How does text become data?

Rob Speer
Luminoso / ConceptNet
rob@luminoso.com

r bæði í land
nilega ekki upp við velgeng
ekki. Margrét stundar nám við
u af frítíma sínum í fótboltaæfi
ð sé svona heillandi við fótbolt
er finnst svo gaman að spila fót
ð fer auðvitað mikill tími í æfin
vini mína. . . ,' Margrét sér fr
um. Hana langar að fara til
ðurlöndin eru

Motivation: Data-driven questions

- Suppose you have a stream of customer support messages coming in.
- What if you consider these messages as a data source?

Classification

- Is this message angry?
- How many angry messages do we receive per day?

Similarity between documents

- How often do we receive messages like this one?
- What’s a typical response to messages like this?

Similarity between terms

- Is this a request about accounts, billing, etc?
- ... but not necessarily using those exact words?
- Are we receiving an unexpected number of requests like this?

In this talk

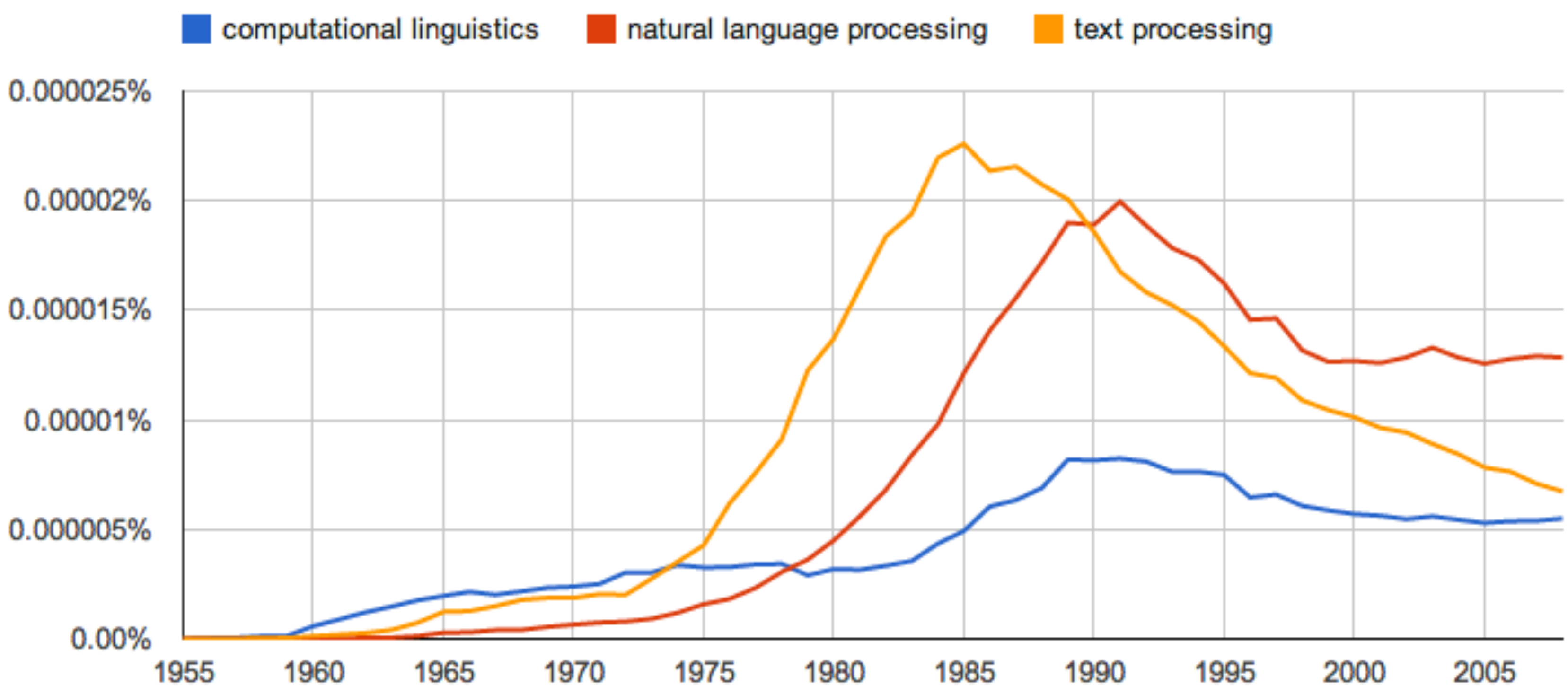
- A tour of useful data-driven NLP techniques
- ... using a small amount of Python code
  - If these solutions seem simplistic, they are!
- Don’t worry, code is online:

  http://github.com/rspeer/text-as-data

How is the text represented?

Simple word counts

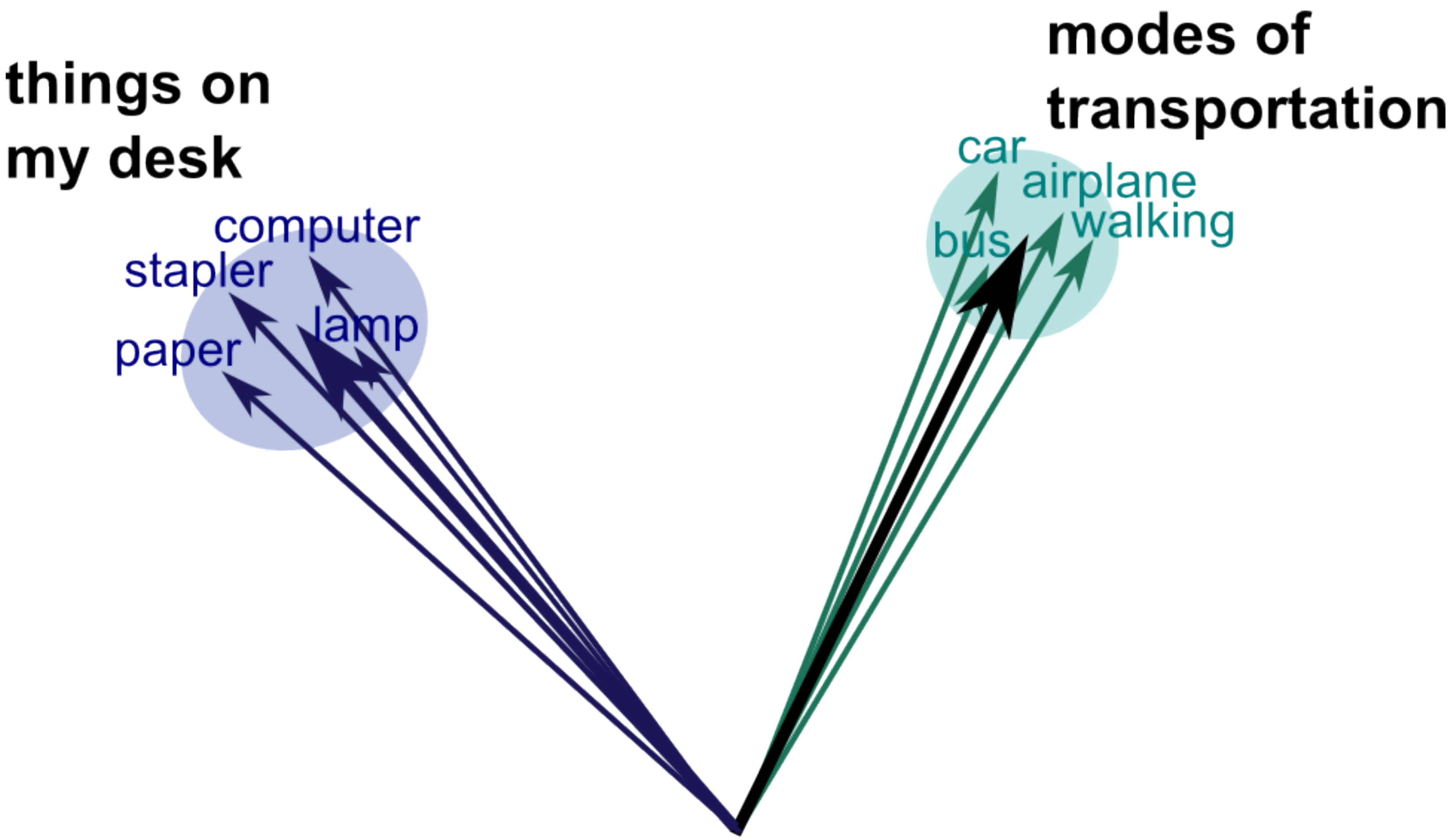
N-gram models



Term-document matrices

	woe	betray	vengeance	death	alas
Julius Caesar	2	1	0	29	8
Hamlet	8	0	2	37	9
Macbeth	2	2	0	20	4

Vector space models



Python example: word splitting and normalizing

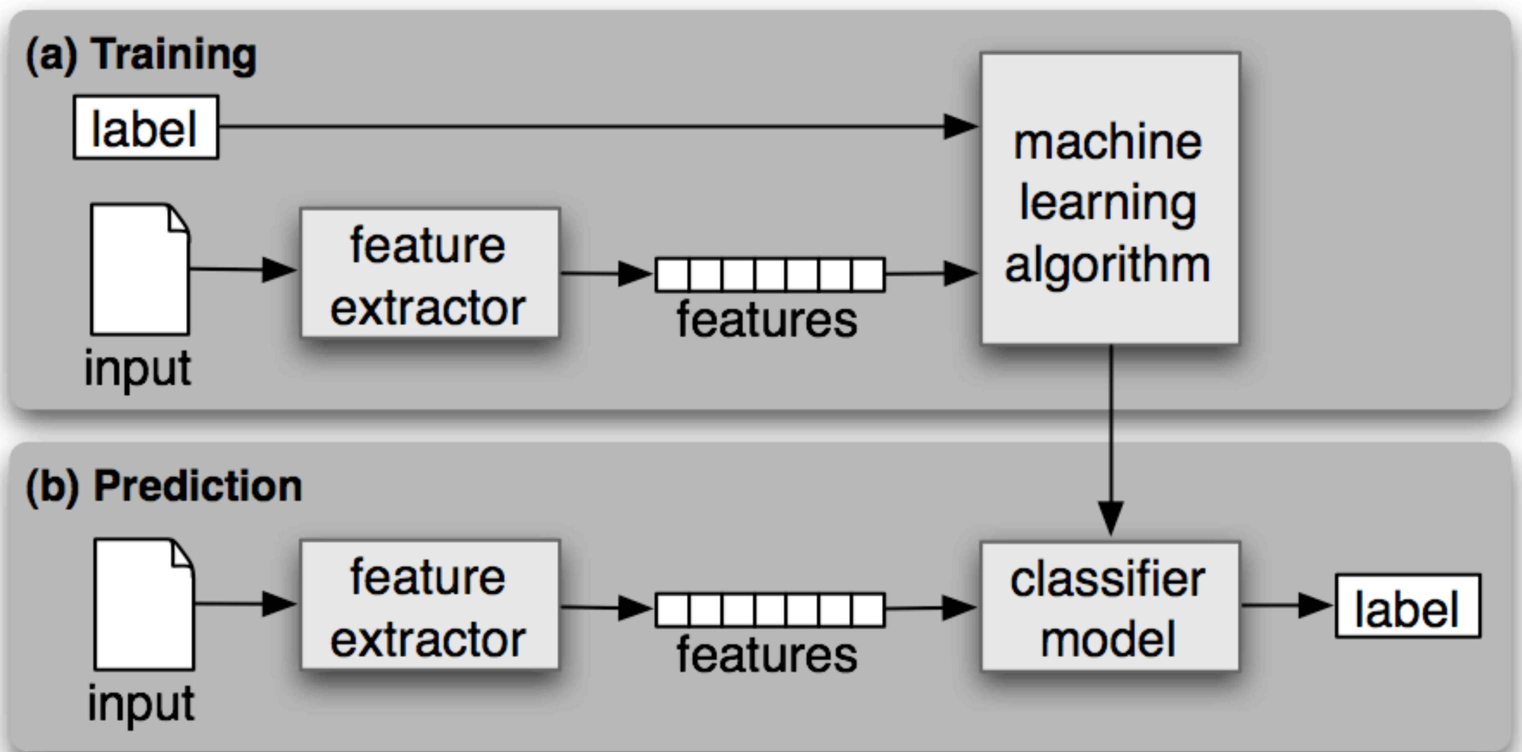
Which N-grams are interesting?

Consider this contingency table:

p(**vice**, **president**)	*p*(**vice**, ~president)
p(~vice, **president**)	*p*(~vice, ~president)

Python example: interesting N-grams

Text classification

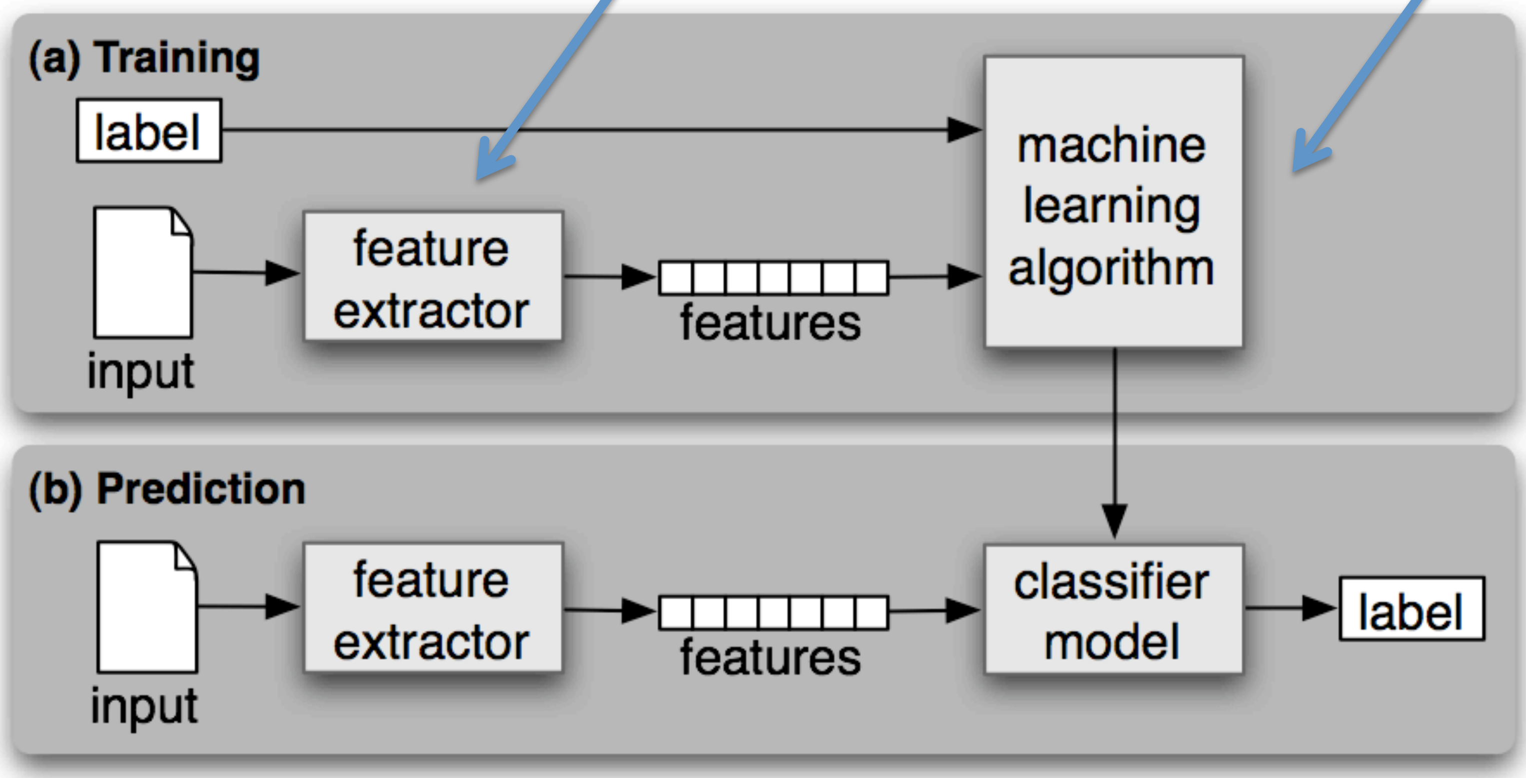


from “Natural Language Processing with Python”,
by Steven Bird, Ewan Klein, and Edward Loper (O’Reilly, 2009)

Text classification

almost certainly
bag-of-words

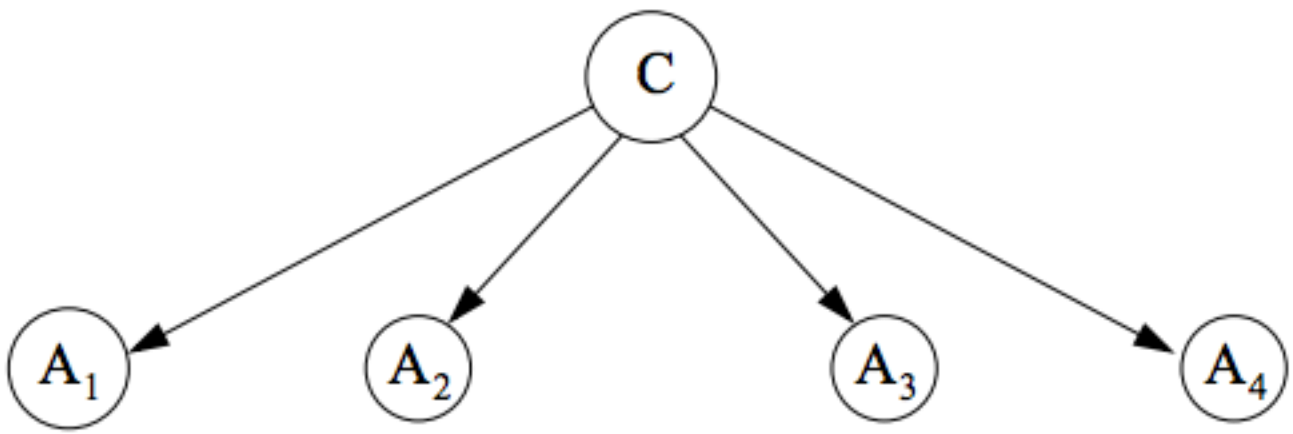
possibly
Naive Bayes



from “Natural Language Processing with Python”,
by Steven Bird, Ewan Klein, and Edward Loper (O’Reilly, 2009)

Overview of Naïve Bayes classification

- The probability that a document is in class *C* depends on its features, A_n
- Assume all features are statistically independent



Python example: Classification with NLTK and scikit-learn

What about stopwords?

- Shouldn’t we remove common words such as “the” and “of”?
- It could help
- It could be premature optimization

Text similarity

- Bags of words can tell us how similar documents are

	woe	**betray**	**vengeance**	**death**	**alas**
Julius Caesar	2	1	0	29	8
Hamlet	8	0	2	37	9
Macbeth	2	2	0	20	4

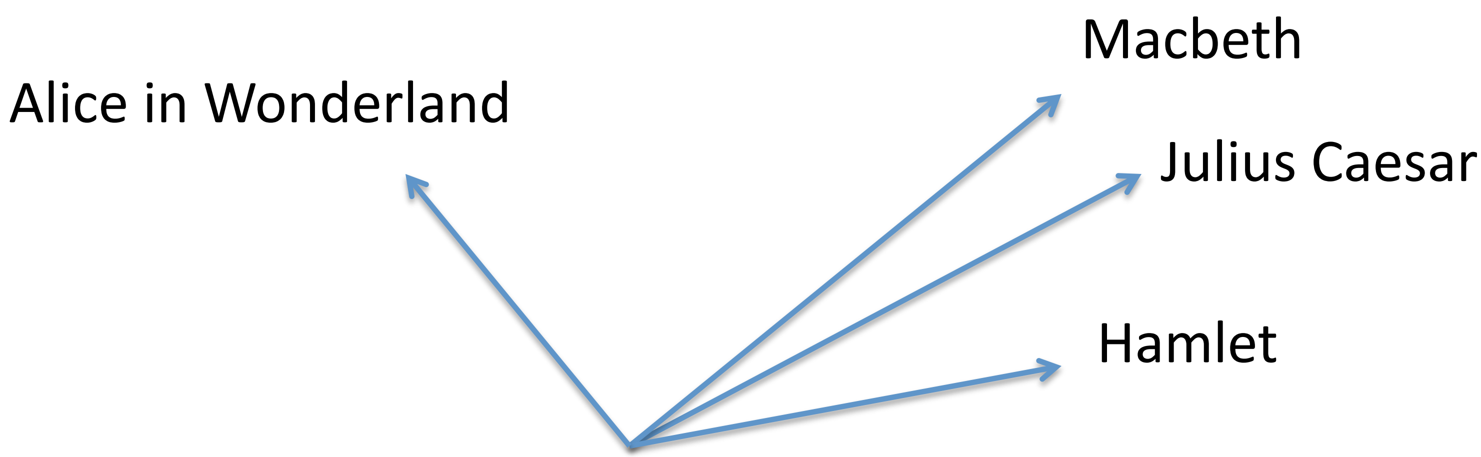
Text similarity

- Bags of words can tell us how similar documents are

	woe	betray	vengeance	death	alas
Julius Caesar	2	1	0	29	8
Hamlet	8	0	2	37	9
Macbeth	2	2	0	20	4
Alice in Wonderland	0	0	0	1	4

Vector-space similarity

- Similar texts have a small angle between them

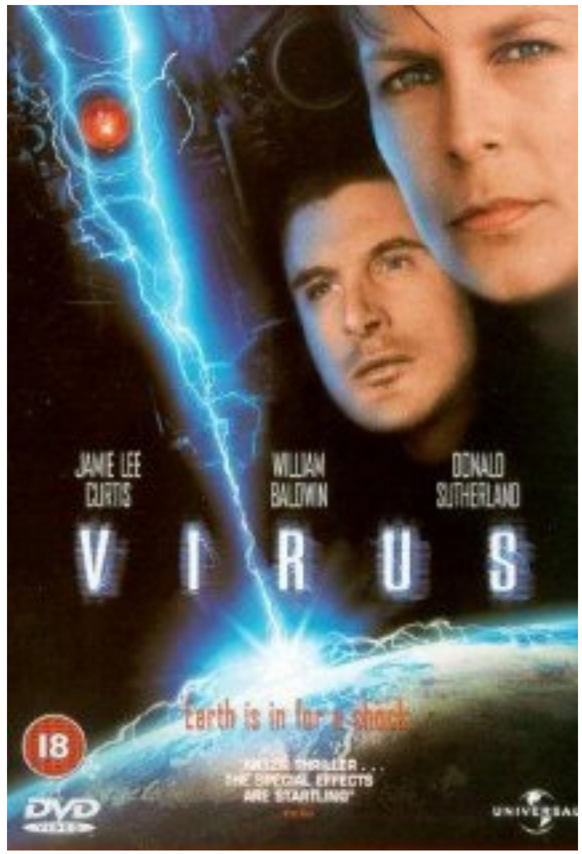


Dimensionality reduction

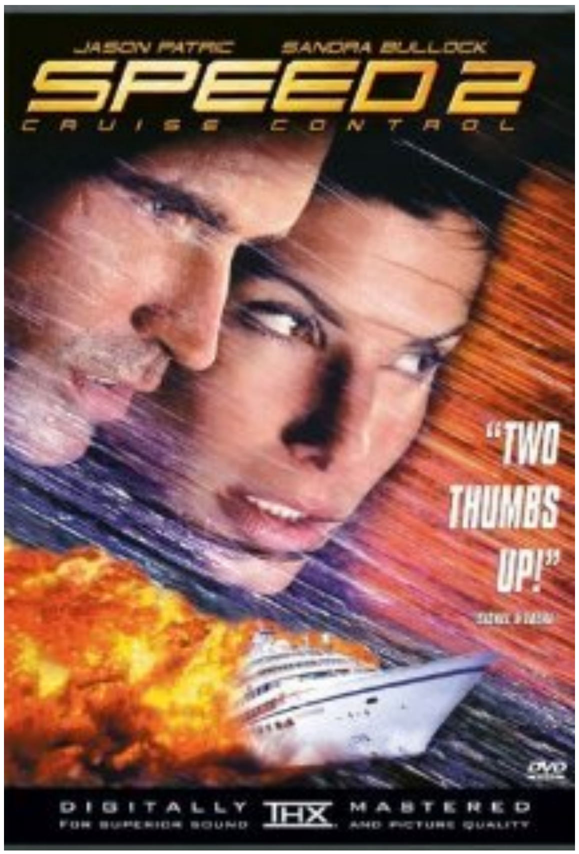
- Put terms and documents in a lower-dimensional space where we can easily compare them
- In NLP, this is called Latent Semantic Analysis or Latent Semantic Inference

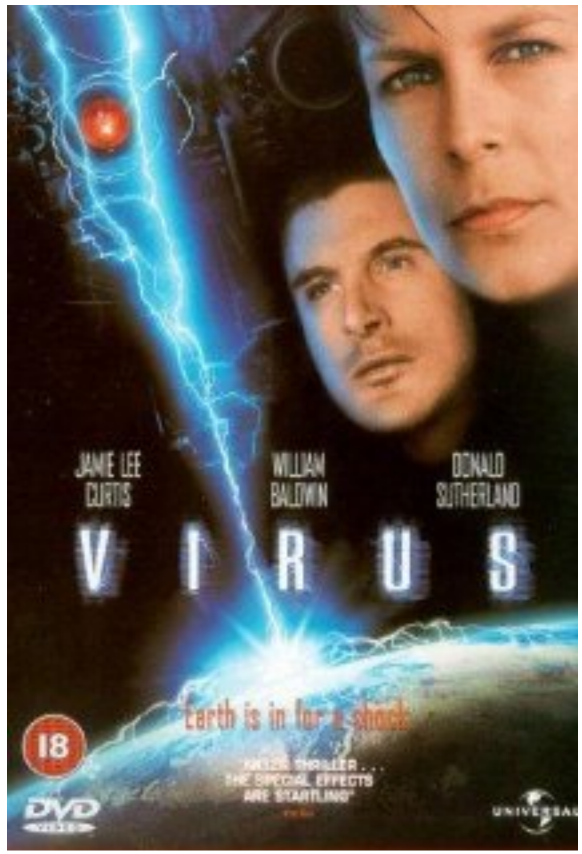
Python example: Unsupervised text similarity using gensim

Similarity of movie reviews

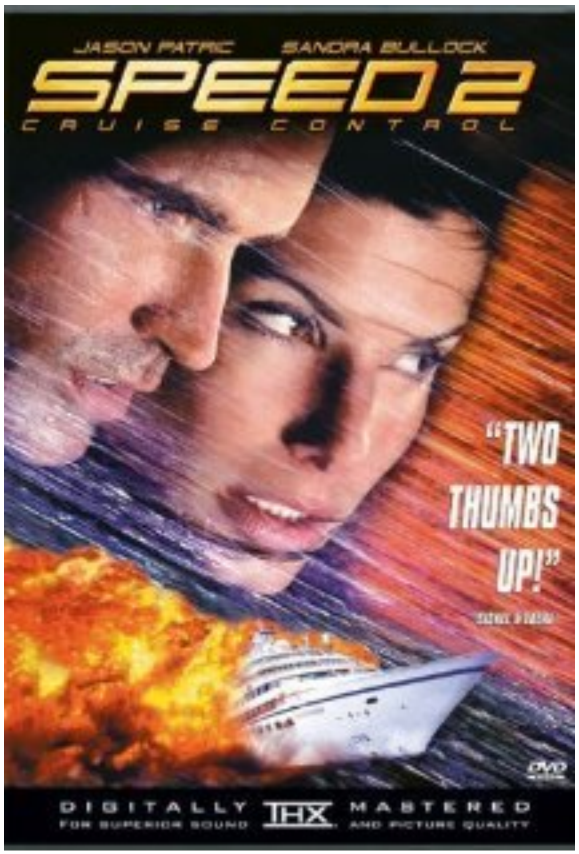


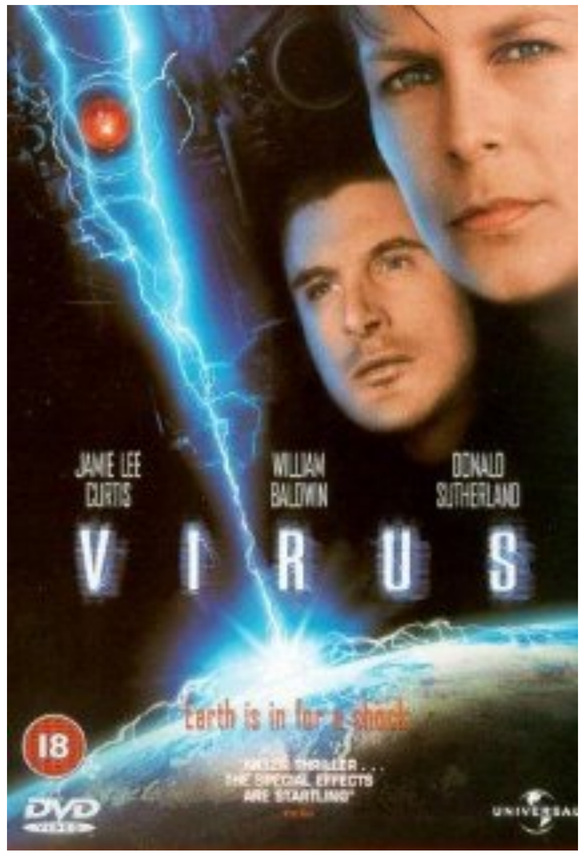
Similarity of movie reviews

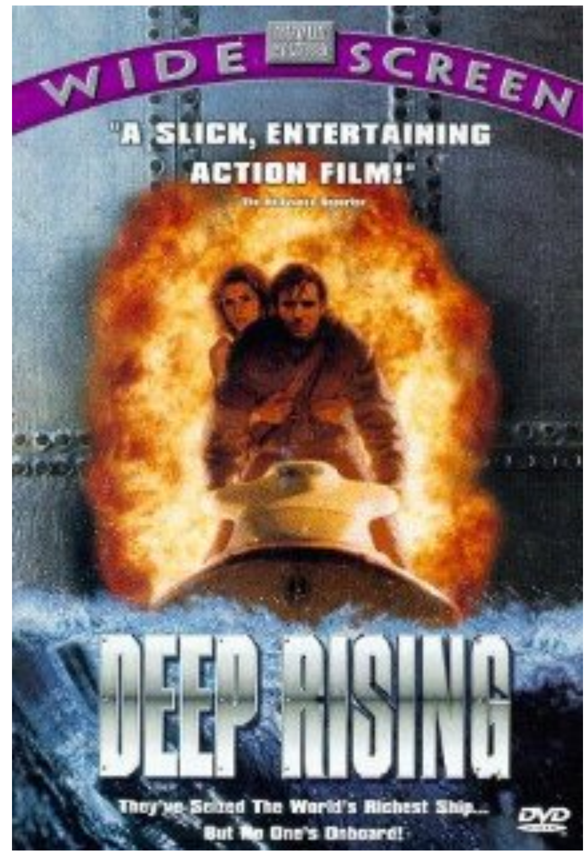




Similarity of movie reviews







Word associations

Word associations

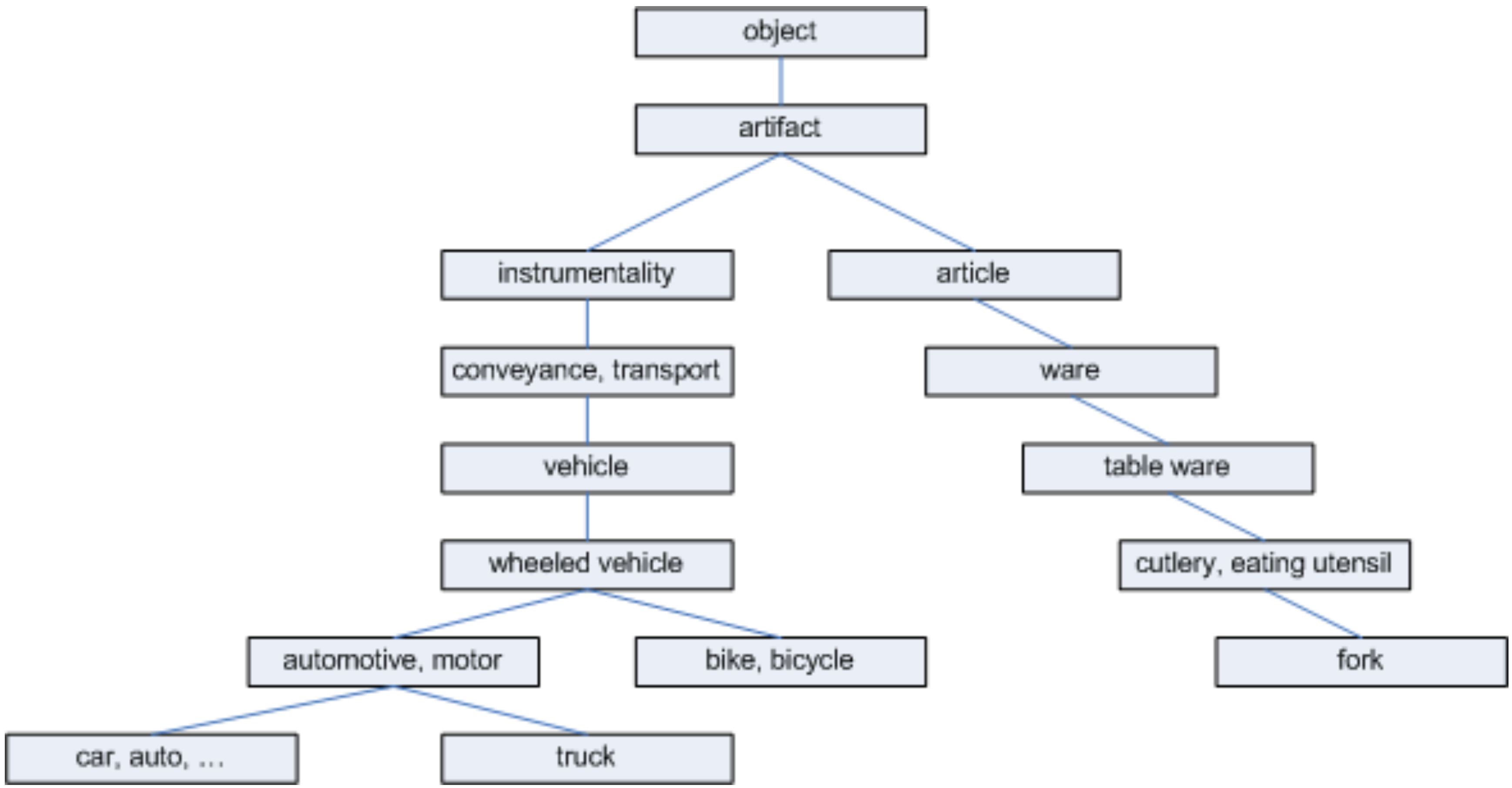
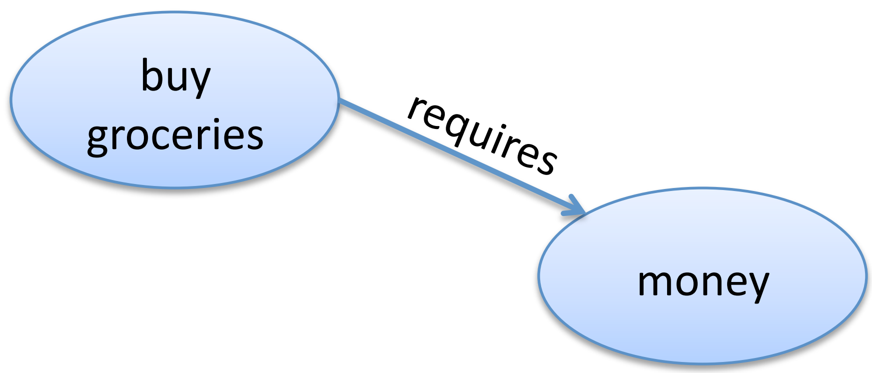
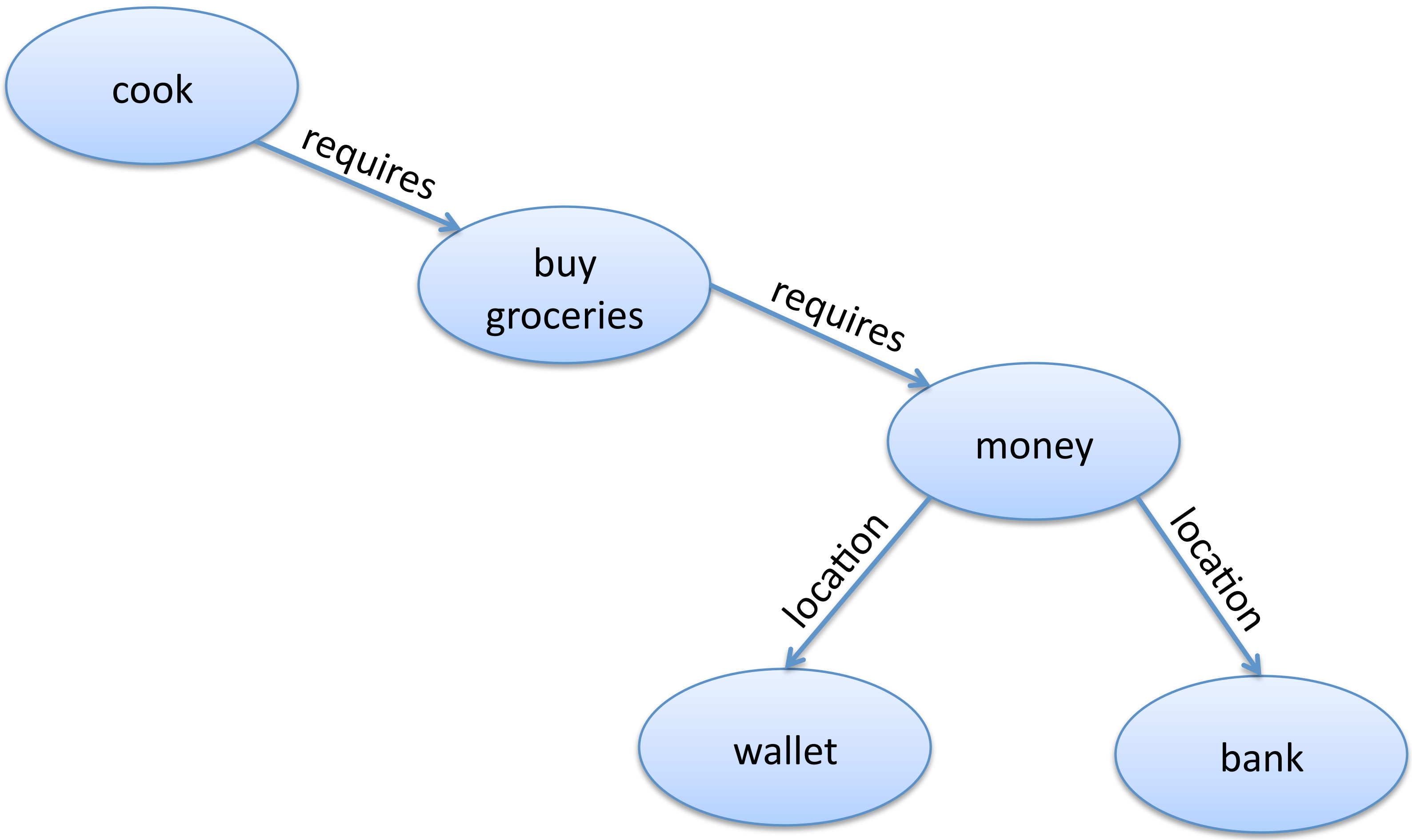
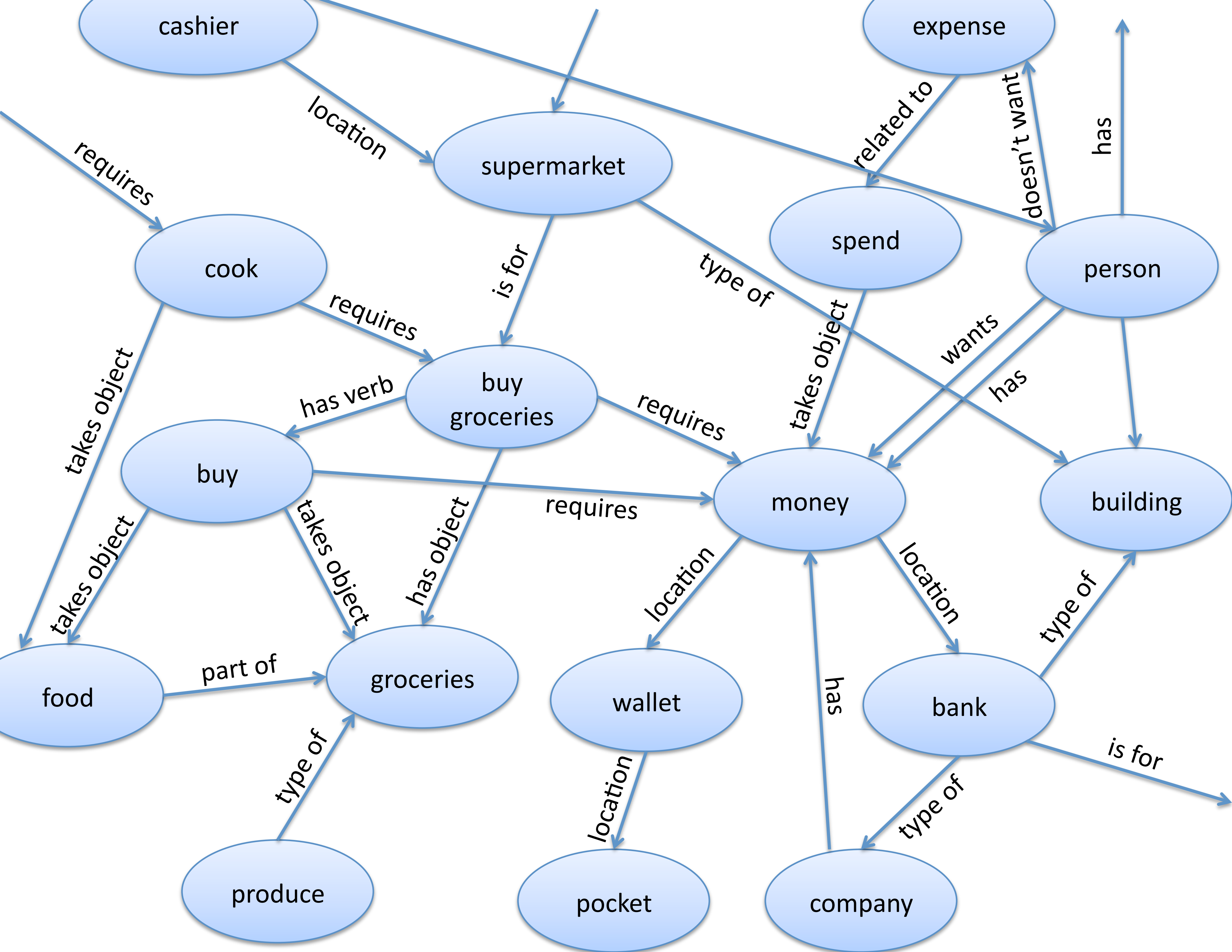


Image source: "WordNet-based semantic similarity measurement"
by Troy Simpson and Thanh Dao, on codeproject.com

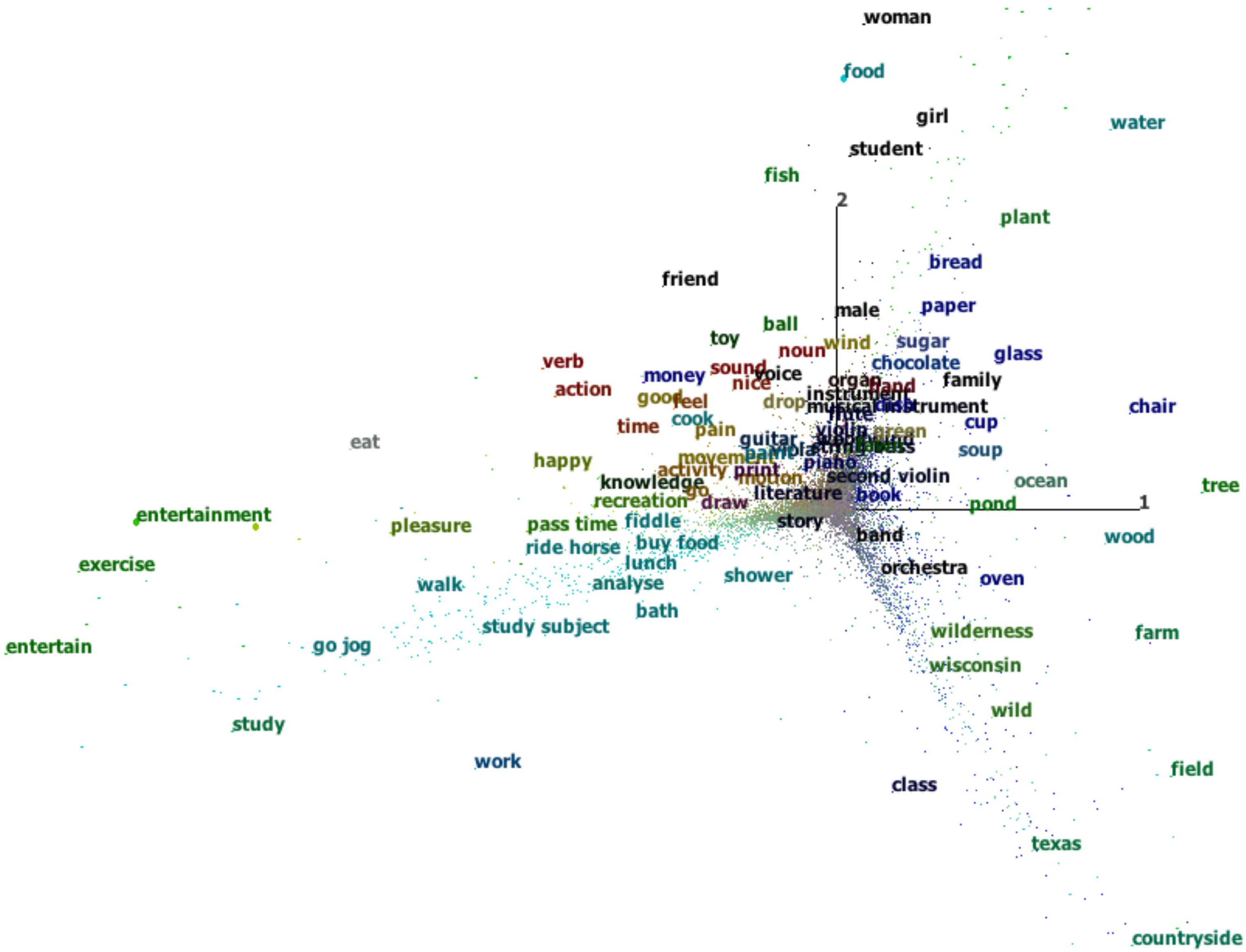
Python example: Querying WordNet


buy
groceries
requires
money


cook
requires
buy groceries
requires
money
location
location
wallet
bank


cashier
expense
location
supermarket
requires
related to
doesn't want
has
spend
cook
is for
type of
person
requires
takes object
wants
buy
groceries
has verb
has
takes object
requires
buy
money
building
requires
takes object
has object
takes object
location
location
type of
food
part of
groceries
wallet
has
bank
is for
type of
location
type of
produce
pocket
company

ConceptNet as a vector space



Python example: Querying ConceptNet

- See API documentation linked from:
  http://conceptnet5.media.mit.edu

Many incompatible representations

- Supervised text classification
- Unsupervised document similarity
- Domain-general word associations

Many incompatible representations

- Supervised text classification
- Unsupervised document similarity
- Domain-general word associations

- It would be nice if one model could do all of these.

LUMINOSO

NLP with "batteries included"

- **nltk** (the basics)
- **scikit-learn** (classification)
- **gensim** (text similarity)
- Interfaces to **WordNet** and **ConceptNet** (word associations)

What is Python missing?

- A good search index.

What is Python missing?

- A good search index.
- Recommendation: use Lucene, or something that uses Lucene.

That's all

Code and slides:

http://github.com/rspeer/text-as-data

Cool things I work on:

http://conceptnet5.media.mit.edu

http://luminoso.com

Extra slides

TF-IDF normalization

- Some documents are longer than others
- Some words appear more than others

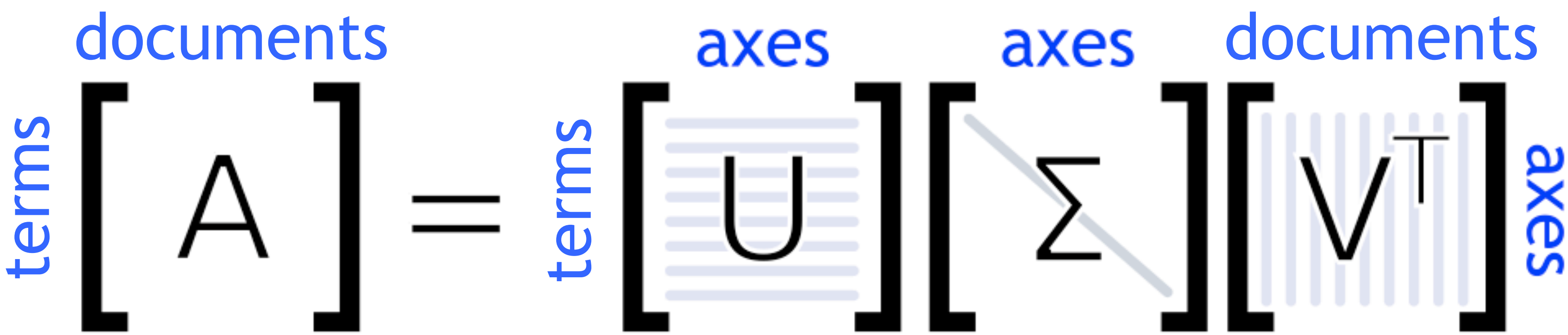
	woe	betray	vengeance	death	alas
Julius Caesar	55.0	32.9	0	0	219.9
Hamlet	38.0	0	73.1	0	171.0
Macbeth	61.4	73.5	0	0	122.7
Alice in Wonderland	0	0	0	0	83.2

(TF-IDF values from NLTK's Project Gutenberg corpus, in micro-bits per word)

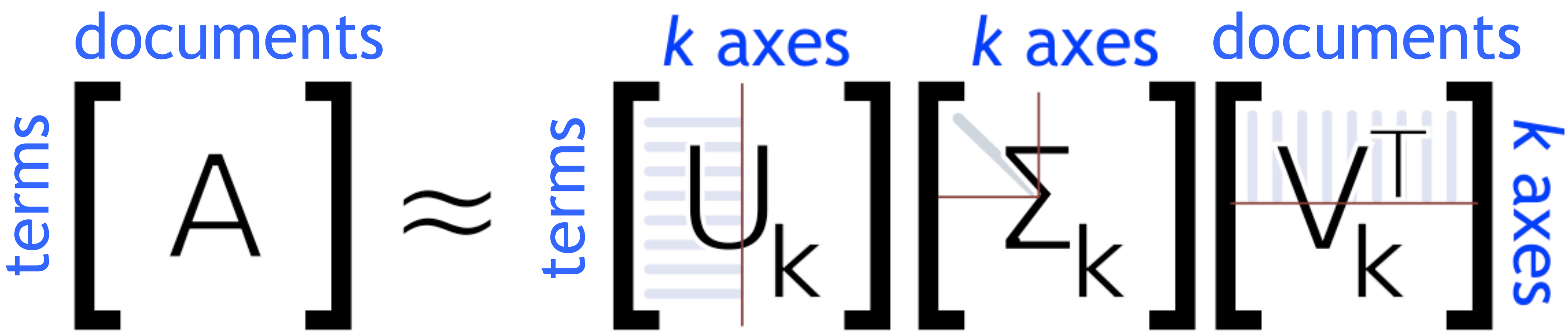
TF-IDF normalization

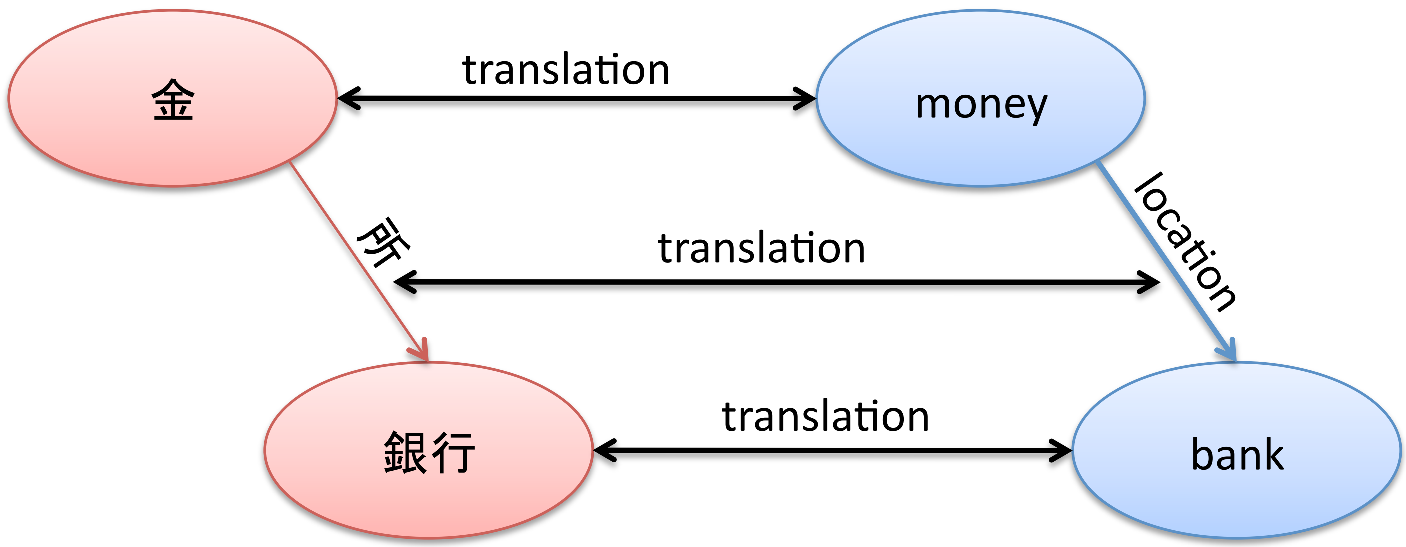
- TF replaces term counts with term frequencies
- IDF tells us how much information we get when a word appears
- In Project Gutenberg:
  - IDF(the) = 0 bits
  - IDF(vengeance) = 1.36 bits
  - IDF(whale) = 2.17 bits
  - IDF(Ishmael) = 3.17 bits

Dimensionality reduction



Dimensionality reduction




金
translation
money
所
translation
location
銀行
translation
bank

But Naïve Bayes is so naïve!

- Sure, its fundamental assumption is wrong
- Often, it works anyway
- On NLP tasks, NB is blazingly fast and surprisingly effective

(See "The Optimality of Naive Bayes", Harry Zhang, AAAI 2004)