Text analysis I

Frequencie based methods

Introduction

- goal of text analysis: deriving meaningful insights from textual data
- for text as data more relevant than working with classical structured data:
  - no clear workflow
  - data, pre-processing, methods and models vary strongly on research question
  - get used to the data: look for errors, word combinations, specialties
  - much more back and forth; check text and results for inconsitencies
  - many decisions on the way (theory & compare & check!)

R-Packages for NLP

Frameworks to manage and analyze text data

- quanteda: Quantitative Analysis of Textual Data
- tidytext: Text mining using dplyr, ggplot2, and other tidy tools
- tidytextmining: XXX
- text2vec: Efficient framework with a concise API for text analysis and NLP
- tm: Text Mining Infrastructure in R
- spacyr: Wrapper for Python library spacy for advanced NLP and ML

... and many more for special tasks (on CRAN)

Bag of Words

- computers and ML algorithms need (vectors of) numbers
- transform raw text into numbers
- called *vectorization* or *feature encoding*
- results in *document-term matrix*
- collection of terms and their frequencies per document
- disregarding grammar, word order, and context
- simple, but easy and useful starting point for many NLP tasks

Bag of What?

- word, term, feature, token, n-grams, types
- emojis

meaningful basic units of text document to encode

corpora > documents > tokens

Bag of Questions

1. What are the rows of a DTM?
2. What are the dimensions of an DTM?
3. What are the benefits of BOW approach?
4. What are the shortcomings of BOW approach?

Bag of Quanteda workflow

1. Create corpus

```
corpus <- corpus(textdata, text_field = "text")
```

2. Tokenization

```
tokens <- tokens(corpus, what = "word")
```

3. Document-feature matrix

```
dfm <- dfm(tokens)
```

Bag of Usecases

- (compare) term frequencies & keyness -> exploration
- document classification (e.g. sentiment analysis)
- topic modeling (e.g. LDA)

Pre-Processing

manipulate raw text data for better results:

- remove punctation? numbers? symbols? separators?
- remove stopwords?
- cases to lower?
- stemming?
- lemmatization?
- n-grams?
- min/max occurence?

N-grams

- keep compound words together and ordered
- names, e.g. United States
- negoiations, e.g. not bad
- VERSTÄRKUNG, e.g. very good
- keep some meaning from documents
- usualy bi-grams (2), but also 3 or 4 possible

Frequencie, co-occurence and keyness

- show workflow

TF-IDF

- weighted scoring approach (not just frequencies)
- penalize words, that are frequent across all documents
- **term frequency**: relative frequencie of term in current document
- **inverse document frequency**: how rare is the word across all documents

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$
$$\mathrm{tf}(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$$
$$\mathrm{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

"(...) text that has undergone selection and refinement for the purpose of more analysis, and distinguished latent from manifest characteristics of the text as the qualities about which the textual data might provide inference." (Benoit 2020)

Annahmen von text-as-data-Ansätzen

- Text beinhaltet beobachtbare Implikationen von latenten Charakteristika
- Extraktionen von "features" kann Texte und deren Inhalt abbilden

- slight changes and different methods can result in different results

-> much more explorative, qualitative understanding of text specifity

https://github.com/microsoft/ML-For-Beginners/blob/main/6-NLP/2-Tasks/README.md#tasks-common-to-nlp

Bag of Words (BOW) vs. Neural Networks (Embeddings)
count based methods vs. predictiv methods
frequency vs. meaning of terms

bag of words know nothing about context -> add relevant n-grams (bi-grams) for negoations, e.g. "not bad"

Tokenization
Probably the first thing most NLP algorithms have to do is to split the text into tokens, or words. While this sounds simple, having to account for punctuation and different languages' word and sentence delimiters can make it tricky. You might have to use various methods to determine demarcations.

tokens can also be sub-words, e.g. do - n't