Preprocessing text

Contents

Introduction

What is preprocesing for?

- Remove *irrelevant* information
- Add *relevant* information
- Convert your text to a suitable data format

Case study

Data:

- Collection of book reviews from *Goodreads*
- Two sets: mixed-language and English-only

Goal:

Today, we are mostly interested in making a **bag of words** for each review

Bag of words

Example 1

This book was great!

Example 2

This was a bad, BAD book.

Example 3

Great! Loved it!

Bags of words of our corpus:

Word	Review 1	Review 2	Review 3
a	0	1	0
bad	0	2	0
book	1	1	0
great	1	0	1
loved	0	0	1
it	0	0	1
this	1	1	0
was	1	1	0

Text cleaning

These book reviews are virtually identical:

Example

unbelievable story, enjoyed it very much

Example

Such an unbelievable story!
I enjoyed it very much.

But the strings are quite different!

What do we clean?

We will cover some common steps in text cleaning

Cleaning can make things easier for future calculations

However, filtering out useful information can also make things harder!

Ignore capitalisation

Convert everything to lowercase

Example

Loved it. Deon Meyer has become one of the best crime writers around.

Lowercase

loved it. deon meyer has become one of the best crime writers around.

Remove numbers

Numbers often don't contain relevant information.

Example

I read this book once back in 1986 while I was walking through the forests where this takes place. I then read it again in 2008. The magic of the book is in the setting - colonial days in a forest on the southern tip of Africa where the elephants hide.

For most tasks, we can safely remove them!

Clean up whitespace

Think of:

- removing double spaces
- removing spaces at the end of a line
- turning all newlines into spaces

Why?

- Remove transcription errors / typos
- Remove meaningless variation
- Make it easier to divide a text into words

Remove punctuation

Punctuation is useful for understanding sentence structure

But when we zoom out, the number of commas or periods in a text will not really tell us what it's about.

Example

Example

Just too many sub-plots going on here. Tons of good ideas, Philip Pullman, but stay focused! Bummed to end the series this way.

lowercase + punctuation removed

just too many subplots going on here tons of good ideas philip pullman but stay focused bummed to end the series this way

Tokenisation

Tokens are the "units" of a text - typically words

For example, the tokens in *"to be or not to be"* are *to*, *be*, *or*, *not*, *to*, and *be*.

Tokenisation means splitting a text into tokens, i.e. words.

That means going from a string of characters to a list of words.

Words as information units

Using words as the "unit" of text (rather than characters) breaks up our text into meaningful chunks

However, using words as atoms means our computer will not look "inside" the word

Removing stopwords

Stopwords are very common words, like *the*, *is*, *for*, etc.

Stopwords make up a lot of a text, but don't really tell you what a text is about.

Example

original

A powerful portrayal of a totalitarian dystopia, but too dull and depressive to really enjoy as a novel.

stopwords removed

powerful portrayal totalitarian dystopia, dull depressive really enjoy novel.

Which words are stopwords?

To remove stopwords, we need a list of words to remove.

You can:

- Use a list assembled for your language
- Make a list from your data, e.g. the 100 most common words

Stemming and lemmatising

The following words are quite similar:

political, *politically*, *politics*, *politician*, *politicians*, *politicise*, *politicised*, *politicising*

We can simplify our further analysis by saying that these are all variations of *politic-*

Stemming

Stemming strips words and only keeps the **stem**

Examples:

- *time* -> *tim*
- *times* -> *tim*
- *timing* -> *tim*

Lemmatisation

Lemmatisation removes inflection

Examples:

- *times* -> *time*
- *timing* -> *time*

Unlike stemming, lemmatisating tries to give complete words as output, rather than stems

Example

Example

Cleverly crafted cases, a plucky problem solver, and a unique backdrop. A delightful way to relax with a book in the evening.

stemmed

clever craft case, a plucki problem solver, and a uniqu backdrop. a delight way to relax with a book in the even.

lemmatised

cleverly craft case, a plucky problem solver, and a unique backdrop. a delightful way to relax with a book in the evening.

Stemming or lemmatising?

- Lemmatising preserves more information
- Stemming is less "careful": it may be more effective
- Lemmatising results in readable output
- Stemming requires fewer resources

POS tagging

Parts of speech (POS) are types of words (nouns, verbs, etc.)

Example

New setting for me. And all the characters are well developed, all flawed. I got lost in it.

POS tags

New/ADJ setting/NOUN for/ADP me/PRON ./PUNCT And/CCONJ all/PRON the/DET characters/NOUN are/AUX well/ADV developed/VERB, all/PRON flawed/VERB ./PUNCT I/PRON got/VERB lost/VERB in/ADP it/PRON ./PUNCT

Why POS tagging?

- Remove ambiguity: *I **walk** to the bus stop* vs *a **walk** in the park*
- Help to parse sentence structure
- Search for particular structures

Language detection

For mixed-language data, categorise the language before moving further

A lot of other preprocessing steps are language-dependent!

Named Entity Recognition

Tag all **named entities**: names of people, places, organisations, etc.

The output of NER can be used to answer questions like:

- Which people get mentioned in a lot of book reviews?
- How many reviews mention a particular place?

In other cases, we are not interested in names, and use NER to filter them out.

Preprocessing choices

There is no single "correct" method for preprocessing. Each of these steps can be helpful *or* detrimental depending on your research!

Important factors:

- language
- type of text
- research question
- amount of data
- analysis method

How to choose?

We try to balance:

- **preserving** information versus removing **noise**
- maximising the **informativity** or **likelihood** of observations

As a general rule:

- With **small dataset** and **simple models**, your analysis will struggle to handle noisy data.
- A **large model** with **lots of data** will benefit from richer input.

Final words

- There are a lot of possible steps to preprocessing!
- Luckily, these are quite conventional, so there are libraries that do the heavy lifting for us.
- However, each of these steps should be applied with caution! Think about your data and your task.