Gutenberg Data via LDA - Documentation

Sebastian Knigge

6 8 2019

Contents

1 Setup

Following packages were used in this script:

```
# loading packages
library(gutenbergr)
library(dplyr)
library(tidyr)
library(stringr)
library(tidytext)
library(udpipe)
library(topicmodels)
library(ggplot2)
library(parallel)
library(foreach)
```

2 Example 1 (6 Books)

2.1 Get Data (Sampling)

Sampling of the books and converting it to the *tidytext*-format is the essential step for setting up the LDA model. Following functions include the sampling procedure of the *gutenbergr* library.

```
sampling_books <- function(seed=1234, n=20){
  # sample n books from the whole library
  set.seed(seed)
```

```r
  gutenberg_works() %>%
    # select works with title
    dplyr::filter(!is.na(title)&!is.na(gutenberg_bookshelf)) %>%
    # set the sample sitze
    sample_n(n) %>%
    # set a special download link
    gutenberg_download(
      mirror = "http://mirrors.xmission.com/gutenberg/")
}

set_up_books <- function(n_books=4, seed=1992){
  # initial book sample
  books <- sampling_books(n=n_books, seed=seed)
  by_chapter <- books %>%
    group_by(gutenberg_id) %>%
    # split in chapters
    mutate(chapter = cumsum(str_detect(text, regex("^chapter ", ignore_case = TRUE)))) %>%
    ungroup() %>%
    # exclude books without chapters
    dplyr::filter(chapter > 0)
  return(by_chapter)
}

shorten_titles <- function(titles){
  # shorten very long book titles by setting
  # a subset of characters of the first line
  # of the title
  sub_inds <- titles %>%
    regexpr(pattern="\\n|\\r")-1
  sub_inds[sub_inds<0] <- nchar(titles)[sub_inds<0]
  sub_inds <- pmin(sub_inds, 45)
  titles %>%
    substr(1,sub_inds)
}

get_titles <- function(x, n_books){
  # get the sampled gutenberg_ids
  unique_ids <- x %>%
    select(gutenberg_id) %>%
    unique() %>% unlist()
  # get the titles
  titles <- gutenberg_works() %>%
    dplyr::filter(gutenberg_id %in% unique_ids) %>%
    select(gutenberg_id, title, author) %>%
    mutate(title=shorten_titles(title))
  # get the number of gutenberg ids
  len <- nrow(titles)
  if(n_books!=len) warning(paste("--- ",n_books-len,
                                 " books have 0 chapters --- "))
  # the output as a list
  ret <- list(
    titles=titles,
    len=len
```

```r
  )
  return(ret)
}

append_by_chapter <- function(x=by_chapter, n_books, seed_index=1){
  # append the books matrix until
  # we get the desired number of books n_books
  titles <- get_titles(x, n_books)
  n <- titles$len
  while (n<n_books) {
    book2add <- sampling_books(n=1, seed=seed_index)
    by_chapter_add <- book2add %>%
      group_by(gutenberg_id) %>%
      # split in chapters
      mutate(chapter = cumsum(str_detect(text, regex("^chapter ", ignore_case = TRUE)))) %>%
      ungroup() %>%
      # exclude books without chapters
      dplyr::filter(chapter > 2)
    titles2add <- get_titles(by_chapter_add, 1)
    # adding the book to by_chapter if there are chapters in the
    # book plus it is not in the data already
    if (titles2add$len==1) if(!titles2add$titles$gutenberg_id%in%titles$titles$gutenberg_id) {
      x <- bind_rows(x, by_chapter_add)
    }
    n<-get_titles(x, n)$len
    seed_index <- seed_index+1
  }
  return(x)
}

exclude_stop_words <- function(x){
  # unite chapter and document title
  by_chapter_word <- x %>%
    unite(document, gutenberg_id, chapter) %>%
    # split into words
    unnest_tokens(word, text)
  # import tibble stop words
  data(stop_words)
  # find document-word counts
  word_counts <- by_chapter_word %>%
    # exclude stop words
    anti_join(stop_words) %>%
    # count each word by chapter
    count(document, word, sort = TRUE) %>%
    ungroup()
  return(word_counts)
}

convert_to_dtm <- function(x, minfq = 2){
  # get into a format lda can handle
  chapters_dtm <- x %>%
    select(doc_id=document, term=word, freq=n) %>%
    document_term_matrix() %>%
```

```r
    # reduce by low frequencies
    dtm_remove_lowfreq(minfreq = minfq)
  return(chapters_dtm)
}


convert_to_dtm_2 <- function(x, n=n, minfq = 2, top=10000){
  # get into a format lda can handle
  chapters_dtm <- x %>%
    select(doc_id=document, term=word, freq=n) %>%
    document_term_matrix() %>%
    # reduce by low frequencies
    dtm_remove_tfidf(top=top)
  return(chapters_dtm)
}
```

Now we can use these functions to get to the initial corpus sample. In this example 6 books are choosen.

```r
n_books <- 6
by_chapter <- set_up_books(n_books=n_books, seed=222)
get_titles(by_chapter, n_books)
```

```
## Warning in get_titles(by_chapter, n_books): --- 5 books have 0 chapters ---
```

```
## $titles
## # A tibble: 1 x 3
##   gutenberg_id title                                      author
##          <int> <chr>                                      <chr>
## 1         2095 Clotelle: A Tale of the Southern States Brown, William Wells
##
## $len
## [1] 1
```

The function *set_up_books()* returns a warning that several books seem to consist of only one chapter. In order to get a corpus consisting out of 6 books, the function *append_by_chaper()* is used, which fills up the corpus to the desired number of books.

```r
appended_by_chapter <- append_by_chapter(x=by_chapter, n_books = n_books)
word_counts <- exclude_stop_words(appended_by_chapter)
```

```
## Joining, by = "word"
```

In table 7 the sampled titles for the book sample with the seed 222 are displayed. It appears through the function *append_by:chapter()* one book was added, called "Clotelle: A Tale of the Southern States".

```r
titles <- get_titles(appended_by_chapter, n_books)
titles$titles %>% stargazer(summary=FALSE, font.size = "footnotesize",
                           header=FALSE, title="Book-titles", rownames=FALSE,
                           label="titles:6books")
```

We also want to check if the book categories (i.e. gutenberg bookshelfes) are different. See Table (2) for comparison.

```r
gbids <- titles$titles$gutenberg_id
categories <- gutenberg_works() %>%
  filter(gutenberg_id %in% gbids) %>%
  select(gutenberg_id, gutenberg_bookshelf)
categories %>% stargazer(summary=FALSE, font.size = "footnotesize",
```

Table 1: Book-titles

gutenberg_id	title	author
2095	Clotelle: A Tale of the Southern States	Brown, William Wells
6315	The Awakening of Helena Richie	Deland, Margaret Wade Campbell
6971	Judaism	Abrahams, Israel
7635	The Disowned — Volume 05	Lytton, Edward Bulwer Lytton, Baron
10319	Dave Darrin's Third Year at Annapolis; Or, Le	Hancock, H. Irving (Harrie Irving)
21039	Boycotted, and Other Stories	Reed, Talbot Baines

```
                      header=FALSE, title="Book-categories", rownames=FALSE,
                      label="categories:6books")
```

Table 2: Book-categories

gutenberg_id	gutenberg_bookshelf
2095	African American Writers
6315	Bestsellers, American, 1895-1923
6971	Judaism
7635	Historical Fiction
10319	Children's Book Series
21039	School Stories

Obviously the corpus is very diverse. It is a good sample, in order to try to cluster the chapters of the books with the aid of LDA.

2.2 Reduction of Dimensionality

In the set up we have another parameter to adjust. The function *convert_to_dtm* takes the parameter *minfq*, which is used to reduce the "bag of words" (i.e. dimensionality). *minfq* is the minimum frequency for the bag of words dictionary. I will refer to this as "embedding". Let us set it to 2 in this case, meaning that we include a word only if the frequency is 2 or more.

```
chapters_dtm <- convert_to_dtm(word_counts, minfq=2)
( M_f2 <- ncol(chapters_dtm) )
```

```
## [1] 8597
```

Let us compare it to the case including all words.

```
chapters_dtm_all <- convert_to_dtm(word_counts, minfq=0)
( M <- ncol(chapters_dtm_all) )
```

```
## [1] 15186
```

We also want to compare this to a reduction of the word dictionary by the TFIDF. We are using a reduction by 50% of the dimension of the original bag of words.

```
chapters_dtm_tfidf <- convert_to_dtm_2(word_counts, top=(0.1*M))
ncol(chapters_dtm_tfidf)
```

```
## [1] 1518
```

2.3 Application of LDA Model to Full Corpus

2.3.1 Fitting via VEM

We set up the LDA model for the shrinked embedding corpus via frequency=2. In a first try we are using the default "VEM-Algorithm" to fit the model.

```
tim1 <- Sys.time()
chapters_lda <- LDA(chapters_dtm,
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_1 <- tim2-tim1
```

In comparison we will set up the LDA model for the full word embedding corpus.

```
tim1 <- Sys.time()
chapters_lda_all <- LDA(chapters_dtm_all,
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_all <- tim2-tim1
```

The third LDA fit builds up on data from the shrinked dictionary/bag of words by TFIDF.

```
tim1 <- Sys.time()
chapters_lda_tfidf <- LDA(chapters_dtm_tfidf,
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_tfidf <- tim2-tim1
chapters_lda
```

```
## A LDA_VEM topic model with 6 topics.
```

Now we evaluate the model all in once - that is - we analyze the clustering on the entire data set.

```
ext_gamma_matrix <- function(model){
  # get gamma matrix for chapter probabilities
  chapters_gamma <- tidy(model, matrix = "gamma")
  # split joint name of book and chapter
  chapters_gamma <- chapters_gamma %>%
    separate(document, c("gutenberg_id", "chapter"), sep = "_", convert = TRUE)
  # get matrix with probabilities for each topic per chapter
  # this matrix is just information and will in this form of the function
  # not be returned
  gamma_per_chapter <- chapters_gamma %>%
    spread(topic, gamma)
  return(chapters_gamma)
}

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

  #First we'd find the topic that was most associated with
  # each chapter using top_n(), which is effectively the
  # "classification" of that chapter
  chapter_classifications <- gamma_matrix %>%
    group_by(gutenberg_id, chapter) %>%
    top_n(1, gamma) %>%
```

```
    ungroup()

  # We can then compare each to the "consensus"
  # topic for each book (the most common topic among its chapters),
  # and see which were most often misidentified.
  book_topics <- chapter_classifications %>%
    count(gutenberg_id, topic) %>%
    group_by(gutenberg_id) %>%
    # just keep the most frequent one
    top_n(1, n) %>%
    ungroup() %>%
    # keep title called census and topic
    transmute(consensus = gutenberg_id, topic)

  # check the fraction of missclassification
  Join <- chapter_classifications %>%
    inner_join(book_topics, by = "topic")
    # missmatches
    Join %>% dplyr::filter(gutenberg_id != consensus) %>%
    nrow()/nrow(Join)
}
```

Now we exclude for each of the 3 embedings the - so called - beta matrix and compare the most likely result with the real results. Depending on how good the LDA will seperate the books, this influences the goodness of the fit.

```
misc.rate_1 <- ext_gamma_matrix(chapters_lda) %>%
  validate_LDAclassification()
```

```
misc.rate_all <- ext_gamma_matrix(chapters_lda_all) %>%
  validate_LDAclassification()
```

```
misc.rate_tfidf <- ext_gamma_matrix(chapters_lda_tfidf) %>%
  validate_LDAclassification()
```

The following matrix gives an overview of the fitting time and the results of the 3 different fits.

```
performance_matrix <- data.frame(freq2.embedding=c(misc.rate_1, u_1),
           all.embedding=c(misc.rate_all, u_all),
           tfidf=c(misc.rate_tfidf, u_tfidf))
rownames(performance_matrix) <- c("missc. rate", "time")
performance_matrix %>%  stargazer(summary=FALSE, header=F, title = "LDA via VEM")
```

Table 3: LDA via VEM

	freq2.embedding	all.embedding	tfidf
missc. rate	0.385	0.462	0.383
time	12.800	9.911	0.492

We run the same calculations for a different sample. This means that we set up a new random sample from the Gutenberg books by again using the sample function with another randomness factor (seed). Again, it has to be a sample that is as diverse as possible, i.e. contains books from different categories. In the following we set up again three bag of words, each using the three embeding methods mentioned above. Lastly, we perform the same evaluation method as just seen to evaluate the method for this sample as well.

```r
n_books_sec <- 6
by_chapter_sec <- set_up_books(n_books=n_books_sec, seed=101)
appended_by_chapter_sec <- append_by_chapter(x=by_chapter_sec, n_books = n_books_sec)
word_counts_sec <- exclude_stop_words(appended_by_chapter_sec)
```

```
## Joining, by = "word"
```

```r
titles_sec <- get_titles(appended_by_chapter_sec, n_books)
gbids_sec <- titles_sec$titles$gutenberg_id
categories_sec <- gutenberg_works() %>%
  filter(gutenberg_id %in% gbids_sec) %>%
  select(gutenberg_id, gutenberg_bookshelf)

# embedding 1
chapters_dtm_sec <- convert_to_dtm(word_counts_sec, minfq=2)
ncol(chapters_dtm_sec)
```

[1] 8565

```r
# embedding 2
chapters_dtm_all_sec <- convert_to_dtm(word_counts_sec, minfq=0)
( M2 <- ncol(chapters_dtm_all_sec) )
```

[1] 15328

```r
# embedding 3
chapters_dtm_tfidf_sec <- convert_to_dtm_2(word_counts_sec, top=(M2*0.5))

# embedding 1
tim1 <- Sys.time()
chapters_lda_sec <- LDA(chapters_dtm_sec,
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_1 <- tim2-tim1

# embedding 2
tim1 <- Sys.time()
chapters_lda_all_sec <- LDA(chapters_dtm_all_sec,
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_all <- tim2-tim1

# embedding 3
tim1 <- Sys.time()
chapters_lda_tfidf_sec <- LDA(chapters_dtm_tfidf_sec,
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_tfidf <- tim2-tim1

# embedding 1
misc.rate_1_sec <- ext_gamma_matrix(chapters_lda_sec) %>%
  validate_LDAclassification()

# embedding 2
misc.rate_all_sec <- ext_gamma_matrix(chapters_lda_all_sec) %>%
  validate_LDAclassification()
```

```r
# embedding 3
misc.rate_tfidf_sec <- ext_gamma_matrix(chapters_lda_tfidf_sec) %>%
  validate_LDAclassification()

# overview
performance_matrix <- data.frame(freq2.embedding=c(misc.rate_1_sec, u_1),
           all.embedding=c(misc.rate_all_sec, u_all),
           tfidf=c(misc.rate_tfidf_sec, u_tfidf))
rownames(performance_matrix) <- c("missc. rate", "time")
performance_matrix %>%  stargazer(summary=FALSE, header=F, title = "LDA via VEM second sample")
```

Table 4: LDA via VEM second sample

	freq2.embedding	all.embedding	tfidf
missc. rate	0.227	0.362	0.222
time	8.244	8.191	1.664

Surprisingly, the run-time to fit the LDA model for the embedding using all words, does not take way longer than the embedding using a lower frequency. The fit of the model with the reduced bag of words via tfidf takes considerably less time.

2.3.2 Comparison to Fit via Gibbs Sampling

For comparison, we will check the results and the run-time for the fit via Gibbs Sampling.

```r
tim1 <- Sys.time()
chapters_lda_Gibbs <- LDA(chapters_dtm, method="Gibbs",
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_1_Gibbs <- tim2-tim1
```

```r
tim1 <- Sys.time()
chapters_lda_all_Gibbs <- LDA(chapters_dtm_all, method = "Gibbs",
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_all_Gibbs <- tim2-tim1
```

```r
tim1 <- Sys.time()
chapters_lda_tfidf_Gibbs <- LDA(chapters_dtm_tfidf,  method="Gibbs",
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_tfidf_Gibbs <- tim2-tim1
```

```r
misc.rate_1_Gibbs <- ext_gamma_matrix(chapters_lda_Gibbs) %>%
  validate_LDAclassification()
```

```r
misc.rate_all_Gibbs <- ext_gamma_matrix(chapters_lda_all_Gibbs) %>%
  validate_LDAclassification()
```

```r
misc.rate_tfidf_Gibbs <- ext_gamma_matrix(chapters_lda_tfidf_Gibbs) %>%
  validate_LDAclassification()
```

```r
performance_matrix <- data.frame(freq2.embedding=c(misc.rate_1_Gibbs, u_1_Gibbs),
           all.embedding=c(misc.rate_all_Gibbs, u_all_Gibbs),
           tfidf=c(misc.rate_tfidf_Gibbs, u_tfidf_Gibbs))
rownames(performance_matrix) <- c("missc. rate", "time")
performance_matrix %>%  stargazer(summary=FALSE, header=F, title = "LDA via Gibbs sampling")
```

Table 5: LDA via Gibbs sampling

	freq2.embedding	all.embedding	tfidf
missc. rate	0.051	0.043	0.077
time	18.359	22.408	7.649

Apparently Gibbs Sampling takes a bit longer than the VEM algorithm, but its results with regards to the correct "classification" (missclassification rate) are way better.

Again we try the second sample, using a different seed. Only Gibbs Sampling is evaluated in this section.

```r
tim1 <- Sys.time()
chapters_lda_Gibbs_sec <- LDA(chapters_dtm_sec, method="Gibbs",
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_1_Gibbs <- tim2-tim1

tim1 <- Sys.time()
chapters_lda_all_Gibbs_sec <- LDA(chapters_dtm_all_sec, method = "Gibbs",
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_all_Gibbs <- tim2-tim1

tim1 <- Sys.time()
chapters_lda_tfidf_Gibbs_sec <- LDA(chapters_dtm_tfidf_sec,  method="Gibbs",
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_tfidf_Gibbs <- tim2-tim1

misc.rate_1_Gibbs_sec <- ext_gamma_matrix(chapters_lda_Gibbs_sec) %>%
  validate_LDAclassification()

misc.rate_all_Gibbs_sec <- ext_gamma_matrix(chapters_lda_all_Gibbs_sec) %>%
  validate_LDAclassification()

misc.rate_tfidf_Gibbs_sec <- ext_gamma_matrix(chapters_lda_tfidf_Gibbs_sec) %>%
  validate_LDAclassification()

# performance matrix
performance_matrix_sec <- data.frame(freq2.embedding=c(misc.rate_1_Gibbs_sec, u_1_Gibbs),
           all.embedding=c(misc.rate_all_Gibbs_sec, u_all_Gibbs),
           tfidf=c(misc.rate_tfidf_Gibbs_sec, u_tfidf_Gibbs))
rownames(performance_matrix) <- c("missc. rate", "time")
performance_matrix_sec %>%  stargazer(summary=FALSE, header=F, title = "Gibbs second sample")
```

Table 6: Gibbs second sample

	freq2.embedding	all.embedding	tfidf
1	0.058	0.163	0.093
2	14.866	16.745	6.965

2.4 Evaluate Model on Test Set

First we split the data randomly into training- and test-sample.

```
split_for_fit <- function(data, test_ratio=0.03, seed=1234){
  # function to set up the training
  # and the testing set from the input data which
  # is an object from the function convert_to_dtm()
  set.seed(seed)
  N <- nrow(data)
  n_test <- (N*test_ratio) %>% ceiling
  test_ind <- sample(1:N, n_test)
  train_ind <- (1:N)[-test_ind]
  ret <- list(train=data[train_ind,],
              test=data[test_ind,])
  return(ret)
}
```

The *fit_n_evaluate()* function will fit the LDA model and evaluate the goodness of fit for an object of the function *split_for_fit*. The section Validation gives insights in the procedure of how the "consensus" is set up.

```
fit_n_evaluate <- function(split, k=n_books){
  LDA_model <- LDA(split$train, method="Gibbs",
                   k = k, control = list(seed = 1234))
  # use the predict function of udpipe
  # the topic predict funtion already extract the most likely topics
  prediction <- predict(LDA_model, newdata=split$test) %>% .$topic
  # get "consensus" via maximum likelihood
  # first extract the gamma matrix of the model fitted on the training
  # data
  chapters_gamma <- ext_gamma_matrix(LDA_model)
  spreaded_gamma <- chapters_gamma  %>% spread(topic, gamma)
  # get pdfs
  plotm <- spreaded_gamma %>%
    group_by(gutenberg_id) %>%
    # note: pdfs are unnormalized
    summarise_at(2:(titles$len+1), sum)
  topic_link <- plotm %>%
    apply(1, function(x) which.max(x[2:length(x)])) %>%
    cbind(plotm$gutenberg_id) %>%
    as.data.frame()
  # exclude the
  consensus <- split$test %>%
    rownames() %>%
    substr(1,regexpr("_",.)-1) %>%
    as.numeric() %>%
    as.data.frame() %>%
```

```
    # merge it to the topic
    merge(topic_link, by.y="V2", sort=FALSE) %>%
    select(..y)
  # missclassification rate will be returned
  sum(consensus!=prediction)/length(prediction)
}
```

We now can evaluate several fits of a model for different splits. Here is a parallelization for the individual for loops applied. In this case only embeding via TFIDF is used.

```
# setting up how many cores to be used
useable_cores <- parallel::detectCores() - 1
# registering cluster
cl <- parallel::makeCluster(useable_cores)
doParallel::registerDoParallel(cl)
n <- 59
results <- foreach(i = 1:n, .combine = 'c', .export = ls(.GlobalEnv), .packages = c("dplyr", "udpipe",


  chapters_dtm_tfidf %>%
    split_for_fit(seed=12*i) %>%
    fit_n_evaluate()


}
```

```
## Warning in e$fun(obj, substitute(ex), parent.frame(), e$data):
## already exporting variable(s): chapters_dtm_tfidf, ext_gamma_matrix,
## fit_n_evaluate, n_books, split_for_fit, titles
```

```
parallel::stopCluster(cl)
```

This is the output of the simulation over 59 splits and fits. The mean is the final result:

```
results %>% summary
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.2500  0.3093  0.5000  1.0000
```

3 Optimal TFIDF reduction

We want to visualize what the best values for a reduction of the original bag-of-words via tfidf is. I.e. by what % should the bag-of-words be reduced for the best results (lowest misclassification ratio).

```
evaluation_for_embedding <- function(word_counts, frac=0.5) {
  # embedding 3
  chapters_dtm_tfidf <- convert_to_dtm_2(word_counts, top=(M2*frac))
  # fitting
  tim1 <- Sys.time()
  chapters_lda_tfidf_Gibbs <- LDA(chapters_dtm_tfidf,  method="Gibbs",
                    k = n_books, control = list(seed = 1234))
  tim2 <- Sys.time()
  u_tfidf_Gibbs <- tim2-tim1
  # calculating missclassification rate
  misc.rate_tfidf_Gibbs <- ext_gamma_matrix(chapters_lda_tfidf_Gibbs) %>%
```

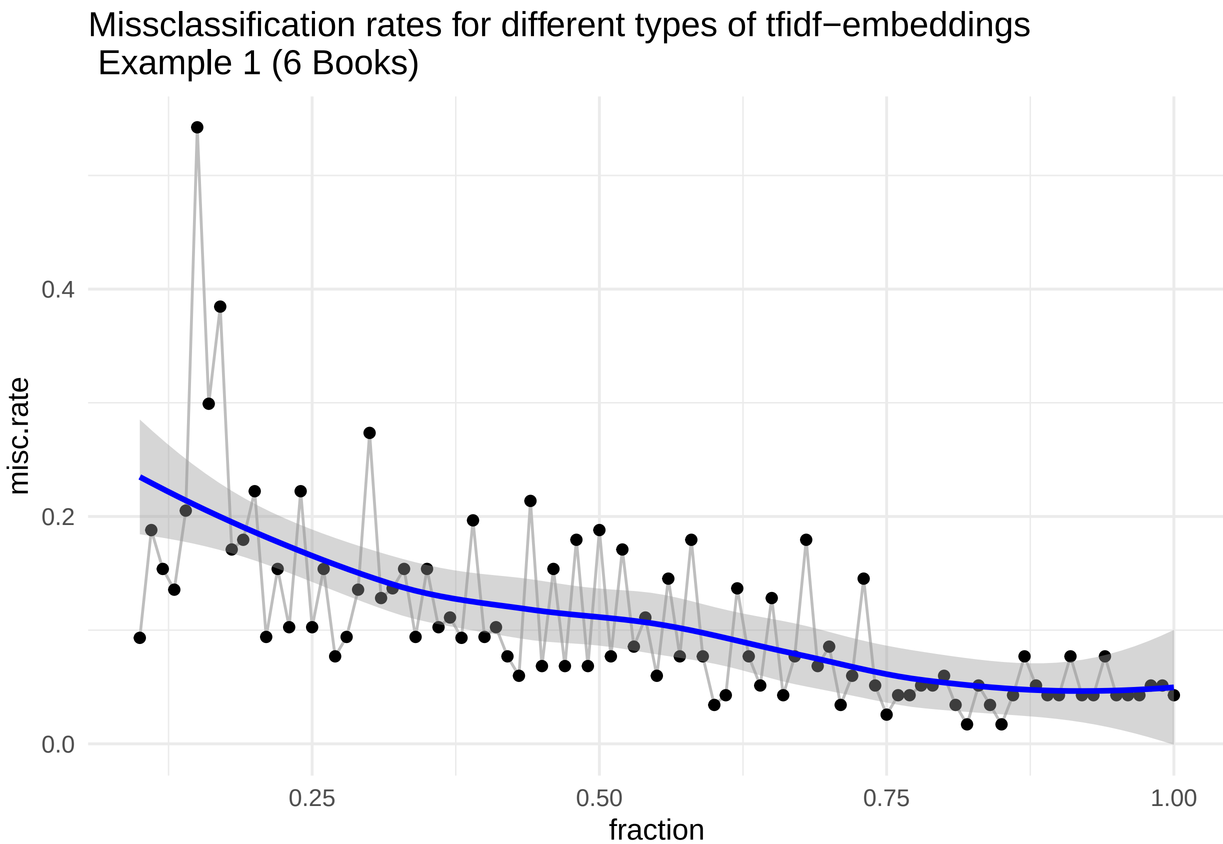
```r
    validate_LDAclassification()
  # function returns a vector including the misc. ratio and the fitting time
  return(c(misc.rate_tfidf_Gibbs,
            as.numeric(u_tfidf_Gibbs)))
}
```

```r
# set up the fractions we want to use
# we use 0.1, 0.2, ..., 1
fractions <- seq(0.1,1,0.01)

# Use parallelization
# registering cluster
cl <- parallel::makeCluster(useable_cores)
doParallel::registerDoParallel(cl)
embedding_performance_matrix <- foreach(i = 1:length(fractions), .combine = 'rbind', .export = ls(.Globa


  evaluation_for_embedding(word_counts, frac=fractions[i]) %>%
    c(.,fractions[i])


}
```

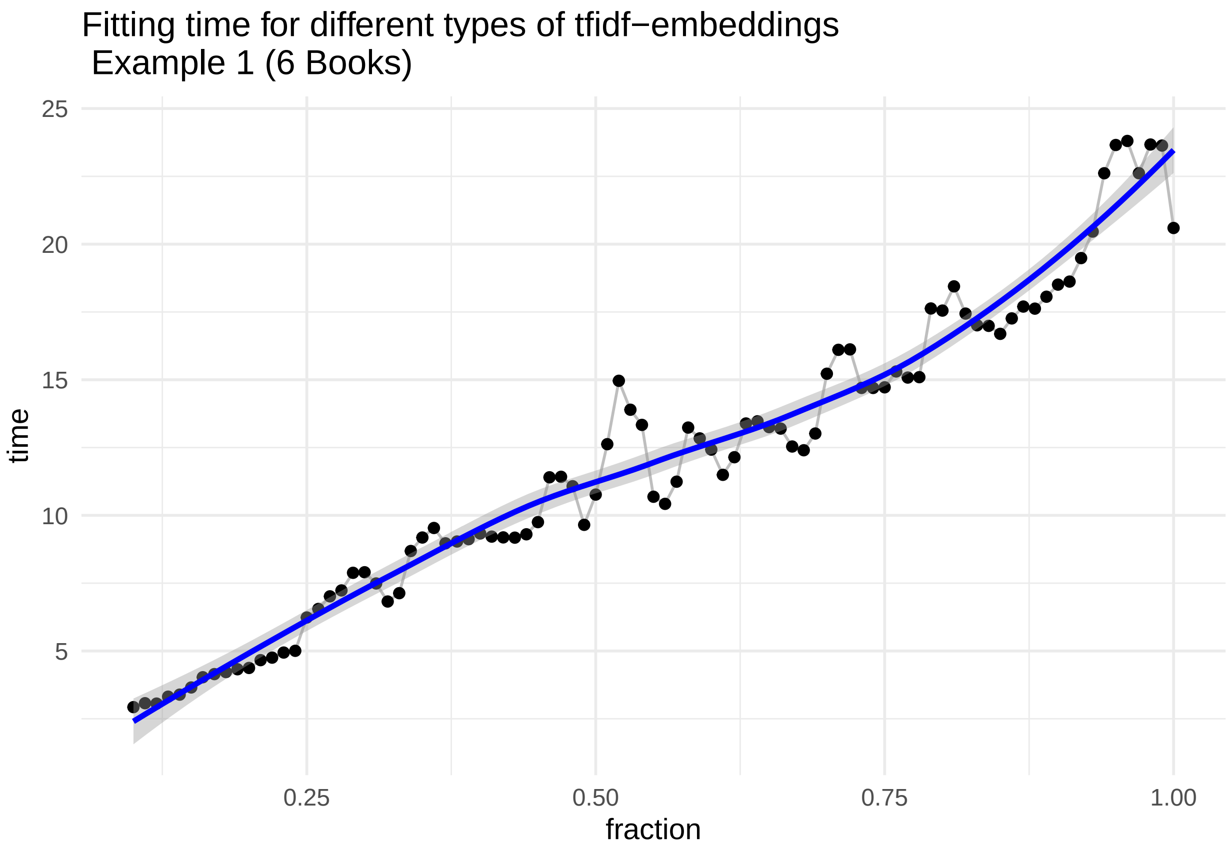
```
## Warning in e$fun(obj, substitute(ex), parent.frame(), e$data): already
## exporting variable(s): convert_to_dtm_2, evaluation_for_embedding,
## ext_gamma_matrix, fractions, M2, n, n_books, validate_LDAclassification,
## word_counts
```

```r
parallel::stopCluster(cl)

# adjust the resulting matrix to a data frame for plotting
colnames(embedding_performance_matrix) <- c("misc.rate","time","fractions")
embedding_performance_matrix <- as.data.frame(embedding_performance_matrix)
```

```r
ggplot(data=embedding_performance_matrix,
       aes(x=fractions, y=misc.rate)) +
  geom_line(col="grey") +
  geom_point() +
  geom_smooth(col="blue", method="loess", span=0.7) +
  theme_minimal() +
  ggtitle("Missclassification rates for different types of tfidf-embeddings \n Example 1 (6 Books)") +
  xlab("fraction")
```




```r
ggplot(data=embedding_performance_matrix,
       aes(x=fractions, y=time)) +
  geom_line(col="grey") +
  geom_point() +
  geom_smooth(col="blue", method="loess", span=0.7) +
  theme_minimal() +
  ggtitle("Fitting time for different types of tfidf-embeddings \n Example 1 (6 Books)")+
  xlab("fraction")
```



Now let us study Example 2.

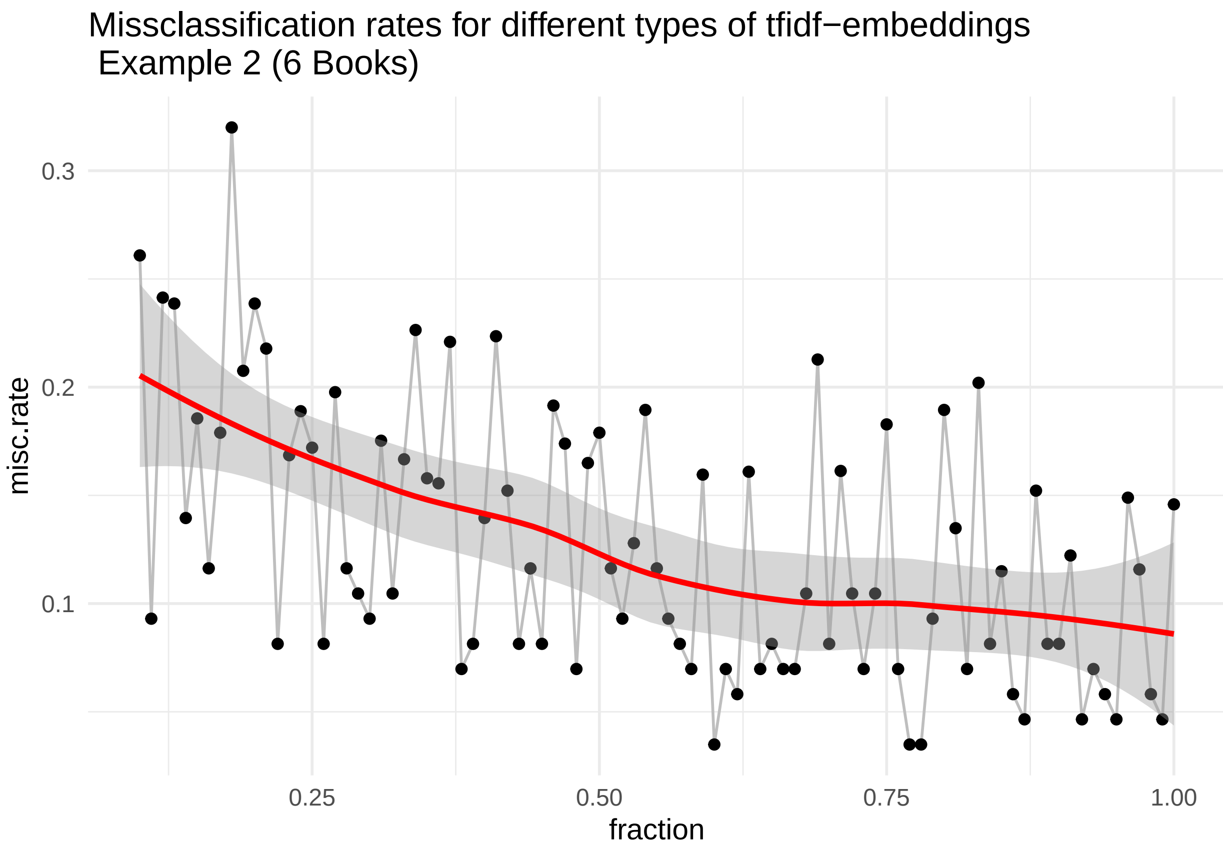
```
# set up the fractions we want to use
# we use 0.1, 0.2, ..., 1
fractions <- seq(0.1,1,0.01)

# Use parallelization
# registering cluster
cl <- parallel::makeCluster(useable_cores)
doParallel::registerDoParallel(cl)
embedding_performance_matrix_sec <- foreach(i = 1:length(fractions), .combine = 'rbind', .export = ls(.(


  evaluation_for_embedding(word_counts_sec, frac=fractions[i]) %>%
    c(.,fractions[i])



}
```

```
## Warning in e$fun(obj, substitute(ex), parent.frame(), e$data): already
## exporting variable(s): convert_to_dtm_2, evaluation_for_embedding,
## ext_gamma_matrix, fractions, M2, n, n_books, validate_LDAclassification,
## word_counts_sec
```

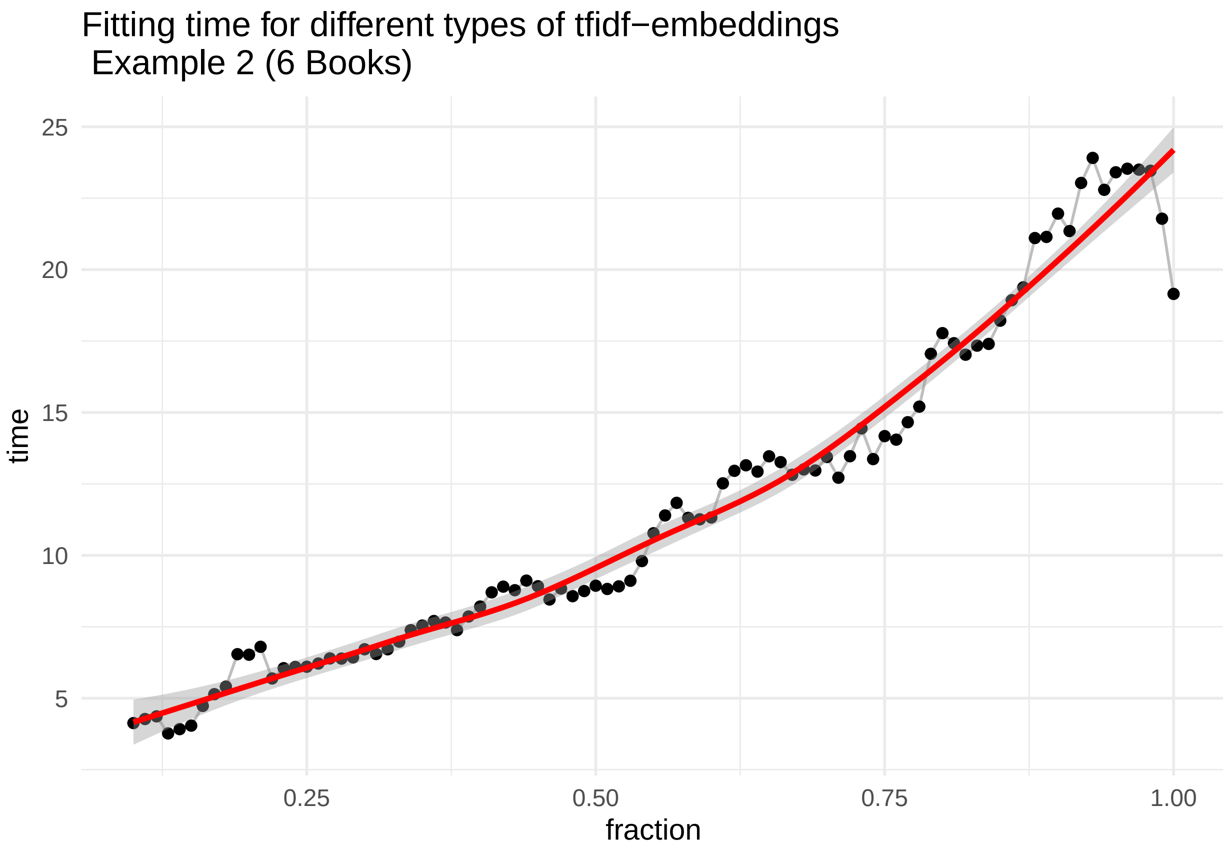
```
parallel::stopCluster(cl)

# adjust the resulting matrix to a data frame for plotting
colnames(embedding_performance_matrix_sec) <- c("misc.rate","time","fractions")
```

```r
embedding_performance_matrix_sec <- as.data.frame(embedding_performance_matrix_sec)
```

```r
ggplot(data=embedding_performance_matrix_sec,
       aes(x=fractions, y=misc.rate)) +
  geom_line(col="grey") +
  geom_point() +
  geom_smooth(col="red", method="loess", span=0.7) +
  theme_minimal() +
  ggtitle("Missclassification rates for different types of tfidf-embeddings \n Example 2 (6 Books)")+
  xlab("fraction")
```




```r
ggplot(data=embedding_performance_matrix_sec,
       aes(x=fractions, y=time)) +
  geom_line(col="grey") +
  geom_point() +
  geom_smooth(col="red", method="loess", span=0.7) +
  theme_minimal() +
  ggtitle("Fitting time for different types of tfidf-embeddings \n Example 2 (6 Books)")+
  xlab("fraction")+
  xlab("fraction")
```

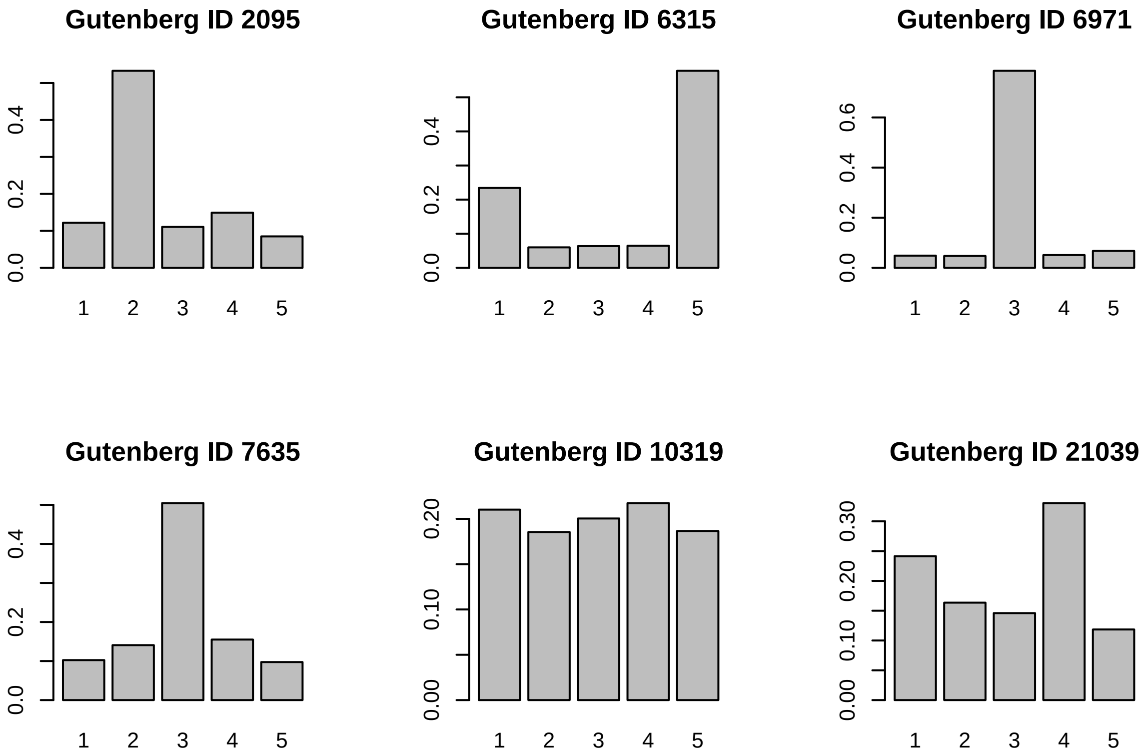


4 Definition of Validation

Since the LDA algorithm just clusters into k topics, it is necessary to evaluate which "topic" refers to which book, in order to calculate a missclassification rate. For each of the books we naively derive a distribution of the assignment to the topics. This is done by accumulating the distributions for each chapter of the book. The most likely assignment of the LDA model is chosen as the "correct" topic. Now calculating the missclassification rate is basically the fraction of those chapters/documents not coinciding with the most likely assignment.

```
LDA_model <- LDA(chapters_dtm_tfidf, method="Gibbs",
                            k = n_books, control = list(seed = 1234))
# get gamma matrix for chapter probabilities
chapters_gamma <- tidy(LDA_model, matrix = "gamma") %>%
  # split joint name of book and chapter
    separate(document, c("gutenberg_id", "chapter"), sep = "_", convert = TRUE)
spreaded_gamma <- chapters_gamma  %>% spread(topic, gamma)
  # get pdfs
plotm <- spreaded_gamma %>%
    group_by(gutenberg_id) %>%
    # note: pdfs are unnormalized
    summarise_at(2:(titles$len+1), sum)
topic_link <- plotm %>%
    apply(1, function(x) which.max(x[2:length(x)])) %>%
    cbind(plotm$gutenberg_id) %>%
    as.data.frame()
```

```r
par(mfrow=c(2,3))
for (i in 1:n_books){
  vec <- plotm[i,2:n_books] %>% unlist()
  barplot(vec/sum(vec), main=paste("Gutenberg ID", plotm[i,1]))
}
```



This procedure does not exclude the fact that two books are assigned to the same topic (in this case e.g. 2 books are assigned to chapter 1 and none to chapter 6). But still, from each of the plots of the "distributions" you can obtain how good the chapters of a single book are classified. The more of the mass of the distribution is on a single topic, the better the chapters of this topic are predicted.

5 Example 3 (10 books)

First we will sample the books, the same way as is it was done in Example 1.

```r
n_books_10 <- 10
by_chapter10 <- set_up_books(n_books=n_books_10, seed=54321)
appended_by_chapter10 <- append_by_chapter(x=by_chapter10, n_books = n_books_10)
word_counts10 <- exclude_stop_words(appended_by_chapter10)
```

```
## Joining, by = "word"
```

```r
titles <- get_titles(appended_by_chapter10, n_books)
```

```
## Warning in get_titles(appended_by_chapter10, n_books): --- -4 books have 0
## chapters ---
```

```r
titles$titles %>% stargazer(summary=FALSE, font.size = "footnotesize",
                            header=FALSE, title="Book-titles Example with 10 Books", rownames=FALSE,
                            label="titles:6books")
```

Table 7: Book-titles Example with 10 Books

gutenberg_id	title	author
1401	Tarzan the Untamed	Burroughs, Edgar Rice
6315	The Awakening of Helena Richie	Deland, Margaret Wade Campbell
6971	Judaism	Abrahams, Israel
7635	The Disowned — Volume 05	Lytton, Edward Bulwer Lytton, Baron
10319	Dave Darrin's Third Year at Annapolis; Or, Le	Hancock, H. Irving (Harrie Irving)
11113	Principal Cairns	Cairns, John
13145	Lippincott's Magazine of Popular Literature a	Various
15735	History of the Negro Race in America From 161	Williams, George Washington
21039	Boycotted, and Other Stories	Reed, Talbot Baines
21710	The Crew of the Water Wagtail	Ballantyne, R. M. (Robert Michael)

```r
gbids <- titles$titles$gutenberg_id
categories <- gutenberg_works() %>%
  filter(gutenberg_id %in% gbids) %>%
  select(gutenberg_id, gutenberg_bookshelf)
categories %>% stargazer(summary=FALSE, font.size = "footnotesize",
                         header=FALSE, title="Book-categories", rownames=FALSE,
                         label="categories:10books")
```

Table 8: Book-categories

gutenberg_id	gutenberg_bookshelf
1401	Adventure/Movie Books
6315	Bestsellers, American, 1895-1923
6971	Judaism
7635	Historical Fiction
10319	Children's Book Series
11113	Famous Scots Series
13145	Lippincott's Magazine
15735	Slavery
21039	School Stories
21710	Children's Fiction

```r
# embedding 1
chapters_dtm_10 <- convert_to_dtm(word_counts10, minfq=2)
( M3_f2 <- ncol(chapters_dtm_10) )
```

[1] 17742

```r
# embedding 2
chapters_dtm_all_10 <- convert_to_dtm(word_counts10, minfq=0)
( M3 <- ncol(chapters_dtm_all_10) )
```

[1] 29101

```r
# embedding 3
chapters_dtm_tfidf_10 <- convert_to_dtm_2(word_counts10, top=(0.5*M3))

# embedding 1
tim1 <- Sys.time()
chapters_lda_10 <- LDA(chapters_dtm_10,  method="Gibbs",
                      k = n_books_10, control = list(seed = 1234))
tim2 <- Sys.time()
u_1 <- tim2-tim1
```

```
# embedding 2
tim1 <- Sys.time()
chapters_lda_all_10 <- LDA(chapters_dtm_all_10,  method="Gibbs",
                          k = n_books_10, control = list(seed = 1234))
tim2 <- Sys.time()
u_all <- tim2-tim1

# embedding 3
tim1 <- Sys.time()
chapters_lda_tfidf_10 <- LDA(chapters_dtm_tfidf_10,  method="Gibbs",
                          k = n_books_10, control = list(seed = 1234))
tim2 <- Sys.time()
u_tfidf <- tim2-tim1

# embedding 1
misc.rate_1_10 <- ext_gamma_matrix(chapters_lda_10) %>%
  validate_LDAclassification()

# embedding 2
misc.rate_all_10 <- ext_gamma_matrix(chapters_lda_all_10) %>%
  validate_LDAclassification()

# embedding 3
misc.rate_tfidf_10 <- ext_gamma_matrix(chapters_lda_tfidf_10) %>%
  validate_LDAclassification()

# overview
performance_matrix_10_Gibbs <- data.frame(freq2.embedding=c(misc.rate_1_10, u_1),
           all.embedding=c(misc.rate_all_10, u_all),
           tfidf=c(misc.rate_tfidf_10, u_tfidf))
rownames(performance_matrix_10_Gibbs) <- c("missc. rate", "time")
performance_matrix_10_Gibbs %>%  stargazer(summary=FALSE, header=F, title = "LDA via Gibbs 10 books exa
```

Table 9: LDA via Gibbs 10 books example

	freq2.embedding	all.embedding	tfidf
missc. rate	0.341	0.322	0.317
time	1.301	1.514	36.493

```
# embedding 1
tim1 <- Sys.time()
chapters_lda_10_VEM <- LDA(chapters_dtm_10,  method="VEM",
                          k = n_books_10, control = list(seed = 1234))
tim2 <- Sys.time()
u_1 <- tim2-tim1

# embedding 2
tim1 <- Sys.time()
chapters_lda_all_10_VEM <- LDA(chapters_dtm_all_10,  method="VEM",
                          k = n_books_10, control = list(seed = 1234))
tim2 <- Sys.time()
u_all <- tim2-tim1
```

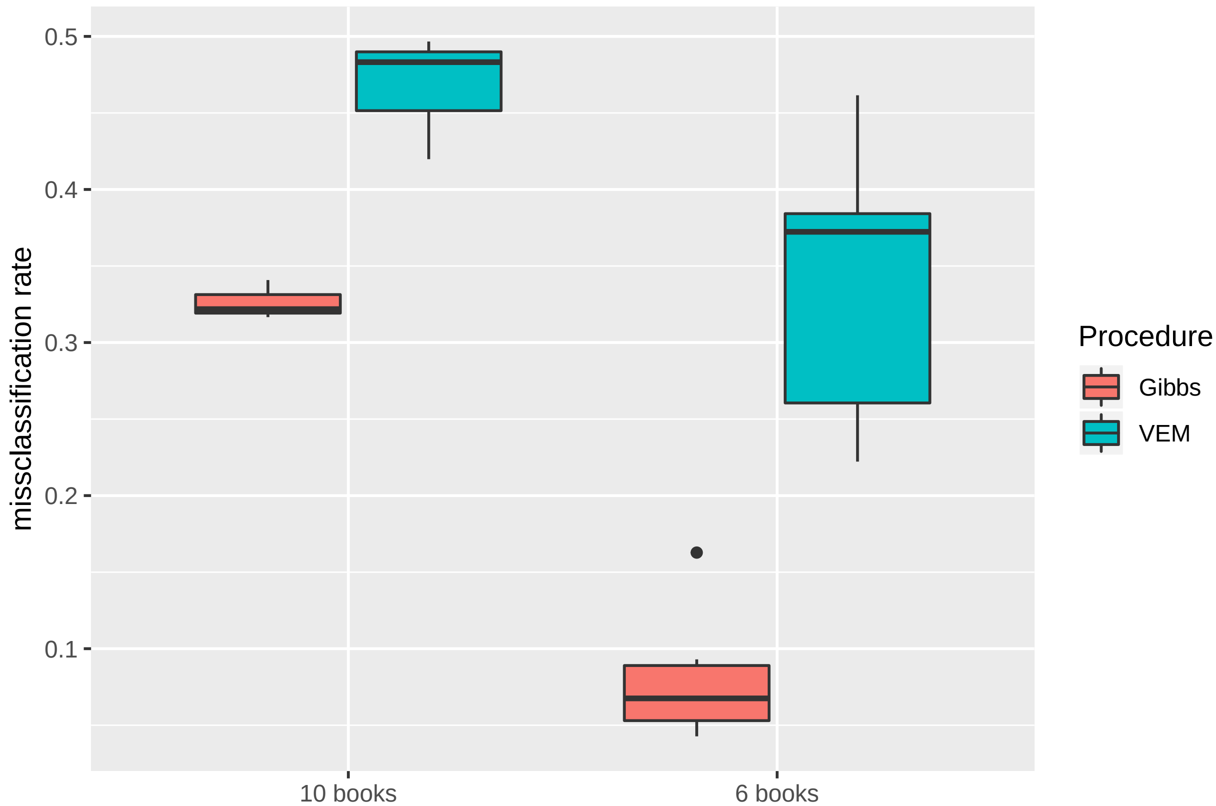
```r
# embedding 3
tim1 <- Sys.time()
chapters_lda_tfidf_10_VEM <- LDA(chapters_dtm_tfidf_10,  method="VEM",
                    k = n_books_10, control = list(seed = 1234))
tim2 <- Sys.time()
u_tfidf <- tim2-tim1

# embedding 1
misc.rate_1_10_VEM <- ext_gamma_matrix(chapters_lda_10_VEM) %>%
  validate_LDAclassification()

# embedding 2
misc.rate_all_10_VEM <- ext_gamma_matrix(chapters_lda_all_10_VEM) %>%
  validate_LDAclassification()

# embedding 3
misc.rate_tfidf_10_VEM <- ext_gamma_matrix(chapters_lda_tfidf_10_VEM) %>%
  validate_LDAclassification()

# overview
performance_matrix_10_VEM <- data.frame(freq2.embedding=c(misc.rate_1_10_VEM, u_1),
           all.embedding=c(misc.rate_all_10_VEM, u_all),
           tfidf=c(misc.rate_tfidf_10_VEM, u_tfidf))
rownames(performance_matrix_10_VEM) <- c("missc. rate", "time")
performance_matrix_10_VEM %>%  stargazer(summary=FALSE, header=F, title = "LDA via VEM 10 books example
```

Table 10: LDA via VEM 10 books example

	freq2.embedding	all.embedding	tfidf
missc. rate	0.483	0.497	0.420
time	59.268	55.789	17.245

The following box-plot should illustrate the missclassification rate of the LDA algorithm in the different use cases.

```r
plot_DF <- data.frame(VEM=c(misc.rate_1_10_VEM, misc.rate_all_10_VEM, misc.rate_tfidf_10_VEM,
                 misc.rate_1, misc.rate_all, misc.rate_tfidf,
                 misc.rate_1_sec, misc.rate_all_sec, misc.rate_tfidf_sec),
           Gibbs=c(misc.rate_1_10, misc.rate_all_10, misc.rate_tfidf_10,
                   misc.rate_1_Gibbs, misc.rate_all_Gibbs, misc.rate_tfidf_Gibbs,
                   misc.rate_1_Gibbs_sec, misc.rate_all_Gibbs_sec, misc.rate_tfidf_Gibbs_sec),
           Use_Case=c(rep("10 books",3),
                      rep("6 books",3),
                      rep("6 books",3)))
plot_DF2 <- gather(plot_DF, key="Procedure", value="accuracy", VEM, Gibbs)

plot_DF2 %>% ggplot(aes(x=Use_Case, y=accuracy, fill=Procedure))+
      geom_boxplot() + xlab("") + ylab("missclassification rate")
```



5.1 In Model Evaluation

In the first place, we try the in-model-validation procedure, for each of the 3 embeddings. Table 11 displays the results of each validation.

```
chapters_dtm2 <- convert_to_dtm(word_counts, minfq=2)
tim1 <- Sys.time()
chapters_lda_Gibbs2 <- LDA(chapters_dtm2, method="Gibbs",
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_1_Gibbs2 <- tim2-tim1
ncol(chapters_dtm2)
```

```
## [1] 8597
```

```
chapters_dtm_all2 <- convert_to_dtm(word_counts, minfq=0)
tim1 <- Sys.time()
chapters_lda_all_Gibbs2 <- LDA(chapters_dtm_all2, method = "Gibbs",
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
u_all_Gibbs2 <- tim2-tim1
M2 <- ncol(chapters_dtm_all2 )
```

```
chapters_dtm_tfidf2 <- convert_to_dtm_2(word_counts, top=0.5*M2)
tim1 <- Sys.time()
chapters_lda_tfidf_Gibbs2 <- LDA(chapters_dtm_tfidf2,  method="Gibbs",
                    k = n_books, control = list(seed = 1234))
tim2 <- Sys.time()
```

```
u_tfidf_Gibbs2 <- tim2-tim1
ncol(chapters_dtm_tfidf)
```

```
## [1] 1518
```

```
misc.rate_1_Gibbs2 <- ext_gamma_matrix(chapters_lda_Gibbs2) %>%
  validate_LDAclassification()
```

```
misc.rate_all_Gibbs2 <- ext_gamma_matrix(chapters_lda_all_Gibbs2) %>%
  validate_LDAclassification()
```

```
misc.rate_tfidf_Gibbs2 <- ext_gamma_matrix(chapters_lda_tfidf_Gibbs2) %>%
  validate_LDAclassification()
```

```
performance_matrix <- data.frame(freq2.embedding=c(misc.rate_1_Gibbs2, u_1_Gibbs2),
           all.embedding=c(misc.rate_all_Gibbs2, u_all_Gibbs2),
           tfidf=c(misc.rate_tfidf_Gibbs2, u_tfidf_Gibbs2))
rownames(performance_matrix) <- c("missc. rate", "time")
performance_matrix %>%  stargazer(summary=FALSE, header=F, title="In-Model-Validation for the 3 embeddi
```

Table 11: In-Model-Validation for the 3 embeddings

	freq2.embedding	all.embedding	tfidf
missc. rate	0.169	0.116	0.042
time	1.804	1.612	39.401