Analysis of text data

Extracting information from natural text

Learning goals

1. Understand the role and concept of text analytics
2. Learn the basic workflow in processing natural text for ML purposes.
3. Understand the potential uses of text analytics.

Text analytics

ve all expansions and have consid
ra dividers to go along with a coup
ning insert but would need to purc
) addition dividers! This would cost
I'm wondering if the rigid, fixed di
t great for going in between every
d type since you can't scoot them
ical card stock-like dividers and, ev
se dividers are clear acrylic, it migh
kingdom cards with so many of th
d in place so close together since t

vs.

ID	Likes_travel	Likes_nature	Likes_bars_clubs	Income	Gender
1	9	5	4	3176	F
2	9	5	6	2633	F
3	10	0	2	4180	M
4	3	2	5	2229	F
5	6	5	5	3089	F
6	7	2	5	2197	F
7	1	1	5	4383	M
8	9	5	10	2213	F
9	0	5	4	3847	F
10	2	7	3	4183	M
11	10	7	7	4809	M
12	5	4	5	2087	M

- Finding previously unknown information from text masses in an automated way.
- The challenge arises from the format of text data.
  - Words, sentences and paragraphs vs. variables and values.

Popular examples

- Word cloud
  - https://worditout.com/word-cloud/create
- Trending vocabulary in social media
  - https://www.trendsmap.com/

”Web mining”

1. Web content mining
   – Parsing HTML documents and analyzing their content
2. Web structure mining
   – Analysis of the structure of the hyperlink network
   – Analogy to the analysis of social media networks.
3. Web usage mining
   – Analysis of log files.

Web content mining in Python

```
from bs4 import BeautifulSoup
from urllib.request import urlopen

url = "https://www.nngroup.com"
page = urlopen(url)
soup = BeautifulSoup(page, 'html.parser')
print(soup.get_text())
```

- In Python, **BeautifulSoup** library contains web scraping functionality.
- However, API access is preferable to ’scraping’.

Focus

- In this session we focus in the techniques of text analytics.
- There are also special challenges related to data aquisiton
    - Largerdata size
    - Need suitable access methods (web scraping vs. API access)
    - Need for preprocessing (e.g. machine-generated ’messy’ HTML format)
    - Dynamic nature of data
- In the realm of deep neural networks, **word2vec** models can be used for recommending/prediction.
    - In this presentation, we focus on the ’shallow’ word-count based approach for analysis of text data.

Idea of text analytics

- The text data is first transformed into tabular format.
  - The variables represent the (postprocessed) number of occurences of each word in each text.
- Once the data is in tabular, numeric format, it can be analyzed
  - using ’traditional’ machine learning methods.
  - … or specific text analytics techniques.

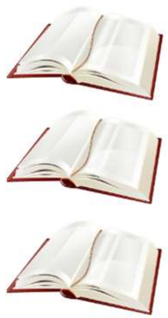
Fields of use

- Sentiment analysis
- Language recognition
- Fraud identification
- Spam email filtering
- Automatization of customer service
- Making summaries
- Generation of metadata
- Recognizing intention (chatbot)
- Finding emergent trends

Text preprocessing

- Let’s get acquainted with the basic text preprocessing pipeline.
- Goals:
  1. Perform positional tagging: analyse the role of each word in a sentence and find patterns.
  2. Carry out sentiment analysis: evaluate the positive / negative values of words.

Text processing



Row No.	aamir	aaron	aaronah	aback	abandon	abbott	abdulrashidh	aber	aberlour	abhishek
1	0	0	2	0	5	0	0	1	1	0
2	1	106	0	1	1	1	1	0	0	1
3	0	0	0	0	2	0	0	0	0	0

- The purpose of text processing phase is to transform the texts into tabular format.
- Each word in text turns into a new variable.
- The value of the variable is the frequency (i.e. number of occurences) of the word in the data set.
  - In the example, there are three documents. Each one produces an observation (a row) into the data set.
- The variable derived from the word, together with its values, is called a word vector.
- It is possible to derive the word vectors for group of subsequent words instead of individual words.
  - These are called n-grams.
  - In practice, only 2-grams are considered.

Text processing

- The processing step is comprised of four phases:
  1. Transform case.
     - Make capital and lower-case words to be treated uniformly.
  2. Tokenize.
     - Find individual words (or n-grams).
  3. Filter stopwords.
     - Ignore common words with little value to the analysis.
  4. Stem/Lemmatize
     - Merge variants of the same word.
- Let's pay further attention into these steps.

Phase 1: Transform case

This is a nice sentence. This sentence is even nicer.

⇩

this is a nice sentence. this sentence is even nicer.

- The distinction between capital and lower-case letters is removed.
- As a consequence, it will make no difference whether the word appears in the beginning of a sentence or not.

Phase 2: Tokenize

this is a nice sentence. this sentence is even nicer.

⇩

this	is	a	nice	sentence	even	nicer
2	2	1	1	2	1	1

- In this phase, each word is turned into a variable.
- The punctuation marks are removed at this point.
- Instead of individual words, 2-grams could be considered.

Phase 3: Filtering stop words

this	is	a	nice	sentence	even	nicer
2	2	1	1	2	1	1

nice	sentence	nicer
1	2	1

- Stopwords are common words that occur frequently in all texts.
- They are not considered interesting for text mining. Thus, they are often removed.
- A stopword list is required.

Phase 4: Stemming and/or lemmatization

nice	sentence	nicer
1	2	1

nice*	sentence*
2	2

- Depending on the language, words can have multiple inflected forms.
- It is not feasible to consider forms of the same word as different words.
  - The problem is most evident in agglutinative languages such as Finnish.
- The merger of inflected words can be done in multiple ways:
  - The words can be replaced by a shorter substring that is free from inflection (see image above).

Stemming vs. lemmatization

grade grading graded

STEMMING *LEMMATIZATION*

grad* **grade**

- Stemming finds the common root of the word.
  - Not necessarily a proper word.
- Lemmatization generates the dictionary form of the word.

What about synonyms?

- Words with multiple meanings may ’dilute’ the data.
- This problem is usually ignored.
- Rationale: in Big Data context, the slight degradation of data cause by synonyms is compensated by a large data size.

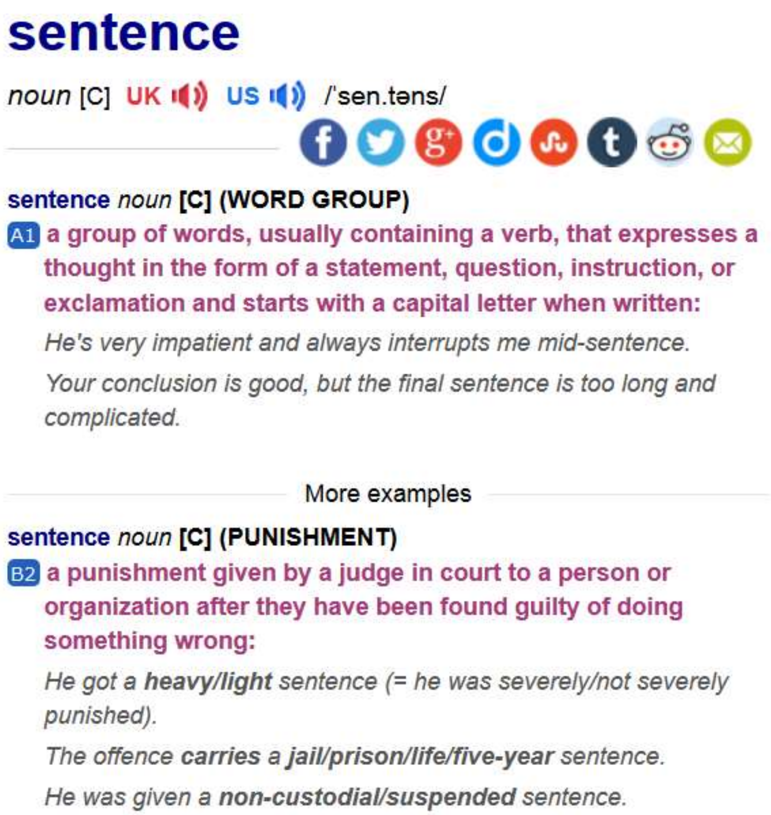


Image: Cambridge dictionaries online.

PoS (Part of Speech) tagging

- The idea of PoS tagging
  1. Recognize the role of each word in the text, and attach a tag
- Rationale for use:
  – Make it possible to look for patterns of tags (usually expressed as RegExps or regular expressions).
    - This is called chunking. It allows creation of derived variables with positional information.
  – Provide context information for better lemmatization.
- For a list of PoS tags in Python NLTK package, see: https://www.guru99.com/pos-tagging-chunking-nltk.html

TF-IDF

Word/Document	Word 1	Word 2	Word 3	Word 4	Word 5
Document 1		4	5		
Document 2	5	2			1
Document 3		3	3		5
Document 4	3	3	1		4
Document 5			4		
Document 6	1	4		3	

Word/Document	Word 1	Word 2	Word 3	Word 4	Word 5
Document 1	0,000	0,035	0,098	0,000	0,000
Document 2	0,188	0,020	0,000	0,000	0,038
Document 3	0,000	0,022	0,048	0,000	0,137
Document 4	0,082	0,022	0,016	0,000	0,109
Document 5	0,000	0,000	0,176	0,000	0,000
Document 6	0,038	0,040	0,000	0,292	0,000

- The absolute term frequencies are usually replaced by measures taking into account:
  1. The length of each document
  2. The rareness of the word in the entire corpus.
- TF-IDF is a commonly used transformation for that.
  - Several weighting schemes, see e.g. https://en.wikipedia.org/wiki/Tf%E2%80%93idf

TF-IDF

							TF					TF-IDF				
Word/Document	**Word 1**	**Word 2**	**Word 3**	**Word 4**	**Word 5**	**Sum_f**	**Word 1**	**Word 2**	**Word 3**	**Word 4**	**Word 5**	**Word 1**	**Word 2**	**Word 3**	**Word 4**	**Wor**
Document 1		4	5			9	0,000	0,444	0,556	0,000	0,000	0,000	0,035	0,098	0,000	0,0
Document 2	5	2			1	8	0,625	0,250	0,000	0,000	0,125	0,188	0,020	0,000	0,000	0,0
Document 3		3	3		5	11	0,000	0,273	0,273	0,000	0,455	0,000	0,022	0,048	0,000	0,1
Document 4	3	3	1		4	11	0,273	0,273	0,091	0,000	0,364	0,082	0,022	0,016	0,000	0,1
Document 5			4			4	0,000	0,000	1,000	0,000	0,000	0,000	0,000	0,176	0,000	0,0
Document 6	1	4		3		8	0,125	0,500	0,000	0,375	0,000	0,038	0,040	0,000	0,292	0,0
Total	9	16	13	3	10	51	3	5	4	1	3					
							0,3	0,08	0,18	0,78	0,3	IDF				

Analysis possibilities

Row No.	a	abid	abound	abund	access	acclaim
1	0	0	1	0	0	0
2	0	2	1	1	1	7
3	0	0	0	1	0	0

- Once the word vectors have been produced, ’traditional’ ML methods can be applied:
  - Clustering: which documents are similar in vocabulary?
    - Finding an unknown author.
  - Association analysis: which words are used together?
    - Gathering domain-specific vocabularies.
  - Decision tree / neural network
    - Automatic classification of customer feedback (angry complaint, error report, suggestion etc.)
    - An learning set is required.

Text analytics in Python

- In the course's Oma workspace, see **Documents/Methods/Data/Texts (demo)**.
  - PoS tagging & chunking
  - Sentiment analysis
  - Simple word-based clustering
- The text analytics functionality is available in the NLTK package.
  - Technical note: various instructions suggest entering a **`nltk.download()`** command to load all models. Should this fail, try **`nltk.download('popular')`**.

```
# conda install nltk
import nltk
nltk.download('popular')
```