Corpus der Entscheidungen des Bundesverwaltungsgerichts

(CE-BVerwG-Source)

Compilation Report

Version 2021-04-15

DOI: 10.5281/zenodo.4625135

Titel	Source Code des »Corpus der Entscheidungen des Bundesverwaltungsgerichts«
Abkürzung	CE-BVerwG-Source
Autor	Seán Fobbe
Version	2021-04-15
Download	https://doi.org/10.5281/zenodo.4625135
Lizenz	MIT No Attribution (MIT-0)

Zitiervorschlag

Seán Fobbe (2021). Source Code des »Corpus der Entscheidungen des Bundesverwaltungsgerichts« (CE-BVerwG-Source). Version 2021-04-15. Zenodo. DOI: 10.5281/zenodo.4625135.

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 2021-04-15. 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)

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

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 kryptographisch sichere Signaturen (SHA2-256 und SHA3-512) berechnet und in einer CSV-Datei hinterlegt.

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.4625123" # checked
```

2.4 Verzeichnis für Analyse-Ergebnisse

Hinweis: Muss mit einem Schrägstrich enden!

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

2.5 Download-Umfang definieren

```
scope <- seq(from = 1,
             to = 25000,
             by = 1000)
```

2.6 Debugging-Modus

Der Debuging-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
```

2.7 Optionen: Quanteda

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

2.8 Optionen: Knitr

2.8.1 Ausgabe-Format

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

2.8.2 DPI für Raster-Grafiken

```
dpi <- 300
```

2.8.3 Ausrichtung von Grafiken im Compilation Report

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

2.9 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] "2021-04-15"
```

3.2 Datum und Uhrzeit (Beginn)

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

```
## [1] "2021-04-15 01:37:44 CEST"
```

3.3 Ordner für Analyse-Ergebnisse erstellen

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

3.4 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 0.84.0
```

```
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: 2.1.2
```

```
## Parallel computing: 2 of 16 threads used.
```

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

```
##
## Attaching package: 'quanteda'
```

```
## The following object is masked from 'package:utils':
##
##     View
```

3.5 Zusätzliche Funktionen einlesen

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

```
source("General_Source_Functions.R")
```

3.6 Quanteda-Optionen setzen

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

3.7 Knitr Optionen setzen

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

3.8 Vollzitate statistischer Software

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

```
## tweaking foreach
```

3.9 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).

3.9.1 Logische Kerne

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

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

3.9.2 Quanteda

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

3.9.3 Data.table

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

4 Download: Weitere Datensätze

4.1 Registerzeichen und Verfahrensarten

Datenquelle: »Seán Fobbe (2021). Aktenzeichen der Bundesrepublik Deutschland (AZ-BRD). Version 1.0.1. Zenodo. DOI: 10.5281/zenodo.4569564.«

```
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

Datenquelle: »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

Datenquelle: »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: 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"
```

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

5.3 Anzahl Links

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

```
## [1] 49904
```

5.4 Anzahl einzigartiger Links

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

```
## [1] 24950
```

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(filenames1,
                            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("^BVerwG_[0-9]{4}-[0-9]{2}-[0-9]{2}_[UBG]_[0-9NA]+_[A-Za-z-]+_
    [0-9]+_[0-9]+_[NAD]+_[0-9]\\.pdf$",
     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] "2021-04-15 01:38:53 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]): 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]): 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] "2021-04-15 08:35:05 CEST"
```

5.17 Dauer: Download

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

```
## Time difference of 6.93655 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] 24945
```

5.19.2 Anzahl heruntergeladener Dateien

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

```
## [1] 24943
```

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$links.pdf[i])
        Sys.sleep(runif(1, 1, 3))
        if (response$headers$"content-type" == "application/pdf" & response$
    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] 24945
```

5.22.2 Anzahl heruntergeladener Dateien

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

```
## [1] 24943
```

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] 24943
```

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 2021-04-15 08:35:18"
## [1] "Processed 24943 files. Runtime was 1.23 mins. Ended at 2021-04-15
    08:36:32."
```

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 größeren 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 Hypothethisches 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

```
vpraesi <- fread("PVP-FCG_2021-04-08_GermanFederalCourts_VicePresidents.csv")
vpraesi <- vpraesi[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 Hypothethisches 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[verzoegerung == "D"]$aktenzeichen <- paste(txt.bverwg[verzoegerung ==
    "D"]$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$verzoegerung),
                                  ".",
                                  txt.bverwg$verzoegerung),
                           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 <- ecli.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
##    ---
## 24939:   090321B3B33.19.0   090321B3B33.19.0
## 24940:  100321B6KSt2.21.0  100321B6KSt2.21.0
## 24941:  100321B6KSt3.21.0  100321B6KSt3.21.0
## 24942:  100321B6KSt4.21.0  100321B6KSt4.21.0
```

```
## 24943:    170321B2B3.21.0    170321B2B3.21.0
```

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:2006:160506B3PKH15.05.0"
##  [2] "ECLI:DE:BVerwG:2002:110402B9B5.02.0"
##  [3] "ECLI:DE:BVerwG:2014:140714B6B2.14.0"
##  [4] "ECLI:DE:BVerwG:2005:170305B7B20.05.0"
##  [5] "ECLI:DE:BVerwG:2006:020206B3B94.05.0"
##  [6] "ECLI:DE:BVerwG:2014:060214B1WB39.13.0"
##  [7] "ECLI:DE:BVerwG:2018:260318B1VR1.18.0"
##  [8] "ECLI:DE:BVerwG:2003:240403U3C15.02.0"
##  [9] "ECLI:DE:BVerwG:2016:230816B4B25.16.0"
## [10] "ECLI:DE:BVerwG:2005:280105B6B68.04.0"
```

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]))
```

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"
```

8.4 Frequenztabellen erstellen

```
prefix <- paste0(datasetname,
                 "_01_Frequenztabelle_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	24943	100	100	100
Total	24943	100	100	100

Frequency Table for Variable: entscheidung_typ

3 unique value(s) detected.

entscheidung_typ	N	exactpercent	roundedpercent	cumulpercent
B	20625	82.6885298	82.69	82.69
G	20	0.0801828	0.08	82.77

(continued)

entscheidung_typ	N	exactpercent	roundedpercent	cumulpercent
U	4298	17.2312873	17.23	100.00
Total	24943	100.0000000	100.00	100.00

Frequency Table for Variable: spruchkoerper_az

12 unique value(s) detected.

spruchkoerper_az	N	exactpercent	roundedpercent	cumulpercent
NA	1	0.0040091	0.00	0.00
1	3513	14.0841118	14.08	14.09
2	3380	13.5508960	13.55	27.64
3	2501	10.0268612	10.03	37.67
4	2892	11.5944353	11.59	49.26
5	2670	10.7044060	10.70	59.96
6	2522	10.1110532	10.11	70.08
7	1824	7.3126729	7.31	77.39
8	2184	8.7559636	8.76	86.14
9	2161	8.6637534	8.66	94.81
10	1007	4.0372048	4.04	98.85
20	288	1.1546326	1.15	100.00
Total	24943	100.0000000	100.00	100.00

Frequency Table for Variable: registerzeichen

28 unique value(s) detected.

registerzeichen	N	exactpercent	roundedpercent	cumulpercent
A	1287	5.1597643	5.16	5.16
AV	170	0.6815539	0.68	5.84

(continued)

registerzeichen	N	exactpercent	roundedpercent	cumulpercent
B	13861	55.5707012	55.57	61.41
BN	1206	4.8350239	4.84	66.25
C	4544	18.2175360	18.22	84.46
CN	170	0.6815539	0.68	85.15
D	99	0.3969049	0.40	85.54
D-PKH	1	0.0040091	0.00	85.55
DB	53	0.2124845	0.21	85.76
DW	3	0.0120274	0.01	85.77
F	281	1.1265686	1.13	86.90
GrSen	1	0.0040091	0.00	86.90
KSt	181	0.7256545	0.73	87.63
P	240	0.9621938	0.96	88.59
PB	379	1.5194644	1.52	90.11
PKH	578	2.3172834	2.32	92.43
VR	501	2.0085796	2.01	94.44
WA	2	0.0080183	0.01	94.44
WB	678	2.7181975	2.72	97.16
WD	344	1.3791444	1.38	98.54
WDB	81	0.3247404	0.32	98.87
WDS-AV	4	0.0160366	0.02	98.88
WDS-KSt	4	0.0160366	0.02	98.90
WDS-PKH	1	0.0040091	0.00	98.90
WDS-VR	139	0.5572706	0.56	99.46
WDW	1	0.0040091	0.00	99.46
WNB	105	0.4209598	0.42	99.88
WRB	29	0.1162651	0.12	100.00
Total	24943	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	0.0040091	0.00	0.00
1	2	0.0080183	0.01	0.01
2	2269	9.0967406	9.10	9.11
3	1948	7.8098064	7.81	16.92
4	1822	7.3046546	7.30	24.22
5	1575	6.3143968	6.31	30.54
6	1670	6.6952652	6.70	37.23
7	1794	7.1923987	7.19	44.43
8	1450	5.8132542	5.81	50.24
9	1440	5.7731628	5.77	56.01
10	1445	5.7932085	5.79	61.80
11	1493	5.9856473	5.99	67.79
12	1293	5.1838191	5.18	72.97
13	964	3.8648118	3.86	76.84
14	864	3.4638977	3.46	80.30
15	936	3.7525558	3.75	84.06
16	1020	4.0893237	4.09	88.14
17	869	3.4839434	3.48	91.63
18	832	3.3356052	3.34	94.96
19	806	3.2313675	3.23	98.20
20	436	1.7479854	1.75	99.94
21	13	0.0521188	0.05	100.00
96	1	0.0040091	0.00	100.00
Total	24943	100.0000000	100.00	100.00

Frequency Table for Variable: verzoegerung

2 unique value(s) detected.

verzoegerung	N	exactpercent	roundedpercent	cumulpercent
NA	24877	99.7353967	99.74	99.74
D	66	0.2646033	0.26	100.00
Total	24943	100.0000000	100.00	100.00

Frequency Table for Variable: kollision

2 unique value(s) detected.

kollision	N	exactpercent	roundedpercent	cumulpercent
0	24925	99.9278355	99.93	99.93
1	18	0.0721645	0.07	100.00
Total	24943	100.0000000	100.00	100.00

Frequency Table for Variable: entscheidungsjahr

22 unique value(s) detected.

entscheidungsjahr	N	exactpercent	roundedpercent	cumulpercent
1997	1	0.0040091	0.00	0.00
2000	1	0.0040091	0.00	0.01
2002	1502	6.0217295	6.02	6.03
2003	2009	8.0543639	8.05	14.08
2004	1817	7.2846089	7.28	21.37
2005	1619	6.4907990	6.49	27.86
2006	1684	6.7513932	6.75	34.61
2007	1690	6.7754480	6.78	41.39
2008	1549	6.2101592	6.21	47.60

(continued)

entscheidungsjahr	N	exactpercent	roundedpercent	cumulpercent
2009	1352	5.4203584	5.42	53.02
2010	1474	5.9094736	5.91	58.93
2011	1553	6.2261957	6.23	65.15
2012	1281	5.1357094	5.14	70.29
2013	1072	4.2977990	4.30	74.59
2014	889	3.5641262	3.56	78.15
2015	845	3.3877240	3.39	81.54
2016	995	3.9890951	3.99	85.53
2017	960	3.8487752	3.85	89.38
2018	836	3.3516417	3.35	92.73
2019	887	3.5561079	3.56	96.28
2020	859	3.4438520	3.44	99.73
2021	68	0.2726216	0.27	100.00
Total	24943	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	0.0040091	0.00	0.00
2000	1	0.0040091	0.00	0.01
2001	2	0.0080183	0.01	0.02
2002	2269	9.0967406	9.10	9.11
2003	1948	7.8098064	7.81	16.92
2004	1822	7.3046546	7.30	24.23
2005	1575	6.3143968	6.31	30.54
2006	1670	6.6952652	6.70	37.24
2007	1794	7.1923987	7.19	44.43

(continued)

eingangsjahr_iso	N	exactpercent	roundedpercent	cumulpercent
2008	1450	5.8132542	5.81	50.24
2009	1440	5.7731628	5.77	56.02
2010	1445	5.7932085	5.79	61.81
2011	1493	5.9856473	5.99	67.79
2012	1293	5.1838191	5.18	72.98
2013	964	3.8648118	3.86	76.84
2014	864	3.4638977	3.46	80.31
2015	936	3.7525558	3.75	84.06
2016	1020	4.0893237	4.09	88.15
2017	869	3.4839434	3.48	91.63
2018	832	3.3356052	3.34	94.97
2019	806	3.2313675	3.23	98.20
2020	436	1.7479854	1.75	99.95
2021	13	0.0521188	0.05	100.00
Total	24943	100.0000000	100.00	100.00

Frequency Table for Variable: praesi

5 unique value(s) detected.

praesi	N	exactpercent	roundedpercent	cumulpercent
Eckertz-Höfer	9295	37.264964	37.26	37.26
Franßen	1049	4.205589	4.21	41.47
Hien	8346	33.460290	33.46	74.93
Rennert	5856	23.477529	23.48	98.41
VACANCY-5	397	1.591629	1.59	100.00
Total	24943	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	12.4964920	12.50	12.50
Eckertz-Höfer	8346	33.4602895	33.46	45.96
Franke	1	0.0040091	0.00	45.96
Hien	1048	4.2015796	4.20	50.16
Hund	6598	26.4523113	26.45	76.61
Korbmacher	1455	5.8332999	5.83	82.45
Rennert	1696	6.7995029	6.80	89.25
VACANCY-3	1398	5.6047789	5.60	94.85
VACANCY-4	1284	5.1477368	5.15	100.00
Total	24943	100.0000000	100.00	100.00

Frequency Table for Variable: verfahrensart

28 unique value(s) detected.

verfahrensart	N	exactpercent	roundedpercent	cumulpercent
Allgemeine Verfahren (Wehrdienstsenate)	4	0.0160366	0.02	0.02
Allgemeine Verfahren (sonstige Anträge außerhalb eines schwebenden Verfahrens)	170	0.6815539	0.68	0.70
Berufung (Wehrdisziplinarsachen)	344	1.3791444	1.38	2.08
Beschwerde (Disziplinarsachen)	53	0.2124845	0.21	2.29
Erstinstanzliche Klage, inklusive Wiederaufnahmeverfahren (Verwaltungsstreitsachen)	1287	5.1597643	5.16	7.45

(continued)

verfahrensart	N	exactpercent	roundedpercent	cumulpercent
Erstinstanzliche Klage, inklusive Wiederaufnahmeverfahren (Wehrdienstsenat)	2	0.0080183	0.01	7.46
Fachsenat (Verwaltungsstreitsachen)	281	1.1265686	1.13	8.58
Großer Senat des Bundesverwaltungsgerichts	1	0.0040091	0.00	8.59
Kostensachen	181	0.7256545	0.73	9.31
Kostensachen (Wehrdienstsenate)	4	0.0160366	0.02	9.33
Nichtzulassungsbeschwerde (Normenkontrollsachen)	1206	4.8350239	4.84	14.16
Nichtzulassungsbeschwerde (Personalvertretungs- und Richtervertretungssachen)	379	1.5194644	1.52	15.68
Nichtzulassungsbeschwerde nach Wehrbeschwerdeordnung	105	0.4209598	0.42	16.10
Nichtzulassungsbeschwerde oder Beschwerde (Verwaltungsstreitsachen)	13861	55.5707012	55.57	71.68
Prozesskostenhilfe	578	2.3172834	2.32	73.99
Prozesskostenhilfe (Disziplinarsachen)	1	0.0040091	0.00	74.00
Prozesskostenhilfe (Wehrdienstsenate)	1	0.0040091	0.00	74.00
Rechtsbeschwerde (Personalvertretungs- und Richtervertretungssachen)	240	0.9621938	0.96	74.96
Rechtsbeschwerde nach Wehrbeschwerdeordnung	29	0.1162651	0.12	75.08
Rechtsmittelverfahren über Anträge, Beschwerden und Vorlagen nach der Wehrdisziplinarordnung	81	0.3247404	0.32	75.40
Revision (Disziplinarsachen)	99	0.3969049	0.40	75.80

(continued)

verfahrensart	N	exactpercent	roundedpercent	cumulpercent
Revision (Normenkontrollsachen)	170	0.6815539	0.68	76.48
Revision (Verwaltungsstreitsachen)	4544	18.2175360	18.22	94.70
Verfahren nach Wehrbeschwerdeordnung	678	2.7181975	2.72	97.42
Vorläufiger Rechtsschutz	501	2.0085796	2.01	99.43
Vorläufiger Rechtsschutz (Wehrdienstsenate)	139	0.5572706	0.56	99.98
Wiederaufnahme (Disziplinarsachen)	3	0.0120274	0.01	100.00
Wiederaufnahme (Wehrdisziplinarsachen)	1	0.0040091	0.00	100.00
Total	24943	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	24943	100	100	100
Total	24943	100	100	100

Frequency Table for Variable: doi_version

1 unique value(s) detected.

doi_version	N	exactpercent	roundedpercent	cumulpercent
10.5281/zenodo.4625123	24943	100	100	100
Total	24943	100	100	100

Frequency Table for Variable: version

1 unique value(s) detected.

version	N	exactpercent	roundedpercent	cumulpercent
2021-04-15	24943	100	100	100
Total	24943	100	100	100

9 Frequenztabellen visualisieren

9.1 Präfix erstellen

```
prefix <- paste0("ANALYSE/",
                 datasetname,
                 "_01_Frequenztabelle_var-")
```

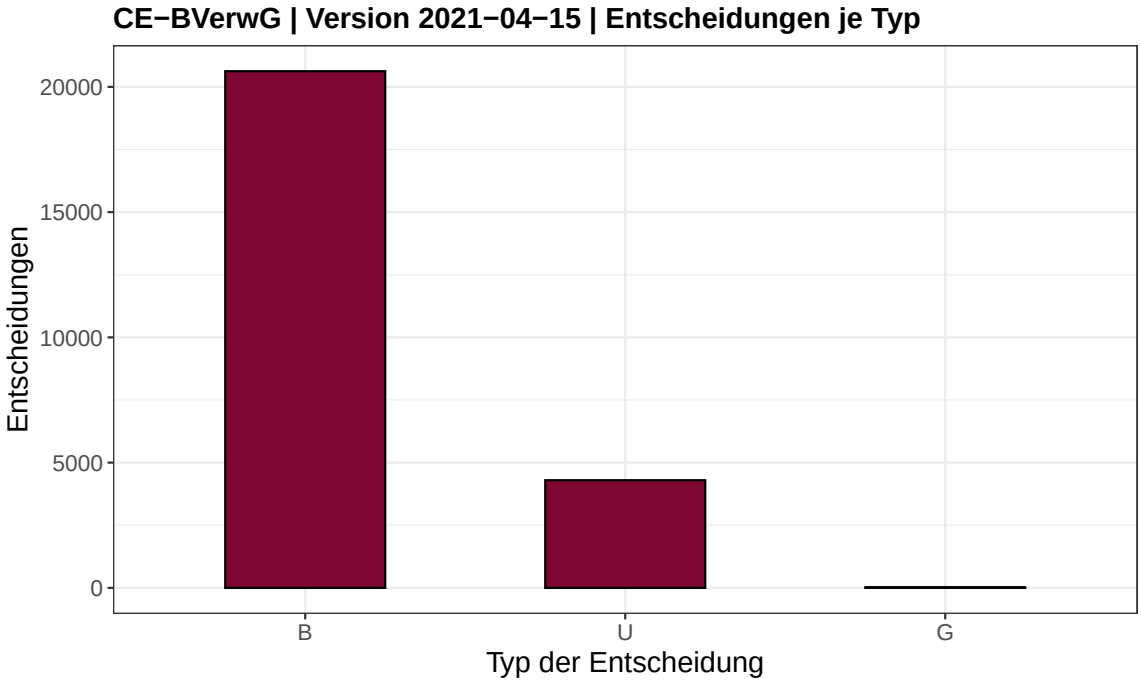
9.2 Tabellen einlesen

```
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 = paste("DOI:",
                        doi.version),
        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)
    )
```

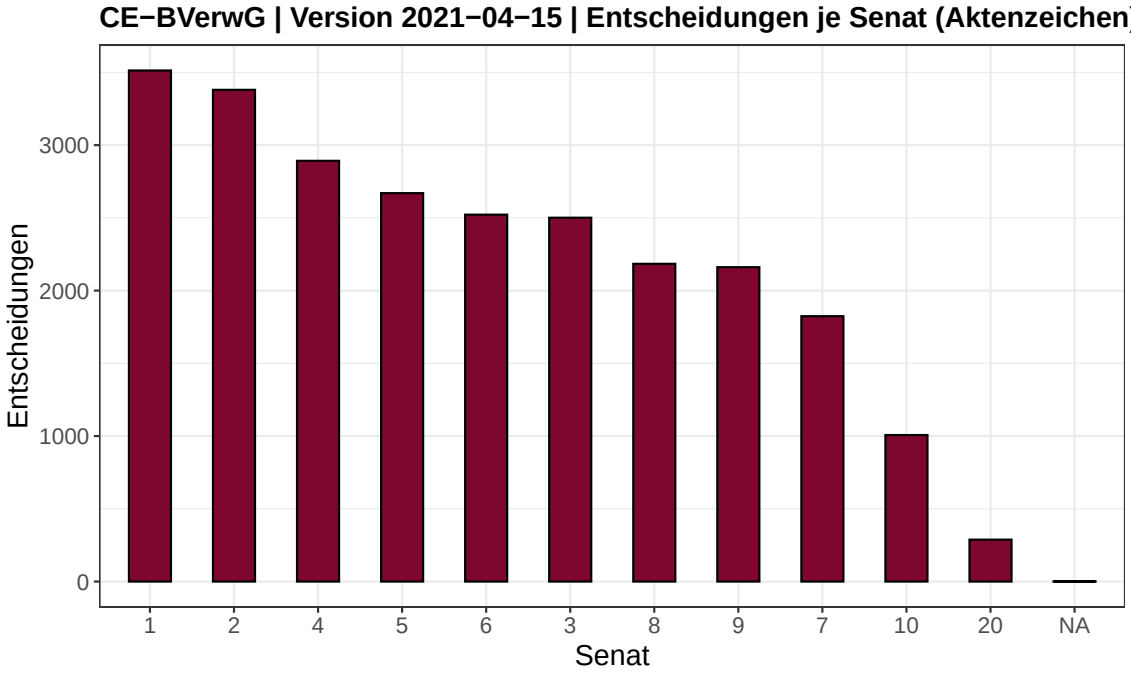


DOI: 10.5281/zenodo.4625123

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 = paste("DOI:",
                        doi.version),
        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)
    )
```

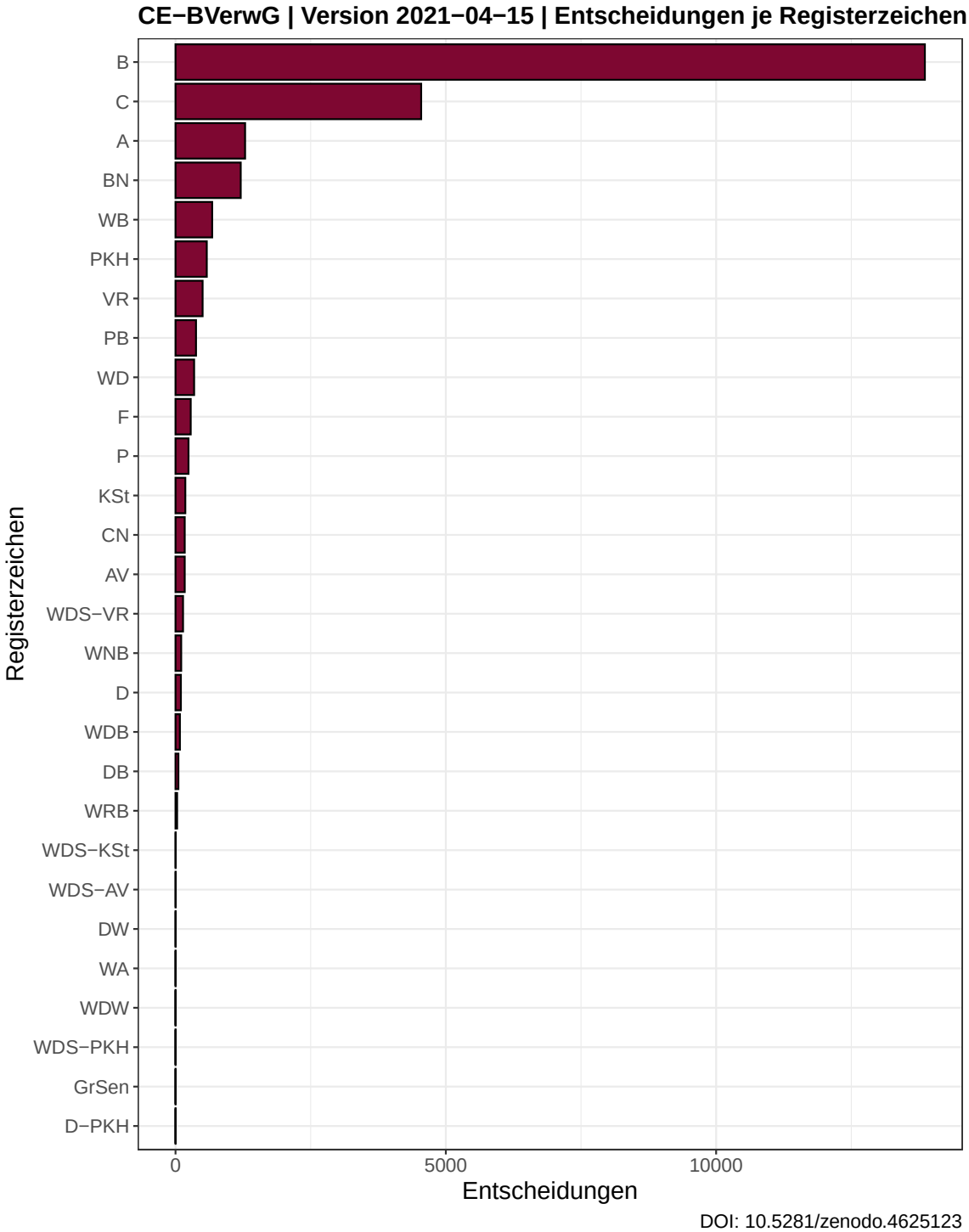


DOI: 10.5281/zenodo.4625123

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 = paste("DOI:",
                        doi.version),
        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)
    )
```

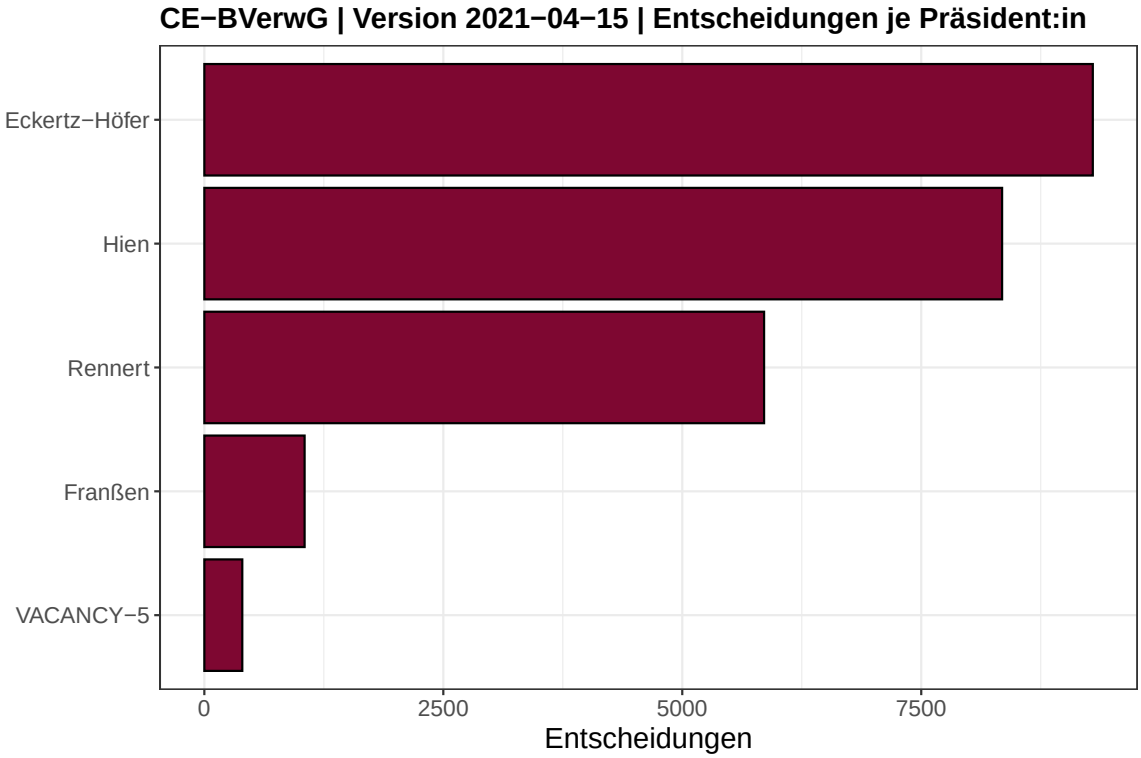


DOI: 10.5281/zenodo.4625123

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 = paste("DOI:",
                        doi.version),
        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)
    )
```

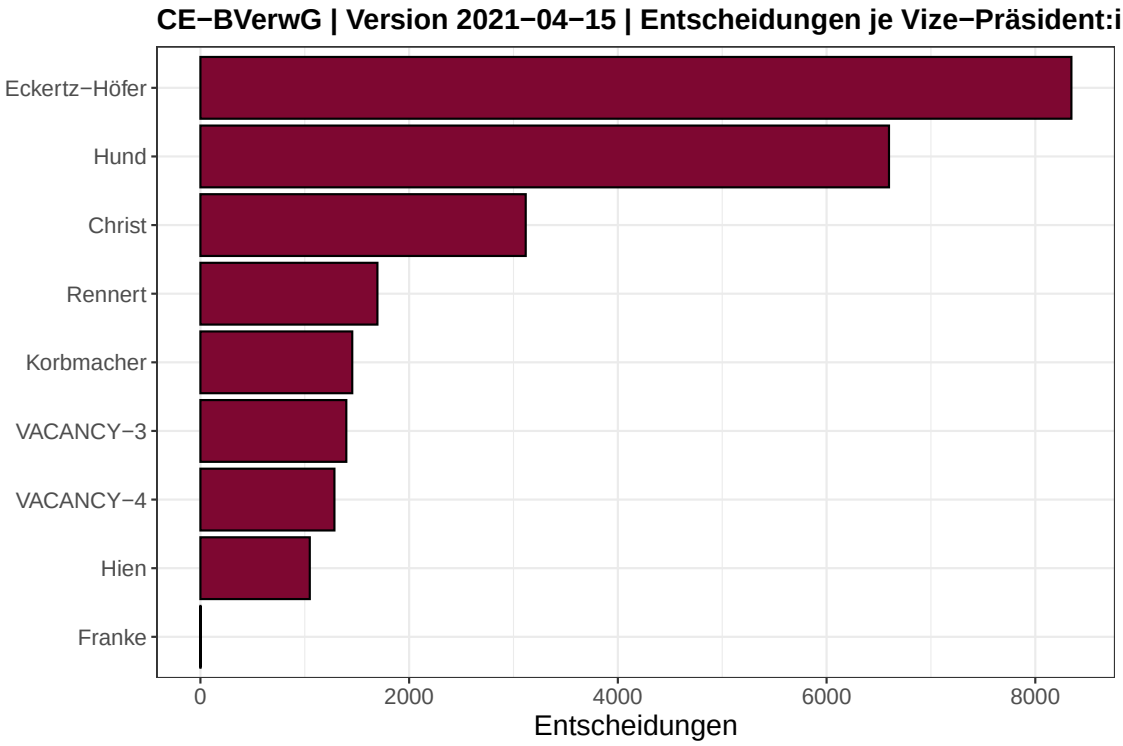


DOI: 10.5281/zenodo.4625123

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 = paste("DOI:",
                        doi.version),
        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)
    )
```

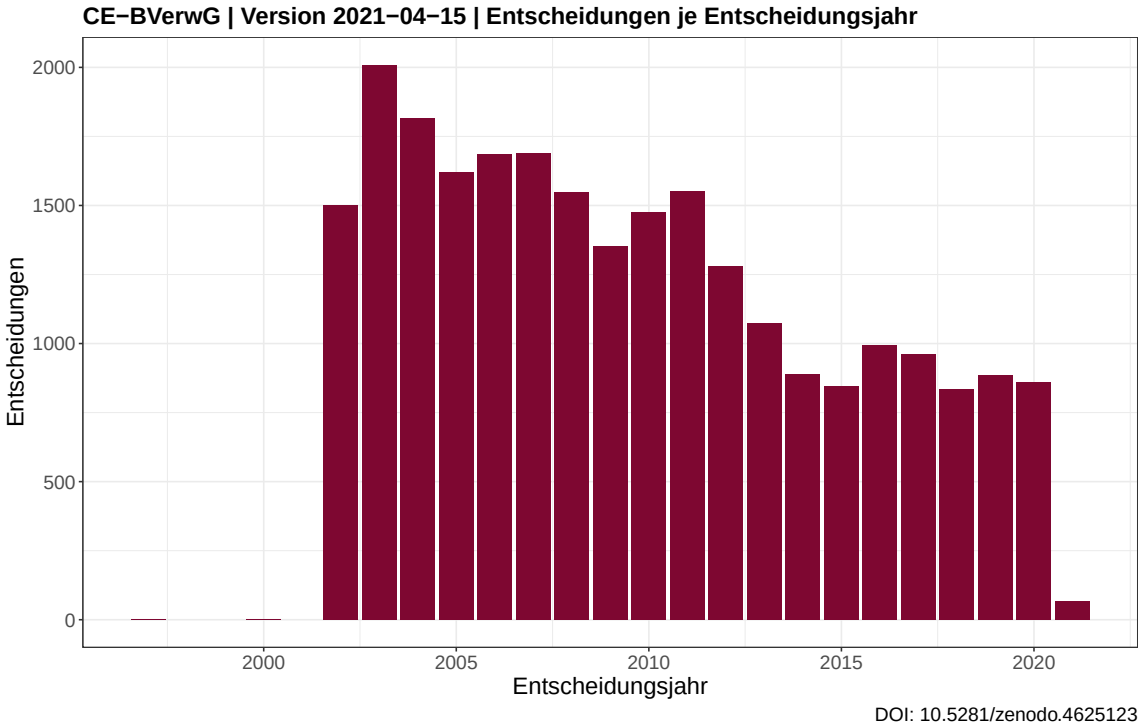


DOI: 10.5281/zenodo.4625123

9.8 Diagramm: Entscheidungsjahr

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

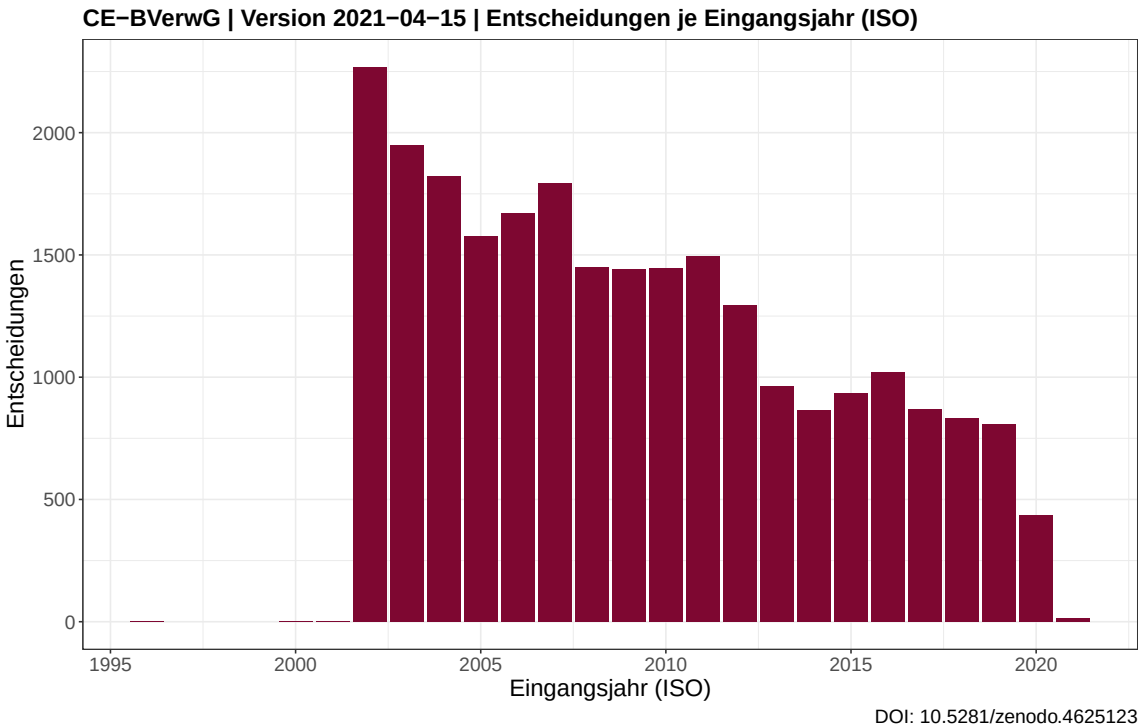
```
ggplot(data = freqtable) +
    geom_bar(aes(x = entscheidungsjahr,
                 y = N),
             stat = "identity",
             fill = "#7e0731") +
    theme_bw() +
    labs(
        title = paste(datasetname,
                      "| Version",
                      datestamp,
                      "| Entscheidungen je Entscheidungsjahr"),
        caption = paste("DOI:",
                        doi.version),
        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)
    )
```



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 = paste("DOI:",
                        doi.version),
        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)
    )
```



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.summarize.iterator,
      threads = fullCores,
      chunksize = 1)
```

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

```
##
##
##     end.dopar <- Sys.time()
##     duration.dopar <- end.dopar - begin.dopar
##
##     summary.corpus <- rbindlist(result)
##
##     setnames(summary.corpus,
##              old = c("Text",
##                      "Tokens",
##                      "Types",
##                      "Sentences"),
##              new = c("doc_id",
##                      "ntokens",
##                      "ntypes",
##                      "nsentences"))
##
##     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.summarize.iterator(txt.bverwg)
```

```
## [1] "Parallel processing using 16 threads. Begin at 2021-04-15 08:37:32.
    Processing 24943 documents with a total length of 338527212 characters."
## [1] "Runtime was 2.52 mins. Ended at 2021-04-15 08:40:03."
```

10.1.3 Variablen-Namen anpassen

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

setnames(txt.bverwg,
         old = "nchars",
         new = "zeichen")
```

10.1.4 Kennwerte dem Korpus hinzufügen

```
txt.bverwg$tokens <- summary.corpus$tokens
txt.bverwg$typen <- summary.corpus$typen
txt.bverwg$saetze <- summary.corpus$saetze
```

10.2 Zusammenfassungen: Linguistische Kennwerte

Hinweis: Typen sind definiert als einzigartige Tokens und werden für jedes Dokument gesondert berechnet. Daher ergibt es an dieser Stelle auch keinen Sinn die Typen zu summieren, denn bezogen auf den Korpus wäre der Kennwert ein anderer. Der Wert wird daher manuell auf »NA« gesetzt.

10.2.1 Zusammenfassungen berechnen

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


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

dt.sums.ling$typen <- NA




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"))
```

10.2.2 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	338,527,212	480	2,851	7,626	13,572.0327	17,877	642,248
tokens	52,027,453	62	410	1,163	2,085.8539	2,753	100,685
saetze	2,991,595	4	30	70	119.9373	156	5,000
typen	NA	47	217	465	610.9180	840	11,507

10.2.3 Zusammenfassungen speichern

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

10.3 Zusammenfassungen: Quantitative Variablen

10.3.1 Entscheidungsdatum

```
summary(as.IDate(summary.corpus$datum))
```

```
##         Min.      1st Qu.       Median         Mean      3rd Qu.         Max.
## "1997-02-26" "2005-07-14" "2009-05-28" "2010-02-11" "2014-02-11" "2021-03-17"
```

10.3.2 Zusammenfassungen berechnen

```
dt.summary.docvars <- summary.corpus[,
                                     lapply(.SD, function(x)unclass(summary(na.
    omit(x)))),
                                     .SDcols = c("entscheidungsjahr",
                                                 "eingangsjahr_iso",
                                                 "eingangsnummer")]


dt.unique.docvars <- summary.corpus[,
                                    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"))
```

10.3.3 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	2009.61737	2014	2021
eingangsjahr_iso	23	1996	2005	2008	2009.14541	2013	2021
eingangsnummer	573	1	10	27	62.47031	59	7005

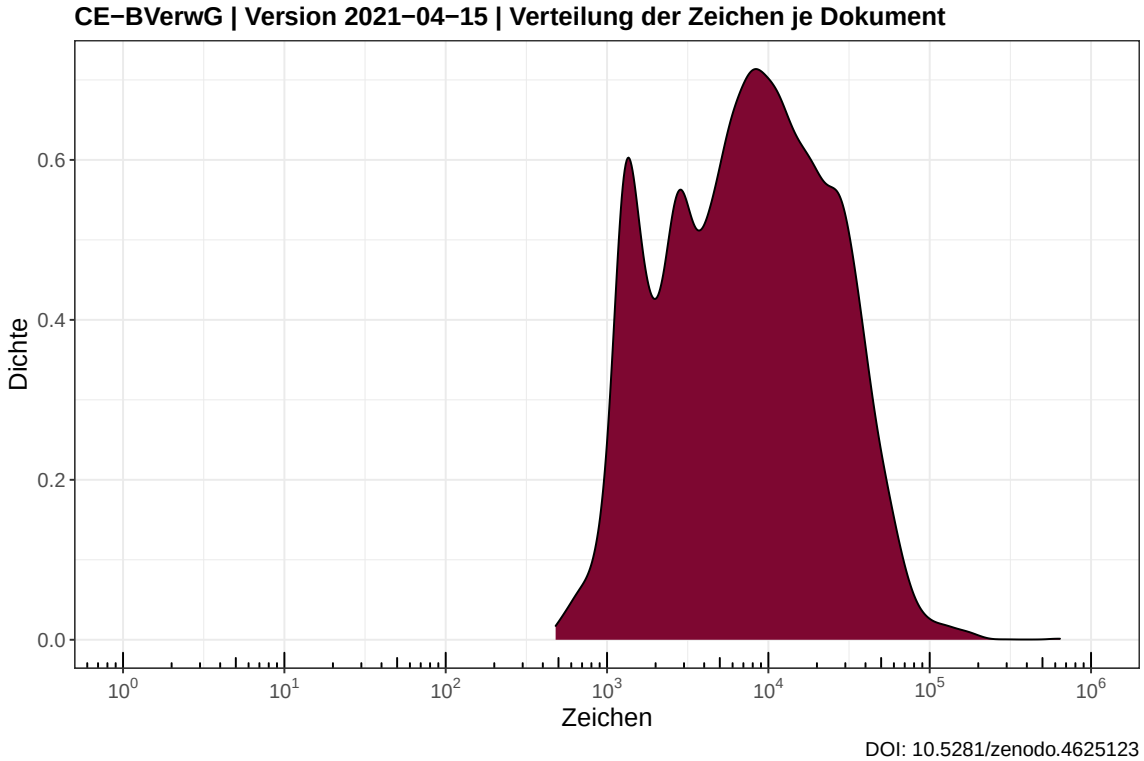
10.3.4 Zusammenfassungen speichern

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

10.4 Verteilungen linguistischer Kennwerte

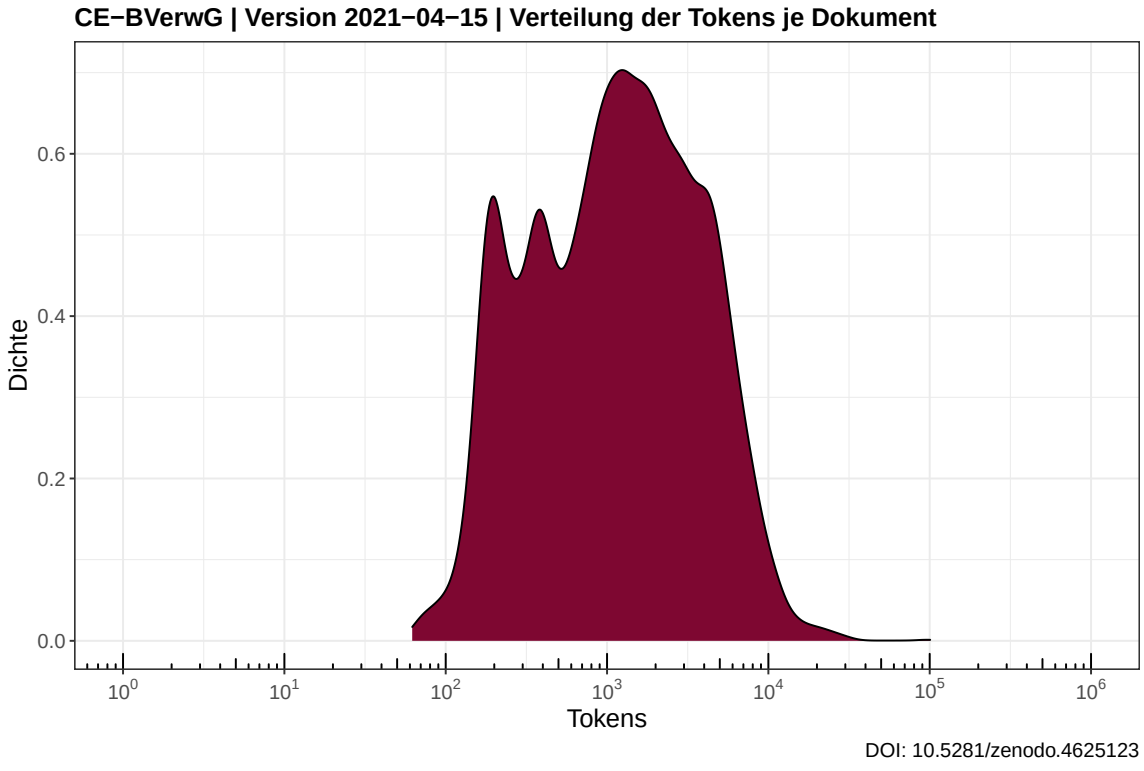
10.4.1 Diagramm: Verteilung Zeichen

```
ggplot(data = summary.corpus)+
    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 = paste("DOI:",
                        doi.version),
        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)
    )
```



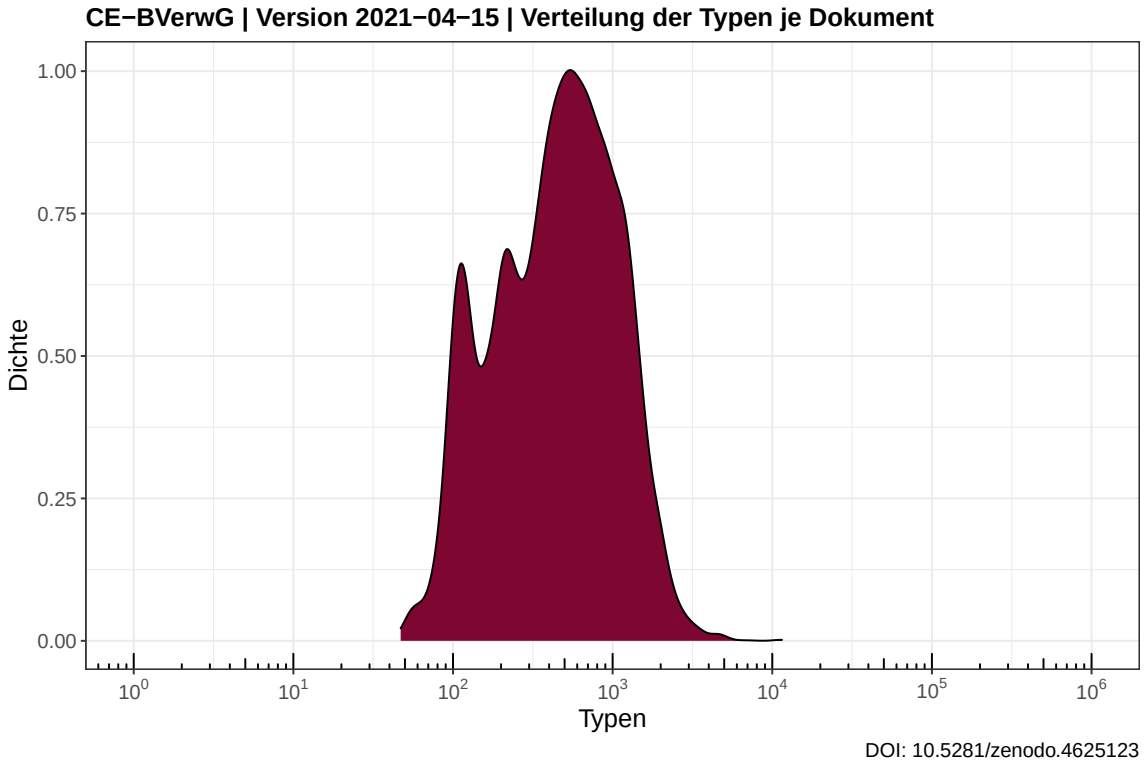
10.4.2 Diagramm: Verteilung Tokens

```
ggplot(data = summary.corpus)+
    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 = paste("DOI:",
                        doi.version),
        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)
    )
```



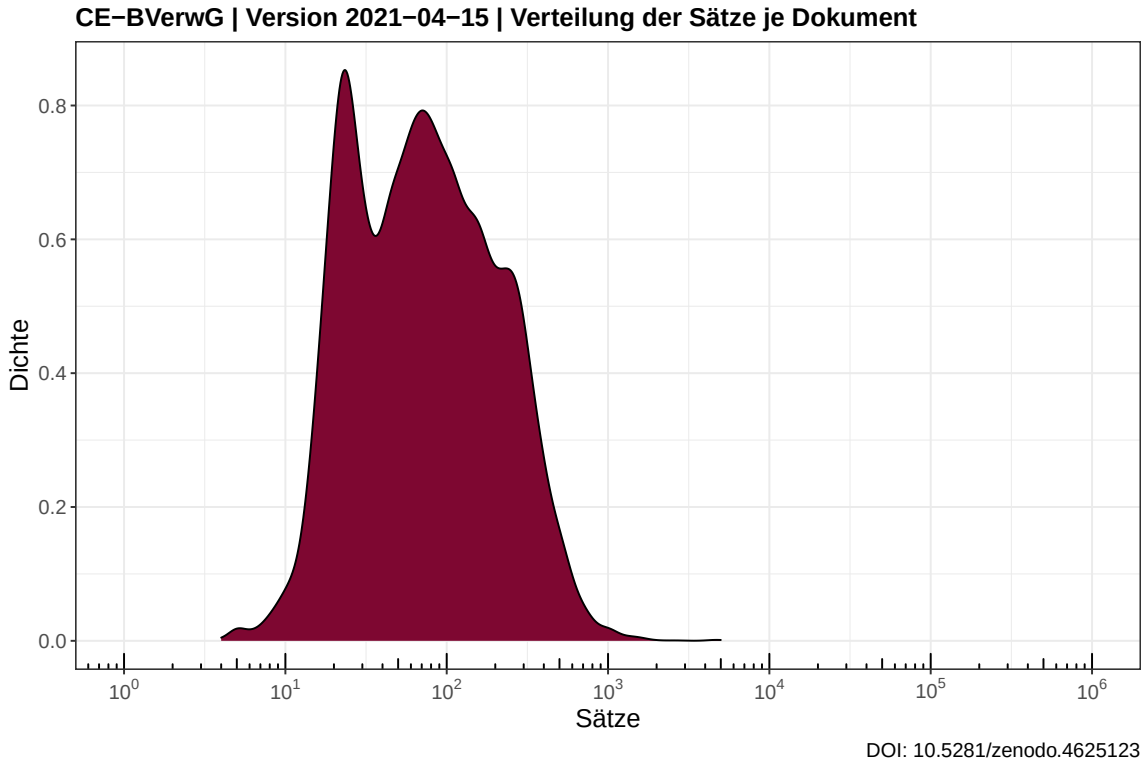
10.4.3 Diagramm: Verteilung Typen

```
ggplot(data = summary.corpus)+
    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 = paste("DOI:",
                        doi.version),
        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)
    )
```



10.4.4 Diagramm: Verteilung Sätze

```
ggplot(data = summary.corpus)+
    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 = paste("DOI:",
                        doi.version),
        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)
    )
```



10.5 Anzahl Variablen im Korpus

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

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

10.6 Namen der Variablen im Korpus

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

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

11 Beispiel-Werte für alle Metadaten anzeigen

```
print(summary.corpus)
```

```
##                                       doc_id typen tokens saetze gericht
##     1:    BVerwG_1997-02-26_U_6_C_3_96_NA_0.txt  1470   5471    267  BVerwG
##     2: BVerwG_2000-12-14_B_1_WB_107_00_NA_0.txt   721   2336    143  BVerwG
##     3:    BVerwG_2002-01-10_B_9_A_9_02_NA_0.txt    84    128     15  BVerwG
##     4:   BVerwG_2002-01-14_B_4_BN_1_02_NA_0.txt   277    557     48  BVerwG
##     5:   BVerwG_2002-01-17_B_1_B_12_02_NA_0.txt   128    213     18  BVerwG
##    ---
## 24939:   BVerwG_2021-03-09_B_3_B_33_19_NA_0.txt   215    425     33  BVerwG
## 24940:  BVerwG_2021-03-10_B_6_KSt_2_21_NA_0.txt   227    531     45  BVerwG
## 24941:  BVerwG_2021-03-10_B_6_KSt_3_21_NA_0.txt   243    573     47  BVerwG
## 24942:  BVerwG_2021-03-10_B_6_KSt_4_21_NA_0.txt   298    748     58  BVerwG
## 24943:    BVerwG_2021-03-17_B_2_B_3_21_NA_0.txt  1084   4505    305  BVerwG
##             datum entscheidung_typ spruchkoerper_az registerzeichen
##     1: 1997-02-26               U               6              C
##     2: 2000-12-14               B               1             WB
##     3: 2002-01-10               B               9              A
##     4: 2002-01-14               B               4             BN
##     5: 2002-01-17               B               1              B
##    ---
## 24939: 2021-03-09               B               3              B
## 24940: 2021-03-10               B               6            KSt
## 24941: 2021-03-10               B               6            KSt
## 24942: 2021-03-10               B               6            KSt
## 24943: 2021-03-17               B               2              B
##        eingangsnummer eingangsjahr_az verzoegerung kollision entscheidungsjahr
##     1:              3              96         <NA>         0              1997
##     2:            107               0         <NA>         0              2000
##     3:              9               2         <NA>         0              2002
##     4:              1               2         <NA>         0              2002
##     5:             12               2         <NA>         0              2002
##    ---
## 24939:             33              19         <NA>         0              2021
## 24940:              2              21         <NA>         0              2021
## 24941:              3              21         <NA>         0              2021
## 24942:              4              21         <NA>         0              2021
## 24943:              3              21         <NA>         0              2021
##        eingangsjahr_iso  praesi   v_praesi      aktenzeichen
##     1:             1996 Franßen     Franke   BVerwG 6 C 3.96
##     2:             2000 Franßen       Hien BVerwG 1 WB 107.0
##     3:             2002 Franßen       Hien    BVerwG 9 A 9.2
##     4:             2002 Franßen       Hien   BVerwG 4 BN 1.2
##     5:             2002 Franßen       Hien   BVerwG 1 B 12.2
##    ---
## 24939:             2019 Rennert Korbmacher  BVerwG 3 B 33.19
## 24940:             2021 Rennert Korbmacher BVerwG 6 KSt 2.21
## 24941:             2021 Rennert Korbmacher BVerwG 6 KSt 3.21
## 24942:             2021 Rennert Korbmacher BVerwG 6 KSt 4.21
## 24943:             2021 Rennert Korbmacher   BVerwG 2 B 3.21
##                                       ecli
```

```
##     1:    ECLI:DE:BVerwG:1997:260297U6C3.96.0
##     2: ECLI:DE:BVerwG:2000:141200B1WB107.00.0
##     3:    ECLI:DE:BVerwG:2002:100102B9A9.02.0
##     4:   ECLI:DE:BVerwG:2002:140102B4BN1.02.0
##     5:   ECLI:DE:BVerwG:2002:170102B1B12.02.0
##    ---
## 24939:   ECLI:DE:BVerwG:2021:090321B3B33.19.0
## 24940:  ECLI:DE:BVerwG:2021:100321B6KSt2.21.0
## 24941:  ECLI:DE:BVerwG:2021:100321B6KSt3.21.0
## 24942:  ECLI:DE:BVerwG:2021:100321B6KSt4.21.0
## 24943:    ECLI:DE:BVerwG:2021:170321B2B3.21.0
##
    verfahrensart
##     1:                                                  Revision (
    Verwaltungsstreitsachen)
##     2:                                                Verfahren nach
    Wehrbeschwerdeordnung
##     3: Erstinstanzliche Klage, inklusive Wiederaufnahmeverfahren (
    Verwaltungsstreitsachen)
##     4:                                   Nichtzulassungsbeschwerde (
    Normenkontrollsachen)
##     5:                Nichtzulassungsbeschwerde oder Beschwerde (
    Verwaltungsstreitsachen)
##    ---
## 24939:                Nichtzulassungsbeschwerde oder Beschwerde (
    Verwaltungsstreitsachen)
## 24940:
    Kostensachen
## 24941:
    Kostensachen
## 24942:
    Kostensachen
## 24943:                Nichtzulassungsbeschwerde oder Beschwerde (
    Verwaltungsstreitsachen)
##                   doi_concept            doi_version    version zeichen
##     1: 10.5281/zenodo.3911067 10.5281/zenodo.4625123 2021-04-15   37358
##     2: 10.5281/zenodo.3911067 10.5281/zenodo.4625123 2021-04-15   13741
##     3: 10.5281/zenodo.3911067 10.5281/zenodo.4625123 2021-04-15     805
##     4: 10.5281/zenodo.3911067 10.5281/zenodo.4625123 2021-04-15    3619
##     5: 10.5281/zenodo.3911067 10.5281/zenodo.4625123 2021-04-15    1384
##    ---
## 24939: 10.5281/zenodo.3911067 10.5281/zenodo.4625123 2021-04-15    2920
## 24940: 10.5281/zenodo.3911067 10.5281/zenodo.4625123 2021-04-15    3424
## 24941: 10.5281/zenodo.3911067 10.5281/zenodo.4625123 2021-04-15    3637
## 24942: 10.5281/zenodo.3911067 10.5281/zenodo.4625123 2021-04-15    4524
## 24943: 10.5281/zenodo.3911067 10.5281/zenodo.4625123 2021-04-15   26556
```

12 CSV-Dateien erstellen

12.1 CSV mit vollem Datensatz speichern

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

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

12.2 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(summary.corpus,
       csvname.meta,
       na = "NA")
```

13 Dateigrößen analysieren

13.1 Gesamtgröße

13.1.1 Korpus-Objekt in RAM (MB)

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

```
## 361 MB
```

13.1.2 CSV Korpus (MB)

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

```
## [1] 351.9359
```

13.1.3 CSV Metadaten (MB)

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

```
## [1] 7.804763
```

13.1.4 PDF-Dateien (MB)

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

pdf.MB <- file.size(files.pdf) / 10^6
sum(pdf.MB)
```

```
## [1] 1293.563
```

13.1.5 TXT-Dateien (MB)

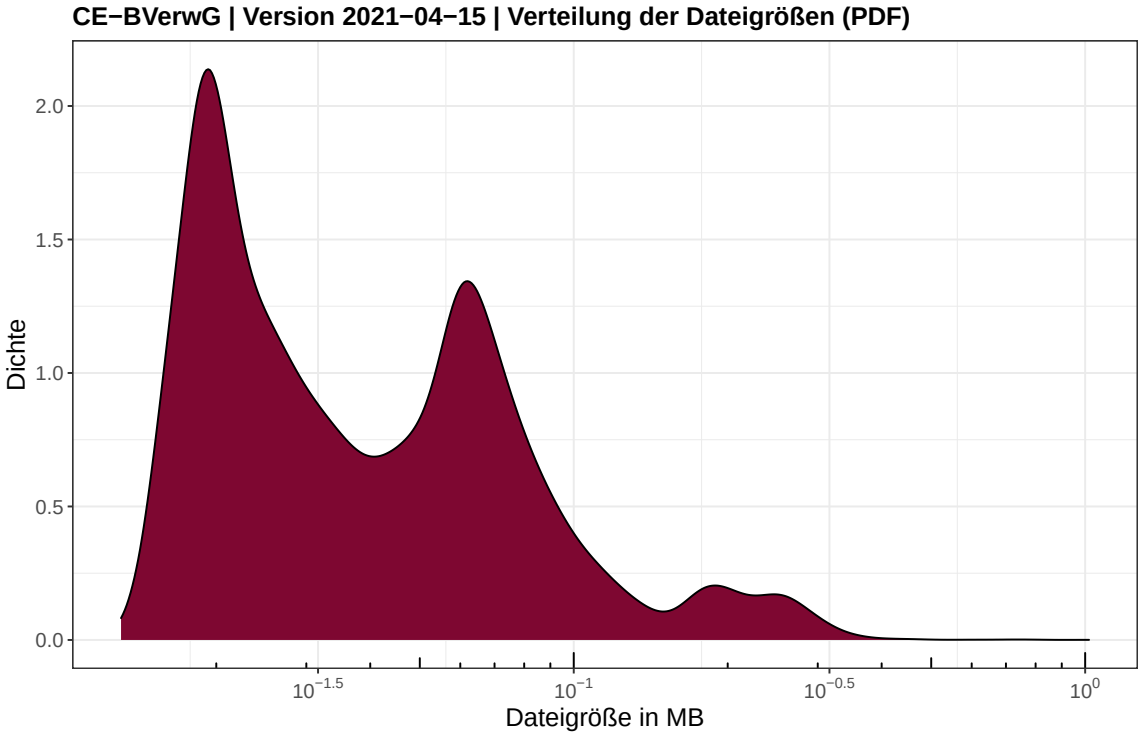
```
files.txt <- list.files(pattern = "\\.txt$",
                        ignore.case = TRUE)

txt.MB <- file.size(files.txt) / 10^6
sum(txt.MB)
```

```
## [1] 351.0307
```

13.2 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 = paste("DOI:",
                        doi.version),
        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)
    )
```

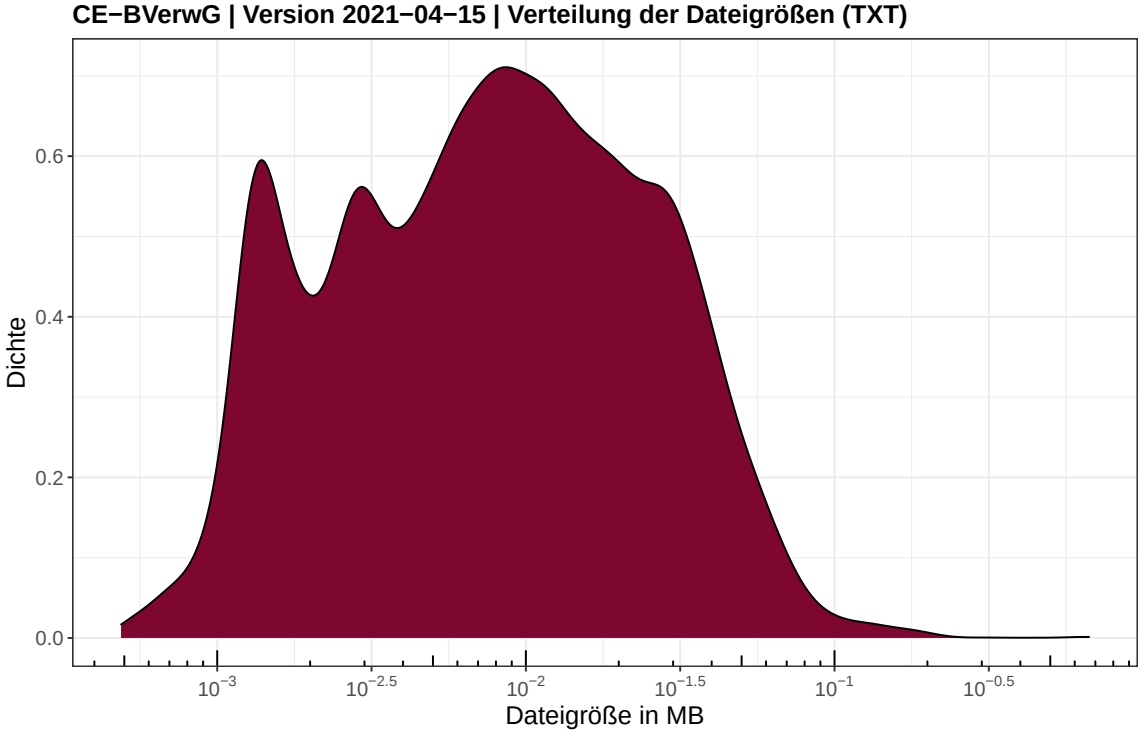


DOI: 10.5281/zenodo.4625123

13.3 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 = paste("DOI:",
                        doi.version),
        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)
    )
```



DOI: 10.5281/zenodo.4625123

14 Erstellen der ZIP-Archive

14.1 Verpacken der CSV-Dateien

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

zip(csvname.full.zip,
    csvname.full)

unlink(csvname.full)
```

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

zip(csvname.meta.zip,
    csvname.meta)

unlink(csvname.meta)
```

14.2 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)
```

14.3 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)
```

14.4 Verpacken der Analyse-Dateien

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

14.5 Verpacken der Source-Dateien

```
files.source <- c(list.files(pattern = "Source"),
                  "buttons")


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

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

15 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.

15.1 Liste der ZIP-Archive erstellen

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

15.2 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)
```

}

15.3 Hashes berechnen

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

```
## [1] "Parallel processing using 16 threads."
## [1] "Processed 6 files. Runtime was 4.47 secs."
```

15.4 In Data Table umwandeln

```
setDT(multihashes)
```

15.5 Index hinzufügen

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

15.6 In Datei schreiben

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

15.7 Leerzeichen hinzufügen um Zeilenumbruch zu ermöglichen

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

15.8 In Bericht anzeigen

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

index	filename
1	CE-BVerwG_2021-04-15_DE_ANALYSE.zip
2	CE-BVerwG_2021-04-15_DE_CSV_Datensatz.zip
3	CE-BVerwG_2021-04-15_DE_CSV_Metadaten.zip
4	CE-BVerwG_2021-04-15_DE_PDF_Datensatz.zip
5	CE-BVerwG_2021-04-15_DE_TXT_Datensatz.zip
6	CE-BVerwG_2021-04-15_Source_Files.zip

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

index	sha2.256
1	6268c8e843ab5a6f260c6d65187bc6b6f447a64c84411cdb7aa4b96c5806a6af
2	750314eef6346cea8e45f55cb2185665ea1722786a967c844324799928a15ecf
3	0dca8048cc2c8d32016750b2d061f814124d5cac58902442209897a0931b485f
4	fcfdcb497f6c4fa7eff22329b82290707149560e4fd51e588049a074d89805ad
5	24b5f0cb086da5709672d0997c38f936af7519db72649daf6d5ae591bc1ab274
6	205369c44bd6da06a7251603d77d6fc963fe7e74a0760ea473eb1121c3431b51

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

index	sha3.512
1	238adc803270a697f5d21dd2dd04f1ee896011c041943416fad9e58b8854913029cb58074a76432b270284141c8dd449d7cab7dc382cb6d7cffd478153c6f2e2
2	01290ec4a22cb3a8094fcb6f23c5ff944eb4b9deb9ea71d8cff92f5618ebd37c4b4ac4a78d596172085ea29761cf62fe1f84131f2c4fe1d9e329cac49c6366d4
3	45edd989a475ebc5b5a887cb973e313d1f3357eccdee1764e805ee2639259a997f2a4e3f7eb1e69fe95d9041c43e550eab119d15dc545bf4c27fc0b17c60bae9
4	1e3d7f5147dc41f4ed5a1ba3291f6286d3c509cfed011dd0fe4d49cb3970d7e001253b036245c7d1eb6cf36776df400294018d903cf33f1e50b966333779f2d7
5	f27b8ef3d67e4970f2d6f8db6672b1169a6d9db66e1f9631a36fd94279364adb2f80d6791e3931eab618e187a35487e363deb0ea7f358a1d423c476ecb7dfb10
6	1b45d6fdf9e5cb6c72954f77428f81f873efc500f015c39131d09a2641914db967c37cd4b979384b429d20251dcf364b6c2659482e6aafb858363cb3da38ff88

16 Abschluss

16.1 Datumsstempel

```
print(datestamp)
```

```
## [1] "2021-04-15"
```

16.2 Datum und Uhrzeit (Anfang)

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

```
## [1] "2021-04-15 01:37:44 CEST"
```

16.3 Datum und Uhrzeit (Ende)

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

```
## [1] "2021-04-15 08:41:20 CEST"
```

16.4 Laufzeit des gesamten Skriptes

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

```
## Time difference of 7.060166 hours
```

16.5 Warnungen

```
print(warnings())
```

```
## NULL
```

17 Parameter für strenge Replikationen

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

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

```
sessionInfo()
```

```
## R version 4.0.4 (2021-02-15)
## Platform: x86_64-redhat-linux-gnu (64-bit)
## Running under: Fedora 33 (Workstation Edition)
##
## Matrix products: default
## BLAS/LAPACK: /usr/lib64/libopenblas-r0.3.12.so
##
## 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] quanteda_2.1.2    readtext_0.80     data.table_1.14.0 scales_1.1.1
##  [5] ggplot2_3.3.3     doParallel_1.0.16 iterators_1.0.13  foreach_1.5.1
##  [9] pdftools_2.3.1    kableExtra_1.3.4  knitr_1.31        rvest_1.0.0
## [13] httr_1.4.2
##
## loaded via a namespace (and not attached):
##  [1] qpdf_1.1           tidyselect_1.1.0   xfun_0.22          purrr_0.3.4
##  [5] lattice_0.20-41    colorspace_2.0-0   vctrs_0.3.6        generics_0.1.0
##  [9] htmltools_0.5.1.1  viridisLite_0.3.0  yaml_2.2.1         utf8_1.2.1
## [13] rlang_0.4.10       pillar_1.5.1       glue_1.4.2         withr_2.4.1
## [17] selectr_0.4-2      lifecycle_1.0.0    stringr_1.4.0      munsell_0.5.0
## [21] gtable_0.3.0       codetools_0.2-18   evaluate_0.14      labeling_0.4.2
## [25] curl_4.3           fansi_0.4.2        highr_0.8          Rcpp_1.0.6
## [29] magick_2.7.1       RcppParallel_5.0.3 webshot_0.5.2      farver_2.1.0
## [33] systemfonts_1.0.1  fastmatch_1.1-0    stopwords_2.2      askpass_1.1
## [37] digest_0.6.27      stringi_1.5.3      dplyr_1.0.5        grid_4.0.4
## [41] tools_4.0.4        magrittr_2.0.1     tibble_3.1.0       crayon_1.4.1
## [45] pkgconfig_2.0.3    Matrix_1.3-2       ellipsis_0.3.1     xml2_1.3.2
## [49] rmarkdown_2.7      svglite_2.0.0      rstudioapi_0.13    R6_2.5.0
## [53] compiler_4.0.4
```

Literaturverzeichnis

Analytics, Revolution, and Steve Weston. 2020. *Iterators: Provides Iterator Construct*. https://github.com/RevolutionAnalytics/iterators.

Benoit, Kenneth, and Adam Obeng. 2020. *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, Jiong Wei Lua, Jouni Kuha, and William Lowe. 2020. *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. 2020. *Pdftools: Text Extraction, Rendering and Converting of Pdf Documents*. https://CRAN.R-project.org/package=pdftools.

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. 2020. *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.