Introduction to Data Science
Lecture 12
Natural Language Processing

EMSE6992

Benjamin Harvey

Outline for this Evening

- Project Presentation Schedule
- N-grams
- Grammars
- Parsing
- Dependencies

Project Presentations

Last_Name	First_Name	ID	Time
Akinbule	Olatunji	akinbule_ola	6:00 - 6:12
Alshamrani	Amirah	amirah_shumrani	6:12 - 6:24
Han	Xiao	hanxiao	6:24 - 6:36
He	Yanjie	heyanjie	6:36 - 6:48
Pei	Lihua	lihuapei	6:48 - 7:00
Ren	Zhijie	zren88	7:00 - 7:12
Stumpf	Jonathan	jcstumpf	7:12 - 7:24
Tarar	Sadiqa	sadiqatarar	7:24 - 7:36
Tu	Ping	ptu16	7:36 - 7:48
Turano	Madison	madly9	7:48 - 8:00
Wang	Peiyu	oliviawpy	8:00 - 8:12
Warndorf	Maddie	mwarndorf65	8:12 - 8:24
Yu	Haotian	yuxx6789	8:24 - 8:36
Zhao	Lujin	lujin	8:36 - 8:48
Zhu	Ziqiu	zzhu158	8:48 - 9:00

Reminders

- Assignment 3 Due 11/29
- Assignment 4 Due 12/7
- We will have project and portfolio presentations on 12/6
  - Go over rubrics

Introduction to Text Analytics

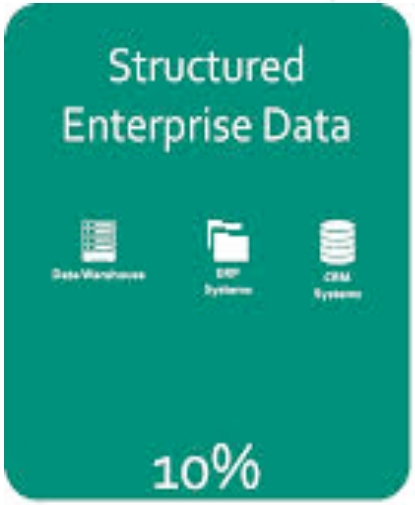
- Text analytics, also known as text mining, commonly refers to the techniques of mining text for information retrieval (semantic search), classification of documents according to known categories, and finding groups of similar documents.
- Another application of text analytics is the derivation of additional features from unstructured data. For example, contact or financial information can be extracted from text and can further be used as additional record attributes.
- Text analytics is also used preform sentiment analysis, which is also known as opinion mining.
- Mining techniques that apply to structured data are not directly applicable to textual data, as textual data requires pre-processing for representation as a feature vector.

Definition: Text Analytics

- The analysis of structured and unstructured text to obtain and organize information:
  - The extraction of information from structured and unstructured text.
  - The resolution of entities
  - Co-reference analysis of pronouns and nouns
  - Mapping of information to ontological structures
  - Synthesize the content in a knowledge structure
  - Understand the content to generate actionable intelligence
- A loosely organized set of competing technologies
- One unifying theme behind these technologies is the concept of "turning text into numbers" so that quantitative methods can be applied to extract information.
- *But, that is not all there is to text analytics!*

Text Mining

- Text Mining is a subset of Text Analytics that is used for:
  - Content and document management
  - Information search and retrieval
  - Processing XML/RDF databases
  - Categorization, classification, and visualizatio
- As a reminder, it has been estimated that 90+% of the information in an organization is represented as unstructured text.



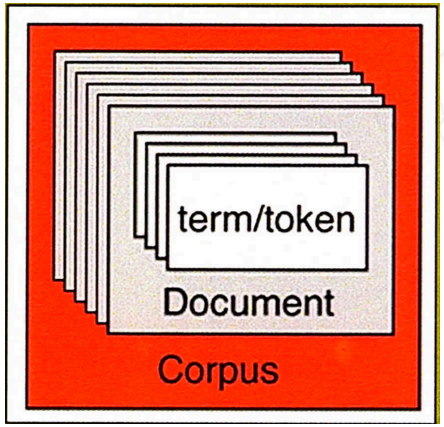
Text Representation

- A key activity in the pre-processing of textual data, prior to applying text mining algorithms is text representation.
- This process involves parsing textual data into its constituent elements and generating certain measures from these elements.
- There are multiple techniques for text representation, such as bag of words, n-grams, and name entity extraction.

Text Representation

Key concepts in text representation:

- A token/term is a single word that can be a noun, verb, pronoun, preposition, adverb, adjective, or article.
- A document is textual data of arbitrary length that can vary between a single line to multiple pages comprising multiple tokens/terms.
- A single instance or record within the dataset is often equivalent to a document that contains multiple tokens/terms.
- A corpus is a collection of multiple documents.



Basic Idea - I

- Text is meant to be read by humans, not machines.
- Much useful information is stored as text
  - Think of all the libraries and their books and papers
  - 100's of times more on-line text than on-line databases
  - Web pages are text files with structure tags (HTML)
- General Natural Language Understanding is still too hard
  - But, General Natural Language Processing has had some successes
- Key Questions:
  - What data can we extract from text to describe, diagnose, predict or prescribe?
  - What transformations should we apply to extracted data to organize it?

Seven Practice Areas

1. *Search and Information Retrieval*: Storage and retrieval of text, including keyword search, phrase search, etc.
2. *Document Clustering*: Group and categorizing terms, phrases, paragraphs, documents using different techniques, including data mining
3. *Document Classification*: classification of documents and subcomponents using different techniques, including data mining
4. *Web Mining*: Applying (2) and (3) to text on the Web with an emphasis on interconnectedness
5. *Information Extraction*: Identification and extraction of relevant information from unstructured text
6. *Natural Language Processing*: language processing and understanding, including semantics
7. *Concept Extraction*: grouping of words and phrases into semantically similar groups to summarize content

Natural Language Processing

Some basic terms:

- Syntax: the allowable structures in the language: sentences, phrases, affixes (-ing, -ed, -ment, etc.).
- Semantics: the meaning(s) of texts in the language.
- Part-of-Speech (POS): the category of a word (noun, verb, preposition etc.).
- Bag-of-words (BoW): a featurization that uses a vector of word counts (or binary) ignoring order.
- N-gram: for a fixed, small N (2-5 is common), an n-gram is a consecutive sequence of words in a text.

Bag of words Featurization

Assuming we have a dictionary mapping words to a unique integer id, a bag-of-words featurization of a sentence could look like this:

Sentence:	The	cat	sat	on	the	mat
word id's:	1	12	5	3	1	14

The BoW featurization would be the vector:

Vector	2,	0,	1,	0,	1,	0,	0,	0,	0,	0,	0,	1,	0,	1
position	1		3		5							12		14

In practice this would be stored as a sparse vector of (id, count)s:

(1,2),(3,1),(5,1),(12,1),(14,1)

Note that the original word order is lost, replaced by the order of id's.

N-grams

Because word order is lost, the sentence meaning is weakened. This sentence has quite a different meaning but the same BoW vector:

Sentence:	The	mat	sat	on	the	cat
word id s:	1	14	5	3	1	12

BoW featurization:

Vector 2, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1

But word order is important, especially the order of nearby words.

N-grams capture this, by modeling tuples of consecutive words.

N-grams

Sentence: The cat sat on the mat

2-grams: the-cat, cat-sat, sat-on, on-the, the-mat

Notice how even these short n-grams "make sense" as linguistic units. For the other sentence we would have different features:

Sentence: The mat sat on the cat

2-grams: the-mat, mat-sat, sat-on, on-the, the-cat

We can go still further and construct 3-grams:

Sentence: The cat sat on the mat

3-grams: the-cat-sat, cat-sat-on, sat-on-the, on-the-mat

Which capture still more of the meaning:

Sentence: The mat sat on the cat

3-grams: the-mat-sat, mat-sat-on, sat-on-the, on-the-cat

N-grams Features

Typically, its advantages to use multiple n-gram features in machine learning models with text, e.g.

unigrams + bigrams (2-grams) + trigrams (3-grams).

The unigrams have higher counts and are able to detect influences that are weak, while bigrams and trigrams capture strong influences that are more specific.

e.g. “the white house” will generally have very different influences from the sum of influences of “the”, “white”, “house”.

N-grams size

N-grams pose some challenges in feature set size.

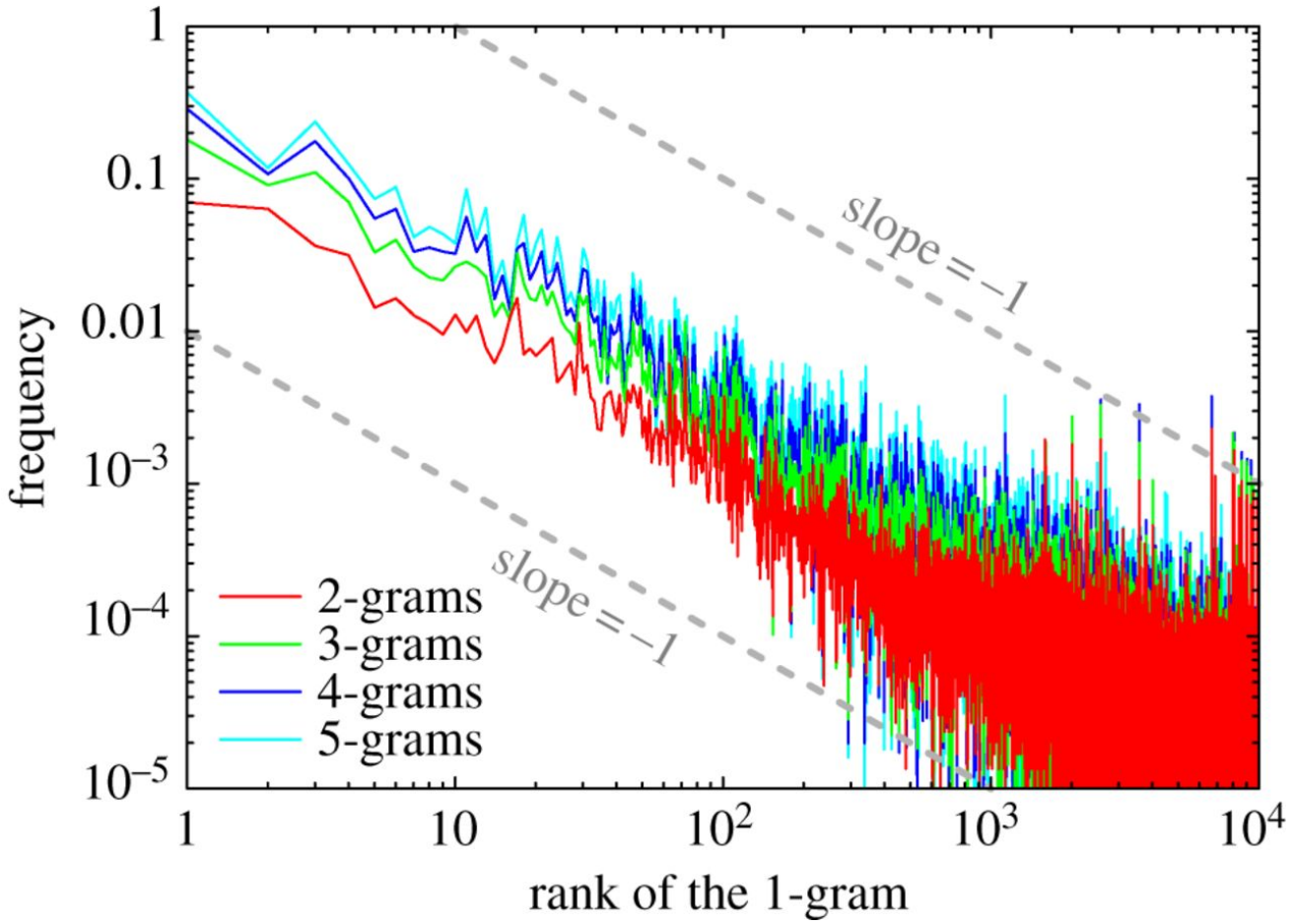
If the original vocabulary size is $|V|$, the number of 2-grams is $|V|^2$

While for 3-grams it is $|V|^3$

Luckily natural language n-grams (including single words) have a **power law** frequency structure. This means that most of the n-grams you see are common. A dictionary that contains the most common n-grams will cover most of the n-grams you see.

Power laws for N-grams

N-grams follow a power law distribution:



N-grams size

Because of this you may see values like this:

- Unigram dictionary size: 40,000
- Bigram dictionary size: 100,000
- Trigram dictionary size: 300,000

With coverage of > 80% of the features occurring in the text.

N-gram Language Models

N-grams can be used to build statistical models of texts.

When this is done, they are called n-gram language models.

An n-gram language model associates a probability with each n-gram, such that the sum over all n-grams (for fixed n) is 1.

You can then determine the overall likelihood of a particular sentence:

The cat sat on the mat

Is much more likely than

The mat sat on the cat

Skip-grams

We can also analyze the meaning of a particular word by looking at the contexts in which it occurs.

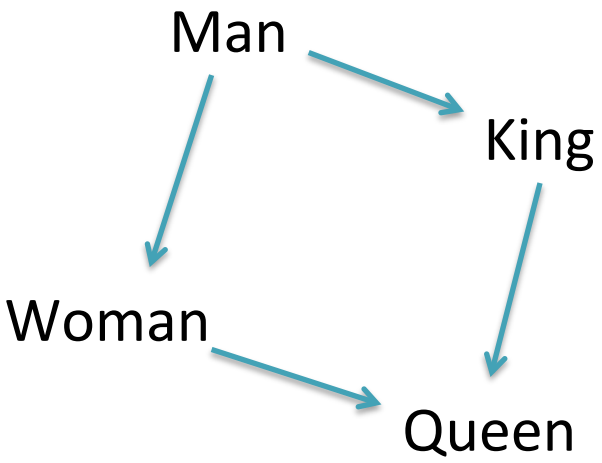
The context is the set of words that occur near the word, i.e. at displacements of …,-3,-2,-1,+1,+2,+3,… in each sentence where the word occurs.

A skip-gram is a set of non-consecutive words (with specified offset), that occur in some sentence.

We can construct a BoSG (bag of skip-gram) representation for each word from the skip-gram table.

Skip-grams

Then with a suitable embedding (DNN or linear projection) of the skip-gram features, we find that word meaning has an algebraic structure:



Man + (King – Man) + (Woman – Man) = Queen

Tomáš Mikolov et al. (2013).
"Efficient Estimation of Word Representations in Vector Space"

5-min break

Outline for this Evening

- Project Presentation Schedule
- N-grams
- Grammars
- Parsing
- Dependencies

Parts of Speech

Thrax's original list (c. 100 B.C):

- Noun
- Verb
- Pronoun
- Preposition
- Adverb
- Conjunction
- Participle
- Article

Parts of Speech

Thrax's original list (c. 100 B.C):

- Noun (boat, plane, Obama)
- Verb (goes, spun, hunted)
- Pronoun (She, Her)
- Preposition (in, on)
- Adverb (quietly, then)
- Conjunction (and, but)
- Participle (eaten, running)
- Article (the, a)

Parts of Speech (Penn Treebank 2014)

1.	CC	Coordinating conjunction
2.	CD	Cardinal number
3.	DT	Determiner
4.	EX	Existential *there*
5.	FW	Foreign word
6.	IN	Preposition or subordinating conjunction
7.	JJ	Adjective
8.	JJR	Adjective, comparative
9.	JJS	Adjective, superlative
10.	LS	List item marker
11.	MD	Modal
12.	NN	Noun, singular or mass
13.	NNS	Noun, plural
14.	NNP	Proper noun, singular
15.	NNPS	Proper noun, plural
16.	PDT	Predeterminer
17.	POS	Possessive ending
18.	PRP	Personal pronoun
19.	PRP$	Possessive pronoun
20.	RB	Adverb
21.	RBR	Adverb, comparative
22.	RBS	Adverb, superlative
23.	RP	Particle
24.	SYM	Symbol
25.	TO	*to*
26.	UH	Interjection
27.	VB	Verb, base form
28.	VBD	Verb, past tense
29.	VBG	Verb, gerund or present participle
30.	VBN	Verb, past participle
31.	VBP	Verb, non-3rd person singular present
32.	VBZ	Verb, 3rd person singular present
33.	WDT	Wh-determiner
34.	WP	Wh-pronoun
35.	WP$	Possessive wh-pronoun
36.	WRB	Wh-adverb

Grammars

Grammars comprise rules that specify acceptable sentences in the language: (S is the sentence or root node)

- S → NP VP
- S → NP VP PP
- NP → DT NN
- VP → VB NP
- VP → VBD
- PP → IN NP
- DT → “the”
- NN → “mat”, “cat”
- VBD → “sat”
- IN → “on”

Grammars

Grammars comprise rules that specify acceptable sentences in the language: (S is the sentence or root node) "the cat sat on the mat"

- S → NP VP
- S → NP VP PP (the cat) (sat) (on the mat)
- NP → DT NN (the cat), (the mat)
- VP → VB NP
- VP → VBD
- PP → IN NP
- DT → "the"
- NN → "mat", "cat"
- VBD → "sat"
- IN → "on"

Grammars

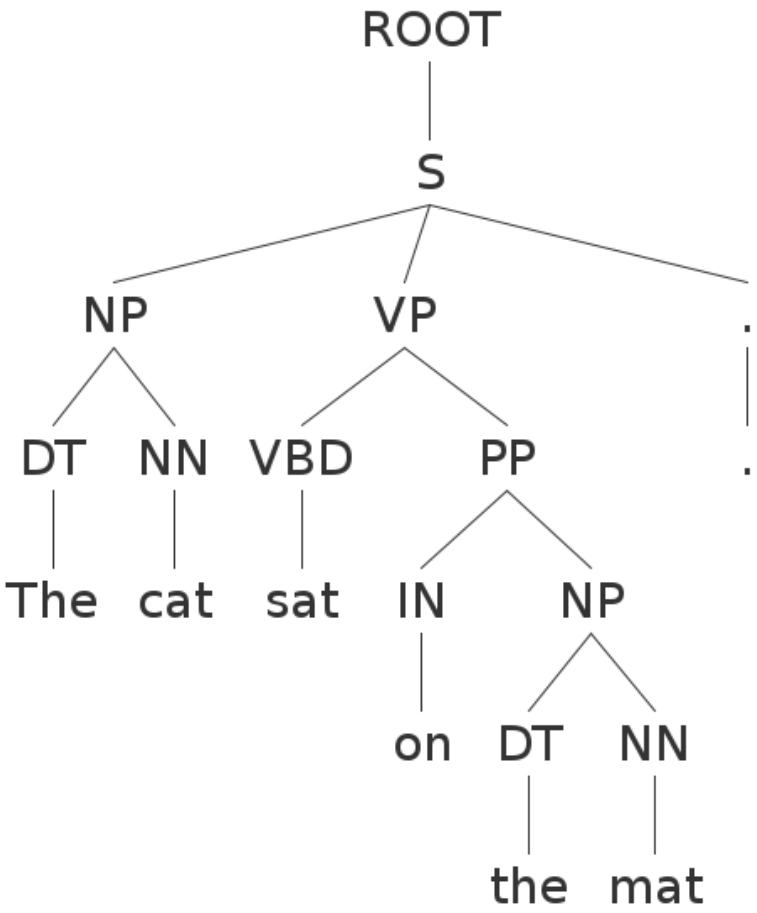
English Grammars are context-free: the productions do not depend on any words before or after the production.

The reconstruction of a sequence of grammar productions from a sentence is called “parsing” the sentence.

It is most conveniently represented as a tree:

Parse Trees

"The cat sat on the mat"



Parse Trees

In bracket notation:

```
(ROOT
  (S
    (NP (DT the) (NN cat))
    (VP (VBD sat)
      (PP (IN on)
        (NP (DT the) (NN mat))))))
```

Grammars

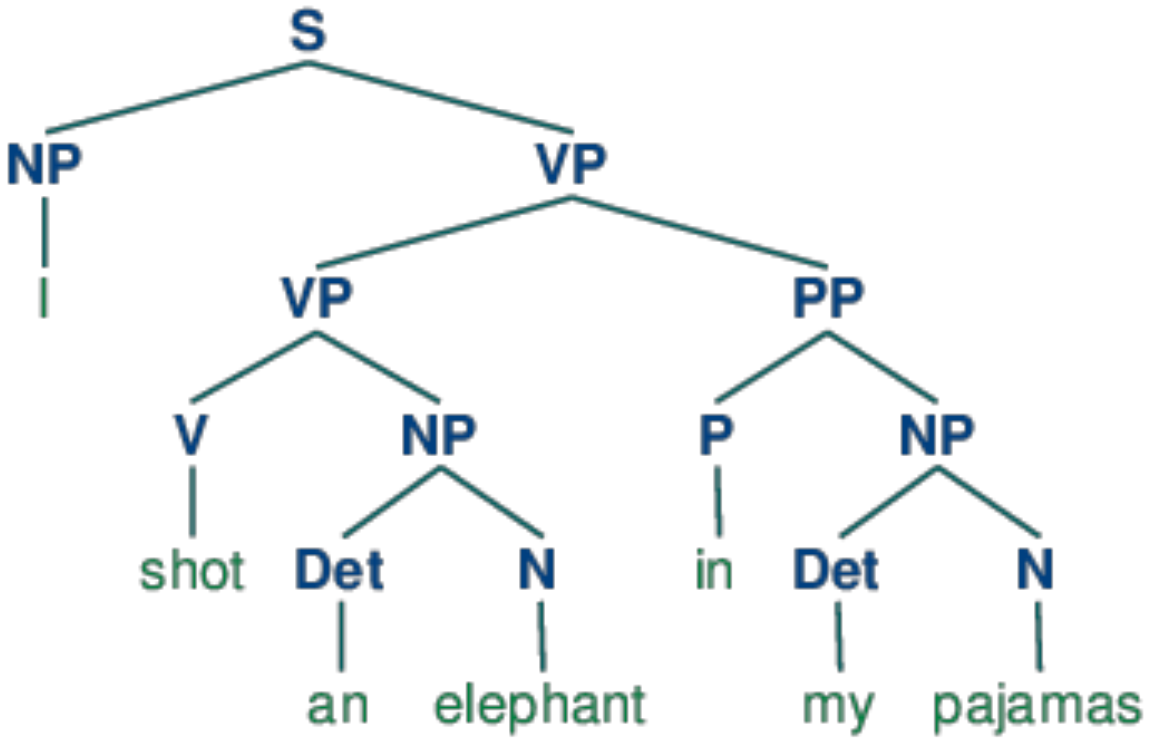
There are typically multiple ways to produce the same sentence. Consider the statement by Groucho Marx:

"While I was in Africa, I shot an elephant in my pajamas"

"How he got into my pajamas, I don't know"

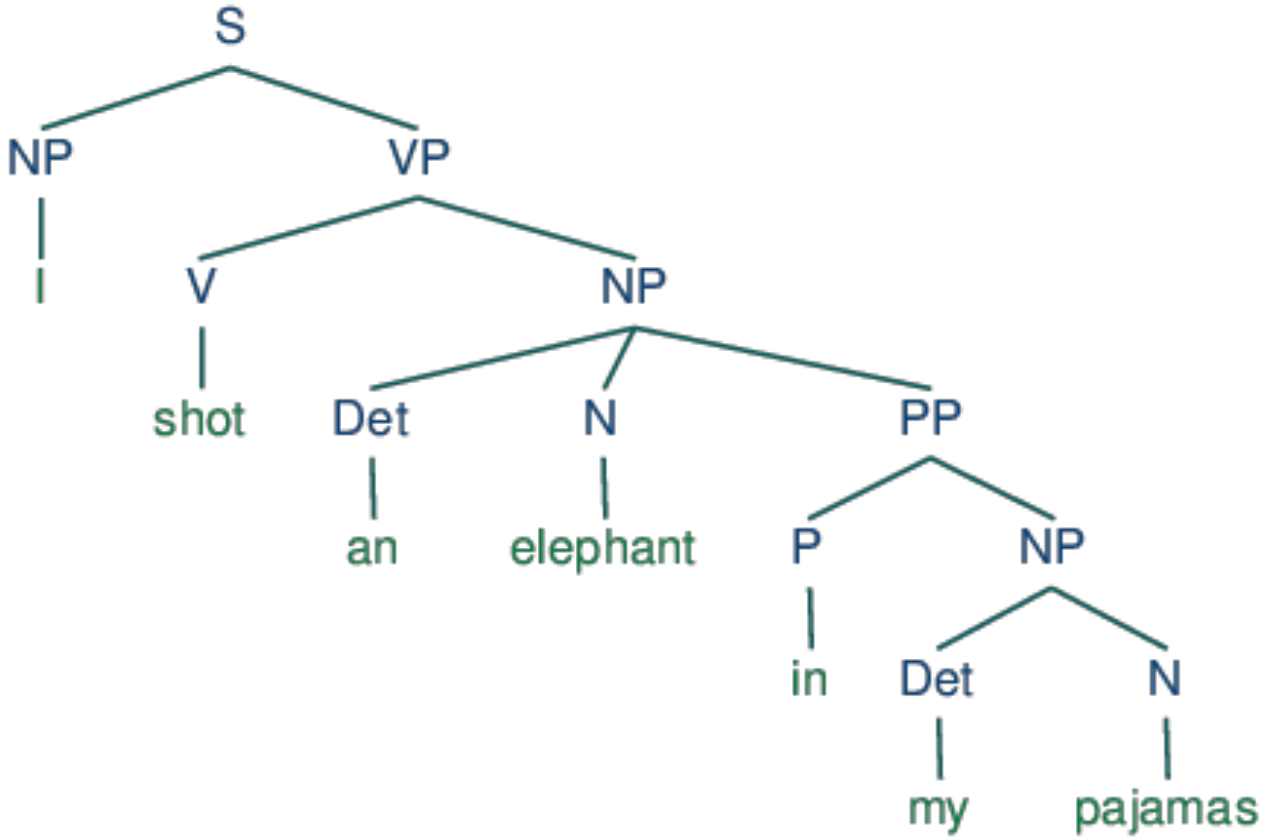
Parse Trees

"…,I shot an elephant in my pajamas" -what people hear first



Parse Trees

Groucho's version



Grammars

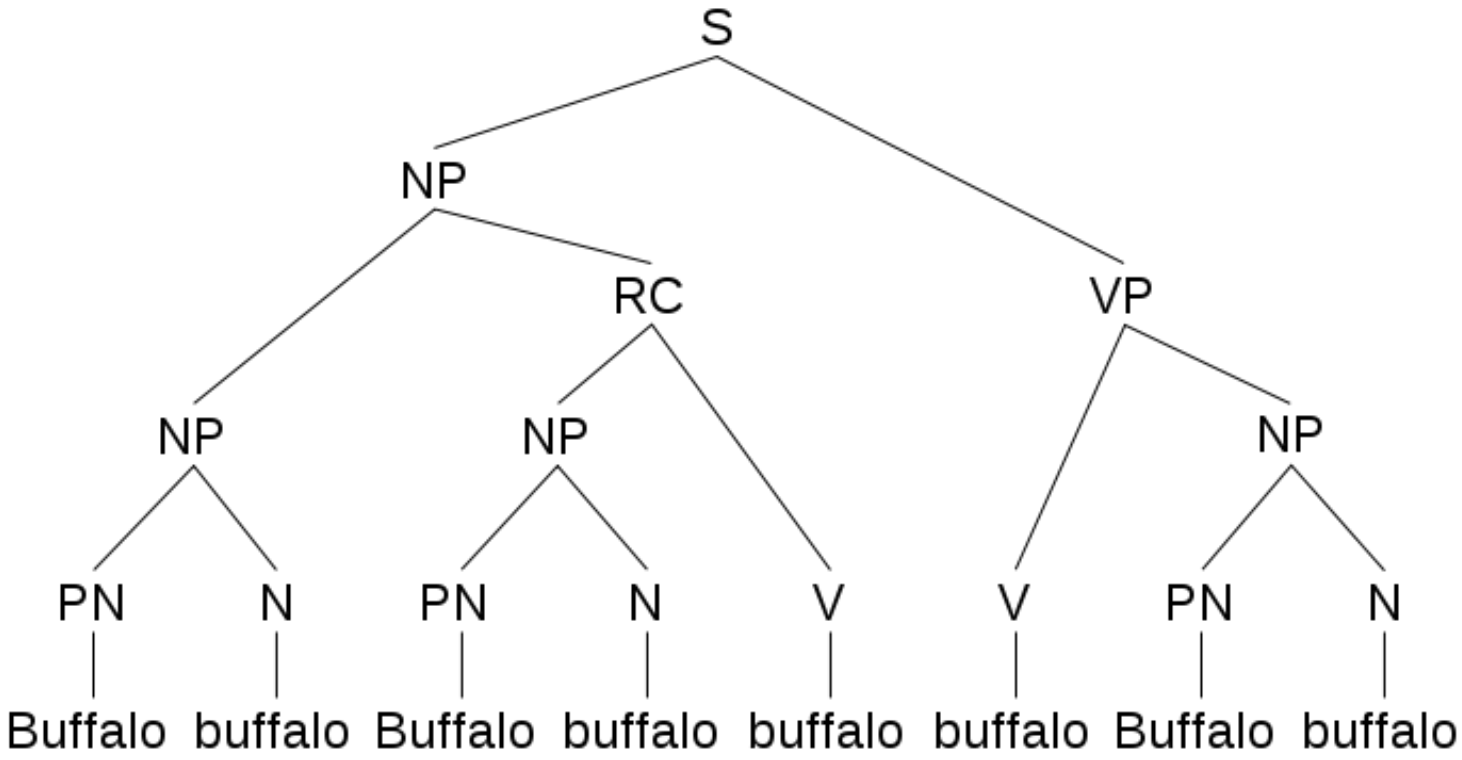
Recursion is common in grammar rules, e.g.

NP → NP RC

Because of this, sentences of arbitrary length are possible.

Recursion in Grammars

"Buffalo buffalo Buffalo buffalo buffalo buffalo Buffalo buffalo".



Grammars

Its also possible to have “sentences” inside other sentences…

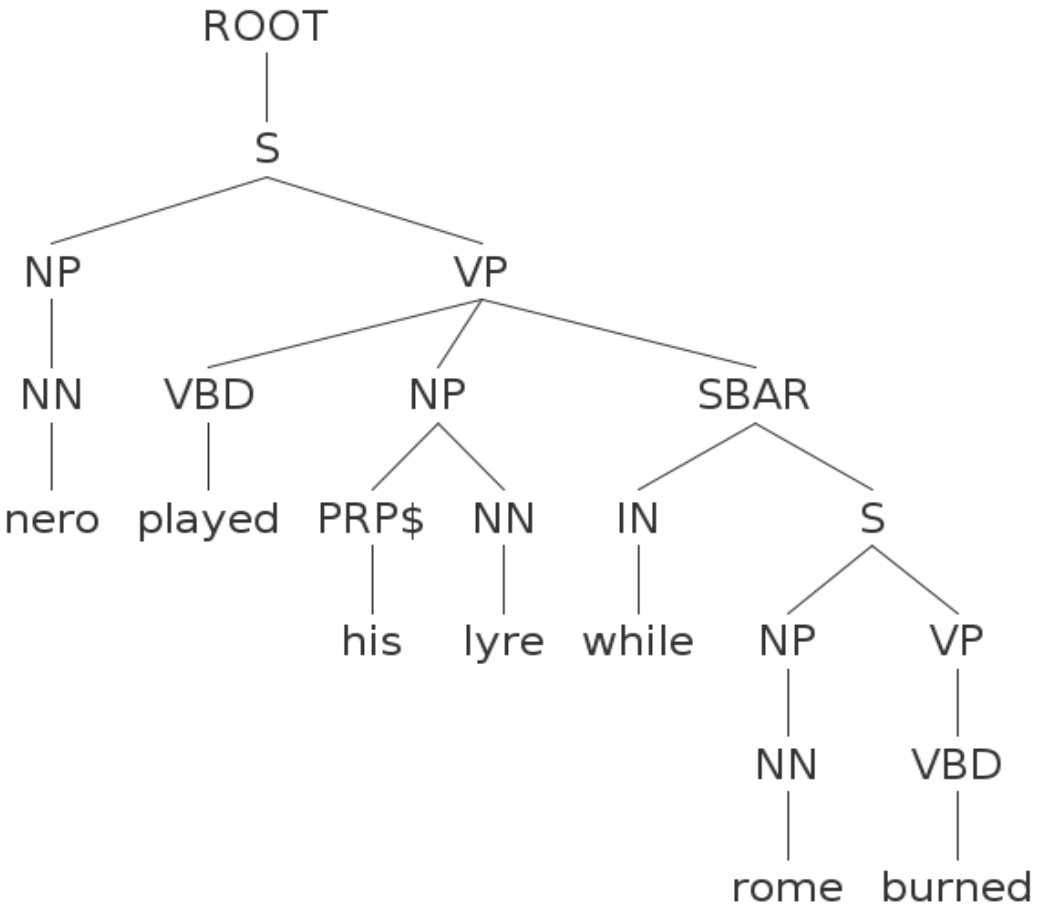
S → NP VP

VP → VB NP SBAR

SBAR → IN S

Recursion in Grammars

"Nero played his lyre while Rome burned".



PCFGs

Complex sentences can be parsed in many ways, most of which make no sense or are extremely improbable (like Groucho's example).

Probabilistic Context-Free Grammars (PCFGs) associate and learn probabilities for each rule:

S → NP VP 0.3

S → NP VP PP 0.7

The parser then tries to find the most likely sequence of productions that generate the given sentence. This adds more realistic "world knowledge" and generally gives much better results.

Most state-of-the-art parsers these days use PCFGs.

Systems

- **NLTK:** Python-based NLP system. Many modules, good visualization tools, but not quite state-of-the-art performance.
- **Stanford Parser:** Another comprehensive suite of tools (also POS tagger), and state-of-the-art accuracy. Has the definitive dependency module.
- **Berkeley Parser:** Slightly higher parsing accuracy (than Stanford) but not as many modules.
- Note: high-quality parsing is usually very slow, but see: https://github.com/dlwh/puck

Outline for this Evening

- Project Suggestions Overview
- N-grams
- Grammars
- Parsing
- Dependencies

Dependencies

In a constituency parse, there is no direct relation between the constituents and words from the sentence (except for leaf nodes which produce a single word).

In dependency parsing, the idea is to decompose the sentence into relations directly between words.

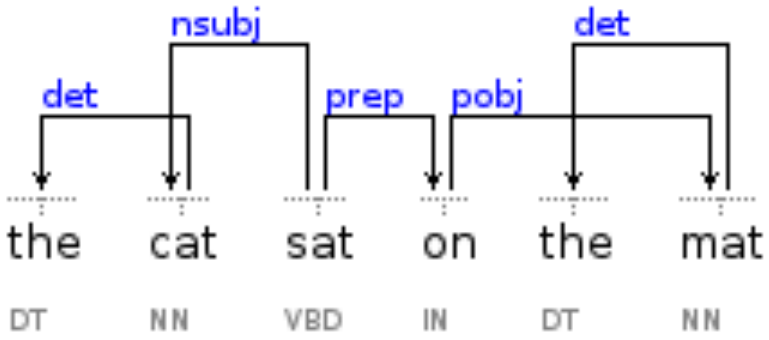
This is an older, and some argue more natural, decomposition of the sentence. It also often makes semantic interpretation (based on the meanings of the words) easier.

Lets look at a simple example:

Dependencies

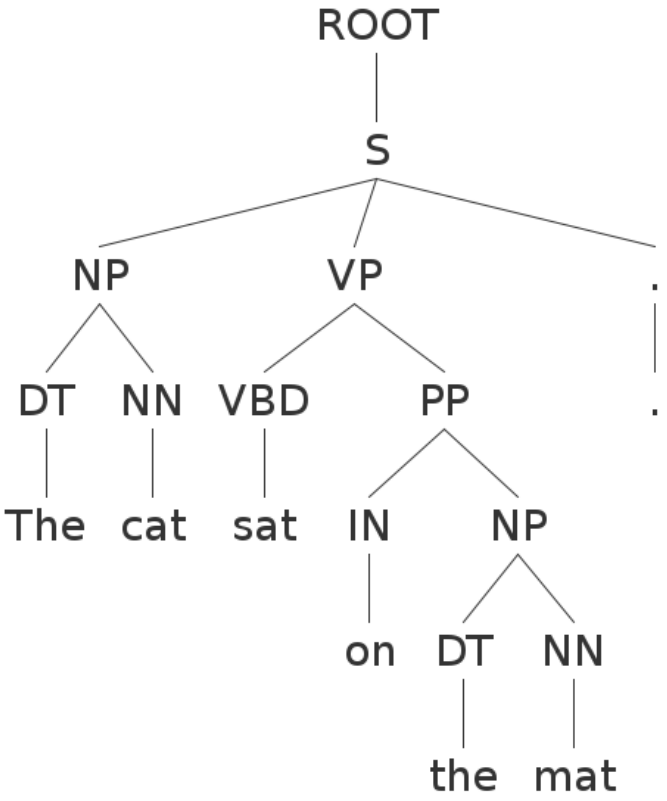
"The cat sat on the mat"

dependency tree



constituency labels of leaf nodes

parse tree



Dependencies

From the dependency tree, we can obtain a “sketch” of the sentence. i.e. by starting at the root we can look down one level to get:

“cat sat on”

And then by looking for the object of the prepositional child, we get:

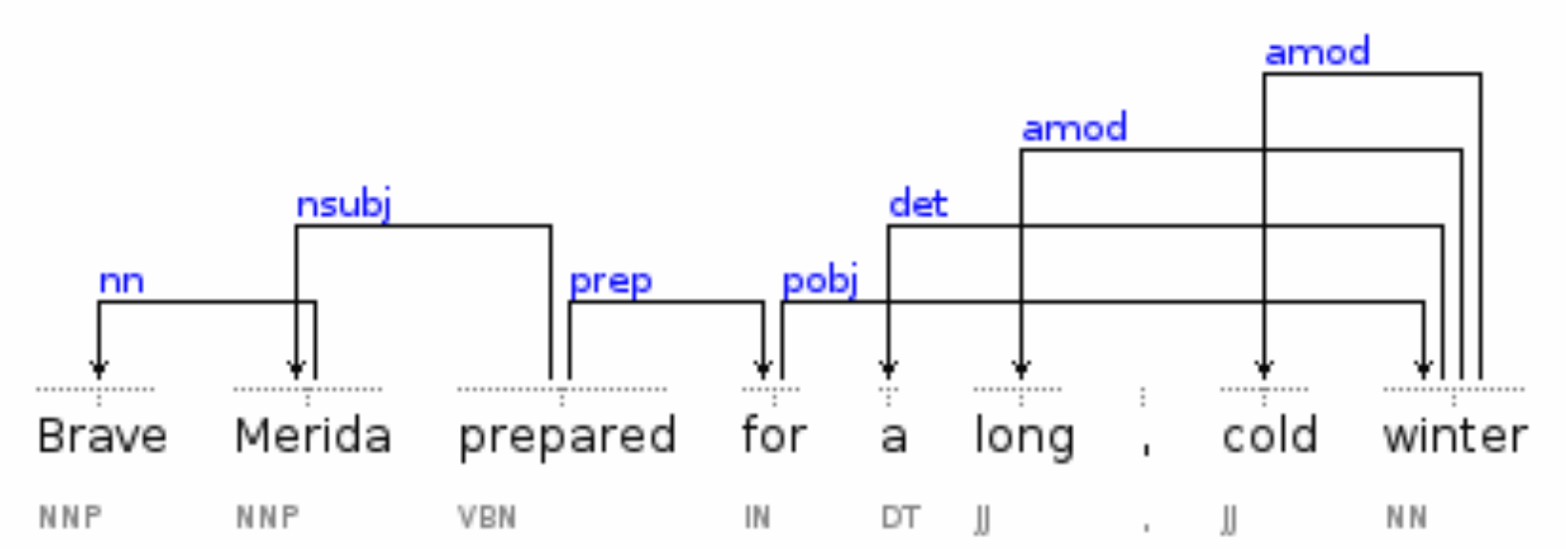
“cat sat on mat”

We can easily ignore determiners “a, the”.

And importantly, adjectival and adverbial modifiers generally connect to their targets:

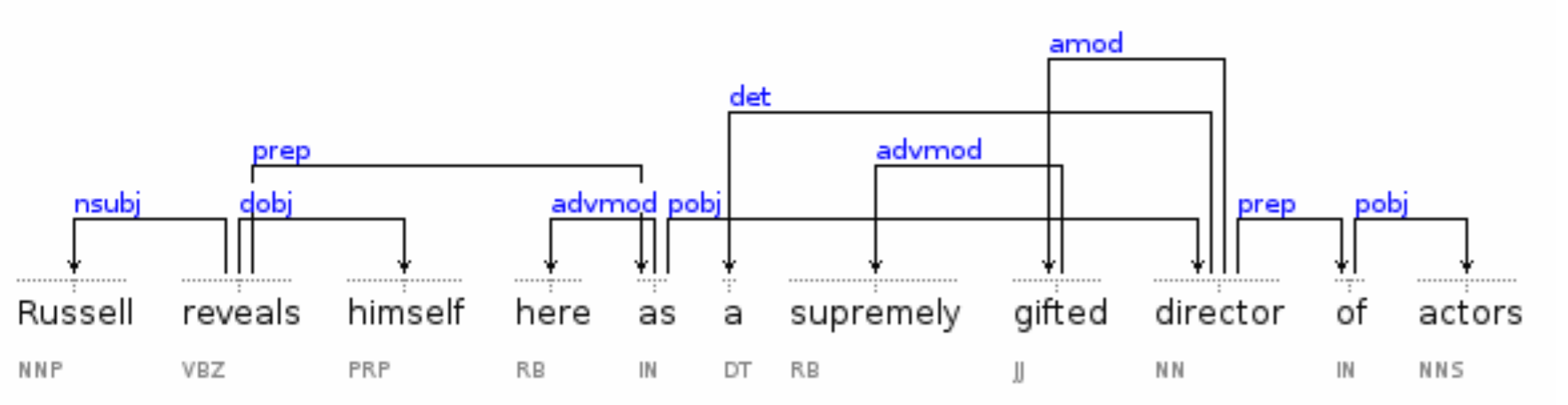
Dependencies

"Brave Merida prepared for a long, cold winter"



Dependencies

"Russell reveals himself here as a supremely gifted director of actors"



Dependencies

Stanford dependencies are constructed from the output of a constituency parser (so you can in principle use other parsers).

The mapping is based on hand-written regular expressions.

Dependency grammars have been widely used for sentiment analysis and for semantic embedings of sentences.

Summary

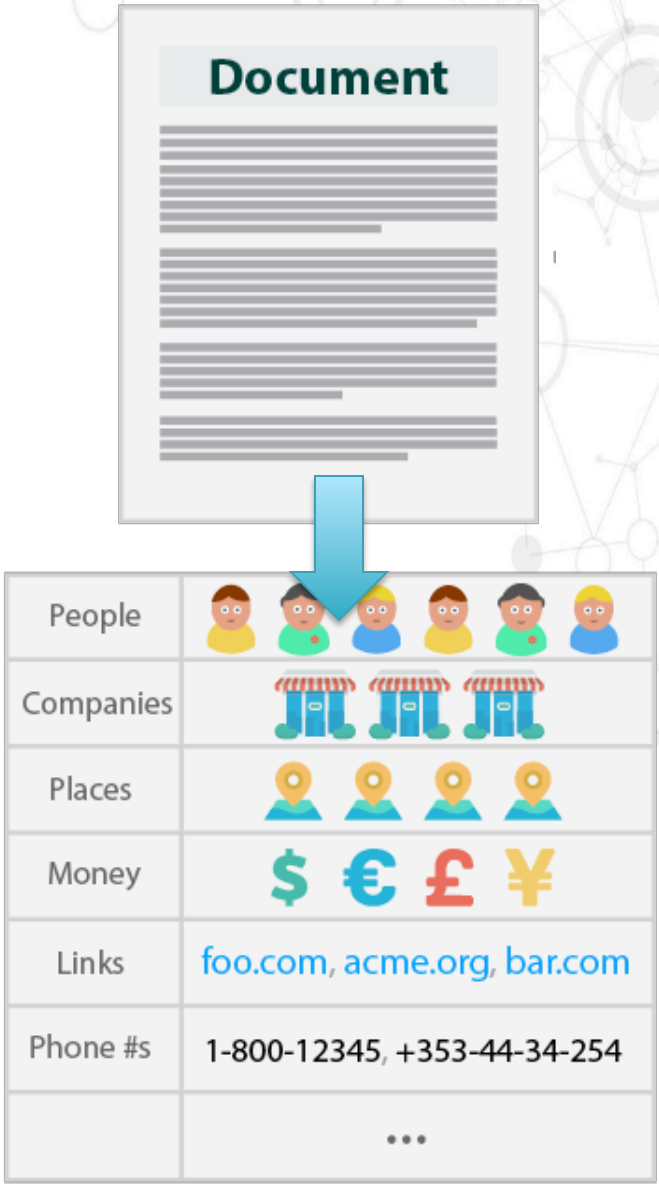
- Project Presentation Schedule
- N-grams
- Grammars
- Parsing
- Dependencies

Named Entity Extraction

- While providing some context to the terms, the n-gram technique still treats each term equally without any differentiation or categorization.
- In order to mine text based on certain semantics, terms can be extracted through named entity extraction, which is a more complex and intelligent technique that uses certain semantic rules for extracting more meaningful terms.
- Meaningful terms are called entities and are extracted based on some classification, such as named entities (person, group, place, or company), pattern-based entities (coordinates or ZIP code), concepts (abstract entity-like vehicle or human), facts (connections between entities), and sentiments (attitude, bias, or emotion).

Named Entity Extraction

- Named entity extraction is generally automated using an entity extractor, which may provide certain common types of entity extraction features out of the box or an interface for creating rules for complex entity extraction.
- Named entity extraction is a resource and time-sensitive approach as the entity extractor may need to be educated, either by manually specifying the entities or automatically, through machine learning (classification).



Cosine Distance

- Document search, classification, and clustering tasks are based on similarity between documents, such as between a search term (a document) and a corpus (multiple documents).
- This similarity is generally determined via cosine similarity, which is based on calculating cosine distance between documents. This can be calculated between two documents, or a document and the whole corpus, in order to find out which ones are the closest match.
- Apart from searching documents, the cosine similarity metric can also be used in classification and clustering algorithms. For example, the cosine similarity metric can be used to determine the distance between the instances for k-NN and k-means algorithms.

  - $\cos(d_1, d_2) = (d_1 \bullet d_2) / ||d_1|| \, ||d_2||$,
    where $\bullet$ indicates vector dot product, $||d|$: the length of vector d

  - Ex: Find the **similarity** between documents 1 and 2.

    $d_1 = (5, 0, 3, 0, 2, 0, 0, 2, 0, 0)$
    $d_2 = (3, 0, 2, 0, 1, 1, 0, 1, 0, 1)$

    $d_1 \bullet d_2 = 5*3+0*0+3*2+0*0+2*1+0*1+0*1+2*1+0*0+0*1 = 25$
    $||d_1|| = (5*5+0*0+3*3+0*0+2*2+0*0+0*0+2*2+0*0+0*0)^{0.5} = (42)^{0.5} = 6.481$
    $||d_2|| = (3*3+0*0+2*2+0*0+1*1+1*1+0*0+1*1+0*0+1*1)^{0.5} = (17)^{0.5} = 4.12$
    $\cos(d_1, d_2) = 0.94$

Types of Text Analytics

- Text analytics make use of supervised and unsupervised learning techniques for classifying and grouping documents.
- Within supervised learning, Naïve Bayes is heavily used for categorizing documents based on the probability of the occurrence of individual terms, such as spam filtering, which makes use of the bag of words and term frequency techniques.
- k-NN is used for classifying similar documents using cosine distance
- Standard Parsing Approach:
  - Rule-based/Case-based Using grammars
  - Augmented Transition Networks
- Statistical:
  - Naïve Bayes
  - K Means Clustering
  - K Nearest Neighbors
  - Decision tree Induction
  - Support Vector Machines
  - Latent Semantic Analysis (albeit, no semantics)
  - Apriori algorithm (extraction of frequent itemsets or terms)