

Corpus der Entscheidungen des Bundesverwaltungsgerichts (CE-BVerwG-Source)

COMPILATION REPORT

Version 2021-10-19

License MIT-0

DOI: 10.5281/zenodo.5576823

Titel	Source Code des »Corpus der Entscheidungen des Bundesverwaltungsgerichts«
Abkürzung	CE-BVerwG-Source
Autor	Seán Fobbe
Version	2021-10-19
Download	https://doi.org/10.5281/zenodo.5576823
Lizenz	MIT No Attribution (MIT-0)

Zitiervorschlag

Seán Fobbe (2021). Source Code des »Corpus der Entscheidungen des Bundesverwaltungsgerichts« (CE-BVerwG-Source). Version 2021-10-19. Zenodo. DOI: 10.5281/zenodo.5576823.

Digital Object Identifier (DOI): Concept DOI und Version DOI

Soweit nicht anders angegeben ist die DOI immer eine »Version DOI« und bezieht sich nur auf eine bestimmte Version der Software. Sie verlinkt daher nur Version 2021-10-19. Für das Gesamtkonzept der Software steht eine »Concept DOI« zur Verfügung, die auf der Zenodo-Seite jeder Version unter »Cite all versions?« zu finden ist. Die »Concept DOI« verlinkt immer die aktuellste Version.

Lizenz: MIT No Attribution (MIT-0)

Copyright — 2021 — Seán Fobbe

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the »Software«), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so.

THE SOFTWARE IS PROVIDED »AS IS«, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Disclaimer

Dieser Datensatz ist eine private wissenschaftliche Initiative und steht in keiner Verbindung zu Behörden, Gerichten oder anderen amtlichen Stellen der Bundesrepublik Deutschland.

Inhaltsverzeichnis

1	Einleitung	8
1.1	Überblick	8
1.2	Funktionsweise	8
1.3	Systemanforderungen	8
1.4	Kompilierung	9
1.4.1	Datensatz	9
1.4.2	Codebook	9
2	Parameter	10
2.1	Name des Datensatzes	10
2.2	DOI des Datensatz-Konzeptes	10
2.3	DOI der konkreten Version	10
2.4	Lizenz	10
2.5	Verzeichnis für Analyse-Ergebnisse	10
2.6	Download-Umfang definieren	10
2.7	Modus: Debugging	10
2.8	Modus: Linguistische Annotationen	11
2.9	Optionen: Quanteda	11
2.10	Optionen: Knitr	11
2.10.1	Ausgabe-Format	11
2.10.2	DPI für Raster-Grafiken	11
2.10.3	Ausrichtung von Grafiken im Compilation Report	11
2.11	Frequenztabellen: Ignorierte Variablen	11
3	Vorbereitung	12
3.1	Datumsstempel	12
3.2	Datum und Uhrzeit (Beginn)	12
3.3	Ordner für Analyse-Ergebnisse erstellen	12
3.4	Quellennachweis für Diagramme erstellen	12
3.5	Packages Laden	13
3.6	Zusätzliche Funktionen einlesen	14
3.7	Quanteda-Optionen setzen	14
3.8	Knitr Optionen setzen	14
3.9	Vollzitate statistischer Software	14
3.10	Parallelisierung aktivieren	15
3.10.1	Logische Kerne	15
3.10.2	Quanteda	15
3.10.3	Data.table	15
4	Download: Weitere Datensätze	16
4.1	Registerzeichen und Verfahrensarten	16
4.2	Personendaten zu Präsident:innen	16
4.3	Personendaten zu Vize-Präsident:innen	16
5	Download: Alle Entscheidungen des BVerwG	17
5.1	Funktion zeigen	17
5.2	Linkliste erstellen	17
5.3	Anzahl Links	18

5.4	Anzahl einzigartiger Links	18
5.5	Vektor auf einzigartige Links reduzieren	19
5.6	Dateinamen erstellen	19
5.6.1	Extrahieren der ECLI-Ordinalzahl	19
5.6.2	Einzelkorrekturen vornehmen	19
5.6.3	Transformation zu Variablen mit Unterstrichen	20
5.6.4	Punkt zu NA	20
5.7	Registerzeichen mit Bindestrichen anpassen	20
5.8	Entscheidungsdatum in ISO-Format transformieren	21
5.9	Dateinamen finalisieren	21
5.10	Strenge REGEX-Validierung der Dateinamen	21
5.10.1	REGEX-Validierung durchführen	21
5.10.2	Ergebnis der REGEX-Validierung	22
5.10.3	Skript stoppen falls REGEX-Validierung gescheitert	22
5.11	Data Table für Download erstellen	22
5.12	Duplikate bereinigen	22
5.12.1	Test auf Duplikate	22
5.12.2	Von Duplikaten betroffene Links	23
5.12.3	Duplikate entfernen	23
5.13	[Debugging Modus] Reduktion des Download-Umfangs	23
5.14	Zeitstempel: Download Beginn	24
5.15	Download durchführen	24
5.16	Zeitstempel: Download Ende	24
5.17	Dauer: Download	25
5.18	[Debugging Modus] Löschen zufälliger Dateien	25
5.19	Download-Ergebnis	25
5.19.1	Anzahl herunterzuladender Dateien	25
5.19.2	Anzahl heruntergeladener Dateien	25
5.19.3	Fehlbetrag	26
5.19.4	Fehlende Dateien	26
5.20	Wiederholungsversuch: Download	26
5.21	Schlussbemerkung	27
5.22	Download: Gesamtergebnis	27
5.22.1	Anzahl herunterzuladender Dateien	27
5.22.2	Anzahl heruntergeladener Dateien	27
5.22.3	Fehlbetrag	27
5.22.4	Fehlende Dateien	27
6	Text-Extraktion	28
6.1	Vektor der zu extrahierenden Dateien erstellen	28
6.2	Anzahl zu extrahierender Dateien	28
6.3	Keine Seitenzählung	28
6.4	PDF extrahieren: Funktion anzeigen	28
6.5	Text Extrahieren	29
7	Korpus Erstellen	30
7.1	TXT-Dateien Einlesen	30
7.2	In Data Table umwandeln	30
7.3	Durch Zeilenumbruch getrennte Wörter zusammenfügen	30
7.3.1	Funktion anzeigen	30

7.3.2	Funktion ausführen	31
7.4	Variable “datum” als Datentyp “IDate” kennzeichnen	31
7.5	Variable “entscheidungsjahr” hinzufügen	31
7.6	Variable “eingangsjahr_iso” hinzufügen	31
7.7	Datensatz nach Datum sortieren	31
7.8	Variable “praesi” hinzufügen	32
7.8.1	Personaldaten einlesen	32
7.8.2	Personaldaten anzeigen	32
7.8.3	Hypothetisches Amtsende für Präsident:in	33
7.8.4	Schleife vorbereiten	33
7.8.5	Vektor erstellen	33
7.8.6	Vektor einfügen	33
7.9	Variable “v_praesi” hinzufügen	33
7.9.1	Personaldaten einlesen	33
7.9.2	Personaldaten anzeigen	34
7.9.3	Hypothetisches Amtsende für Vize-Präsident:in	34
7.9.4	Schleife vorbereiten	35
7.9.5	Vektor erstellen	35
7.9.6	Vektor einfügen	35
7.10	Variable “aktenzeichen” hinzufügen	35
7.11	Variable “ecli” hinzufügen	35
7.11.1	Ordinalzahl erstellen	36
7.11.2	Tabelle der Soll- und Ist-Ordinalzahlen vorbereiten	36
7.11.3	Tabelle der Soll- und Ist-Ordinalzahlen erstellen	37
7.11.4	Prüfung ob alle ECLI-Ordinalzahlen korrekt sind	38
7.11.5	Fehlerhafte ECLI-Ordinalzahlen anzeigen	38
7.11.6	Vollständige ECLI erstellen und einfügen	38
7.11.7	ECLI-Beispiele	38
7.12	Variable “verfahrensart” hinzufügen	38
7.12.1	Datensatz einlesen	39
7.12.2	Datensatz auf relevante Daten reduzieren	39
7.12.3	Indizes bestimmen	39
7.12.4	Vektor der Verfahrensarten erstellen und einfügen	39
7.13	Variable “doi_concept” hinzufügen	39
7.14	Variable “doi_version” hinzufügen	39
7.15	Variable “version” hinzufügen	39
7.16	Variable “lizenz” hinzufügen	39
8	Frequenztabellen erstellen	40
8.1	Funktion anzeigen	40
8.2	Ignorierte Variablen	41
8.3	Liste zu prüfender Variablen	41
8.4	Frequenztabellen erstellen	42
9	Frequenztabellen visualisieren	53
9.1	Präfix erstellen	53
9.2	Tabellen einlesen	53
9.3	Diagramm: Typ der Entscheidung	54
9.4	Diagramm: Spruchkörper (Aktenzeichen)	56
9.5	Diagramm: Registerzeichen	58

9.6	Diagramm: Präsident:in	60
9.7	Diagramm: Vize-Präsident:in	62
9.8	Diagramm: Entscheidungsjahr	64
9.9	Diagramm: Eingangsjahr (ISO)	65
10	Korpus-Analytik	66
10.1	Berechnung linguistischer Kennwerte	66
10.1.1	Funktion anzeigen	66
10.1.2	Berechnung durchführen	68
10.1.3	Variablen-Namen anpassen	68
10.2	Kennwerte dem Korpus hinzufügen	68
10.3	Variante mit Metadaten erstellen	68
10.4	Linguistische Kennwerte	69
10.4.1	Zusammenfassungen berechnen	69
10.4.2	Zusammenfassungen anzeigen	70
10.4.3	Zusammenfassungen speichern	70
10.5	Quantitative Variablen	71
10.5.1	Entscheidungsdatum	71
10.5.2	Zusammenfassungen berechnen	71
10.5.3	Zusammenfassungen anzeigen	72
10.5.4	Zusammenfassungen speichern	72
10.6	Verteilungen linguistischer Kennwerte	73
10.6.1	Diagramm: Verteilung Zeichen	73
10.6.2	Diagramm: Verteilung Tokens	74
10.6.3	Diagramm: Verteilung Typen	75
10.6.4	Diagramm: Verteilung Sätze	76
11	Linguistische Annotationen berechnen	77
12	Kontrolle der Variablen	78
12.1	Semantische Sortierung der Variablen	78
12.1.1	Variablen sortieren: Hauptdatensatz	78
12.1.2	Variablen sortieren: Metadaten	79
12.2	Anzahl Variablen der Datensätze	80
12.3	Alle Variablen-Namen der Datensätze	80
13	Beispiel-Werte für alle Metadaten anzeigen	82
14	CSV-Dateien erstellen	85
14.1	CSV mit vollem Datensatz speichern	85
14.2	CSV mit Metadaten speichern	85
14.3	CSV mit Annotationen speichern	85
15	Dateigrößen analysieren	86
15.1	Gesamtgröße	86
15.1.1	Korpus-Objekt in RAM (MB)	86
15.1.2	CSV Korpus (MB)	86
15.1.3	CSV Metadaten (MB)	86
15.1.4	CSV Annotiert (MB)	86
15.1.5	PDF-Dateien (MB)	86

15.1.6	TXT-Dateien (MB)	87
15.2	Diagramm: Verteilung der Dateigrößen (PDF)	88
15.3	Diagramm: Verteilung der Dateigrößen (TXT)	90
16	Erstellen der ZIP-Archive	92
16.1	Verpacken der CSV-Dateien	92
16.1.1	Vollständiger Datensatz	92
16.1.2	Metadaten	92
16.1.3	Annotiert	92
16.2	Verpacken der PDF-Dateien	92
16.3	Verpacken der TXT-Dateien	93
16.4	Verpacken der Analyse-Dateien	93
16.5	Verpacken der Source-Dateien	93
17	Kryptographische Hashes	94
17.1	Liste der ZIP-Archive erstellen	94
17.2	Funktion anzeigen	94
17.3	Hashes berechnen	95
17.4	In Data Table umwandeln	95
17.5	Index hinzufügen	95
17.6	In Datei schreiben	96
17.7	Leerzeichen hinzufügen um Zeilenumbruch zu ermöglichen	96
17.8	In Bericht anzeigen	96
18	Abschluss	99
18.1	Datumsstempel	99
18.2	Datum und Uhrzeit (Anfang)	99
18.3	Datum und Uhrzeit (Ende)	99
18.4	Laufzeit des gesamten Skriptes	99
18.5	Warnungen	99
19	Parameter für strenge Replikationen	100
	Literaturverzeichnis	102

1 Einleitung

1.1 Überblick

Dieses R-Skript lädt alle auf <https://www.bverwg.de/> veröffentlichten Entscheidungen des Bundesverwaltungsgerichts (BVerwG) herunter und kompiliert sie in einen reichhaltigen menschen- und maschinenlesbaren Korpus. Es ist die Basis für den **Corpus der Entscheidungen des Bundesverwaltungsgerichts (CE-BVerwG)**.

Alle mit diesem Skript erstellten Datensätze werden dauerhaft kostenlos und urheberrechtsfrei auf Zenodo, dem wissenschaftlichen Archiv des CERN, veröffentlicht. Jede Version ist mit ihrem eigenen, persistenten Digital Object Identifier (DOI) versehen. Die neueste Version des Datensatzes ist immer über den Link der Concept DOI erreichbar: <https://doi.org/10.5281/zenodo.3911067>

1.2 Funktionsweise

Primäre Endprodukte des Skripts sind folgende ZIP-Archive:

1. Der volle Datensatz im CSV-Format
2. Die reinen Metadaten im CSV-Format (wie unter 1, nur ohne Entscheidungstexte)
3. Alle Entscheidungen im TXT-Format (reduzierter Umfang an Metadaten)
4. Alle Entscheidungen im PDF-Format (reduzierter Umfang an Metadaten)
5. Alle Analyse-Ergebnisse (Tabellen als CSV, Grafiken als PDF und PNG)
6. Der Source Code und alle weiteren Quelldaten

Zusätzlich werden für alle ZIP-Archive kryptographische Signaturen (SHA2-256 und SHA3-512) berechnet und in einer CSV-Datei hinterlegt. Weiterhin kann optional ein PDF-Bericht erstellt werden (siehe unter “Kompilierung”).

1.3 Systemanforderungen

Das Skript in seiner veröffentlichten Form kann nur unter Linux ausgeführt werden, da es Linux-spezifische Optimierungen (z.B. Fork Cluster) und Shell-Kommandos (z.B. OpenSSL) nutzt. Das Skript wurde unter Fedora Linux entwickelt und getestet. Die zur Kompilierung benutzte Version entnehmen Sie bitte dem `sessionInfo()`-Ausdruck am Ende dieses Berichts.

In der Standard-Einstellung wird das Skript vollautomatisch die maximale Anzahl an Rechenkernen/Threads auf dem System zu nutzen. Wenn die Anzahl Threads (Variable “fullCores”) auf 1 gesetzt wird, ist die Parallelisierung deaktiviert.

Auf der Festplatte sollten 6 GB Speicherplatz vorhanden sein.

Um die PDF-Berichte kompilieren zu können benötigen Sie das R package **rmarkdown**, eine vollständige Installation von \LaTeX und alle in der Präambel-TEX-Datei angegebenen \LaTeX Packages.

1.4 Kompilierung

Mit der Funktion `render()` von **rmarkdown** können der **vollständige Datensatz** und das **Codebook** kompiliert und die Skripte mitsamt ihrer Rechenergebnisse in ein gut lesbares PDF-Format überführt werden.

Alle Kommentare sind im roxygen2-Stil gehalten. Die beiden Skripte können daher auch **ohne** `render()` regulär als R-Skripte ausgeführt werden. Es wird in diesem Fall kein PDF-Bericht erstellt und Diagramme werden nicht abgespeichert.

1.4.1 Datensatz

Um den **vollständigen Datensatz** zu kompilieren und einen **PDF-Bericht** zu erstellen, kopieren Sie bitte alle im Source-Archiv bereitgestellten Dateien in einen leeren Ordner und führen mit R diesen Befehl aus:

```
rmarkdown::render(input = "CE-BVerwG_Source_CorpusCreation.R",
                  output_file = paste0("CE-BVerwG_",
                                       Sys.Date(),
                                       "_CompilationReport.pdf"),
                  envir = new.env())
```

1.4.2 Codebook

Um das **Codebook** zu kompilieren und einen PDF-Bericht zu erstellen, führen Sie bitte im Anschluss an die Kompilierung des Datensatzes (!) untenstehenden Befehl mit R aus.

Bei der Prüfung der GPG-Signatur wird ein Fehler auftreten und im Codebook dokumentiert, weil die Daten nicht mit meiner Original-Signatur versehen sind. Dieser Fehler hat jedoch keine Auswirkungen auf die Funktionalität und hindert die Kompilierung nicht.

```
rmarkdown::render(input = "CE-BVerwG_Source_CodebookCreation.R",
                  output_file = paste0("CE-BVerwG_",
                                       Sys.Date(),
                                       "_Codebook.pdf"),
                  envir = new.env())
```

2 Parameter

2.1 Name des Datensatzes

```
datasetname <- "CE-BVerwG"
```

2.2 DOI des Datensatz-Konzeptes

```
doi.concept <- "10.5281/zenodo.3911067" # checked
```

2.3 DOI der konkreten Version

```
doi.version <- "10.5281/zenodo.5576822" # checked
```

2.4 Lizenz

```
license <- "Creative Commons Zero 1.0 Universal"
```

2.5 Verzeichnis für Analyse-Ergebnisse

Hinweis: Muss mit einem Schrägstrich enden!

```
outputdir <- paste0(getwd(),  
                    "/ANALYSE/")
```

2.6 Download-Umfang definieren

```
scope <- seq(from = 1,  
             to = 26000,  
             by = 1000)
```

2.7 Modus: Debugging

Der Debugging-Modus reduziert den Download-Umfang auf den in der Variable “debug.sample” definierten Umfang zufällig ausgewählter Entscheidungen und löscht im Anschluss fünf zufällig ausgewählte Entscheidungen um den Wiederholungsversuch zu testen. Nur für Test- und Demonstrationszwecke.

```
mode.debug <- FALSE  
debug.sample <- 100
```

2.8 Modus: Linguistische Annotationen

Wenn dieser Modus aktiviert ist wird mittels spacyr eine zusätzliche Variante des Datensatzes mit umfangreichen linguistischen Annotationen berechnet. Dieser Modus ist sehr rechenintensiv! Kann mit anderen Modi kombiniert werden.

```
mode.annotate <- TRUE
```

2.9 Optionen: Quanteda

```
tokens_locale <- "de_DE"
```

2.10 Optionen: Knitr

2.10.1 Ausgabe-Format

```
dev <- c("pdf",  
        "png")
```

2.10.2 DPI für Raster-Grafiken

```
dpi <- 300
```

2.10.3 Ausrichtung von Grafiken im Compilation Report

```
fig.align <- "center"
```

2.11 Frequenztabellen: Ignorierte Variablen

Diese Variablen werden bei der Erstellung der Frequenztabellen nicht berücksichtigt.

```
varremove <- c("text",  
              "eingangsnummer",  
              "datum",  
              "doc_id",  
              "ecli",  
              "aktenzeichen")
```

3 Vorbereitung

3.1 Datumsstempel

Dieser Datumsstempel wird in alle Dateinamen eingefügt. Er wird am Anfang des Skripts gesetzt, für den Fall, dass die Laufzeit die Datumsbarriere durchbricht.

```
datestamp <- Sys.Date()
print(datestamp)
```

```
## [1] "2021-10-19"
```

3.2 Datum und Uhrzeit (Beginn)

```
begin.script <- Sys.time()
print(begin.script)
```

```
## [1] "2021-10-19 02:51:58 CEST"
```

3.3 Ordner für Analyse-Ergebnisse erstellen

```
dir.create(outputdir)
```

3.4 Quellennachweis für Diagramme erstellen

```
figure.caption <- paste("DOI:",
                        doi.version,
                        "| Fobbe")
print(figure.caption)
```

```
## [1] "DOI: 10.5281/zenodo.5576822 | Fobbe"
```

3.5 Packages Laden

```
library(httr)          # HTTP-Werkzeuge
library(rvest)         # HTML/XML-Extraktion
library(knitr)         # Professionelles Reporting
library(kableExtra)    # Verbesserte Kable Tabellen
library(pdftools)      # Verarbeitung von PDF-Dateien
```

```
## Using poppler version 21.01.0
```

```
library(doParallel)    # Parallelisierung
```

```
## Loading required package: foreach
```

```
## Loading required package: iterators
```

```
## Loading required package: parallel
```

```
library(ggplot2)       # Fortgeschrittene Datenvisualisierung
library(scales)        # Skalierung von Diagrammen
library(data.table)    # Fortgeschrittene Datenverarbeitung
```

```
## data.table 1.14.0 using 8 threads (see ?getDTthreads). Latest news: r-  
datatable.com
```

```
library(readtext)      # TXT-Dateien einlesen
library(quanteda)      # Fortgeschrittene Computerlinguistik
```

```
## Package version: 3.1.0
## Unicode version: 13.0
## ICU version: 67.1
```

```
## Parallel computing: 16 of 16 threads used.
```

```
## See https://quanteda.io for tutorials and examples.
```

```
library(spacyr)      # Linguistische Annotationen
```

3.6 Zusätzliche Funktionen einlesen

Hinweis: Die hieraus verwendeten Funktionen werden jeweils vor der ersten Benutzung in vollem Umfang angezeigt um den Lesefluss zu verbessern.

```
source("R-fobbe-proto-package/f.linkextract.R")
source("R-fobbe-proto-package/f.year.iso.R")
source("R-fobbe-proto-package/f.dopar.pdfextract.R")
source("R-fobbe-proto-package/f.hyphen.remove.R")
source("R-fobbe-proto-package/f.fast.freqtable.R")
source("R-fobbe-proto-package/f.lingsummarize.iterator.R")
source("R-fobbe-proto-package/f.dopar.multihashes.R")
source("R-fobbe-proto-package/f.dopar.spacyparse.R")
```

3.7 Quanteda-Optionen setzen

```
quanteda_options(tokens_locale = tokens_locale)
```

3.8 Knitr Optionen setzen

```
knitr::opts_chunk$set(fig.path = outputdir,
                        dev = dev,
                        dpi = dpi,
                        fig.align = fig.align)
```

3.9 Vollzitate statistischer Software

```
knitr::write_bib(c(.packages()),
                 "packages.bib")
```

```
## tweaking foreach
```

3.10 Parallelisierung aktivieren

Parallelisierung wird zur Beschleunigung der Konvertierung von PDF zu TXT und der Datenanalyse mittels **quanteda** und **data.table** verwendet. Die Anzahl threads wird automatisch auf das verfügbare Maximum des Systems gesetzt, kann aber auch nach Belieben auf das eigene System angepasst werden. Die Parallelisierung kann deaktiviert werden, indem die Variable **fullCores** auf 1 gesetzt wird.

Der Download der Dateien ist bewusst nicht parallelisiert, damit das Skript nicht versehentlich als DoS-Tool verwendet wird.

Die hier verwendete Funktion **makeForkCluster()** ist viel schneller als die Alternativen, funktioniert aber nur auf Unix-basierten Systemen (Linux, MacOS).

3.10.1 Logische Kerne

```
fullCores <- detectCores()
print(fullCores)
```

```
## [1] 16
```

3.10.2 Quanteda

```
quanteda_options(threads = fullCores)
```

3.10.3 Data.table

```
setDTthreads(threads = fullCores)
```

4 Download: Weitere Datensätze

4.1 Registerzeichen und Verfahrensarten

Die Registerzeichen werden im Laufe des Skripts mit ihren detaillierten Bedeutungen aus dem folgenden Datensatz abgeglichen: "Seán Fobbe (2021). Aktenzeichen der Bundesrepublik Deutschland (AZ-BRD). Version 1.0.1. Zenodo. DOI: 10.5281/zenodo.4569564." Das Ergebnis des Abgleichs wird in die Variable "verfahrensart" in den Datensatz eingefügt.

```
if (file.exists("AZ-BRD_1-0-1_DE_Registerzeichen_Datensatz.csv") == FALSE){  
  download.file("https://zenodo.org/record/4569564/files/AZ-BRD_1-0-1_DE_  
  Registerzeichen_Datensatz.csv?download=1",  
  "AZ-BRD_1-0-1_DE_Registerzeichen_Datensatz.csv")  
}
```

4.2 Personendaten zu Präsident:innen

Die Personendaten stammen aus folgendem Datensatz: »Seán Fobbe and Tilko Swalve (2021). Presidents and Vice-Presidents of the Federal Courts of Germany (PVP-FCG). Version 2021-04-08. Zenodo. DOI: 10.5281/zenodo.4568682«.

```
if (file.exists("PVP-FCG_2021-04-08_GermanFederalCourts_Presidents.csv") == FALSE  
) {  
  download.file("https://zenodo.org/record/4568682/files/PVP-FCG_2021-04-08_  
  GermanFederalCourts_Presidents.csv?download=1",  
  "PVP-FCG_2021-04-08_GermanFederalCourts_Presidents.csv")  
}
```

4.3 Personendaten zu Vize-Präsident:innen

Die Personendaten stammen aus folgendem Datensatz: »Seán Fobbe and Tilko Swalve (2021). Presidents and Vice-Presidents of the Federal Courts of Germany (PVP-FCG). Version 2021-04-08. Zenodo. DOI: 10.5281/zenodo.4568682«.

```
if (file.exists("PVP-FCG_2021-04-08_GermanFederalCourts_VicePresidents.csv") ==  
FALSE){  
  download.file("https://zenodo.org/record/4568682/files/PVP-FCG_2021-04-08_  
  GermanFederalCourts_VicePresidents.csv?download=1",  
  "PVP-FCG_2021-04-08_GermanFederalCourts_VicePresidents.csv")  
}
```


5 Download: Alle Entscheidungen des BVerwG

5.1 Funktion zeigen

```
print(f.linkextract)
```

```
## function(URL){  
##   tryCatch({  
##     read_html(URL) %>%  
##       html_nodes("a")%>%  
##       html_attr('href')},  
##     error = function(cond) {  
##       return(NA)}  
##   )  
## }
```

5.2 Linkliste erstellen

```
links.list <- vector("list",  
                     length(scope))  
  
for (i in seq_along(scope)){  
  
  URL <- paste0("https://www.bverwg.de/suche?q=+*&db=e&dt=&lim=1000&start=",  
               scope[i])  
  
  volatile <- f.linkextract(URL)  
  
  links.temp <- grep ("/de/[0-9] [0-9] [0-9] [0-9]",  
                    volatile,  
                    ignore.case = TRUE,  
                    value = TRUE)  
  
  links.temp <- gsub(pattern="/de/",  
                    replacement="",  
                    links.temp)  
  
  links.temp <- paste0("https://www.bverwg.de/entscheidungen/pdf/",  
                      links.temp,  
                      ".pdf")  
  
  links.temp <- gsub(" ",  
                    "",  
                    links.temp,  
                    fixed=TRUE)  
  
  links.list[[i]] <- links.temp
```

```
print(paste(scope[i], "bis", scope[i] + 999))

Sys.sleep(runif(1, 1, 2))
}
```

```
## [1] "1 bis 1000"
## [1] "1001 bis 2000"
## [1] "2001 bis 3000"
## [1] "3001 bis 4000"
## [1] "4001 bis 5000"
## [1] "5001 bis 6000"
## [1] "6001 bis 7000"
## [1] "7001 bis 8000"
## [1] "8001 bis 9000"
## [1] "9001 bis 10000"
## [1] "10001 bis 11000"
## [1] "11001 bis 12000"
## [1] "12001 bis 13000"
## [1] "13001 bis 14000"
## [1] "14001 bis 15000"
## [1] "15001 bis 16000"
## [1] "16001 bis 17000"
## [1] "17001 bis 18000"
## [1] "18001 bis 19000"
## [1] "19001 bis 20000"
## [1] "20001 bis 21000"
## [1] "21001 bis 22000"
## [1] "22001 bis 23000"
## [1] "23001 bis 24000"
## [1] "24001 bis 25000"
## [1] "25001 bis 26000"
```

```
links.pdf <- unlist(links.list)
```

5.3 Anzahl Links

```
length(links.pdf)
```

```
## [1] 50738
```

5.4 Anzahl einzigartiger Links

```
uniqueN(links.pdf)
```

```
## [1] 25367
```

5.5 Vektor auf einzigartige Links reduzieren

```
links.pdf <- unique(links.pdf)
```

5.6 Dateinamen erstellen

5.6.1 Extrahieren der ECLI-Ordinalzahl

Die Links zu jeder Entscheidung enthalten das Ordinalzahl-Element ihres jeweiligen ECLI-Codes. Struktur und Inhalt der ECLI für deutsche Gerichte sind auf dem Europäischen Justizportal näher erläutert.¹

```
filenames <- basename(links.pdf)

filenames <- gsub(".pdf",
  "",
  filenames)

filenames <- gsub("\\;",
  "",
  filenames)

filenames <- gsub("\\-",
  "",
  filenames)
```

5.6.2 Einzelkorrekturen vornehmen

Soweit ich erkennen kann, handelt es sich bei dem “S” in der Ordinalzahl “280814U2WD20.13S.0” um einen Fehler im Link. Jedenfalls taucht das S nirgendwo in der Dokumentation der ECLI-Strukturen oder dem Volltext des Urteils selber auf.

```
filenames <- gsub("280814U2WD20.13S.0",
  "280814U2WD20.13.0",
  filenames)
```

Bei den folgenden Entscheidungen sind die Links fehlerhaft. Die korrekten ECLI-Ordinalzahlen für die beiden Entscheidungen stammen jeweils aus dem Text des PDF-Dokumentes.

```
filenames <- gsub("030206BPKH1.062.06.0",
  "030206B8PKH1.06.0",
  filenames)

filenames <- gsub("190405BVR1011.041012.04.0",
```

¹ https://e-justice.europa.eu/content_european_case_law_identifier_ecli-175-de-de.do?member=1

```
"190405B4VR1011.04.0",
filenames)
```

5.6.3 Transformation zu Variablen mit Unterstrichen

```
filenames1 <- gsub("([0-9]{2})([0-9]{2})([0-9]{2})([A-Z])([0-9]{1,2})([A-Za-z\\-]
*) ([0-9]*) [\\.\\.] ([0-9]*) ([D\\.\\.])",
  "BVerwG_\\3-\\2-\\1_\\4_\\5_\\6_\\7_\\8_\\9_",
  filenames)

filenames1 <- gsub("020320BGrSen1.19.0",
  "BVerwG_20-03-02_B_NA_GrSen_1_19_NA_0",
  filenames1)
```

5.6.4 Punkt zu NA

```
filenames1 <- gsub("\\\\.",
  "NA",
  filenames1)
```

5.7 Registerzeichen mit Bindestrichen anpassen

```
filenames1 <- gsub("WDSKSt",
  "WDS-KSt",
  filenames1)

filenames1 <- gsub("WDSVR",
  "WDS-VR",
  filenames1)

filenames1 <- gsub("WDSPKH",
  "WDS-PKH",
  filenames1)

filenames1 <- gsub("WDSAV",
  "WDS-AV",
  filenames1)

filenames1 <- gsub("DPKH",
  "D-PKH",
  filenames1)
```

5.8 Entscheidungsdatum in ISO-Format transformieren

```
variable.split <- tstrsplit(filenamees1,
                           split = "_")
setDT(variable.split)

date.split <- tstrsplit(variable.split$V2,
                       split = "-")
setDT(date.split)

date.split[, V1 := list(lapply(V1, function(x)f.year.iso(as.numeric(x))))]

date.combine <- paste(date.split$V1,
                      date.split$V2,
                      date.split$V3,
                      sep = "-")

variable.combine <- variable.split
variable.combine$V2 <- date.combine
```

5.9 Dateinamen finalisieren

```
filenames.final <- apply(variable.combine,
                         1,
                         function(x)paste(x, collapse = "_"))

filenames.final <- paste0(filenames.final,
                          ".pdf")
```

5.10 Strenge REGEX-Validierung der Dateinamen

5.10.1 REGEX-Validierung durchführen

```
regex.test <- grep(paste0("^BVerwG", # Gericht
                          "_",
                          "[0-9]{4}-[0-9]{2}-[0-9]{2}", # Datum
                          "_",
                          "[UBG]", # Entscheidungstyp
                          "_",
                          "[0-9NA]+", # Spruchkörper
                          "_",
                          "[A-Za-z-]+", # Registerzeichen
                          "_",
                          "[0-9]+", # Eingangsnummer
                          "_",
                          "[0-9]{1,2}", # Eingangsjahr
```

```

        "-",
        "[NAD]+", # Zusatz
        "-",
        "[0-9]", # Kollision
        "\\..pdf$"), # Dateiendung
filenames.final,
value = TRUE,
invert = TRUE)

```

5.10.2 Ergebnis der REGEX-Validierung

Hinweis: Das Ergebnis bei Erfolg sollte ein leerer Vektor sein!

```
print(regex.test)
```

```
## character(0)
```

5.10.3 Skript stoppen falls REGEX-Validierung gescheitert

```

if (length(regex.test) != 0){
  stop("REGEX VALIDIERUNG GESCHEITERT: DATEINAMEN ENTSPRECHEN NICHT DEM
CODEBOOK-SCHEMA!")
}

```

5.11 Data Table für Download erstellen

```

dt.download <- data.table(links.pdf,
                           filenames.final)

```

5.12 Duplikate bereinigen

5.12.1 Test auf Duplikate

```
sort(dt.download$filenames.final[duplicated(dt.download$filenames.final)])
```

```

## [1] "BVerwG_2007-01-11_B_1_WDS-VR_7_06_NA_0.pdf"
## [2] "BVerwG_2012-07-06_B_1_WDS-VR_5_12_NA_0.pdf"
## [3] "BVerwG_2012-07-18_B_1_WDS-VR_1_12_NA_0.pdf"
## [4] "BVerwG_2012-08-29_B_1_WDS-VR_3_12_NA_0.pdf"
## [5] "BVerwG_2014-08-28_U_2_WD_20_13_NA_0.pdf"

```

5.12.2 Von Duplikaten betroffene Links

Alle diese Links wurden manuell überprüft und es handelt sich tatsächlich um Duplikate. Bei den Entscheidungen mit dem Registerzeichen WDS-VR finden sich Links mit und ohne Trennstrich, beide verweisen aber auf die gleichen Dateien. Die Entscheidungen mit den Ordinalzahlen »280814U2WD20.13.0.pdf« und »280814U2WD20.13S.0.pdf« sind identisch, das »S« nach der 13 ist vermutlich ein Tippfehler, siehe auch weiter oben bei der Einzelkorrektur.

```
sort(dt.download$links.pdf[duplicated(dt.download$filenames.final,
                                     fromLast = FALSE)])
```

```
## [1] "https://www.bverwg.de/entscheidungen/pdf/060712B1WDS-VR5.12.0.pdf"
## [2] "https://www.bverwg.de/entscheidungen/pdf/110107B1WDS-VR7.06.0.pdf"
## [3] "https://www.bverwg.de/entscheidungen/pdf/180712B1WDS-VR1.12.0.pdf"
## [4] "https://www.bverwg.de/entscheidungen/pdf/280814U2WD20.13S.0.pdf"
## [5] "https://www.bverwg.de/entscheidungen/pdf/290812B1WDS-VR3.12.0.pdf"
```

```
sort(dt.download$links.pdf[duplicated(dt.download$filenames.final,
                                     fromLast = TRUE)])
```

```
## [1] "https://www.bverwg.de/entscheidungen/pdf/060712B1WDSVR5.12.0.pdf"
## [2] "https://www.bverwg.de/entscheidungen/pdf/110107B1WDSVR7.06.0.pdf"
## [3] "https://www.bverwg.de/entscheidungen/pdf/180712B1WDSVR1.12.0.pdf"
## [4] "https://www.bverwg.de/entscheidungen/pdf/280814U2WD20.13.0.pdf"
## [5] "https://www.bverwg.de/entscheidungen/pdf/290812B1WDSVR3.12.0.pdf"
```

5.12.3 Duplikate entfernen

```
dt.download <- dt.download[!duplicated(dt.download$filenames.final)]
```

5.13 [Debugging Modus] Reduktion des Download-Umfangs

```
print(mode.debug)
```

```
## [1] FALSE
```

```
if (mode.debug == TRUE){
  dt.download <- dt.download[sample(.N, debug.sample)]
}
```

5.14 Zeitstempel: Download Beginn

```
begin.download <- Sys.time()
print(begin.download)
```

```
## [1] "2021-10-19 02:53:08 CEST"
```

5.15 Download durchführen

```
for (i in sample(dt.download[,.N])){

  tryCatch({download.file(dt.download$links.pdf[i],
                          dt.download$filenames.final[i])
  },
  error = function(cond) {
    return(NA)}
  )

  Sys.sleep(runif(1, 0.3, 1))

}
```

```
## Warning in download.file(dt.download$links.pdf[i],
## dt.download$filenames.final[i]): downloaded length 0 != reported length 259
```

```
## Warning in download.file(dt.download$links.pdf[i],
## dt.download$filenames.final[i]): cannot open URL 'https://www.bverwg.de/
## entscheidungen/pdf/080610B1WB49.09-.0.pdf': HTTP status was '404 Not Found'
```

```
## Warning in download.file(dt.download$links.pdf[i],
## dt.download$filenames.final[i]): downloaded length 0 != reported length 259
```

```
## Warning in download.file(dt.download$links.pdf[i],
## dt.download$filenames.final[i]): cannot open URL 'https://www.bverwg.de/
## entscheidungen/pdf/030804U1C30.02.0.pdf': HTTP status was '404 Not Found'
```

5.16 Zeitstempel: Download Ende

```
end.download <- Sys.time()
print(end.download)
```



```
## [1] "2021-10-19 08:57:38 CEST"
```

5.17 Dauer: Download

```
end.download - begin.download
```

```
## Time difference of 6.074995 hours
```

5.18 [Debugging Modus] Löschen zufälliger Dateien

Dient dazu den Wiederholungsversuch zu testen.

```
print(mode.debug)
```

```
## [1] FALSE
```

```
if (mode.debug == TRUE){  
  files.pdf <- list.files(pattern = "\\..pdf")  
  unlink(sample(files.pdf, 5))  
}
```

5.19 Download-Ergebnis

5.19.1 Anzahl herunterzuladender Dateien

```
dt.download[,.N]
```

```
## [1] 25362
```

5.19.2 Anzahl heruntergeladener Dateien

```
files.pdf <- list.files(pattern = "\\..pdf")  
length(files.pdf)
```

```
## [1] 25360
```

5.19.3 Fehlbetrag

```
missing.N <- dt.download[,.N] - length(files.pdf)
print(missing.N)
```

```
## [1] 2
```

5.19.4 Fehlende Dateien

```
missing.names <- setdiff(dt.download$filenames.final,
                        files.pdf)
print(missing.names)
```

```
## [1] "BVerwG_2004-08-03_U_1_C_30_02_NA_0.pdf"
## [2] "BVerwG_2010-06-08_B_1_WB_49_09_NA_0.pdf"
```

5.20 Wiederholungsversuch: Download

Download für fehlende Dokumente wiederholen.

```
if(missing.N > 0){

  dt.retry <- dt.download[filenames.final %in% missing.names]

  for (i in 1:dt.retry[,.N]){

    response <- GET(dt.retry$links.pdf[i])

    Sys.sleep(runif(1, 1, 3))

    if (grepl("application/pdf", response$headers$"content-type") == TRUE &
        response$status_code == 200){
      tryCatch({download.file(url = dt.retry$links.pdf[i],
                             destfile = dt.retry$filenames.final[i])
            },
              error = function(cond) {
                return(NA)}
            )
    }else{
      print(paste0(dt.retry$filenames1[i], " : kein PDF vorhanden"))
    }
    Sys.sleep(runif(1, 2, 5))
  }
}
```

```
## [1] " : kein PDF vorhanden"  
## [1] " : kein PDF vorhanden"
```

5.21 Schlussbemerkung

Für die verbleibenden zwei Dateien waren auch nach manueller Überprüfung keine PDF-Datei auffindbar. Diese werden vorerst nicht in den Datensatz aufgenommen.

5.22 Download: Gesamtergebnis

5.22.1 Anzahl herunterzuladender Dateien

```
dt.download[,.N]
```

```
## [1] 25362
```

5.22.2 Anzahl heruntergeladener Dateien

```
files.pdf <- list.files(pattern = "\\\\.pdf")  
length(files.pdf)
```

```
## [1] 25360
```

5.22.3 Fehlbetrag

```
missing.N <- dt.download[,.N] - length(files.pdf)  
print(missing.N)
```

```
## [1] 2
```

5.22.4 Fehlende Dateien

```
missing.names <- setdiff(dt.download$filenames.final,  
                        files.pdf)  
print(missing.names)
```

```
## [1] "BVerwG_2004-08-03_U_1_C_30_02_NA_0.pdf"  
## [2] "BVerwG_2010-06-08_B_1_WB_49_09_NA_0.pdf"
```

6 Text-Extraktion

6.1 Vektor der zu extrahierenden Dateien erstellen

```
files.pdf <- list.files(pattern = "\\\\.pdf$",  
                        ignore.case = TRUE)
```

6.2 Anzahl zu extrahierender Dateien

```
length(files.pdf)
```

```
## [1] 25360
```

6.3 Keine Seitenzählung

Normalerweise würde an dieser Stelle eine Zählung der PDF-Seiten vorgenommen werden, diese führt aber zu Fehlermeldungen, weil die tatsächliche Anzahl Seiten von der Anzahl Seiten in den Metadaten bei vielen Dateien abweicht.

6.4 PDF extrahieren: Funktion anzeigen

```
print(f.dopar.pdfextract)
```

```
function(x, threads = detectCores()){
```

```
begin.extract <- Sys.time()  
  
print(paste("Parallel processing using", threads, "threads. Begin at", begin.  
extract))  
  
cl <- makeForkCluster(threads)  
registerDoParallel(cl)  
  
newnames <- gsub("\\\\.pdf",  
                "\\\\.txt",  
                x)  
  
result <- foreach(i = seq_along(x),  
                  .errorhandling = 'pass') %dopar% {  
  
    ## Extract text layer from PDF  
    pdf.extracted <- pdf_text(x[i])
```

```

        ## Write TXT to Disk
        write.table(pdf.extracted,
                    newnames[i],
                    quote = FALSE,
                    row.names = FALSE,
                    col.names = FALSE)
    }
stopCluster(cl)

end.extract <- Sys.time()
duration.extract <- end.extract - begin.extract

print(paste0("Processed ",
             length(result),
             " files. Runtime was ",
             round(duration.extract,
                   digits = 2),
             " ",
             attributes(duration.extract)$units,
             ". Ended at ",
             end.extract, "."))

return(result)
}

```

6.5 Text Extrahieren

```

result <- f.dopar.pdfextract(files.pdf,
                             threads = fullCores)

```

```

## [1] "Parallel processing using 16 threads. Begin at 2021-10-19 08:57:53"
## [1] "Processed 25360 files. Runtime was 1.44 mins. Ended at 2021-10-19
      08:59:19."

```

7 Korpus Erstellen

7.1 TXT-Dateien Einlesen

```
txt.bverwg <- readtext("./*.txt",
  docvarsfrom = "filenames",
  docvarnames = c("gericht",
    "datum",
    "entscheidung_typ",
    "spruchkoerper_az",
    "registerzeichen",
    "eingangsnummer",
    "eingangsjahr_az",
    "verzoegerung",
    "kollision"),
  dvsep = "_",
  encoding = "UTF-8")
```

7.2 In Data Table umwandeln

```
setDT(txt.bverwg)
```

7.3 Durch Zeilenumbruch getrennte Wörter zusammenfügen

Durch Zeilenumbrüche getrennte Wörter stellen bei aus PDF-Dateien gewonnene Text-Korpora ein erhebliches Problem dar. Wörter werden dadurch in zwei sinnentleerte Tokens getrennt, statt ein einzelnes und sinnvolles Token zu bilden. Dieser Schritt entfernt die Bindestriche, den Zeilenumbruch und ggf. dazwischenliegende Leerzeichen.

7.3.1 Funktion anzeigen

```
print(f.hyphen.remove)
```

```
## function(text){
##   ## Examples: Ham-\nburg, Mei-\n   nungsäußerung
##   text.out <- gsub("([a-zöäüß])-[:blank:]*\n[:blank:]*([a-zöäüß])",
##     "\\1\\2",
##     text)
##   ## Examples: SARS-CoV-\n2
##   text.out <- gsub("([a-zA-ZöäüÖÄÜß])-[:blank:]*\n[:blank:]*([A-Z0-9ÖÄÜß
  ])",
##     "\\1-\\2",
##     text.out)
##   ## Example: hat-   2\nte, Unsterb-   6\nliche
##   text.out <- gsub("([a-zöäüß])-[:blank:]*[0-9]+[:blank:]*\n[:blank:]*
  ([a-zöäüß])",
```

```
##           "\\1\\2",
##           text.out)
##
##   ## Example: hat- \n 2 te, Unsterb- \n 6 liche
##   text.out <- gsub("([a-zöäüß])-[:space:]*[0-9]+[:blank:]*([a-zöäüß))",
##                   "\\1\\2",
##                   text.out)
##
##   return(text.out)
## }
```

7.3.2 Funktion ausführen

```
txt.bverwg[, text := lapply(.(text), f.hyphen.remove)]
```

7.4 Variable “datum” als Datentyp “IDate” kennzeichnen

```
txt.bverwg$datum <- as.IDate(txt.bverwg$datum)
```

7.5 Variable “entscheidungsjahr” hinzufügen

```
txt.bverwg$entscheidungsjahr <- year(txt.bverwg$datum)
```

7.6 Variable “eingangsjahr_iso” hinzufügen

```
txt.bverwg$eingangsjahr_iso <- f.year.iso(txt.bverwg$eingangsjahr_az)
```

7.7 Datensatz nach Datum sortieren

Aufgrund der Position der Datums-Variable ist der Datensatz vermutlich schon von Linux nach Datum sortiert worden. Die Erstellung der Variablen für Präsidenten und Vize-Präsidenten trifft allerdings die starke Annahme, dass eine aufsteigende Sortierung nach Datum besteht. Wäre das nicht der Fall, würden dort Fehler auftreten. Diese Sortierung ist als fail-safe gedacht.

```
setorder(txt.bverwg,
          datum)
```

7.8 Variable “praesi” hinzufügen

Diese Variable dokumentiert für jede Entscheidung welche/r Präsident:in am Tag der Entscheidung im Amt war.

Die Personendaten stammen aus folgendem Datensatz: »Seán Fobbe and Tilko Swalve (2021). Presidents and Vice-Presidents of the Federal Courts of Germany (PVP-FCG). Version 2021-04-08. Zenodo. DOI: 10.5281/zenodo.4568682«.

7.8.1 Personaldaten einlesen

```
praesi <- fread("PVP-FCG_2021-04-08_GermanFederalCourts_Presidents.csv")
praesi <- praesi[court == "BVerwG", c(1:3, 5:6)]
```

7.8.2 Personaldaten anzeigen

```
kable(praesi,
      format = "latex",
      align = "r",
      booktabs = TRUE,
      longtable = TRUE) %>% kable_styling(latex_options = "repeat_header")
```

court	name_last	name_first	term_begin_date	term_end_date
BVerwG	Frege	Ludwig	1953-03-28	1954-12-31
BVerwG	VACANCY-1	VACANCY-1	1955-01-01	1955-04-28
BVerwG	Egidi	Hans	1955-04-29	1958-06-30
BVerwG	VACANCY-2	VACANCY-2	1958-07-01	1958-07-17
BVerwG	Werner	Fritz	1958-07-18	1969-12-26
BVerwG	VACANCY-3	VACANCY-3	1969-12-27	1970-06-14
BVerwG	Zeidler	Wolfgang	1970-06-15	1975-11-07
BVerwG	VACANCY-4	VACANCY-4	1975-11-08	1976-08-18
BVerwG	Fürst	Walther	1976-08-19	1980-02-29
BVerwG	Sendler	Horst	1980-03-01	1991-06-30
BVerwG	Franßen	Everhardt	1991-07-01	2002-09-30
BVerwG	Hien	Eckart	2002-10-01	2007-05-31
BVerwG	Eckertz-Höfer	Marion	2007-06-01	2014-01-31
BVerwG	VACANCY-5	VACANCY-5	2014-02-01	2014-06-30
BVerwG	Rennert	Klaus	2014-07-01	NA

7.8.3 Hypothetisches Amtsende für Präsident:in

Weil der/die aktuelle Präsident:in noch im Amt ist, ist der Wert für das Amtsende “NA”. Dieser ist aber für die verwendete Logik nicht greifbar, weshalb an dieser Stelle ein hypothetisches Amtsende in einem Jahr ab dem Tag der Datensatzerstellung fingiert wird. Es wird nur an dieser Stelle verwendet und danach verworfen.

```
praesi[is.na(term_end_date)]$term_end_date <- Sys.Date() + 365
```

7.8.4 Schleife vorbereiten

```
N <- praesi[, .N]

praesi.list <- vector("list", N)
```

7.8.5 Vektor erstellen

```
for (i in seq_len(N)){
  praesi.N <- txt.bverwg[datum >= praesi$term_begin_date[i] & datum <= praesi$
    term_end_date[i], .N]
  praesi.list[[i]] <- rep(praesi$name_last[i],
    praesi.N)
}
```

7.8.6 Vektor einfügen

```
txt.bverwg$praesi <- unlist(praesi.list)
```

7.9 Variable “v_praesi” hinzufügen

Diese Variable dokumentiert für jede Entscheidung welche/r Vize-Präsident:in am Tag der Entscheidung im Amt war.

Die Personendaten stammen aus folgendem Datensatz: »Seán Fobbe and Tilko Swalve (2021). Presidents and Vice-Presidents of the Federal Courts of Germany (PVP-FCG). Version 2021-04-08. Zenodo. DOI: 10.5281/zenodo.4568682«.

7.9.1 Personaldaten einlesen

```
vpPraesi <- fread("PVP-FCG_2021-04-08_GermanFederalCourts_VicePresidents.csv")
vpPraesi <- vpPraesi[court == "BVerwG", c(1:3, 5:6)]
```

7.9.2 Personaldaten anzeigen

```
kable(vpraesi,
      format = "latex",
      align = "r",
      booktabs = TRUE,
      longtable = TRUE) %>% kable_styling(latex_options = "repeat_header")
```

court	name_last	name_first	term_begin_date	term_end_date
BVerwG	Külz	Helmut R.	1970-12-23	1971-07-31
BVerwG	VACANCY-1	VACANCY-1	1971-08-01	1971-11-15
BVerwG	Fürst	Walther	1971-11-16	1976-08-18
BVerwG	Sendler	Horst	1976-08-19	1980-02-29
BVerwG	Oppenheimer	Johannes	1980-03-01	1986-07-31
BVerwG	Zehner	Günter	1986-08-01	1990-08-31
BVerwG	Schlichter	Otto	1990-09-01	1993-09-30
BVerwG	Franke	Ingeborg	1993-10-01	2000-05-31
BVerwG	VACANCY-2	VACANCY-2	2000-06-01	2000-06-21
BVerwG	Hien	Eckart	2000-06-22	2002-09-30
BVerwG	Eckertz-Höfer	Marion	2002-10-01	2007-05-31
BVerwG	Hund	Michael	2007-06-01	2011-10-31
BVerwG	VACANCY-3	VACANCY-3	2011-11-01	2012-11-20
BVerwG	Rennert	Klaus	2012-11-21	2014-06-30
BVerwG	Christ	Josef	2014-07-01	2017-11-30
BVerwG	VACANCY-4	VACANCY-4	2017-12-01	2019-05-21
BVerwG	Korbmacher	Andreas	2019-05-22	NA

7.9.3 Hypothetisches Amtsende für Vize-Präsident:in

Weil der/die aktuelle Vize-Präsident:in noch im Amt ist, ist der Wert für das Amtsende “NA”. Dieser ist aber für die verwendete Logik nicht greifbar, weshalb an dieser Stelle ein hypothetisches Amtsende in einem Jahr ab dem Tag der Datensatzerstellung fingiert wird. Es wird nur an dieser Stelle verwendet und danach verworfen.

```
vpraesi[is.na(term_end_date)]$term_end_date <- Sys.Date() + 365
```

7.9.4 Schleife vorbereiten

```
N <- vpraesi[,.N]

vpraesi.list <- vector("list", N)
```

7.9.5 Vektor erstellen

```
for (i in seq_len(N)){
  vpraesi.N <- txt.bverwg[datum >= vpraesi$term_begin_date[i] & datum <=
vpraesi$term_end_date[i], .N]
  vpraesi.list[[i]] <- rep(vpraesi$name_last[i],
                          vpraesi.N)
}
```

7.9.6 Vektor einfügen

```
txt.bverwg$v_praesi <- unlist(vpraesi.list)
```

7.10 Variable “aktenzeichen” hinzufügen

```
txt.bverwg$aktenzeichen <- paste0("BVerwG",
                                " ",
                                txt.bverwg$spruchkoerper_az,
                                " ",
                                txt.bverwg$registerzeichen,
                                " ",
                                txt.bverwg$eingangsnummer,
                                ". ",
                                txt.bverwg$eingangsjahr_az)

txt.bverwg[verzoeigerung == "D"]$aktenzeichen <- paste(txt.bverwg[verzoeigerung ==
"D"]$aktenzeichen,
                                                       "D")

txt.bverwg$aktenzeichen <- gsub(" NA ",
                                " ",
                                txt.bverwg$aktenzeichen)
```

7.11 Variable “ecli” hinzufügen

Struktur und Inhalt der ECLI für deutsche Gerichte sind auf dem Europäischen Justizportal näher erläutert.²

² https://e-justice.europa.eu/content_european_case_law_identifier_ecli-175-de-de.do?member=1

Sofern die Variablen korrekt extrahiert wurden lässt sich die ECLI vollständig rekonstruieren.

7.11.1 Ordinalzahl erstellen

```
ecli.ordinalzahl <- paste0(format(txt.bverwg$datum,
                                "%d%m%y"),
                           txt.bverwg$entscheidung_typ,
                           txt.bverwg$spruchkoerper_az,
                           gsub("-",
                                "",
                                txt.bverwg$registerzeichen),
                           txt.bverwg$eingangsnummer,
                           ".",
                           formatC(txt.bverwg$eingangsjahr_az,
                                    width = 2,
                                    flag = "0"),
                           ifelse(is.na(txt.bverwg$verzoeigerung),
                                   ".",
                                   txt.bverwg$verzoeigerung),
                           txt.bverwg$kollision)

ecli.ordinalzahl <- gsub("NA",
                        "",
                        ecli.ordinalzahl)
```

7.11.2 Tabelle der Soll- und Ist-Ordinalzahlen vorbereiten

Die im vorigen Schritt generierten ECLI-Ordinalzahlen (“ist”) werden nun noch einmal mit den in den Links angegebenen ECLI-Ordinalzahlen (“soll”) gegenübergestellt um sicherzustellen, dass es sich um gültige ECLIs handelt.

Berücksichtigt werden im folgenden nur Dateien die tatsächlich heruntergeladen wurden.

```
files.pdf <- list.files(pattern = "\\\\.pdf$",
                        ignore.case = TRUE)

eclitest.index <- dt.download$filenames.final %in% files.pdf

eclitest.links <- dt.download$links.pdf[eclitest.index]
eclitest.pdf <- dt.download$filenames.final[eclitest.index]

soll <- basename(eclitest.links)
```

Bei einigen PDF-Dateien sind die Soll-Ordinalzahlen (z.B. “010806B1WDS-VR3.06.0” und “130607B1WDS-VR2.07.0”) in den Links fehlerhaft, weil Bindestriche im ECLI-System nicht vorgesehen sind. Für diese Dateien werden vor dem Abgleich Korrekturen vorgenommen.

```
soll <- gsub("\\\\.pdf",
```

```

        "",
        soll)

soll <- gsub(";",
            "",
            soll)

soll <- gsub("WDS-([VRAVPKH]{1,3})",
            "WDS\\1",
            soll)

soll <- gsub("D-PKH",
            "DPKH",
            soll)

```

Bei diesen Entscheidungen sind die Links fehlerhaft. Die korrekten ECLI-Ordinalzahlen für die beiden Entscheidungen stammen jeweils aus dem Text des PDF-Dokumentes.

```

soll <- gsub("030206BPKH1.062.06.0",
            "030206B8PKH1.06.0",
            soll)

soll <- gsub("190405BVR1011.041012.04.0",
            "190405B4VR1011.04.0",
            soll)

```

7.11.3 Tabelle der Soll- und Ist-Ordinalzahlen erstellen

```

ist <- eclli.ordinalzahl

soll <- soll[match(files.pdf,
                  eclitest.pdf)]

ecli.o.check <- data.table(soll,
                          ist)

print(ecli.o.check)

```

##		soll	ist
##	1:	260297U6C3.96.0	260297U6C3.96.0
##	2:	141200B1WB107.00.0	141200B1WB107.00.0
##	3:	100102B9A9.02.0	100102B9A9.02.0
##	4:	140102B4BN1.02.0	140102B4BN1.02.0
##	5:	170102B1B12.02.0	170102B1B12.02.0
##	---		
##	25356:	170921B8B14.21.0	170921B8B14.21.0
##	25357:	210921B1AV5.21.0	210921B1AV5.21.0
##	25358:	210921B4BN2.21.0	210921B4BN2.21.0
##	25359:	210921B8B4.21.0	210921B8B4.21.0
##	25360:	270921B6BN1.21.0	270921B6BN1.21.0

7.11.4 Prüfung ob alle ECLI-Ordinalzahlen korrekt sind

```
identical(ecli.o.check$soll,  
          ecli.o.check$ist)
```

```
## [1] TRUE
```

7.11.5 Fehlerhafte ECLI-Ordinalzahlen anzeigen

```
diff <- setdiff(ecli.o.check$soll, ecli.o.check$ist)  
ecli.o.check[soll %in% diff]
```

```
## Empty data.table (0 rows and 2 cols): soll,ist
```

7.11.6 Vollständige ECLI erstellen und einfügen

```
txt.bverwg$ecli <- paste0("ECLI:DE:BVerwG:",  
                          txt.bverwg$entscheidungsjahr,  
                          ":",  
                          ecli.ordinalzahl)
```

7.11.7 ECLI-Beispiele

```
sample(txt.bverwg$ecli,  
       10)
```

```
## [1] "ECLI:DE:BVerwG:2017:151217B2B46.17.0"  
## [2] "ECLI:DE:BVerwG:2015:170915U2C26.14.0"  
## [3] "ECLI:DE:BVerwG:2016:141216B5PKH48.16.0"  
## [4] "ECLI:DE:BVerwG:2002:221002B3PKH18.02.0"  
## [5] "ECLI:DE:BVerwG:2017:040517U2C52.16.0"  
## [6] "ECLI:DE:BVerwG:2008:280408B7B16.08.0"  
## [7] "ECLI:DE:BVerwG:2011:170511B8B98.10.0"  
## [8] "ECLI:DE:BVerwG:2006:150306B4PKH1.06.0"  
## [9] "ECLI:DE:BVerwG:2015:050215B5KSt2.15.0"  
## [10] "ECLI:DE:BVerwG:2010:080910B7B57.10.0"
```

7.12 Variable “verfahrensart” hinzufügen

Die Registerzeichen werden an dieser Stelle mit ihren detaillierten Bedeutungen aus dem folgenden Datensatz abgeglichen: “Seán Fobbe (2021). Aktenzeichen der Bundesrepublik Deutschland (AZ-BRD). Version 1.0.1. Zenodo. DOI: 10.5281/zenodo.4569564.” Das Ergebnis des Abgleichs wird in der Variable “verfahrensart” in den Datensatz eingefügt.

7.12.1 Datensatz einlesen

```
az.source <- fread("AZ-BRD_1-0-1_DE_Registerzeichen_Datensatz.csv")
```

7.12.2 Datensatz auf relevante Daten reduzieren

```
az.bverwg <- az.source[stelle == "BVerwG" & position == "hauptzeichen"]
```

7.12.3 Indizes bestimmen

```
targetindices <- match(txt.bverwg$registerzeichen,  
  az.bverwg$zeichen_code)
```

7.12.4 Vektor der Verfahrensarten erstellen und einfügen

```
txt.bverwg$verfahrensart <- az.bverwg$bedeutung[targetindices]
```

7.13 Variable “doi_concept” hinzufügen

```
txt.bverwg$doi_concept <- rep(doi.concept,  
  txt.bverwg[,.N])
```

7.14 Variable “doi_version” hinzufügen

```
txt.bverwg$doi_version <- rep(doi.version,  
  txt.bverwg[,.N])
```

7.15 Variable “version” hinzufügen

```
txt.bverwg$version <- as.character(rep(datestamp,  
  txt.bverwg[,.N]))
```

7.16 Variable “lizenz” hinzufügen

```
txt.bverwg$lizenz <- as.character(rep(license,  
  txt.bverwg[,.N]))
```

8 Frequenztabellen erstellen

8.1 Funktion anzeigen

```
print(f.fast.freqtable)
```

```
function(x, varlist = names(x), sumrow = TRUE, output.list = TRUE, output.kable = FALSE, output.csv = FALSE, outputdir = "./", prefix = „“, align = "r"){
```

```
## Begin List
freqtable.list <- vector("list", length(varlist))

## Calculate Frequency Table
for (i in seq_along(varlist)){

  varname <- varlist[i]

  freqtable <- x[, .N, keyby=c(paste0(varname))]

  freqtable[, c("exactpercent",
               "roundedpercent",
               "cumulpercent") := {
    exactpercent <- N/sum(N)*100
    roundedpercent <- round(exactpercent, 2)
    cumulpercent <- round(cumsum(exactpercent), 2)
    list(exactpercent,
         roundedpercent,
         cumulpercent)}]

  ## Calculate Summary Row
  if (sumrow == TRUE){
    colsums <- cbind("Total",
                    freqtable[, lapply(.SD, function(x){round(sum(x))}),
                      .SDcols = c("N",
                                   "exactpercent",
                                   "roundedpercent")
                    ], round(max(freqtable$cumulpercent)))

    colnames(colsums)[c(1,5)] <- c(varname, "cumulpercent")
    freqtable <- rbind(freqtable, colsums)
  }

  ## Add Frequency Table to List
  freqtable.list[[i]] <- freqtable

  ## Write CSV
  if (output.csv == TRUE){

    fwrite(freqtable,
           paste0(outputdir,
                  prefix,
                  varname,
```



```

        ".csv"),
        na = "NA")

}

## Output Kable
if (output.kable == TRUE){

  cat("\n-----\n")
  cat(paste0("Frequency Table for Variable:  ", varname, "\n"))
  cat("-----\n")
  cat(paste0("\n ",
             x[, .N, keyby=c(paste0(varname))][, .N],
             " unique value(s) detected.\n\n"))

  print(kable(freqtable,
              format = "latex",
              align = align,
              booktabs = TRUE,
              longtable = TRUE) %>% kable_styling(latex_options = "repeat_
header"))
}
}

## Return List of Frequency Tables
if (output.list == TRUE){
  return(freqtable.list)
}

}

```

8.2 Ignorierte Variablen

```
print(varremove)
```

```
## [1] "text"          "eingangsnummer" "datum"          "doc_id"
## [5] "ecli"          "aktenzeichen"
```

8.3 Liste zu prüfender Variablen

```
varlist <- names(txt.bverwg)

varlist <- setdiff(varlist,
                  varremove)

print(varlist)
```

```
## [1] "gericht"      "entscheidung_typ" "spruchkoerper_az"
## [4] "registerzeichen" "eingangsjahr_az"  "verzoegerung"
## [7] "kollision"      "entscheidungsjahr" "eingangsjahr_iso"
## [10] "praesi"         "v_praesi"         "verfahrensart"
## [13] "doi_concept"    "doi_version"      "version"
## [16] "lizenz"
```

8.4 Frequenztabellen erstellen

```
prefix <- paste0(datasetname,
  "_01_Frequenztafel_var-")
```

```
f.fast.freqtable(txt.bverwg,
  varlist = varlist,
  sumrow = TRUE,
  output.list = FALSE,
  output.kable = TRUE,
  output.csv = TRUE,
  outputdir = outputdir,
  prefix = prefix,
  align = c("p{5cm}",
    rep("r", 4)))
```

Frequency Table for Variable: gericht

1 unique value(s) detected.

gericht	N	exactpercent	roundedpercent	cumulpercent
BVerwG	25360	100	100	100
Total	25360	100	100	100

Frequency Table for Variable: entscheidung_typ

3 unique value(s) detected.

entscheidung_typ	N	exactpercent	roundedpercent	cumulpercent
B	20913	82.4645110	82.46	82.46
G	21	0.0828076	0.08	82.55

(continued)

entscheidung_typ	N	exactpercent	roundedpercent	cumulpercent
U	4426	17.4526814	17.45	100.00
Total	25360	100.0000000	100.00	100.00

Frequency Table for Variable: spruchkoerper_az

13 unique value(s) detected.

spruchkoerper_az	N	exactpercent	roundedpercent	cumulpercent
NA	1	0.0039432	0.00	0.00
1	3594	14.1719243	14.17	14.18
2	3430	13.5252366	13.53	27.70
3	2530	9.9763407	9.98	37.68
4	2960	11.6719243	11.67	49.35
5	2695	10.6269716	10.63	59.98
6	2553	10.0670347	10.07	70.04
7	1842	7.2634069	7.26	77.31
8	2243	8.8446372	8.84	86.15
9	2208	8.7066246	8.71	94.86
10	1010	3.9826498	3.98	98.84
20	292	1.1514196	1.15	99.99
30	2	0.0078864	0.01	100.00
Total	25360	100.0000000	100.00	100.00

Frequency Table for Variable: registerzeichen

30 unique value(s) detected.

registerzeichen	N	exactpercent	roundedpercent	cumulpercent
A	1310	5.1656151	5.17	5.17

(continued)

registerzeichen	N	exactpercent	roundedpercent	cumulpercent
AV	182	0.7176656	0.72	5.88
B	14006	55.2287066	55.23	61.11
BN	1244	4.9053628	4.91	66.02
C	4656	18.3596215	18.36	84.38
CN	175	0.6900631	0.69	85.07
D	99	0.3903785	0.39	85.46
D-PKH	1	0.0039432	0.00	85.46
DB	53	0.2089905	0.21	85.67
DW	3	0.0118297	0.01	85.68
F	285	1.1238170	1.12	86.81
GS	2	0.0078864	0.01	86.81
GrSen	1	0.0039432	0.00	86.82
KSt	185	0.7294953	0.73	87.55
P	241	0.9503155	0.95	88.50
PB	386	1.5220820	1.52	90.02
PKH	579	2.2831230	2.28	92.30
VR	512	2.0189274	2.02	94.32
WA	2	0.0078864	0.01	94.33
WB	702	2.7681388	2.77	97.10
WD	354	1.3958991	1.40	98.49
WDB	87	0.3430599	0.34	98.84
WDS-AV	4	0.0157729	0.02	98.85
WDS-KSt	4	0.0157729	0.02	98.87
WDS-PKH	1	0.0039432	0.00	98.87
WDS-VR	142	0.5599369	0.56	99.43
WDW	1	0.0039432	0.00	99.44
WNB	108	0.4258675	0.43	99.86
WRB	31	0.1222397	0.12	99.98
WVR	4	0.0157729	0.02	100.00

(continued)

registerzeichen	N	exactpercent	roundedpercent	cumulpercent
Total	25360	100.0000000	100.00	100.00

Frequency Table for Variable: eingangsjahr_az

23 unique value(s) detected.

eingangsjahr_az	N	exactpercent	roundedpercent	cumulpercent
0	1	0.0039432	0.00	0.00
1	2	0.0078864	0.01	0.01
2	2269	8.9471609	8.95	8.96
3	1948	7.6813880	7.68	16.64
4	1822	7.1845426	7.18	23.82
5	1575	6.2105678	6.21	30.04
6	1670	6.5851735	6.59	36.62
7	1794	7.0741325	7.07	43.69
8	1450	5.7176656	5.72	49.41
9	1440	5.6782334	5.68	55.09
10	1445	5.6979495	5.70	60.79
11	1493	5.8872240	5.89	66.68
12	1293	5.0985804	5.10	71.77
13	964	3.8012618	3.80	75.58
14	864	3.4069401	3.41	78.98
15	936	3.6908517	3.69	82.67
16	1020	4.0220820	4.02	86.70
17	869	3.4266562	3.43	90.12
18	836	3.2965300	3.30	93.42
19	856	3.3753943	3.38	96.79
20	680	2.6813880	2.68	99.48
21	132	0.5205047	0.52	100.00

(continued)

eingangsjahr_az	N	exactpercent	roundedpercent	cumulpercent
96	1	0.0039432	0.00	100.00
Total	25360	100.0000000	100.00	100.00

Frequency Table for Variable: verzoegerung

2 unique value(s) detected.

verzoegerung	N	exactpercent	roundedpercent	cumulpercent
NA	25290	99.7239748	99.72	99.72
D	70	0.2760252	0.28	100.00
Total	25360	100.0000000	100.00	100.00

Frequency Table for Variable: kollision

2 unique value(s) detected.

kollision	N	exactpercent	roundedpercent	cumulpercent
0	25341	99.9250789	99.93	99.93
1	19	0.0749211	0.07	100.00
Total	25360	100.0000000	100.00	100.00

Frequency Table for Variable: entscheidungsjahr

22 unique value(s) detected.

entscheidungsjahr	N	exactpercent	roundedpercent	cumulpercent
1997	1	0.0039432	0.00	0.00
2000	1	0.0039432	0.00	0.01
2002	1502	5.9227129	5.92	5.93
2003	2009	7.9219243	7.92	13.85

(continued)

entscheidungsjahr	N	exactpercent	roundedpercent	cumulpercent
2004	1817	7.1648265	7.16	21.02
2005	1619	6.3840694	6.38	27.40
2006	1684	6.6403785	6.64	34.04
2007	1690	6.6640379	6.66	40.71
2008	1549	6.1080442	6.11	46.81
2009	1352	5.3312303	5.33	52.15
2010	1474	5.8123028	5.81	57.96
2011	1553	6.1238170	6.12	64.08
2012	1281	5.0512618	5.05	69.13
2013	1072	4.2271293	4.23	73.36
2014	889	3.5055205	3.51	76.87
2015	845	3.3320189	3.33	80.20
2016	995	3.9235016	3.92	84.12
2017	960	3.7854890	3.79	87.91
2018	836	3.2965300	3.30	91.20
2019	887	3.4976341	3.50	94.70
2020	871	3.4345426	3.43	98.13
2021	473	1.8651420	1.87	100.00
Total	25360	100.0000000	100.00	100.00

Frequency Table for Variable: eingangsjahr_iso

23 unique value(s) detected.

eingangsjahr_iso	N	exactpercent	roundedpercent	cumulpercent
1996	1	0.0039432	0.00	0.00
2000	1	0.0039432	0.00	0.01
2001	2	0.0078864	0.01	0.02
2002	2269	8.9471609	8.95	8.96

(continued)

eingangsjahr_iso	N	exactpercent	roundedpercent	cumulpercent
2003	1948	7.6813880	7.68	16.64
2004	1822	7.1845426	7.18	23.83
2005	1575	6.2105678	6.21	30.04
2006	1670	6.5851735	6.59	36.62
2007	1794	7.0741325	7.07	43.70
2008	1450	5.7176656	5.72	49.42
2009	1440	5.6782334	5.68	55.09
2010	1445	5.6979495	5.70	60.79
2011	1493	5.8872240	5.89	66.68
2012	1293	5.0985804	5.10	71.78
2013	964	3.8012618	3.80	75.58
2014	864	3.4069401	3.41	78.99
2015	936	3.6908517	3.69	82.68
2016	1020	4.0220820	4.02	86.70
2017	869	3.4266562	3.43	90.13
2018	836	3.2965300	3.30	93.42
2019	856	3.3753943	3.38	96.80
2020	680	2.6813880	2.68	99.48
2021	132	0.5205047	0.52	100.00
Total	25360	100.0000000	100.00	100.00

Frequency Table for Variable: praesi

5 unique value(s) detected.

praesi	N	exactpercent	roundedpercent	cumulpercent
Eckertz-Höfer	9295	36.652208	36.65	36.65
Franßen	1049	4.136435	4.14	40.79
Hien	8346	32.910095	32.91	73.70

(continued)

praesi	N	exactpercent	roundedpercent	cumulpercent
Rennert	6273	24.735804	24.74	98.43
VACANCY-5	397	1.565457	1.57	100.00
Total	25360	100.000000	100.00	100.00

Frequency Table for Variable: v_praesi

9 unique value(s) detected.

v_praesi	N	exactpercent	roundedpercent	cumulpercent
Christ	3117	12.2910095	12.29	12.29
Eckertz-Höfer	8346	32.9100946	32.91	45.20
Franke	1	0.0039432	0.00	45.21
Hien	1048	4.1324921	4.13	49.34
Hund	6598	26.0173502	26.02	75.35
Korbmacher	1872	7.3817035	7.38	82.74
Rennert	1696	6.6876972	6.69	89.42
VACANCY-3	1398	5.5126183	5.51	94.94
VACANCY-4	1284	5.0630915	5.06	100.00
Total	25360	100.0000000	100.00	100.00

Frequency Table for Variable: verfahrensart

29 unique value(s) detected.

verfahrensart	N	exactpercent	roundedpercent	cumulpercent
NA	6	0.0236593	0.02	0.02
Allgemeine Verfahren (Wehr- dienstsenate)	4	0.0157729	0.02	0.04

(continued)

verfahrensart	N	exactpercent	roundedpercent	cumulpercent
Allgemeine Verfahren (sonstige Anträge außerhalb eines schwebenden Verfahrens)	182	0.7176656	0.72	0.76
Berufung (Wehrdisziplinarsachen)	354	1.3958991	1.40	2.15
Beschwerde (Disziplinarsachen)	53	0.2089905	0.21	2.36
Erstinstanzliche Klage, inklusive Wiederaufnahmeverfahren (Verwaltungsstreitsachen)	1310	5.1656151	5.17	7.53
Erstinstanzliche Klage, inklusive Wiederaufnahmeverfahren (Wehrdienstsenat)	2	0.0078864	0.01	7.54
Fachsenat (Verwaltungsstreitsachen)	285	1.1238170	1.12	8.66
Großer Senat des Bundesverwaltungsgerichts	1	0.0039432	0.00	8.66
Kostensachen	185	0.7294953	0.73	9.39
Kostensachen (Wehrdienstsenate)	4	0.0157729	0.02	9.41
Nichtzulassungsbeschwerde (Normenkontrollsachen)	1244	4.9053628	4.91	14.31
Nichtzulassungsbeschwerde (Personalvertretungs- und Richtervertretungssachen)	386	1.5220820	1.52	15.84
Nichtzulassungsbeschwerde nach Wehrbeschwerdeordnung	108	0.4258675	0.43	16.26
Nichtzulassungsbeschwerde oder Beschwerde (Verwaltungsstreitsachen)	14006	55.2287066	55.23	71.49
Prozesskostenhilfe	579	2.2831230	2.28	73.77
Prozesskostenhilfe (Disziplinarsachen)	1	0.0039432	0.00	73.78
Prozesskostenhilfe (Wehrdienstsenate)	1	0.0039432	0.00	73.78

(continued)

verfahrensart	N	exactpercent	roundedpercent	cumulpercent
Rechtsbeschwerde (Personalvertretungs- und Richterververtretungssachen)	241	0.9503155	0.95	74.73
Rechtsbeschwerde nach Wehr- beschwerdeordnung	31	0.1222397	0.12	74.85
Rechtsmittelverfahren über Anträge, Beschwerden und Vorlagen nach der Wehrdiszi- plinarordnung	87	0.3430599	0.34	75.20
Revision (Disziplinarsachen)	99	0.3903785	0.39	75.59
Revision (Normenkontrollsa- chen)	175	0.6900631	0.69	76.28
Revision (Verwaltungsstreitsa- chen)	4656	18.3596215	18.36	94.64
Verfahren nach Wehrbeschwer- deordnung	702	2.7681388	2.77	97.41
Vorläufiger Rechtsschutz	512	2.0189274	2.02	99.42
Vorläufiger Rechtsschutz (Wehrdienstsenate)	142	0.5599369	0.56	99.98
Wiederaufnahme (Disziplinar- sachen)	3	0.0118297	0.01	100.00
Wiederaufnahme (Wehrdiszi- plinarsachen)	1	0.0039432	0.00	100.00
Total	25360	100.0000000	100.00	100.00

Frequency Table for Variable: doi_concept

1 unique value(s) detected.

doi_concept	N	exactpercent	roundedpercent	cumulpercent
10.5281/zenodo.3911067	25360	100	100	100
Total	25360	100	100	100

Frequency Table for Variable: doi_version

1 unique value(s) detected.

doi_version	N	exactpercent	roundedpercent	cumulpercent
10.5281/zenodo.5576822	25360	100	100	100
Total	25360	100	100	100

Frequency Table for Variable: version

1 unique value(s) detected.

version	N	exactpercent	roundedpercent	cumulpercent
2021-10-19	25360	100	100	100
Total	25360	100	100	100

Frequency Table for Variable: lizenz

1 unique value(s) detected.

lizenz	N	exactpercent	roundedpercent	cumulpercent
Creative Commons Zero 1.0 Universal	25360	100	100	100
Total	25360	100	100	100

9 Frequenztabellen visualisieren

9.1 Präfix erstellen

```
prefix <- paste0("ANALYSE/",  
                 datasetname,  
                 "_01_Frequenztafel_var-")
```

9.2 Tabellen einlesen

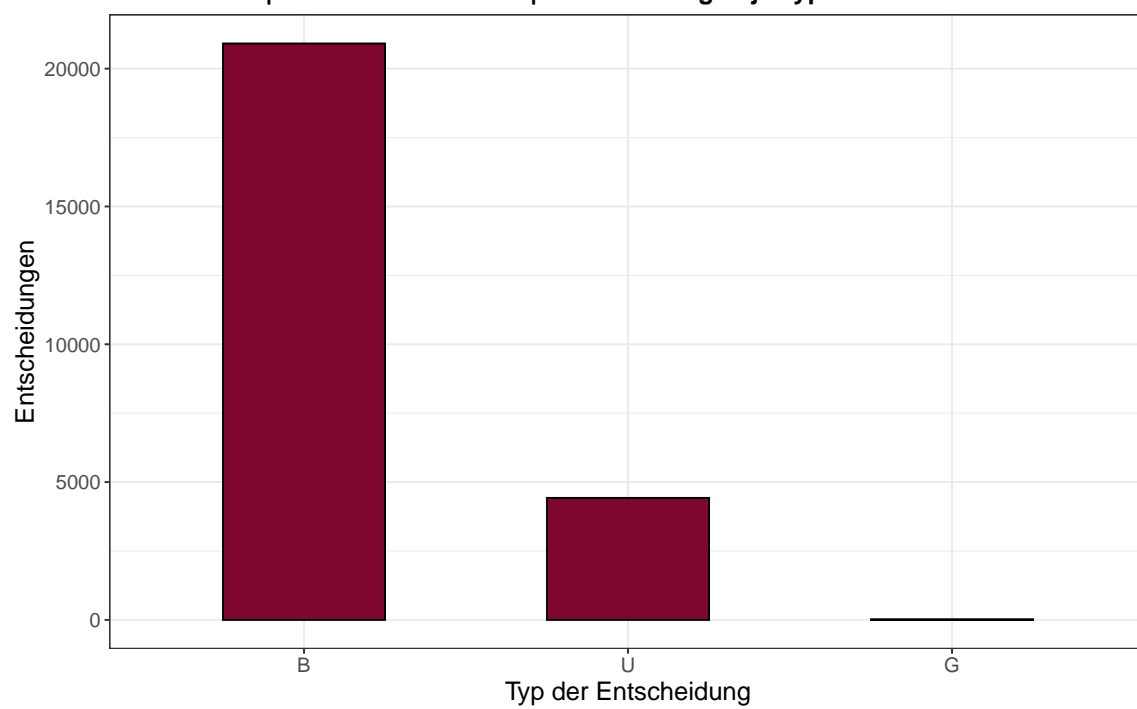
```
table.entsch.typ <- fread(paste0(prefix,  
                                  "entscheidung_typ.csv"))  
  
table.spruch.az <- fread(paste0(prefix,  
                                  "spruchkoerper_az.csv"))  
  
table.regz <- fread(paste0(prefix,  
                             "registerzeichen.csv"))  
  
table.jahr.eingangISO <- fread(paste0(prefix,  
                                       "eingangsjahr_iso.csv"))  
  
table.jahr.entscheid <- fread(paste0(prefix,  
                                       "entscheidungsjahr.csv"))  
  
table.output.praesi <- fread(paste0(prefix,  
                                      "praesi.csv"))  
  
table.output.vpraesi <- fread(paste0(prefix,  
                                      "v_praesi.csv"))
```

9.3 Diagramm: Typ der Entscheidung

```
freqtable <- table.entsch.typ[-.N]
```

```
ggplot(data = freqtable) +  
  geom_bar(aes(x = reorder(entscheidung_typ,  
                           -N),  
               y = N),  
           stat = "identity",  
           fill = "#7e0731",  
           color = "black",  
           width = 0.5) +  
  theme_bw() +  
  labs(  
    title = paste(datasetname,  
                  "| Version",  
                  datestamp,  
                  "| Entscheidungen je Typ"),  
    caption = figure.caption,  
    x = "Typ der Entscheidung",  
    y = "Entscheidungen"  
  ) +  
  theme(  
    text = element_text(size = 14),  
    plot.title = element_text(size = 14,  
                               face = "bold"),  
    legend.position = "none",  
    plot.margin = margin(10, 20, 10, 10)  
  )
```

CE-BVerwG | Version 2021-10-19 | Entscheidungen je Typ



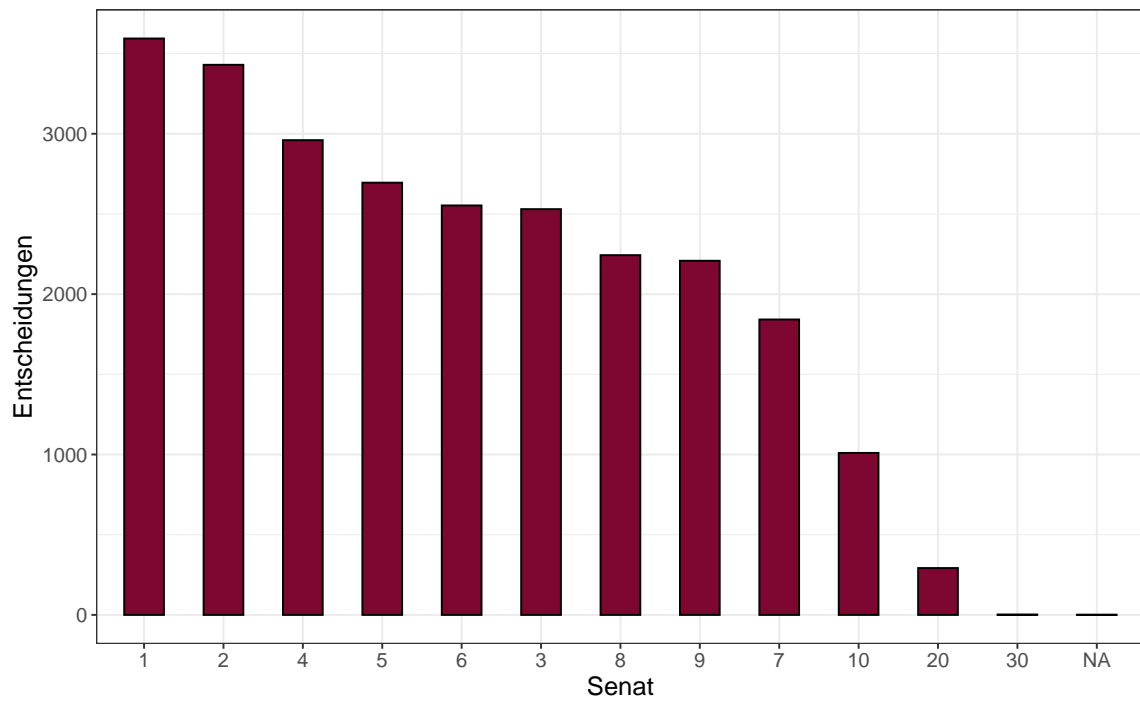
DOI: 10.5281/zenodo.5576822 | Fobbe

9.4 Diagramm: Spruchkörper (Aktenzeichen)

```
freqtable <- table.spruch.az[-.N][,lapply(.SD, as.numeric)]
```

```
ggplot(data = freqtable) +  
  geom_bar(aes(x = reorder(spruchkoerper_az,  
                           -N),  
               y = N),  
           stat = "identity",  
           fill = "#7e0731",  
           color = "black",  
           width = 0.5) +  
  theme_bw() +  
  labs(  
    title = paste(datasetname,  
                  "| Version",  
                  datestamp,  
                  "| Entscheidungen je Senat (Aktenzeichen)" ),  
    caption = figure.caption,  
    x = "Senat",  
    y = "Entscheidungen"  
  ) +  
  theme(  
    text = element_text(size = 14),  
    plot.title = element_text(size = 14,  
                               face = "bold"),  
    legend.position = "none",  
    plot.margin = margin(10, 20, 10, 10)  
  )
```


CE-BVerwG | Version 2021-10-19 | Entscheidungen je Senat (Aktenzeichen)



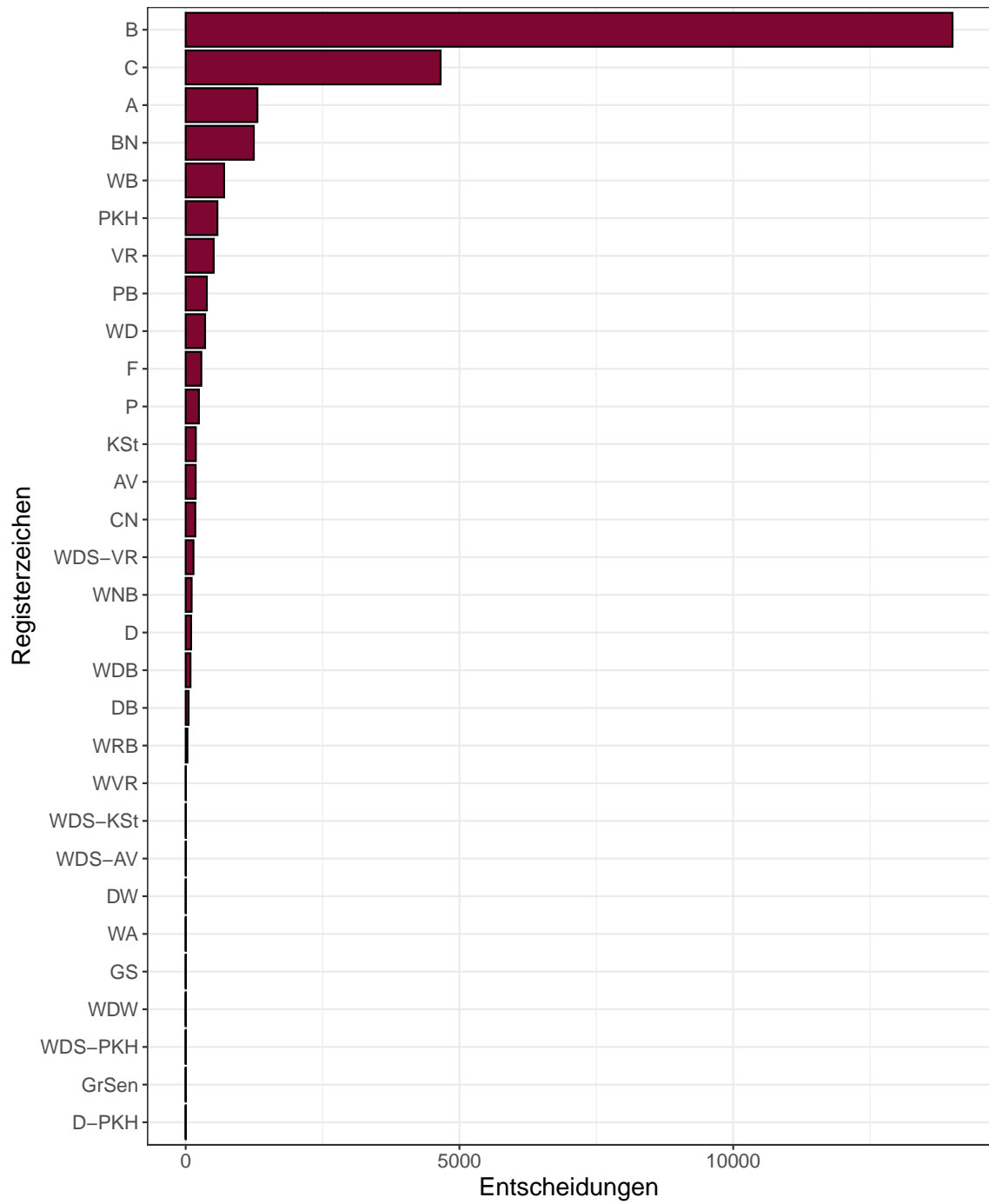
DOI: 10.5281/zenodo.5576822 | Fobbe

9.5 Diagramm: Registerzeichen

```
freqtable <- table.regz[-.N]
```

```
ggplot(data = freqtable) +  
  geom_bar(aes(x = reorder(registerzeichen,  
                           N),  
              y = N),  
          stat = "identity",  
          fill = "#7e0731",  
          color = "black") +  
  coord_flip() +  
  theme_bw() +  
  labs(  
    title = paste(datasetname,  
                  "| Version",  
                  datestamp,  
                  "| Entscheidungen je Registerzeichen"),  
    caption = figure.caption,  
    x = "Registerzeichen",  
    y = "Entscheidungen"  
  ) +  
  theme(  
    text = element_text(size = 14),  
    plot.title = element_text(size = 14,  
                              face = "bold"),  
    legend.position = "none",  
    plot.margin = margin(10, 20, 10, 10)  
  )
```

CE-BVerwG | Version 2021-10-19 | Entscheidungen je Registerzeichen



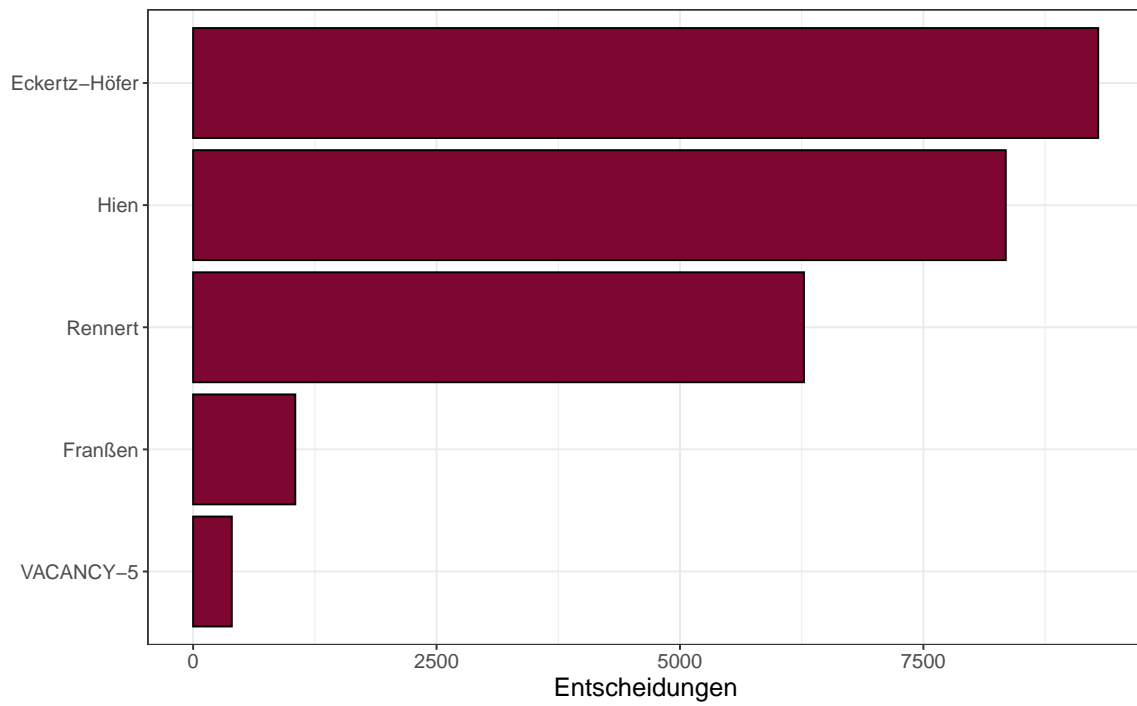
DOI: 10.5281/zenodo.5576822 | Fobbe

9.6 Diagramm: Präsident:in

```
freqtable <- table.output.praesi[-.N]
```

```
ggplot(data = freqtable) +  
  geom_bar(aes(x = reorder(praesi,  
                           N),  
               y = N),  
           stat = "identity",  
           fill = "#7e0731",  
           color = "black") +  
  coord_flip() +  
  theme_bw() +  
  labs(  
    title = paste(datasetname,  
                  "| Version",  
                  datestamp,  
                  "| Entscheidungen je Präsident:in"),  
    caption = figure.caption,  
    x = "Präsident:in",  
    y = "Entscheidungen"  
  ) +  
  theme(  
    axis.title.y = element_blank(),  
    text = element_text(size = 14),  
    plot.title = element_text(size = 14,  
                              face = "bold"),  
    legend.position = "none",  
    plot.margin = margin(10, 20, 10, 10)  
  )
```

CE-BVerwG | Version 2021-10-19 | Entscheidungen je Präsident:in



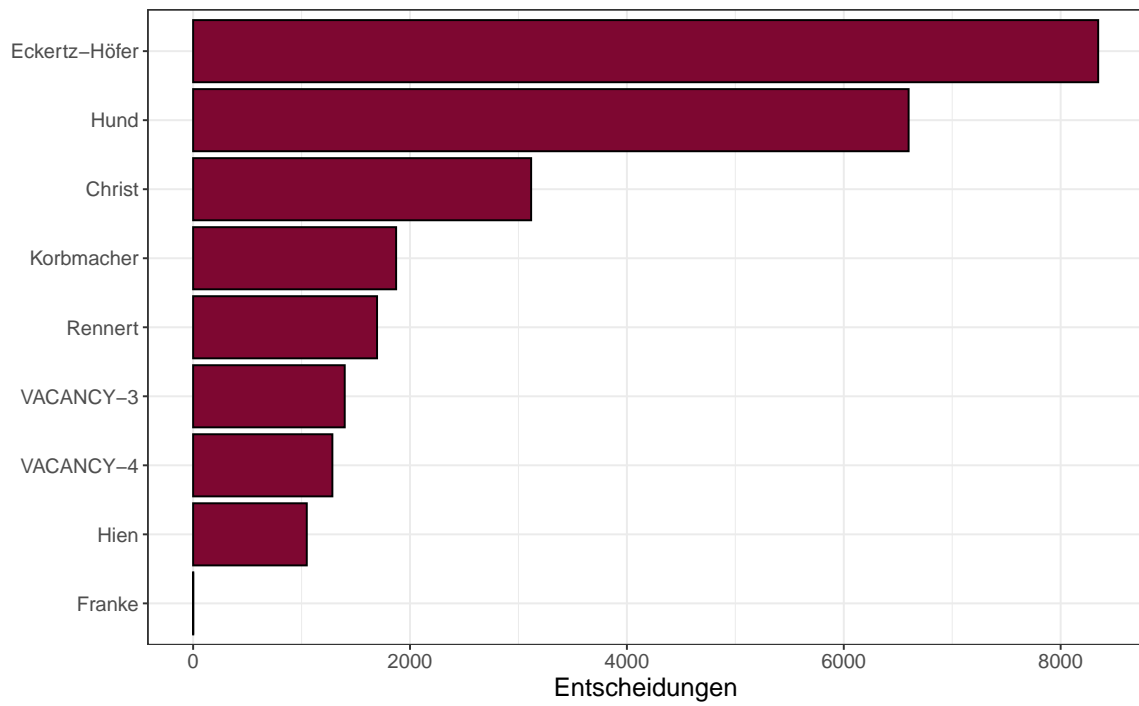
DOI: 10.5281/zenodo.5576822 | Fobbe

9.7 Diagramm: Vize-Präsident:in

```
freqtable <- table.output.vpraesi[-.N]
```

```
ggplot(data = freqtable) +  
  geom_bar(aes(x = reorder(v_praesi,  
                           N),  
               y = N),  
           stat = "identity",  
           fill = "#7e0731",  
           color = "black") +  
  coord_flip() +  
  theme_bw() +  
  labs(  
    title = paste(datasetname,  
                  "| Version",  
                  datestamp,  
                  "| Entscheidungen je Vize-Präsident:in"),  
    caption = figure.caption,  
    x = "Vize-Präsident:in",  
    y = "Entscheidungen"  
  ) +  
  theme(  
    axis.title.y = element_blank(),  
    text = element_text(size = 14),  
    plot.title = element_text(size = 14,  
                              face = "bold"),  
    legend.position = "none",  
    plot.margin = margin(10, 20, 10, 10)  
  )
```

CE-BVerwG | Version 2021-10-19 | Entscheidungen je Vize-Präsident:in

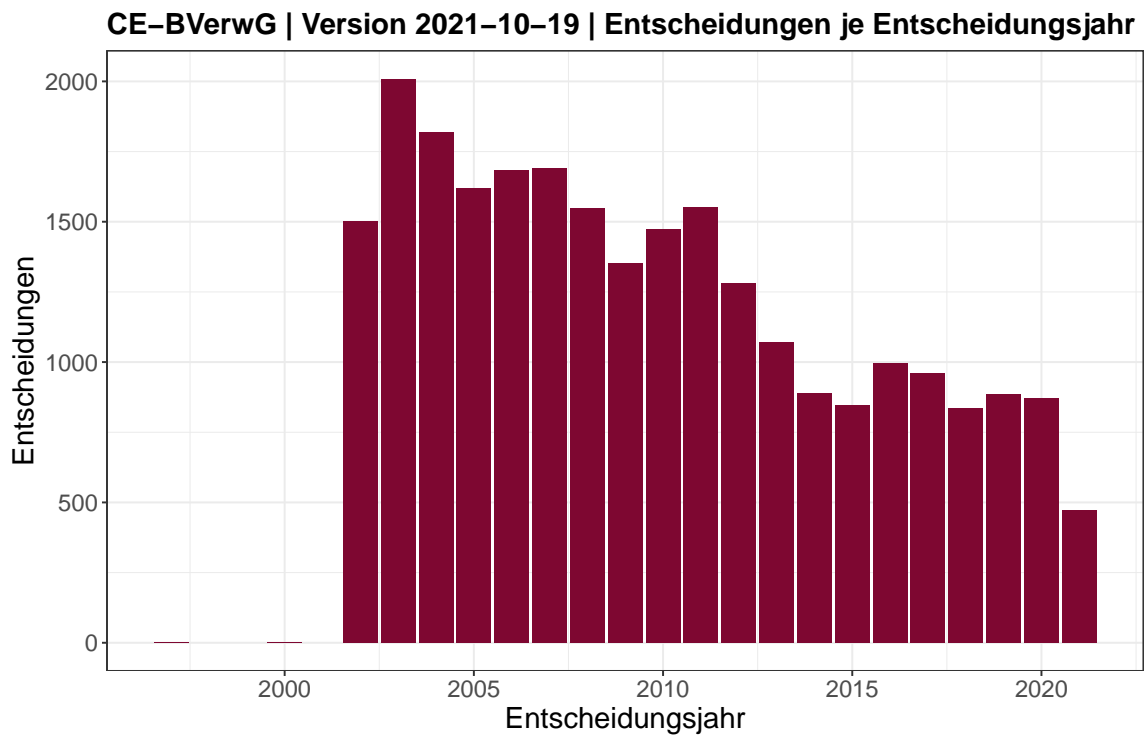


DOI: 10.5281/zenodo.5576822 | Fobbe

9.8 Diagramm: Entscheidungsjahr

```
freqtable <- table.jahr.entscheid[-.N][,lapply(.SD, as.numeric)]
```

```
ggplot(data = freqtable) +  
  geom_bar(aes(x = entscheidungsjahr,  
               y = N),  
           stat = "identity",  
           fill = "#7e0731") +  
  theme_bw() +  
  labs(  
    title = paste(datasetname,  
                  "| Version",  
                  datestamp,  
                  "| Entscheidungen je Entscheidungsjahr"),  
    caption = figure.caption,  
    x = "Entscheidungsjahr",  
    y = "Entscheidungen"  
  ) +  
  theme(  
    text = element_text(size = 16),  
    plot.title = element_text(size = 16,  
                              face = "bold"),  
    legend.position = "none",  
    plot.margin = margin(10, 20, 10, 10)  
  )
```

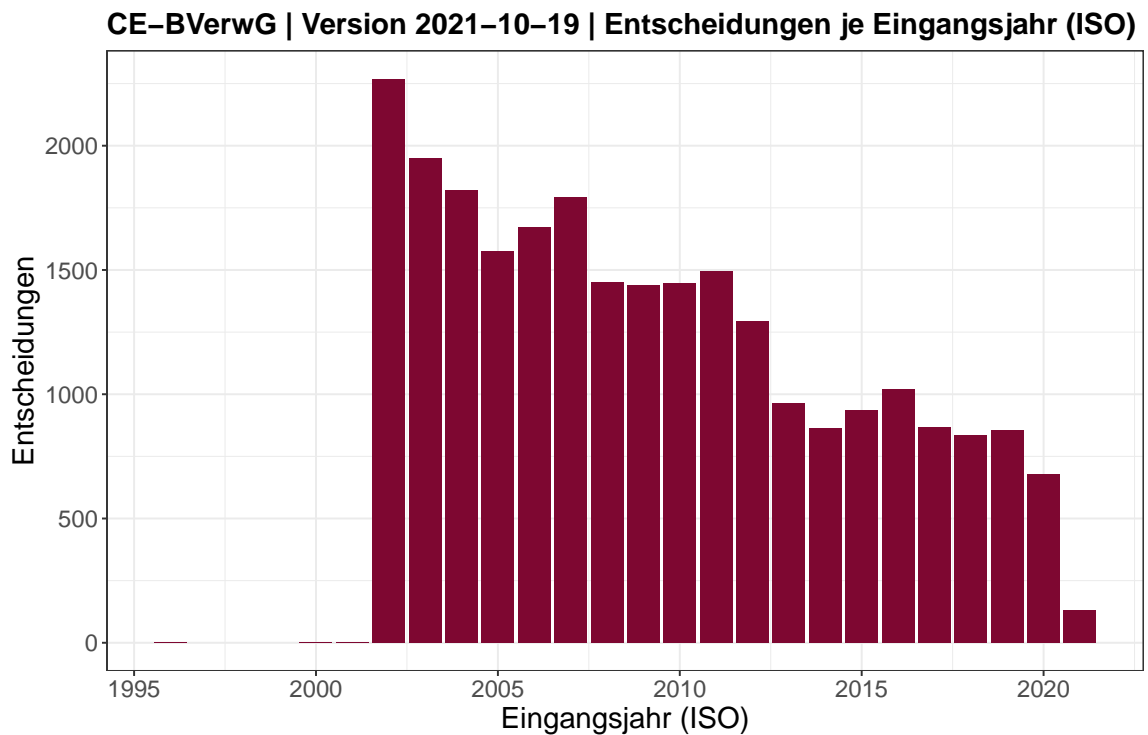


DOI: 10.5281/zenodo.5576822 | Fobbe

9.9 Diagramm: Eingangsjahr (ISO)

```
freqtable <- table.jahr.eingangISO[-.N][,lapply(.SD, as.numeric)]
```

```
ggplot(data = freqtable) +  
  geom_bar(aes(x = eingangsjahr_iso,  
               y = N),  
           stat = "identity",  
           fill = "#7e0731") +  
  theme_bw() +  
  labs(  
    title = paste(datasetname,  
                  "| Version",  
                  datestamp,  
                  "| Entscheidungen je Eingangsjahr (ISO)" ),  
    caption = figure.caption,  
    x = "Eingangsjahr (ISO)",  
    y = "Entscheidungen"  
  ) +  
  theme(  
    text = element_text(size = 16),  
    plot.title = element_text(size = 16,  
                              face = "bold"),  
    legend.position = "none",  
    plot.margin = margin(10, 20, 10, 10)  
  )
```



DOI: 10.5281/zenodo.5576822 | Fobbe

10 Korpus-Analytik

10.1 Berechnung linguistischer Kennwerte

An dieser Stelle werden für jedes Dokument die Anzahl Tokens, Typen und Sätze berechnet und mit den jeweiligen Metadaten verknüpft. Das Ergebnis ist grundsätzlich identisch mit dem eigentlichen Datensatz, nur ohne den Text der Entscheidungen.

10.1.1 Funktion anzeigen

```
print(f.lingsummarize.iterator)
```

```
## function(dt,
##                                     threads = detectCores(),
##                                     chunksize = 1){
##
##     begin.dopar <- Sys.time()
##     dt <- dt[,.(doc_id, text)]
##     nchars <- dt[, lapply(.(text), nchar)]
##     print(paste0("Parallel processing using ",
##                  threads,
##                  " threads. Begin at ",
##                  begin.dopar,
##                  ". Processing ",
##                  dt[,.N],
##                  " documents with a total length of ",
##                  sum(nchars),
##                  " characters."))
##
##     ord <- order(-nchars)
##     dt <- dt[ord]
##
##     cl <- makeForkCluster(threads)
##     registerDoParallel(cl)
##
##     itx <- iter(dt["nchars" > 0],
##                 by = "row",
##                 chunksize = chunksize)
##
##     result.list <- foreach(i = itx,
##                             .errorhandling = 'pass') %dopar% {
##
##         corpus <- corpus(i)
##
##         tokens <- tokens(corpus,
##                           what = "word",
```

```

##                                     remove_punct = FALSE,
##                                     remove_symbols = FALSE,
##                                     remove_numbers = FALSE,
##                                     remove_url = FALSE,
##                                     remove_separators = TRUE,
##                                     split_hyphens = FALSE,
##                                     include_docvars = FALSE,
##                                     padding = FALSE
##                                     )
##
##                                     ntokens <- unname(ntoken(tokens))
##                                     ntypes <- unname(ntype(tokens))
##                                     nsentences <- unname(nsentence(corpus))
##
##                                     temp <- data.table(ntokens,
##                                                         ntypes,
##                                                         nsentences)
##
##                                     return(temp)
##                                     }
##
## stopCluster(cl)
##
##
## end.dopar <- Sys.time()
## duration.dopar <- end.dopar - begin.dopar
##
## result.dt <- rbindlist(result.list)
##
## summary.corpus <- cbind(nchars[ord],
##                         result.dt)
##
## setnames(summary.corpus,
##          "V1",
##          "nchars")
##
##
## if(dt["nchars" == 0, .N] > 0){
##
##     dt.charnull <- dt["nchars" == 0]
##     dt.charnull$text <- NULL
##     dt.charnull$ntokens <- rep(0, dt.charnull[,.N])
##     dt.charnull$ntypes <- rep(0, dt.charnull[,.N])
##     dt.charnull$nsentences <- rep(0, dt.charnull[,.N])
##
##     summary.corpus <- rbind(summary.corpus,
##                             dt.charnull)
## }
##
##
## summary.corpus <- summary.corpus[order(ord)]
##
##
## print(paste0("Runtime was ",
##              round(duration.dopar,
##                    digits = 2),

```

```
##           " ",
##           attributes(duration.dopar)$units,
##           ". Ended at ",
##           end.dopar, ".")
##
##   return(summary.corpus)
##
## }
```

10.1.2 Berechnung durchführen

```
summary.corpus <- f.lingsummarize.iterator(txt.bverwg,
                                           threads = fullCores,
                                           chunksize = 1)
```

```
## [1] "Parallel processing using 16 threads. Begin at 2021-10-19 09:00:21.
      Processing 25360 documents with a total length of 355628762 characters."
## [1] "Runtime was 1.61 mins. Ended at 2021-10-19 09:01:58."
```

10.1.3 Variablen-Namen anpassen

```
setnames(summary.corpus,
         old = c("nchars",
                 "ntokens",
                 "ntypes",
                 "nsentences"),
         new = c("zeichen",
                 "tokens",
                 "typen",
                 "saetze"))
```

10.2 Kennwerte dem Korpus hinzufügen

```
txt.bverwg <- cbind(txt.bverwg,
                    summary.corpus)
```

10.3 Variante mit Metadaten erstellen

```
meta.bverwg <- txt.bverwg[, !"text"]
```

10.4 Linguistische Kennwerte

10.4.1 Zusammenfassungen berechnen

```
dt.summary.ling <- meta.bverwg[, lapply(.SD,
                                     function(x) unclass(summary(x))),
                              .SDcols = c("zeichen",
                                           "tokens",
                                           "typen",
                                           "saetze")]

dt.sums.ling <- meta.bverwg[,
                             lapply(.SD, sum),
                             .SDcols = c("zeichen",
                                           "tokens",
                                           "typen",
                                           "saetze")]

tokens.temp <- tokens(corpus(txt.bverwg),
                     what = "word",
                     remove_punct = FALSE,
                     remove_symbols = FALSE,
                     remove_numbers = FALSE,
                     remove_url = FALSE,
                     remove_separators = TRUE,
                     split_hyphens = FALSE,
                     include_docvars = FALSE,
                     padding = FALSE
                     )

dt.sums.ling$typen <- nfeat(dfm(tokens.temp))

dt.stats.ling <- rbind(dt.sums.ling,
                      dt.summary.ling)

dt.stats.ling <- transpose(dt.stats.ling,
                           keep.names = "names")

setnames(dt.stats.ling, c("Variable",
                          "Sum",
                          "Min",
                          "Quart1",
                          "Median",
                          "Mean",
                          "Quart3",
                          "Max"))
```

10.4.2 Zusammenfassungen anzeigen

```
kable(dt.stats.ling,  
      format.args = list(big.mark = ","),  
      format = "latex",  
      booktabs = TRUE,  
      longtable = TRUE)
```

Variable	Sum	Min	Quart1	Median	Mean	Quart3	Max
zeichen	355,628,762	497	2,928	7,864.0	14,023.2162	18,436.5	782,167
tokens	53,680,746	62	416	1,179.5	2,116.7487	2,788.0	119,583
typen	360,820	47	219	470.0	616.5420	847.0	12,955
saetze	3,086,167	4	30	71.0	121.6943	158.0	5,127

10.4.3 Zusammenfassungen speichern

```
fwrite(dt.stats.ling,  
       paste0(outputdir,  
              datasetname,  
              "_00_KorpusStatistik_ZusammenfassungLinguistisch.csv"),  
       na = "NA")
```

10.5 Quantitative Variablen

10.5.1 Entscheidungsdatum

```
summary(as.IDate(meta.bverwg$datum))
```

##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	"1997-02-26"	"2005-08-11"	"2009-07-23"	"2010-04-19"	"2014-06-12"	"2021-09-27"

10.5.2 Zusammenfassungen berechnen

```
dt.summary.docvars <- meta.bverwg[,  
                                lapply(.SD, function(x)unclass(summary(na.  
                                omit(x))))),  
                                .SDcols = c("entscheidungsjahr",  
                                "eingangsjahr_iso",  
                                "eingangsnummer")]  
  
dt.unique.docvars <- meta.bverwg[,  
                                lapply(.SD, function(x)length(unique(na.omit(  
                                x))))),  
                                .SDcols = c("entscheidungsjahr",  
                                "eingangsjahr_iso",  
                                "eingangsnummer")]  
  
dt.stats.docvars <- rbind(dt.unique.docvars,  
                           dt.summary.docvars)  
  
dt.stats.docvars <- transpose(dt.stats.docvars,  
                              keep.names = "names")  
  
setnames(dt.stats.docvars, c("Variable",  
                              "Unique",  
                              "Min",  
                              "Quart1",  
                              "Median",  
                              "Mean",  
                              "Quart3",  
                              "Max"))
```

10.5.3 Zusammenfassungen anzeigen

```
kable(dt.stats.docvars,  
      format = "latex",  
      booktabs = TRUE,  
      longtable = TRUE)
```

Variable	Unique	Min	Quart1	Median	Mean	Quart3	Max
entscheidungsjahr	22	1997	2005	2009	2009.80406	2014	2021
eingangsjahr_iso	23	1996	2005	2009	2009.32630	2013	2021
eingangsnummer	573	1	10	27	61.76226	59	7005

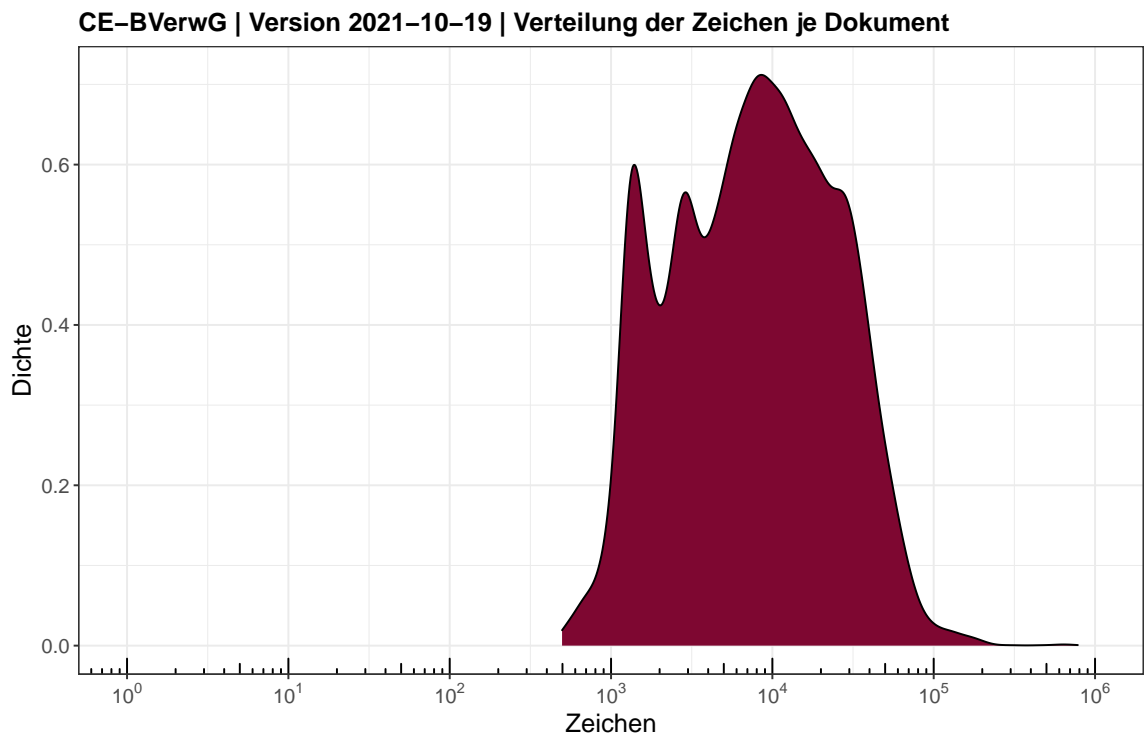
10.5.4 Zusammenfassungen speichern

```
fwrite(dt.stats.docvars,  
       paste0(outputdir,  
               datasetname,  
               "_00_KorpusStatistik_ZusammenfassungDocvarsQuantitativ.csv"),  
       na = "NA")
```


10.6 Verteilungen linguistischer Kennwerte

10.6.1 Diagramm: Verteilung Zeichen

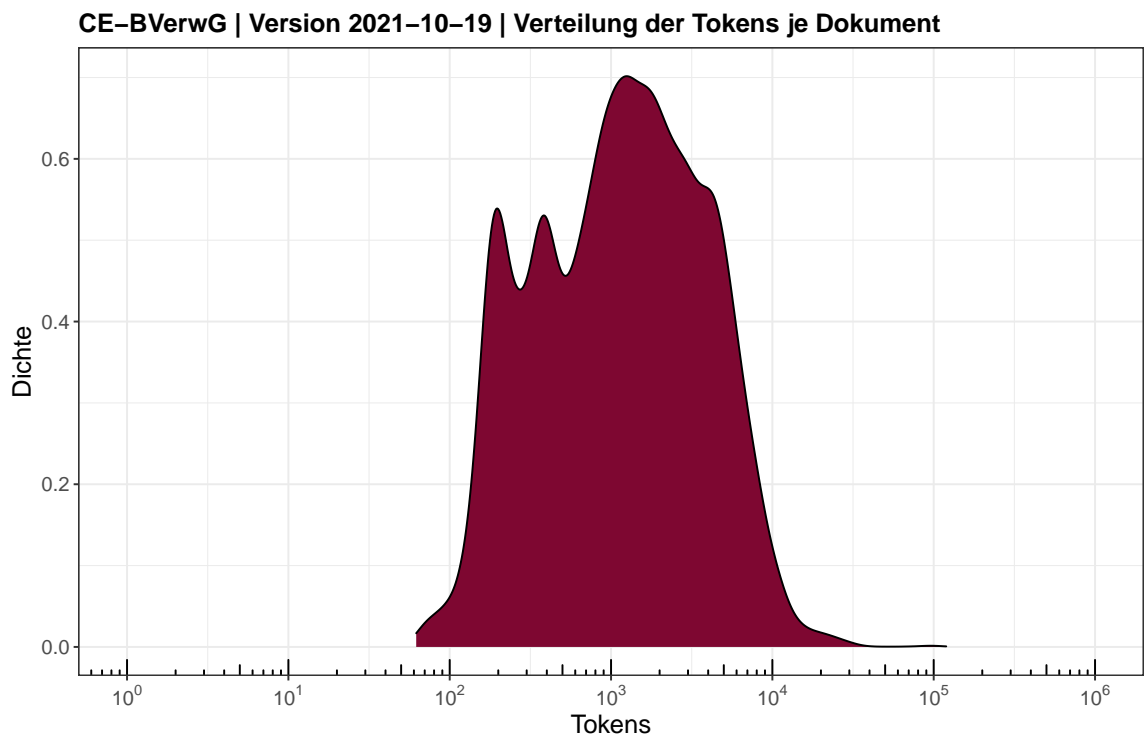
```
ggplot(data = meta.bverwg)+  
  geom_density(aes(x = zeichen),  
               fill = "#7e0731")+  
  scale_x_log10(breaks = trans_breaks("log10", function(x) 10^x),  
               labels = trans_format("log10", math_format(10^.x)))+  
  annotation_logticks(sides = "b")+  
  coord_cartesian(xlim = c(1, 10^6))+  
  theme_bw()+  
  labs(  
    title = paste(datasetname,  
                  "| Version",  
                  datestamp,  
                  "| Verteilung der Zeichen je Dokument"),  
    caption = figure.caption,  
    x = "Zeichen",  
    y = "Dichte"  
  )+  
  theme(  
    text = element_text(size = 14),  
    plot.title = element_text(size = 14,  
                              face = "bold"),  
    legend.position = "none",  
    plot.margin = margin(10, 20, 10, 10)  
  )
```



DOI: 10.5281/zenodo.5576822 | Fobbe

10.6.2 Diagramm: Verteilung Tokens

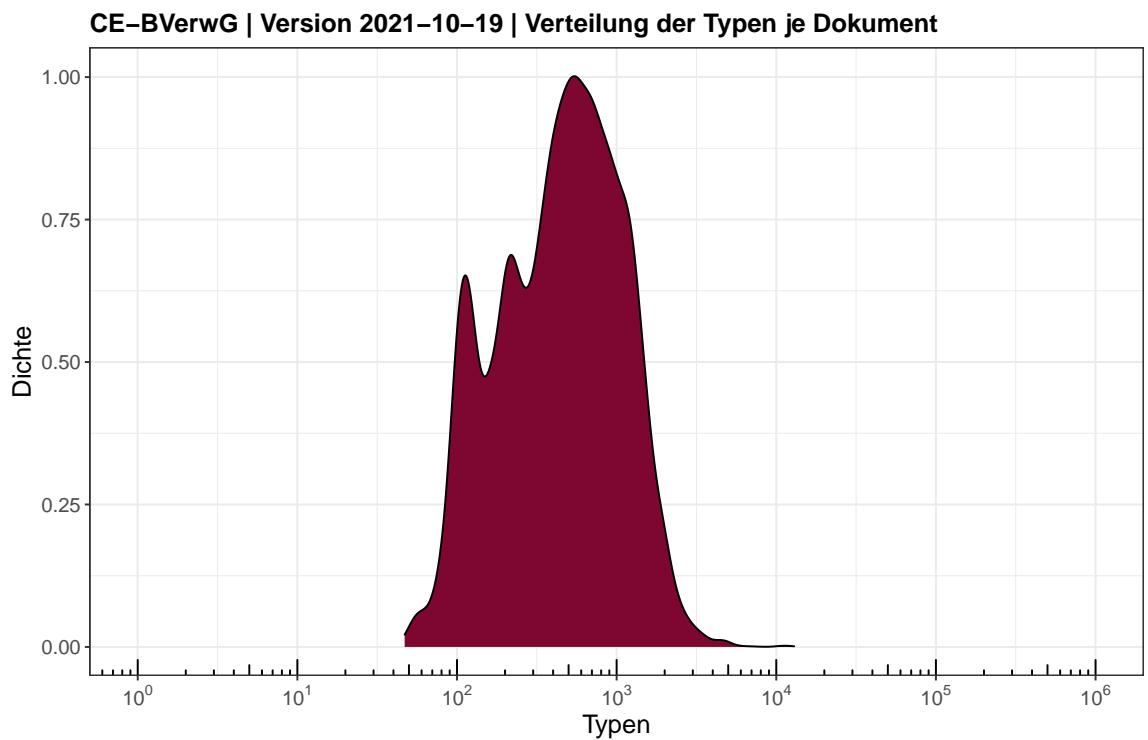
```
ggplot(data = meta.bverwg)+
  geom_density(aes(x = tokens),
    fill = "#7e0731")+
  scale_x_log10(breaks = trans_breaks("log10", function(x) 10^x),
    labels = trans_format("log10", math_format(10^.x)))+
  annotation_logticks(sides = "b")+
  coord_cartesian(xlim = c(1, 10^6))+
  theme_bw()+
  labs(
    title = paste(datasetname,
      "| Version",
      datestamp,
      "| Verteilung der Tokens je Dokument"),
    caption = figure.caption,
    x = "Tokens",
    y = "Dichte"
  )+
  theme(
    text = element_text(size = 14),
    plot.title = element_text(size = 14,
      face = "bold"),
    legend.position = "none",
    plot.margin = margin(10, 20, 10, 10)
  )
```



DOI: 10.5281/zenodo.5576822 | Fobbe

10.6.3 Diagramm: Verteilung Typen

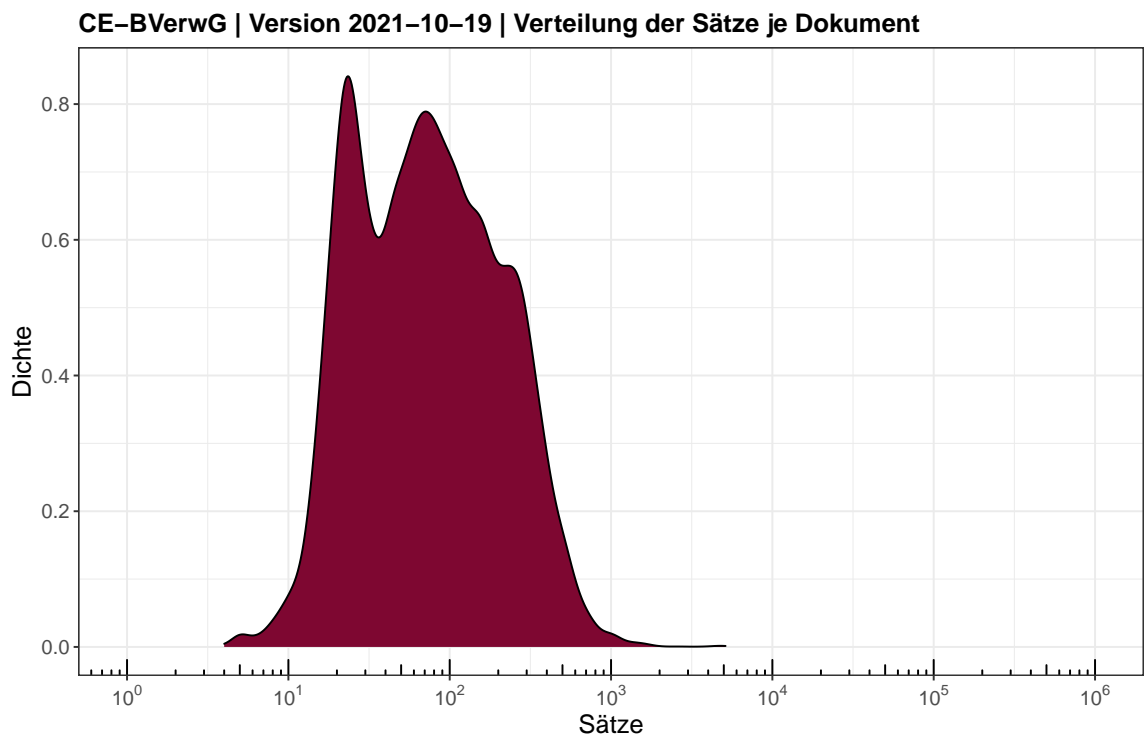
```
ggplot(data = meta.bverwg)+
  geom_density(aes(x = typen),
    fill = "#7e0731")+
  scale_x_log10(breaks = trans_breaks("log10", function(x) 10^x),
    labels = trans_format("log10", math_format(10^.x)))+
  annotation_logticks(sides = "b")+
  coord_cartesian(xlim = c(1, 10^6))+
  theme_bw()+
  labs(
    title = paste(datasetname,
      "| Version",
      datestamp,
      "| Verteilung der Typen je Dokument"),
    caption = figure.caption,
    x = "Typen",
    y = "Dichte"
  )+
  theme(
    text = element_text(size = 14),
    plot.title = element_text(size = 14,
      face = "bold"),
    legend.position = "none",
    plot.margin = margin(10, 20, 10, 10)
  )
```



DOI: 10.5281/zenodo.5576822 | Fobbe

10.6.4 Diagramm: Verteilung Sätze

```
ggplot(data = meta.bverwg)+
  geom_density(aes(x = saetze),
    fill = "#7e0731")+
  scale_x_log10(breaks = trans_breaks("log10", function(x) 10^x),
    labels = trans_format("log10", math_format(10^.x)))+
  annotation_logticks(sides = "b")+
  coord_cartesian(xlim = c(1, 10^6))+
  theme_bw()+
  labs(
    title = paste(datasetname,
      "| Version",
      datestamp,
      "| Verteilung der Sätze je Dokument"),
    caption = figure.caption,
    x = "Sätze",
    y = "Dichte"
  )+
  theme(
    text = element_text(size = 14),
    plot.title = element_text(size = 14,
      face = "bold"),
    legend.position = "none",
    plot.margin = margin(10, 20, 10, 10)
  )
```



DOI: 10.5281/zenodo.5576822 | Fobbe

11 Linguistische Annotationen berechnen

```
if (mode.annotate == TRUE){  
  
  txt.annotated <- f.dopar.spacyparse(txt.bverwg,  
                                     threads = detectCores(),  
                                     chunksize = 1,  
                                     model = "de_core_news_sm",  
                                     pos = TRUE,  
                                     tag = TRUE,  
                                     lemma = TRUE,  
                                     entity = TRUE,  
                                     dependency = TRUE,  
                                     nounphrase = TRUE)  
  
}
```

```
## Found 'spacy_condaenv'. spacyr will use this environment
```

```
## successfully initialized (spaCy Version: 3.1.1, language model: de_core_news_  
sm)
```

```
## (python options: type = "condaenv", value = "spacy_condaenv")
```

```
## [1] "Parallel processing using 16 threads. Begin at 2021-10-19 09:03:55.  
Processing 25360 documents"  
## [1] "Runtime was 16.15 mins. Ended at 2021-10-19 09:20:04."
```

12 Kontrolle der Variablen

12.1 Semantische Sortierung der Variablen

12.1.1 Variablen sortieren: Hauptdatensatz

```
setcolorder(txt.bverwg,  
  c("doc_id",  
    "text",  
    "gericht",  
    "datum",  
    "entscheidung_typ",  
    "spruchkoerper_az",  
    "registerzeichen",  
    "verfahrensart",  
    "eingangsnummer",  
    "eingangsjahr_az",  
    "eingangsjahr_iso",  
    "entscheidungsjahr",  
    "verzoegerung",  
    "kollision",  
    "aktenzeichen",  
    "ecli",  
    "praesi",  
    "v_praesi",  
    "zeichen",  
    "tokens",  
    "typen",  
    "saetze",  
    "version",  
    "doi_concept",  
    "doi_version",  
    "lizenz"))
```

12.1.2 Variablen sortieren: Metadaten

```
setcolorder(meta.bverwg,  
  c("doc_id",  
    "gericht",  
    "datum",  
    "entscheidung_typ",  
    "spruchkoerper_az",  
    "registerzeichen",  
    "verfahrensart",  
    "eingangsnummer",  
    "eingangsjahr_az",  
    "eingangsjahr_iso",  
    "entscheidungsjahr",  
    "verzoegerung",  
    "kollision",  
    "aktenzeichen",  
    "ecli",  
    "praesi",  
    "v_praesi",  
    "zeichen",  
    "tokens",  
    "typen",  
    "saetze",  
    "version",  
    "doi_concept",  
    "doi_version",  
    "lizenz"))
```

12.2 Anzahl Variablen der Datensätze

```
length(txt.bverwg)
```

```
## [1] 26
```

```
length(meta.bverwg)
```

```
## [1] 25
```

```
length(txt.annotated)
```

```
## [1] 12
```

12.3 Alle Variablen-Namen der Datensätze

```
names(txt.bverwg)
```

```
## [1] "doc_id"      "text"        "gericht"
## [4] "datum"      "entscheidung_typ" "spruchkoerper_az"
## [7] "registerzeichen" "verfahrensart" "eingangsnummer"
## [10] "eingangsjahr_az" "eingangsjahr_iso" "entscheidungsjahr"
## [13] "verzoegerung" "kollision" "aktenzeichen"
## [16] "ecli"       "praesi"     "v_praesi"
## [19] "zeichen"    "tokens"     "typen"
## [22] "saetze"     "version"    "doi_concept"
## [25] "doi_version" "lizenz"
```

```
names(meta.bverwg)
```

```
## [1] "doc_id"      "gericht"     "datum"
## [4] "entscheidung_typ" "spruchkoerper_az" "registerzeichen"
## [7] "verfahrensart" "eingangsnummer" "eingangsjahr_az"
## [10] "eingangsjahr_iso" "entscheidungsjahr" "verzoegerung"
## [13] "kollision"     "aktenzeichen" "ecli"
## [16] "praesi"       "v_praesi"    "zeichen"
## [19] "tokens"      "typen"       "saetze"
## [22] "version"     "doi_concept" "doi_version"
## [25] "lizenz"
```



```
names(txt.annotated)
```

```
## [1] "doc_id"      "sentence_id" "token_id"     "token"  
## [5] "lemma"       "pos"          "tag"          "head_token_id"  
## [9] "dep_rel"     "entity"       "nounphrase"   "whitespace"
```

13 Beispiel-Werte für alle Metadaten anzeigen

```
print(meta.bverwg)
```

```
##                                doc_id gericht      datum
##      1:   BVerwG_1997-02-26_U_6_C_3_96_NA_0.txt BVerwG 1997-02-26
##      2: BVerwG_2000-12-14_B_1_WB_107_00_NA_0.txt BVerwG 2000-12-14
##      3:   BVerwG_2002-01-10_B_9_A_9_02_NA_0.txt BVerwG 2002-01-10
##      4:   BVerwG_2002-01-14_B_4_BN_1_02_NA_0.txt BVerwG 2002-01-14
##      5:   BVerwG_2002-01-17_B_1_B_12_02_NA_0.txt BVerwG 2002-01-17
##      ---
## 25356:   BVerwG_2021-09-17_B_8_B_14_21_NA_0.txt BVerwG 2021-09-17
## 25357:   BVerwG_2021-09-21_B_1_AV_5_21_NA_0.txt BVerwG 2021-09-21
## 25358:   BVerwG_2021-09-21_B_4_BN_2_21_NA_0.txt BVerwG 2021-09-21
## 25359:   BVerwG_2021-09-21_B_8_B_4_21_NA_0.txt BVerwG 2021-09-21
## 25360:   BVerwG_2021-09-27_B_6_BN_1_21_NA_0.txt BVerwG 2021-09-27
##      entscheidung_typ spruchkoerper_az registerzeichen
##      1:                U                6                C
##      2:                B                1                WB
##      3:                B                9                A
##      4:                B                4                BN
##      5:                B                1                B
##      ---
## 25356:                B                8                B
## 25357:                B                1                AV
## 25358:                B                4                BN
## 25359:                B                8                B
## 25360:                B                6                BN
##
##      verfahrensart
##      1:                Revision (
##      Verwaltungsstreitsachen)
##      2:                Verfahren nach
##      Wehrbeschwerdeordnung
##      3: Erstinstanzliche Klage, inklusive Wiederaufnahmeverfahren (
##      Verwaltungsstreitsachen)
##      4:                Nichtzulassungsbeschwerde (
##      Normenkontrollsachen)
##      5:                Nichtzulassungsbeschwerde oder Beschwerde (
##      Verwaltungsstreitsachen)
##      ---
## 25356:                Nichtzulassungsbeschwerde oder Beschwerde (
##      Verwaltungsstreitsachen)
## 25357:      Allgemeine Verfahren (sonstige Anträge außerhalb eines schwebenden
##      Verfahrens)
## 25358:                Nichtzulassungsbeschwerde (
##      Normenkontrollsachen)
## 25359:                Nichtzulassungsbeschwerde oder Beschwerde (
##      Verwaltungsstreitsachen)
## 25360:                Nichtzulassungsbeschwerde (
##      Normenkontrollsachen)
##      eingangsnummer eingangsjahr_az eingangsjahr_iso entscheidungsjahr
##      1:                3                96                1996                1997
```

##	2:	107	0	2000	2000
##	3:	9	2	2002	2002
##	4:	1	2	2002	2002
##	5:	12	2	2002	2002
##	---				
##	25356:	14	21	2021	2021
##	25357:	5	21	2021	2021
##	25358:	2	21	2021	2021
##	25359:	4	21	2021	2021
##	25360:	1	21	2021	2021
##	verzoegerung kollision aktenzeichen				
##	1:	<NA>	0	BVerwG 6 C 3.96	
##	2:	<NA>	0	BVerwG 1 WB 107.0	
##	3:	<NA>	0	BVerwG 9 A 9.2	
##	4:	<NA>	0	BVerwG 4 BN 1.2	
##	5:	<NA>	0	BVerwG 1 B 12.2	
##	---				
##	25356:	<NA>	0	BVerwG 8 B 14.21	
##	25357:	<NA>	0	BVerwG 1 AV 5.21	
##	25358:	<NA>	0	BVerwG 4 BN 2.21	
##	25359:	<NA>	0	BVerwG 8 B 4.21	
##	25360:	<NA>	0	BVerwG 6 BN 1.21	
##	tokens				
##	1:	ECLI:DE:BVerwG:1997:260297U6C3.96.0	Franßen	Franke	37523
		5471			
##	2:	ECLI:DE:BVerwG:2000:141200B1WB107.00.0	Franßen	Hien	13855
		2336			
##	3:	ECLI:DE:BVerwG:2002:100102B9A9.02.0	Franßen	Hien	826
		128			
##	4:	ECLI:DE:BVerwG:2002:140102B4BN1.02.0	Franßen	Hien	3646
		557			
##	5:	ECLI:DE:BVerwG:2002:170102B1B12.02.0	Franßen	Hien	1401
		213			
##	---				
##	25356:	ECLI:DE:BVerwG:2021:170921B8B14.21.0	Rennert	Korbmacher	2667
		373			
##	25357:	ECLI:DE:BVerwG:2021:210921B1AV5.21.0	Rennert	Korbmacher	2859
		434			
##	25358:	ECLI:DE:BVerwG:2021:210921B4BN2.21.0	Rennert	Korbmacher	2578
		358			
##	25359:	ECLI:DE:BVerwG:2021:210921B8B4.21.0	Rennert	Korbmacher	10205
		1501			
##	25360:	ECLI:DE:BVerwG:2021:270921B6BN1.21.0	Rennert	Korbmacher	13939
		2224			
##	typen saetze version doi_concept doi_version				
##	1:	1470	267	2021-10-19	10.5281/zenodo.3911067 10.5281/zenodo.5576822
##	2:	721	143	2021-10-19	10.5281/zenodo.3911067 10.5281/zenodo.5576822
##	3:	84	15	2021-10-19	10.5281/zenodo.3911067 10.5281/zenodo.5576822
##	4:	277	48	2021-10-19	10.5281/zenodo.3911067 10.5281/zenodo.5576822
##	5:	128	18	2021-10-19	10.5281/zenodo.3911067 10.5281/zenodo.5576822
##	---				
##	25356:	195	31	2021-10-19	10.5281/zenodo.3911067 10.5281/zenodo.5576822
##	25357:	195	31	2021-10-19	10.5281/zenodo.3911067 10.5281/zenodo.5576822
##	25358:	193	30	2021-10-19	10.5281/zenodo.3911067 10.5281/zenodo.5576822
##	25359:	535	87	2021-10-19	10.5281/zenodo.3911067 10.5281/zenodo.5576822

```
## 25360: 677 146 2021-10-19 10.5281/zenodo.3911067 10.5281/zenodo.5576822
## lizenz
## 1: Creative Commons Zero 1.0 Universal
## 2: Creative Commons Zero 1.0 Universal
## 3: Creative Commons Zero 1.0 Universal
## 4: Creative Commons Zero 1.0 Universal
## 5: Creative Commons Zero 1.0 Universal
## ---
## 25356: Creative Commons Zero 1.0 Universal
## 25357: Creative Commons Zero 1.0 Universal
## 25358: Creative Commons Zero 1.0 Universal
## 25359: Creative Commons Zero 1.0 Universal
## 25360: Creative Commons Zero 1.0 Universal
```

14 CSV-Dateien erstellen

14.1 CSV mit vollem Datensatz speichern

```
csvname.full <- paste(datasetname,
                      datestamp,
                      "DE_CSV_Datensatz.csv",
                      sep = "_")

fwrite(txt.bverwg,
       csvname.full,
       na = "NA")
```

14.2 CSV mit Metadaten speichern

Diese Datei ist grundsätzlich identisch mit dem eigentlichen Datensatz, nur ohne den Text der Entscheidungen.

```
csvname.meta <- paste(datasetname,
                     datestamp,
                     "DE_CSV_Metadaten.csv",
                     sep = "_")

fwrite(meta.bverwg,
       csvname.meta,
       na = "NA")
```

14.3 CSV mit Annotationen speichern

```
if (mode.annotate == TRUE){

  csvname.annotated <- paste(datasetname,
                           datestamp,
                           "DE_CSV_Annotiert.csv",
                           sep = "_")

  fwrite(txt.annotated,
        csvname.annotated,
        na = "NA")

}
```

15 Dateigrößen analysieren

15.1 Gesamtgröße

15.1.1 Korpus-Objekt in RAM (MB)

```
print(object.size(txt.bverwg),  
      standard = "SI",  
      humanReadable = TRUE,  
      units = "MB")
```

```
## 373.3 MB
```

15.1.2 CSV Korpus (MB)

```
file.size(csvname.full) / 10 ^ 6
```

```
## [1] 370.2953
```

15.1.3 CSV Metadaten (MB)

```
file.size(csvname.meta) / 10 ^ 6
```

```
## [1] 8.897624
```

15.1.4 CSV Annotiert (MB)

```
file.size(csvname.annotated) / 10 ^ 6
```

```
## [1] 5583.158
```

15.1.5 PDF-Dateien (MB)

```
files.pdf <- list.files(pattern = "\\..pdf$",  
                        ignore.case = TRUE)  
  
pdf.MB <- file.size(files.pdf) / 10^6  
sum(pdf.MB)
```

```
## [1] 1390.184
```

15.1.6 TXT-Dateien (MB)

```
files.txt <- list.files(pattern = "\\..txt$",  
                        ignore.case = TRUE)  
  
txt.MB <- file.size(files.txt) / 10^6  
sum(txt.MB)
```

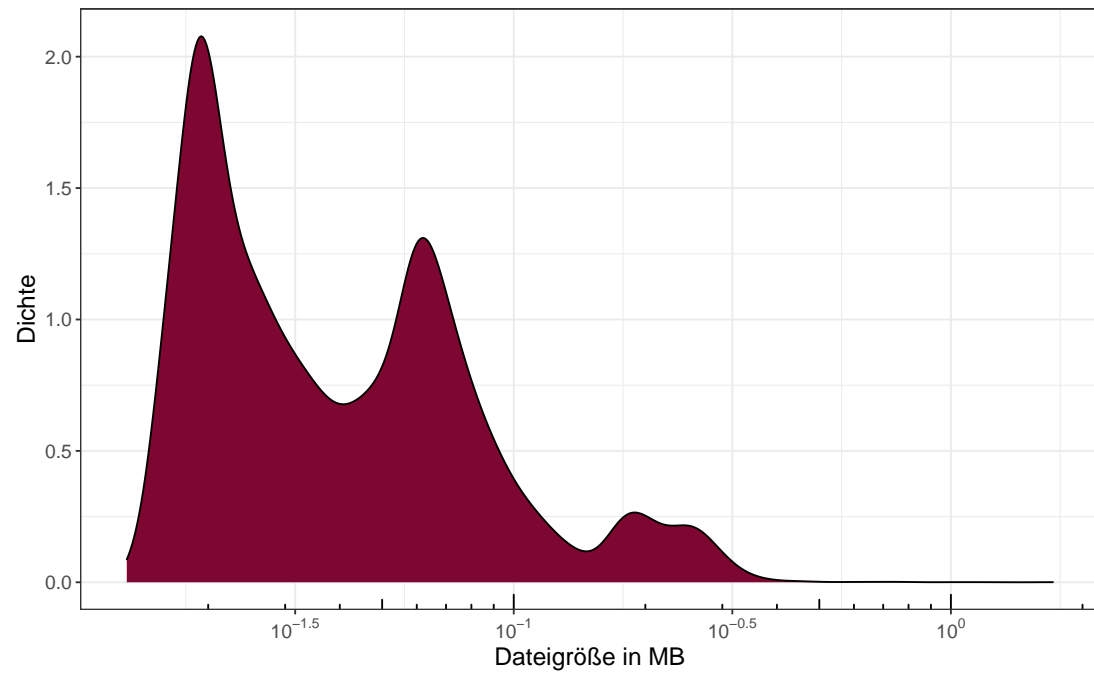
```
## [1] 370.6425
```

15.2 Diagramm: Verteilung der Dateigrößen (PDF)

```
dt.plot <- data.table(pdf.MB)
```

```
ggplot(data = dt.plot,
  aes(x = pdf.MB)) +
  geom_density(fill = "#7e0731") +
  scale_x_log10(breaks = trans_breaks("log10", function(x) 10^x),
    labels = trans_format("log10", math_format(10^.x)))+
  annotation_logticks(sides = "b")+
  theme_bw() +
  labs(
    title = paste(datasetname,
      "| Version",
      datestamp,
      "| Verteilung der Dateigrößen (PDF)" ),
    caption = figure.caption,
    x = "Dateigröße in MB",
    y = "Dichte"
  )+
  theme(
    text = element_text(size = 14),
    plot.title = element_text(size = 14,
      face = "bold"),
    legend.position = "none",
    panel.spacing = unit(0.1, "lines"),
    plot.margin = margin(10, 20, 10, 10)
  )
```


CE-BVerwG | Version 2021-10-19 | Verteilung der Dateigrößen (PDF)

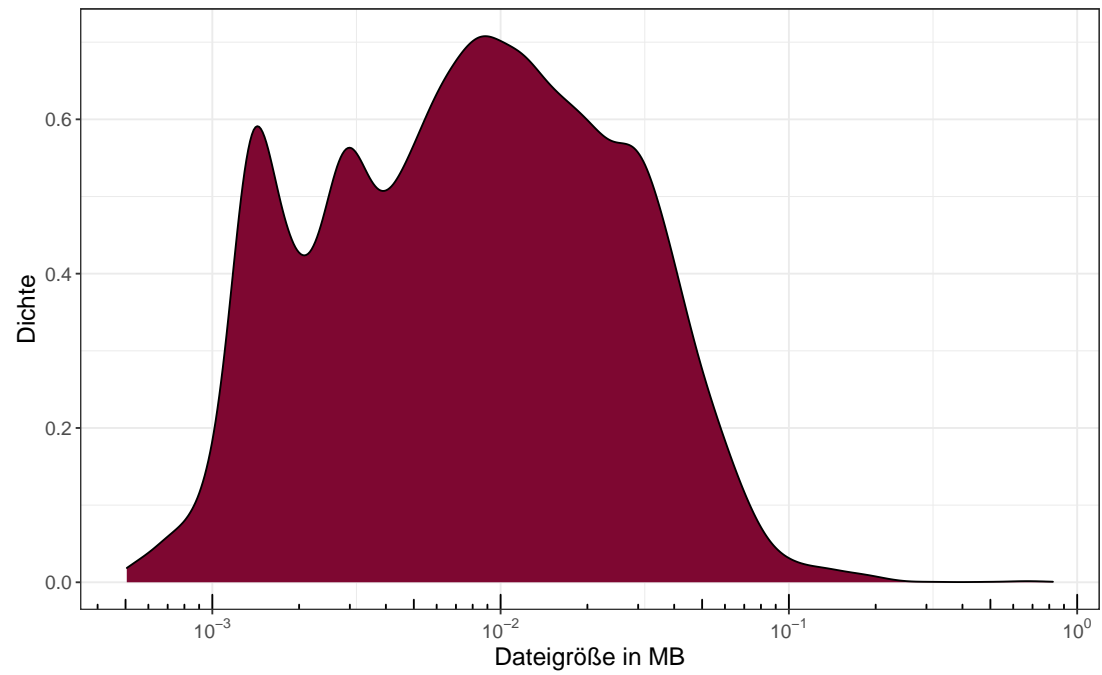


15.3 Diagramm: Verteilung der Dateigrößen (TXT)

```
dt.plot <- data.table(txt.MB)
```

```
ggplot(data = dt.plot,
       aes(x = txt.MB)) +
  geom_density(fill = "#7e0731") +
  scale_x_log10(breaks = trans_breaks("log10", function(x) 10^x),
               labels = trans_format("log10", math_format(10^.x)))+
  annotation_logticks(sides = "b")+
  theme_bw() +
  labs(
    title = paste(datasetname,
                  "| Version",
                  datestamp,
                  "| Verteilung der Dateigrößen (TXT)" ),
    caption = figure.caption,
    x = "Dateigröße in MB",
    y = "Dichte"
  )+
  theme(
    text = element_text(size = 14),
    plot.title = element_text(size = 14,
                              face = "bold"),
    legend.position = "none",
    panel.spacing = unit(0.1, "lines"),
    plot.margin = margin(10, 20, 10, 10)
  )
```

CE-BVerwG | Version 2021-10-19 | Verteilung der Dateigrößen (TXT)



16 Erstellen der ZIP-Archive

16.1 Verpacken der CSV-Dateien

16.1.1 Vollständiger Datensatz

```
csvname.full.zip <- gsub(".csv",  
                        ".zip",  
                        csvname.full)  
  
zip(csvname.full.zip,  
    csvname.full)  
  
unlink(csvname.full)
```

16.1.2 Metadaten

```
csvname.meta.zip <- gsub(".csv",  
                        ".zip",  
                        csvname.meta)  
  
zip(csvname.meta.zip,  
    csvname.meta)  
  
unlink(csvname.meta)
```

16.1.3 Annotiert

```
if (mode.annotate == TRUE){  
  
    csvname.annotated.zip <- gsub(".csv",  
                                ".zip",  
                                csvname.annotated)  
  
    zip(csvname.annotated.zip,  
        csvname.annotated)  
  
    unlink(csvname.annotated)  
  
}
```

16.2 Verpacken der PDF-Dateien

```
files.pdf <- list.files(pattern = "\\..pdf",  
                        ignore.case = TRUE)  
  
zip(paste(datasetname,
```

```

        datestamp,
        "DE_PDF_Datensatz.zip",
        sep = "_"),
files.pdf)

unlink(files.pdf)

```

16.3 Verpacken der TXT-Dateien

```

files.txt <- list.files(pattern = "\\..txt",
                        ignore.case = TRUE)

zip(paste(datasetname,
           datestamp,
           "DE_TXT_Datensatz.zip",
           sep = "_"),
    files.txt)

unlink(files.txt)

```

16.4 Verpacken der Analyse-Dateien

```

zip(paste0(datasetname,
            "_",
            datestamp,
            "_DE_",
            basename(outputdir),
            ".zip"),
    basename(outputdir))

```

16.5 Verpacken der Source-Dateien

```

files.source <- c(list.files(pattern = "Source"),
                  "buttons",
                  "R-fobbe-proto-package")

files.source <- grep("spin",
                    files.source,
                    value = TRUE,
                    ignore.case = TRUE,
                    invert = TRUE)

zip(paste(datasetname,
           datestamp,
           "Source_Files.zip",
           sep = "_"),
    files.source)

```

17 Kryptographische Hashes

Dieses Modul berechnet für jedes ZIP-Archiv zwei Arten von Hashes: SHA2-256 und SHA3-512. Mit diesen kann die Authentizität der Dateien geprüft werden und es wird dokumentiert, dass sie aus diesem Source Code hervorgegangen sind. Die SHA-2 und SHA-3 Algorithmen sind äußerst resistent gegenüber *collision* und *pre-imaging* Angriffen, sie gelten derzeit als kryptographisch sicher. Ein SHA3-Hash mit 512 bit Länge ist nach Stand von Wissenschaft und Technik auch gegenüber quantenkryptoanalytischen Verfahren unter Einsatz des *Grover-Algorithmus* hinreichend resistent.

17.1 Liste der ZIP-Archive erstellen

```
files.zip <- list.files(pattern = "\\\\.zip$",  
                        ignore.case = TRUE)
```

17.2 Funktion anzeigen

```
print(f.dopar.multihashes)
```

```
function(x, threads = detectCores()){
```

```
  print(paste("Parallel processing using", threads, "threads."))  
  
  begin <- Sys.time()  
  
  cl <- makeForkCluster(threads)  
  registerDoParallel(cl)  
  
  multihashes <- foreach(filename = x,  
                        .errorhandling = 'pass',  
                        .combine = 'rbind') %dopar% {  
  
    sha2.256 <- system2("openssl",  
                      paste("sha256",  
                            filename),  
                      stdout = TRUE)  
  
    sha2.256 <- gsub("^.*\\= ",  
                  "",  
                  sha2.256)  
  
    sha3.512 <- system2("openssl",  
                      paste("sha3-512",  
                            filename),  
                      stdout = TRUE)  
  
    sha3.512 <- gsub("^.*\\= ",  
                  "",
```

```

                                sha3.512)

                                out <- data.frame(filename,
                                                    sha2.256,
                                                    sha3.512)
                                return(out)
                                }
stopCluster(cl)

end <- Sys.time()
duration <- end - begin

print(paste0("Processed ",
             length(x),
             " files. Runtime was ",
             round(duration,
                    digits = 2),
             " ",
             attributes(duration)$units,
             "."))

return(multihashes)

}

```

17.3 Hashes berechnen

```
multihashes <- f.dopar.multihashes(files.zip)
```

```
## [1] "Parallel processing using 16 threads."
## [1] "Processed 7 files. Runtime was 6.07 secs."
```

17.4 In Data Table umwandeln

```
setDT(multihashes)
```

17.5 Index hinzufügen

```
multihashes$index <- seq_len(multihashes[,.N])
```

17.6 In Datei schreiben

```
fwrite(multihashes,
      paste(datasetname,
            datestamp,
            "KryptographischeHashes.csv",
            sep = "_"),
      na = "NA")
```

17.7 Leerzeichen hinzufügen um Zeilenumbruch zu ermöglichen

```
multihashes$sha3.512 <- paste(substr(multihashes$sha3.512, 1, 64),
                              substr(multihashes$sha3.512, 65, 128))
```

17.8 In Bericht anzeigen

```
kable(multihashes[,.(index,filename)],
      format = "latex",
      align = c("p{1cm}",
                "p{13cm}"),
      booktabs = TRUE,
      longtable = TRUE)
```

index	filename
1	CE-BVerwG_2021-10-19_DE_ANALYSE.zip
2	CE-BVerwG_2021-10-19_DE_CSV_Annotiert.zip
3	CE-BVerwG_2021-10-19_DE_CSV_Datensatz.zip
4	CE-BVerwG_2021-10-19_DE_CSV_Metadaten.zip
5	CE-BVerwG_2021-10-19_DE_PDF_Datensatz.zip
6	CE-BVerwG_2021-10-19_DE_TXT_Datensatz.zip
7	CE-BVerwG_2021-10-19_Source_Files.zip


```
kable(multihashes[,.(index,sha2.256)],
      format = "latex",
      align = c("c",
                "p{13cm}"),
      booktabs = TRUE,
      longtable = TRUE)
```

index	sha2.256
1	8730b97889ba94523f0f457b899fc7af36b501e640de5a9f4e96a2e7f0cb9911
2	786fdb96f4611831cf1956cf18a08bc6e830e4e2bc8eb4a9140778be31b49730
3	676a4856bace1349526bca751d2898ef9fca6044e57c08970332c334026dfec7
4	64057416da32652cf056e63171bb97284fff29dd4bfd27c757d52d345c9cfb40
5	edb9f6b5ac44245ce19b6d4e2bb10297bc78062dd517bd4a124ea36b0ea3e3c0
6	d6f26e5dd28ce946c561e304076d68fd1827436b2bf8c0b26aaa6b3c9441f566
7	5e3abec8f020f3e2ba16687127a35edbdab8c87780e2be5bcf844259b5ed5a8b

```
kable(multihashes[,.(index,sha3.512)],
      format = "latex",
      align = c("c",
                "p{13cm}"),
      booktabs = TRUE,
      longtable = TRUE)
```

index	sha3.512
1	9176e447818e735942aaa845a4a063275dcae15b31956b9f5f15c6ef6b7e2527 56d62e96c3fabe54ec1bf0a8f5ff2aa2866714838100cef101601910045010a9
2	67c4706ffcbd94a4b159ede9c2e7b486fff05ed25695a5a7a2d1c25d55874751 79b310eaabf147e4c0cb46d8da5276174664da96a4286aee5b831971899bfb7a
3	d62969dc9cfdfa75f90597a1186f7094fda082c0a4a620604b4b39efe3387722 109b84149c6bad80997940150a2c27dbe2d61a961426a9e9c329e2028cfec69f
4	9e3cb6ef4530a7e6fac8660bafd7119f16d1dac0ea0d84f5054b07c3a7dbc39c 16e7f3b19933cf4e8af00ffa2c20337da2069f955270f603ae44792febfb86850
5	79154c3d14085457df43925ed291b608f2226cd60fbc16c3ad85bbb6d451cc5c 0fd11fed40fabb1947b77d5929f956d014e4e2fa9c68124a53ec51935c70b0ef
6	63c1c66f08b32d8f2882e8f2da1f2fc1f56bf592b0f50b099aeb0cf714598fc9 ae8826fc42caa1832c5ddb34d169f8e9c6364faa3b3b5464e19c4bad419ace34

7 dca04354d6d0fc8988787cfb21bf7a82533c3063157d04ab782d4df795e005e
79533d2d768bb8b7cc87ea9fd511f1ce6eeb2e87c7fcd45fb8169f606f404823

18 Abschluss

18.1 Datumsstempel

```
print(datestamp)
```

```
## [1] "2021-10-19"
```

18.2 Datum und Uhrzeit (Anfang)

```
print(begin.script)
```

```
## [1] "2021-10-19 02:51:58 CEST"
```

18.3 Datum und Uhrzeit (Ende)

```
end.script <- Sys.time()  
print(end.script)
```

```
## [1] "2021-10-19 09:24:55 CEST"
```

18.4 Laufzeit des gesamten Skriptes

```
print(end.script - begin.script)
```

```
## Time difference of 6.54903 hours
```

18.5 Warnungen

```
warnings()
```

19 Parameter für strenge Replikationen

```
system2("openssl", "version", stdout = TRUE)
```

```
## [1] "OpenSSL 1.1.1k 25 Mar 2021"
```

```
sessionInfo()
```

```
## R version 4.0.5 (2021-03-31)
## Platform: x86_64-redhat-linux-gnu (64-bit)
## Running under: Fedora 34 (Workstation Edition)
##
## Matrix products: default
## BLAS/LAPACK: /usr/lib64/libflexiblas.so.3.0
##
## locale:
##  [1] LC_CTYPE=en_US.utf8      LC_NUMERIC=C
##  [3] LC_TIME=en_US.utf8      LC_COLLATE=en_US.utf8
##  [5] LC_MONETARY=en_US.utf8  LC_MESSAGES=en_US.utf8
##  [7] LC_PAPER=en_US.utf8     LC_NAME=C
##  [9] LC_ADDRESS=C            LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.utf8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] parallel stats      graphics grDevices utils      datasets methods
## [8] base
##
## other attached packages:
##  [1] spacyr_1.2.1      quanteda_3.1.0    readtext_0.81     data.table_1.14.0
##  [5] scales_1.1.1      ggplot2_3.3.5     doParallel_1.0.16 iterators_1.0.13
##  [9] foreach_1.5.1     pdftools_3.0.1    kableExtra_1.3.4  knitr_1.34
## [13] rvest_1.0.1       httr_1.4.2
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.7        svglite_2.0.0     lattice_0.20-44   png_0.1-7
##  [5] digest_0.6.27     utf8_1.2.2        R6_2.5.1          evaluate_0.14
##  [9] highr_0.9         pillar_1.6.2      rlang_0.4.11      curl_4.3.2
## [13] rstudioapi_0.13   magick_2.7.3      Matrix_1.3-4      reticulate_1.20
## [17] rmarkdown_2.10    qpdf_1.1          labeling_0.4.2     webshot_0.5.2
## [21] stringr_1.4.0     selectr_0.4-2     munsell_0.5.0     compiler_4.0.5
## [25] xfun_0.25         pkgconfig_2.0.3   askpass_1.1       systemfonts
## [29] htmtltools_0.5.2  tidyselect_1.1.1  tibble_3.1.4      codetools_0.2-18
## [33] fansi_0.5.0       viridisLite_0.4.0 crayon_1.4.1      dplyr_1.0.7
## [37] withr_2.4.2       rappdirs_0.3.3    grid_4.0.5        jsonlite_1.7.2
## [41] gtable_0.3.0      lifecycle_1.0.0   magrittr_2.0.1    RcppParallel
## [45] stringi_1.7.4     farver_2.1.0      xml2_1.3.2        ellipsis_0.3.2
## [49] stopwords_2.2     generics_0.1.0    vctrs_0.3.8       fastmatch_1.1-3
```

## [53]	tools_4.0.5	glue_1.4.2	purrr_0.3.4	fastmap_1.1.0
## [57]	yaml_2.2.1	colorspace_2.0-2		

Literaturverzeichnis

- Analytics, Revolution, and Steve Weston. 2020. *Iterators: Provides Iterator Construct*. <https://github.com/RevolutionAnalytics/iterators>.
- Benoit, Kenneth, and Akitaka Matsuo. 2020. *Spacyr: Wrapper to the spaCy 'Nlp' Library*. <https://spacyr.quanteda.io>.
- Benoit, Kenneth, and Adam Obeng. 2021. *Readtext: Import and Handling for Plain and Formatted Text Files*. <https://github.com/quanteda/readtext>.
- Benoit, Kenneth, Kohei Watanabe, Haiyan Wang, Paul Nulty, Adam Obeng, Stefan Müller, and Akitaka Matsuo. 2018. "Quanteda: An R Package for the Quantitative Analysis of Textual Data." *Journal of Open Source Software* 3 (30): 774. <https://doi.org/10.21105/joss.00774>.
- Benoit, Kenneth, Kohei Watanabe, Haiyan Wang, Paul Nulty, Adam Obeng, Stefan Müller, Akitaka Matsuo, and William Lowe. 2021. *Quanteda: Quantitative Analysis of Textual Data*. <https://quanteda.io>.
- Corporation, Microsoft, and Steve Weston. 2020. *DoParallel: Foreach Parallel Adaptor for the Parallel Package*. <https://CRAN.R-project.org/package=doParallel>.
- Dowle, Matt, and Arun Srinivasan. 2021. *Data.table: Extension of 'Data.frame'*. <https://CRAN.R-project.org/package=data.table>.
- Ooms, Jeroen. 2021. *Pdftools: Text Extraction, Rendering and Converting of Pdf Documents*. <https://CRAN.R-project.org/package=pdfutils>.
- R Core Team. 2021. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. <https://www.R-project.org/>.
- Revolution Analytics, and Steve Weston. n.d. *Foreach: Provides Foreach Looping Construct*.
- Wickham, Hadley. 2016. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. <https://ggplot2.tidyverse.org>.
- . 2020. *Httr: Tools for Working with Urls and Http*. <https://CRAN.R-project.org/package=httr>.
- . 2021. *Rvest: Easily Harvest (Scrape) Web Pages*. <https://CRAN.R-project.org/package=rvest>.
- Wickham, Hadley, Winston Chang, Lionel Henry, Thomas Lin Pedersen, Kohske Takahashi, Claus Wilke, Kara Woo, Hiroaki Yutani, and Dewey Dunnington. 2021. *Ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics*. <https://CRAN.R-project.org/package=ggplot2>.
- Wickham, Hadley, and Dana Seidel. 2020. *Scales: Scale Functions for Visualization*. <https://CRAN.R-project.org/package=scales>.
- Xie, Yihui. 2014. "Knitr: A Comprehensive Tool for Reproducible Research in R." In *Implementing Reproducible Computational Research*, edited by Victoria Stodden, Friedrich Leisch, and Roger D. Peng. Chapman; Hall/CRC. <http://www.crcpress.com/product/isbn/9781466561595>.

- . 2015. *Dynamic Documents with R and Knitr*. 2nd ed. Boca Raton, Florida: Chapman; Hall/CRC. <https://yihui.org/knitr/>.
- . 2021. *Knitr: A General-Purpose Package for Dynamic Report Generation in R*. <https://yihui.org/knitr/>.
- Zhu, Hao. 2021. *KableExtra: Construct Complex Table with Kable and Pipe Syntax*. <https://CRAN.R-project.org/package=kableExtra>.