Regulatory Documents via LDA - Documentation

Sebastian Knigge

18 8 2019

1 Setup

Following libraries are used in the code:

```
library(dplyr)
library(tidytext)
library(pdftools)
library(tidyr)
library(stringr)
library(tidytext)
library(udpipe)
library(topicmodels)
library(ggplot2)
library(wordcloud)
library(tm)
library(SnowballC)
library(RColorBrewer)
library(RCurl)
library(XML)
library(openxlsx)
library(keras)
```

2 Import data

The Documents had to be preprocessed. For the documents wp2.5 all list of contents had to be deleted, because they were the same in each of these documents. No more adjustments had to be made.

In this code reulatory documents are red in and processed via LDA. This first part focusses on reading in the pdf documents.

```
# getting the right order
setwd('..')
documents <- read.xlsx("Docs_classes.xlsx")[,2]
classes <- read.xlsx("Docs_classes.xlsx")[,c(1,3)]
documents <- paste0(documents,".pdf")
documents %>% as.data.frame() %>% stargazer(summary=FALSE,
                                            header = FALSE,
                                            title="Document Titles")
```

```
# getting the right directory
library(here)
setwd("../")
path <- getwd() %>%
  file.path("TextDocs")
setwd(path)
```

Table 1: Document Titles

	.
1	admin-wp1.1_analysis_legal_institutional_environment_final.pdf
2	admin-wp1.2_good_practices_final.pdf
3	admin-wp2.1_estimation_methods1.pdf
4	admin-wp2.2_estimation_methods2.pdf
5	admin-wp2.3-estimation_methods3.pdf
6	admin-wp2.4_examples.pdf
7	admin-wp2.5_alignment.pdf
8	admin-wp2.5_editing.pdf
9	admin-wp2.5_greg.pdf
10	admin-wp2.5_imputation.pdf
11	admin-wp2.5_macro_integration.pdf
12	admin-wp2.5_macro_integration.pdf
13	admin-wp2.6_good_practices.pdf
14	admin-wp2.6_guidelines.pdf
15	admin-wp3.1_quality1.pdf
16	admin-wp3.2_quality2.pdf
17	admin-wp3.3_quality.pdf
18	admin-wp3.4_quality.pdf
19	admin-wp3.5_quality_measures.pdf
20	admin-wp3_coherence.pdf
21	admin-wp3_growth_rates.pdf
22	admin-wp3_suitability1.pdf
23	admin-wp3_suitability2.pdf
24	admin-wp3_suitability3.pdf
25	admin-wp3_uncertainty.pdf
26	admin-wp5_frames.pdf
27	admin-wp5_frames_examples.pdf
28	admin-wp5_frames_recommendation.pdf

Following functions are used to set up and analyze the pdfs. When cleaning up data, we have to take into account certain circumstances of the regulatory documents. For example, there are many formulas and technical abbreviations in the documents. Every variable, every estimator, and every index is included as a single word in the bag of words. These terms sometimes have a big influence on the documents, because they are very specific for individual documents and occur quite often. To avoid this, we exclude all mixed words with characters and numeric values, as well as all terms with special characters (e.g. Greek letters).

```
read_pdf_clean <- function(document){
  # This function loads the document given per name
  # and excludes the stop words inclusive numbers
  pdf1 <- pdf_text(file.path(path, document)) %>%
    strsplit(split = "\n") %>%
    do.call("c",.) %>%
    as_tibble() %>%
    unnest_tokens(word,value) %>%
    # also exclude all words including numbers and special characters
    filter(grepl("^[a-z]+$", word))
  # load stopword library
  data(stop_words)
  # add own words to stop word library - here the numbers from 1 to 10
  new_stop_words <- tibble(word=as.character(0:9),
                           lexicon=rep("own",10)) %>%
                           bind_rows(stop_words)
  # stop words are excluded via anti_join
  pdf1 %>%
    anti_join(new_stop_words)
}

plot_most_freq_words <- function(pdf, n=7){
  # plots a bar plot via ggplot
  pdf %>% count(word) %>% arrange(desc(n)) %>% head(n) %>%
    ggplot(aes(x=word,y=n)) +
    geom_bar(stat="identity")+
    # no labels for x and y scale
    theme(axis.title.y=element_blank(),
          axis.title.x=element_blank())
}
```

Now we can read in all documents in a for loop:

```
setwd(path)
# inital set up for the corpus
pdf1 <- read_pdf_clean(documents[1])
corpus <- tibble(document=1, word=pdf1$word)
# adding the documents iteratively
for (i in 2:length(documents)){
  pdf_i <- read_pdf_clean(documents[i])
  corpus <- tibble(document=i, word=pdf_i$word) %>% bind_rows(corpus,.)
}
```

3 LDA

The LDA model is applied. First the document term matrix has to be set up.

```
dtm <- corpus %>% count(document, word, sort = TRUE) %>%
  select(doc_id=document, term=word, freq=n) %>%
  document_term_matrix()
c(N,M) %<-% dim(dtm)
```

We use term frequency 2 embedding because in the example with the Gutenberg Data it turned out to be advantageous with regard to the “predictive power” of the LDA algorithm.

```
dtm_tf2 <- dtm %>%
    # reduce by low frequencies
    dtm_remove_lowfreq(minfreq = 2)
ncol(dtm_tf2 )
```

```
## [1] 5717
```

Using the function LDA sets up the model and prediction/evaluation is done via predict(). But first of all it shall be verified whether the Predict function actually delivers the same classification as the export of the gamma matrix directly from the LDA model. Therefore both gamma matrices of the single functions are compared. Table 2 displays the output of the gamma matrix received by the predict() function and Table 3 displays the gamma matrix returned by the LDA model itself.

```
set.seed(123)
documents_lda <- LDA(dtm_tf2, method = "Gibbs",
                    k = 7, control = list(seed = 1234))
```

```
prediction5 <- predict(documents_lda, newdata=dtm_tf2, type="topic")

prediction5 <- merge(prediction5, classes, by.x="doc_id", by.y="No")

prediction5 %>%
  select(doc_id,topic_001,topic_002,topic_003,topic_004,topic_005,
         topic_006, topic_007) %>%
  mutate_each(funs(as.numeric),
              doc_id,topic_001,topic_002,topic_003,topic_004,
              topic_005, topic_006, topic_007) %>%
  arrange(desc(-doc_id)) %>%
  round(2) %>%
  stargazer(summary=F, rownames = F, header = F,
            title="Gamma matrix for predict function", label="predict")
```

```
ext_gamma_matrix <- function(model=documents_lda){
  # get gamma matrix for chapter probabilities
  chapters_gamma <- tidy(model, matrix = "gamma")
  # get matrix with probabilities for each topic per chapter
  spreaded_gamma <- chapters_gamma %>% spread(topic, gamma)
  spreaded_gamma %>%
    mutate_each(funs(as.numeric), document,1,2,3,4,5,6,7) %>%
  arrange(desc(-document))
}

ext_gamma_matrix(documents_lda) %>%
  round(2) %>%
  stargazer(summary=F, rownames = F, header=F,
            title="Gamma matrix extracted from model",
            label="extract")
```

Table 2: Gamma matrix for predict function

doc_id	topic_001	topic_002	topic_003	topic_004	topic_005	topic_006	topic_007
1	0	0.010	0	0.960	0	0.020	0.010
2	0	0.050	0	0.830	0	0.090	0.020
3	0.010	0.030	0.200	0.020	0.160	0.520	0.060
4	0.030	0.020	0.270	0.020	0.030	0.620	0.030
5	0.050	0.020	0.770	0.010	0.040	0.110	0.010
6	0.100	0.010	0.640	0.020	0.120	0.110	0.010
7	0.040	0.010	0.840	0.010	0.040	0.030	0.030
8	0.030	0.030	0.790	0.020	0.020	0.090	0.020
9	0.030	0.030	0.740	0.010	0.070	0.070	0.040
10	0.030	0	0.920	0	0.010	0.020	0.010
11	0.070	0.010	0.810	0.010	0.060	0.030	0.010
12	0.060	0.020	0.820	0.010	0.050	0.040	0.010
13	0	0.040	0.020	0.120	0.010	0.770	0.040
14	0.020	0.010	0.230	0.010	0.020	0.690	0.010
15	0.010	0.690	0.020	0.020	0.020	0.230	0.010
16	0.060	0.030	0.020	0	0.840	0.050	0
17	0.050	0.060	0.030	0.010	0.090	0.040	0.730
18	0.050	0.030	0.010	0	0.850	0.050	0.010
19	0.070	0.140	0.100	0	0.610	0.040	0.050
20	0.050	0.810	0.040	0.010	0.010	0.060	0.020
21	0.930	0.010	0	0.010	0.040	0.010	0.010
22	0.050	0.010	0.010	0.010	0.900	0.020	0.010
23	0.090	0.020	0.010	0.010	0.820	0.020	0.040
24	0.060	0.010	0.020	0.010	0.860	0.030	0.010
25	0.880	0.010	0.060	0	0.030	0.010	0
26	0.010	0.090	0.010	0.020	0.010	0.170	0.670
27	0	0.020	0.010	0.190	0	0.090	0.690
28	0.010	0.630	0.010	0.070	0.010	0.110	0.170

Table 3: Gamma matrix extracted from model

document	1	2	3	4	5	6	7
1	0	0.01	0	0.95	0	0.02	0.01
2	0	0.06	0	0.81	0.01	0.1	0.02
3	0.02	0.03	0.2	0.02	0.16	0.52	0.06
4	0.02	0.02	0.27	0.01	0.02	0.6	0.05
5	0.04	0.03	0.76	0.01	0.04	0.12	0.01
6	0.1	0	0.62	0.02	0.12	0.11	0.02
7	0.05	0.02	0.83	0	0.03	0.03	0.04
8	0.03	0.04	0.79	0.02	0.02	0.08	0.03
9	0.03	0.03	0.72	0.01	0.1	0.04	0.05
10	0.03	0.01	0.91	0	0.02	0.03	0.01
11	0.06	0.01	0.83	0.01	0.04	0.04	0.01
12	0.06	0.01	0.81	0.01	0.06	0.04	0.01
13	0	0.05	0.02	0.14	0.01	0.74	0.04
14	0.02	0.02	0.24	0.01	0.02	0.67	0.01
15	0.01	0.67	0.02	0.03	0.04	0.22	0.01
16	0.05	0.03	0.02	0	0.83	0.05	0.01
17	0.05	0.06	0.03	0.01	0.09	0.04	0.71
18	0.06	0.03	0.02	0	0.84	0.05	0.01
19	0.07	0.14	0.1	0.01	0.59	0.04	0.06
20	0.05	0.79	0.04	0.01	0.02	0.06	0.02
21	0.91	0.01	0.01	0.01	0.04	0.01	0.01
22	0.07	0.01	0.01	0.01	0.87	0.02	0.01
23	0.1	0.03	0.01	0.01	0.79	0.02	0.05
24	0.05	0.01	0.03	0.01	0.85	0.04	0.02
25	0.86	0.01	0.05	0.01	0.04	0.01	0.01
26	0.01	0.09	0.02	0.03	0.01	0.17	0.67
27	0.01	0.02	0.01	0.19	0.01	0.1	0.67
28	0.01	0.61	0.01	0.09	0.01	0.09	0.18

The tables below summarize which document refers to which topic, according to the LDA model.

Table 4: Documents for Topic 1

Topic	doc_id	Group
1	21	6
1	25	6

Table 5: Documents for Topic 2

Topic	doc_id	Group
2	15	5
2	20	5
2	28	7

Table 6: Documents for Topic 3

Topic	doc_id	Group
3	10	4
3	11	4
3	12	4
3	5	2
3	6	3
3	7	4
3	8	4
3	9	4

Table 7: Documents for Topic 4

Topic	doc_id	Group
4	1	1
4	2	1

Table 8: Documents for Topic 5

Topic	doc_id	Group
5	16	5
5	18	5
5	19	5
5	22	6
5	23	6
5	24	6

Table 9: Documents for Topic 6

Topic	doc_id	Group
6	13	3
6	14	4
6	3	2
6	4	2

Table 10: Documents for Topic 7

Topic	doc_id	Group
7	17	5
7	26	7
7	27	7

4 Wordclouds

To check what topics tackle which context, we produce wordclouds using the TFIDF and the TF itself.

```
plot_wordcloud <- function(corpus, selection="ALL",
                           max.words=50, i, freq="tfidf",
                           scale=c(3,0.2)){
  # setting up a tibble which returns tfidf and tf and frequency for
  # the whole corpus
  tfidf <- corpus %>% count(document, word, sort = TRUE) %>%
    bind_tf_idf(word, document, n)
  # include all documents for selection if selection="ALL"
  if (all(selection=="ALL")) {
    selection <- corpus %>%
      select(document) %>%
      unique() %>%
      unlist() %>%
      sort()}
  # filter for all selected documents
  # use either ft or tfidf
  if (freq=="tfidf"){
    dtm_selected <- tfidf %>% filter(document%in%selection) %>%
      select(word, tf_idf) %>% count(word, wt=tf_idf, sort=TRUE)
  } else {
    dtm_selected <- tfidf %>% filter(document%in%selection) %>%
      select(word, tf) %>% count(word, wt=tf, sort=TRUE)
  }
  wordcloud(words = dtm_selected$word, freq = dtm_selected$n,
            min.freq = 1,max.words=max.words, random.order=FALSE,
            colors=brewer.pal(8, "Dark2"), scale=scale,
            main="Title", use.r.layout = TRUE)
  text(x=0.5, y=1, paste("Topic", i))
}
```

For getting specific and more individual words for each cloud, we use the TFIDF in the first step.

4.1 Wordclouds using tfidf

```
par(mfrow=c(2,4))
par(mar=c(1,1,1,1))
set.seed(123)
plot_wordcloud(corpus, selection=ind1[,1], i=1, scale=c(2.1,0.05),
               max.words = 45)
plot_wordcloud(corpus, selection=ind2[,1], i=2)
plot_wordcloud(corpus, selection=ind3[,1], i=3, scale=c(1.6,0.005),
               max.words = 40)
plot_wordcloud(corpus, selection=ind4[,1], i=4)
plot_wordcloud(corpus, selection=ind5[,1], i=5, scale=c(1.8,0.001),
               max.words = 40)
plot_wordcloud(corpus, selection=ind6[,1], i=6, scale=c(1.8,0.001),
               max.words = 40)
plot_wordcloud(corpus, selection=ind7[,1], i=7, scale=c(2,0.2),
               max.words = 45)
```

4.2 Wordclouds using tf

The same can be done using the regular term frequency.

```
par(mfrow=c(2,4))
par(mar=c(1,1,0.5,1))
set.seed(123)
plot_wordcloud(corpus, selection=ind1[,1], i=1, freq="tf", scale=c(2,0.05))
plot_wordcloud(corpus, selection=ind2[,1], i=2, freq="tf", scale=c(2,0.05))
plot_wordcloud(corpus, selection=ind3[,1], i=3, freq="tf", scale=c(2,0.03),
               max.words = 40)
plot_wordcloud(corpus, selection=ind4[,1], i=4, freq="tf", scale=c(2,0.3))
```

```r
plot_wordcloud(corpus, selection=ind5[,1], i=5, freq="tf", scale=c(2.5,0.1))
plot_wordcloud(corpus, selection=ind6[,1], i=6, freq="tf")
plot_wordcloud(corpus, selection=ind7[,1], i=7, freq="tf", scale=c(2.5,0.2),
               max.words = 45)
```

5 Embedding via tfidf

Now it's interesting to see if embedding with tfidf will cluster other groups or the same. So we will reduce the Document Term Matrix to $0.8 * \mathrm{M}$ words, which amounts to a reduction by 20%

```r
dtm_50 <- dtm %>% dtm_remove_tfidf(top=0.8*M)
set.seed(123)
documents_lda_2 <- LDA(dtm_50, method="Gibbs",
                    k = 7, control = list(seed = 1234))
```

```r
prediction5_2 <- predict(documents_lda_2, newdata=dtm_50, type="topic")
prediction5_2 <- merge(prediction5_2, classes, by.x="doc_id", by.y="No")
# compare topic 1 with topic 2, 3, 4 and 5
ind1_2 <- prediction5_2 %>% filter(topic==1) %>% select(doc_id, Group)
ind2_2 <- prediction5_2 %>% filter(topic==2) %>% select(doc_id, Group)
ind3_2 <- prediction5_2 %>% filter(topic==3) %>% select(doc_id, Group)
ind4_2 <- prediction5_2 %>% filter(topic==4) %>% select(doc_id, Group)
ind5_2 <- prediction5_2 %>% filter(topic==5) %>% select(doc_id, Group)
ind6_2 <- prediction5_2 %>% filter(topic==6) %>% select(doc_id, Group)
ind7_2 <- prediction5_2 %>% filter(topic==7) %>% select(doc_id, Group)
```

```r
ext_gamma_matrix(documents_lda_2) %>%
  round(2) %>%
  stargazer(summary=F, rownames = F, header=F,
          title="Gamma matrix extracted from model for embedding with tfidf",
```

Table 11: Documents for Topic 1

Topic_embedding_0.8	doc_id	Group
1	16	5
1	18	5
1	19	5
1	21	6

Table 12: Documents for Topic 2

Topic_embedding_0.8	doc_id	Group
2	17	5
2	26	7
2	27	7

Table 13: Documents for Topic 3

Topic_embedding_0.8	doc_id	Group
3	10	4
3	11	4
3	12	4
3	3	2
3	4	2
3	5	2
3	6	3
3	7	4
3	8	4
3	9	4

Table 14: Documents for Topic 4

Topic_embedding_0.8	doc_id	Group
4	1	1
4	2	1

Table 15: Documents for Topic 5

Topic_embedding_0.8	doc_id	Group
5	13	3
5	14	4
5	15	5
5	20	5
5	28	7

Table 16: Documents for Topic 6

Topic_embedding_0.8	doc_id	Group
6	25	6

Table 17: Documents for Topic 7

Topic_embedding_0.8	doc_id	Group
7	22	6
7	23	6
7	24	6

```
label="extract2")
```

Table 18: Gamma matrix extracted from model for embedding with tfidf

document	1	2	3	4	5	6	7
1	0	0	0	0.98	0.02	0	0
2	0.01	0.01	0	0.85	0.11	0.01	0
3	0.22	0.06	0.42	0.02	0.26	0.01	0.02
4	0.02	0.04	0.54	0.02	0.33	0.04	0.01
5	0.08	0.01	0.85	0.01	0.03	0.02	0.01
6	0.23	0.01	0.7	0.02	0.01	0.01	0.02
7	0.05	0.02	0.86	0.01	0.02	0.04	0.01
8	0.03	0.03	0.84	0.02	0.04	0.01	0.02
9	0.07	0.04	0.76	0.01	0.04	0.05	0.03
10	0.02	0.01	0.94	0	0	0.02	0.01
11	0.08	0.01	0.87	0.01	0.01	0.01	0.01
12	0.08	0	0.88	0.01	0.01	0.01	0.01
13	0.02	0.04	0.04	0.17	0.73	0	0
14	0.05	0.01	0.43	0.03	0.48	0	0
15	0.05	0.01	0.02	0.02	0.88	0.01	0.01
16	0.91	0.01	0.01	0	0.01	0	0.06
17	0.05	0.81	0.01	0.01	0.03	0.05	0.03
18	0.91	0.01	0.01	0	0.02	0	0.06
19	0.78	0.05	0.05	0.01	0.1	0.01	0.01
20	0.06	0.02	0.03	0.01	0.82	0.05	0.01
21	0.95	0.01	0.01	0.01	0.01	0.01	0.01
22	0.04	0	0.01	0	0.01	0	0.93
23	0.17	0.03	0	0.01	0.02	0.01	0.76
24	0.05	0.01	0.01	0	0.01	0.01	0.91
25	0.02	0	0.01	0	0.01	0.95	0.01
26	0.02	0.73	0.02	0.04	0.18	0	0.01
27	0	0.68	0.01	0.21	0.09	0.01	0.01
28	0.02	0.14	0.01	0.12	0.69	0.01	0.01

5.1 Wordclouds

```
par(mfrow=c(2,4))
par(mar=c(1,1,0.5,1))
set.seed(123)
plot_wordcloud(corpus, selection=ind1_2[,1], i=1, scale=c(2,0.2))
plot_wordcloud(corpus, selection=ind2_2[,1], i=2, scale=c(2.5,0.1))
plot_wordcloud(corpus, selection=ind3_2[,1], i=3, scale=c(1.5,0.001),
                max.words = 30)
plot_wordcloud(corpus, selection=ind4_2[,1], i=4, scale=c(2,0.1))
plot_wordcloud(corpus, selection=ind5_2[,1], i=5, scale=c(2.5,0.1),
                max.words = 40)
plot_wordcloud(corpus, selection=ind6_2[,1], i=6, scale=c(2,0.1),
                max.words = 40)
plot_wordcloud(corpus, selection=ind7_2[,1], i=7, scale=c(2,0.03),
                max.words = 35)
```

We want to give an overview over the clustered documents using the tfidf embedding.

Table 19: Documents for Topic 1

Topic	doc_id	Group
1	16	5
1	18	5
1	19	5
1	21	6

Now we use the validation measure we used for the Example 1.

```
validate_LDAclassification <- function(predict_table){
  # gamma_matrix is an object of the function ext_gamma_matrix()
```

Table 20: Documents for Topic 2

Topic	doc_id	Group
2	17	5
2	26	7
2	27	7

Table 21: Documents for Topic 3

Topic	doc_id	Group
3	10	4
3	11	4
3	12	4
3	3	2
3	4	2
3	5	2
3	6	3
3	7	4
3	8	4
3	9	4

Table 22: Documents for Topic 4

Topic	doc_id	Group
4	1	1
4	2	1

Table 23: Documents for Topic 5

Topic	doc_id	Group
5	13	3
5	14	4
5	15	5
5	20	5
5	28	7

Table 24: Documents for Topic 6

Topic	doc_id	Group
6	25	6

Table 25: Documents for Topic 7

Topic	doc_id	Group
7	22	6
7	23	6
7	24	6

```r
  # First we'd find the topic that was most associated with
  # each chapter
  conversion <- predict_table %>%
    select(Group, topic) %>%
    group_by(Group) %>%
    top_n(1,topic) %>%
    unique()

  predict_table %>%
    left_join(conversion, by=c("topic")) %>%
    filter(Group.x!=Group.y) %>%
    nrow()/nrow(predict_table)
}
```

On both full bag of words and 80% embedding via Tfidf

```r
predict_table <- prediction5 %>% select(doc_id, topic) %>%
  merge( y=classes, by.x=1, by.y=1)

validate_LDAclassification(predict_table)
```

```
## [1] 0.5
```

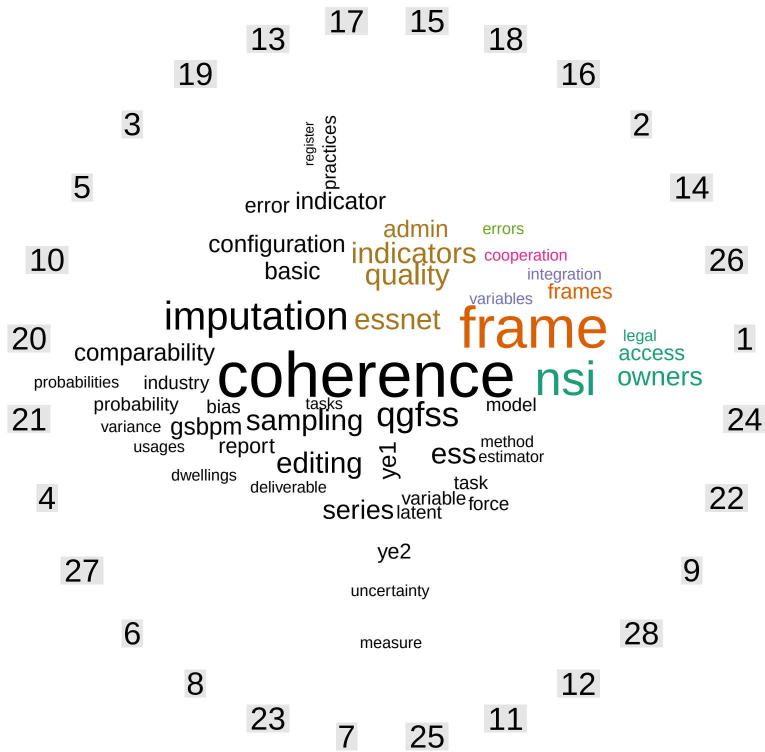
```r
predict_table2 <- prediction5_2 %>% select(doc_id, topic) %>%
  merge( y=classes, by.x=1, by.y=1)

validate_LDAclassification(predict_table2)
```

```
## [1] 0.7857143
```

6 coherence cloud

```r
dtm_50 %>% as.matrix() %>% t() %>% comparison.cloud(scale=c(2,.05),
                                                     max.words=50, title.size = 1)
```

```
## Warning in brewer.pal(max(3, ncol(term.matrix)), "Dark2"): n too large, allowed maximum for palette I
## Returning the palette you asked for with that many colors
```


17
15
13
18
19
16
3
2
5
14
10
26
20
1
21
24
4
22
27
9
6
28
8
12
23
7
25
11
register
practices
error
indicator
admin
errors
configuration
indicators
cooperation
basic
quality
integration
variables
frames
imputation
essnet
frame
legal
access
comparability
coherence
nsi
owners
probabilities
industry
probability
bias
tasks
qgfss
model
variance
gsbpm
sampling
usages
report
method
editing
ye1
ess
estimator
dwellings
deliverable
task
variable
force
series
latent
ye2
uncertainty
measure