DIPARTIMENTO DI MATEMATICA E INFORMATICA

UNIVERSITÀ degli STUDI di CATANIA

Social Media Data Analysis 2023/2024

Introduction to Text Analysis

Francesco Ragusa
francesco.ragusa@unict.it
https://iplab.dmi.unict.it/ragusa/

IMAGE PROCESSING LABORATORY

https://iplab.dmi.unict.it/fpv/

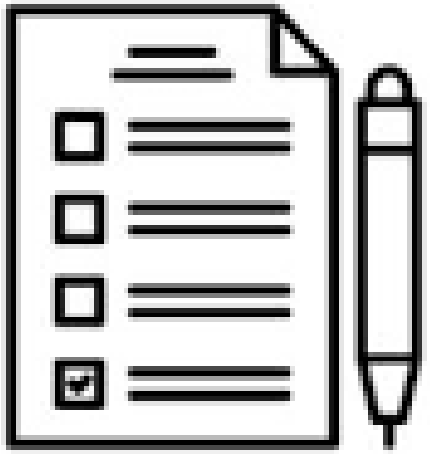
- Giovedì 8 Febbraio
- Giovedì 22 Febbraio
- Giovedì 27 Giugno
- Giovedì 11 Luglio
- Venerdì 13 Settembre
- Venerdì 27 Settembre

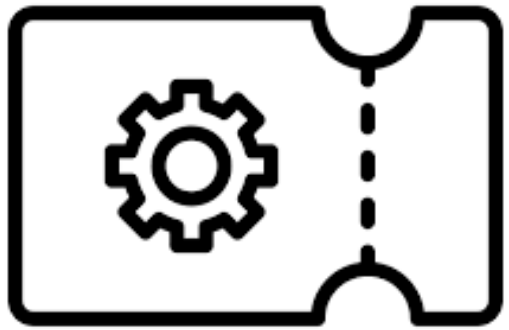
What?

Text analysis is *the* process to extract and classify relevant information out of them using different techniques from unstructured text(s)

Topic extraction, Sentiment analysis, Aspect classification, Named entity extraction, and more ..

Where?





Do I need to perform text analysis?

Don’t tell mom about this!”

Sentiment: *Negative*

Emotion: *Fear*

Named entity: *Mom*

Topic Category: *A usual Sibling thing!*

Why?

- Text Classification

Why?

- Text Classification
- Sentiment Analysis

Why?

- Text Classification
- Sentiment Analysis
- Topic Analysis

Why?

- Text Classification
- Sentiment Analysis
- Topic Analysis
- Intent Detection

Why?

- Text Classification
- Sentiment Analysis
- Topic Analysis
- Intent Detection
- Text Extraction

Why?

- Text Classification
- Sentiment Analysis
- Topic Analysis
- Intent Detection
- Text Extraction
- Word Frequency

Text

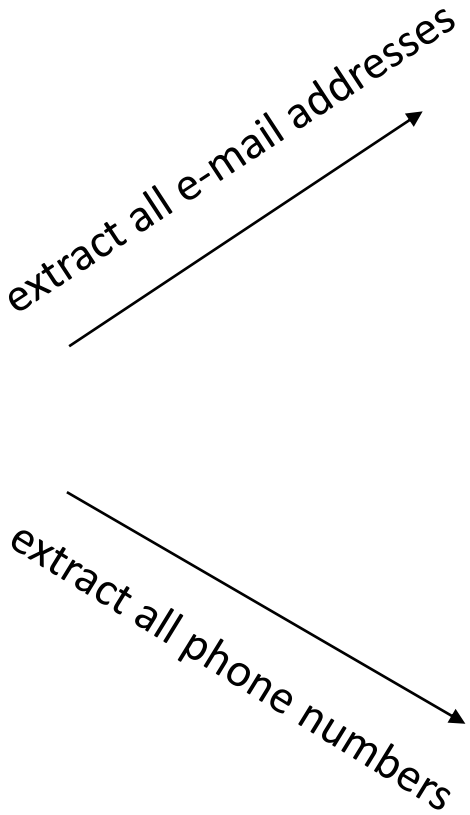
Regular Expressions (REGEX) (.*)

Natural Language Processing

Example: Information Extraction (structured)

Text

You should send that information to address1@example.com, address2@domain.eu, and add@ex.ck.pt. Also, you should cc aaa@bb.cc and 1234@mail.co.uk. Also remember to call 134-22-90 and 999-22-12.



address1@example.com
address2@domain.eu
add@ex.ck.pt
aaa@bb.cc
1234@mail.co.uk

134-22-90
999-22-12

Example: Information Extraction (unstructured)

Text

Hy John,
We should meet tomorrow from 12:00 to 13:00 at the R305 for the syllabus meeting.

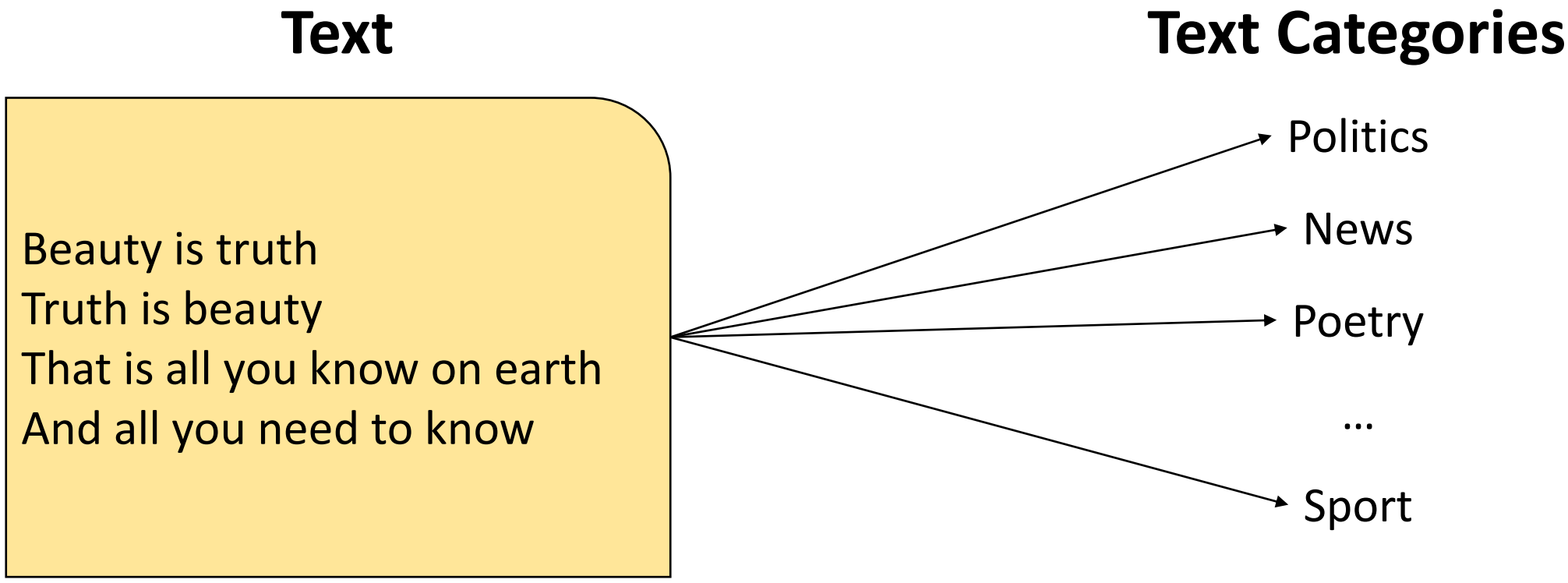
- James

Create calendar entry

Calendar Entry

Event: Syllabus Meeting
Date: Tomorrow
Start: 12:00
End: 13:00
Location: R305

Example: Text Categorization



Example: Sentiment Analysis

- Unbelievably disappointing

- Full of zany characters and richly applied satire, and some great plot twists

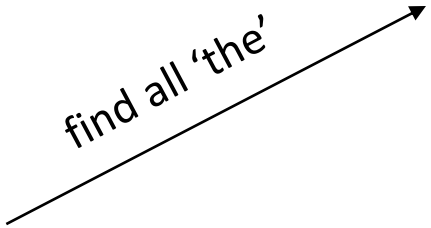
- this is the greatest screwball comedy ever filmed

- It was pathetic. The worst part about it was the boxing scenes.

Zoom is a great app that has helped me so much during the COVID 19 pandemic	en	0.8	😀
Zoom is a great virtual tool for my group of ladies. We are able to have our meetings and accomplish our task via Zoom. The meetings are extremely productive and fun.	en	0.7	😀
So far so good. But the competition may soon get the upper hand by offering Video-audio Communication facilities free of any charge. “Make hay while the sun shines,” so says an English saying. But, seriously, this is a wonderful invention whose current optimum output coincided with the emergence of a terrible disease that, literally, forces hundreds of thousands of people to stay out within the confines of their own homes. So, y’all came JUST IN TIME when your services are needed most.	en	-0.6	😠

Regular Expressions

The other day, I was walking and I found the reason of the situation. The theme is always the same for them.



The other day, I was walking and I found **the** reason of **the** situation. **The** theme is always **the** same for them.

≠

The o**the**r day, I was walking and I found **the** reason of **the** situation. **The** **the**me is always **the** same for **the**m.

Regular Expressions: A string is a regular expression

- ‘the’ ->

The other day, I was walking and I found the reason of the situation. The theme is always the same for them.

- ‘I’ ->

The other day, I was walking and I found the reason of the situation. The theme is always the same for them.

- ‘a’->

The other day, I was walking and I found the reason of the situation. The theme is always the same for them.

Regular Expressions: Sets

‘[Tt]’ ->

The other day, I was walking and I found the reason of the situation. The theme is always the same for them.

‘[Tt]he’ ->

The other day, I was walking and I found the reason of the situation. The theme is always the same for them.

‘[A-Z]’ ->

The other day, I was walking and I found the reason of the situation. The theme is always the same for them.

‘[a-z]’ ->

The other day, I was walking and I found the reason of the situation. The theme is always the same for them.

‘[0-9]’ ->

The other day (day 1), I was walking and I found the reason of the situation (2). The theme is always the same for them.

‘[A-Z0-9.]’ ->

The other day (day 1), I was walking and I found the reason of the situation (2). The theme is always the same for them.

Regular Expressions: Negations and Compounds

[^A-Za-z] ->

```
The other day (day 1), I was walking and I found the reason of the
situation (2). The theme is always the same for them.
```

[0-9][0-9] ->

```
The other day (day 11), I was walking and I found the reason of
the situation (2). The theme is always the same for them.
```

[a-z][a-z]e ->

```
The other day (day 11), I was walking and I found the reason of
the situation (2). The theme is always the same for them.
```

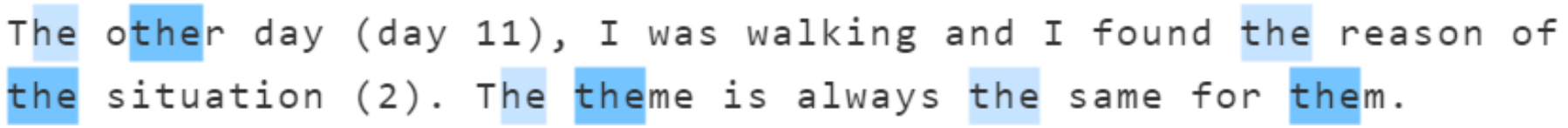
[0-9][13579] -> 12-22-45 128 352 405 902

day|and ->

```
The other day (day 11), I was walking and I found the reason of
the situation (2). The theme is always the same for them.
```

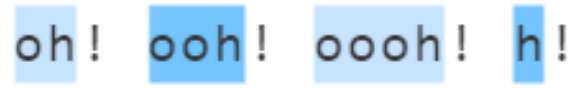
Regular Expressions: Quantifiers ? * + . {}

t?he ->

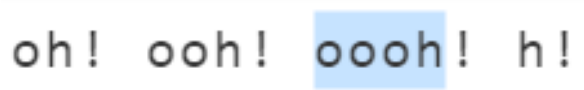


o+h ->

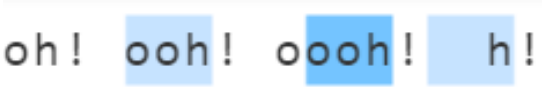
o*h ->



o{3}h ->

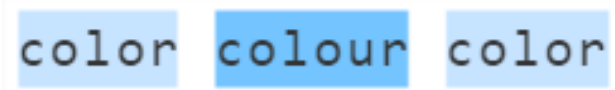


.{2}h ->

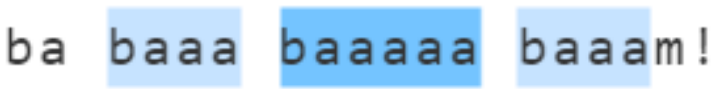


[oh]+! ->

colou?r->

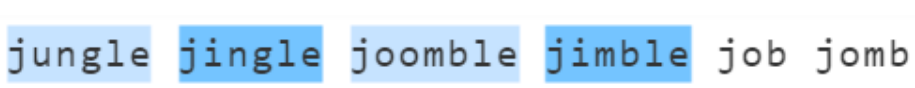


baa+->



ba+m?[!?]->

j[aeiou]+[mn].{3}->



[A-Za-z_]+->

Regular Expressions: Escape Sequences

e. -> I'm there.

.? -> Where am I?

.* -> Type '*';

.{2} -> Type {2}

\s -> Hello how are you?

[^\s]+ing -> doing eating ingenous ingenious

e\. -> I'm there.

.\? -> Where am I?

.* -> Type '*';

.\{2\} -> Type {2}

^[Tt]he -> The the The The the

[Tt]he$ -> The the The The the

Regular Expressions: Groups

([Tt]he)+ -> Them them theorem thethe theTheThethe

([0-9][0-9]_?)+ -> 01_02_04 02_05_09 _02_12_18 12__34 123_43

([0-9]+_[a-z]+)-([a-z]+_[0-9]+) -> 123_abc-defg_398 123_ggg-123_ggg 123_123-fff_234

(#[0-9]+#_?){3} -> #123#_#42567#_#22# #123#_#22#

(d[aeiou]p)+ -> depdopdipdup dopddt depdipdnp

Regular Expressions: examples

[^\.](\.?[A-Za-z_]+)+\.([A-Za-z]+)

google.com dmi.unict.it unict.it ex_ampe.com gp.co.uk exmaple

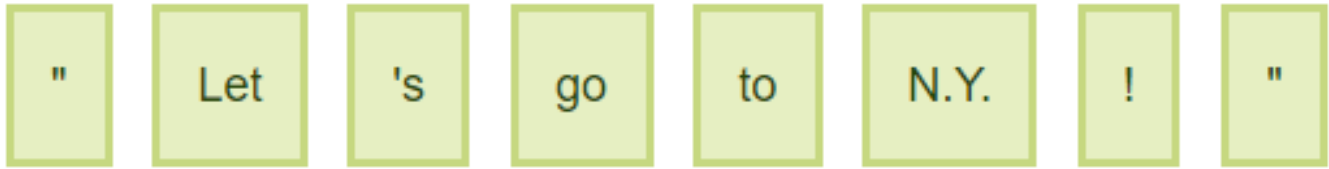
https?:\/\/([A-Za-z]+)?.[^\.](\.?[A-Za-z_]+)+\.([A-Za-z]+)(\/[A-Za-z_\/.]*)?([#?][A-Za-z=0-9&_]*)?

https://www.goo_gle.com/path/to/url.php?page=2,

Natural Language Processing (NLP)

Word Tokenization

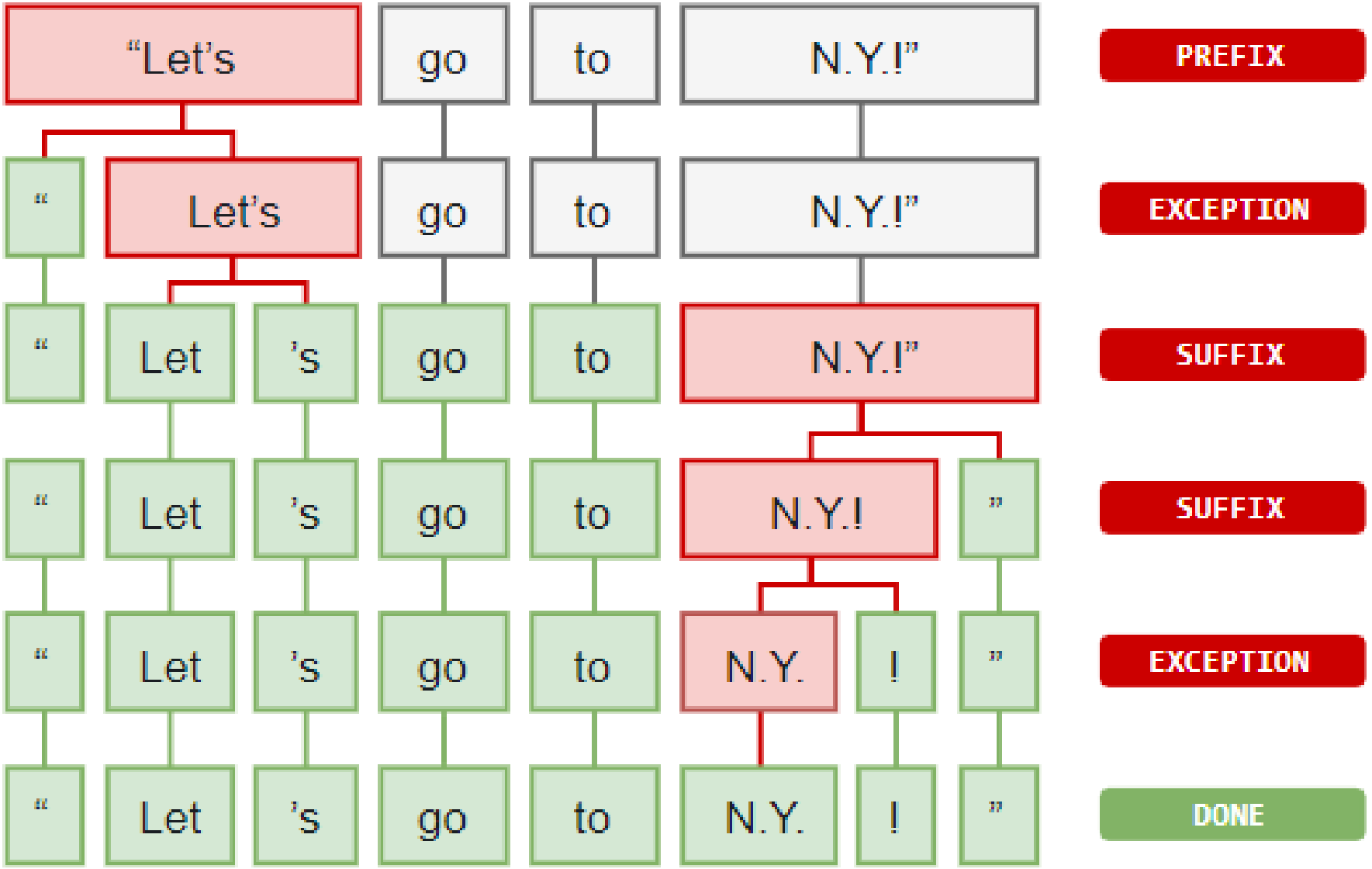
"Let’s go to N.Y.!"



Word Tokenization: Prefix, Suffix, Infix, Exceptions

- Prefix: characters at the beginning; "hello, $21.2, (for example
- Suffix: characters at the end; said", 22km, gain)
- Infix: characters in between; so-called, email_address
- Exception: Special-case rules to split a string into several tokens or prevent a token from being split when punctuation rules are applied N.Y., Let's

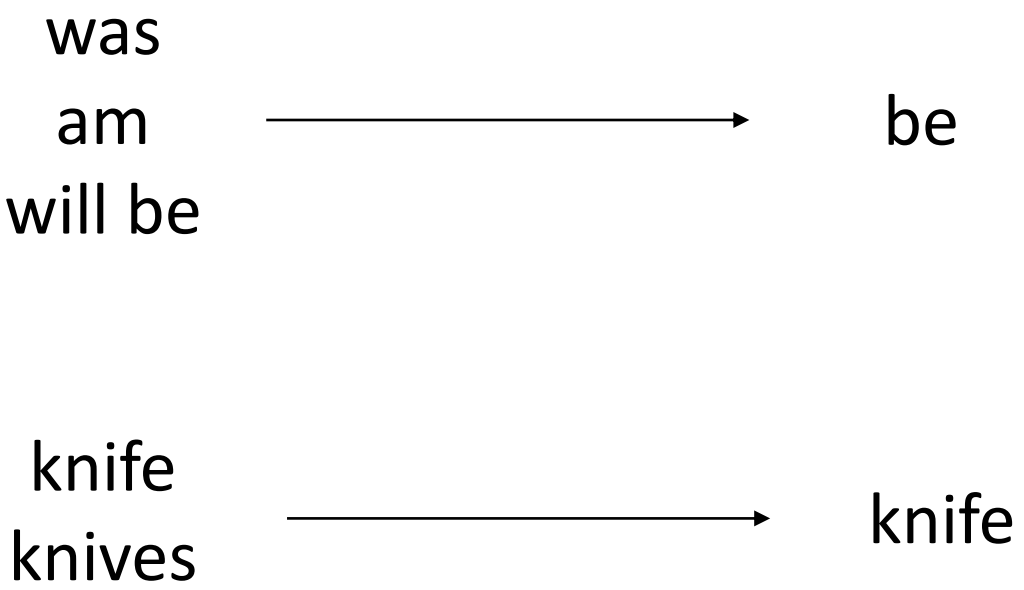
Word Tokenization: Example



match words with a vocabulary + context!

(e.g., meeting vs meeting)

Lemmatization



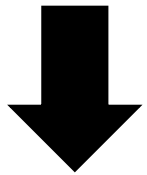
How to write «HELLO» in mandarin?

Stop Words

i me my myself we our ours ourselves you your yours yourself yourselves he him his himself she her hers herself it its itself they them their theirs themselves what which who whom this that these those am is are was were be been being have has had having do does did doing a an the and but if or because as until while of at by for with about against between into through during before after above below to from up down in out on off over under again further then once here there when where why how all any both each few more most other some such no nor not only own same so than too very s t can will just don should now

Part of Speech (POS) Tagging

John likes the blue house at the end of the street.



John likes the blue house at the end of the street .

- Adjective
- Adverb
- Conjunction
- Determiner
- Noun
- Number
- Preposition
- Pronoun
- Verb

Coarse-Grained POS Tags

POS	DESCRIPTION	EXAMPLES
ADJ	adjective	*big, old, green, incomprehensible, first*
ADP	adposition	*in, to, during*
ADV	adverb	*very, tomorrow, down, where, there*
AUX	auxiliary	*is, has (done), will (do), should (do)*
CONJ	conjunction	*and, or, but*
CCONJ	coordinating conjunction	*and, or, but*
DET	determiner	*a, an, the*
INTJ	interjection	*psst, ouch, bravo, hello*
NOUN	noun	*girl, cat, tree, air, beauty*
NUM	numeral	*1, 2017, one, seventy-seven, IV, MMXIV*
PART	particle	*'s, not,*
PRON	pronoun	*I, you, he, she, myself, themselves, somebody*
PROPN	proper noun	*Mary, John, London, NATO, HBO*
PUNCT	punctuation	*., (,), ?*
SCONJ	subordinating conjunction	*if, while, that*
SYM	symbol	*$, %, §, ©, +, −, ×, ÷, =, :), 😝*
VERB	verb	*run, runs, running, eat, ate, eating*
X	other	*sfpksdpsxmsa*
SPACE	space	

Fine-Grained POS Tags

POS	Description	Fine-grained Tag	Description	Morphology
ADJ	adjective	AFX	affix	Hyph=yes
ADJ		JJ	adjective	Degree=pos
ADJ		JJR	adjective, comparative	Degree=comp
ADJ		JJS	adjective, superlative	Degree=sup
ADJ		PDT	predeterminer	AdjType=pdt PronType=prn
ADJ		PRP$	pronoun, possessive	PronType=prs Poss=yes
ADJ		WDT	wh-determiner	PronType=int rel
ADJ		WP$	wh-pronoun, possessive	Poss=yes PronType=int rel
ADP	adposition	IN	conjunction, subordinating or preposition	
ADV	adverb	EX	existential there	AdvType=ex
ADV		RB	adverb	Degree=pos
ADV		RBR	adverb, comparative	Degree=comp
ADV		RBS	adverb, superlative	Degree=sup
ADV		WRB	wh-adverb	PronType=int rel
CONJ	conjunction	CC	conjunction, coordinating	ConjType=coor
DET	determiner	DT	determiner	
INTJ	interjection	UH	interjection	
NOUN	noun	NN	noun, singular or mass	Number=sing
NOUN		NNS	noun, plural	Number=plur
NOUN		WP	wh-pronoun, personal	PronType=int rel
NUM	numeral	CD	cardinal number	NumType=card
PART	particle	POS	possessive ending	Poss=yes
PART		RP	adverb, particle	

Fine-Grained POS Tags (2)

POS	Description	Fine-grained Tag	Description	Morphology
PART		TO	infinitival to	PartType=inf VerbForm=inf
PRON	pronoun	PRP	pronoun, personal	PronType=prs
PROPN	proper noun	NNP	noun, proper singular	NounType=prop Number=sign
PROPN		NNPS	noun, proper plural	NounType=prop Number=plur
PUNCT	punctuation	-LRB-	left round bracket	PunctType=brck PunctSide=ini
PUNCT		-RRB-	right round bracket	PunctType=brck PunctSide=fin
PUNCT		,	punctuation mark, comma	PunctType=comm
PUNCT		:	punctuation mark, colon or ellipsis	
PUNCT		.	punctuation mark, sentence closer	PunctType=peri
PUNCT		''	closing quotation mark	PunctType=quot PunctSide=fin
PUNCT		""	closing quotation mark	PunctType=quot PunctSide=fin
PUNCT		``	opening quotation mark	PunctType=quot PunctSide=ini
PUNCT		HYPH	punctuation mark, hyphen	PunctType=dash
PUNCT		LS	list item marker	NumType=ord
PUNCT		NFP	superfluous punctuation	
SYM	symbol	#	symbol, number sign	SymType=numbersign

Fine-Grained POS Tags (3)

POS	Description	Fine-grained Tag	Description	Morphology
SYM		$	symbol, currency	SymType=currency
SYM		SYM	symbol	
VERB	verb	BES	auxiliary "be"	
VERB		HVS	forms of "have"	
VERB		MD	verb, modal auxiliary	VerbType=mod
VERB		VB	verb, base form	VerbForm=inf
VERB		VBD	verb, past tense	VerbForm=fin Tense=past
VERB		VBG	verb, gerund or present participle	VerbForm=part Tense=pres Aspect=prog
VERB		VBN	verb, past participle	VerbForm=part Tense=past Aspect=perf
VERB		VBP	verb, non-3rd person singular present	VerbForm=fin Tense=pres
VERB		VBZ	verb, 3rd person singular present	VerbForm=fin Tense=pres Number=sing Person=3
X	other	ADD	email	
X		FW	foreign word	Foreign=yes
X		GW	additional word in multi-word expression	
X		XX	unknown	
SPACE	space	_SP	space	
		NIL	missing tag	

Named Entity Recognition (NER)

Boris Johnson is to offer EU leaders a historic grand bargain on Brexit — help deliver his new deal this week or agree a “no-deal” departure by October 31.

Boris Johnson PERSON is to offer EU ORG leaders a historic grand bargain on Brexit GPE — help deliver his new deal this week DATE or agree a “no-deal” departure by October 31 DATE .

An Apple computer can cost a few thousand dollars. However, prices should drop down by 20% in a few weeks.

An Apple ORG computer can cost a few thousand dollars MONEY . However, prices should drop down by 20% PERCENT in a few weeks DATE .

List of Named Entities

TYPE	DESCRIPTION	EXAMPLE
`PERSON`	People, including fictional.	*Fred Flintstone*
`NORP`	Nationalities or religious or political groups.	*The Republican Party*
`FAC`	Buildings, airports, highways, bridges, etc.	*Logan International Airport, The Golden Gate*
`ORG`	Companies, agencies, institutions, etc.	*Microsoft, FBI, MIT*
`GPE`	Countries, cities, states.	*France, UAR, Chicago, Idaho*
`LOC`	Non-GPE locations, mountain ranges, bodies of water.	*Europe, Nile River, Midwest*
`PRODUCT`	Objects, vehicles, foods, etc. (Not services.)	*Formula 1*
`EVENT`	Named hurricanes, battles, wars, sports events, etc.	*Olympic Games*
`WORK_OF_ART`	Titles of books, songs, etc.	*The Mona Lisa*
`LAW`	Named documents made into laws.	*Roe v. Wade*
`LANGUAGE`	Any named language.	*English*
`DATE`	Absolute or relative dates or periods.	*20 July 1969*
`TIME`	Times smaller than a day.	*Four hours*
`PERCENT`	Percentage, including "%".	*Eighty percent*
`MONEY`	Monetary values, including unit.	*Twenty Cents*
`QUANTITY`	Measurements, as of weight or distance.	*Several kilometers, 55kg*
`ORDINAL`	"first", "second", etc.	*9th, Ninth*
`CARDINAL`	Numerals that do not fall under another type.	*2, Two, Fifty-two*

Sentence Segmentation

An Apple computer can cost a few thousand dollars. However, prices should drop down by 25.5% in a few weeks. Hope I can be myself one.

An Apple computer can cost a few thousand dollars.

However, prices should drop down by 25.5% in a few weeks.

Hope I can be myself one.

NLP Pipeline

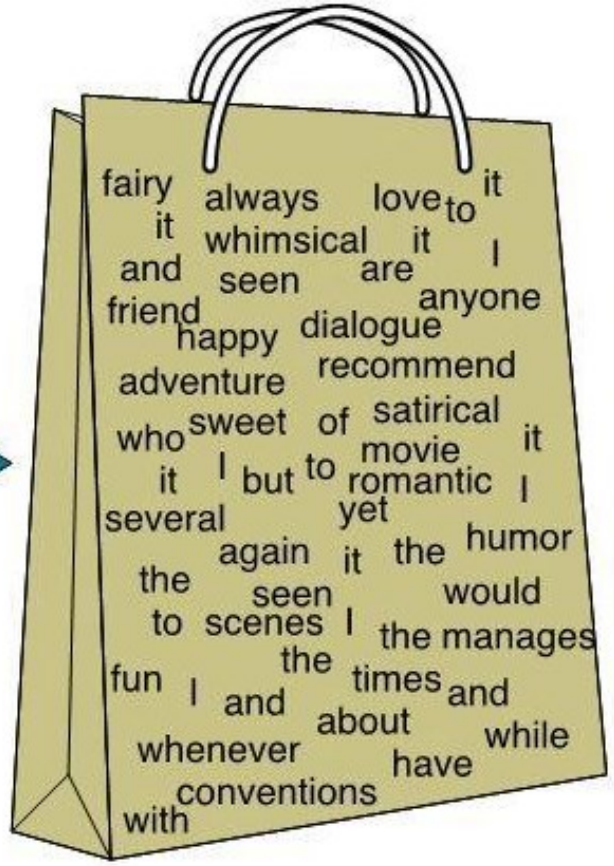
To summarize, a common NLP pipeline, generally involves the following steps:

- Word tokenization;
- Stop words removal;
- Lemmatization;
- POS tagging (depending on the application);
- NER tagging (depending on the application);
- Sentence segmentation.

how can we obtain a fixed-length representation for documents, which summarizes some statistical properties of the data?

Bag of Words Representation

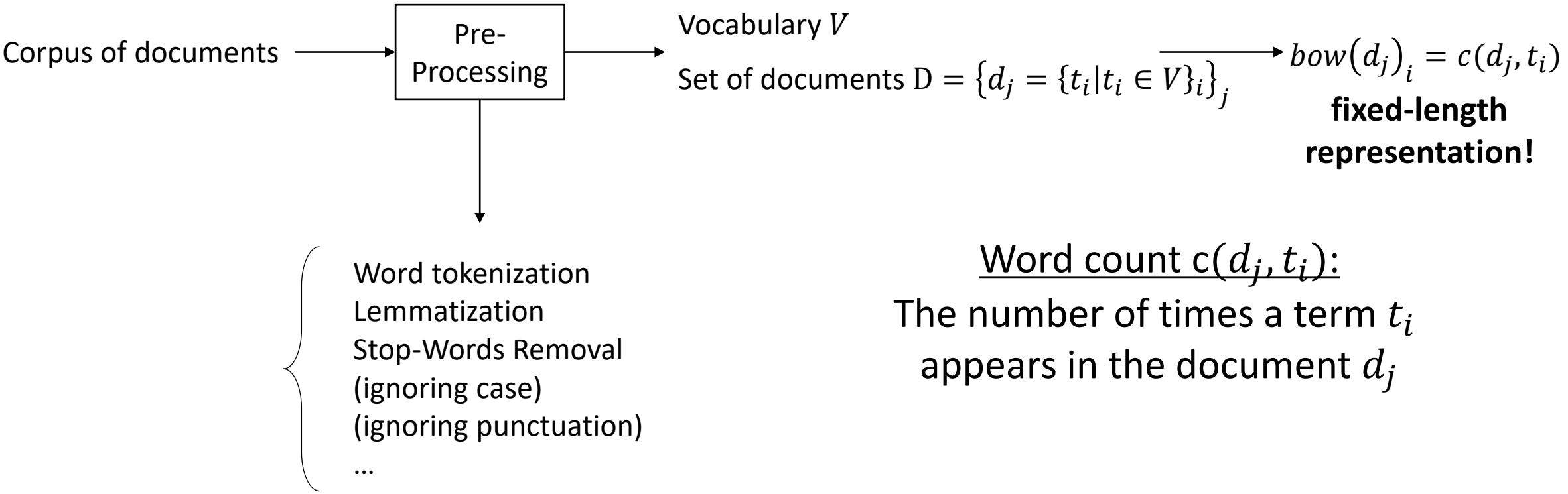
I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

documents are a collection of words: discarding syntax, yet powerful!

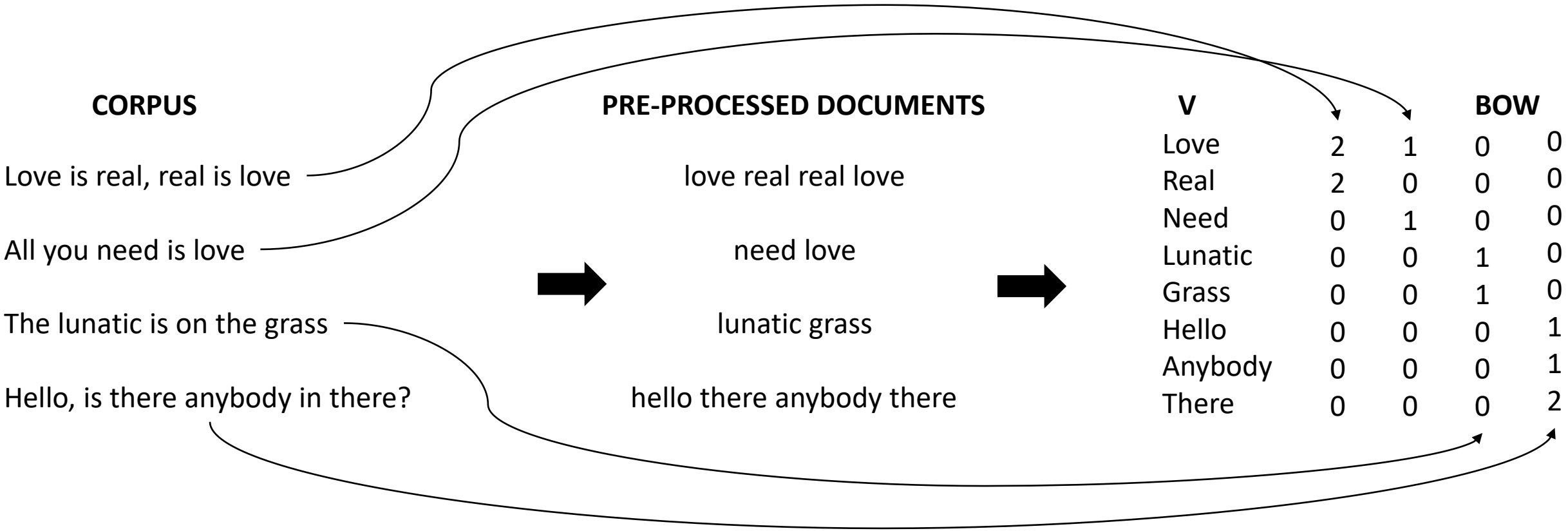
Bag of Words Representation



Word count c(d_j, t_i):

The number of times a term t_i appears in the document d_j

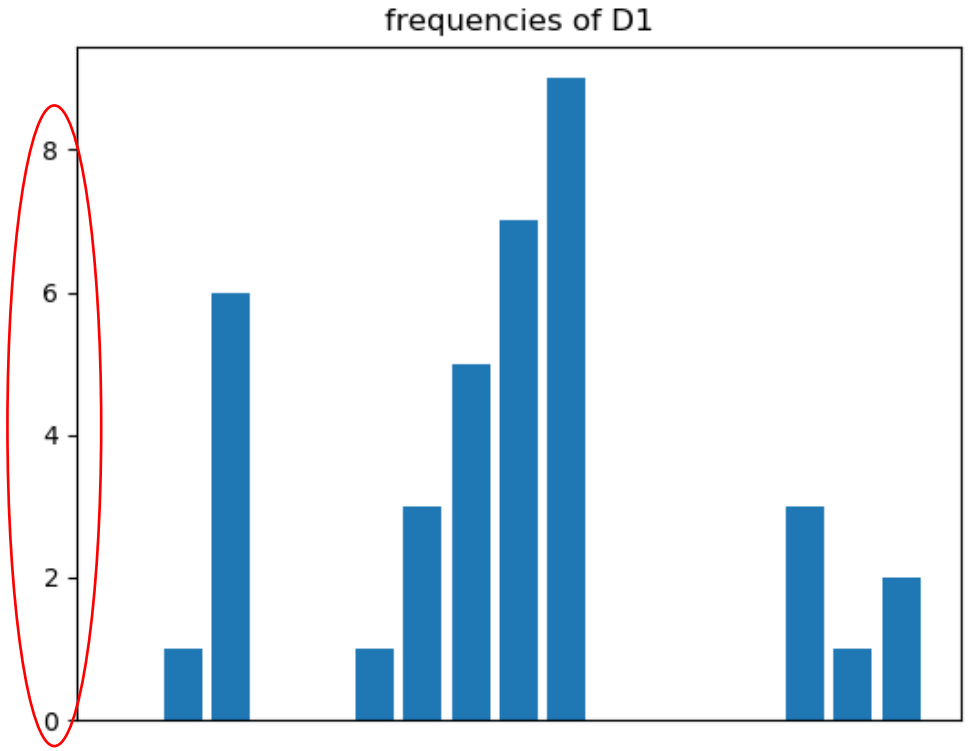
Bag of Words - Example

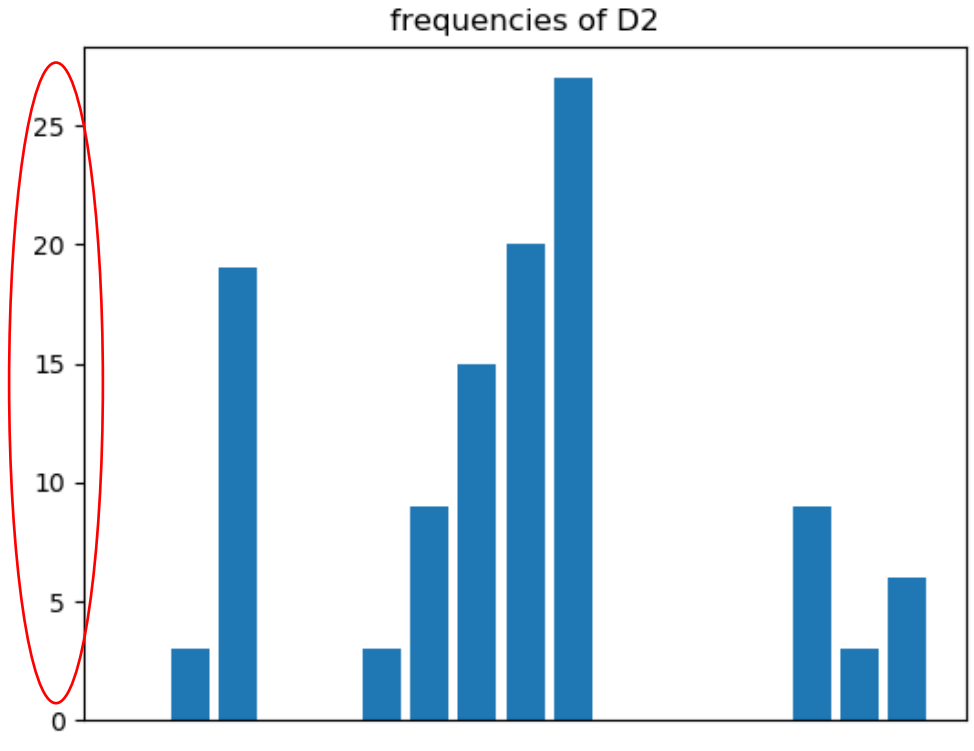


Bag of Words – Feature Normalization

```
D1: [0, 1, 6, 0, 0, 1, 3, 5, 7, 9, 0, 0, 0, 0, 3, 1, 2], #terms in D1: 38

D2: [ 0,  3, 19,  0,  0,  3,  9, 15, 20, 27,  0,  0,  0,  0,  9,  3,  6], #terms in D2: 114
```

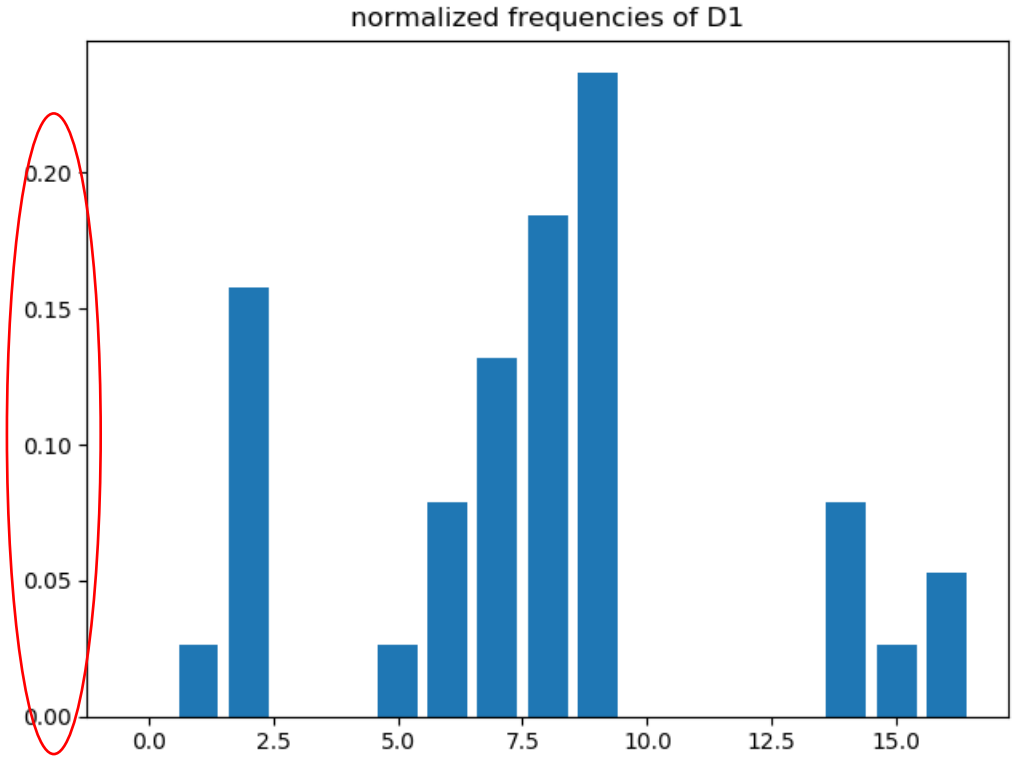


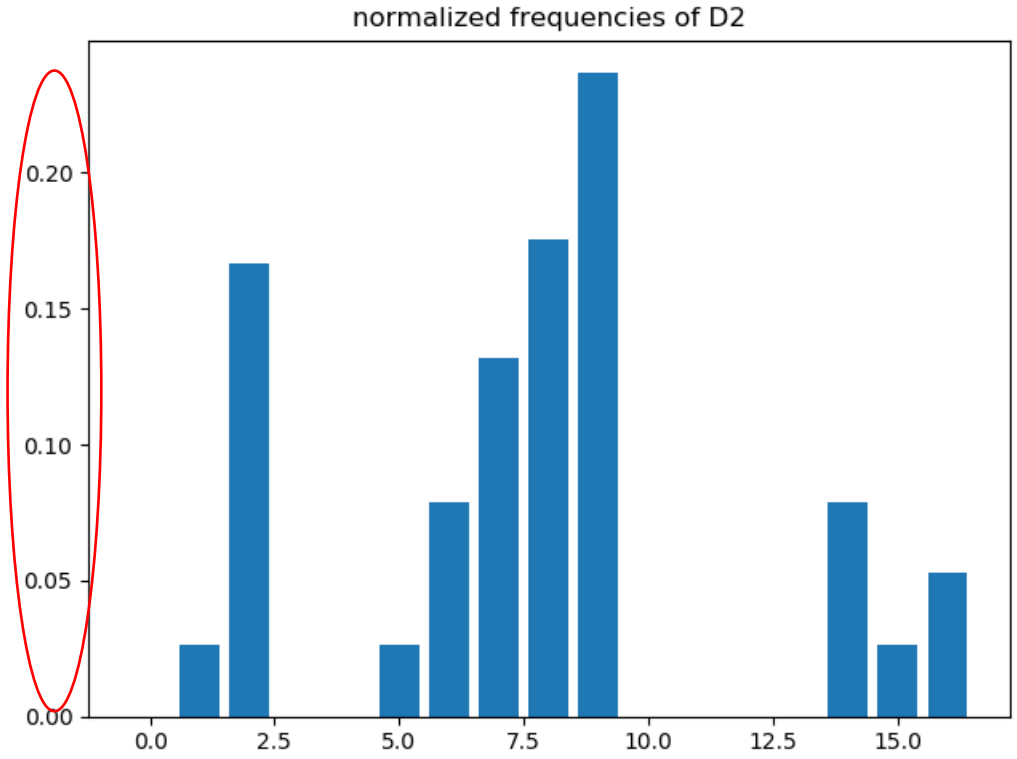


Bag of Words – Feature Normalization

divide by the number of words $nbow(d_j)_i = \frac{bow(d_j)_i}{\sum_i bow(d_j)_i} = \frac{c(d_j, t_i)}{\sum_i c(d_j, t_i)} = tf(d_j, t_i)$ ***tf*: term frequency**

```
D1: [ 0.00 0.03 0.16 0.00 0.00 0.03 0.08 0.13 0.18 0.24 0.00 0.00 0.00 0.00 0.08 0.03 0.05 ]
D2: [ 0.00 0.03 0.17 0.00 0.00 0.03 0.08 0.13 0.18 0.24 0.00 0.00 0.00 0.00 0.08 0.03 0.05 ]
```





Bag of Words – Feature Normalization

L1 Normalization

- $\overline{\boldsymbol{x}} = \frac{\boldsymbol{x}}{\sum_i |x_i|}$, where $\boldsymbol{x} = (x_1, \ldots, x_n)$;

Bag of Words – TF-IDF

Term Frequency – Inverse Document Frequency

weigh words depending on how informative they are

Bag of Words – TF-IDF

Term Frequency – Inverse Document Frequency

weigh words depending on how informative they are

n documents $\{d_i\}_i$

number of terms contained in document i as $m_i = \sum_j c(d_i, t_j)$

$$nbow(d_j)_i = \frac{bow(d_j)_i}{\sum_i bow(d_j)_i} = \frac{c(d_j, t_i)}{\sum_i c(d_j, t_i)} = tf(d_j, t_i)$$

$$tf(d_i, t_j) = \frac{c(d_i, t_j)}{m_i}$$

Bag of Words – TF-IDF

Term Frequency – Inverse Document Frequency

weigh words depending on how informative they are

IDF $p(d_i, t_j) = \begin{cases} 1 \; if \; c(d_i, t_j) > 0 \\ 0 \; otherwise \end{cases}$ $c(t_j) = \sum_i p(d_i, t_j)$

$$idf(t_j) = -\log \frac{c(t_j)}{n} = \log \frac{n}{c(t_j)} \text{ (self information)}$$

Bag of Words – TF-IDF

Term Frequency – Inverse Document Frequency

weigh words depending on how informative they are

IDF

$$idf(t_j) = -\log\frac{c(t_j)}{n} = \log\frac{n}{c(t_j)}$$

$$df(t_j) = \log\frac{c(t_j)}{n}$$

$$P(T = t_j) = \frac{c(t_j)}{n}$$

$$idf(t_j) = -logP(T = t_j)$$ **(self information $T = t_j$)**

Bag of Words – TF-IDF

Term Frequency – Inverse Document Frequency

weigh words depending on how informative they are

10 documents, and we want to calculate the idf of the word “dog”

If only one document contains the word “dog”:

$$idf(t_j) = \log \frac{n}{c(t_j)} = \log \frac{10}{1} = \mathbf{1}$$

More rare and specific

If 10 documents contain the word “dog”:

$$idf(t_j) = \log \frac{n}{c(t_j)} = \log \frac{10}{10} = \mathbf{0}$$

More common and general

Bag of Words – TF-IDF

Term Frequency – Inverse Document Frequency

weigh words depending on how informative they are

$$bow(d_i)_j = tf(d_i, t_j) \cdot idf(t_j) = \frac{c(d_i, t_j)}{\sum_j c(d_i, t_j)} \log \frac{n}{c(t_j)}$$

Bag of Words – TF-IDF

Term Frequency – Inverse Document Frequency

weigh words depending on how informative they are

$$bow(d_i)_j = tf(d_i, t_j) \cdot idf(t_j) = \frac{c(d_i, t_j)}{\sum_j c(d_i, t_j)} \log \frac{n}{c(t_j)}$$

$$nbow(d_j)_i = \frac{bow(d_j)_i}{\sum_i bow(d_j)_i} = \frac{c(d_j, t_i)}{\sum_i c(d_j, t_i)} = tf(d_j, t_i)$$

Bag of Words – TF-IDF

Term Frequency – Inverse Document Frequency

weigh words depending on how informative they are

$$bow(d_i)_j = tf(d_i, t_j) \cdot idf(t_j) = \frac{c(d_i, t_j)}{\sum_j c(d_i, t_j)} \log \frac{n}{c(t_j)}$$

$$nbow(d_j)_i = \frac{bow(d_j)_i}{\sum_i bow(d_j)_i} = \frac{c(d_j, t_i)}{\sum_i c(d_j, t_i)} = tf(d_j, t_i)$$

$$idf(t_j) = \log \frac{n}{c(t_j)}$$

Bag of Words – TF-IDF

Term Frequency – Inverse Document Frequency

Example

- D1: The cat is on the mat.
- D2: My dog and cat are the best.
- D3: The locals are playing.

- D1: cat mat
- D2: dog cat best
- D3: locals playing

V = [cat, mat, dog, best, locals, playing]

Bag of Words – TF-IDF

Term Frequency – Inverse Document Frequency

Example

- **TF(D1)** = [1/2, 1/2, 0/2, 0/2, 0/2, 0/2] = [0.5, 0.5, 0, 0, 0, 0]
- **TF(D2)** = [1/3, 0/3, 1/3, 1/3, 0/3, 0/3] = [0.33, 0, 0.33, 0.33, 0, 0]
- **TF(D3)** = [0/2, 0/2, 0/2, 0/2, 1/2, 1/2] = [0, 0, 0 ,0 ,0.5 ,0.5]

- **IDF(cat)** = log(3/2) = 0.18
- **IDF(mat)** = log(3/1) = 0.48
- **IDF(dog)** = log(3/1) = 0.48
- **IDF(best)** = log(3/1) = 0.48
- **IDF(locals)** = log(3/1) = 0.48
- **IDF(playing)** = log(3/1) = 0.48

- **TF-IDF(D1)** = [0.5 * 0.18 ,0.5 * 0.48 ,0 * 0.48 ,0 * 0.48 ,0 * 0.48 ,0 * 0.48] = [0.09 ,0.24 ,0 ,0 ,0 ,0]
- **TF-IDF(D2)** = [0.33 * 0.18 ,0 * 0.48 ,0.33 * 0.48 ,0.33 * 0.48 ,0 * 0.48 ,0 * 0.48] = [0.06 ,0 ,0.16 ,0.16 ,0 ,0]
- **TF-IDF(D3)** = [0 * 0.18 ,0 * 0.48 ,0 * 0.48 ,0 * 0.48 ,0.5 * 0.48 ,0.5 * 0.48] = [0,0 ,0 ,0 ,0.24 ,0.24]

References

- https://medium.com/factory-mind/regex-tutorial-a-simple-cheatsheet-by-examples-649dc1c3f285
- https://regex101.com/
- https://text-processing.com/demo/tokenize/