WARWICK BUSINESS SCHOOL
THE UNIVERSITY OF WARWICK

For the Open Minded

Ram Gopal

2023-2024

Text Preprocessing & Data Structures

IB9CW0 Text Analytics

Text Data

- Text(ual) data have a unique **challenge** from the perspective that they require the handling of text fields individually.

- That means that for each text field a separate process needs to be instantiated.

- Cleaning the text data before further analysis is crucial.

Text Cleaning steps in NLP

Depending on the context you may also want to do some of the following:

- **Typing errors:** You cannot expect that text that is supplied from humans do not contain grammatical and syntax mistakes in some parts.
- **White space handling:** Some text may contain more than one space in between words, this can have adverse results in subsequent analysis.
- **Word elongation:** Replacing forms of text using predefined words.
  - In informal writing people may use a form of text embellishment to emphasize or alter word meanings called elongation (a.k.a. “word lengthening”).
  - For example, the use of “Whyyyyy” conveys frustration. **This can be converted to** *:* Why
- **Contraction handling**: Some common phrases are contracted and need to be replaced with multi-word formats or vice-versa.
  - e.g., NLP can be transformed to Natural Language Processing.
- **Handling of symbols, numbers and time:** Sometimes it is important to address cases such as the writing of numbers: e.g., I bought it for five dollars
  - This can also be converted to : I bought it for $5 (or vice versa).

Text Data Structures

Corpus: A collection of documents which can be grouped together either by the same content or the same author or target object.

Corpus

Document: A collection of text representing a single instance of an entity (e.g., a complaint document).

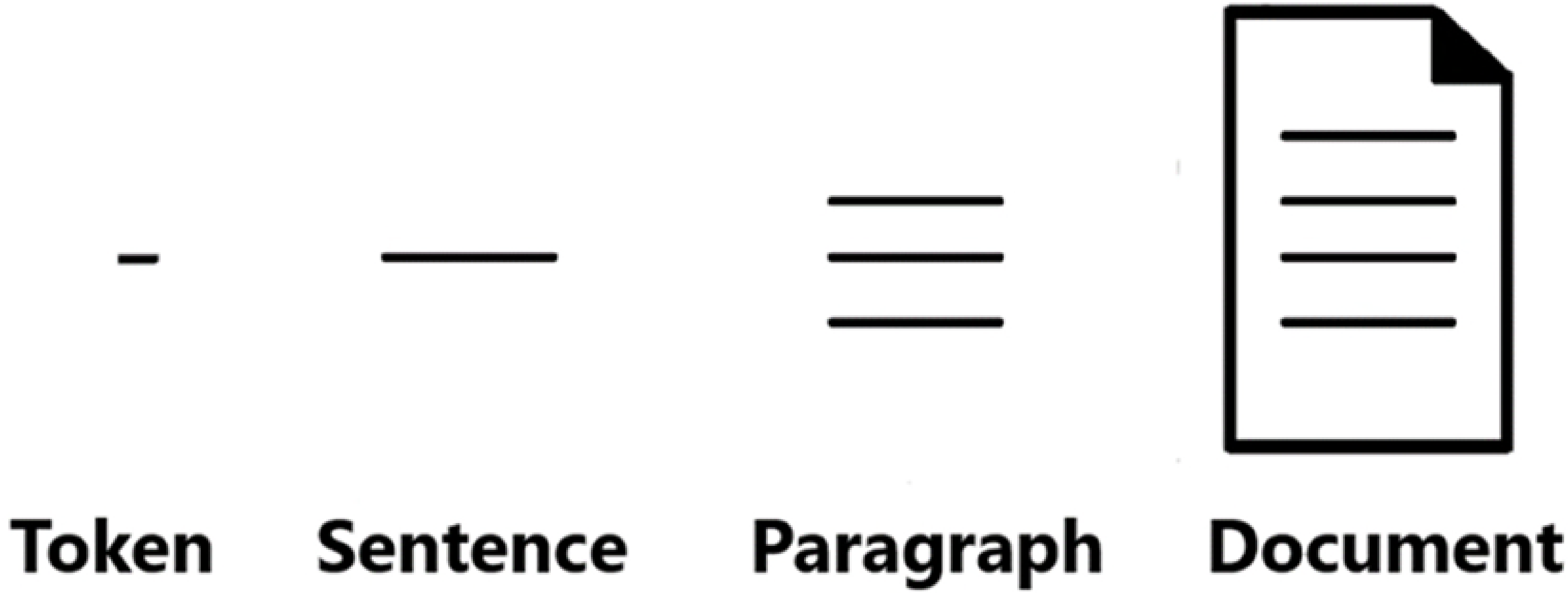
Document

Documents and Text

- *Documents* describe any collection of textual information that can be attributed to an
  - author
  - entity (e.g., company)
  - collection (multiple documents of the same entity)
- Documents do contain *metadata* which can be easily binded together either from properties of the file itself (if is on the file system) or by the additional fields/columns that are supplied on the database table.

Constituents of a Document

- **Words** - also known as **tokens.**
- **Sentences** - multiple tokens combined with syntax and punctuation marks.
- **Paragraphs** - A collection of sentences.



Tokens

- **n-grams :** Represent combination of words and these are used for **phrase extraction**.
  - When n=1 then we have a **unigram** (one word phrase)
  - When n=2 then we have a **bigram** (two word phrase)
  - When n=3 / 4/ 5 etc. then we have a **trigram/quadragram/pentagram etc.** (three, four, five word phrases)

NLP Pipelines

- The process of establishing a **staging** process for analyzing a field which contains **natural language** (language that has been provided by human subjects) is called **natural language processing.**
- The process involves many different steps where the output of one step is supplied as an input to the other.

Components of NLP

- **Tokenization**: Also known as word segmentation. The most common task in dealing with a document.
  - Individual words (**unigrams**)
  - Word combinations (e.g., first and second, second and third etc.). This is also called **n-gram tokenization**.
  - **Sentence segmentation** - Similar with tokenization but involves splitting using sentence dividers (., ?, ! etc.). This is also known as phrase extraction.

- **Stop-word removal**: This is an essential step as it involves removing words that are very common in the text and do not add any meaning (e.g., the, to, a, an, in, at, etc.).

- **Stemming** and **Lemmatization**

- **Part-of-speech (POS) tagging**: Involves identifying the part of speech for each word. As we are going to see later - POS tagging is an essential requirement for statistical language processing.
  - An immediate step is the filtering of the text to keep only those parts of speech that carry high level of semantic information.
  - For indo-european languages these are: **nouns, adjectives and adverbs.**

Tokenization

- Tokenization is easily carried using a split modifier.
- For natural language this is the space.

Consider for example the following text:

```
text <- "the quick brown fox came to eat its lunch"
```

Tokenization will produce a vector of words as:

```
c("the","quick","brown","fox","came","to","eat","its","lunch“)
```

Stop-words

- Stop-words are words that from non-linguistic view do not carry information
  - They have mainly functional role
  - Usually, we remove them to help the methods to perform better

- Stop words are language dependent – examples:
  - English: A, ABOUT, ABOVE, ACROSS, AFTER, AGAIN, AGAINST, ALL, ALMOST, ALONE, ALONG, ALREADY, ...
  - Dutch: de, en, van, ik, te, dat, die, in, een, hij, het, niet, zijn, is, was, op, aan, met, als, voor, had, er, maar, om, hem, dan, zou, of, wat, mijn, men, dit, zo, ...
  - Slovenian: A, AH, AHA, ALI, AMPAK, BAJE, BODISI, BOJDA, BRŽKONE, BRŽČAS, BREZ, CELO, DA, DO, ...

Stemming

- Different forms of the same word are usually problematic for text data analysis, because they have different spelling and similar meaning (e.g., learns, learned, learning,…). These are called **inflections**.

- Stemming is a process of transforming a word into its stem (normalized form)
  - Stemming provides an inexpensive mechanism to merge

Stemming

- For English mostly used Porter stemmer at http://www.tartarus.org/~martin/PorterStemmer/
- Example cascade rules used in English Porter stemmer

Stemming	Example
ATIONAL -> ATE	relational -> relate
TIONAL -> TION	conditional -> condition
ENCI -> ENCE	valenci -> valence
ANCI -> ANCE	hesitanci -> hesitance
IZER -> IZE	digitizer -> digitize
ABLI -> ABLE	conformabli -> conformable
ALLI -> AL	radicalli -> radical
ENTLI -> ENT	differentli -> different
ELI -> E	vileli -> vile
OUSLI -> OUS	analogousli -> analogous

Simplified Porter Stemmer

Step 1a

sses	→	ss	caresses	→	caress
ies	→	i	ponies	→	poni
ss	→	ss	caress	→	caress
s	→	ø	cats	→	cat

Step 1b

(*v*)ing	→	ø	walking	→	walk
			sing	→	sing
(*v*)ed	→	ø	plastered	→	plaster

Simplified Porter Stemmer

Step 2 (for long stems)

```
ational→ ate   relational→ relate
izer→ ize      digitizer → digitize
ator→ ate      operator  → operate
```

Step 3 (for longer stems)

```
al     → ø  revival    → reviv
able   → ø  adjustable → adjust
ate    → ø  activate   → activ
...
```

Lemmatization

Lemmatization considers the root form of the word that is in question. In other words, it can be defined as the act of *"grouping together the inflected forms of a word so they can be analyzed as a single item"*

For example, consider the following forms of the *to be* verb.

```
bw <- c('are', 'am', 'being', 'been', 'be')
```

The output will be:

```
## [1] "be" "be" "be" "be" "be"
```

Text mining data structures (matrix)

- The **document-term matrix:** provides a matrix where the **presence (boolean 1,0)** of terms is recorded for each document (for each term we have a membership if present in a document).
- **The document-term frequency matrix:** provides a matrix where the **frequency** of terms is recorded for each document (for each term we have a membership if present in a document)
- The **term document matrix:** same as the above but documents are represented in rows instead of columns.

Process of creating DTM (document-term matrix)

document	text
1	good product. Like it
2	product ready to use. use easy peasy peasy peasy
3	product of like work
4	not sure about product use

Step 1: Tokenize

document	text
1	good product. Like it
2	product ready to use. use easy peasy peasy peasy
3	product of like work
4	not sure about product use

document	token
1	good
1	product
1	like
1	it
2	product
2	ready
2	to
2	use
2	use
2	easy
2	peasy
2	peasy
2	peasy
3	product
3	of
3	like
3	work
4	not
4	sure
4	about
4	product
4	use

Step 2: Group documents at token level

document	token
1	good
1	product
1	like
1	it
2	product
2	ready
2	to
2	use
2	use
2	easy
2	peasy
2	peasy
2	peasy
3	product
3	of
3	like
3	work
4	not
4	sure
4	about
4	product
4	use

document	word	count
1	good	1
1	it	1
1	like	1
1	product	1
2	peasy	3
2	use	2
2	easy	1
2	product	1
2	ready	1
2	to	1
3	like	1
3	of	1
3	product	1
3	work	1
4	about	1
4	not	1
4	product	1
4	sure	1
4	use	1

Step 3: Create DTM (document-term matrix) of counts

document	word	count
1	good	1
1	it	1
1	like	1
1	product	1
2	peasy	3
2	use	2
2	easy	1
2	product	1
2	ready	1
2	to	1
3	like	1
3	of	1
3	product	1
3	work	1
4	about	1
4	not	1
4	product	1
4	sure	1
4	use	1

document	peasy	use	good	it	like	product	easy	ready	to	of	work	about	not	sure
1	0	0	1	1	1	1	0	0	0	0	0	0	0	0
2	3	2	0	0	0	1	1	1	1	0	0	0	0	0
3	0	0	0	0	1	1	0	0	0	1	1	0	0	0
4	0	1	0	0	0	1	0	0	0	0	0	1	1	1

DTM (document-term matrix) of counts

document	peasy	use	good	it	like	product	easy	ready	to	of	work	about	not	sure
1			1	1	1	1								
2	3	2				1	1	1	1					
3					1	1				1	1			
4		1				1						1	1	1

document	text
1	good product. Like it
2	product ready to use. use easy peasy peasy peasy
3	product of like work
4	not sure about product use

What is the challenge here?

Length bias

Documents that are longer will always have high term counts

Solution

Use term frequencies instead of counts

Term Frequency

- Let $freq(t, d)$ denote the frequency (e.g., raw count) of term t in document d
- Let $TF(t, d)$ denote the proportion of term t in document d
- TF can be then calculated by weighting the document term matrix with the total words for this document containing n terms.

$$TF = \frac{freq(t, d)}{\sum_i^n (freq(t_i, d)}$$

DTM of counts

document	peasy	use	good	it	like	product	easy	ready	to	of	work	about	not	sure
1			1	1	1	1								
2	3	2				1	1	1	1					
3					1	1				1	1			
4		1				1						1	1	1

DTM of tf (term frequencies)

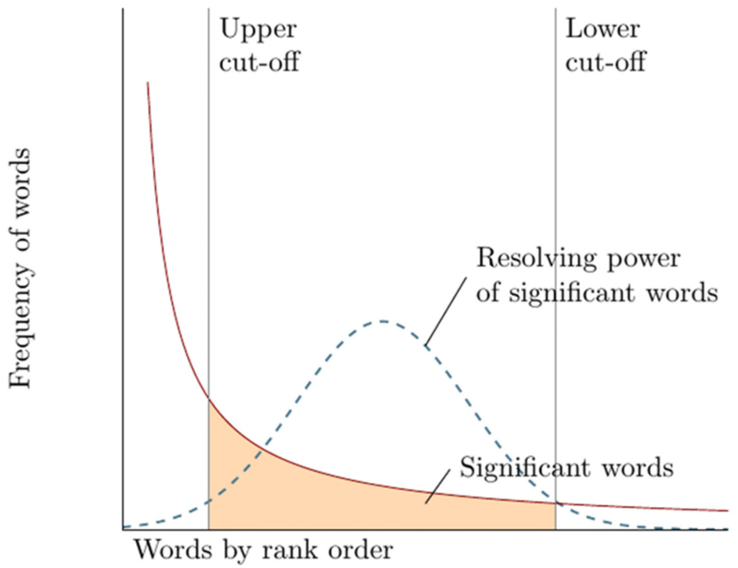
document	peasy	use	good	it	like	product	easy	ready	to	of	work	about	not	sure
1			0.25	0.25	0.25	0.25								
2	0.33	0.22				0.11	0.11	0.11	0.11					
3					0.25	0.25				0.25	0.25			
4		0.2				0.2						0.2	0.2	0.2

Product appears in every document thus it does not provide much insights

Luhn’s method

Luhn proposed that the significance of each word in a document signifies how important it is. The idea is that any sentence with maximum occurrences of the highest frequency words(stop-words) and least occurrences are not important to the meaning of document than others.

1. Too low frequent words are not significant
2. Too high frequent words are also not significant (e.g. “is”, “and”)
3. Removing low frequent words is easy
    1. set a minimum frequency-threshold
4. Removing common (high frequent) words:
    1. Setting a maximum frequency threshold (statistically obtained)
    2. Comparing to a common-word list



Inverse document frequency (IDF)

- Let N be the total number of documents in the corpus.
- Let $count(t)$ be the total number of documents where the term t is present

$$IDF(t) = \ln\left(\frac{N}{count(t)}\right)$$

- Understandably when the number of documents containing the term goes towards the total number of documents in the corpus then $lim\big(IDF(t)\big) \rightarrow 0$
- In other words the more popular the term, the more is penalized.

TF/IDF combined

- By combining Term frequency with the inverse document frequency

$$TF/IDF(t,d) = TF(t,d) * IDF(t,d)$$

- With that approach we can replace the counts of the term frequency on the document term matrix with the TF/IDF score

DTM of tf (term frequencies)

document	peasy	use	good	it	like	product	easy	ready	to	of	work	about	not	sure
1			0.25	0.25	0.25	0.25								
2	0.33	0.22				0.11	0.11	0.11	0.11					
3					0.25	0.25				0.25	0.25			
4		0.2				0.2						0.2	0.2	0.2

DTM of tf-idf

document	peasy	use	good	it	like	product	easy	ready	to	of	work	about	not	sure
1			0.347	0.347	0.173									
2	0.462	0.154					0.154	0.154	0.154					
3					0.173					0.347	0.347			
4		0.139										0.277	0.277	0.277

purpose: to further condense the number of words and only keep words that add value. eg product appears in many document so it is not use

From text to numbers

document	text
1	good product. Like it
2	product ready to use. use easy peasy peasy peasy
3	product of like work
4	not sure about product use

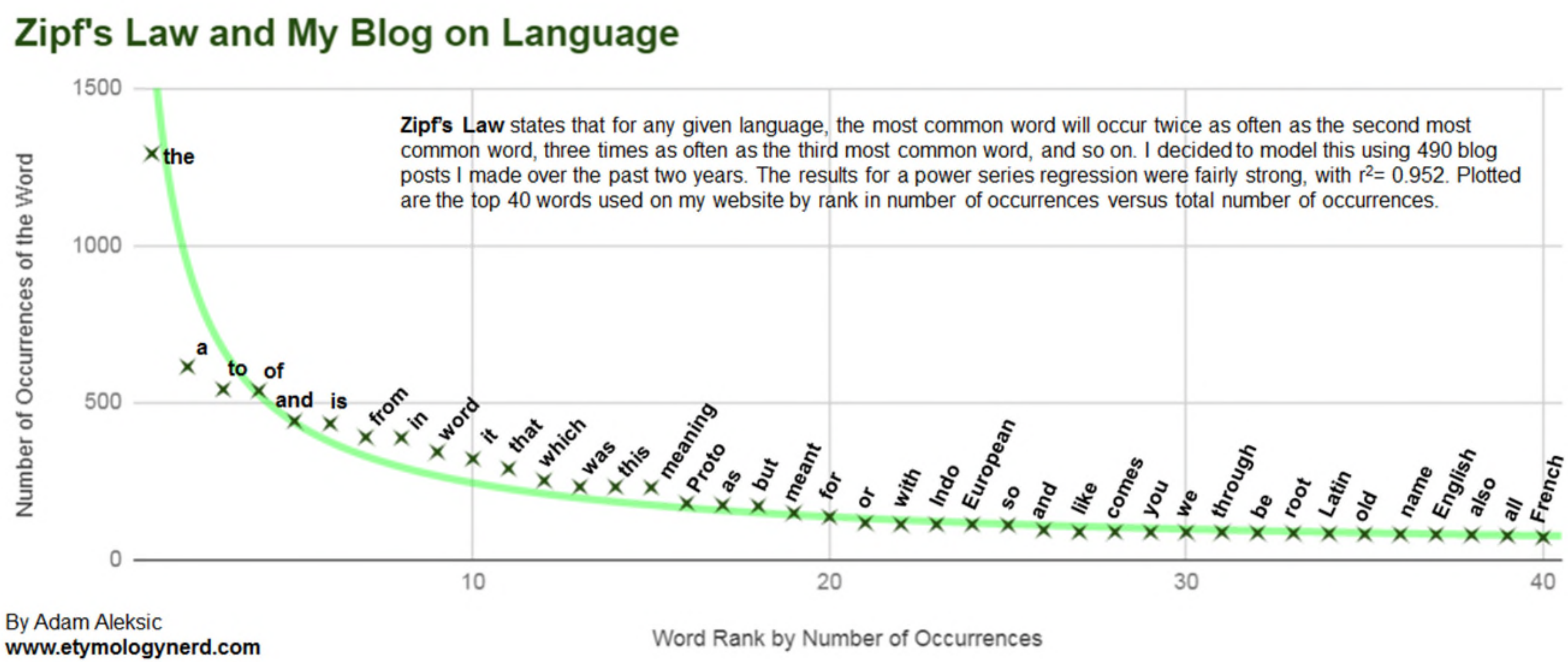
DTM of tf-idf

document	peasy	use	good	it	like	product	easy	ready	to	of	work	about	not	sure
1			0.347	0.347	0.173									
2	0.462	0.154					0.154	0.154	0.154					
3					0.173					0.347	0.347			
4		0.139										0.277	0.277	0.277

With DTM, we can now formally analyze text data – importance of words, association between words, clustering documents, sentiments, etc.

Zipf's Law

Given a large sample of words used, the frequency of any word is inversely proportional to its rank in the document-term matrix. So word number *n* has a frequency proportional to 1/*n*.



Google n-gram Corpus

- In September 2006 Google announced availability of n-gram corpus:
  - http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html#links
  - Some statistics of the corpus:
    - File sizes: approx. 24 GB compressed (gzip'ed) text files
    - Number of tokens: 1,024,908,267,229
    - Number of sentences: 95,119,665,584
    - Number of unigrams: 13,588,391
    - Number of bigrams: 314,843,401
    - Number of trigrams: 977,069,902
    - Number of fourgrams: 1,313,818,354
    - Number of fivegrams: 1,176,470,663

https://books.google.com/ngrams/graph?content=Hi&year_start=1800&year_end=2019&corpus=26&smoothing=3&case_insensitive=true

Google n-grams

ceramics collectables collectibles 55
ceramics collectables fine 130
ceramics collected by 52
ceramics collectible pottery 50
ceramics collectibles cooking 45
ceramics collection , 144
ceramics collection . 247
ceramics collection </S> 120
ceramics collection and 43
ceramics collection at 52
ceramics collection is 68
ceramics collection of 76
ceramics collection | 59
ceramics collections , 66
ceramics collections . 60
ceramics combined with 46
ceramics come from 69
ceramics comes from 660
ceramics community , 109
ceramics community . 212
ceramics community for 61
ceramics companies . 53
ceramics companies consultants 173
ceramics company ! 4432
ceramics company , 133
ceramics company . 92
ceramics company </S> 41
ceramics company facing 145
ceramics company in 181
ceramics company started 137
ceramics company that 87
ceramics component (76
ceramics composed of 85

serve as the incoming 92
serve as the incubator 99
serve as the independent 794
serve as the index 223
serve as the indication 72
serve as the indicator 120
serve as the indicators 45
serve as the indispensable 111
serve as the indispensible 40
serve as the individual 234
serve as the industrial 52
serve as the industry 607
serve as the info 42
serve as the informal 102
serve as the information 838
serve as the informational 41
serve as the infrastructure 500
serve as the initial 5331
serve as the initiating 125
serve as the initiation 63
serve as the initiator 81
serve as the injector 56
serve as the inlet 41
serve as the inner 87
serve as the input 1323
serve as the inputs 189
serve as the insertion 49
serve as the insourced 67
serve as the inspection 43
serve as the inspector 66
serve as the inspiration 1390
serve as the installation 136
serve as the institute 187