Text Data Processing and Preparation

SOFIA UNIVERSITY
ST. KLIMENT OHRIDSKI
EST. 1888

FACULTY
OF ECONOMICS
AND BUSINESS
ADMINISTRATION

What you will learn

- Structuring text data – the concept of "tokenization".
- Main techniques for text data processing – case normalization, special characters removal, stemming, lemmatization etc.
- How to apply all these techniques in Python?
- Useful Python libraries for exploratory text data analysis and text data processing.

1.

Text Data Processing

Common techniques for text data structuring, cleaning and normalization

Credits: http://solutions.ait.ac.th/garbage-in-garbage-out/

Text Data Processing

- **Crucial** in any text mining project!
- This is Step 3 in the CRISP-DM model (**Data Preparation**).
- The **text processing pipeline is problem-specific** – it might include various techniques depending on the business case and data at hand.
- Text normalization **leads to**: 1. Dimension reduction; 2. A decrease in computational time; 3. Data with better quality.

Text Data Structuring - Tokenization

- **Tokenization** – the process of chopping up a given text into pieces, called **tokens**.
- **Tokens:** words, phrases, symbols, emoticons, punctuation etc.
- **Types of tokenization** – character/word/sentence/paragraph tokenization.
- **Word tokenization** is the most frequently applied type of tokenization.
- In classical text classification – individual words in the text are used as **explanatory variables**.

Word Tokenization (1)

- Example of word tokenization:

“I LOVE this movie !!! ♡ ♡ ♡”

[“I”, “LOVE”, “this”, “movie”, “!!!” , “♡”, “♡”, “♡”]

- The task highly depends on the text data language!
- Bulgarian and English - white space is used as a separator between words.
- What about Chinese? – “我喜歡這部電影♡♡♡”

Word Tokenization (2)

Important things to consider:

- **Is punctuation important**? You can remove punctuation or store it as a separate token.
- For text data in English – how to handle **words with apostrophes** during tokenization?

[“I won’t do my homework!”]

[“I”, “won’t”, “do”, “my”, “homework”,”!”]

[“I”, “won”, “t”, “do”, “my”, “homework”]

[“I”, “won t”, “do”, “my”, “homework”]

Word Tokenization (3)

More on how to handle contractions in English:

- https://github.com/kootenpv/contractions - a Python library for resolving contractions in English
- https://www.geeksforgeeks.org/nlp-expand-contractions-in-text-processing/ - how to use the “contractions“ library in Python
- https://stackoverflow.com/questions/19790188/expanding-english-language-contractions-in-python - general discussion on the topic…

Common text processing techniques (1)

- Most of the programming languages are **case-sensitive**.
- Python will treat “Love” and “love” as different words (tokens).
- The answer: **Case normalization**
- Use **lowercase()** in Python:

“I LOVE this movie !!! ♡ ♡ ♡”

[“i”, “love”, “this”, “movie”, “!!!”, “♡”, “♡”, “♡”]

Common text processing techniques (2)

- **Special characters removal** – depending on your project this removal might include punctuation, digits, symbols etc.
- Use **isalpha**() in Python.
- **Regex?**
- **NB:** if you want to use the emoticons in further analyses, extract them before special characters removal!

→ "I LOVE this movie !!! ♡ ♡ ♡"

["i", "love", "this", "movie"]

Common text processing techniques (3)

- **Regular expressions (regex) -** a sequence of characters that specifies a search pattern in text.
- Regex have their **own syntax** and can be used in different programming languages (not only Python)!
- Regex might be very useful for text data cleaning.
- **Extremely complex logic** might be incorporated in your code by using regex.
- Very helpful tutorial on regex usage in Python - https://www.youtube.com/watch?v=K8L6KVGG-7o (use the library 're')
- Cheat sheet 1: https://www.rexegg.com/regex-quickstart.html
- Cheat sheet 2: https://www.dataquest.io/wp-content/uploads/2019/03/python-regular-expressions-cheat-sheet.pdf
- Test your regex here - https://regex101.com/

Common text processing techniques (4)

- Remove **html tags** if such exist!
- Use **BeautifulSoup** library in Python.
- Remove **URLs** or extract them for further usage (depending on your project).
- You can use **BeautifulSoup** or **URLextract** - https://urlextract.readthedocs.io/en/latest/urlextract.html

Common text processing techniques (5)

- Do you need Emoticons and Emoji?

  😂 😉 😋 😒 😭

- What’s the difference? - https://www.theguardian.com/technology/2015/feb/06/difference-between-emoji-and-emoticons-explained
- The **emoji** library in Python - emoji · PyPI
- The **emot** library in Python - https://github.com/NeelShah18/emot

Common text processing techniques (6)

- **Stop words removal**
- Usually as „stop words“ are treated words that frequently occur in the text and do not convey any useful information.
- The **list of stop words** depends on the text data language and the special characteristics of data at hand.
- Example of a “stop word” in English – “the”.
- Such words have **low predictive power** in text classification problems and **add noise** to data.
- **NB:** Be careful and always check the list of stop words before removing them!
- What about Bulgarian?

Common text processing techniques (7)

- **Stemming** - the process of reducing inflected words to their word stem, base or root form. No grammatical information is used during this process.

[“run”, “running”, “runs”]

⬇

[“run”]

- Stemming often produces words that do not exist. Example: (history, historical -> **histori**)
- Stemming is a **language-specific task**.
- There are useful tools for stemming in Python for both English and Bulgarian.

Common text processing techniques (8)

- **Lemmatization -** the process of grouping together the inflected forms of a word so they can be analyzed as a single item (“lemma”).
- Lemmatization is a **more advanced technique** than stemming since it uses grammatical information (such as part of speech tags) to determine the lemma.
- Example with the word “parking” in English.
- Lemmatization is **slower than stemming**!

Common text processing techniques (9)

- Stemming/Lemmatization leads to dimension reduction and might help in reducing the noise in textual data.
- **BUT be careful –** stemming/lemmatization might worsen the performance of machine learning models.
- My advice: **let the data speak for itself** – experiment with or without the application of stemming/lemmatization.

Part of speech tagging (POS tagging)

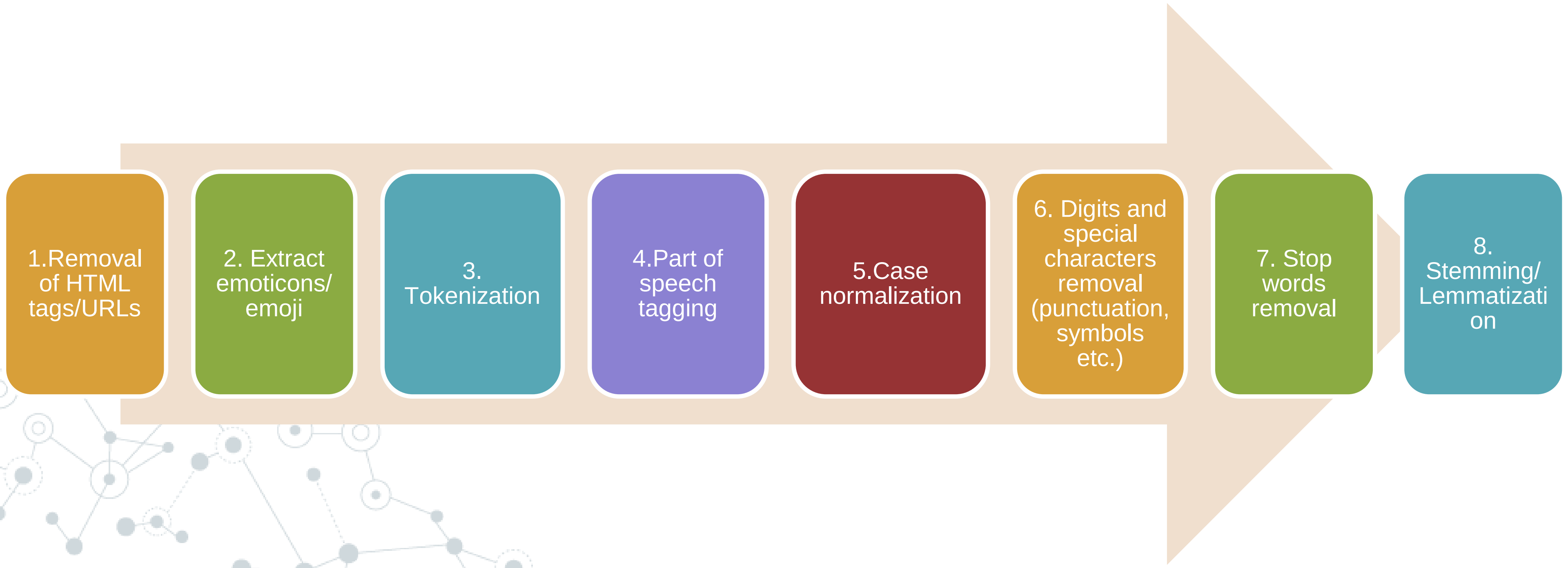
- **POS tagging** is the process of marking up a word in a text as corresponding to a particular part of speech, based on both its definition and its context.

Дума	Част на речта
Здравейте	междуметие
имам	глагол
въпрос	съществително име
за	предлог
конкретен	прилагателно име
ваш	местоимение
продукт	съществително име

- **NB:** Apply POS tagging before stop words removal.
- Lemmatization requires POS tagging.
- POS tagging might be **useful in feature selection** – in sentiment analysis, topic modeling etc.

NLP techniques - summary

- The combination of text processing techniques **always** depends on the project and data at hand.
- Example of a text processing pipeline:


1.Removal of HTML tags/URLs
2. Extract emoticons/ emoji
3. Tokenization
4.Part of speech tagging
5.Case normalization
6. Digits and special characters removal (punctuation, symbols etc.)
7. Stop words removal
8. Stemming/ Lemmatization

2. Useful Python Libraries for Text Data Exploration, Processing and Analysis

Useful Python Libraries (1)

- **NLTK** - https://www.nltk.org/
  - NLTK is one of the largest platforms for text data processing and analysis in Python.
  - Provides access to many linguistic resources and tools – for example, tokenization, stemming, lemmatization, part of speech tagging etc.
  - It is part of the Anaconda distribution! You don’t have to install it separately.
  - Documentation- https://www.nltk.org/py-modindex.html#

Useful Python Libraries (2)

- **Wordcloud** - create beautiful wordclouds.
  - A small library for text data visualizations.
  - Documentation and examples - http://amueller.github.io/word_cloud/

Alice
little
thing
Queen
time
went
one
head
Hatter
know
now
King
way
see
began
well
Gryphon
Duchess
thought
Mock Turtle
Mouse
Cat

Source: http://amueller.github.io/word_cloud/auto_examples/masked.html#sphx-glr-auto-examples-masked-py

Useful Python Libraries (3)

- **Beautiful Soup** - https://www.crummy.com/software/BeautifulSoup/bs4/doc/
  - Very useful library for cleaning text data scraped from the Internet – you can remove html tags and extract only the textual data in a given document.
  - Example: https://www.crummy.com/software/BeautifulSoup/bs4/doc/#quick-start

Useful Python Libraries (4)

- spaCy - https://spacy.io/
- gensim - https://radimrehurek.com/gensim/
- TextBlob - https://textblob.readthedocs.io/en/dev/
- polyglot - https://polyglot.readthedocs.io/en/latest/
- emot – https://github.com/NeelShah18/emot
- URLextract - https://urlextract.readthedocs.io/en/latest/urlextract.html
- contractions - https://github.com/kootenpv/contractions
- re – https://docs.python.org/3/library/re.html

If you want to learn more... 🤓

Check out the following textbooks:

- Miner et al., “**Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications**” - Chapter 3
- NLTK Book :
  - Chapter 1. Language Processing and Python
  - Chapter 3. Processing Raw Text

Thanks!

Any questions?

You can find me at:

g.hristova@feb.uni-sofia.bg