KI-Campus
Die Lernplattform
für Künstliche Intelligenz

Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

berlin

GEFÖRDERT VOM
Bundesministerium
für Bildung
und Forschung

Pre-processing

Salar Mohtaj | DFKI

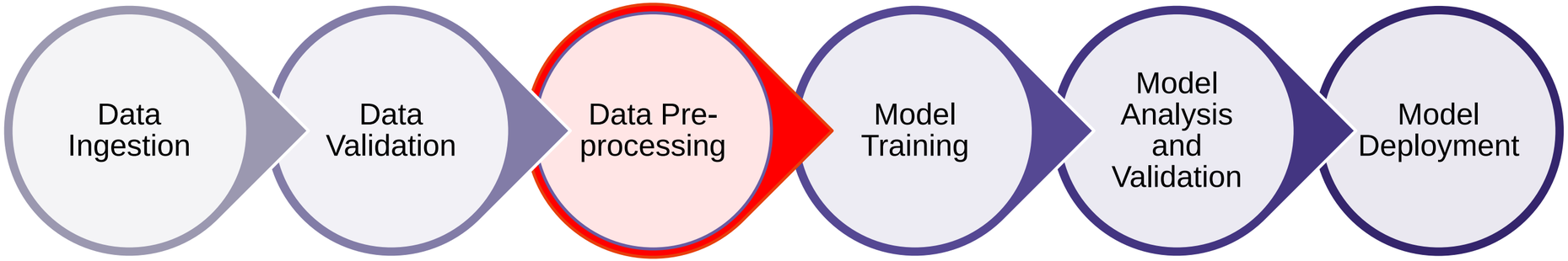
Text pre-processing

- What is text pre-processing?
- Why is it important?
- How to pre-process textual data?
- Python packages for text pre-processing

Text pre-processing

- What is text pre-processing?
- Why is it important?
- How to pre-process textual data?
- Python packages for text pre-processing

What is text pre-processing?



- To pre-process your text means to bring your text into a form that is **predictable** and **analyzable** for your task
- The steps and components are highly depend on the target task

Text pre-processing

- What is text pre-processing?
- Why is it important?
- How to pre-process textual data?
- Python packages for text pre-processing

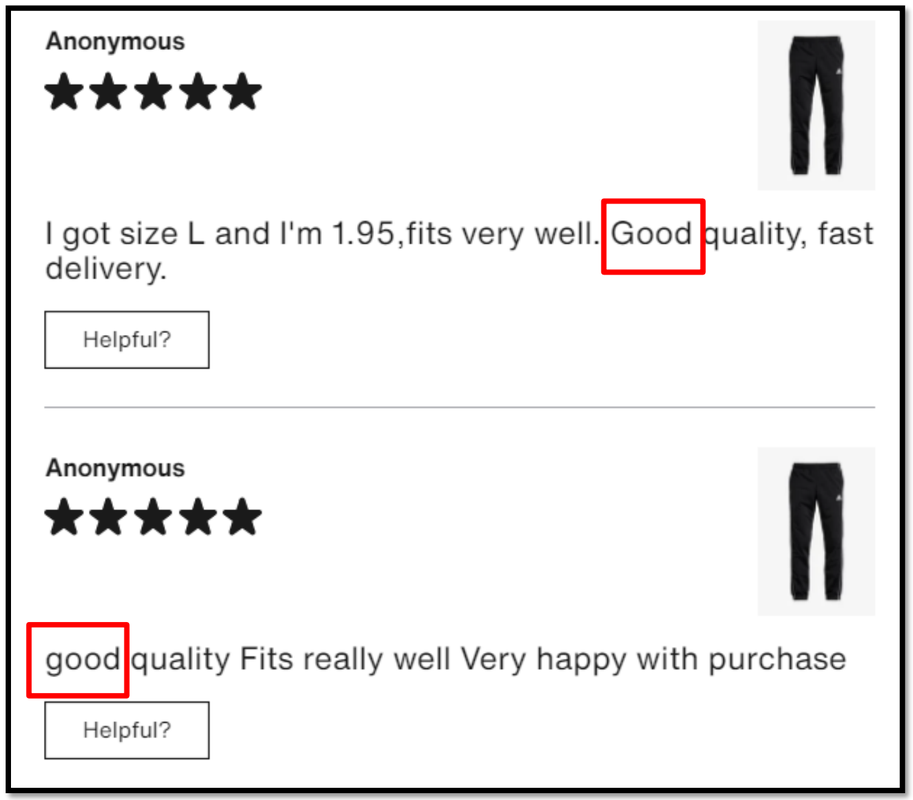
Why text pre-processing is important?

- It transforms text into a more digestible form so that machine learning algorithms can perform better
- It helps to get rid of unhelpful parts of the data (e.g., noises and outliers)
- Some experiments show simple text pre-processing steps could lead to significant improvement of final results, even in complex deep neural models

Why text pre-processing is important?

- To illustrate the importance of text preprocessing, let's consider a couple of customer reviews

- Good → 71 111 111 100
- good → 103 111 111 100



Text pre-processing

- What is text pre-processing?
- Why is it important?
- How to pre-process textual data?
- Python packages for text pre-processing

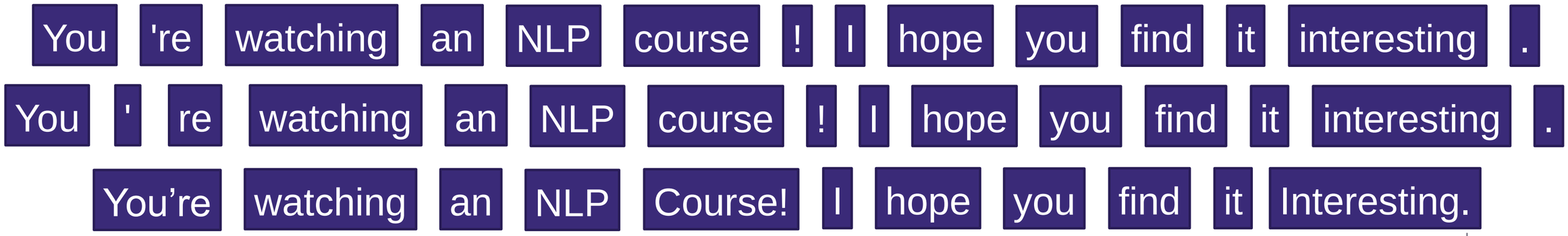
How to pre-process textual data?

- The most important pre-processing steps includes:
  - Tokenization and segmentation
  - Noise removal
  - Normalization

Tokenization / Segmentation

- **Tokenization** is essentially splitting a phrase, sentence, paragraph, or an entire text document into smaller units, such as **individual words or terms**
- Text **segmentation** is the process of dividing written text into meaningful units, such as words, **sentences**, or topics

You're watching an NLP course! I hope you find it interesting.



Tokenization / Segmentation

- Challenges
  - Multi token words (e.g., “New York”)
  - Continuous script languages

  - “.” doesn’t mean a sentence boundary in all sentences (segmentation)

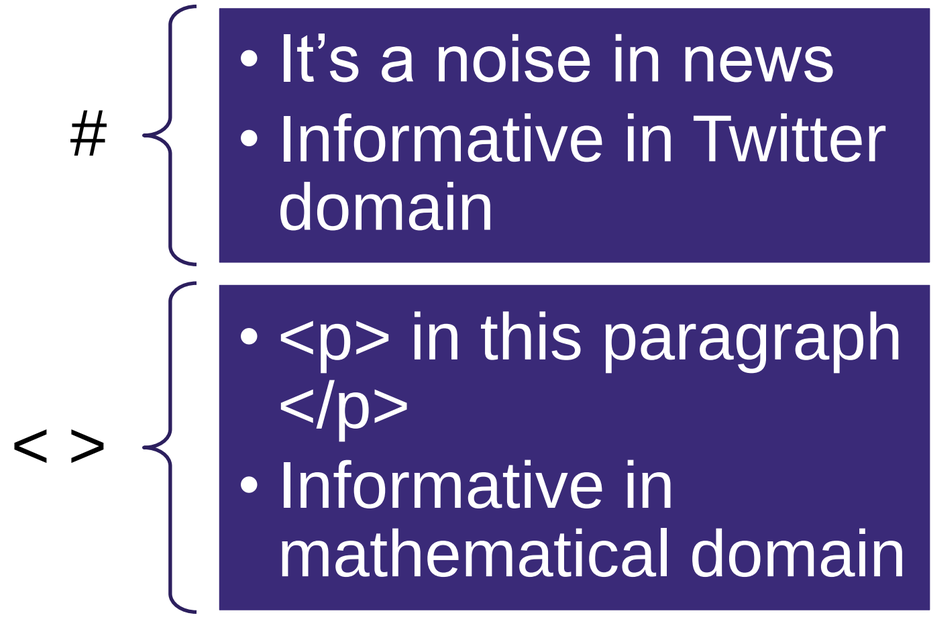
I am 1.75m tall, it was suit for me.

How to pre-process textual data?

- The most important pre-processing steps includes:
  - Tokenization and segmentation
  - Noise removal
  - Normalization

Noise removal

- ***Noise removal*** is about removing characters, digits and pieces of text that can interfere with your text analysis
- It’s highly domain dependent



How to pre-process textual data?

- The most important pre-processing steps includes:
  - Tokenization and segmentation
  - Noise removal
  - Normalization

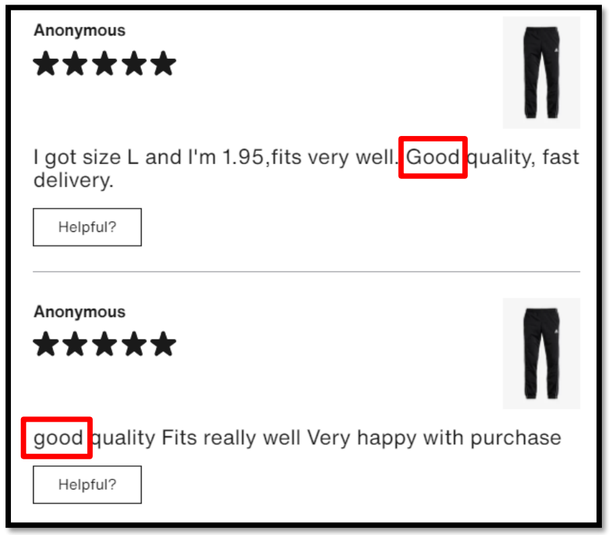
Normalization

- **Text normalization** is the process of transforming text into a single canonical form that it might not have had before
  - Lower casing
  - Removing punctuation
  - Removing / Converting numbers to their word equivalents
  - Strip white space
  - Removing stop words
  - Stemming / Lemmatization

Normalization

- **Text normalization** is the process of transforming text into a single canonical form that it might not have had before

  - Lower casing



I have been living in Berlin for 5 years. → i have been living in berlin for 5 years.

Normalization

- **Text normalization** is the process of transforming text into a single canonical form that it might not have had before

  - Removing punctuation

I have been living in Berlin for 5 years. → I have been living in Berlin for 5 years

Normalization

- **Text normalization** is the process of transforming text into a single canonical form that it might not have had before

  - Removing / Converting numbers to their word equivalents

I have been living in Berlin for 5 years. → I have been living in Berlin for five years.

Normalization

- **Text normalization** is the process of transforming text into a single canonical form that it might not have had before

  - Strip white space

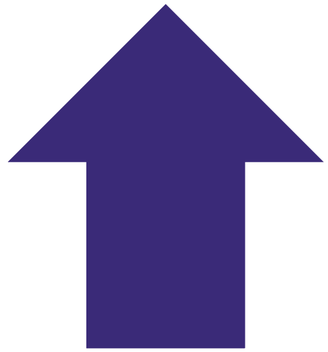
Stop word removal

- ***Stop words*** usually refers to the most common words in a language
  - Such as ***the***, ***and***, ***at***, ***a***, and ***on***
- There is no single universal list of stop words used by all NLP tools
- Stop words could be general or task specific
  - e.g., "***editorial***" in news domain

I have been living in Berlin for 5 years. → I have been living ~~in~~ Berlin ~~for~~ 5 years.

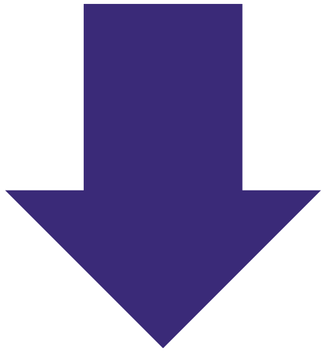
Stemming / Lemmatization

- The goal of both ***stemming*** and ***lemmatization*** is to reduce inflectional forms of a word to a common base form
  - ***Stemming*** is the process of eliminating affixes (suffixed, prefixes, infixes, circumfixes) from a word in order to obtain a word stem
  - ***Lemmatization*** is related to stemming, differing in that lemmatization is able to capture canonical forms based on a word's lemma



Lemmatization

- universal → universal
- university → university
- universities → university
- universe → universe



Stemming

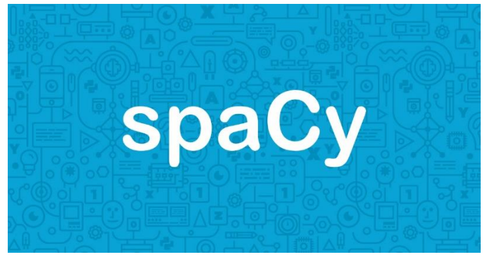
- universal → univers
- university → univers
- universities → univers
- universe → univers

Text pre-processing

- What is text pre-processing?
- Why is it important?
- How to pre-process textual data?
- Python packages for text pre-processing

Python packages for text pre-processing?

- Python **built-in** methods
- **re** (regular expression)
- **spaCy**
- **NLTK**



Python built-in methods

- Python includes lots of built-in methods to manipulate string in different shapes
  - upper()
  - lower()
  - title()
  - capitalize()

```
>>> string = “Sample String”
>>> lower_cased_string = string.lower()
>>> print(lower_cased_string)
sample string
>>> swap_cased_string = string.swapcase()
>>> print(swap_cased_string)
sAMPLE sTRING
```

Python built-in methods

- Common text pre-processing use cases:
  - Text normalization
    - Convert case (upper, lower, capitalize, swap, …)
  - Text tokenization (splitting text into sentence/words)
    - `split( sep=“ “ )`
    - `split( sep=“!“ )`
  - Noise removal
    - `replace( “#” , “ ” )`

re (Regular Expression)

- RE are essentially a tiny, highly specialized programming language embedded inside Python
  - The most common use cases of REs are to find strings that match a pattern (e.g., email address and phone number validation)
- Using this little language, you specify the rules for the set of possible strings that you want to match

```
>>> import re
>>> string = “a random. string? with.punctuation!”
>>> string = re.sub('([.,!?()])', r' \1 ‘, string)
>>> print(string)
a random .  string ?  with . punctuation !
```

re (Regular Expression)

- Common text pre-processing use cases:
  - Text normalization
    - Padding punctuation with white spaces
    - Removing numbers, punctuation, …
    - Replacing multiple spaces with a single space
  - Noise removal

spaCy

- **spaCy** is a modern Python library for industrial-strength Natural Language Processing
- The processing pipeline of spaCy includes lots of methods to preprocess and process textual input data

```
>>> import spacy
>>> nlp = spacy.load("en_core_web_sm")
>>> string = “it's a text”
>>> doc = nlp(string)
>>> for token in doc:
        print(token)
it
's
a
text
```

spaCy

- Common text pre-processing use cases:
  - Text normalization
    - Stop words removal
    - Stemming and Lemmatization
  - Text tokenization

NLTK

- **Natural language ToolKit (NLTK)** has lots of methods for pre-processing natural language text in python
- It’s one of the earliest and also easiest NLP libraries that you’ll use

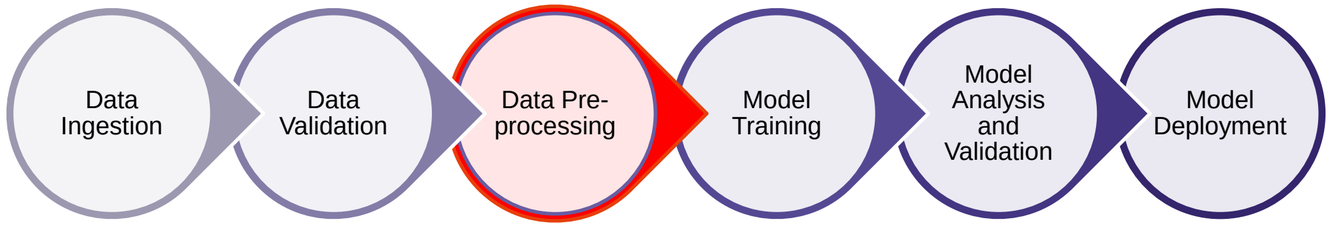

```
>>> from nltk.tokenize import sent_tokenize
>>> string = “Today is great! The sun is in the sky.”
>>> print(sent_tokenize(string))

[Today is great!', 'The sun is in the sky.']
```

NLTK

- Common text pre-processing use cases:
  - Text normalization
    - Punctuation removal
    - Stop words removal
    - Stemming and Lemmatization
  - Text tokenization

Summary



- Tokenization and segmentation
- Noise removal
- Normalization
  - Lower casing
  - Removing punctuation
  - Removing / Converting numbers to their word equivalents
  - Strip white space
  - Removing stop words
  - Stemming / Lemmatization

spaCy

NLTK
Natural Language Toolkit