



Session 1: **Einführung & Preprocessing**





Agenda

1. Vorstellungsrunde, Ablauf & Einführung
2. Daten einlesen
3. Preprocessing: Bereinigung & Normalisierung
4. Text-as-Data Repräsentation
5. Outro

1. **Vorstellungsrunde**, Ablauf & Einführung

Wer sind wir?



Dr. Valerie Hase
IfKW, LMU Munich



valeriehase



valerie-hase.com

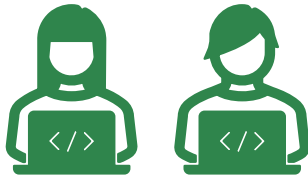


Luisa Kutlar
IfKW, LMU Munich



luisakutlar

Wer seid ihr?

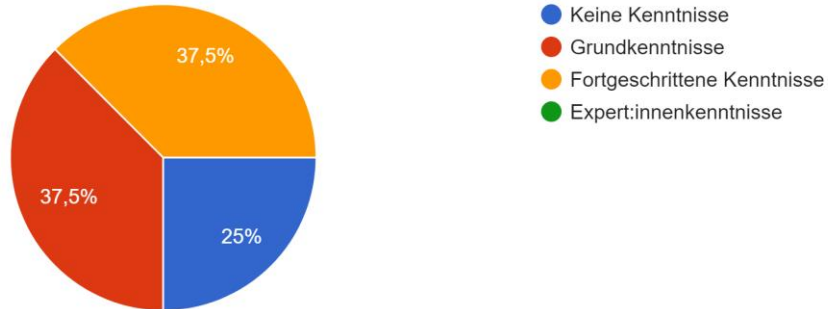


- Forschungsschwerpunkte
- Erfahrung mit R (oder Python, etc.)
- Erfahrung mit automatisierter Inhaltsanalyse
- Was ist euer „Ziel“ für diesen Workshop?

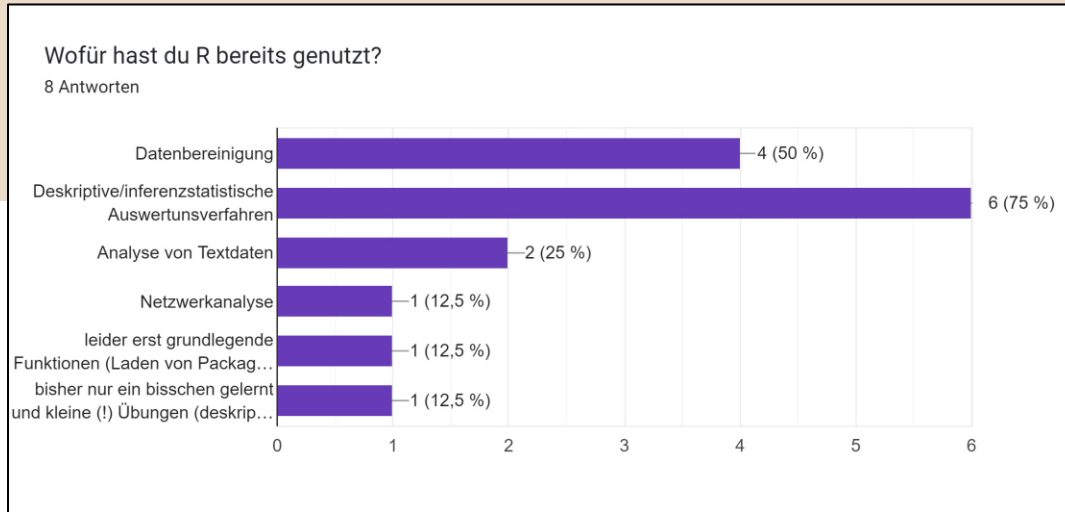
Wer seid ihr?

Wie würden du deine R-Kenntnisse beschreiben?

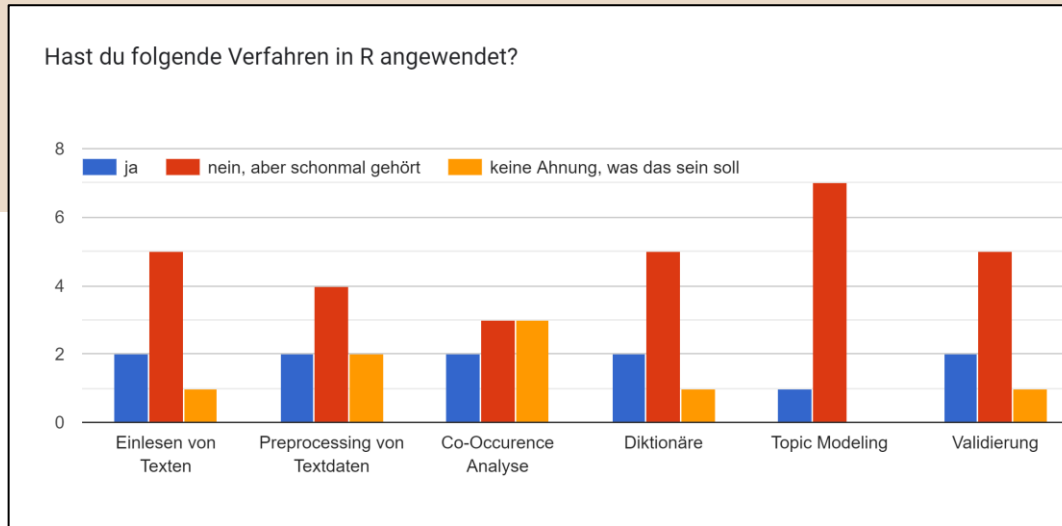
8 Antworten



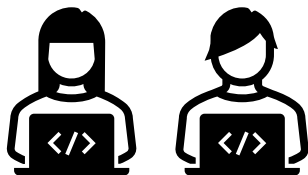
Wer seid ihr?



Wer seid ihr?



Was wollt ihr?



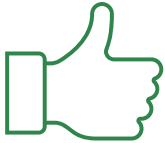
- Tempo für Anfänger:innen wie auch Fortgeschrittene
- Bereinigung
- Topic Modeling (etwa BERTTopic)
- Reporting, z. B. für Paper

Was lernt ihr in diesem Workshop (nicht)?

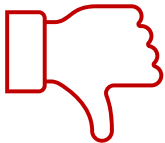


- Wichtigste Schritte und basale Verfahren der automatisierten Inhaltsanalyse mit Beispielen in R
- Chancen und Grenzen der Methode

Was lernt ihr in diesem Workshop (nicht)?



- Wichtigste Schritte und basale Verfahren der automatisierten Inhaltsanalyse mit Beispielen in R
- Chancen und Grenzen der Methode








- Fortgeschrittene Repräsentationen/Methoden (z. B. Transformer)
- Methoden für andere Daten (Bilder, Videos)

1. Vorstellungsrunde, **Ablauf** & Einführung


Wie läuft der Workshop ab?

ZEITPLAN

 Mi, 24. Juli

- 09:00 - 12:00:  *Einführung & Preprocessing*
- 12:00 - 13:00:  *Mittagspause*
- 13:00 - 15:00:  *Co-Occurrence-Analysen*
- 15:00 - 17:00:  *Diktionäre*

 Do, 25. Juli

- 09:00 - 12:00:  *Topic Modeling*
- 12:00 - 13:00:  *Mittagspause*
- 13:00 - 15:00:  *Qualitätskriterien*
- 15:00 - 16:00:  *Ausblick*



Wie läuft der Workshop ab?

Je Verfahren....

- Input (Methode & Voraussetzungen; Anwendungsbeispiele; Chancen & Grenzen)
- Beispiele in R
- Übungsaufgaben in R (Basis & Fortgeschritten)



Wo finde ich Materialien?

- Materialien:
 - Folien
 - R Code (Skript, Tutorial)
 - Daten
 - Weiterführende Texte & Tutorials



<https://valeriehase.github.io/textasdata-ms>



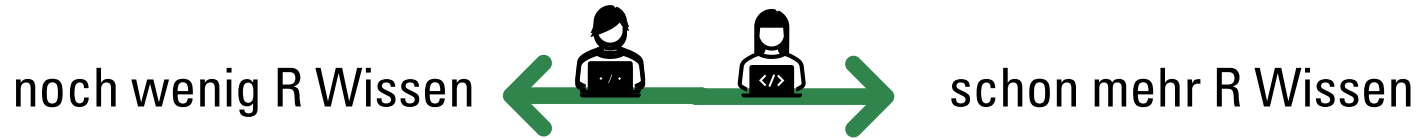
Wie kann ich Code ausführen?

Zwei Optionen

1. R-Skript öffnen & Code parallel ausführen
2. Tutorial öffnen & Code parallel anschauen

Was noch?

- Es gibt keine dummen Fragen – nur Dinge, die ihr **noch** nicht wisst



„dumm“ ist auf dieser Skala nicht vorhanden




Was noch?

- Es gibt keine dummen Fragen – nur Dinge, die ihr **noch** nicht wisst
- **Bitte fragt direkt**, wenn Dinge unklar sind:
 - Uns bringt es nichts, wenn wir euch „verlieren“
 - Euch bringt es nichts, wenn ihr 2 Tage hier „vergeudet“


1. Vorstellungsrunde, Ablauf & **Einführung**

Was passiert jetzt?

ZEITPLAN

 Mi, 24. Juli

- 09:00 - 12:00: **1** *Einführung & Preprocessing*
- 12:00 - 13:00: 🌿 *Mittagspause*
- 13:00 - 15:00: **2** *Co-Occurrence-Analysen*
- 15:00 - 17:00: **3** *Diktionäre*

 Do, 25. Juli

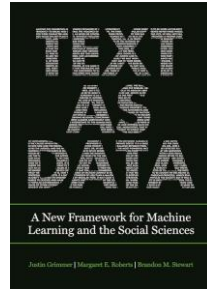
- 09:00 - 12:00: **4** *Topic Modeling*
- 12:00 - 13:00: 🌿 *Mittagspause*
- 13:00 - 15:00: **5** *Qualitätskriterien*
- 15:00 - 16:00: **6** *Ausblick*

Automatisierte Inhaltsanalyse: Definition

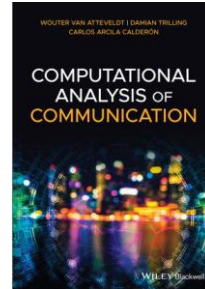
Automatisierte Inhaltsanalyse beschreibt die **automatisierte** (z. B. via Programmierskript) **Analyse von Inhalten** (z. B. Text, Bilder). Dabei **unterstützen Forschende/manuelle Codierer:innen**, etwa durch die Validierung von Ergebnissen. (Hase, 2023)



Benoit, 2020



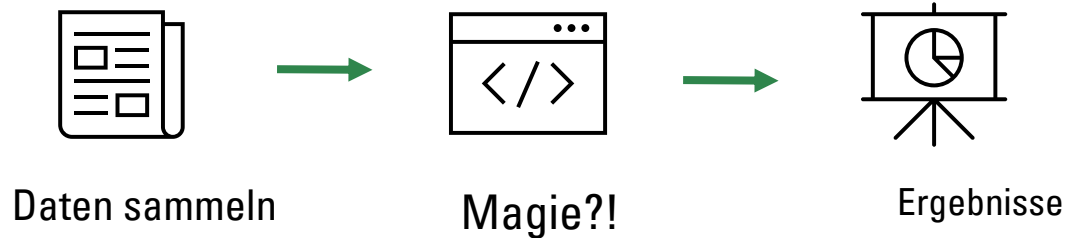
Grimmer et al., 2022



van Atteveldt et al., 2022

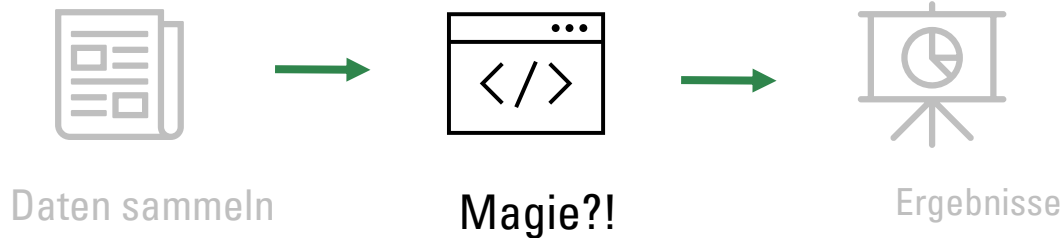
Automatisierte Inhaltsanalyse: Definition

Automatisierte Inhaltsanalyse beschreibt die **automatisierte** (z. B. via Programmierskript) **Analyse von Inhalten** (z. B. Text, Bilder). Dabei **unterstützen Forschende/manuelle Codierer:innen**, etwa durch die Validierung von Ergebnissen. (Hase, 2023)



Automatisierte Inhaltsanalyse: Definition

Automatisierte Inhaltsanalyse beschreibt die **automatisierte** (z. B. via Programmierskript) **Analyse von Inhalten** (z. B. Text, Bilder). Dabei **unterstützen Forschende/manuelle Codierer:innen**, etwa durch die Validierung von Ergebnissen. (Hase, 2023)



„Magie“ verstehen: Klassische Schritte

1. Preprocessing

*«In most cities in Germany,
the weather is sunny today»*

*Einlesen, bereinigen,
normalisieren &
Repräsentation wählen*

«in»	(2x)
«most»	(1x)
«cities»	(1x)
...	...

- Wir lesen Inhalte (hier: Text) in R ein.
- Wir bereinigen/normalisieren Inhalte.
- Wir wählen eine angemessene Text-as-Data Repräsentation.

„Magie“ verstehen: Klassische Schritte

1. Preprocessing

«*In most cities in Germany,
the weather is sunny today*»

*Einlesen, bereinigen,
normalisieren &
Repräsentation wählen*

«in»	(2x)
«most»	(1x)
«cities»	(1x)
...	...

2. Analyse

Frage: Wie beschreibt der Artikel das Wetter in Deutschland?

Antwort: Der Text enthält mehr Features, die mit positivem Sentiment assoziiert werden (z. B. «*sunny*»)

Wir wählen eine Methode, um auf Basis *manifester Features* (z.B. Anzahl von Wörtern, die mit positivem Sentiment assoziiert werden) auf *latente Konstrukte* zu schließen (z.B. Meinung über das Wetter).

„Magie“ verstehen: Klassische Schritte

1. Preprocessing

«*In most cities in Germany,
the weather is sunny today*»

*Einlesen, bereinigen,
normalisieren &
Repräsentation wählen*

«in»	(2x)
«most»	(1x)
«cities»	(1x)
...	...

2. Analyse

Frage: Wie beschreibt der Artikel das Wetter in Deutschland?

Antwort: Der Text enthält mehr Features, die mit positivem Sentiment assoziiert werden (z. B. «*sunny*»)

3. Test auf Qualitätskriterien

Wir evaluieren unsere Ergebnisse im Hinblick auf Qualitätskriterien (z.B. Reproduzierbarkeit, Replizierbarkeit, Validität).

„Magie“ verstehen: Klassische Schritte

1. Preprocessing

2. Analyse

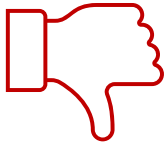
3. Test auf
Qualitätskriterien

Session 1
(Einführung &
Preprocessing)

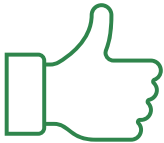
Session 2-4
(Co-Occurrence,
Diktionäre,
Topic Modeling)

Session 5
(Qualitätskriterien)

Klassische „Vorurteile“

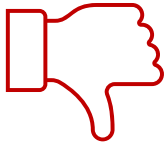


Bei automatisierten Inhaltsanalysen macht der Computer die ganze Arbeit.

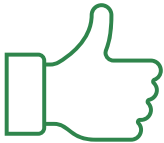


Automatisierte Methoden “augment humans, not replace them” (Grimmer & Stewart, [2013](#), S. 270). Automatisierte Methoden können durch manuelles Schreiben von Code, Validierung, etc. sogar aufwendiger sein als manuelle Inhaltsanalysen.

Klassische „Vorurteile“

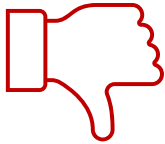


Automatisierte Methoden „demokratisieren“ die Forschung:
Jede:r kann diese anwenden!

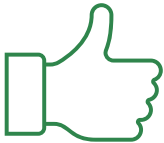


Ja, aber trotzdem Barrieren wie „English before everything“
(Baden et al., [2022](#), S. 9) – viele Methoden wurden für spezifischen Daten,
etwa Textdaten (Hase et al., [2022](#)), oder Sprachen, etwa Englisch (Baden
et al., [2022](#)), entwickelt.

Klassische „Vorurteile“

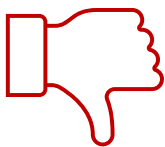


Indem wir „manifeste Indikatoren“ (etwa: Worthäufigkeiten) messen und mit automatisierten Methoden modellieren, können wir theoretische, latente Konstrukte abbilden.

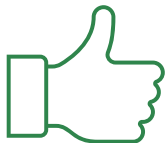


(Systematische) Fehler: “All quantitative models of language are wrong—but some are useful” (Grimmer & Stewart, [2013](#), S. 269).

Klassische „Vorurteile“



Es gibt nur eine einzige Methode, mit der wir Variablen automatisiert messen/modellieren können.



Optimististische Perspektive: “There is no globally best method”

(Grimmer & Stewart, [2013](#), p. 270) – variiert nach Daten, Epistemologien.

Pessimistische Perspektive: „Specialization before integration“

(Baden et al., [2022](#), p. 6)

Chancen

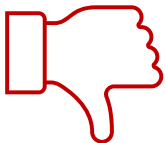


- Theoretisch fundierte Analysen für „große“ Korpora (etwa über Zeit, über Länder, über Sprachen)
- Exploration neuer Datentypen & Variablen
- Interdisziplinäre Perspektive auf Theorien & Methoden

Chancen & Herausforderungen



- Theoretisch fundierte Analysen für „große“ Korpora (etwa über Zeit, über Länder, über Sprachen)
- Exploration neuer Datentypen & Variablen
- Interdisziplinäre Perspektive auf Theorien & Methoden



- Text als Analyseeinheit, westlicher Bias
- Qualitätskriterien?



Take-aways



- **Definition automatisierte Inhaltsanalyse:** automatische Analyse von Inhalten, bei der Forschende/manuelle Codierer:innen weiterhin beteiligt sind, z. B. für Validierung.
- **Typische Schritte**
 - Preprocessing
 - Analyse
 - Test auf Qualitätskriterien

2. Daten einlesen

„Magie“ verstehen: Klassische Schritte

1. Preprocessing

«*In most cities in Germany,
the weather is sunny today*»

*Einlesen, bereinigen,
normalisieren &
Repräsentation wählen*

«in»	(2x)
«most»	(1x)
«cities»	(1x)
...	...

2. Analyse

Frage: Wie beschreibt der Artikel das Wetter in Deutschland?

Antwort: Der Text enthält mehr Features, die mit positivem Sentiment assoziiert werden (z. B. «*sunny*»)

3. Test auf Qualitätskriterien

Wir evaluieren unsere Ergebnisse im Hinblick auf Qualitätskriterien (z.B. Reproduzierbarkeit, Replizierbarkeit, Validität).



Wie kann ich Textdaten sammeln?

Nachrichten

- via Application Programming Interfaces (APIs)
- via Scraping
- via Nachrichtendatenbanken (e.g., Nexis, Factiva, Media Cloud)

Social Media Inhalte

- via Application Programming Interfaces (APIs)
- via Tracking
- via Datenspenden

Als weiterführende Quelle: Puschmann ([2022](#)); Haim & Hase ([2023](#))



Was gilt es dabei zu beachten?

- Legale Aspekte (z.B. „*Darf ich Daten scrapen?*“)
- Ethische Aspekte (z.B. „*Sollte ich Daten scrapen?*“)
- Methodische Aspekte (z.B. „*Welchen Bias können Scraping-Daten haben, z. B. durch fehlende Daten?*“)



Wie kann ich Textdaten in R einlesen?

- Kleine Auswahl möglicher Formate
 - Lokale Datei im Excel-Format (.csv, .xlsx)
 - Lokale Datei im Text-Format (.doc, .pdf, .txt)
 - Lokale Datei im ZIP-Format (.tar, .zip)
 - Lokale Datei anderer Statistikprogramme (.sav)
 - Bzw. direkt von URLs (z.B. durch Download, Scraping)
- Kleine Auswahl möglicher R-Pakete
 - „*base R*“ bzw. „readr“ (z.B. für Excel-Format)
 - „readtext“ (Teil von „quanteda“, z.B. für Text-Format)
 - „haven“ bzw. „foreign“ (z.B. für SPSS-Format)
 - „httr“ bzw. „RCurl“ bzw. „rvest“ bzw. „polite“ (z.B. für Download, Scraping)



Pakete aktivieren

```
#install.packages("tidyverse")  
#install.packages("quanteda")  
#install.packages("quanteda.textplots")  
#install.packages("RCurl")
```

```
library("tidyverse")  
library("quanteda")  
library("quanteda.textplots")  
library("RCurl")
```

Text-Daten einlesen

```
data <- read.csv2("data_tvseries.csv")
```

Aus einer
lokale Datei

```
library("RCurl")  
url <- getURL("https://raw.githubusercontent.com/valeriehase/textasdata-ms/main/data/  
data <- read.csv2(text = url)
```

Aus einer URL

Daten inspizieren

```
head(data)
```

	Title	Year	Parental.Rating	Rating	Number.of.Votes
1	1. Game of Thrones	2011-2019	TV-MA	9.2	2.3M
2	2. Breaking Bad	2008-2013	TV-MA	9.5	2.1M
3	3. Stranger Things	2016-2025	TV-14	8.7	1.3M
4	4. Friends	1994-2004	TV-14	8.9	1.1M
5	5. The Walking Dead	2010-2022	TV-MA	8.1	1.1M
6	6. Sherlock	2010-2017	TV-14	9.1	1M

1

2

3

4

5

6 The quirky spin on Conan Doyle's iconic sleuth pitches him as a "high-functioning soc

◀

▶

Daten inspizieren

```
data %>%
```

```
#Auswahl der Variable "Description"
```

```
select(Description) %>%
```

```
#Reduktion auf ersten Text
```

```
slice(1)
```

```
1 Nine noble families fight for control over the lands of Westeros, while an ancient en
```

Encoding-Probleme

- Computer speichern Schriftzeichen (z.B. Buchstaben "w", "o", "r", "d") mit Zahlenwerten (bytes), das jedem Schriftzeichen zugeordnet wird
- "Word" wird abgespeichert als *"01110111 01101111 01110010 01100100"*.
- Encodings, d.h. die Zeichenkodierung sind der "Schlüssel" um Zahlenwerte in Schriftzeichen zu verwandeln



Encoding-Probleme

- Das Problem:
 - Sprachspezifische Sonderzeichen wie „ü“, „ß“, Emojis, etc.
 - Unterschiedliche Encodings (etwa „UTF-8“, „ASCII“, „latin“)

Lösung

1. Texte direct mit richtigem Encoding einlesen
2. Andernfalls: Bereinigung/Normalisierung



Encoding-Probleme

```
#Beispiel-Satz  
string <- "Schöne Grüße aus München"
```

```
#Encoding prüfen  
Encoding(string)
```

```
[1] "UTF-8"
```

```
#Encoding testweise ändern  
Encoding(string) <- "latin1"  
string
```

```
[1] "Schöñe GrÄÿe aus MÄñchen"
```

```
#Mit Hilfe von regulären Ausdrücken bereinigen  
string %>%
```

```
#Ersatz für falsches Encoding "ö"  
gsub(pattern = "Ã¶", replacement = "ö") %>%
```

```
#Ersatz für falsches Encoding "ü"  
gsub(pattern = "Ã¼", replacement = "ü") %>%
```

```
#Ersatz für falsches Encoding "ß"  
gsub(pattern = "ÃŸ", replacement = "ß")
```



Take-Aways

- **Daten in R einlesen:** Endlose Möglichkeiten
 - Daten sammeln (API, scraping, etc.)
 - lokale .csv/.txt/.pdf-.Datei nutzen
 -
- An **rechtliche/ethische/methodische** Kontexte denken
- **Encoding-Probleme** beachten bzw. bereinigen („falsche“ Übersetzung zwischen Schrift- und Zahlenwerten, z. B. für verschiedene Sprachen oder Computersysteme)



Pause

3. Preprocessing: Bereinigung & Normalisierung



Definition Preprocessing

Reduziert über Bereinigung und Normalisierung **die Komplexität** von Textdaten, **ohne** deren substantielle Bedeutung zu minimieren

Ziel:

- (systematische) Fehler reduzieren (*Bereinigung*, etwa via regulären Ausdrücken)
- Text über Dokumente, Sprache, Plattformen, etc. vergleichbar machen (*Normalisierung*)

Problem:

Wie reduzieren wir die Komplexität von Textdaten, ohne zu viel Information zu verlieren?

Abwägungen beim Preprocessing



Komplexität von
Text reduzieren

Substanzielle
Bedeutung von
Text beibehalten



Auswahl: Preprocessing-Schritte (für einen Überblick, siehe Chai, [2023](#))

- Bereinigung (z.B. Encodings-Probleme mit regulären Ausdrücken entfernen)
- Normalisierung, etwa:
 - Tokenisierung
 - Kleinschreibung
 - Entfernung von Sonderzeichen
 - Entfernung von Stopwörtern
 - Lemmatisierung/Stemming
 - Häufige/seltene Features entfernen


3. Preprocessing: **Bereinigung** & Normalisierung



Preprocessing-Schritte

Headline: On the state of the Germany economy:
Will we have another financial crisis in Germany in
2023?

Preprocessing-Schritte



Headline: On the state of the Germany economy:
Will we have another financial crisis in Germany in
2023?

Bereinigung (z.B. Überbleibsel vom
Scraping entfernen)

Bereinigung mit regulären Ausdrücken

- **String patterns**: Zeichenabfolgen (z.B. Buchstaben)

Beispiel: `"Headline"`: Zeichenabfolge, die das Wort *"Headline"* ergibt.

- **Reguläre Ausdrücke**: string patterns with *nicht-wörtlicher Bedeutung*

Beispiel: `„[H|h]eadline“`: regulärer Ausdruck, der die Wörter *„Headline“* oder *„headline“* findet

Mehr Informationen: Buch von Sanchez ([2014](#)); Tutorial von Hase ([2021](#))

Bereinigung mit regulären Ausdrücken

- Ihr kennt die Logik von regulären Ausdrücke bereits!
- Wie sucht ihr z. B. auf Google oder Datenbanken nach Inhalten?

The screenshot shows the Nexis Uni search interface. At the top, there is a navigation bar with the Nexis Uni logo, a menu, and links for DE - Deutsch, Verlauf, Hilfe, Anmelden, and Registrieren. Below the navigation bar, there is a search bar containing the query 'klimakris* OR klimawand*', which is highlighted with a green box. To the right of the search bar are dropdown menus for 'Gesamter Zeitraum' and 'Alle Inhaltstypen', and a red search button. Below the search bar, there are links for 'Kürzlich ausgeführt und Favoriten', 'Erweiterte Suche', 'Suchtipps', and 'Get a Doc (US Recht)'. The main section is titled 'Suche mit Filtern' and contains three columns. The first column, 'Was suchen Sie?', has buttons for 'Nachrichten', 'eine Publikation', 'US Entscheidungen', 'Rechtszeitschriften (Englisch)', 'Firmeninformationen', and 'Ländersuche'. The second column, 'In allen Nachrichten suchen nach', has a text input field labeled 'Suchbegriffe eingeben'. The third column, 'Einen Datumsbereich auswählen', has a dropdown menu labeled 'Alle verfügbaren Daten' and a 'Suchen' button.



Bereinigung mit regulären Ausdrücken

- Ihr kennt die Logik von regulären Ausdrücke bereits!
- Wie sucht ihr z. B. auf Google oder Datenbanken nach Inhalten?
- Mit regulären Ausdrücken könnt ihr für Textanalysen z. B.
 - Texte identifizieren (z.B. Texte zu Klimawandel: `"Klimakris* OR Klimawand*"`)
 - Texte bereinigen (z.B. Entfernung von Formattierung: `"[h|Headline]"`)
 - Texte analysieren (z.B. Suche nach Sentiment): `„gut | toll | fantastisch“`



Bereinigung mit regulären Ausdrücken

- Ihr kennt die Logik von regulären Ausdrücke bereits!
 - Wie sucht ihr z. B. auf Google oder Datenbanken nach Inhalten?
 - Mit regulären Ausdrücken könnt ihr für Textanalysen z. B.
 - Texte identifizieren (z.B. Texte zu Klimawandel: `"Klimakris* OR Klimawand*"`)
 - **Texte bereinigen** (z.B. Entfernung von Formattierung: `"[h|Headline]"` **(jetzt!)**)
 - Texte analysieren (z.B. Suche nach Sentiment): `„gut | toll | fantastisch“`
- (Sitzung 3!)**



Bereinigung mit regulären Ausdrücken

Mit regulären Ausdrücken können wir Texte bereinigen, etwa via...

- **Logischen Operatoren** ("`&`" bezeichnet z.B. verknüpfende "und"-Bedingung)

Beispiel: "`Headline & headline`" identifiziert: "*Headline*" and "*headline*"

Bereinigung mit regulären Ausdrücken

Mit regulären Ausdrücken können wir Texte bereinigen, etwa via...

- **Character classes** ("`[a-z]`" identifiziert "irgendein Kleinbuchstabe")

Beispiel: "`[a-z]eadline`" identifiziert: "*headline*" and "*deadline*"

character.classes	meaning
[a-z]	finds any letter (lowercase)
[A-Z]	finds any letter (uppercase)
[:alpha:]	finds any letter (lowercase and uppercase)
[0-9]	finds any number
[a-zA-Z0-9]	finds any letter (lowercase and uppercase) or number
[:blank:]	finds spaces, among others
[:punct:]	finds punctuation, including ! # : ; .

Bereinigung mit regulären Ausdrücken

Mit regulären Ausdrücken können wir Texte bereinigen, etwa via...

- **Quantifier** ("*" sagt, dass der vorherige Ausdruck vorkommen kann – oder nicht)

Beispiel: "[Head]*line" identifiziert: "*Headline*" and "*line*"

quantifier	meaning
?	The preceding expression occurs at most once
+	The preceding expression occurs at least once
*	The preceding expression may or may not occur
{n}	The preceding expression occurs exactly n times
{n,}	The preceding expression occurs at least n times
{n,m}	The preceding expression occurs at least n times and at most m times

Achtung, Meta-Charakter!

- Weil einige Zeichen eine “nicht-wörtliche” Bedeutung haben (z.B. “?” sagt, dass der vorherige Ausdruck maximal einmal vorkommt), werden diese auch so interpretiert

Beispiel: “Is this a headline?” identifiziert

- “*Is this a headline?*”
- “*Is this a*”

Grund: “?” wird nicht wörtlich interpretiert (als Fragezeichen) sondern bedeutet, dass “headline” maximal einmal vorkommt

Achtung, Meta-Charakter!

- Weil einige Zeichen eine “nicht-wörtliche” Bedeutung haben (z.B. “?” sagt, dass der vorherige Ausdruck maximal einmal vorkommt), werden diese auch so interpretiert
- Lösung:
 - “Escape” mit `\\`
 - “Wörtliche Bedeutung” mit `fixed == TRUE`

Achtung, Meta-Charakter!

Metacharacters	Escape	Fix
*	*	fixed = TRUE
+	\+	fixed = TRUE
?	\?	fixed = TRUE
	\	fixed = TRUE
{	\{	fixed = TRUE
}	\}	fixed = TRUE
(\(fixed = TRUE
)	\)	fixed = TRUE



Bereinigung mit regulären Ausdrücken

In R können wir mit regulären Ausdrücken....

- **Texte identifizieren, die bestimmte Ausdrücke beinhalten**

Beispiel: Welcher Text beinhaltet das Wort "*headline*"?

- **Bestimmte Ausdrücke entfernen/ersetzen**

Beispiel: Kann ich das Wort "*headline*" aus meinem Text entfernen?

- **Zählen wie häufig bestimmte Ausdrücke vorkommen**

Beispiel: Kann ich zählen, wie häufig das Wort "*headline*" in meinem Text vorkommt?

Wie kann ich in R reguläre Ausdrücken nutzen?

Function.name	package	operation
<code>grep(pattern, x, value=FALSE)</code>	Base R	Returns the position of the elements in the vector x that contain a pattern
<code>grep(pattern, x, value=TRUE)</code>	Base R	Returns the content of elements in vector x that contain a pattern
<code>grepl(pattern, x)</code>	Base R	For all elements in vector x, indicates whether they contain a pattern
<code>sub(pattern, replacement, x)</code>	Base R	Replaces the first match for a pattern in vector x with replacement
<code>gsub(pattern, replacement, x)</code>	Base R	Replaces all matches for a pattern in vector x with replacement



Wie kann ich Textdaten in R bereinigen?

- Flexible string patterns mit regulären Ausdrücken schreiben
 - logische Operatoren
 - character classes
 - quantifier, etc.
- Kleine Auswahl möglicher R-Pakete
 - „*base R*“ (z.B. `grep()`, `gsub()`, etc.)
 - „*stringi*“ (z.B. `str_count()`, etc.)

```
function(scope, element, attr, ngSwitchController) {  
  var watchExpr = attr.ngSwitch || attr.on,  
      selectedTranscludes = [],  
      selectedElements = [],  
      previousElements = [],  
      selectedScopes = [];  
  
  scope.$watch(watchExpr, function ngSwitchWatchAction(value) {  
    var i, ii;  
    for (i = 0, ii = previousElements.length; i < ii; ++i) {  
      previousElements[i].remove();  
    }  
    previousElements.length = 0;  
  
    for (i = 0, ii = selectedScopes.length; i < ii; ++i) {  
      var selected = selectedElements[i];  
      selectedScope[i].destroy();  
    }  
  });  
  
  selectedElements.length = 0;  
  selectedScopes.length = 0;  
  
  if ((selectedTransclude = ...))
```

Zeit für R!

```
selectedElements.length = 0;  
selectedScopes.length = 0;  
  
if ((selectedTransclude = ...))
```

Bereinigung mit regulären Ausdrücken

```
data %>%  
  select(Title) %>%  
  head(5)
```

	Title
1	1. Game of Thrones
2	2. Breaking Bad
3	3. Stranger Things
4	4. Friends
5	5. The Walking Dead

```
#Entfernung der Zeichen vor dem Titel der TV-Serie  
data <- data %>%  
  mutate(Title = gsub("^[0-9]+[[:punct:]] ", "", Title))
```

#So sieht das Ergebnis aus:

```
data %>%  
  select(Title) %>%  
  head(5)
```

	Title
1	Game of Thrones
2	Breaking Bad
3	Stranger Things
4	Friends
5	The Walking Dead

Bereinigung mit regulären Ausdrücken

```
"^[0-9]+[[:punct:]]"
```



^ = Identifiziert Anfang einer
Zeichenabfolge



Bereinigung mit regulären Ausdrücken

```
"^[0-9]+[[:punct:]]"
```



`[0-9]+` = Identifiziert eine
Abfolge von Zahlen, die mind.
1 Zahl umfasst



Bereinigung mit regulären Ausdrücken

```
"^[0-9]+[[:punct:]]"
```

`[[:punct:]]` entfernt
Punktuaton



Bereinigung mit regulären Ausdrücken

```
"^[0-9]+[[:punct:]]"
```

= entfernt Leerzeichen



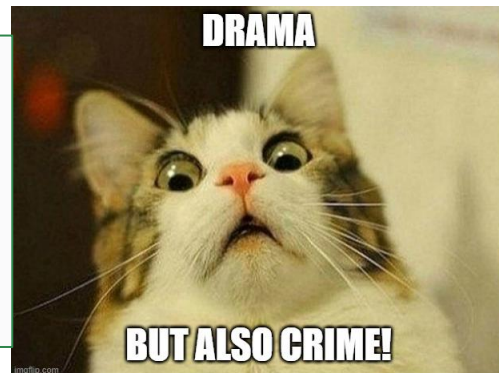
Filterung mit regulären Ausdrücken

```
data %>%  
  
# filtern aller TV_Serien, die "Drama" in der Beschreibung beinhalten  
filter(grepl("[D|d]rama", Description)) %>%  
  
# Inspektion der ersten fünf Titel  
select(Title) %>%  
head(5)
```



Filterung mit regulären Ausdrücken

```
data %>%  
  
# filtern aller TV_Serien, die "Drama" und "Crime" in der Beschreibung beinhalten  
filter(grepl("[D|d]rama|[C|c]rime", Description)) %>%  
  
# Inspektion der ersten fünf Titel  
select(Title) %>%  
head(5)
```



Jetzt seid ihr dran!



Könnt ihr...

- ? Basis: Alle Serien identifizieren, die in Deutschland spielen?
- ? Fortgeschritten: Alle Serien identifizieren, in denen es um Superhelden geht und "*superhero/superheroes*" in der Variable "`Description`" mit "*fancy R programmers*" ersetzen?

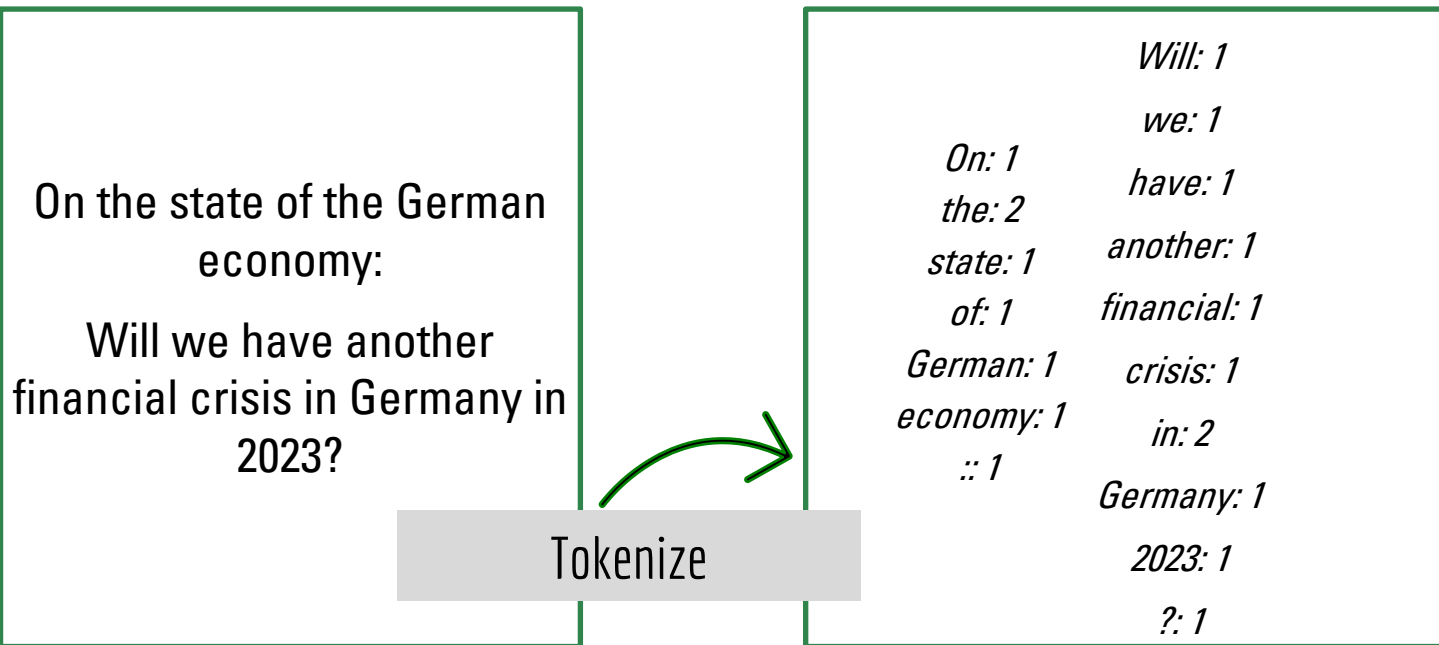
3. Preprocessing: Bereinigung & **Normalisierung**



Auswahl: Preprocessing-Schritte (für einen Überblick, siehe Chai, [2023](#))

- ~~Bereinigung (z.B. Encodings-Probleme mit regulären Ausdrücken entfernen)~~
- Normalisierung, etwa:
 - Tokenisierung
 - Kleinschreibung
 - Entfernung von Sonderzeichen
 - Entfernung von Stopwörtern
 - Lemmatisierung/Stemming
 - Häufige/seltene Features entfernen

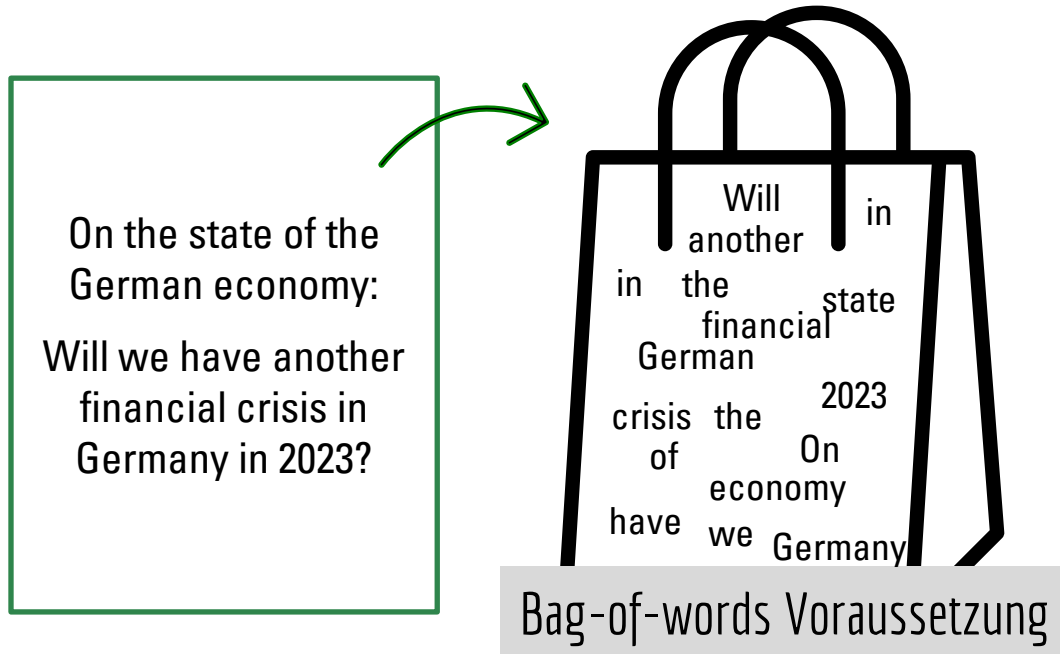
Preprocessing-Schritte



Preprocessing-Schritte



Preprocessing-Schritte



Durch Tokenisierung (d.h. Reduzierung des Textes auf Features) nehmen wir an, dass wir Text interpretieren können, **ohne** die Reihenfolge bzw. den Kontext von Features zu beachten.

Preprocessing-Schritte

On the state of the
German economy:
Will we have another
financial crisis in
Germany in 2023?



Bag-of-words Voraussetzung



Kennt ihr Beispiele,
bei denen diese
Voraussetzung
verletzt wäre?

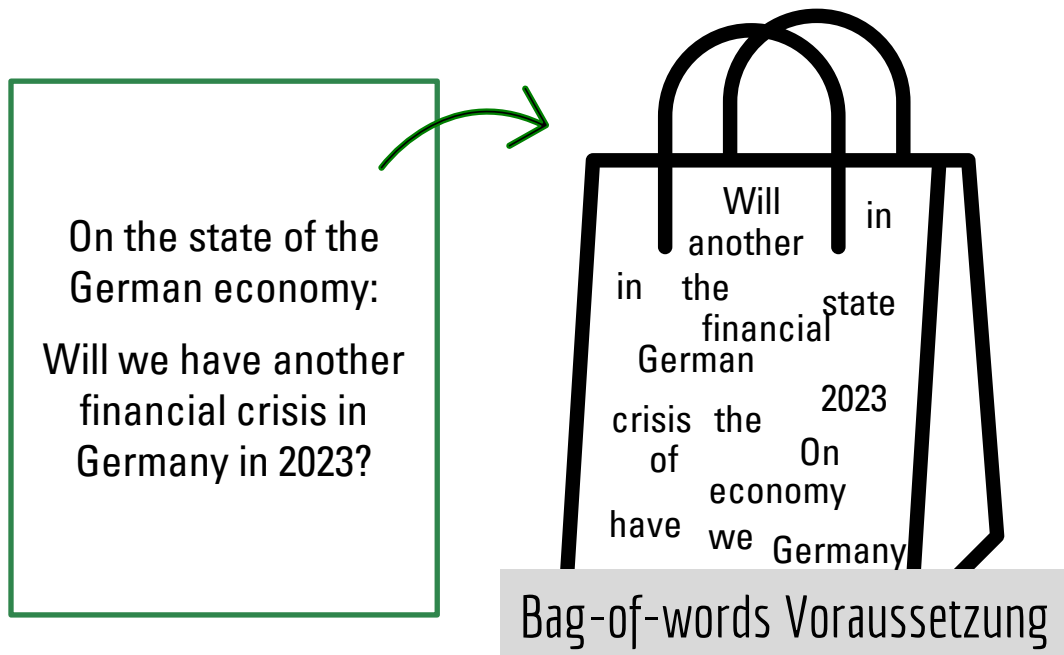
Abwägungen beim Preprocessing



Komplexität von
Text reduzieren
*(z.B. für schnellere
Analysen)*

Substanzielle
Bedeutung von Text
beibehalten
*(z.B. für tiefergehende
Analysen)*

Preprocessing-Schritte



Die Reihenfolge bzw. der Kontext von Features ist wichtig!

Deshalb lernen wir später (*Sitzung 6*) andere Text-as-Data-Repräsentationen, die diese Voraussetzung nicht verletzen.

Preprocessing-Schritte

On the state of the German
economy:
Will we have another
financial crisis in Germany in
2023?

Kleinschreibung

Will: 1
we: 1
have: 1
another: 1
financial: 1
crisis: 1
in: 2
Germany: 1
2023: 1
?: 1

On: 1
the: 2
state: 1
of: 1
German: 1
economy: 1
:: 1

Preprocessing-Schritte

On the state of the German
economy:
Will we have another
financial crisis in Germany in
2023?

Kleinschreibung

will: 1
we: 1
have: 1
another: 1
financial: 1
crisis: 1
in: 2
germany: 1
2023: 1
?: 1
on: 1
the: 2
state: 1
of: 1
german: 1
economy: 1
:: 1

Ist Kleinschreibung immer sinnvoll?

- Oft verändert Kleinschreibung die Bedeutung von Features nicht:
 - „In this seminar, you will spend some hours on **learning** R.“
 - „**Learning** R will be something you will spend some hours on in this seminar.“
- Aber es gibt Ausnahmen:

Bild



vs.

BILD



Preprocessing-Schritte

On the state of the German
economy:
Will we have another
financial crisis in Germany in
2023?

Entfernung von Sonderzeichen

will: 1
we: 1
have: 1
another: 1
financial: 1
crisis: 1
in: 2
germany: 1
2023: 1
?: 1
on: 1
the: 2
state: 1
of: 1
german: 1
economy: 1
:: 1

Preprocessing-Schritte

On the state of the German
economy:

Will we have another
financial crisis in Germany in
2023?

will: 1
we: 1
have: 1
another: 1
financial: 1
crisis: 1
in: 2
germany: 1
on: 1
the: 2
state: 1
of: 1
german: 1
economy: 1

Entfernung von Sonderzeichen

Ist die Entfernung von Sonderzeichen immer sinnvoll?

- Oft sind Sonderzeichen (z.B. Satzzeichen) nicht von substantieller Bedeutung.

Aber es gibt Ausnahmen:

- #metoo
- Is this true ???!!!!!!!
- 👍 😡 🤔 🐵
- www.zeit.de
- G7

Preprocessing-Schritte

On the state of the German
economy:
Will we have another
financial crisis in Germany in
2023?

Entfernung von Stopwörtern

will: 1
we: 1
on: 1
the: 2
state: 1
of: 1
german: 1
economy: 1
have: 1
another: 1
financial: 1
crisis: 1
in: 2
germany: 1

Ist die Entfernung von Stopwörtern immer sinnvoll?

- Es gibt keine „Konsensus“-Definition von Stopwörtern – stark kontextabhängig!
- Manche R-Pakete bieten euch „off-the-shelf“-Stopwortlisten

Beispiel – Stopwortliste im Paket quanteda

```
> stopwords("english")
[1] "i" "me" "my" "myself" "we" "our"
[10] "your" "yours" "yourself" "yourselves" "he" "him"
[19] "her" "hers" "herself" "it" "its" "itself"
[28] "theirs" "themselves" "what" "which" "who" "whom"
[37] "those" "am" "is" "are" "was" "were"
[46] "have" "has" "had" "having" "do" "does"
[55] "should" "could" "ought" "i'm" "you're" "he's"
[64] "they're" "i've" "you've" "we've" "they've" "i'd"
[73] "we'd" "they'd" "i'll" "you'll" "he'll" "she'll"
[82] "aren't" "wasn't" "weren't" "hasn't" "haven't" "hadn't"
[91] "won't" "wouldn't" "shan't" "shouldn't" "can't" "cannot"
[100] "that's" "who's" "what's" "here's" "there's" "when's"
```

Ist die Entfernung von Stopwörtern immer sinnvoll?

- Es gibt keine „Konsensus“-Definition von Stopwörtern – stark kontextabhängig!
- Manche R-Pakete bieten euch „off-the-shelf“-Stopwortlisten
- Solche Listen sollte man nur **sehr** vorsichtig nutzen – oft macht es Sinn, eigene „organische“ Stopwortlisten zu definieren



Für welche Analysen wollen wir „Stopwörter“ wie „our“, „we“ nicht entfernen, weil diese von substanziellem Interesse sind?

Preprocessing-Schritte

On the state of the German
economy:
Will we have another
financial crisis in Germany in
2023?

Entfernung von Stopwörtern

will: 1
we: 1
have: 1
another: 1
financial: 1
crisis: 1
in: 2
germany: 1
on: 1
the: 2
state: 1
of: 1
german: 1
economy: 1

Preprocessing-Schritte

On the state of the German
economy:

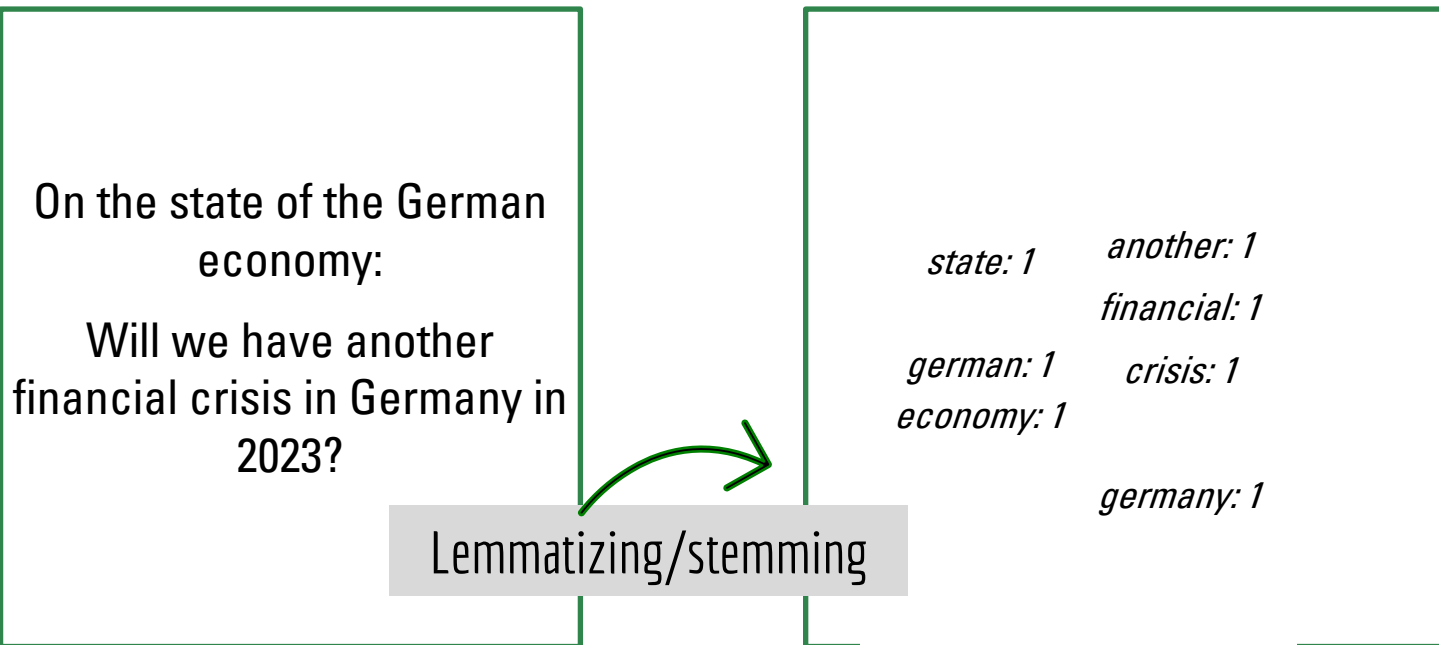
Will we have another
financial crisis in Germany in
2023?

state: 1 another: 1
financial: 1
german: 1 crisis: 1
economy: 1

germany: 1

Entfernung von Stopwörtern

Preprocessing-Schritte



Ist Lemmatizing/Stemming immer hilfreich?

Stemming: Reduziert Features auf ihren Wordstamm durch Entfernung des Suffixes

- „running“ „runs“ „run“ → „run“

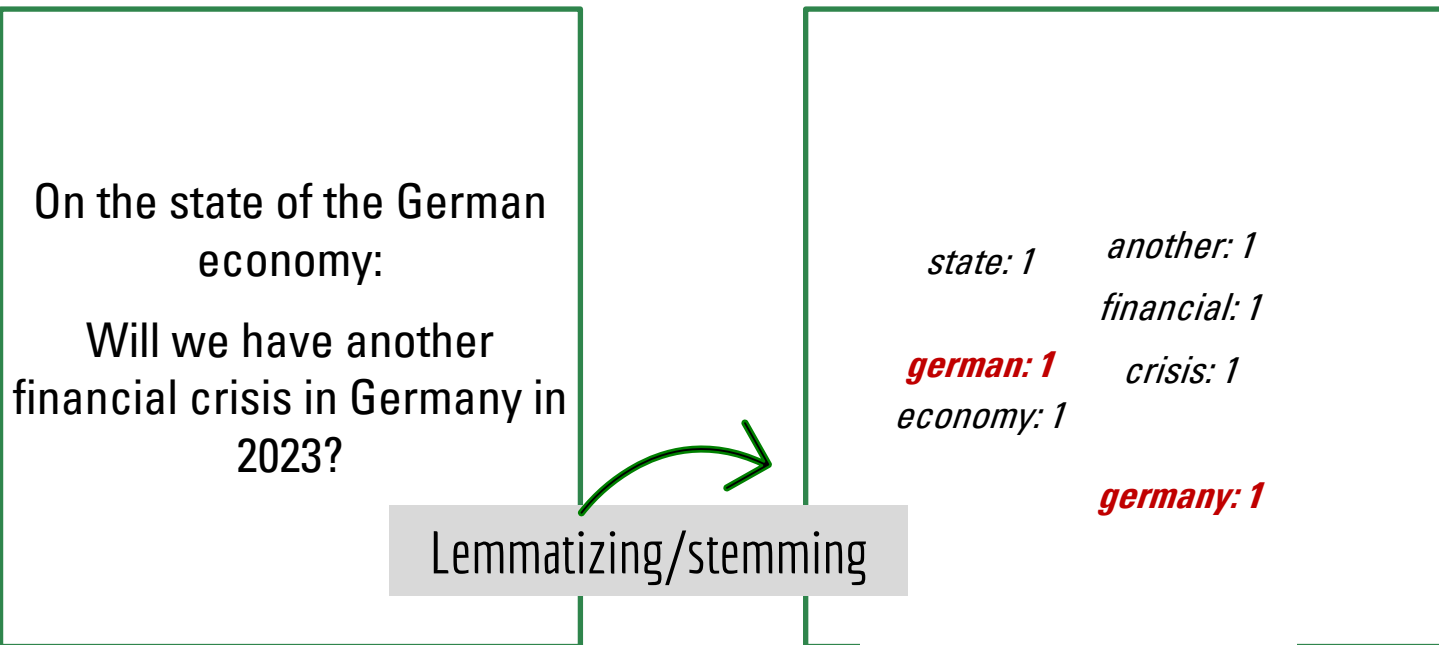
Funktioniert nicht immer gut

„run“ „ran“ → „run“ „ran“

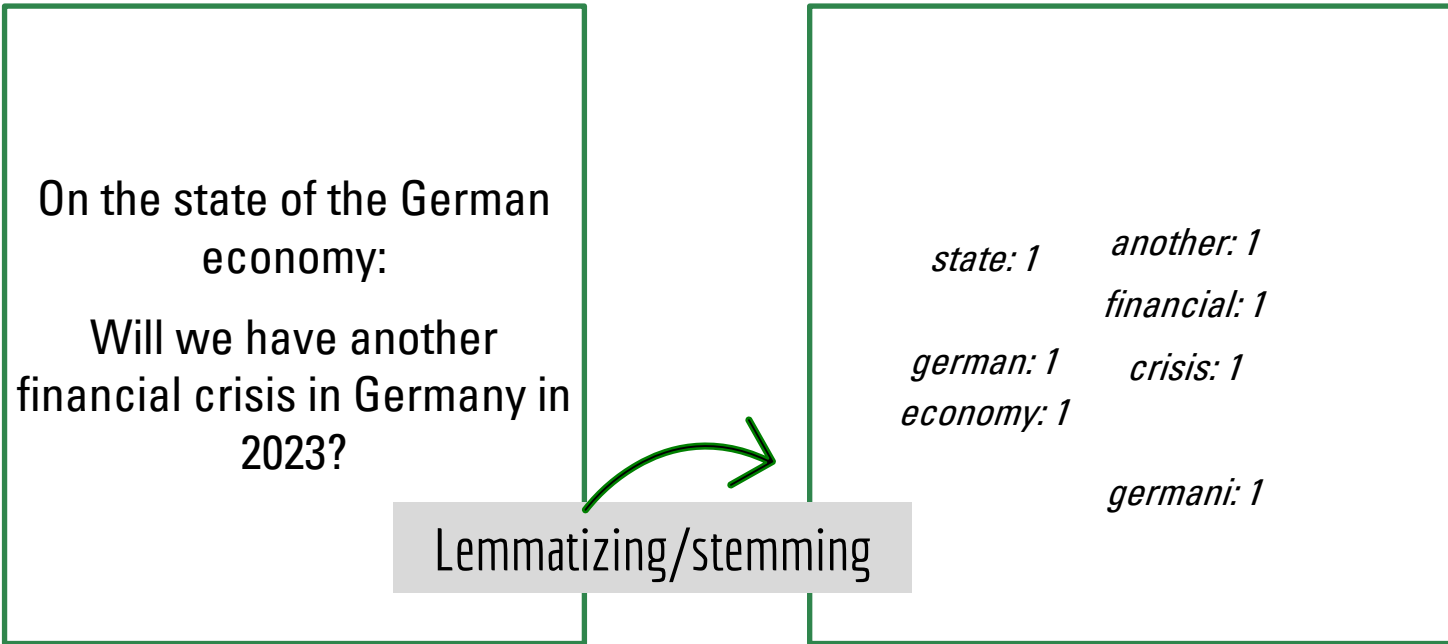
Lemmatizing: Reduziert Features auf ihre Grundform

„running“ „ran“ → „run“

Preprocessing-Schritte



Preprocessing-Schritte



Zur Reihenfolge von Preprocessing-Schritten

On the state of the German
economy:

Will we have another
financial crisis in Germany in
2023?



Könnt ihr euch
vorstellen, inwiefern die
Reihenfolge von
Preprocessing-Schritten
einen Einfluss haben
kann?

Zur Reihenfolge von Preprocessing-Schritten

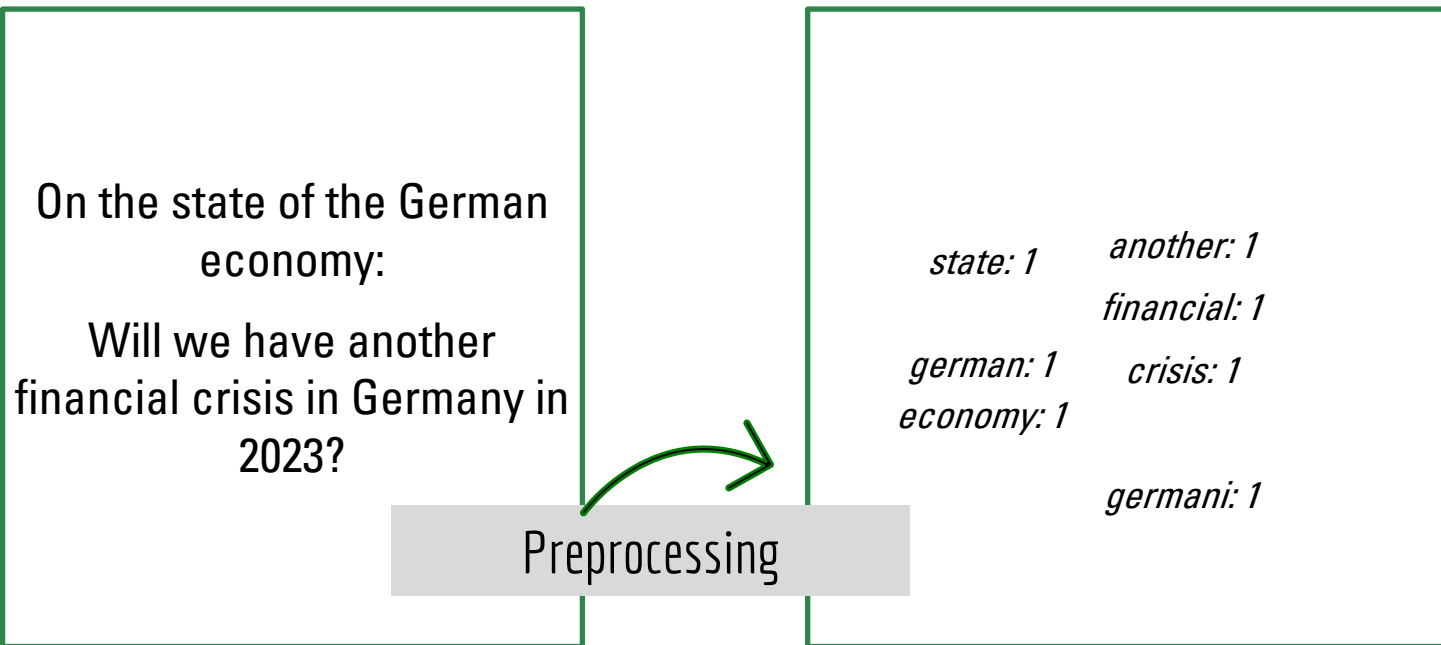
On the state of the German
economy:

Will we have another
financial crisis in Germany in
2023?



Wenn man **zuerst** Stemming anwendet («**yourselves**» → «**yourself**») und **danach** Stopwörter entfernt (Feature nur in Stopwort-Liste als «**yourselves**», nicht als «**yourself**»), wird das Stopwort nicht erkannt.

Text vor und nach Preprocessing





Wie kann ich Textdaten in R normalisieren?

- Entscheiden, welche Preprocessing-Schritte in welcher Reihenfolge notwendig sind
 - Tokenisierung
 - Kleinschreibung
 - Entfernung von Sonderzeichen bzw. Stopwörtern
 - Lemmatisierung/Stemming
 - Entfernung häufiger/seltener Features
- Kleine Auswahl möglicher R-Pakete
 - „quanteda“ bzw. „tidytext“ (v.a. für Tokenisierung, Entfernung von Features)
 - „udpipe“ bzw. „spacyr“ (v.a. für Stemming, Lemmatizing → s. Sitzung 2)
 - „preText“ (v.a. für Einfluss von Preprocessing, allerdings nicht mehr aktualisiert)

```
function(scope, element, attr, ngSwitchController) {  
  var watchExpr = attr.ngSwitch || attr.on,  
      selectedTranscludes = [],  
      selectedElements = [],  
      previousElements = [],  
      selectedScopes = [];  
  
  scope.$watch(watchExpr, function ngSwitchWatchAction(value) {  
    var i, ii;  
    for (i = 0, ii = previousElements.length; i < ii; ++i) {  
      previousElements[i].remove();  
    }  
    previousElements.length = 0;  
  
    for (i = 0, ii = selectedScopes.length; i < ii; ++i) {  
      var selected = selectedElements[i];  
      selectedScope[i].destroy();  
    }  
  });  
  
  selectedElements.length = 0;  
  selectedScopes.length = 0;  
  
  if ((selectedTransclude = ...))
```

Zeit für R!

```
selectedElements.length = 0;  
selectedScopes.length = 0;  
  
if ((selectedTransclude = ...))
```

Normalisierung von Text

```
tokens <- tokens(data$Description,  
                  what = "word", #Tokenisierung, hier zu Wörtern als Analyseeinheit  
                  remove_punct = TRUE, #Entfernung von Satzzeichen  
                  remove_numbers = TRUE) %>% #Entfernung von Zahlen  
  
#Kleinschreibung  
tokens_tolower() %>%  
  
#Entfernung von Stoppwörtern  
tokens_remove(stopwords("english")) %>%  
  
#Stemming  
tokens_wordstem()
```

Normalisierung von Text

```
#So sah unser erster Text vor dem Preprocessing aus  
data$Description[1]
```

```
[1] "Nine noble families fight for control over the lands of Westeros, while an ancient
```

```
#Und so danach  
tokens[1]
```

```
Tokens consisting of 1 document.
```

```
text1 :
```

```
[1] "nine"      "nobl"      "famili"    "fight"     "control"   "land"  
[7] "westero"   "ancient"   "enemi"     "return"    "dormant"   "millennia"
```

Mehr zu Stoppwörtern

```
#Wörter aus der quanteda Stoppwortliste entfernen
stoppwörter <- stopwords("english")
stoppwörter <- stoppwörter[!stoppwörter %in% c("i", "me")]

#Beispielhafte Anwendung
tokens(data$Description,
        what = "word", #Tokenisierung, hier zu Wörtern als Analyseeinheit
        remove_punct = TRUE, #Entfernung von Satzzeichen
        remove_numbers = TRUE) %>% #Entfernung von Zahlen

#Kleinschreibung
tokens_tolower() %>%

#Entfernung von Stoppwörtern - hier z.B. reduzierte quanteda-Liste
tokens_remove(stoppwörter) %>%

#Stemming
tokens_wordstem() %>%

#Ausgabe des ersten Textes
head(1)
```

Text vor und nach Preprocessing

Nine noble families fight for
control over the lands of
Westeros, while an ancient
enemy returns after being
dormant for millennia.

Preprocessing

<i>nine</i>	<i>westero</i>
<i>nobl</i>	<i>ancient</i>
<i>famili</i>	<i>enemi</i>
<i>fight</i>	<i>return</i>
<i>control</i>	<i>dormant</i>
<i>land</i>	<i>millenia</i>



Jetzt seid ihr dran!



Könnt ihr...

- ? Basis: Eine Liste mit 3-5 Stopwörtern erstellen und diese als Teil des Preprocessings zusätzlich entfernen?
- ? Fortgeschritten: Dafür sorgen, dass Namen von Städten (hier als Beispiel „*New York*“) als ein **einzelnes** Feature beibehalten werden?



Take-Aways

- **Preprocessing:** Reduziert über *Bereinigung* und *Normalisierung* die Komplexität von Textdaten, ohne deren substantielle Bedeutung zu minimieren
- **Features:** Analyseeinheit, auf die Texte reduziert werden (oft: Wörter)
- **Tokenization:** Prozess des «Herunterbrechens» von Texten auf Features
- **Bag-of-words Voraussetzung:** Annahme, dass die Reihenfolge und der Kontext von Features keinen Einfluss auf ihre Bedeutung haben (unwahrscheinlich!)



Pause

4. Text-as-Data Repräsentation



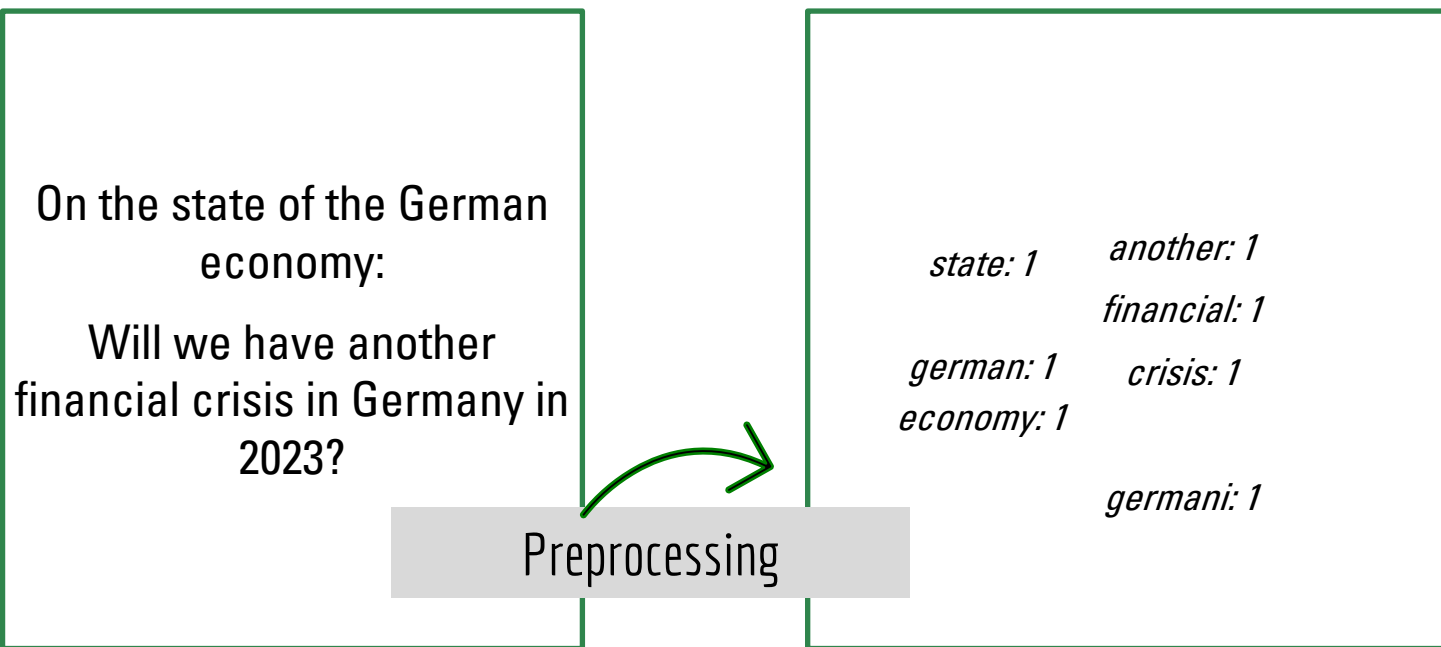
Text-as-Data Repräsentationen

Damit Computer Text verstehen können, müssen wir diesen in ein numerisches Format umwandeln.

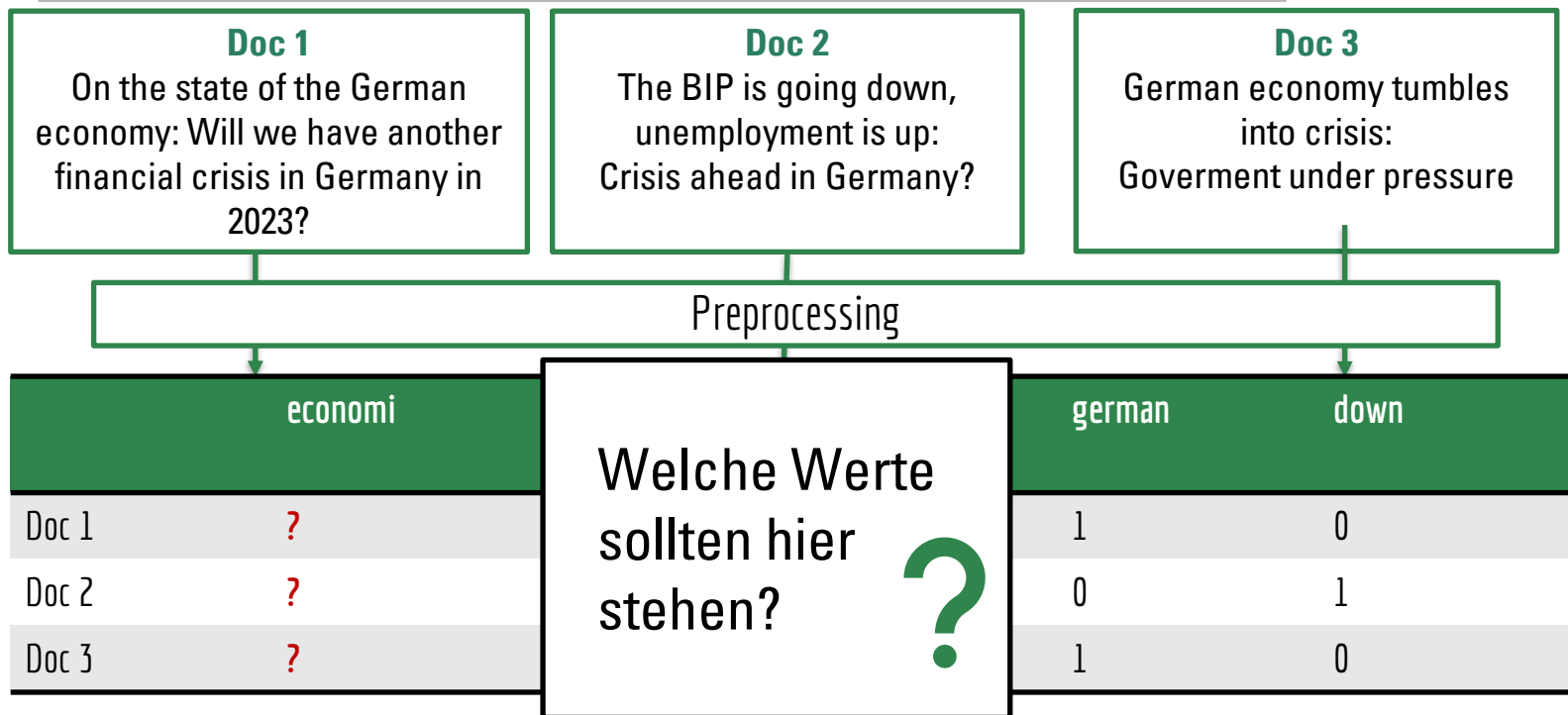
Beispiele für Repräsentationen

- Bag-of-words (document-feature-matrix)

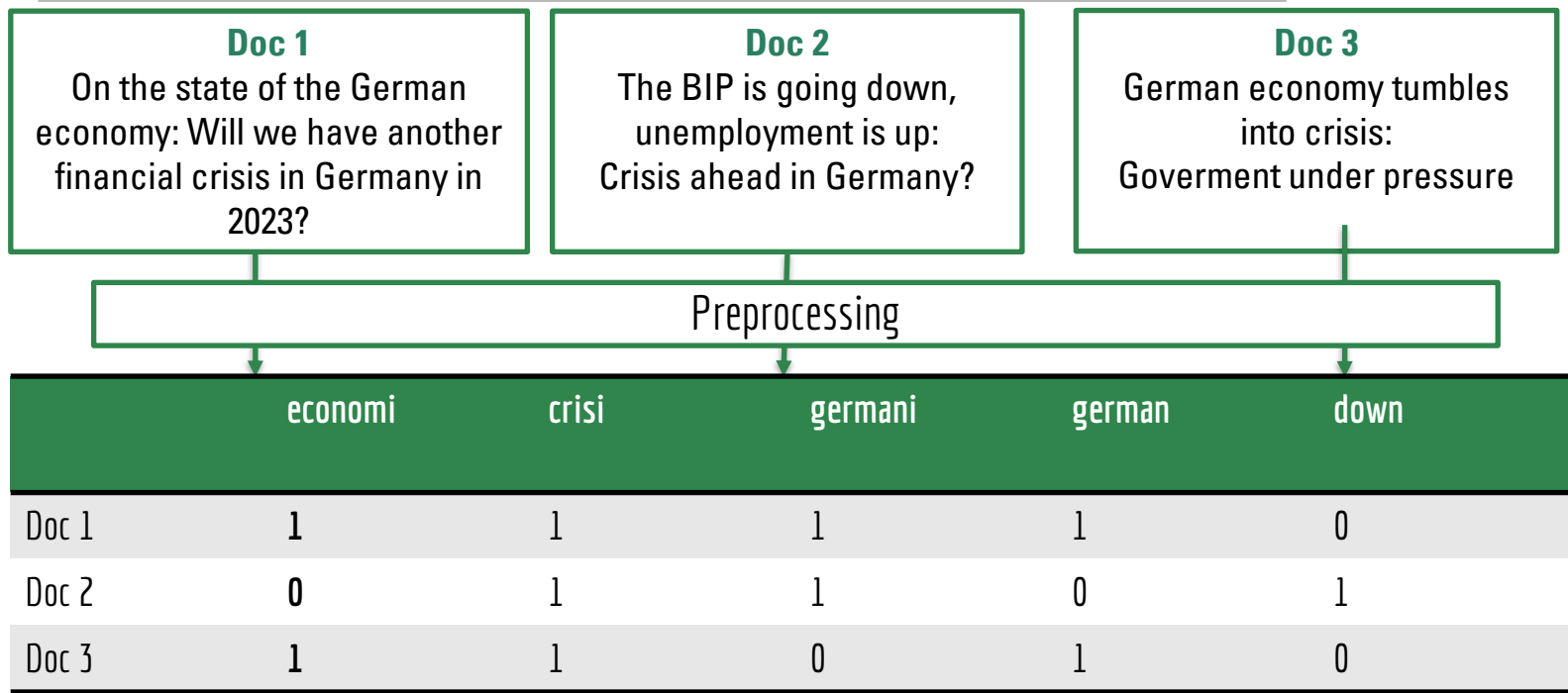
Bag-of-words



Document-feature-matrix



Document-feature-matrix





Document-feature-matrix

- Matrix, in der
 - Zeilen = Texte,
 - Spalten = Features
 - Zellen = Feature-Häufigkeiten
- Numerische Darstellung von Text auf Basis der bag-of-words-Voraussetzung
- Kann als “Input” für Methoden wie Diktionäre oder Machine Learning genutzt werden
- Mit wenig Preprocessing wird diese Matrix groß und enthält viele 0-Werte (Feature kommt nicht vor), d.h. ist “sparse”. Analysen können dann länger dauern.



Auswahl: Preprocessing-Schritte (für einen Überblick, siehe Chai, [2023](#))

- ~~Bereinigung (z.B. Encodings-Probleme mit regulären Ausdrücken entfernen)~~
- ~~Normalisierung, etwa:~~
 - ~~Tokenisierung~~
 - ~~Kleinschreibung~~
 - ~~Entfernung von Sonderzeichen~~
 - ~~Entfernung von Stopwörtern~~
 - ~~Lemmatisierung/Stemming~~
 - Häufige/seltene Features entfernen



Häufig/seltene Features entfernen

- Ein weiterer Preprocessing-Schritt, der möglich ist, sobald wir die Document-Feature-Matrix erstellt haben („Relative pruning“)
- Entfernung irrelevanter Features..
 - **Häufige** (z.B. in mind. 99% aller Texte), weil diese zu häufig sind, um so Texte zu unterscheiden
 - **Seltene** (z.B. in max. 0.5% aller Texte), weil diese zu selten sind, um so Texte zu unterscheiden

```
function(scope, element, attr, ngSwitchController) {  
  var watchExpr = attr.ngSwitch || attr.on,  
      selectedTranscludes = [],  
      selectedElements = [],  
      previousElements = [],  
      selectedScopes = [];  
  
  scope.$watch(watchExpr, function ngSwitchWatchAction(value) {  
    var i, ii;  
    for (i = 0, ii = previousElements.length; i < ii; ++i) {  
      previousElements[i].remove();  
    }  
    previousElements.length = 0;  
  
    for (i = 0, ii = selectedScopes.length; i < ii; ++i) {  
      var selected = selectedElements[i];  
      selectedScope[i].destroy();  
    }  
  });  
  
  selectedElements.length = 0;  
  selectedScopes.length = 0;  
  
  if ((selectedTransclude = ...))
```

Zeit für R!

Document-Feature-Matrix

```
#Wir erstellen eine Document-Feature matrix
dfm <- tokens %>%
  dfm()
```

```
#So sieht das Ergebnis aus
dfm
```



Document-feature matrix of: 900 documents, 4,246 features (99.66% sparse) and 0 docvars
features

docs	nine	nobl	famili	fight	control	land	westero	ancient	enemi	return
text1	1	1	1	1	1	1	1	1	1	1
text2	0	0	1	0	0	0	0	0	0	0
text3	0	0	0	0	0	0	0	0	0	0
text4	0	0	0	0	0	0	0	0	0	0
text5	0	0	0	0	0	0	0	0	0	0
text6	0	0	0	0	0	0	0	0	0	0

[reached max_ndoc ... 894 more documents, reached max_nfeat ... 4,236 more features]

Relative Pruning für weitere Normalisierung

```
#Anzahl Features vor relative pruning  
dfm
```

```
#Anwendung des relative pruning  
dfm <- dfm %>%  
  dfm_trim( min_docfreq = 0.005,  
            max_docfreq = 0.99,  
            docfreq_type = "prop",  
            verbose = TRUE)
```

```
#Anzahl Features nach relative pruning  
dfm
```

Document-feature matrix of: 900 documents, 605 features (98.58% sparse) and 0 docvars.

	features									
docs	famili	fight	control	land	ancient	enemi	return	turn	former	student
text1	1	1	1	1	1	1	1	0	0	0
text2	1	0	0	0	0	0	0	1	1	1
text3	0	0	0	0	0	0	0	0	0	0
text4	0	0	0	0	0	0	0	0	0	0
text5	0	0	0	0	0	0	0	0	0	0
text6	0	0	0	0	0	0	0	0	0	0

[reached max_ndoc ... 894 more documents, reached max_nfeat ... 595 more features]

Erste Analysen: Top Features

```
topfeatures(dfm, 10) %>%
```

```
#Umwandlung in einen "schöneren" Dataframe mit der Spalte "Häufigkeit"
```

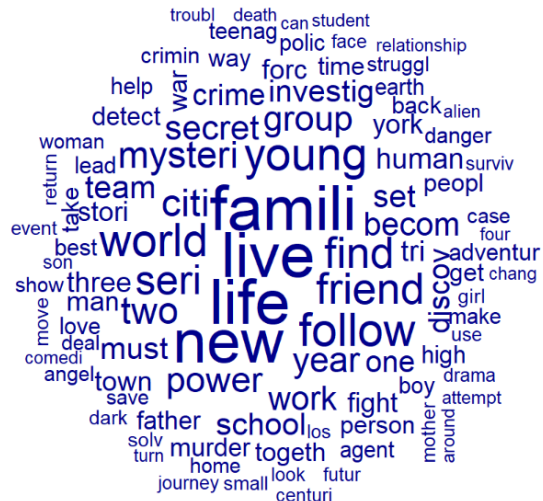
```
as.data.frame() %>%
```

```
rename("Häufigkeit" = '.')
```

	Häufigkeit
live	108
life	108
famili	107
new	103
young	74
follow	74
world	74
friend	70
find	69
seri	65

Erste Analysen: die berühmt-berüchtigte Word-Cloud

```
textplot_wordcloud(dfm, max_words = 100)
```



Warum die bag-of-word Annahme problematisch ist

Vermutlich  falsche Annahme:

- Can we "treat every word as having a distinct, unique meaning" (Grimmer et al., [2022](#), S. 79)?
- Can represent text "as if it were a bag of words, [...] with their position ignored, keeping only their frequency in the document" (Jurafsky & Martin, [2023](#), S. 60)?

Warum die bag-of-word Annahme problematisch ist

Annahme ✗ verletzt bzw. nicht hilfreich bei ...

- **Polysemie:** "Ich gehe zur Bank." vs. "Der Typ ist ne Bank!"
- **Verneinung:** "Nicht schlecht!"
- **Named Entities:** "Vereinigte Staaten", "Olaf Scholz"
- **Features mit ähnlichen Bedeutungen:** "I mag Gemüse." vs. "I mag Grünzeug."



Andere Text-as-Data Repräsentationen

Damit Computer Text verstehen können, müssen wir diesen in ein numerisches Format umwandeln.

Beispiele für Repräsentationen

- Bag-of-words (document-feature-matrix)

Andere Text-as-Data Repräsentationen

Damit Computer Text verstehen können, müssen wir diesen in ein numerisches Format umwandeln.

Beispiele für Repräsentationen

- Bag-of-words (document-feature-matrix)
- Nutzung von Ngrams ← Sitzung 2
- Nutzung von Syntax ← Sitzung 2

Andere Text-as-Data Repräsentationen

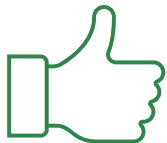
Damit Computer Text verstehen können, müssen wir diesen in ein numerisches Format umwandeln.

Beispiele für Repräsentationen

- Bag-of-words (document-feature-matrix)
- Nutzung von Ngrams ← Sitzung 2
- Nutzung von Syntax ← Sitzung 2
- Semantische Distributionen (z. B. Word Embeddings) ← Sitzung 6

5. Outro

Vor- und Nachteile von Preprocessing

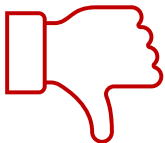


- Schnellere Analyse durch Reduktion auf relevante Features (und Entfernung irrelevanter Features)
- Normalisierung von Text (etwa über verschiedene Arten von Dokumenten), um Vergleichbarkeit zu gewährleisten

Vor- und Nachteile von Preprocessing



- Schnellere Analyse durch Reduktion auf relevante Features (und Entfernung irrelevanter Features)
- Normalisierung von Text (etwa über verschiedene Arten von Dokumenten), um Vergleichbarkeit zu gewährleisten



- Viele Freiheitsgrade bei methodischen Entscheidungen (relevante vs. irrelevante Features, Reihenfolge der Schritte)
- Downstream-Effekte auf Analyse (Denny & Spirling, [2018](#); Maier et al., [2020](#); Pipal et al., [2023](#)) bis hin zu systematischen Messfehlern

Wie berichte ich Preprocessing in Papern?

3.3.1. Preprocessing

First, we identified collocations related to noun phrases (“climate change”) or named entities (“United States”). We then reduced our corpus to nouns, proper nouns, verbs, and adjectives to eliminate features with little discriminative value. Subsequently, we applied lower-case conversion, tokenization to unigrams, removed punctuation, and eliminated stop words unique to our corpus. We then applied relative pruning to remove extremely rare or frequent words.

Beispiel aus Hase et al. ([2021](#)). Climate change in news media across the globe: An automated analysis of issue attention and themes in climate change coverage in 10 countries (2006–2018). *Global Environmental Change*.

Wie berichte ich Preprocessing in Papern?

3.3.1. Preprocessing

Subsequently, we applied lower-case conversion, tokenization to unigrams, removed punctuation, and eliminated stop words unique to our corpus. We then applied relative pruning to remove extremely rare or frequent words.

← Kennt ihr!

Beispiel aus Hase et al. ([2021](#)). Climate change in news media across the globe: An automated analysis of issue attention and themes in climate change coverage in 10 countries (2006–2018). *Global Environmental Change*.

Wie berichte ich Preprocessing in Papern?

3.3.1. Preprocessing

First, we identified collocations related to noun phrases (“climate change”) or named entities (“United States”). We then reduced our corpus to nouns, proper nouns, verbs, and adjectives to eliminate features with little discriminative value.

← Lernt ihr in
Sitzung 2!

Beispiel aus Hase et al. (2021). Climate change in news media across the globe: An automated analysis of issue attention and themes in climate change coverage in 10 countries (2006–2018). *Global Environmental Change*.







Wie berichte ich Preprocessing in Papern?

- **Immer:** Relevante Schritte kurz nennen & im Appendix ausführen
- **Noch besser:** Code (und ggf. Daten) teilen
- **Am besten:** Mit Multiverse-Analysen testen, wie robust Ergebnisse bei verschiedenen Preprocessing-Entscheidungen bleiben (Denny & Spirling, [2018](#); Maier et al., [2020](#); Pipal et al., [2023](#))





Wie geht es weiter?

ZEITPLAN

 Mi, 24. Juli

- 09:00 - 12:00:  *Einführung & Preprocessing*
- 12:00 - 13:00:  *Mittagspause*
- 13:00 - 15:00:  *Co-Occurrence-Analysen*
- 15:00 - 17:00:  *Diktionäre*

 Do, 25. Juli

- 09:00 - 12:00:  *Topic Modeling*
- 12:00 - 13:00:  *Mittagspause*
- 13:00 - 15:00:  *Qualitätskriterien*
- 15:00 - 16:00:  *Ausblick*

Danke! Fragen?



Dr. Valerie Hase
IfKW, LMU Munich



valeriehase



valerie-hase.com



Luisa Kutlar
IfKW, LMU Munich



luisakutlar