

# DSP\_tfidf\_ds3

Melanie Weissenboeck

2022-10-16

## Laden von Bibliotheken und Daten

```
library("xlsx")
library(word2vec)
library(udpipe)
library(SnowballC)
library(ggplot2)
library(tm)

## Lade nötiges Paket: NLP

##
## Attache Paket: 'NLP'

## Das folgende Objekt ist maskiert 'package:ggplot2':
##
##      annotate

library(wordcloud)

## Lade nötiges Paket: RColorBrewer

library(tidytext)
library(tidyr)
library(e1071)
library(caret)

## Lade nötiges Paket: lattice

library(mlbench)
```

## Vorverarbeiten der Texte

### Erstellen eines Korpus

Am Beginn der Vorverarbeitung wird zum `ds3` ein Wortkorpus erstellt. Dazu wird das `tm`-Package verwendet. Als Parameter wird angegeben, dass es sich um deutsche Texte handelt.

```
# CREATING CORPUS
# Corpus, VCorpus or SimpleCorpus -> SimpleCorpus
corp_ds3 = SimpleCorpus(VectorSource(ds3$ANF_BESCHREIBUNG), control = list(language = "de"))
corp_ds3

## <<SimpleCorpus>>
## Metadata:  corpus specific: 1, document level (indexed): 0
```

```
## Content:  documents: 1446
```

## Entfernen von störenden Zeichen

Es werden alle numerischen Zeichen aus den Texten entfernt. Zur Veranschaulichung wird ein Beispiel vor und nach dem Verarbeitungsschritt ausgegeben.

```
# REMOVING NUMBERS
corp_ds3 = tm_map(corp_ds3, removeNumbers)
```

Nach den numerischen Zeichen werden auch die Satzzeichen entfernt.

```
# REMOVE PUNCTUATION
corp_ds3 = tm_map(corp_ds3, removePunctuation)
```

Leerzeichen und Abstände werden ebenfalls aus dem Datenset entfernt.

```
# STRIPWHITESPACE
corp_ds3 = tm_map(corp_ds3, stripWhitespace)
```

```
# REMOVING STOPWORDS
corp_ds3 <- tm_map(corp_ds3, removeWords, stopwords("german"))
```

```
# STEMMING
corp_ds3 <- tm_map(corp_ds3, stemDocument)
#writeLines(as.character(corp_ds3[[3]]))
```

## Erstellen von Matrizen

Mit den vorverarbeiteten Texten kann nun eine Matrix erstellt werden. Zunächst wird eine **term-document**-Matrix erstellt und danach eine **document-term**-Matrix. Bei letzterer muss angegeben werden, welche Gewichtungsfunktion verwendet wird, in diesem Fall wird anhand **tf-idf** gewichtet.

```
# term document matrix
tdm_ds3 <- TermDocumentMatrix(corp_ds3, control = list(removeSparseTerms = TRUE,
                                                         removePunctuation = TRUE,
                                                         removeNumbers = TRUE,
                                                         stopwords = TRUE,
                                                         stemming = TRUE))
```

```
# document term matrix
dtm_ds3 <- DocumentTermMatrix(corp_ds3, control = list(weighting = function(x)
                                                         weightTfIdf(x, normalize = FALSE), stopwords = TRUE))
```

```
tdm_ds3
```

```
## <<TermDocumentMatrix (terms: 1598, documents: 1446)>>
## Non-/sparse entries: 48924/2261784
## Sparsity           : 98%
## Maximal term length: 94
## Weighting           : term frequency (tf)
dtm_ds3
```

```
## <<DocumentTermMatrix (documents: 1446, terms: 1898)>>
## Non-/sparse entries: 51507/2693001
## Sparsity           : 98%
## Maximal term length: 94
## Weighting           : term frequency - inverse document frequency (tf-idf)
```

Die document-term-Matrix ist nur sehr dünn besetzt. Um den Speicherbedarf zu reduzieren, werden einige Terme weggelassen und somit die Dimension reduziert.

```
# A term-document matrix where those terms from x are removed
# which have at least a sparse percentage of empty
# (i.e., terms occurring 0 times in a document) elements.
# Resulting matrix contains only terms with a sparse factor less than sparse
dtm_ds3 <- removeSparseTerms(dtm_ds3, 0.85)
dtm_ds3
```

```
## <<DocumentTermMatrix (documents: 1446, terms: 31)>>
## Non-/sparse entries: 11202/33624
## Sparsity          : 75%
## Maximal term length: 13
## Weighting          : term frequency - inverse document frequency (tf-idf)
inspect(tdm_ds3)
```

```
## <<TermDocumentMatrix (terms: 1598, documents: 1446)>>
## Non-/sparse entries: 48924/2261784
## Sparsity          : 98%
## Maximal term length: 94
## Weighting          : term frequency (tf)
## Sample            :
##
##      Docs
## Terms  450 451 452 453 454 455 456 457 458 459
## anforder      6  6  6  6  6  6  6  6  6  6
## angezeigt     2  2  2  2  2  2  2  2  2  2
## ausgewählt    0  0  0  0  0  0  0  0  0  0
## button        0  0  0  0  0  0  0  0  0  0
## detailregist  0  0  0  0  0  0  0  0  0  0
## komponent     8  8  8  8  8  8  8  8  8  8
## navig         1  1  1  1  1  1  1  1  1  1
## objekt        1  1  1  1  1  1  1  1  1  1
## testfal       7  7  7  7  7  7  7  7  7  7
## testfall      16 16 16 16 16 16 16 16 16 16
```

```
inspect(dtm_ds3)
```

```
## <<DocumentTermMatrix (documents: 1446, terms: 31)>>
## Non-/sparse entries: 11202/33624
## Sparsity          : 75%
## Maximal term length: 13
## Weighting          : term frequency - inverse document frequency (tf-idf)
## Sample            :
##      Terms
## Docs  befehl button detailregist list löschen navig objekt testfal
## 450 11.13194      0              0 6.707271      0 1.182702 1.677673 13.105
## 451 11.13194      0              0 6.707271      0 1.182702 1.677673 13.105
## 452 11.13194      0              0 6.707271      0 1.182702 1.677673 13.105
## 453 11.13194      0              0 6.707271      0 1.182702 1.677673 13.105
## 454 11.13194      0              0 6.707271      0 1.182702 1.677673 13.105
## 455 11.13194      0              0 6.707271      0 1.182702 1.677673 13.105
## 456 11.13194      0              0 6.707271      0 1.182702 1.677673 13.105
## 457 11.13194      0              0 6.707271      0 1.182702 1.677673 13.105
## 458 11.13194      0              0 6.707271      0 1.182702 1.677673 13.105
```

```
## 459 11.13194 0 0 6.707271 0 1.182702 1.677673 13.105
## Terms
## Docs testfäll testset
## 450 33.74886 23.88565
## 451 33.74886 23.88565
## 452 33.74886 23.88565
## 453 33.74886 23.88565
## 454 33.74886 23.88565
## 455 33.74886 23.88565
## 456 33.74886 23.88565
## 457 33.74886 23.88565
## 458 33.74886 23.88565
## 459 33.74886 23.88565
```

## Darstellen einer Wortwolke

Zur Veranschaulichung der reduzierten document-term-Matrix wird eine Wortwolke erstellt, in der die wichtigsten Begriffe des Datensets gezeigt werden.

```
freq = data.frame(sort(colSums(as.matrix(dtm_ds3)), decreasing=TRUE))
wordcloud(rownames(freq), freq[,1], max.words=100, colors=brewer.pal(5, "Dark2"))
```



## Konvertieren der Dokument Term Matrix zu Dataframe

Um die konstruierte Matrix mit den anderen Prädiktoren zusammengefügt werden kann, muss diese zunächst in einen Dataframe transformiert werden.

```
tmp_df_ds3 <- tidy(dtm_ds3)
head(tmp_df_ds3)
```

```
## # A tibble: 6 x 3
##   document term      count
##   <chr>    <chr>    <dbl>
## 1 1      angezeigt  1.26
## 2 1      befehl    2.23
## 3 1      komponenten 2.52
## 4 1      navig     1.18
## 5 2      auswahl    2.07
## 6 2      beim      2.42
```

```
tmp_df_ds3 <- tmp_df_ds3 |> pivot_wider(names_from = term, values_from = count,
                                         names_repair = "unique", values_fill = 0)
```

```
colnames(tmp_df_ds3)[1] <- "doc_id"
tmp_df_ds3$doc_id <- as.integer(tmp_df_ds3$doc_id)
tmp_df_ds3$row_sum <- rowSums(tmp_df_ds3)
```

```
rbind(tmp_df_ds3, sum(tmp_df_ds3[, 1:length(tmp_df_ds3)]))
```

```
## # A tibble: 1,421 x 33
##   doc_id angezeigt befehl komponenten navig auswahl beim button detailregist
##   <dbl>    <dbl>  <dbl>      <dbl> <dbl>    <dbl> <dbl>  <dbl>      <dbl>
## 1     1      1.26  2.23      2.52  1.18     0      0      0          0
## 2     2      0     0      0      2.37    2.07    2.42  1.51      1.41
## 3     3      1.26  2.23      0      2.37    4.14    2.42  3.02      1.41
## 4     4      0     0      0      2.37    2.07    2.42  1.51      1.41
## 5     5      0     4.45     0      3.55    2.07     0      0          2.82
## 6     6      0     0      0      5.91     0      0      0          0
## 7     7      0     0      0      5.91     0      0      0          0
## 8     8      0     0      0      0      2.07     0      0          1.41
## 9     9      0     0      0      0      2.07     0      0          1.41
## 10    10      0     0      0      0      2.07     0      0          1.41
## # ... with 1,411 more rows, and 24 more variables: folgenden <dbl>, gibt <dbl>,
## #   innerhalb <dbl>, klick <dbl>, listenansicht <dbl>, testfal <dbl>,
## #   testfäll <dbl>, toolbar <dbl>, anforderungen <dbl>, ausgewählt <dbl>,
## #   ausgewählten <dbl>, via <dbl>, geändert <dbl>, typ <dbl>, list <dbl>,
## #   löschen <dbl>, wert <dbl>, objekt <dbl>, sieh <dbl>, testset <dbl>,
## #   anforderung <dbl>, kapitel <dbl>, testfällen <dbl>, row_sum <dbl>
```

## Zusammenführen mit anderen Prädiktoren

Im Folgenden werden alle Prädiktoren in einem Dataframe zusammengefasst. Dieser stellt die Ausgangslage für die Klassifikation dar.

```
final_ds3 <- cbind(ds3$ANF_RISIKO, ds3$ANF_FEHLERWAHRSCHEINLICHKEIT,
                  ds3$ANF_FEHLERKOSTEN, ds3$TF_ABDECKUNG, ds3$AKT_RES_RELEASE,
                  ds3$AKT_RES_STATUS, tmp_df_ds3[1:1446, 2:33])
```

```
final_ds3 <- final_ds3[1:1420, 1:37]
final_ds3 <- final_ds3[1:1420, -c(2,3)]
```

```
df <- final_ds3
names(df)[names(df)=="ds3$ANF_RISIKO"] <- "ANF_RISIKO"
```

```
names(df)[names(df)=="ds3$TF_ABDECKUNG"] <- "TF_ABDECKUNG"
names(df)[names(df)=="ds3$AKT_RES_RELEASE"] <- "AKT_RES_RELEASE"
names(df)[names(df)=="ds3$AKT_RES_STATUS"] <- "AKT_RES_STATUS"
```

```
summary(df)
```

```
##   ANF_RISIKO      TF_ABDECKUNG      AKT_RES_RELEASE      AKT_RES_STATUS
## Length:1420      Min.   : 0.00      Length:1420      Length:1420
## Class :character  1st Qu.: 6.67      Class :character  Class :character
## Mode  :character  Median : 14.29      Mode  :character  Mode  :character
##                      Mean   : 22.75
##                      3rd Qu.: 33.30
##                      Max.   :100.00
##   angezeigt      befehl      komponenten      navig
## Min.   : 0.0000      Min.   : 0.0000      Min.   : 0.0000      Min.   : 0.000
## 1st Qu.: 0.0000      1st Qu.: 0.0000      1st Qu.: 0.0000      1st Qu.: 0.000
## Median : 0.0000      Median : 0.0000      Median : 0.0000      Median : 0.000
## Mean   : 0.7224      Mean   : 0.9062      Mean   : 0.7757      Mean   : 1.169
## 3rd Qu.: 1.2571      3rd Qu.: 0.0000      3rd Qu.: 0.0000      3rd Qu.: 2.365
## Max.   :10.0565      Max.   :13.3583      Max.   :20.1646      Max.   :17.741
##   auswahl      beim      button      detailregist
## Min.   : 0.0000      Min.   :0.0000      Min.   :0.000      Min.   : 0.0000
## 1st Qu.: 0.0000      1st Qu.:0.0000      1st Qu.:0.000      1st Qu.: 0.0000
## Median : 0.0000      Median :0.0000      Median :0.000      Median : 0.0000
## Mean   : 0.8812      Mean   :0.6411      Mean   :1.167      Mean   : 0.9952
## 3rd Qu.: 0.0000      3rd Qu.:0.0000      3rd Qu.:1.509      3rd Qu.: 1.4104
## Max.   :12.4295      Max.   :9.6841      Max.   :7.546      Max.   :28.2078
##   folgenden      gibt      innerhalb      klick
## Min.   :0.000      Min.   :0.0000      Min.   : 0.0000      Min.   :0.000
## 1st Qu.:0.000      1st Qu.:0.0000      1st Qu.: 0.0000      1st Qu.:0.000
## Median :0.000      Median :0.0000      Median : 0.0000      Median :0.000
## Mean   :0.637      Mean   :0.7447      Mean   : 0.7149      Mean   :0.828
## 3rd Qu.:1.320      3rd Qu.:1.3056      3rd Qu.: 0.0000      3rd Qu.:0.000
## Max.   :2.641      Max.   :5.2222      Max.   :11.9991      Max.   :6.265
## listenansicht      testfal      testfäll      toolbar
## Min.   : 0.0000      Min.   : 0.000      Min.   : 0.000      Min.   :0.0000
## 1st Qu.: 0.0000      1st Qu.: 0.000      1st Qu.: 0.000      1st Qu.:0.0000
## Median : 0.0000      Median : 0.000      Median : 0.000      Median :0.0000
## Mean   : 0.7888      Mean   : 1.231      Mean   : 1.412      Mean   :0.7228
## 3rd Qu.: 1.9017      3rd Qu.: 1.872      3rd Qu.: 0.000      3rd Qu.:1.7665
## Max.   :17.1150      Max.   :20.594      Max.   :33.749      Max.   :5.2996
## anforderungen      ausgewählt      ausgewählten      via
## Min.   : 0.0000      Min.   :0.0000      Min.   : 0.0000      Min.   :0.0000
## 1st Qu.: 0.0000      1st Qu.:0.0000      1st Qu.: 0.0000      1st Qu.:0.0000
## Median : 0.0000      Median :0.0000      Median : 0.0000      Median :0.0000
## Mean   : 0.8277      Mean   :0.6857      Mean   : 0.8673      Mean   :0.5189
## 3rd Qu.: 0.0000      3rd Qu.:0.0000      3rd Qu.: 0.0000      3rd Qu.:0.0000
## Max.   :14.7536      Max.   :9.4537      Max.   :24.5417      Max.   :7.5962
##   geändert      typ      list      löschen
## Min.   : 0.0000      Min.   : 0.0000      Min.   : 0.0000      Min.   : 0.0000
## 1st Qu.: 0.0000      1st Qu.: 0.0000      1st Qu.: 0.0000      1st Qu.: 0.0000
## Median : 0.0000      Median : 0.0000      Median : 0.0000      Median : 0.0000
## Mean   : 0.6849      Mean   : 0.7847      Mean   : 0.9903      Mean   : 0.9916
## 3rd Qu.: 0.0000      3rd Qu.: 0.0000      3rd Qu.: 0.0000      3rd Qu.: 0.0000
```

```
## Max. :13.5825 Max. :13.6721 Max. :20.1218 Max. :14.7990
## wert objekt sieh testset
## Min. : 0.0000 Min. : 0.0000 Min. :0.0000 Min. : 0.000
## 1st Qu.: 0.0000 1st Qu.: 0.0000 1st Qu.:0.0000 1st Qu.: 0.000
## Median : 0.0000 Median : 0.0000 Median :0.0000 Median : 0.000
## Mean : 0.7655 Mean : 0.9924 Mean :0.6763 Mean : 1.336
## 3rd Qu.: 0.0000 3rd Qu.: 1.6777 3rd Qu.:1.9017 3rd Qu.: 0.000
## Max. :12.1319 Max. :11.7437 Max. :7.6066 Max. :23.886
## anforderung kapitel testfällen
## Min. : 0.0000 Min. : 0.0000 Min. : 0.0000
## 1st Qu.: 0.0000 1st Qu.: 0.0000 1st Qu.: 0.0000
## Median : 0.0000 Median : 0.0000 Median : 0.0000
## Mean : 0.8198 Mean : 0.5231 Mean : 0.7623
## 3rd Qu.: 0.0000 3rd Qu.: 0.0000 3rd Qu.: 0.0000
## Max. :28.9703 Max. :10.3879 Max. :14.1498
```

## Normalisieren numerischer Spalten

Mittels min-max-Normalisierung werden die numerischen Spalten auf eine gemeinsame Skalierung gebracht. Zur besseren Übersicht wird am Ende nochmal eine Zusammenfassung ausgegeben.

```
set.seed(1234)
# definiere normalisierungsfunktion
min_max_norm <- function(x) {
  (x - min(x)) / (max(x) - min(x))
}
# alle spalten normalisieren
df[, 5:35] <- as.data.frame(lapply(df[, 5:35], min_max_norm))
df[2] <- as.data.frame(lapply(df[2], min_max_norm))
df$ANF_RISIKO <- as.factor(df$ANF_RISIKO)
df$AKT_RES_STATUS <- as.factor(df$AKT_RES_STATUS)
df$AKT_RES_RELEASE <- as.factor(df$AKT_RES_RELEASE)
summary(df)
```

```
## ANF_RISIKO TF_ABDECKUNG AKT_RES_RELEASE AKT_RES_STATUS angezeigt
## gering:238 Min. :0.0000 21x : 48 FAILED: 45 Min. :0.00000
## hoch :538 1st Qu.:0.0667 22.10 :1070 OK :1373 1st Qu.:0.00000
## mittel:644 Median :0.1429 22.20 : 3 OPEN : 2 Median :0.00000
## Mean :0.2275 22.30 : 296 Mean :0.07183
## 3rd Qu.:0.3330 OLDERT21: 3 3rd Qu.:0.12500
## Max. :1.0000 Max. :1.00000
## befehl komponenten navig auswahl
## Min. :0.00000 Min. :0.00000 Min. :0.00000 Min. :0.00000
## 1st Qu.:0.00000 1st Qu.:0.00000 1st Qu.:0.00000 1st Qu.:0.00000
## Median :0.00000 Median :0.00000 Median :0.00000 Median :0.00000
## Mean :0.06784 Mean :0.03847 Mean :0.06587 Mean :0.07089
## 3rd Qu.:0.00000 3rd Qu.:0.00000 3rd Qu.:0.13333 3rd Qu.:0.00000
## Max. :1.00000 Max. :1.00000 Max. :1.00000 Max. :1.00000
## beim button detailregist folgenden
## Min. :0.0000 Min. :0.0000 Min. :0.00000 Min. :0.0000
## 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.00000 1st Qu.:0.0000
## Median :0.0000 Median :0.0000 Median :0.00000 Median :0.0000
## Mean :0.0662 Mean :0.1546 Mean :0.03528 Mean :0.2412
## 3rd Qu.:0.0000 3rd Qu.:0.2000 3rd Qu.:0.05000 3rd Qu.:0.5000
## Max. :1.0000 Max. :1.0000 Max. :1.00000 Max. :1.0000
```

##	gibt	innerhalb	klick	listenansicht
##	Min. :0.0000	Min. :0.00000	Min. :0.0000	Min. :0.00000
##	1st Qu.:0.0000	1st Qu.:0.00000	1st Qu.:0.0000	1st Qu.:0.00000
##	Median :0.0000	Median :0.00000	Median :0.0000	Median :0.00000
##	Mean :0.1426	Mean :0.05958	Mean :0.1322	Mean :0.04609
##	3rd Qu.:0.2500	3rd Qu.:0.00000	3rd Qu.:0.0000	3rd Qu.:0.11111
##	Max. :1.0000	Max. :1.00000	Max. :1.0000	Max. :1.00000
##	testfal	testfäll	toolbar	anforderungen
##	Min. :0.00000	Min. :0.00000	Min. :0.0000	Min. :0.0000
##	1st Qu.:0.00000	1st Qu.:0.00000	1st Qu.:0.0000	1st Qu.:0.0000
##	Median :0.00000	Median :0.00000	Median :0.0000	Median :0.0000
##	Mean :0.05980	Mean :0.04183	Mean :0.1364	Mean :0.0561
##	3rd Qu.:0.09091	3rd Qu.:0.00000	3rd Qu.:0.3333	3rd Qu.:0.0000
##	Max. :1.00000	Max. :1.00000	Max. :1.0000	Max. :1.0000
##	ausgewählt	ausgewählten	via	geändert
##	Min. :0.00000	Min. :0.00000	Min. :0.00000	Min. :0.00000
##	1st Qu.:0.00000	1st Qu.:0.00000	1st Qu.:0.00000	1st Qu.:0.00000
##	Median :0.00000	Median :0.00000	Median :0.00000	Median :0.00000
##	Mean :0.07254	Mean :0.03534	Mean :0.06831	Mean :0.05042
##	3rd Qu.:0.00000	3rd Qu.:0.00000	3rd Qu.:0.00000	3rd Qu.:0.00000
##	Max. :1.00000	Max. :1.00000	Max. :1.00000	Max. :1.00000
##	typ	list	löschen	wert
##	Min. :0.00000	Min. :0.00000	Min. :0.000	Min. :0.0000
##	1st Qu.:0.00000	1st Qu.:0.00000	1st Qu.:0.000	1st Qu.:0.0000
##	Median :0.00000	Median :0.00000	Median :0.000	Median :0.0000
##	Mean :0.05739	Mean :0.04922	Mean :0.067	Mean :0.0631
##	3rd Qu.:0.00000	3rd Qu.:0.00000	3rd Qu.:0.000	3rd Qu.:0.0000
##	Max. :1.00000	Max. :1.00000	Max. :1.000	Max. :1.0000
##	objekt	sieh	testset	anforderung
##	Min. :0.00000	Min. :0.00000	Min. :0.00000	Min. :0.0000
##	1st Qu.:0.00000	1st Qu.:0.00000	1st Qu.:0.00000	1st Qu.:0.0000
##	Median :0.00000	Median :0.00000	Median :0.00000	Median :0.0000
##	Mean :0.08451	Mean :0.08891	Mean :0.05595	Mean :0.0283
##	3rd Qu.:0.14286	3rd Qu.:0.25000	3rd Qu.:0.00000	3rd Qu.:0.0000
##	Max. :1.00000	Max. :1.00000	Max. :1.00000	Max. :1.0000
##	kapitel	testfällen		
##	Min. :0.00000	Min. :0.00000		
##	1st Qu.:0.00000	1st Qu.:0.00000		
##	Median :0.00000	Median :0.00000		
##	Mean :0.05035	Mean :0.05387		
##	3rd Qu.:0.00000	3rd Qu.:0.00000		
##	Max. :1.00000	Max. :1.00000		

## Klassifikation

### Erstellen von Train- / Test-Split

Die vorliegenden Daten werden in Trainings- und Testdaten aufgeteilt im Verhältnis 80:20.

```
# partition erstellen
part <- createDataPartition(df$ANF_RISIKO, times = 1, p = 0.80)
# extract training set
X_train <- df[part$Resample1, ]
# extract testing set
```



```
X_test <- df[-part$Resample1, ]
# extract target
y_train <- df[part$Resample1, 1]
y_test <- df[-part$Resample1, 1]
```

## NaiveBayes Klassifikation

Ein Naive-Bayes Klassifikator wird erstellt und mit den Trainingsdaten trainiert. Anhand der Testdaten wird das Modell evaluiert. Die Ergebnisse werden in einer Confusionmatrix angegeben.

```
model_nb = naiveBayes(ANF_RISIKO ~ ., data = X_train)
```

```
pred_nb <- predict(model_nb, X_test)
mat.nb <- confusionMatrix(pred_nb, X_test$ANF_RISIKO, mode = "prec_recall")
mat.nb
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction gering hoch mittel
##      gering      22   22    28
##      hoch       8   30    25
##      mittel     17   55    75
##
## Overall Statistics
##
##              Accuracy : 0.4504
##              95% CI : (0.3913, 0.5105)
##      No Information Rate : 0.4539
##      P-Value [Acc > NIR] : 0.5704727
##
##              Kappa : 0.1359
##
##      McNemar's Test P-Value : 0.0001355
##
## Statistics by Class:
##
##              Class: gering Class: hoch Class: mittel
## Precision              0.30556      0.4762      0.5102
## Recall                  0.46809      0.2804      0.5859
## F1                      0.36975      0.3529      0.5455
## Prevalence              0.16667      0.3794      0.4539
## Detection Rate          0.07801      0.1064      0.2660
## Detection Prevalence    0.25532      0.2234      0.5213
## Balanced Accuracy       0.62766      0.5459      0.5592
```

## KNN Klassifikation

Analog zum Naive-Bayes Klassifikator wird auch ein KNN Modell trainiert. Auch hier wird das Ergebnis anhand einer Confusionmatrix gezeigt.

```
model_knn <- train(ANF_RISIKO ~ ., data = X_train, "knn",
  trControl = trainControl(method = "cv", number = 5))
model_knn
```

```

## k-Nearest Neighbors
##
## 1138 samples
## 34 predictor
## 3 classes: 'gering', 'hoch', 'mittel'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 911, 910, 910, 910, 911
## Resampling results across tuning parameters:
##
## k Accuracy Kappa
## 5 0.5720458 0.2971401
## 7 0.5448064 0.2454533
## 9 0.5351611 0.2275982
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 5.

```

```

pred_knn <- predict(model_knn, X_test, type = "raw")
mat.knn <- confusionMatrix(pred_knn, X_test$ANF_RISIKO, mode = "prec_recall")
mat.knn

```

```

## Confusion Matrix and Statistics
##
##           Reference
## Prediction gering hoch mittel
##      gering      24    5    11
##      hoch       11   52   35
##      mittel     12   50   82
##
## Overall Statistics
##
##              Accuracy : 0.5603
##              95% CI : (0.5002, 0.6191)
##      No Information Rate : 0.4539
##      P-Value [Acc > NIR] : 0.0002163
##
##              Kappa : 0.2824
##
## Mcnemar's Test P-Value : 0.1762035
##
## Statistics by Class:
##
##              Class: gering Class: hoch Class: mittel
## Precision          0.60000      0.5306      0.5694
## Recall              0.51064      0.4860      0.6406
## F1                  0.55172      0.5073      0.6029
## Prevalence          0.16667      0.3794      0.4539
## Detection Rate       0.08511      0.1844      0.2908
## Detection Prevalence 0.14184      0.3475      0.5106
## Balanced Accuracy    0.72128      0.6116      0.6190

```