GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

NATURAL LANGUAGE PROCESSING

AGENDA

Introduction to NLP

State of the art in NLP

Common Approaches to NLP problems

Topic Modeling

Intro to NLP

ABSTRACT QUESTIONS NLP LOOKS TO ANSWER

- What are common patterns that occur in language use?
- What kinds of things do people say/write?
- What do these utterances say/ask about the world?
- What is a ***grammatical*** utterance?

GRAMMATICAL OR NOT?

- John I believe Sally said Bill believed Sue saw
- What did Sally whisper that she had secretly read
- John wants very much for himself to win
- Those are the books you should read before it becomes too difficult to talk about
- Who did Joe think said John saw him
- The children read Mary’s stories about each other

GENERATIVE VS EMPIRICIST APPROACHES

Generative: seeks to describe language model of the mind for which real-world data (e.g. text) provide only indirect evidence

- “Poverty of the stimulus”
  - real-world language is full of errors
- “Colorless green ideas sleep furiously”
  - grammatical but meaningless

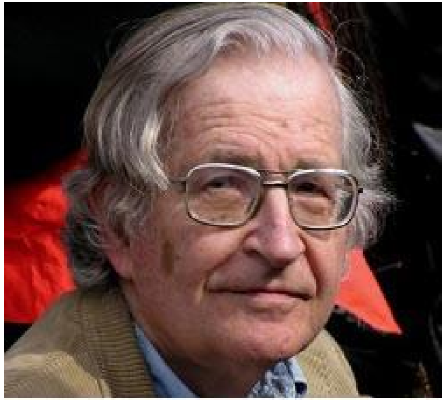
Generative vs Empiricist Approaches

Generative: seeks to describe language model of the mind for which real-world data (e.g. text) provide only indirect evidence

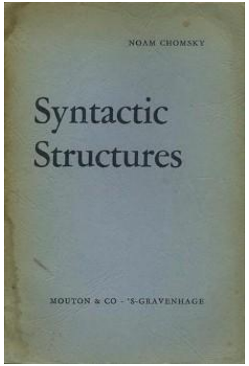
Empiricist: interested in describing language as it actually occurs (ignoring the underlying language models of the mind)

→ Statistical NLP models

GENERATIVE VS EMPIRICIST APPROACHES



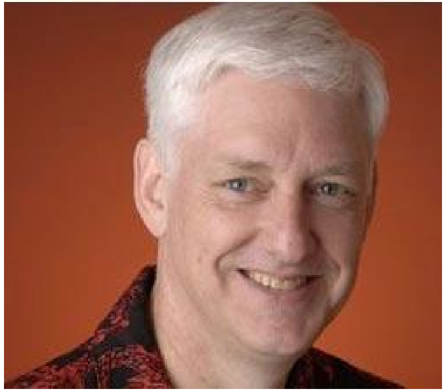
Noam Chomsky:



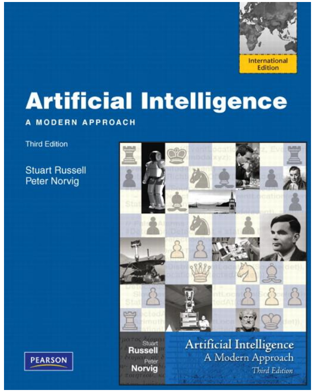
NOAM CHOMSKY

Syntactic Structures

MOUTON & CO · 'S-GRAVENHAGE

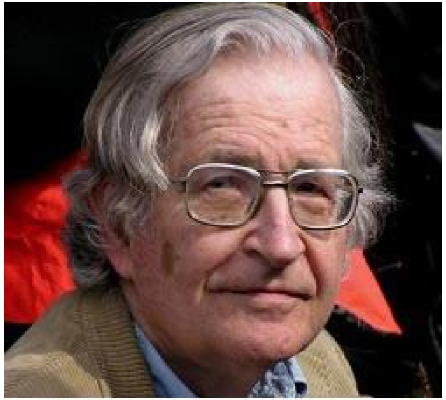


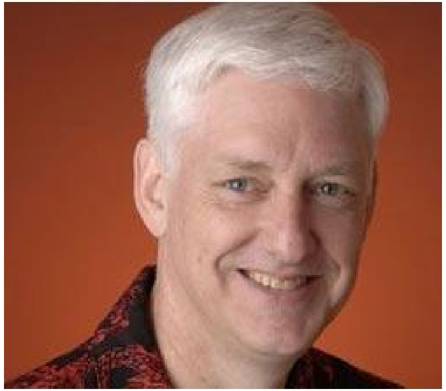
Peter Norvig:



source: On Chomsky and the Two Cultures of Statistical Learning

Generative vs Empiricist Approaches





Noam Chomsky: *[...] there's been a lot of work on trying to apply statistical models to various linguistic problems. I think there have been some successes, but a lot of failures. There is a notion of success ... which I think is novel in the history of science. It interprets success as approximating unanalyzed data*

Peter Norvig: *Science is a combination of gathering facts and making theories; neither can progress on its own. I think Chomsky is wrong to push the needle so far towards theory over facts; in the history of science, the laborious accumulation of facts is the dominant mode, not a novelty. The science of understanding language is no different than other sciences in this respect.*

source: [On Chomsky and the Two Cultures of Statistical Learning]

GENERATIVE VS EMPIRICIST APPROACHES

Chomsky has derided researchers in machine learning who use purely statistical methods to produce behavior that mimics something in the world, but who don’t try to understand the meaning of that behavior.

To Chomsky, building such models is like studying the dance made by a bee returning to the hive, and producing a statistically based simulation of such a dance --without attempting to understand why the bee behaved that way.

source: Norvig vs. Chomsky and the Fight for the Future of AI

CORPORA

Q: What is a Corpus?

CORPORA

Q: What is a Corpus?

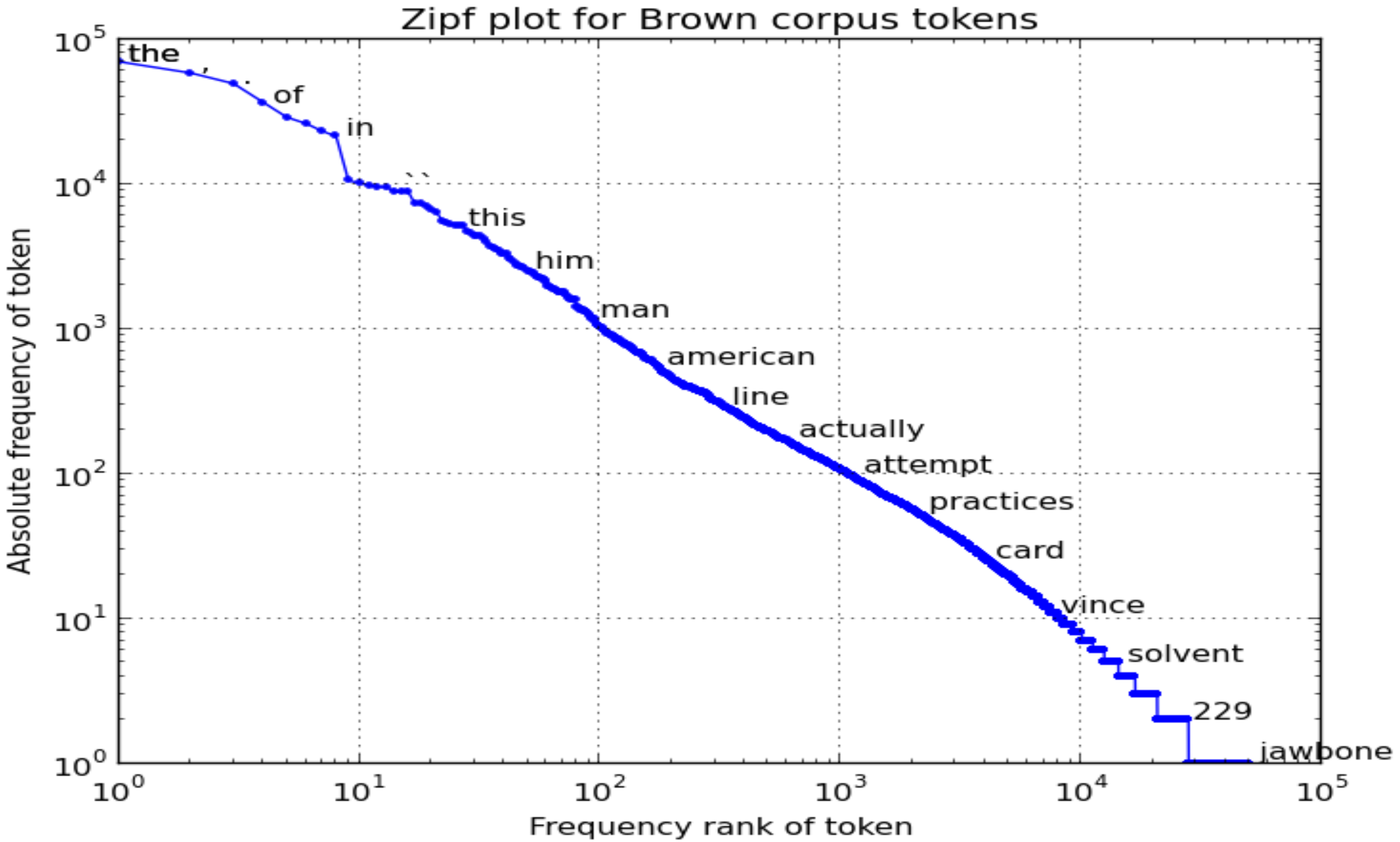
A: A large and structured set of texts

e.g. the ***Brown Corpus***, contains 500 samples of English-language text, totaling roughly one million words, compiled from works published in the United States in 1961

ZIPF'S LAW

$$f \propto 1 / r$$

- Given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table
- The most frequent word will occur approximately twice as often as the second most frequent word, three times as often as the third most frequent word, etc


Zipf plot for Brown corpus tokens
Absolute frequency of token
Frequency rank of token
10^5
10^4
10^3
10^2
10^1
10^0
10^0
10^1
10^2
10^3
10^4
10^5
the
,
.
of
in
``
this
him
man
american
line
actually
attempt
practices
card
vince
solvent
229
jawbone

STATE OF THE ART IN NLP

Why NLP is Hard

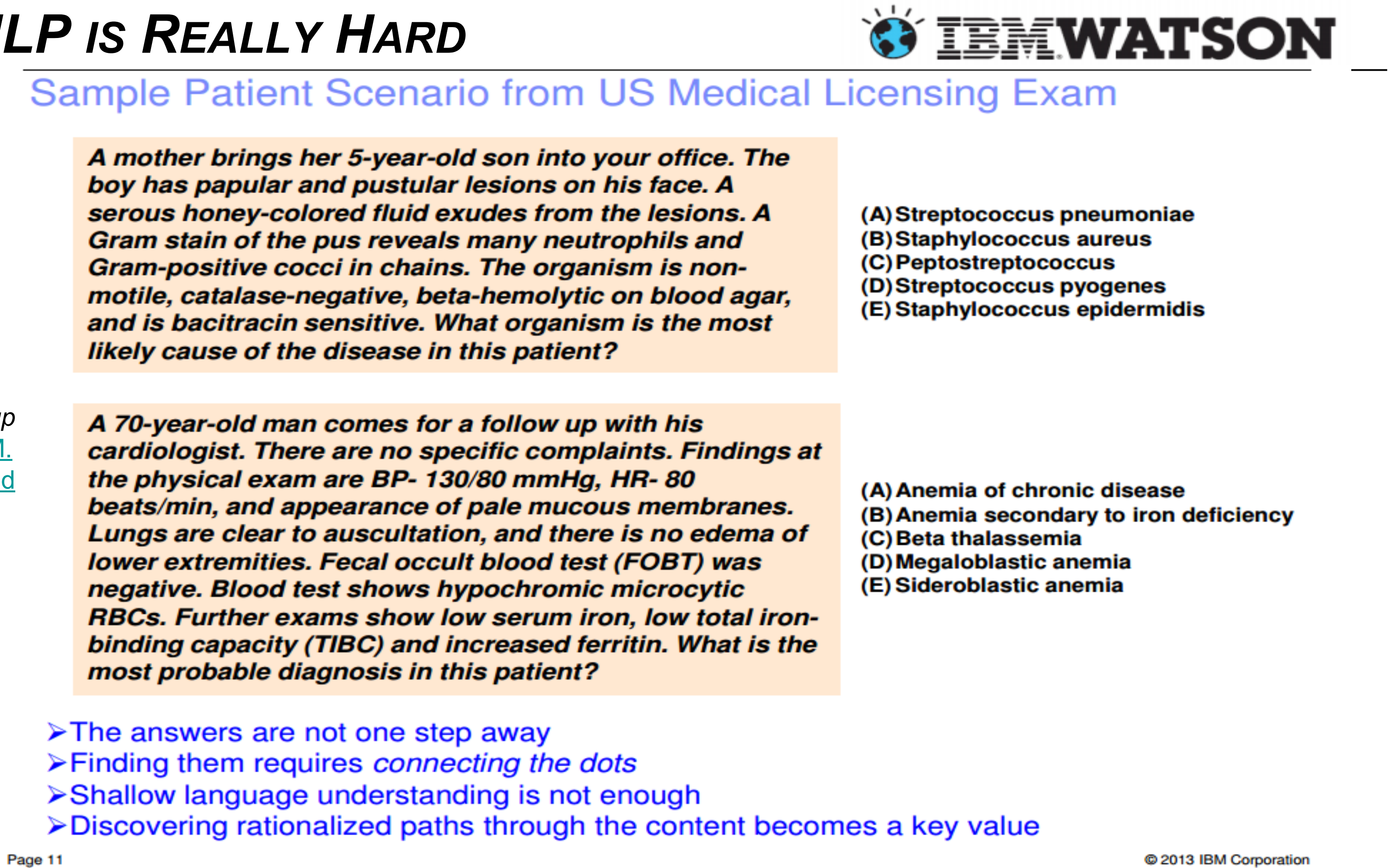
WHY NLP IS HARD

homographs—identically spelled words with multiple meanings:

- “the spirit is willing, but the flesh is weak”
- *translated to Russian and back*:
  “the vodka is agreeable, but the meat is spoiled”

WHY NLP IS REALLY HARD

source: *New York Knowledge Engineering Meetup* (08/04/2014): I.B.M. Watson present and future



Sample Patient Scenario from US Medical Licensing Exam

A mother brings her 5-year-old son into your office. The boy has papular and pustular lesions on his face. A serous honey-colored fluid exudes from the lesions. A Gram stain of the pus reveals many neutrophils and Gram-positive cocci in chains. The organism is non-motile, catalase-negative, beta-hemolytic on blood agar, and is bacitracin sensitive. What organism is the most likely cause of the disease in this patient?

(A) Streptococcus pneumoniae
(B) Staphylococcus aureus
(C) Peptostreptococcus
(D) Streptococcus pyogenes
(E) Staphylococcus epidermidis

A 70-year-old man comes for a follow up with his cardiologist. There are no specific complaints. Findings at the physical exam are BP- 130/80 mmHg, HR- 80 beats/min, and appearance of pale mucous membranes. Lungs are clear to auscultation, and there is no edema of lower extremities. Fecal occult blood test (FOBT) was negative. Blood test shows hypochromic microcytic RBCs. Further exams show low serum iron, low total iron-binding capacity (TIBC) and increased ferritin. What is the most probable diagnosis in this patient?

(A) Anemia of chronic disease
(B) Anemia secondary to iron deficiency
(C) Beta thalassemia
(D) Megaloblastic anemia
(E) Sideroblastic anemia

- ➢The answers are not one step away
- ➢Finding them requires *connecting the dots*
- ➢Shallow language understanding is not enough
- ➢Discovering rationalized paths through the content becomes a key value

NLP Tasks

Low Level	High Level
Lexical parsing Morphological (word) segmentation Optical character recognition (OCR) Part-of-speech (POS) tagging Sentence boundary disambiguation Speech/phoneme segmentation	Automatic summarization Discourse analysis Machine translation Named entity recognition (NER) Natural language generation Natural language understanding Sentiment analysis Speech recognition Topic segmentation and recognition Word sense disambiguation

Python NLP Packages

	Pros	Cons
NLTK: Natural Language Toolkit	- Well-documented - Active dev community - Lots of features	- Unsuitable for high-performance applications
Gensim: Topic Modeling for Humans	- Built-in distributed computing support - Can index datasets larger than RAM - Great docs, tutorials	- Narrow application focus - Smaller support community (than NLTK)
Sklearn	- Part of familiar set of tools for Machine Learning - Good lib to explore small datasets	- Limited set of NLP features - “Blows up with memory errors much sooner than other libs”
corenlp: Wrapper for Stanford Core NLP	Stanford Core NLP ... - Written in Java - Gold standard for serious NLP work	- Python wrapper around a package written in Java - Relatively little support

COMMON APPROACHES

COMMON TECHNIQUES WORKING WITH TEXT

0. Lowercase, remove punctuation
1. Word/Sentence segmentation
2. Stemming/ Lemmatization: normalize word forms
3. Filtering stop-words
4. Term-frequency Inverse Document Frequency (TF-IDF)
5. Vectorizing the document (n-grams)

Common Techniques Working With Text

0. Lowercase, remove punctuation
1. **Word/Sentence segmentation**
2. Stemming/ Lemmatization: normalize word forms
3. Filtering stop-words
4. Term-frequency Inverse Document Frequency (TF-IDF)
5. Vectorizing the document (n-grams)

WORD-SENTENCE SEGMENTATION

The standard approaches to locate the end of a sentence:

- If it's a period, it ends a sentence
- If the preceding token is in the hand-compiled list of abbreviations, then it doesn't end a sentence
- If the next token is capitalized, then it ends a sentence

This strategy gets about 95% of sentences correct.

source: Wikipedia: Sentence boundary disambiguation

COMMON TECHNIQUES WORKING WITH TEXT

0. Lowercase, remove punctuation
1. Word/Sentence segmentation
2. **Stemming/ Lemmatization: normalize word forms**
3. Filtering stop-words
4. Term-frequency Inverse Document Frequency (TF-IDF)
5. Vectorizing the document (n-grams)

STEMMING/ LEMMATIZATION

- LinkedIn sees 6,000+ variations on the job title, “Software Engineer”
- They see 8,000+ variations on the company, “IBM”

They have to recognize all of these and understand they are the same

STEMMING/ LEMMATIZATION

On a smaller scale, it is often useful to strip away conjugations and other modifiers:

science, scientist → ***scien***

swim, swimming, swimmer → ***swim***

The resulting text is often unreadable, but retains semantic content

Common Techniques Working With Text

0. Lowercase, remove punctuation
1. Word/Sentence segmentation
2. Stemming/ Lemmatization: normalize word forms
3. **Filtering stop-words**
4. Term-frequency Inverse Document Frequency (TF-IDF)
5. Vectorizing the document (n-grams)

FILTERING STOP-WORDS

“Certain things have come to light. And, you know, has it ever occurred to you, that, instead of, uh, you know, running around, uh, uh, blaming me, you know, given the nature of all this new sh*t, you know, I-I-I-I... this could be a-a-a-a lot more, uh, uh, uh, uh, uh, uh, complex, I mean, it's not just, it might not be just such a simple... uh, you know?”

-***The Dude, The Big Lebowski***

FILTERING STOP-WORDS

- Some words are so common that they provide little useful information to a statistical language model
- Different languages have different stop words
- Any group of words can be chosen as the stop words for a given purpose

Filtering stop-words

Q: How would you go about identifying and removing stop words from a corpus?

Filtering stop-words

Q: How would you go about identifying and removing stop words from a corpus?

A: Two approaches:

1. Look up in a list
2. Define terms with the largest ***document frequency*** as stopwords. The threshold above which you call something a stopword is tunable

Common Techniques Working With Text

0. Lowercase, remove punctuation
1. Word/Sentence segmentation
2. Stemming/ Lemmatization: normalize word forms
3. Filtering stop-words
4. **Term-frequency Inverse Document Frequency (TF-IDF)**
5. Vectorizing the document (n-grams)

TF-IDF

Term Frequency:

$$tf(t,d) = N_{term\ occurences\ in\ doc}$$

Document Frequency:

$$df(t, D) = N_{documents\ containing\ term}/N_{documents}$$

t = single term

d = single document

D = all documents

TF-IDF

Term Frequency:

$$tf(t,d) = N_{\text{term occurences in doc}}$$

Inverse Document Frequency:

$$idf(t, D) = \log(N_{\text{documents}}/N_{\text{documents containing term}})$$

t = single term

d = single document

D = all documents

TF-IDF

Inverse document frequency is a measure of how much information a word provides, i.e., whether a term is common or rare across all documents

$$\boldsymbol{tfidf(t, d, D) = (N_{term}/N_{terms\ in\ document}) * log(N_{documents}/N_{documents\ containing\ term})}$$

TF-IDF is larger for words that occur more frequently in a document, but occur in fewer documents overall

TF-IDF

TF-IDF example:

doc1	
term	**count**
this	1
is	1
a	3
pen	2

doc2	
term	**count**
this	2
example	2
has	1
pen	2

doc3	
term	**count**
a	3
pen	4
example	1
shines	2

TF-IDF

TF-IDF example:

doc1	
term	**count**
this	1
is	1
a	3
pen	2

doc2	
term	**count**
this	2
example	2
has	1
pen	2

doc3	
term	**count**
a	3
pen	4
example	1
shines	2

tfidf(“this”, doc1, D) =

TF-IDF

TF-IDF example:

doc1	
term	**count**
this	1
is	1
a	3
pen	2

doc2	
term	**count**
this	2
example	2
has	1
pen	2

doc3	
term	**count**
a	3
pen	4
example	1
shines	2

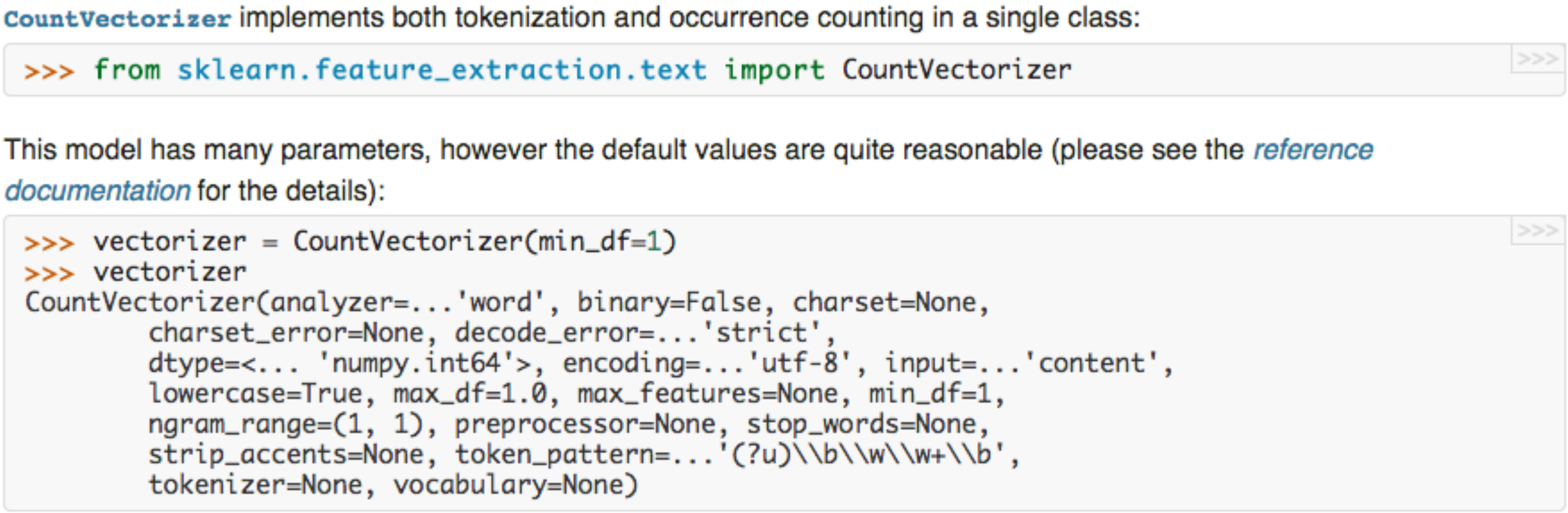
tfidf(“shines”, doc3, D) =

COMMON TECHNIQUES WORKING WITH TEXT

0. Lowercase, remove punctuation
1. Word/Sentence segmentation
2. Stemming/ Lemmatization: normalize word forms
3. Filtering stop-words
4. Term-frequency Inverse Document Frequency (TF-IDF)
5. **Vectorizing the document (n-grams)**

VECTORIZING

Process of turning a collection of text documents into numerical feature vectors



`CountVectorizer` implements both tokenization and occurrence counting in a single class:

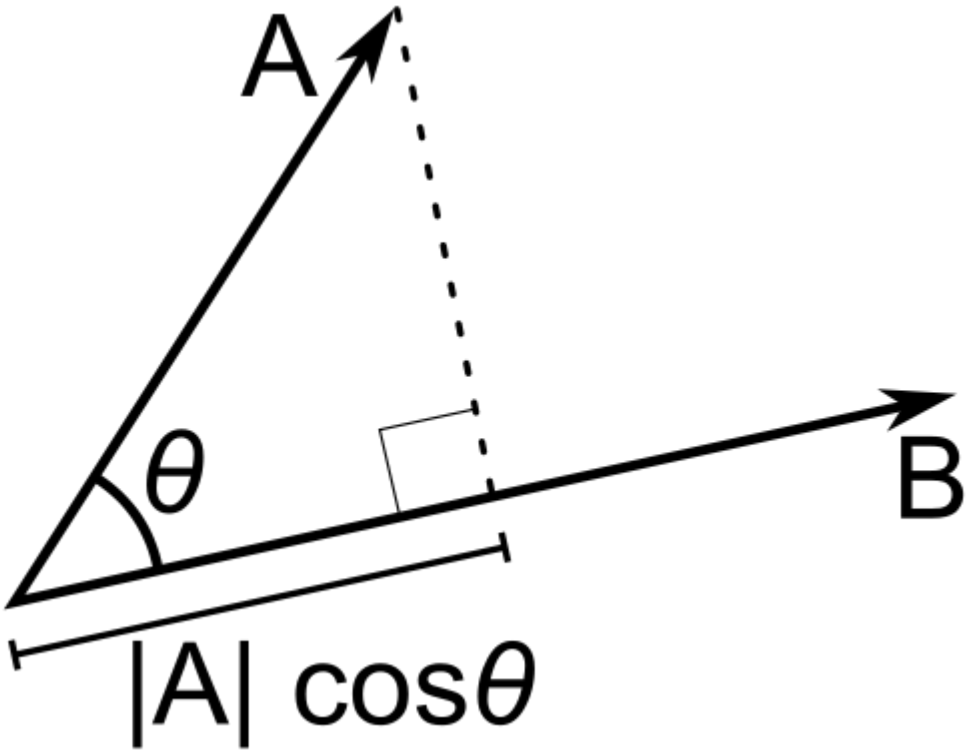
```
>>> from sklearn.feature_extraction.text import CountVectorizer
```

This model has many parameters, however the default values are quite reasonable (please see the *reference documentation* for the details):

```
>>> vectorizer = CountVectorizer(min_df=1)
>>> vectorizer
CountVectorizer(analyzer=...'word', binary=False, charset=None,
        charset_error=None, decode_error=...'strict',
        dtype=<... 'numpy.int64'>, encoding=...'utf-8', input=...'content',
        lowercase=True, max_df=1.0, max_features=None, min_df=1,
        ngram_range=(1, 1), preprocessor=None, stop_words=None,
        strip_accents=None, token_pattern=...'(?u)\\b\\w\\w+\\b',
        tokenizer=None, vocabulary=None)
```

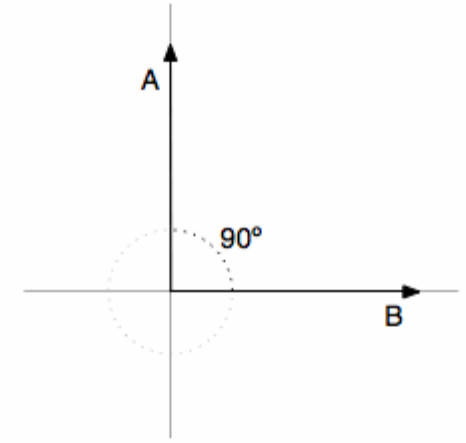
DOCUMENT SIMILARITY

COSINE SIMILARITY

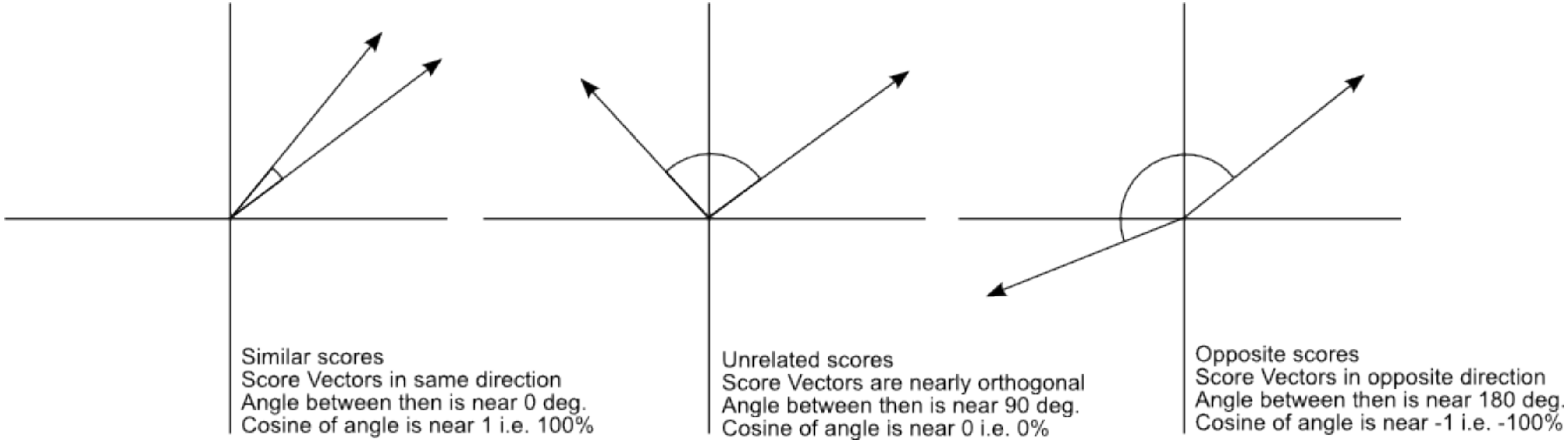


COSINE SIMILARITY

Q: What is the cosine of ninety degrees?



COSINE SIMILARITY

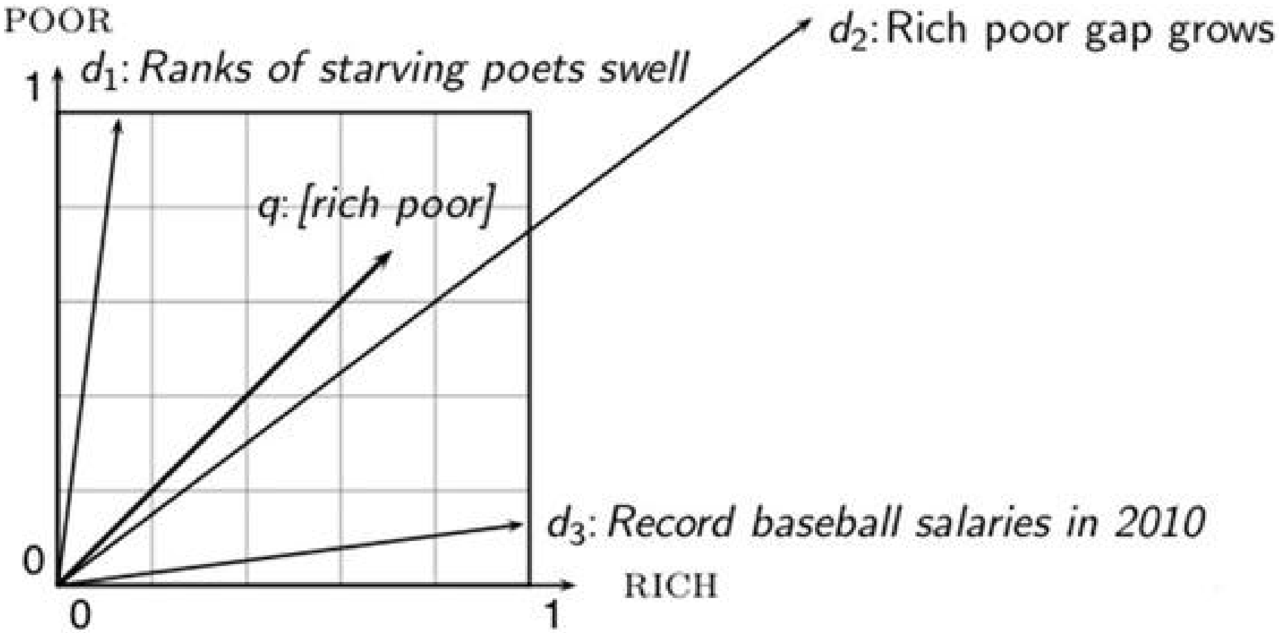


Similar scores
Score Vectors in same direction
Angle between then is near 0 deg.
Cosine of angle is near 1 i.e. 100%

Unrelated scores
Score Vectors are nearly orthogonal
Angle between then is near 90 deg.
Cosine of angle is near 0 i.e. 0%

Opposite scores
Score Vectors in opposite direction
Angle between then is near 180 deg.
Cosine of angle is near -1 i.e. -100%

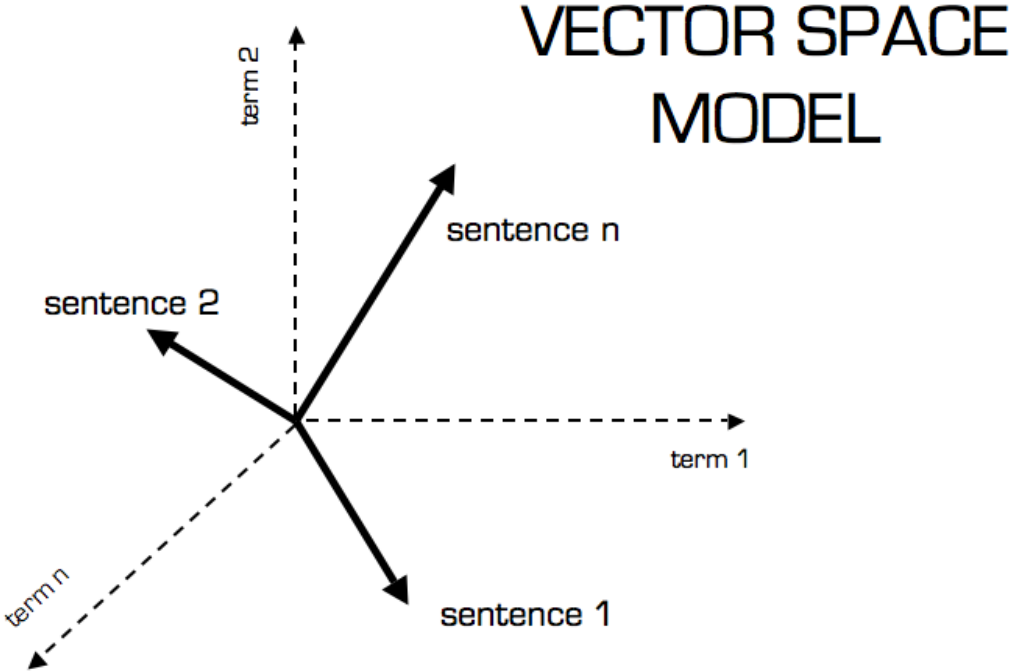
COSINE SIMILARITY



POOR
d_1: Ranks of starving poets swell
d_2: Rich poor gap grows
q: [rich poor]
d_3: Record baseball salaries in 2010
1
0
0
1
RICH

VECTOR SPACE MODEL

Modeled as vectors (with TF-IDF weights)



INTRO TO DATA SCIENCE

LAB: PRACTICE

TOPIC MODELING

TOPIC MODELING

- Topic Modeling is statistical modeling technique to identify themes, or topics, in a set of documents
- Intuitively, different topics generate words at particular frequencies, so you can work backwards from the words in a document to the topics
- Useful for news aggregators, segmenting a corpus

TOPIC MODELING: LDA

The most common implementation of a topic model is ***Latent Dirichlet Allocation (LDA)***

In LDA,

- each document may be viewed as a mixture of various topics
- the topic distribution is assumed to have a ***Dirichlet*** distribution

LDA: *Dirichlet distribution*

The Dirichlet distribution:

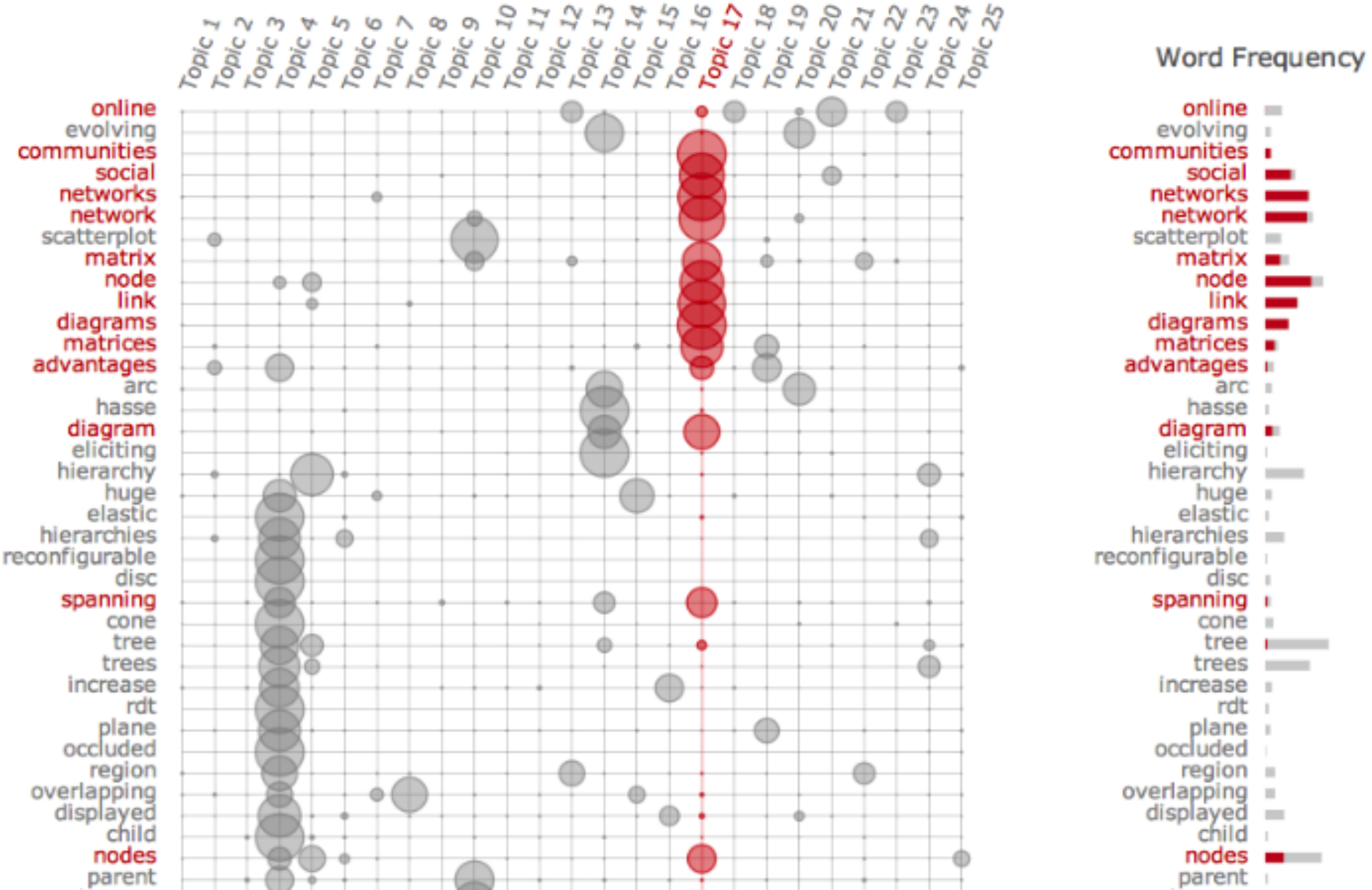
- a family of continuous multivariate probability distributions parameterized by a vector □ of positive reals
- gives the probability of choosing a given collection of ***m*** items from a set of ***n*** items with probabilities of each choice given by $\boldsymbol{p_1, \ldots, p_n}$

Latent Dirichlet Allocation (LDA) example

- An LDA model might have topics that can be classified as ***CAT_related*** and ***DOG_related***
- The CAT_related topic has a high probability of generating various words like ***milk, meow, and kitten***
- The DOG_related topic likewise has a high probability of generating words like ***puppy, bark, and bone***
- Words without special relevance (e.g. stop words), will have roughly even probability among topics.

Latent Dirichlet Allocation (LDA) visualization

source: Termite: Visualization Techniques for Assessing Textual Topic Models



Latent Dirichlet Allocation (LDA) example

Stanford Dissertation Browser

Latent Dirichlet Allocation (LDA) example

(1) `Fruits and vegetables are healthy.`

(2) I `like apples, oranges, and avocados. I don’t like the flu or colds.`

Let’s remove stop words, giving:

(1) `fruits vegetables healthy`

(2) `apples oranges avocados flu colds`

source: These Are Your Tweets on LDA, Part I

Latent Dirichlet Allocation (LDA) example

Topic 1 = Fruits, Vegetables, Apples, Oranges, Avocados

Topic 2 = Healthy, Flu, Colds

And:

doc1 1 = (2/3) Topic 1, (1/3) Topic 2

doc 2 = (3/5) Topic 1, (2/5) Topic 2

source: These Are Your Tweets on LDA, Part I

Latent Dirichlet Allocation (LDA) example

Each topic in LDA is a probability distribution over the words. In our case, LDA would give ***k* = 2** distributions of size ***V* = 8**. Each item of the distribution corresponds to a word in the vocabulary. For instance, let’s call one of these distributions $\square_1$. It might look something like:

$$\square_1 = [0.4, 0.2, 0.15, 0.05, 0.05, 0.05, 0.05]$$

$\square_1$ lets us answer questions like: given that our topic is Topic1 (‘Food’), what is the probability of generating word1 (‘Fruits’)?