words that are actually distinct.
- Does not recognize all morphological derivations.

- What can we do now with the data [terms] extracted from each of the documents ?

Term-Document Incidence Matrix

Main idea: record for each document whether it contains each word out of all the different words Shakespeare used (about 32K).



	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0
. . .						

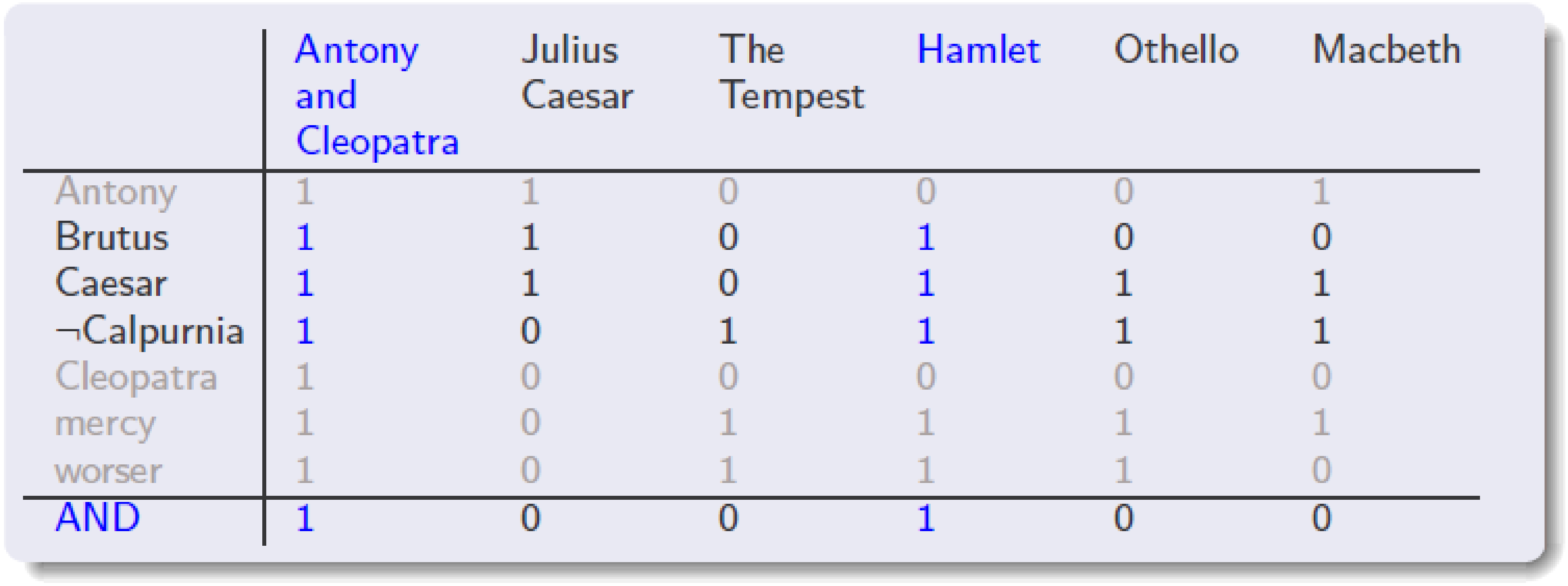
Matrix element (t, d) is 1 if the play in column d contains the word in row t, 0 otherwise.

Query "Brutus AND Caesar AND NOT Calpunia"

We compute the results for our query as the bitwise AND between vectors for Brutus, Caesar and complement (Calpurnia):

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0
. . .						

- Which documents match the query ?



	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
¬Calpurnia	1	0	1	1	1	1
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0
AND	1	0	0	1	0	0

Bitwise AND returns two documents, "Antony and Cleopatra" and "Hamlet".

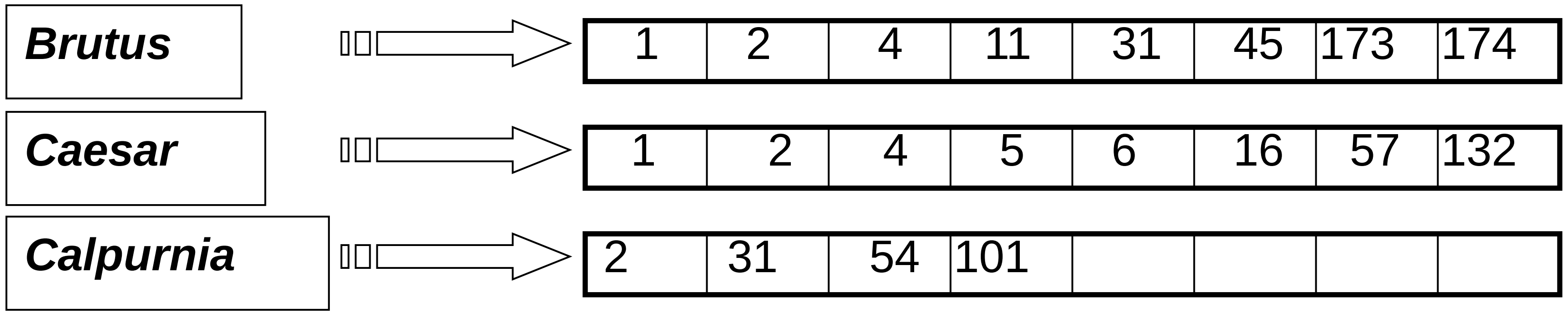
STEP 2: INVERTED INDEX

Inverted Index: Motivation

- N = 1 million documents, each with about 1000 words.
- M = 500K *distinct* terms among these.
- 500K x 1M matrix has half-a-trillion 0's and 1's.
- But it has no more than one billion 1's
  - matrix is extremely sparse.
- A better representation?
  - Only record the 1 positions.

Inverted index

- For each term *t*, store a list of all documents that contain *t*.
  - Identify each by a **docID**, a document serial number
  - Optionally store the number & position of occurrence of t in d.



Inverted Index – in practice

- We need:
  - One entry for each word/term in the vocabulary
  - For each such entry:
    - Keep a list of all the documents where it appears together with the corresponding frequency → TF
  - For each such entry, keep the total number of occurrences in all documents:
    - DF

Indexer steps: Token sequence

- Sequence of (Modified token, Document ID) pairs.

Doc 1

I did enact Julius Caesar I was killed i' the Capitol; Brutus killed me.

Doc 2

So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious

Term	docID
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

Indexer steps: Sort

- Sort by terms
  - And then docID

Core indexing step

Term	docID
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

$\rightarrow$

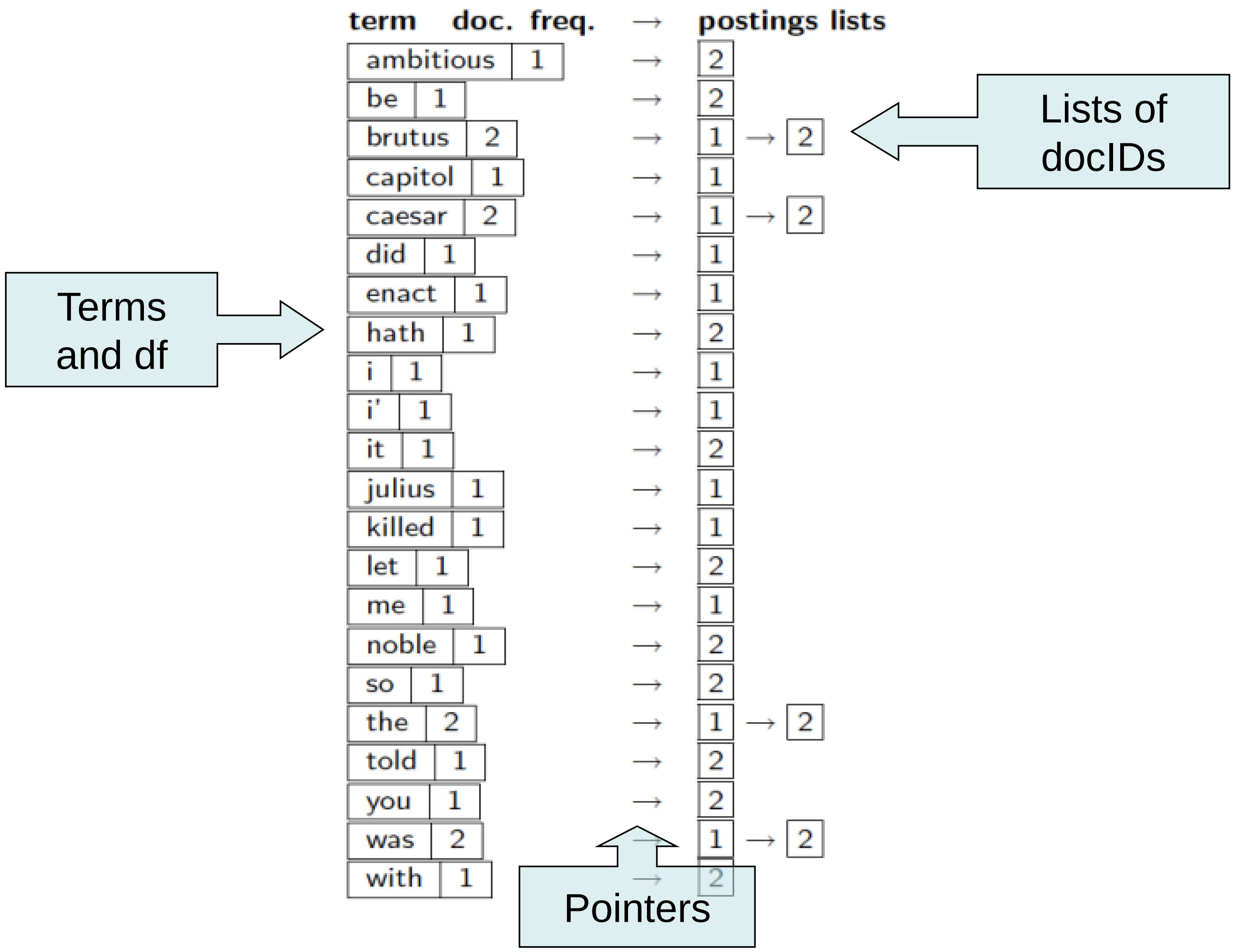
Term	docID
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

Indexer steps: Dictionary & Postings

- Multiple term entries in a single document are merged.
- Split into Dictionary and Postings
- Document frequency information is added.

Term	docID
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

term	doc. freq.	→	postings lists
ambitious	1	→	2
be	1	→	2
brutus	2	→	1 → 2
capitol	1	→	1
caesar	2	→	1 → 2
did	1	→	1
enact	1	→	1
hath	1	→	2
i	1	→	1
i'	1	→	1
it	1	→	2
julius	1	→	1
killed	1	→	1
let	1	→	2
me	1	→	1
noble	1	→	2
so	1	→	2
the	2	→	1 → 2
told	1	→	2
you	1	→	2
was	2	→	1 → 2
with	1	→	2



term	doc. freq.	→	postings lists
ambitious	1	→	2
be	1	→	2
brutus	2	→	1 → 2
capitol	1	→	1
caesar	2	→	1 → 2
did	1	→	1
enact	1	→	1
hath	1	→	2
i	1	→	1
i'	1	→	1
it	1	→	2
julius	1	→	1
killed	1	→	1
let	1	→	2
me	1	→	1
noble	1	→	2
so	1	→	2
the	2	→	1 → 2
told	1	→	2
you	1	→	2
was	2	→	1 → 2
with	1	→	2

Inverted Index: Another Example

Doc 1	Doc 2	Doc 3	Doc 4
one fish, two fish	red fish, blue fish	cat in the hat	green eggs and ham

	1	2	3	4
blue		1		
cat			1	
egg				1
fish	1	1		
green				1
ham				1
hat			1	
one	1			
red		1		
two	1			

$$\begin{tikzcd}
\text{blue} \arrow[r] & \boxed{2} \\
\text{cat} \arrow[r] & \boxed{3} \\
\text{egg} \arrow[r] & \boxed{4} \\
\text{fish} \arrow[r] & \boxed{1} & \boxed{2} \\
\text{green} \arrow[r] & \boxed{4} \\
\text{ham} \arrow[r] & \boxed{4} \\
\text{hat} \arrow[r] & \boxed{3} \\
\text{one} \arrow[r] & \boxed{1} \\
\text{red} \arrow[r] & \boxed{2} \\
\text{two} \arrow[r] & \boxed{1}
\end{tikzcd}$$

Inverted Index: Ranked Retrieval

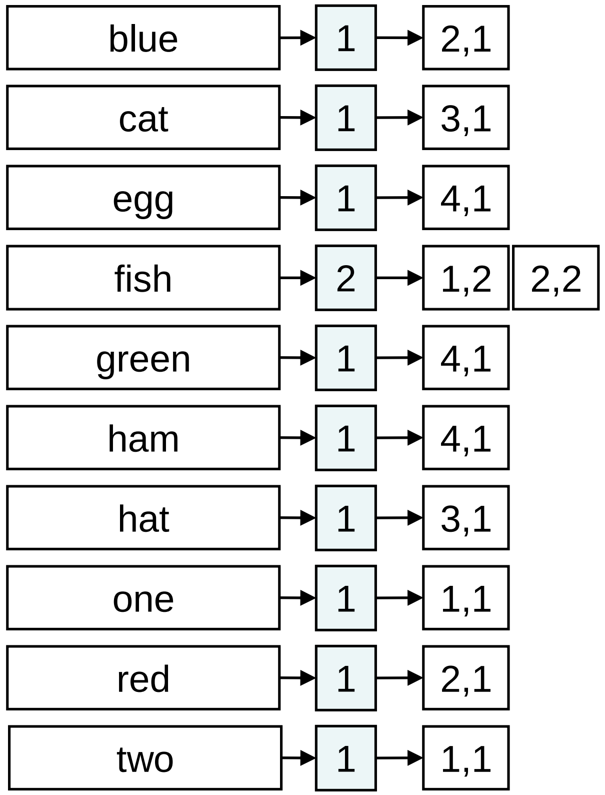
Doc 1
one fish, two fish

Doc 2
red fish, blue fish

Doc 3
cat in the hat

Doc 4
green eggs and ham

	tf 1	2	3	4	*df*
blue		1			1
cat			1		1
egg				1	1
fish	2	2			2
green				1	1
ham				1	1
hat			1		1
one	1				1
red		1			1
two	1				1



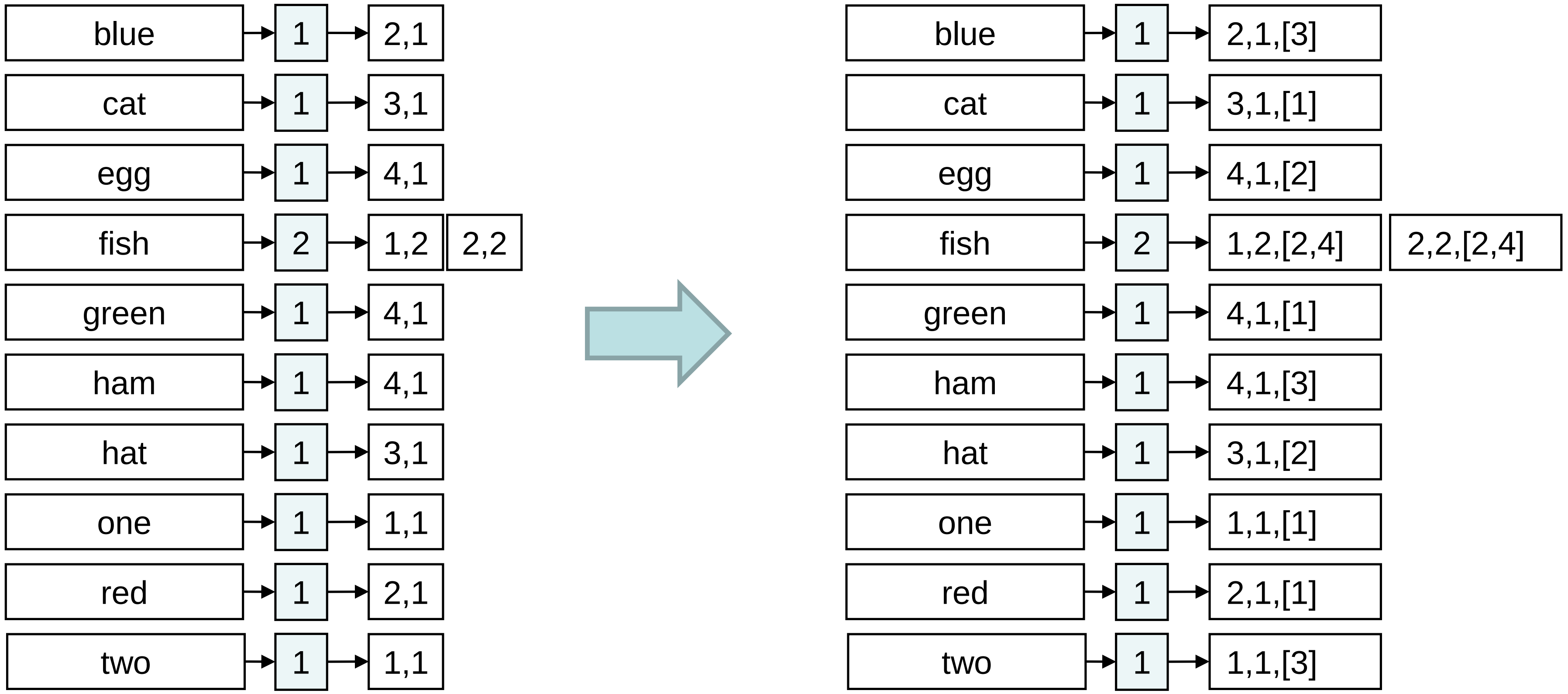
Inverted Index: Positional Information

Doc 1
one fish, two fish

Doc 2
red fish, blue fish

Doc 3
cat in the hat

Doc 4
green eggs and ham



Review
[Documents on web ➔ Bag of Words]

1. Crawling ->

   Raw document storage ->

   Corpus [a corpus (plural corpora) or text corpus is a large and structured set of texts]

2. Corpus ->

   Preprocessing ->

   Bag of Words/Tokens [can get counts]

- Preprocessing: tokenization, stop word removal, stemming, lemmatization, … all Domain & Problem dependent

Two approaches

- Bag of Words model
  - ❑Order-less document representation as frequencies of words from a dictionary/vocabulary.
  - ❑Positional, proximity details not retained.
- BoW and more [probabilistic]

Term-Document Matrix [TDM]

- We can now construct a structured form
- Terms in rows
- Documents in columns
- Cells show the term’s frequency [count] for a specific term in a specific document

- Sometimes constructed as “Documents-Term Matrix” [Docs in rows and Terms in columns]

Term – Document Matrix

- What is the word at the end of this -----?

Human Word Prediction

- One can predict future words in an utterance.
- How?
  - Domain knowledge: red blood vs. red hat
  - Syntactic knowledge: the…<adj|noun>
  - Lexical knowledge: baked <potato vs. chicken>
- A useful part of the knowledge needed to allow Word Prediction can be captured using simple statistical techniques
- In particular, we'll be interested in the notion of the probability of a sequence (of letters, words,…)

Applications of Word Prediction

- Why do we want to predict a word, given some preceding words?
  - Rank the likelihood of sequences containing various alternative hypotheses
  - Assess the likelihood/goodness of a sentence
- Spelling checkers
  - They are leaving in about fifteen **minuets** to go to her house
  - The study was conducted mainly **be** John Black.
  - I need to **notified** the bank of this problem.
- Mobile phone texting
  - He is trying to **fine** out.
  - Hopefully, all **with** continue smoothly in my absence.
- Speech recognition
  - Theatre owners say **popcorn/unicorn** sales have doubled...
- Handwriting recognition
- Disabled users
- Machine Translation

Language Model

- Compute the probability of a sentence or sequence of words:

  $P(W) = P(w_1,w_2,w_3,w_4,w_5 \ldots w_n)$

- Compute probability of an upcoming word:

  $P(w_5 | w_1,w_2,w_3,w_4)$

- A model that computes either of these:

  $P(W)$ or $P(w_n | w_1,w_2 \ldots w_{n-1})$

is called a **Language Model**

How to do Word Prediction?

- Use the previous N-1 words in a sequence to predict the next word
- Language Model (LM)
  - unigrams, bigrams, trigrams,…
- How do we train these models?
  - Very large corpora
  - Corpora are online collections of text and speech
    - Wall Street Journal
    - Newswire

Bigram Model

- Approximate P(w_n|w_1,w_2…w_{n-1}) by $P(w_n|w_{n-1})$
  - E.g., P(unicorn|the mythical) by P(unicorn|mythical)

- Markov *assumption*: the probability of a word depends only on the probability of a limited history

- ***Generalization: the probability of a word depends only on the probability of the n previous words***
  - trigrams, 4-grams, 5-grams,…
  - the higher n is, the more data needed to train

Sample Bigram Probabilities

Eat on	.16	Eat Thai	.03
Eat some	.06	Eat breakfast	.03
Eat lunch	.06	Eat in	.02
Eat dinner	.05	Eat Chinese	.02
Eat at	.04	Eat Mexican	.02
Eat a	.04	Eat tomorrow	.01
Eat Indian	.04	Eat dessert	.007
Eat today	.03	Eat British	.001

<start> I	.25	Want some	.04
<start> I'd	.06	Want Thai	.01
<start> Tell	.04	To eat	.26
<start> I'm	.02	To have	.14
I want	.32	To spend	.09
I would	.29	To be	.02
I don't	.08	British food	.60
I have	.04	British restaurant	.15
Want to	.65	British cuisine	.01
Want a	.05	British lunch	.01

Text Generation with Bigram Model

- P(I want to eat British food) = P(I|<start>) P(want|I) P(to|want) P(eat|to) P(British|eat) P(food|British)

= .25*.32*.65*.26*.001*.60 = .000080

  – How would we calculate I want to eat Chinese food ?

- Probabilities roughly capture "syntactic" facts and "world knowledge"
  – British food is not too popular
    - "eat British" has lower prob than "eat Chinese"
    - "British food" has lower prob than "Chinese food"

- Back to BoW model

For calculating the tf-idf weight of a term in a particular document, it is necessary to know two things:

[1] how often does it occur in the document (term frequency = tf), and

[2] in how many documents of the collection does that term appear (document frequency = df).

N = total number of documents in corpus
idf = Inverted Document Frequency

$$idf = \log\left(\frac{N}{df}\right)$$

Example: (Corpus is of 1000 Documents)

Term	tf(doc1)	tf(doc2)	tf(doc3)	df(t)	idf(t)
method	4,250	3,400	5,100	850	0.07
the	50,000	43,000	55,000	1,000	0.00
water	7,600	4,000	2,000	400	0.40
bioreactor	600	0	25	25	1.6

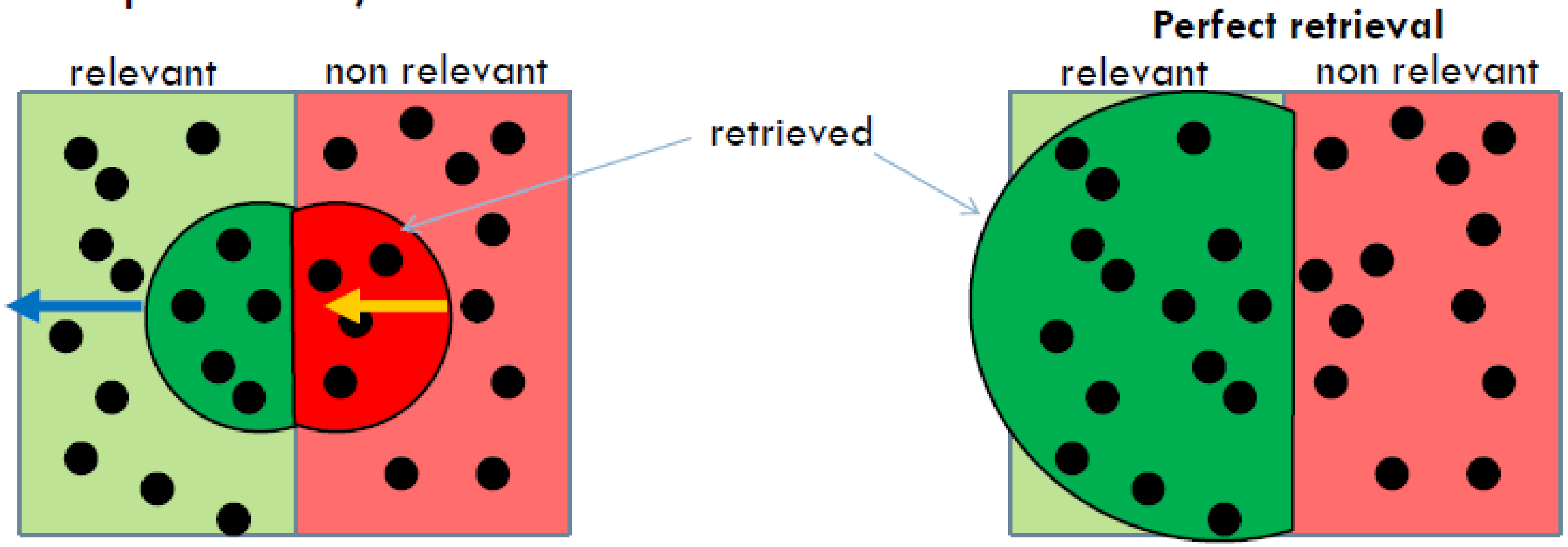
Term	tf-idf(doc1)	tf-idf(doc2)	tf-idf(doc3)
method	299.97	239.98	359,96
the	0.00	0.00	0.00
water	3,024.34	1,591.76	795.88
bioreactor	961.24	0.00	40.05

- See how the inverse document frequency (idf) influences the tf-idf value.

- Even though the term "method" occurs nearly as often as "water" in document doc2, the tf-idf value of "water" is 6.6 times higher

How do we measure quality of retrieval?

- **Recall:** portion of the relevant documents that the system retrieved (blue arrow points in the direction of higher recall)
- **Precision:** portion of retrieved documents that are relevant (yellow arrow points in the direction of higher precision)



A different way to look at :

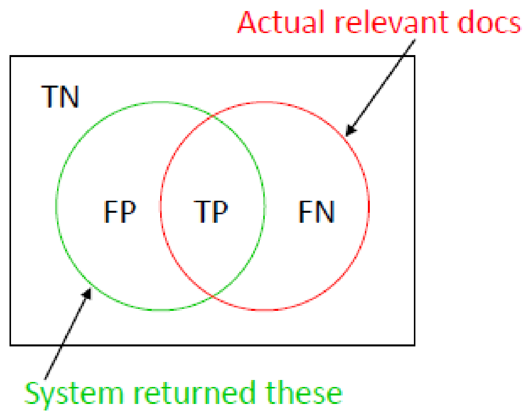
Recall = 1 whole truth

Precision = 1 nothing but the truth

In the language of TP, FP….

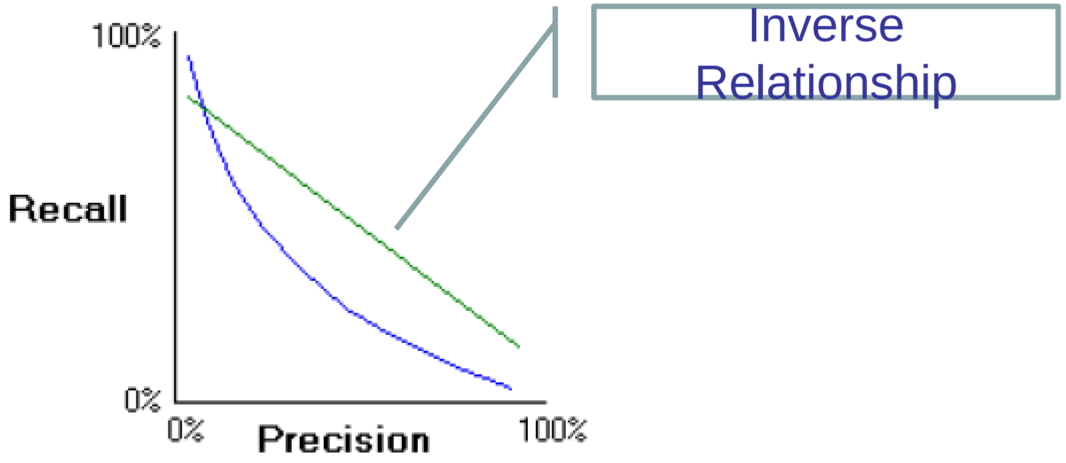
- **Precision**: fraction of retrieved docs that are relevant
- **Recall**: fraction of relevant docs that are retrieved

	Relevant	Nonrelevant
Retrieved	tp	fp
Not Retrieved	fn	tn



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

Precision vs Recall : Trade-off



In the graph above, the two lines may represent the performance of different search systems. While the exact slope of the curve may vary between systems, the general inverse relationship between recall and precision remains.

Quiz:

- A corpus contains **80** records on a particular topic of interest to user
- **60** records were retrieved as result of user search on the topic
- Out of 60 records that were retrieved, **45 records** were relevant.

recall = ?

precision = ?

Recall = 45/80 * 100% = **56%**

Precision = 45/60 * 100% = **75%**

F-score

- F measure: $F = \frac{2PR}{P+R}$

- Where did this come from ?

Harmonic Mean of P and R

$$F_1 = 2 \cdot \frac{1}{\frac{1}{\mathrm{recall}} + \frac{1}{\mathrm{precision}}} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}.$$

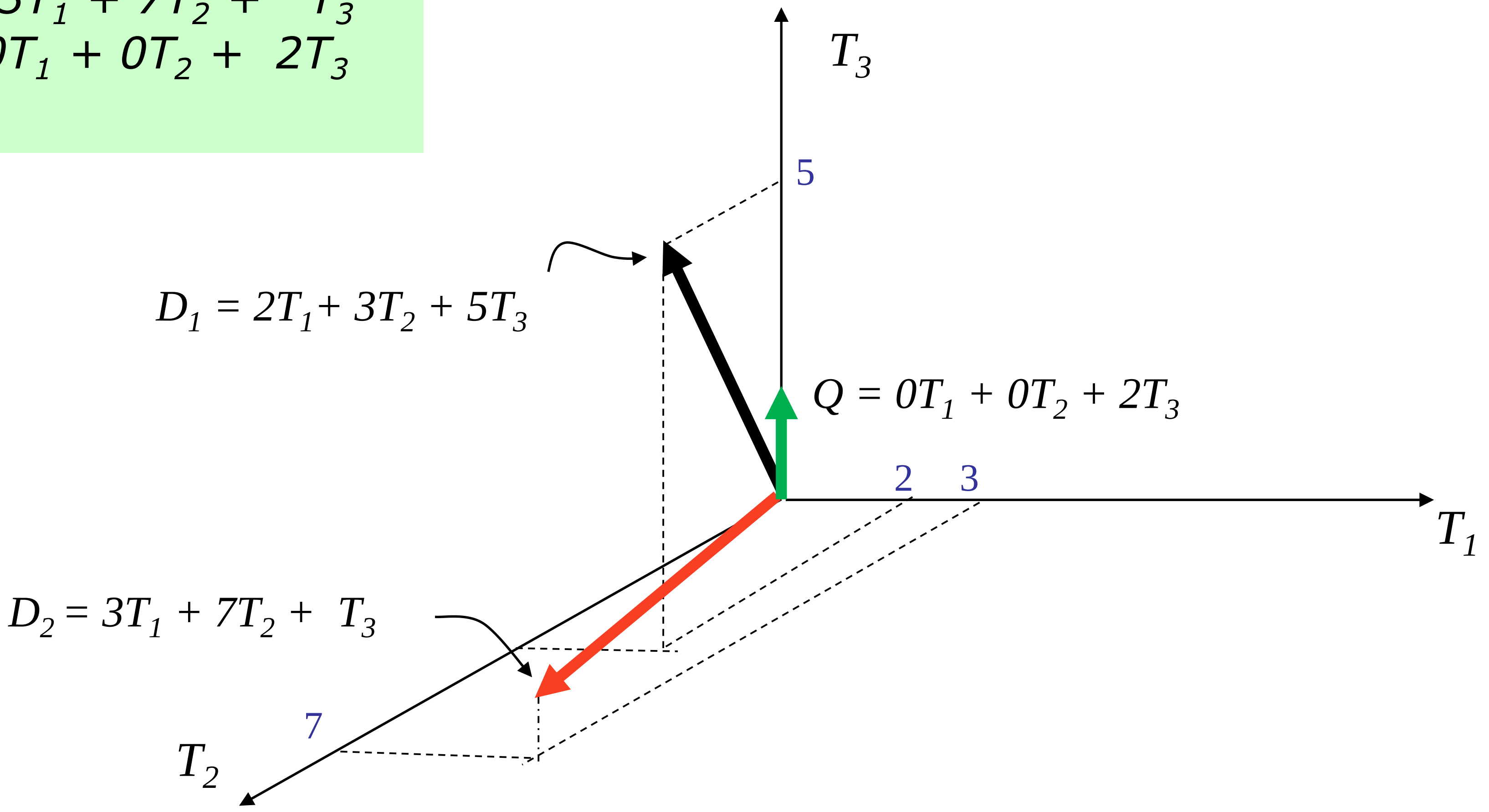
Vector Space Representation

Example:

$D_1 = 2T_1 + 3T_2 + 5T_3$

$D_2 = 3T_1 + 7T_2 + T_3$

$Q = 0T_1 + 0T_2 + 2T_3$



What do we do with all this ?

- We created BoW model and are able to visualize documents in the vector space
- What questions we want to answer?
  - Is D_1 or D_2 more similar to Q?
  - How to measure the degree of similarity?
  - Distance? Angle? Projection?

Similarity Measure

- We now have vectors for all documents in the collection, a vector for the query, how to compute similarity?

- Using a similarity measure between the query and each document:

  – It is possible to rank the retrieved documents in the order of presumed relevance (query-dependent ranking).

  – It is possible to enforce a certain threshold so that the size of the retrieved set can be controlled.

Desiderata for proximity

- If d_1 is near d_2, then d_2 is near d_1.
- If d_1 near d_2, and d_2 near d_3, then d_1 is not far from d_3.
- No document is closer to d than d itself.

First cut: Euclidean distance

- Distance between vectors d_1 and d_2 is the length of the vector $|d_1 - d_2|$.
  - Euclidean distance

- Why is this not a great idea?

- Length normalization
  - Document length will play a role, not just topic

- Second Cut: Same problem with Manhattan distance.

- We could implicitly normalize by looking at *angles* instead

Third cut: Inner Product

- Similarity between vectors for the document ***d****$_i$* and query ***q*** can be computed as the vector inner product:

$$\text{sim}(\boldsymbol{d}_j, \boldsymbol{q}) = \boldsymbol{d}_j \bullet \boldsymbol{q} = \sum_{i=1}^{t} w_{ij} \cdot w_{iq}$$

where w_{ij} is the weight of term *i* in document *j* and w_{iq} is the weight of term *i* in the query

- For binary vectors, the inner product is the number of matched query terms in the document (size of intersection)

- For weighted term vectors, it is the sum of the products of the weights of the matched terms.

Properties of Inner Product

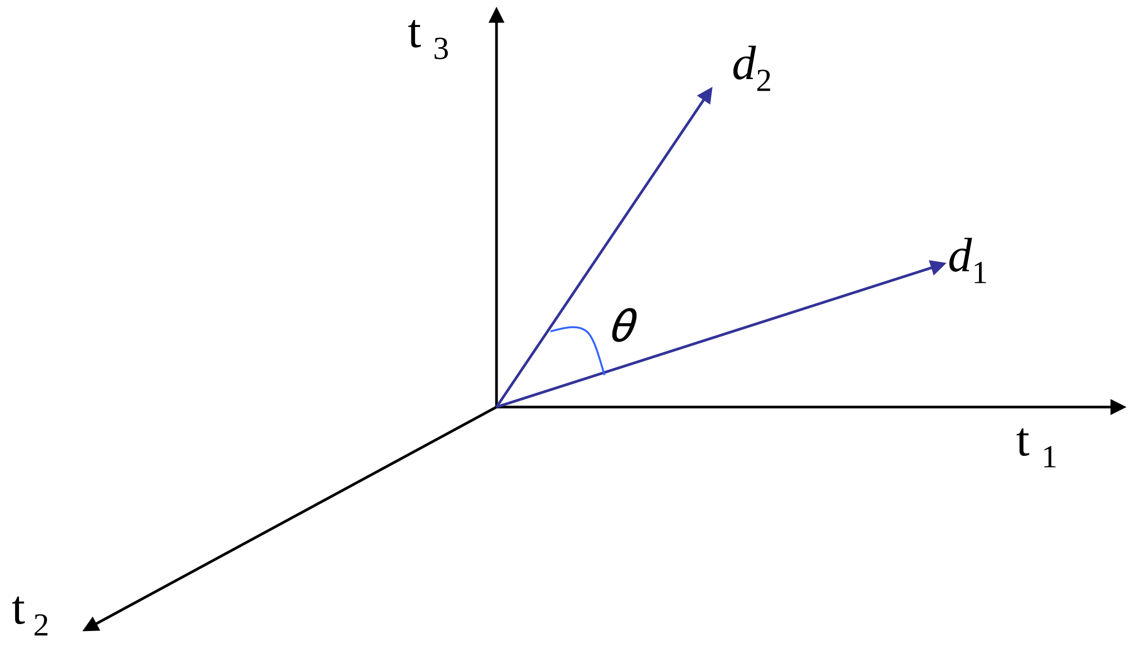
- Favors long documents with a large number of unique terms.
  - Again, the issue of length normalization

- Measures how many terms matched but not how many terms are *not* matched.

Cosine similarity

- Distance between vectors d_1 and d_2 *captured* by the cosine of the angle *x* between them.

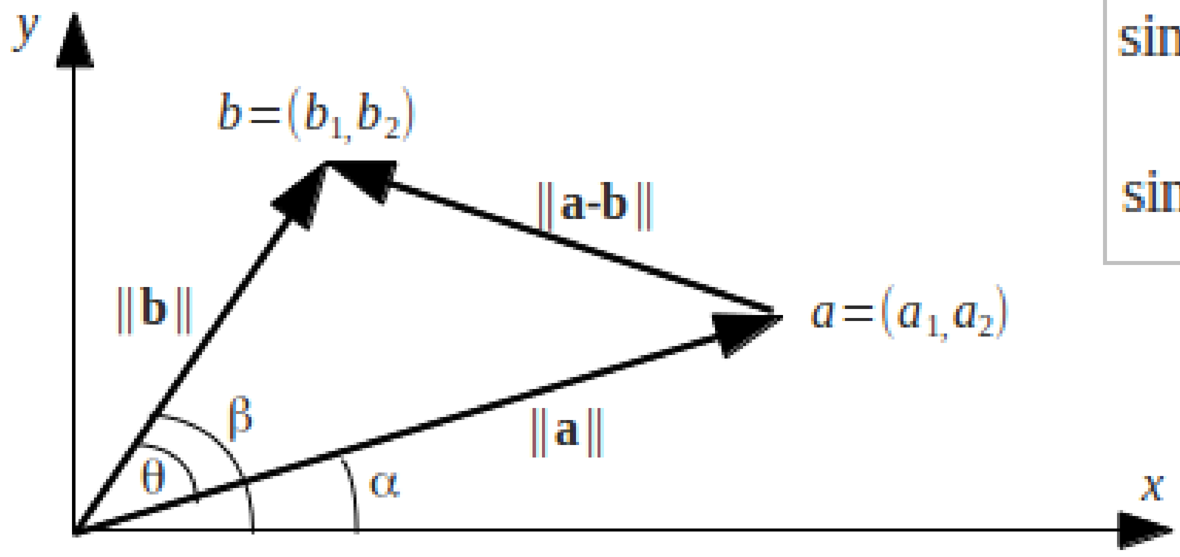
- Note – this is actually *similarity*, not distance



Cosine similarity

$$sim(d_j, d_k) = \frac{\vec{d}_j \cdot \vec{d}_k}{\left|\vec{d}_j\right|\left|\vec{d}_k\right|} = \frac{\sum_{i=1}^{n} w_{i,j} w_{i,k}}{\sqrt{\sum_{i=1}^{n} w_{i,j}^2}\sqrt{\sum_{i=1}^{n} w_{i,k}^2}}$$

- Cosine of angle between two vectors
- The denominator involves the lengths of the vectors
- The cosine measure is also known as the *normalized inner product*



The cosine of the angle $\theta=\beta-\alpha$

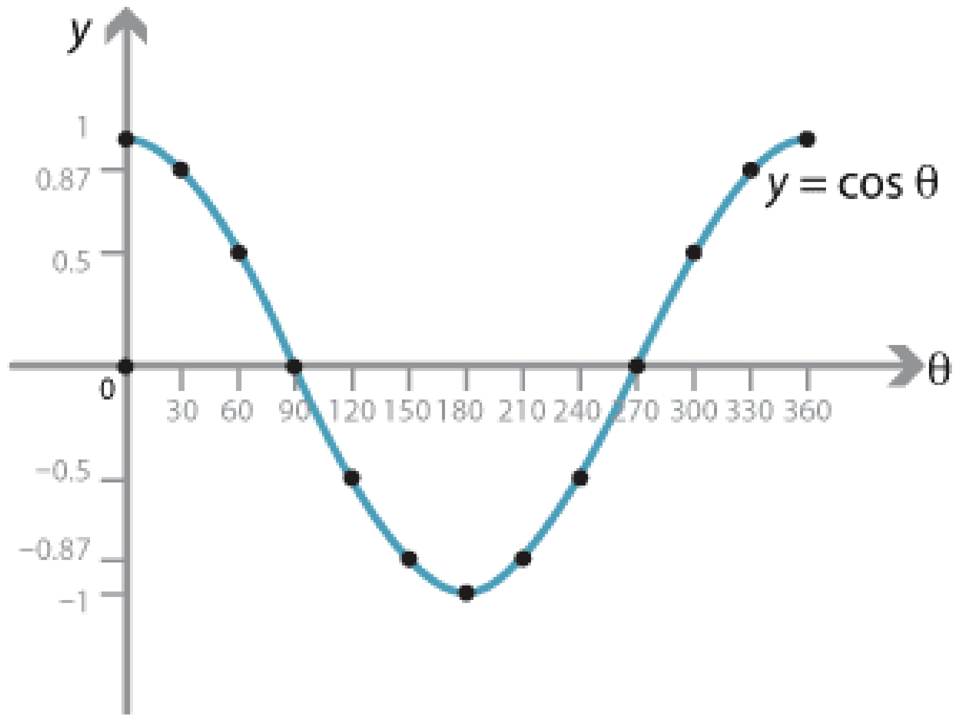
$$\sin\alpha=\frac{a_2}{\|\mathbf{a}\|} \qquad \cos\alpha=\frac{a_1}{\|\mathbf{a}\|}$$
$$\sin\beta=\frac{b_2}{\|\mathbf{b}\|} \qquad \cos\beta=\frac{b_1}{\|\mathbf{b}\|}$$

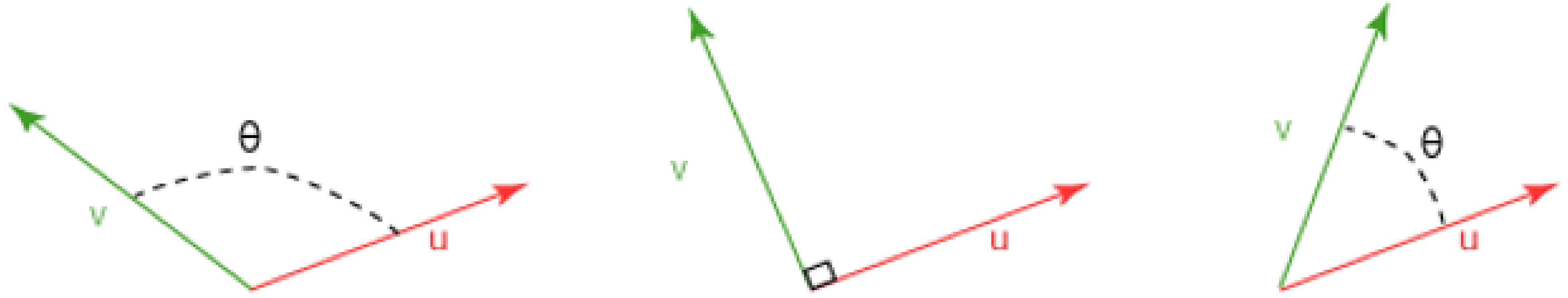
$$\cos(x-y)=\cos x\cos y+\sin x\sin y$$

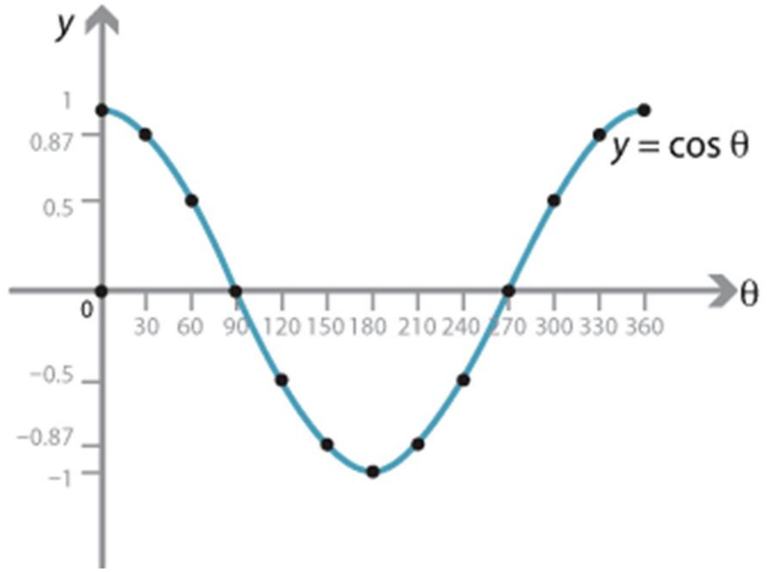
More importantly, Why cos θ for similarity measure ?

Hint:

What does "cos θ" function/graph look like?


y
1
0.87
0.5
0
-0.5
-0.87
-1
30 60 90 120 150 180 210 240 270 300 330 360
θ
y = cos θ


θ
v
u
v
u
v
θ
u


y
1
0.87
0.5
0
−0.5
−0.87
−1
30 60 90 120 150 180 210 240 270 300 330 360
θ
y = cos θ

Ranking in the vector space model

- Represent both query and document as vectors in the |V|-dimensional space
- Use cosine similarity as the similarity measure
  - Incorporates length normalization automatically (longer vs shorter documents)

$$\cos(\vec{q},\vec{d}) = \frac{\vec{q}\bullet\vec{d}}{|\vec{q}||\vec{d}|} = \frac{\vec{q}}{|\vec{q}|}\bullet\frac{\vec{d}}{|\vec{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2}\sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

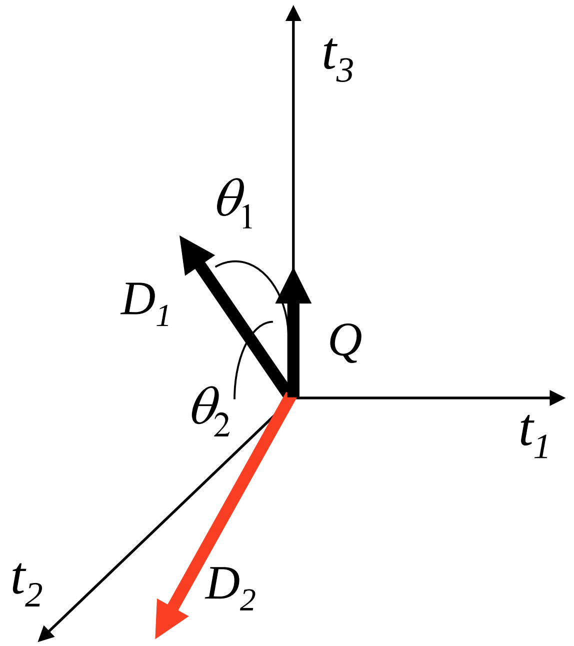
- q_i is the tf-idf weight of term i in the query
- d_i is the tf-idf weight of term i in the document

Cosine Similarity vs. Inner Product

$D_1 = 2T_1 + 3T_2 + 5T_3$ CosSim(D_1 , Q) = 10 / √(4+9+25)(0+0+4) = 0.81

$D_2 = 3T_1 + 7T_2 + 1T_3$ CosSim(D_2 , Q) = 2 / √(9+49+1)(0+0+4) = 0.13

$Q = 0T_1 + 0T_2 + 2T_3$

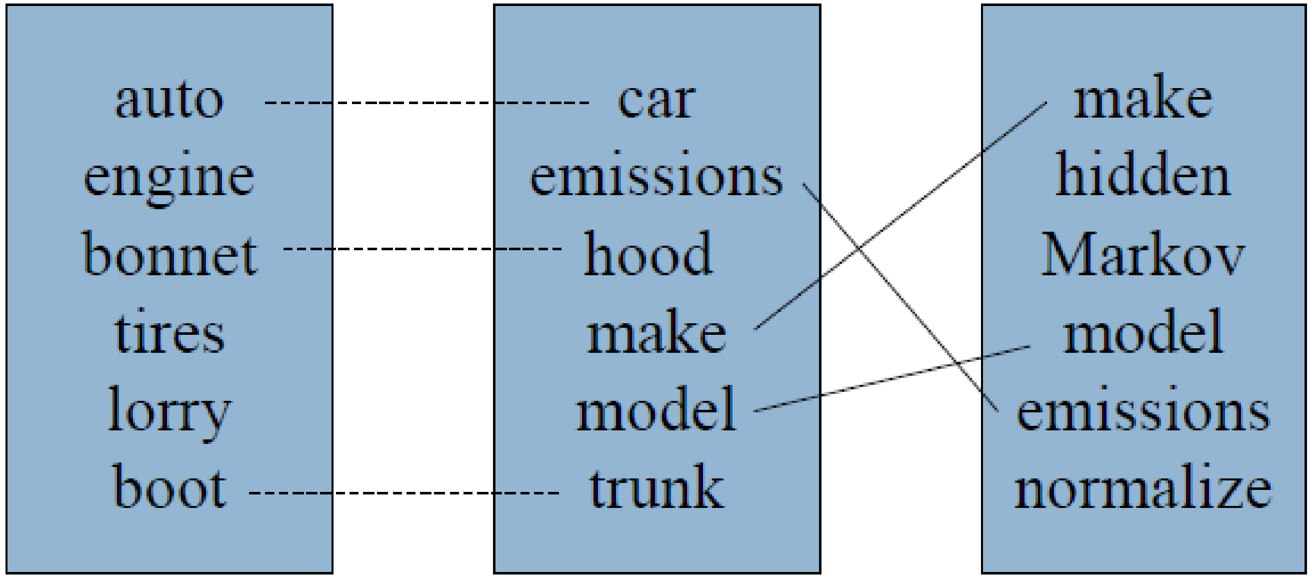


D_1 is 6 times better than D_2 using cosine similarity but only 5 times better using inner product.

Comments on Vector Space Models

- Simple, mathematics based approach.
- Provides partial matching and ranked results.
- Tends to work quite well in practice despite obvious weaknesses [we will see shortly, can you guess?].
- Allows efficient implementation for large document collections.

- Synonemy
- Polysemi



Synonymy

Will have small cosine

but are related

Polysemy

Will have large cosine

but not truly related

Problems with Vector Space Model

- Missing syntactic information (e.g. phrase structure, word order, proximity information).
- Missing semantic information (e.g. word sense).
- Assumption of term independence (e.g. ignores synonomy).
- Lacks the control of a Boolean model (e.g., *requiring* a term to appear in a document).
  - Given a two-term query “A B”, may prefer a document containing A frequently but not B, over a document that contains both A and B, but both less frequently.

Latent Semantic Analysis

LSI/LSA

- Latent Semantic Indexing/Analysis (LSI/LSA) is an application of Singular Value Decomposition Technique (SVD) on the word [term]-document matrix used in Information Retrieval.

- Latent Semantic Indexing was proposed in 1990 by a team of researchers of which Susan Dumais is the most prominent.

Some SVD background before we get into LSI/LSA

Factorization

- Not much can be said about 1728
- But, a lot can be said about

$$2^6 X 3^3$$

Same is the case with “matrix decomposition”

- Decomposition reveals hidden secrets about original matrix
- Eigen-decomposition applicable only to square matrices
- For non-square matrices, SVD comes to rescue.

- SVD has other applications in addition to text mining

- Image Compression
- Noise Reduction
- Recommendation Engines

Singular Value Decomposition

For an $m \times n$ matrix $\mathbf{A}$ of rank r there exists a factorization (Singular Value Decomposition = **SVD**) as follows:

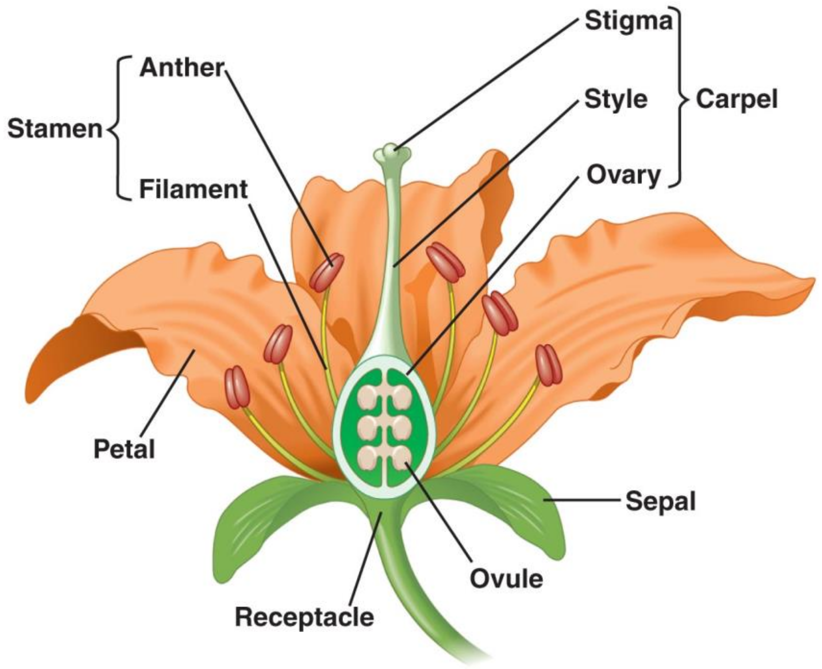
$$\begin{tikzcd} A = & U & \Sigma & V^T \\ & \boxed{m\times n} \arrow[u] & \boxed{n\times n} \arrow[u] & \boxed{V \text{ is } n\times n} \arrow[u] \end{tikzcd}$$

Eigenvalues $\lambda_1 \ldots \lambda_r$ of $\boldsymbol{AA^T}$ are the eigenvalues of $\boldsymbol{A^TA}$.

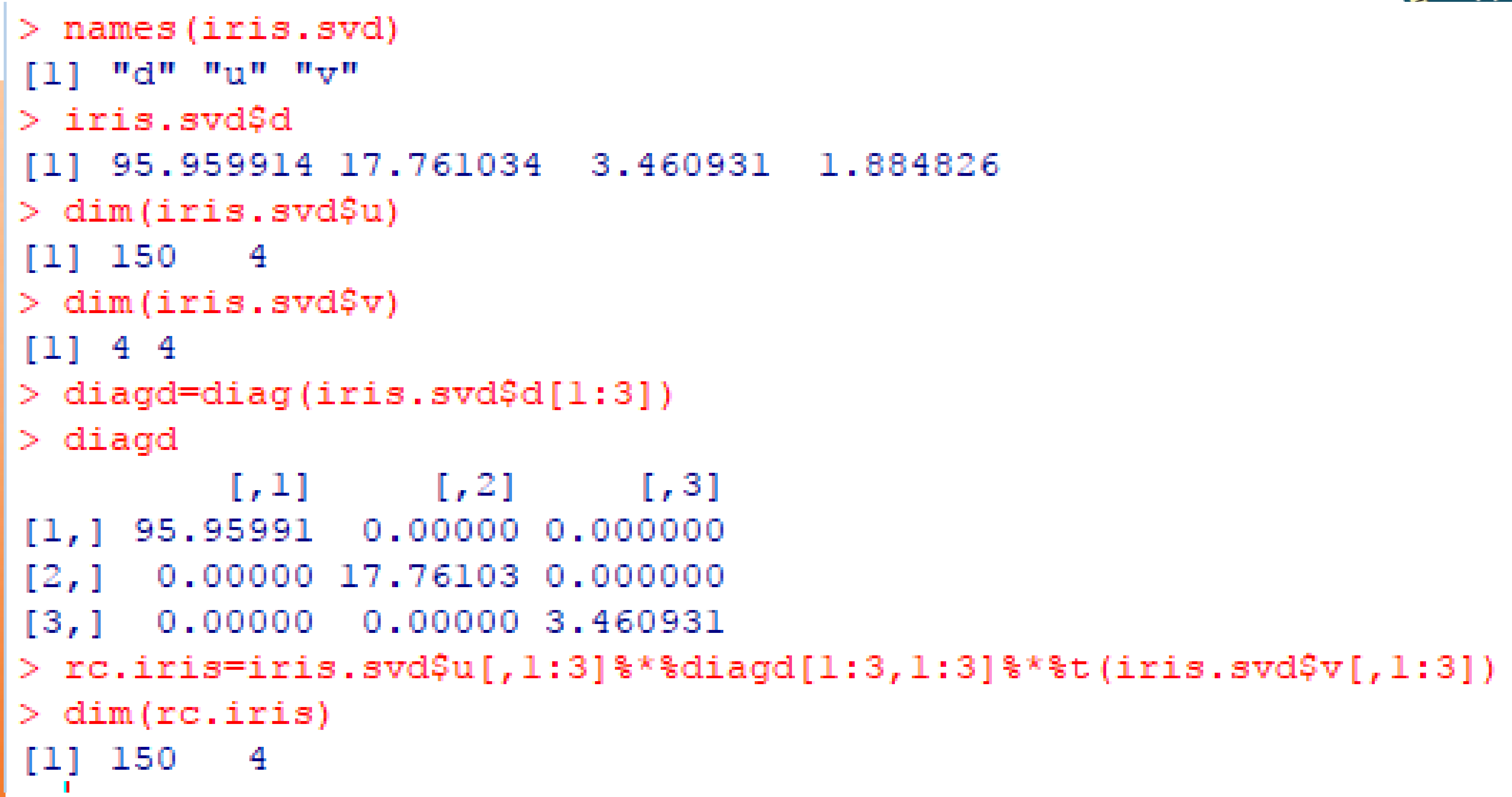
$$\sigma_i = \sqrt{\lambda_i}$$

$$\Sigma = diag(\sigma_1 \ldots \sigma_r) \longleftarrow \boxed{\textit{Singular values.}}$$

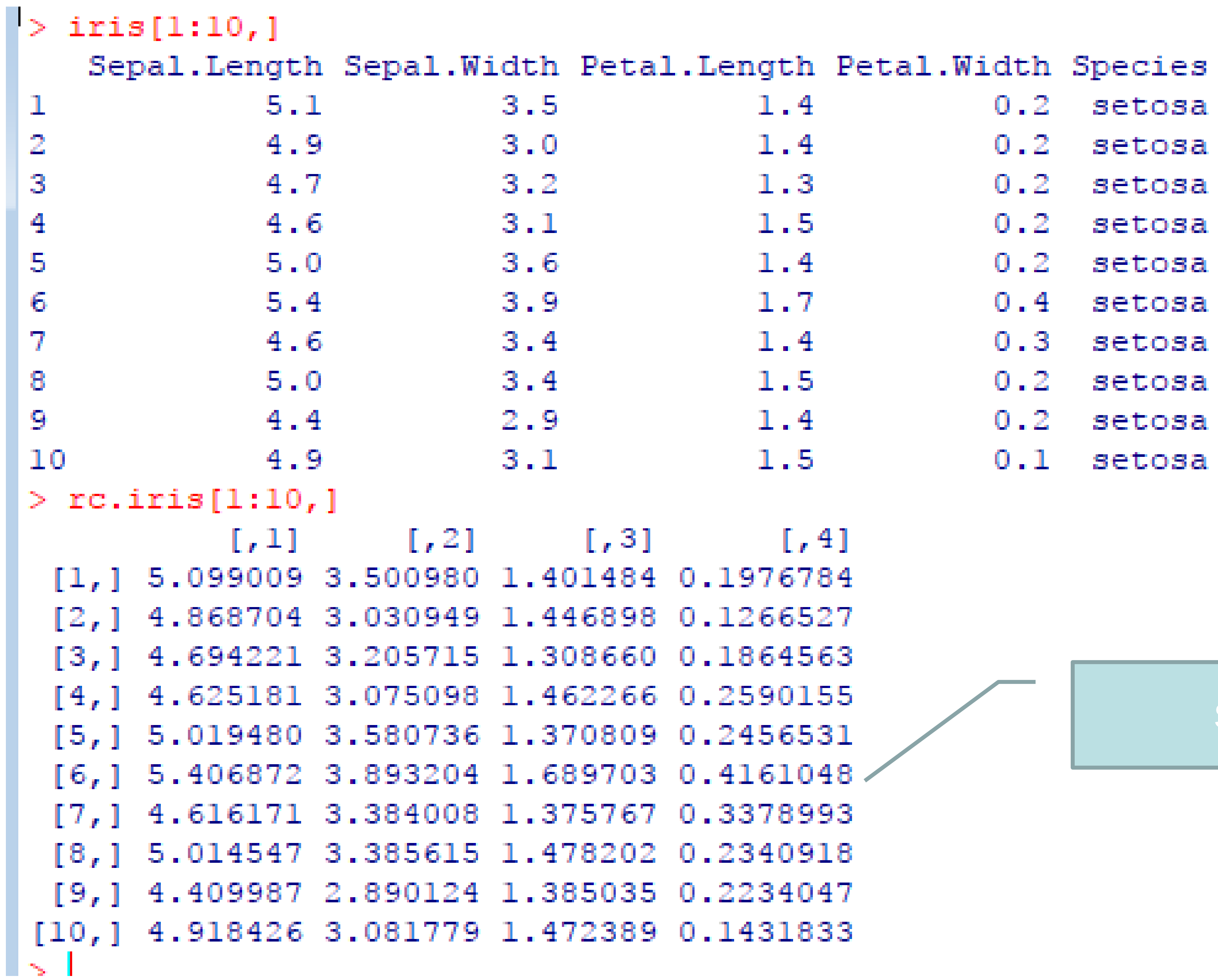
Iris dataset



```
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> iris.svd = svd(iris[,1:4])
```

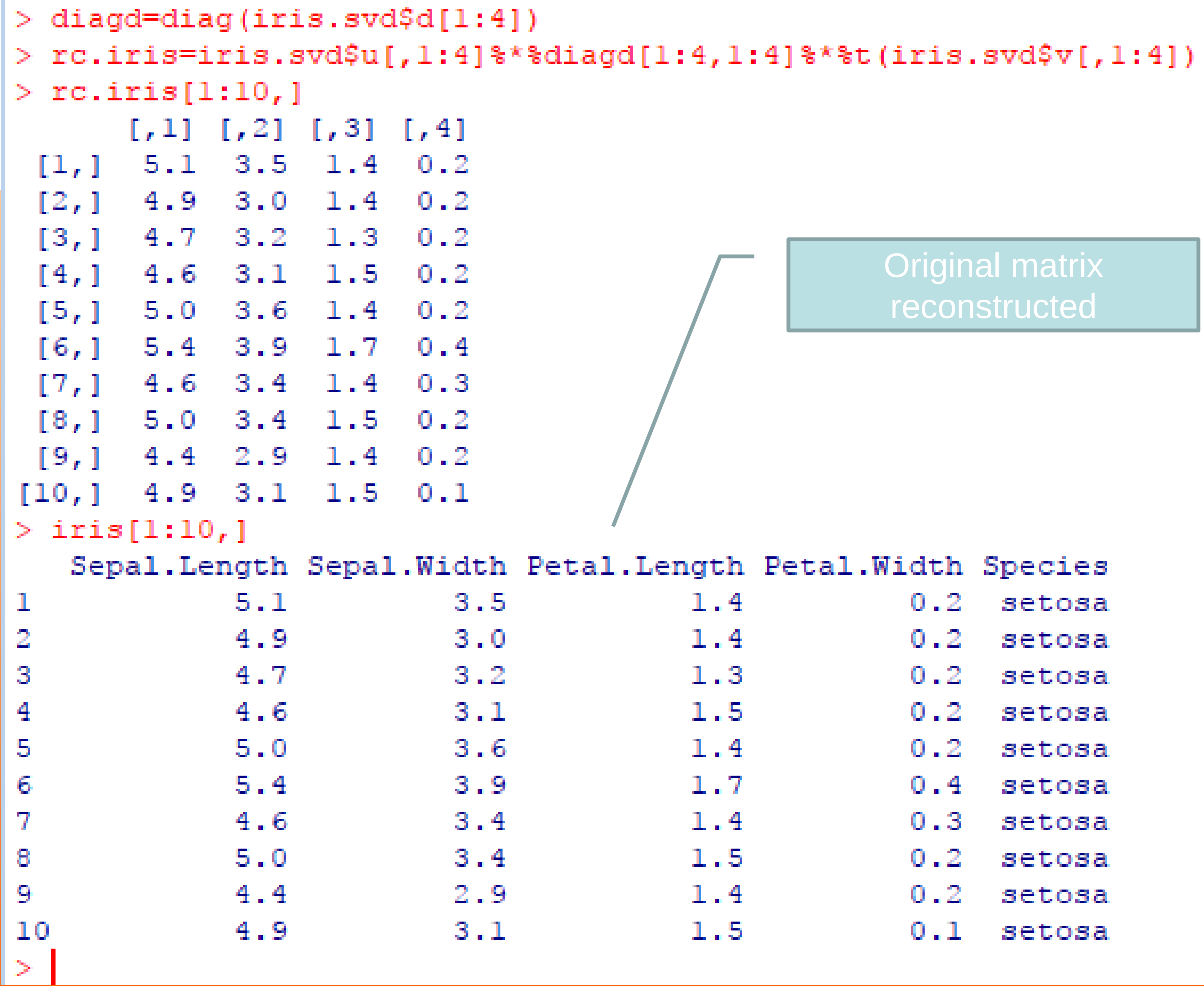


```
> names(iris.svd)
[1] "d" "u" "v"
> iris.svd$d
[1] 95.959914 17.761034  3.460931  1.884826
> dim(iris.svd$u)
[1] 150   4
> dim(iris.svd$v)
[1] 4 4
> diagd=diag(iris.svd$d[1:3])
> diagd
         [,1]     [,2]     [,3]
[1,] 95.95991  0.00000 0.000000
[2,]  0.00000 17.76103 0.000000
[3,]  0.00000  0.00000 3.460931
> rc.iris=iris.svd$u[,1:3]%*%diagd[1:3,1:3]%*%t(iris.svd$v[,1:3])
> dim(rc.iris)
[1] 150   4
```



```
> iris[1:10,]
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1           5.1         3.5          1.4         0.2  setosa
2           4.9         3.0          1.4         0.2  setosa
3           4.7         3.2          1.3         0.2  setosa
4           4.6         3.1          1.5         0.2  setosa
5           5.0         3.6          1.4         0.2  setosa
6           5.4         3.9          1.7         0.4  setosa
7           4.6         3.4          1.4         0.3  setosa
8           5.0         3.4          1.5         0.2  setosa
9           4.4         2.9          1.4         0.2  setosa
10          4.9         3.1          1.5         0.1  setosa
> rc.iris[1:10,]
          [,1]     [,2]     [,3]      [,4]
 [1,] 5.099009 3.500980 1.401484 0.1976784
 [2,] 4.868704 3.030949 1.446898 0.1266527
 [3,] 4.694221 3.205715 1.308660 0.1864563
 [4,] 4.625181 3.075098 1.462266 0.2590155
 [5,] 5.019480 3.580736 1.370809 0.2456531
 [6,] 5.406872 3.893204 1.689703 0.4161048
 [7,] 4.616171 3.384008 1.375767 0.3378993
 [8,] 5.014547 3.385615 1.478202 0.2340918
 [9,] 4.409987 2.890124 1.385035 0.2234047
[10,] 4.918426 3.081779 1.472389 0.1431833
> 
```

Some loss



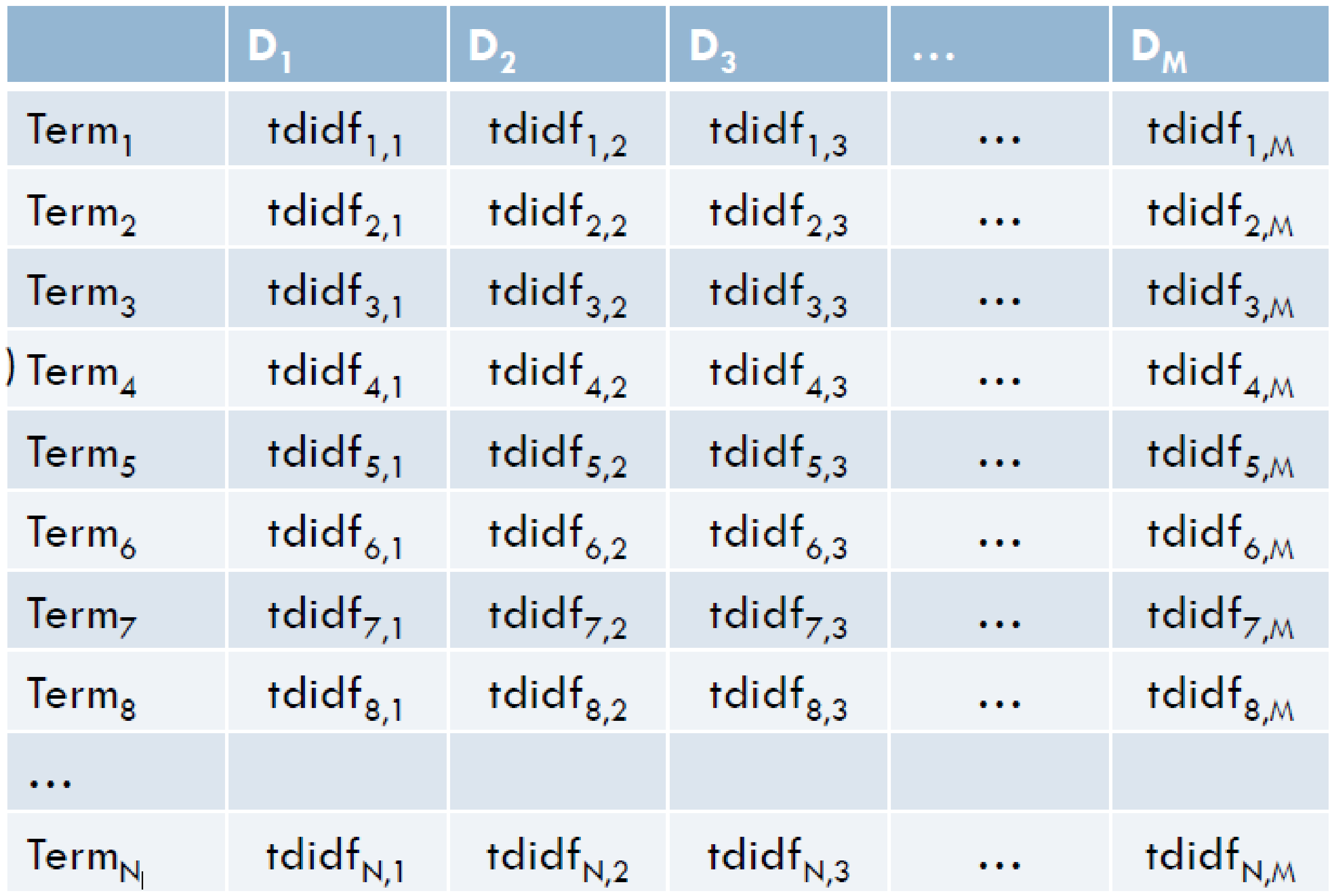
```
> diagd=diag(iris.svd$d[1:4])
> rc.iris=iris.svd$u[,1:4]%*%diagd[1:4,1:4]%*%t(iris.svd$v[,1:4])
> rc.iris[1:10,]
      [,1] [,2] [,3] [,4]
 [1,]  5.1  3.5  1.4  0.2
 [2,]  4.9  3.0  1.4  0.2
 [3,]  4.7  3.2  1.3  0.2
 [4,]  4.6  3.1  1.5  0.2
 [5,]  5.0  3.6  1.4  0.2
 [6,]  5.4  3.9  1.7  0.4
 [7,]  4.6  3.4  1.4  0.3
 [8,]  5.0  3.4  1.5  0.2
 [9,]  4.4  2.9  1.4  0.2
[10,]  4.9  3.1  1.5  0.1
> iris[1:10,]
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1           5.1         3.5          1.4         0.2  setosa
2           4.9         3.0          1.4         0.2  setosa
3           4.7         3.2          1.3         0.2  setosa
4           4.6         3.1          1.5         0.2  setosa
5           5.0         3.6          1.4         0.2  setosa
6           5.4         3.9          1.7         0.4  setosa
7           4.6         3.4          1.4         0.3  setosa
8           5.0         3.4          1.5         0.2  setosa
9           4.4         2.9          1.4         0.2  setosa
10          4.9         3.1          1.5         0.1  setosa
> 
```

Original matrix reconstructed

- SVD is a means to the end goal.

- The end goal is dimension reduction, i.e. get another version of term-document matrix [in different space] with lower rank k [100, 300, 500]

Term – Document Matrix



	D_1	D_2	D_3	…	D_M
$Term_1$	$tdidf_{1,1}$	$tdidf_{1,2}$	$tdidf_{1,3}$	…	$tdidf_{1,M}$
$Term_2$	$tdidf_{2,1}$	$tdidf_{2,2}$	$tdidf_{2,3}$	…	$tdidf_{2,M}$
$Term_3$	$tdidf_{3,1}$	$tdidf_{3,2}$	$tdidf_{3,3}$	…	$tdidf_{3,M}$
$Term_4$	$tdidf_{4,1}$	$tdidf_{4,2}$	$tdidf_{4,3}$	…	$tdidf_{4,M}$
$Term_5$	$tdidf_{5,1}$	$tdidf_{5,2}$	$tdidf_{5,3}$	…	$tdidf_{5,M}$
$Term_6$	$tdidf_{6,1}$	$tdidf_{6,2}$	$tdidf_{6,3}$	…	$tdidf_{6,M}$
$Term_7$	$tdidf_{7,1}$	$tdidf_{7,2}$	$tdidf_{7,3}$	…	$tdidf_{7,M}$
$Term_8$	$tdidf_{8,1}$	$tdidf_{8,2}$	$tdidf_{8,3}$	…	$tdidf_{8,M}$
…					
$Term_N$	$tdidf_{N,1}$	$tdidf_{N,2}$	$tdidf_{N,3}$	…	$tdidf_{N,M}$

Given N x M matrix

Fig Ref : Crista Lopes

N and M used differently in later slides

K=6

	D_1	D_2	D_3	…	D_M
Concept_1	$v_{1,1}$	$v_{1,2}$	$v_{1,3}$	…	$v_{1,M}$
Concept_2	$v_{2,1}$	$v_{2,2}$	$v_{2,3}$	…	$v_{2,M}$
Concept_3	$v_{3,1}$	$v_{3,2}$	$v_{3,3}$	…	$v_{3,M}$
Concept_4	$v_{4,1}$	$v_{4,2}$	$v_{4,3}$	…	$v_{4,M}$
Concept_5	$v_{5,1}$	$v_{5,2}$	$v_{5,3}$	…	$v_{5,M}$
Concept_6	$v_{6,1}$	$v_{6,2}$	$v_{6,3}$	…	$v_{6,M}$

Squeeze terms such that they reflect **concepts**

Query matching is performed in the concept space too

PCA ?

Described in a different way….

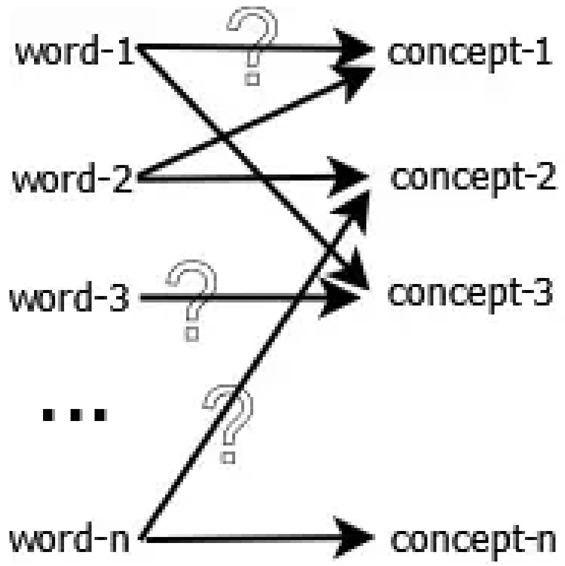
- Latent semantic indexing works by projecting this large, multidimensional space down into a smaller number of dimensions.

- In doing so, keywords that are semantically similar will get squeezed together

- Since authors have a wide choice of words available when they write, the concepts can be obscured due to different word choices from different authors. This essentially random choice of words introduces noise into the word-concept relationship.

- Latent Semantic Analysis filters out some of this noise and also attempts to find the smallest set of concepts that spans all the documents.

- LSA maps words and documents into this smaller "concept" space and does the comparison in this space.



For example, the word bank when used together with mortgage, loans, and rates probably means a financial institution. However, the word bank when used together with lures, casting, and fish probably means a stream or river bank

Singular Value Decomposition

For an $m \times n$ matrix $\mathbf{A}$ of rank r there exists a factorization (Singular Value Decomposition = **SVD**) as follows:

$$A = \underbrace{U}_{m \times n} \underbrace{\Sigma}_{n \times n} \underbrace{V^T}_{V \text{ is } n \times n}$$

Eigenvalues $\lambda_1 \ldots \lambda_r$ of $\boldsymbol{AA^T}$ are the eigenvalues of $\boldsymbol{A^TA}$.

$$\sigma_i = \sqrt{\lambda_i}$$

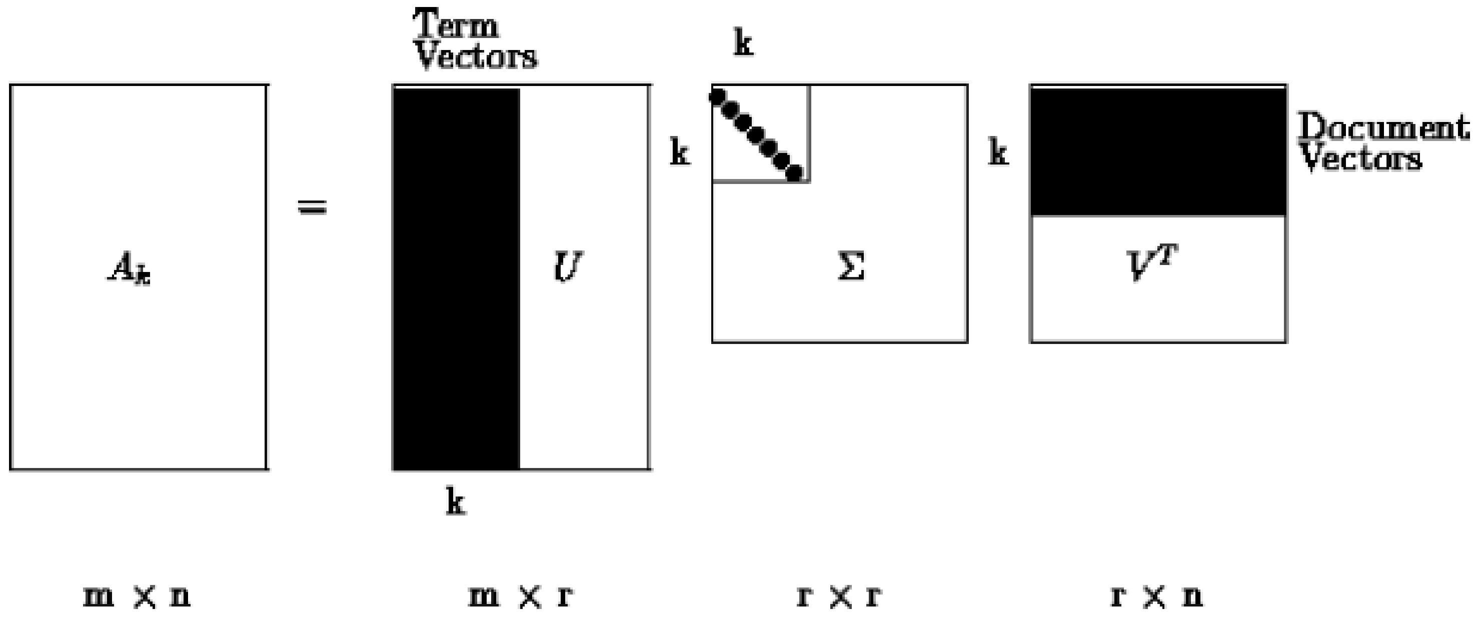
$$\Sigma = diag(\sigma_1 \ldots \sigma_r)$$

Singular values.

What is it?

- From term-doc matrix A, we compute the approximation A_{k}.
- There is a row for each term and a column for each doc in A_k
- Thus docs live in a space of $k<<r$ dimensions
  - These dimensions are not the original axes

SVD – Visual



Term x Document Matrix	Term x Factor Matrix	Singular Values Matrix	Factor x Document Matrix

- Singular values represent the strength of latent concepts in the corpus. Each concept emerges from word co-occurrences. (hence the word “latent”)

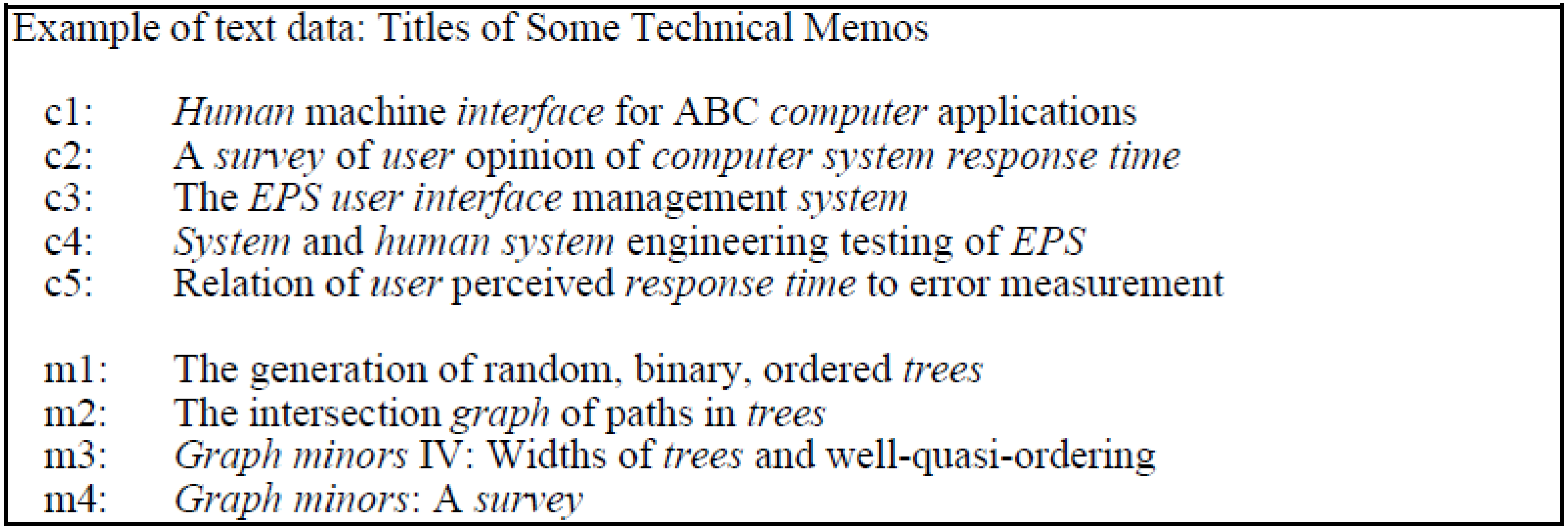
- By truncating, we are selecting the k strongest concepts

- Usually in low hundreds

- When forced to squeeze the terms/documents down to a k-dimensional space, the SVD should bring together terms with similar co-occurrences.

- The Example

Dataset



Example of text data: Titles of Some Technical Memos

c1:	*Human* machine *interface* for ABC *computer* applications
c2:	A *survey* of *user* opinion of *computer system response time*
c3:	The *EPS user interface* management *system*
c4:	*System* and *human system* engineering testing of *EPS*
c5:	Relation of *user* perceived *response time* to error measurement
m1:	The generation of random, binary, ordered *trees*
m2:	The intersection *graph* of paths in *trees*
m3:	*Graph minors* IV: Widths of *trees* and well-quasi-ordering
m4:	*Graph minors*: A *survey*

Deerwester et al., 1990; Landauer & Dumais

Example of text data: Titles of Some Technical Memos

c1:	*Human* machine *interface* for ABC *computer* applications
c2:	A *survey* of *user* opinion of *computer system response time*
c3:	The *EPS user interface* management *system*
c4:	*System* and *human system* engineering testing of *EPS*
c5:	Relation of *user* perceived *response time* to error measurement
m1:	The generation of random, binary, ordered *trees*
m2:	The intersection *graph* of paths in *trees*
m3:	*Graph minors* IV: Widths of *trees* and well-quasi-ordering
m4:	*Graph minors*: A *survey*

$$\{X\} =$$

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

12x9 T-D Matrix

12 terms
9 documents

0.22	-0.11	0.29	-0.41	-0.11	-0.34	0.52	-0.06	-0.41
0.20	-0.07	0.14	-0.55	0.28	0.50	-0.07	-0.01	-0.11
0.24	0.04	-0.16	-0.59	-0.11	-0.25	-0.30	0.06	0.49
0.40	0.06	-0.34	0.10	0.33	0.38	0.00	0.00	0.01
0.64	-0.17	0.36	0.33	-0.16	-0.21	-0.17	0.03	0.27
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.30	-0.14	0.33	0.19	0.11	0.27	0.03	-0.02	-0.17
0.21	0.27	-0.18	-0.03	-0.54	0.08	-0.47	-0.04	-0.58
0.01	0.49	0.23	0.03	0.59	-0.39	-0.29	0.25	-0.23
0.04	0.62	0.22	0.00	-0.07	0.11	0.16	-0.68	0.23
0.03	0.45	0.14	-0.01	-0.30	0.28	0.34	0.68	0.18

12x9

3.34								
	2.54							
		2.35						
			1.64					
				1.50				
					1.31			
						0.85		
							0.56	
								0.36

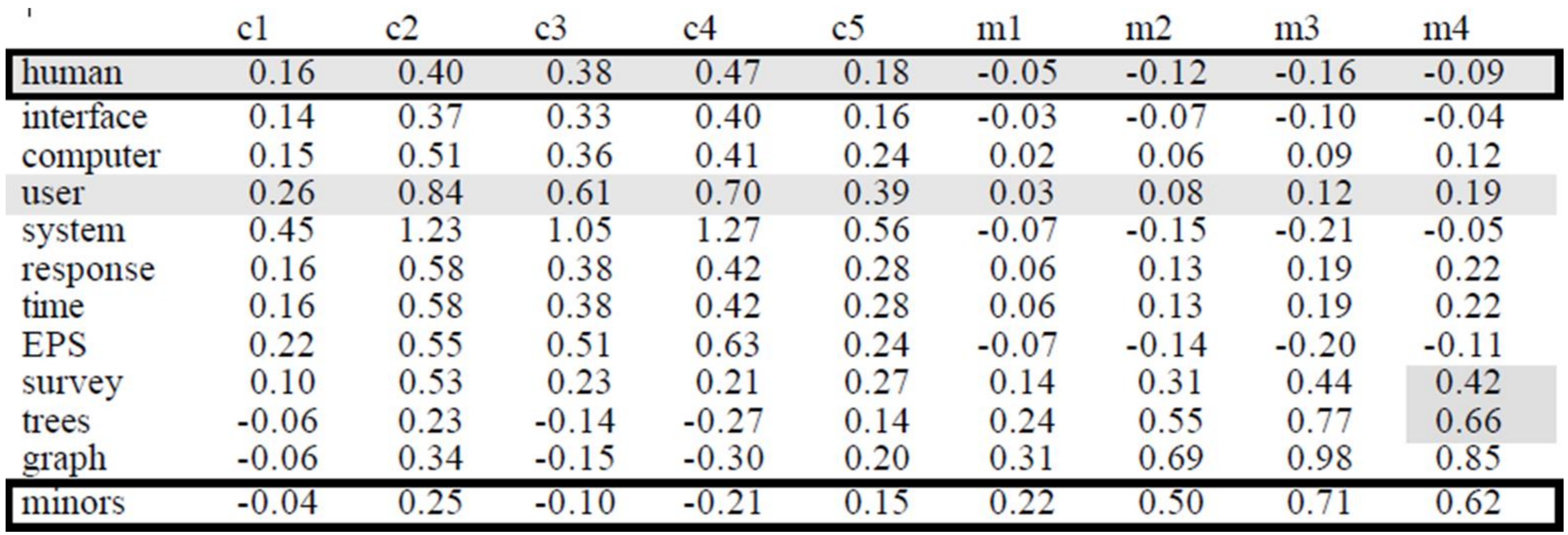
9x9

V

$$
\begin{array}{ccccccccc}
0.20 & -0.06 & 0.11 & -0.95 & 0.05 & -0.08 & 0.18 & -0.01 & -0.06 \\
0.61 & 0.17 & -0.50 & -0.03 & -0.21 & -0.26 & -0.43 & 0.05 & 0.24 \\
0.46 & -0.13 & 0.21 & 0.04 & 0.38 & 0.72 & -0.24 & 0.01 & 0.02 \\
0.54 & -0.23 & 0.57 & 0.27 & -0.21 & -0.37 & 0.26 & -0.02 & -0.08 \\
0.28 & 0.11 & -0.51 & 0.15 & 0.33 & 0.03 & 0.67 & -0.06 & -0.26 \\
0.00 & 0.19 & 0.10 & 0.02 & 0.39 & -0.30 & -0.34 & 0.45 & -0.62 \\
0.01 & 0.44 & 0.19 & 0.02 & 0.35 & -0.21 & -0.15 & -0.76 & 0.02 \\
0.02 & 0.62 & 0.25 & 0.01 & 0.15 & 0.00 & 0.25 & 0.45 & 0.52 \\
0.08 & 0.53 & 0.08 & -0.03 & -0.60 & 0.36 & 0.04 & -0.07 & -0.45
\end{array}
$$

9x9

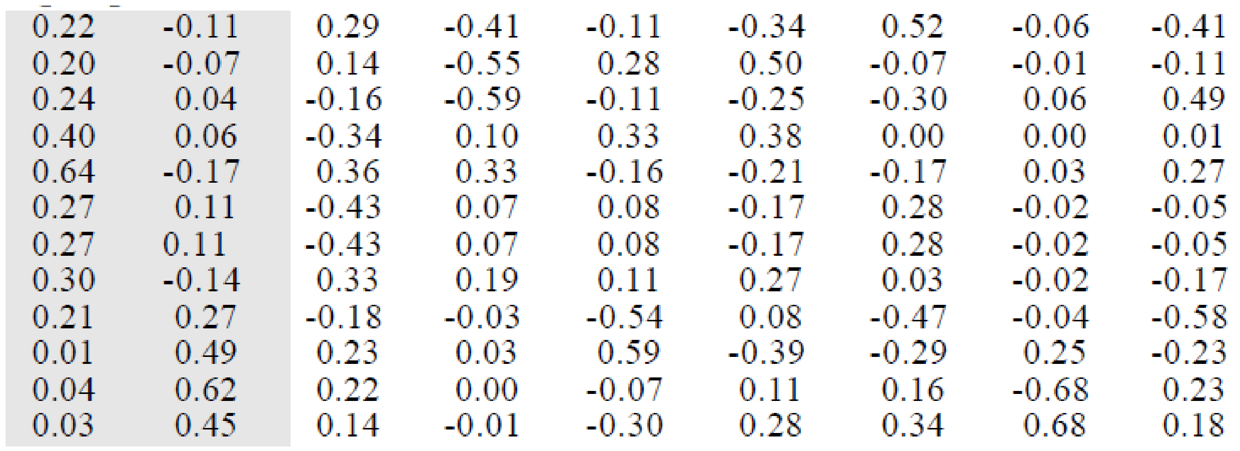
Recomputed original matrix with only 2 dim



	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	0.16	0.40	0.38	0.47	0.18	-0.05	-0.12	-0.16	-0.09
interface	0.14	0.37	0.33	0.40	0.16	-0.03	-0.07	-0.10	-0.04
computer	0.15	0.51	0.36	0.41	0.24	0.02	0.06	0.09	0.12
user	0.26	0.84	0.61	0.70	0.39	0.03	0.08	0.12	0.19
system	0.45	1.23	1.05	1.27	0.56	-0.07	-0.15	-0.21	-0.05
response	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
time	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
EPS	0.22	0.55	0.51	0.63	0.24	-0.07	-0.14	-0.20	-0.11
survey	0.10	0.53	0.23	0.21	0.27	0.14	0.31	0.44	0.42
trees	-0.06	0.23	-0.14	-0.27	0.14	0.24	0.55	0.77	0.66
graph	-0.06	0.34	-0.15	-0.30	0.20	0.31	0.69	0.98	0.85
minors	-0.04	0.25	-0.10	-0.21	0.15	0.22	0.50	0.71	0.62

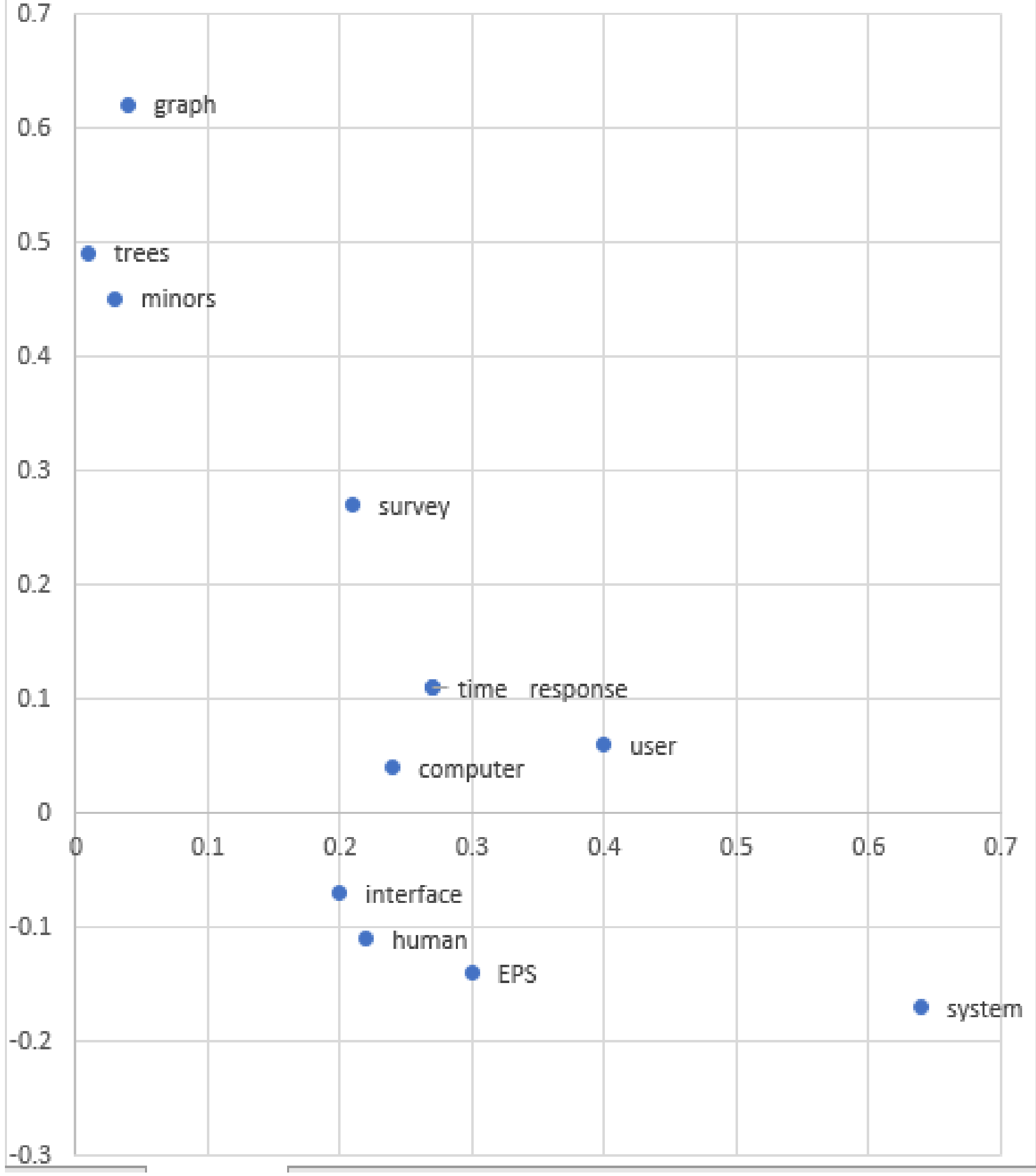
Look at the two shaded cells for survey and trees in column m4. The word tree did not appear in this graph theory title. But because m4 did contain graph and minors, the zero entry for tree has been replaced with 0.66, which can be viewed as an estimate of how many times it would occur in each of an infinite sample of titles containing graph and minors. By contrast, the value 1.00 for survey, which appeared once in m4, has been replaced by 0.42 reflecting the fact that it is unexpected in this context and should be counted as unimportant in characterizing the passage.

How to visualize documents and terms in new dimensional space ?



0.22	-0.11	0.29	-0.41	-0.11	-0.34	0.52	-0.06	-0.41
0.20	-0.07	0.14	-0.55	0.28	0.50	-0.07	-0.01	-0.11
0.24	0.04	-0.16	-0.59	-0.11	-0.25	-0.30	0.06	0.49
0.40	0.06	-0.34	0.10	0.33	0.38	0.00	0.00	0.01
0.64	-0.17	0.36	0.33	-0.16	-0.21	-0.17	0.03	0.27
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.30	-0.14	0.33	0.19	0.11	0.27	0.03	-0.02	-0.17
0.21	0.27	-0.18	-0.03	-0.54	0.08	-0.47	-0.04	-0.58
0.01	0.49	0.23	0.03	0.59	-0.39	-0.29	0.25	-0.23
0.04	0.62	0.22	0.00	-0.07	0.11	0.16	-0.68	0.23
0.03	0.45	0.14	-0.01	-0.30	0.28	0.34	0.68	0.18

This is the U [SVD Term Matrix] after SVD decomposition
Let k=2
Focus on grey columns



	LSI Dim 1	LSI Dim 2
human	0.22	-0.11
interface	0.2	-0.07
computer	0.24	0.04
user	0.4	0.06
system	0.64	-0.17
response	0.27	0.11
time	0.27	0.11
EPS	0.3	-0.14
survey	0.21	0.27
trees	0.01	0.49
graph	0.04	0.62
minors	0.03	0.45

Concept 1

Concept 2

So what all we have learnt ?

Web:	http://www.insofe.edu.in
Facebook:	https://www.facebook.com/insofe
Twitter:	https://twitter.com/Insofeedu
YouTube:	http://www.youtube.com/InsofeVideos
SlideShare:	http://www.slideshare.net/INSOFE
LinkedIn:	http://www.linkedin.com/company/international-school-of-engineering