MA22019 2025 - Solutions for Problem Sheet 5

Term frequency - inverse document frequency and topic modelling

Overview

This week's problem sheet focuses on the text data analysis techniques covered in Sections 3.3.2 and 3.4 of the lecture notes. Exercises 1-2 help you with revising the content of the lecture in Week 6. You can check your solutions to these questions yourself by answering the Moodle quiz.

Tutorial Question 1 is quite extensive and will require you to apply most of the techniques from Chapter 3. Tutorial Question 2 focuses on the function grep(), which was used before, but we never explored its full potential. Should time permit, you may want to work on the Homework Question during the tutorial.

Your answer to the Homework Question can be submitted on Moodle to your tutor for feedback. The submission deadline is 17:00 on Thursday 20 March 2025. You should submit a single PDF or Word file that provides your R code, any created R output and all your comments.

You may want to load the following packages before starting the exercise:

```
library( dplyr )
library( ggplot2 )
library( tidytext )
library( stringr )
library( tidyr )
library( topicmodels )
```

When working on a University PC, you have to first install the tidytext package and any dependencies using

```
install.packages( "tidytext", dependencies = TRUE )
```

For the sentiment analysis you can load the sentiment lexicons using

```
AFINN <- read.csv("AFINN Sentiment Lexicon.csv")
Bing <- read.csv("Bing Sentiment Lexicon.csv")
```

Exercise 1 - Comparing Moby Dick and Robinson Crusoe

We want to consider the books *Moby Dick* and *The Life and Adventures of Robinson Crusoe*. The text for both books is provided in the file "AdventureBooks.csv" and we load it using

```
Books <- read.csv( "AdventureBooks.csv" )
```

Consider the following two questions:

a) Which five words (excluding stop words) are the most common in *Moby Dick*?

Let's start by extracting the lines from the book Moby Dick:

```
MobyDick_raw <- filter( Books, title == "Moby Dick" )
```

We then combine the different steps from Section 3.1.2 and extract the five most common words(excluding stop words) using the following code:

```
MobyDick_raw %>%
  unnest_tokens( word, text ) %>%
  mutate( word = gsub( "_", "", word ) ) %>%
  count( word, sort=TRUE ) %>%
  anti_join( stop_words, by="word" ) %>%
  slice_head( n=5 )
```

```
##    word    n
## 1 whale 1031
## 2  ahab  436
## 3   sea  436
## 4  ship  429
## 5    ye  426
```

We find that "whale", "ahab", "sea", "ship" and "ye" are the five most common words excluding stop words).

b) When considering a corpus which only includes *Moby Dick* and *The Life and Adventures of Robinson Crusoe*, which five words have the highest term frequency - inverse document frequency (tf-idf)?

We combine the different steps from Section 3.3.2 and extract the five words with the highest term frequency -inverse document frequency using the following code:

```
Books %>%
  unnest_tokens( word, text ) %>%
  mutate( word = gsub( "_", "", word ) ) %>%
  count( title, word, sort=TRUE ) %>%
  bind_tf_idf( word, title, n ) %>%
  arrange( desc(tf_idf) ) %>%
  slice_head( n=6 )
```

```
##             title   word    n          tf       idf       tf_idf
```

```
## 1       Moby Dick  whale 1031 0.004851536 0.6931472 0.0033628288
## 2       Moby Dick   ahab  436 0.002051668 0.6931472 0.0014221080
## 3 Robinson Crusoe friday  183 0.001506235 0.6931472 0.0010440424
## 4       Moby Dick whales  246 0.001157593 0.6931472 0.0008023820
## 5       Moby Dick  sperm  236 0.001110536 0.6931472 0.0007697649
## 6       Moby Dick  stubb  233 0.001096419 0.6931472 0.0007599797
```

The output shows that "whale", "ahab", "friday", "sperm" and "stubb" have the highest term frequency - inverse document frequency. Here we ignored the word "whales" because it's the plural of a word with higher tf-idf.

Exercise 2 - Analysis of news articles

In Section 4.4 we applied topic modelling to articles published in the New York Times. We now study another example. The file "Articles.csv" on Moodle provides the text for 2692 news articles from 2015. The articles were published either in the "business" or the "sports" category. In this exercise you will first repeat the steps from Section 4.4, and then explore how well your fitted model performs at identifying whether an article belongs to the "business" or "sports" category.

a) Treading each article as a separate document, derive the document term matrix for the set of articles and store it as **Articles_dtm**.

We start by loading the text data:

```
Articles <- read.csv( "Articles.csv" )
```

The next step is to create an identifier for each article. To aid the analysis in the next steps, we use the unite() function to create an identifier which includes the news category:

```
Articles <- Articles %>%
  mutate( ID=1:nrow(Articles) ) %>%
  unite( col=document, NewsType, ID )
```

We are now ready to remove stop words and count the number of occurrences for each word and article

```
Articles_count <- Articles  %>%
  unnest_tokens( word, Article ) %>%
  anti_join( stop_words, by="word" ) %>%
  count( document, word )
```

The last step is create the document term matrix:

```
Articles_dtm <- Articles_count %>% cast_dtm( document, word, n )
```

With the document term matrix having been derived, we estimate the parameters of an LDA model with $K = 2$ topics using:

```
Articles_LDA <- LDA( Articles_dtm, k = 2, method="Gibbs", control = list(seed=2024) )
```

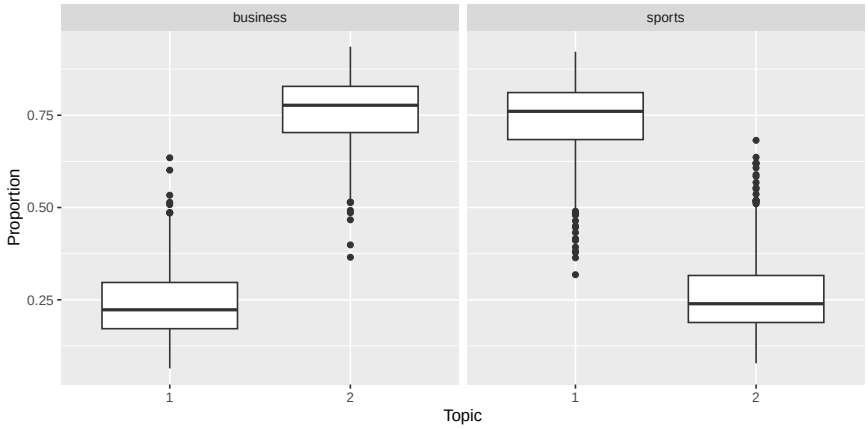
b) For each article, extract the proportions with which the different topics feature. Is there a difference in proportions between "business" and "sports" articles?

From the fitted model, we can extract the proportions, together with news category, using

```
Articles_topics <- tidy( Articles_LDA , matrix = "gamma" ) %>%
  separate( document, c("NewsType", "ID"), sep="_", convert=TRUE )
```

All that is left is to create a facet plot, which visualizes the estimated proportions for the two news categories:

```
ggplot( Articles_topics, aes( x=factor(topic), y=gamma ) ) +
  facet_wrap( ~NewsType ) + geom_boxplot() +
  labs( x="Topic", y="Proportion" )
```



The box plots indicate that the two topics tend to reflect the differences in content. More precisely, documents which predominately feature Topic 1 belong to the "sports" category, while documents mostly made up of Topic 2 fall usually into the "business" category. Consequently, there is a difference in proportions between "business" and "sports" articles

c) Which are the five most common words in each of the two topics?

We adapt the code used in Section 4.3 to extract the most common words for each category:

```
tidy( Articles_LDA , matrix = "beta" ) %>%
  group_by( topic ) %>%
  slice_max( beta, n=5 )
```

```
## # A tibble: 10 x 3
## # Groups:   topic [2]
##    topic term        beta
##    <int> <chr>      <dbl>
##  1     1 pakistan 0.00842
##  2     1 world    0.00728
```

```
##  3     1 test     0.00675
##  4     1 england  0.00659
##  5     1 cricket  0.00658
##  6     2 percent  0.0135
##  7     2 oil      0.00815
##  8     2 million  0.00595
##  9     2 prices   0.00559
## 10     2 strong   0.00533
```

Tutorial Question 1 - Comparing books

We are asked to explore and compare the books *Anne of Green Gables* by L.M. Montgomery and *Rebecca of Sunnybrook Farm* by Kate Douglas Wiggin. The two books are provided together in the file "Books Tutorial Question 1.csv" and the following information is provided:

- **text** - Text as printed in the book
- **title** - Book the text comes from
- **chapter** - Chapter the text belongs to

Perform the following tasks using the techniques described in Chapter 3 of the lecture notes.

a) Extract the two words with the highest term frequency-inverse document frequency for each book, with the corpus only containing *Anne of Green Gables* and *Rebecca of Sunnybrook Farm*.

Let's start by loading the data:

```
Books_raw <- read.csv( "Books Tutorial Question 1.csv" )
```

We first split the text into individual words and remove any underscores, before counting the number of occurrences of each word per book:

```
Books_count <- Books_raw %>%
  unnest_tokens( word, text ) %>%
  mutate( word = gsub( "\\_", "", word ) ) %>%
  count( title, word )
```

We can now use the bind_tf_idf() function to derive the tf-idf and look for the top 2 for each book:

```
Books_count %>%
  bind_tf_idf( word, title, n ) %>%
  group_by( title ) %>%
  arrange( desc(tf_idf) ) %>%
  slice_head( n=3 )
```

```
## # A tibble: 6 x 6
## # Groups:   title [2]
```

```
##   title                       word          n      tf   idf   tf_idf
##   <chr>                       <chr>     <int>   <dbl> <dbl>    <dbl>
## 1 Anne of Green Gables        anne       1102 0.0107  0.693 0.00740
## 2 Anne of Green Gables        marilla     795 0.00770 0.693 0.00534
## 3 Anne of Green Gables        diana       385 0.00373 0.693 0.00258
## 4 Rebecca of Sunnybrook Farm rebecca      571 0.00771 0.693 0.00535
## 5 Rebecca of Sunnybrook Farm rebecca's    105 0.00142 0.693 0.000983
## 6 Rebecca of Sunnybrook Farm cobb          90 0.00122 0.693 0.000843
```

The output shows that "Anne" and "Marilla" have the highest tf-idf for "Anne of Green Gables". When looking at "Rebecca of Sunnybrook Farm", "Rebecca" and "Rebecca's" should not be considered different words. As such, "Rebecca" and "Cobb" are the two words with the highest tf-idf in that book.

b) Use sentiment analysis to explore how the emotional intent has evolved over the two books. How do the two books compare?

To begin with, we require a sentiment lexicon, such as AFINN:

```
AFINN <- read.csv("AFINN Sentiment Lexicon.csv")
```

We want to consider sentiment per chapter. Before calculating the sentiment score, we need to split and clean the text data :

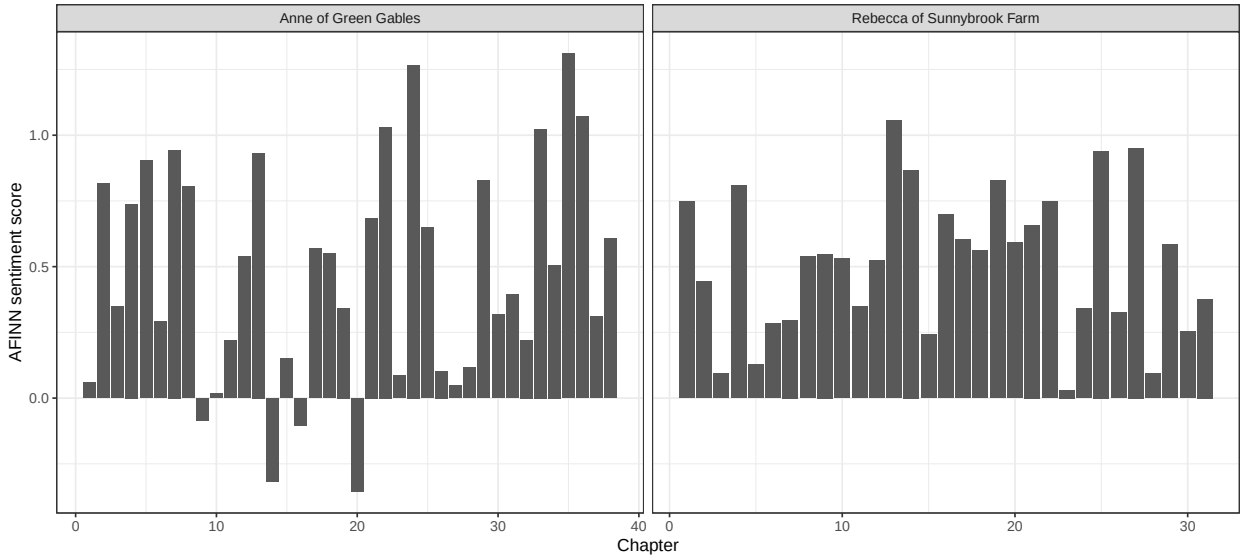
```
Books <- Books_raw %>%
  unnest_tokens( word, text ) %>%
  mutate( word = gsub( "_", "", word ) )
```

The next step is to calculate the sentiment for each chapter and book. We can adapt the code from Section 4.2 for this purpose:

```
Books <- Books %>%
  inner_join( AFINN, by="word" ) %>%
  group_by( title, chapter ) %>%
  summarise( sentiment = mean(value) )
```

Finally, we visualize the sentiment per chapter for each book:

```
ggplot( Books, aes( x=chapter, y=sentiment ) ) +
  facet_wrap( ~title, scales = "free_x" ) + geom_col() + theme_bw() +
  labs( x="Chapter", y="AFINN sentiment score" )
```



We find that both books used generally a positive sentiment in terms of the AFINN sentiment lexicon. Looking at the individual plots, it's difficult to identify a clear pattern. For "Anne of Green Gables", one may argue that the last chapters are slightly more positive than the beginning of the book, but it's not really conclusive. For "Rebbeca of Sunnybrook Farm", the plot shows a slight increase in sentiment before the book ending on a slightly less "posiitve" sentiment. Overall, apart from the general "positive" sentiment, there are little obvious similarities in terms of how the emotional intent evolves throughout the books.

c) Suppose each book chapter is considered as a separate document (as we did in Section 3.4.3 in the lecture notes). Use Latent Dirichlet Allocation to derive $K = 2$ topics, and then study the estimated proportions provided by the model. What do you conclude?

The data is already in a nice format and we can thus perform the same steps as in the analysis of the two books by Charles Dickens:

We first create the document term matrix

```
Books_chapters <- Books_raw %>%
  unite( col=document, title, chapter ) %>%
  unnest_tokens( word, text ) %>%
  mutate( word = gsub( "_", "", word ) ) %>%
  anti_join( stop_words, by="word" ) %>%
  count( document, word, sort = TRUE )
Books_dtm <- Books_chapters %>% cast_dtm( document, word, n )
```

Now we can fit the LDA model:

```
Books_LDA <- LDA( Books_dtm, k=2, method = "Gibbs", control=list(seed=123) )
```

We start by analyzing the make-up of the individual chapters:

```
tidy( Books_LDA , matrix = "gamma" ) %>%
  separate( document, c("title", "chapter"), sep="_", convert=TRUE ) %>%
```

```
  ggplot( aes( x=factor(topic), y=gamma ) ) +
  facet_wrap( ~title ) + geom_boxplot() +
  labs( x="Topic", y="Proportion" )
```

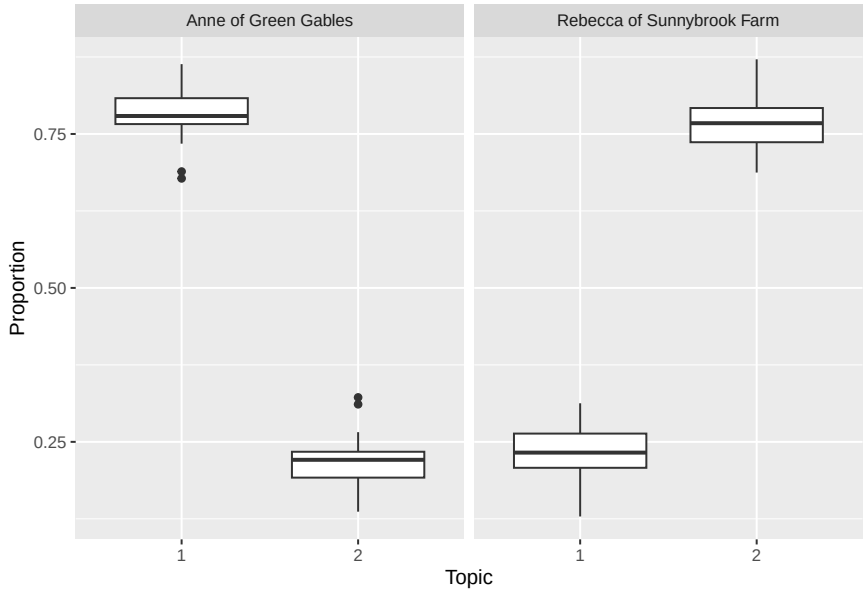


Figure 1: Illustration of the estimated proportions for each chapter within each book.

The results suggest that the fitted model does a good job at identifying which book the indiviudal chapters come from. The proportion of Topic 1 is high for chapters from "Anne of Green Gables", while Topic 2 features predominately in "Rebecca of Sunnybrook Farm".

Finally, we look at proportions for the individual words within th two topics:

```
tidy( Books_LDA , matrix = "beta" ) %>%
  mutate( topic = case_when( topic==1 ~ "Topic1", topic==2 ~ "Topic2") ) %>%
  pivot_wider( names_from = topic, values_from = beta, values_fill = 0 ) %>%
  ggplot( aes(x=Topic1, y=Topic2) ) + geom_point() +
  geom_text( aes(label=term), check_overlap = TRUE, vjust=1 ) +
  coord_trans( x="sqrt", y="sqrt" ) + theme_bw() +
  labs( x="Term Frequency in Topic 1", y="Term Frequency in Topic 2" )
```

We find a similar pattern to that in Section 4.4.3. The character names are mostly allocated to one of the two topics, while the majority of words features with a similar frequency in both topics.

d) Some scholars claim that *Anne of Green Gables* is patterned after *Rebecca of Sunnybrook Farm*. Discuss whether your results in parts a)-c) support this claim or not.

The results in part c) and e) give some support, as the clearest differences are observed for the character names, while the remaining words have a similar frequency and occur in both books. However, the differences in sentiment from part d) make this claim seem less likely.

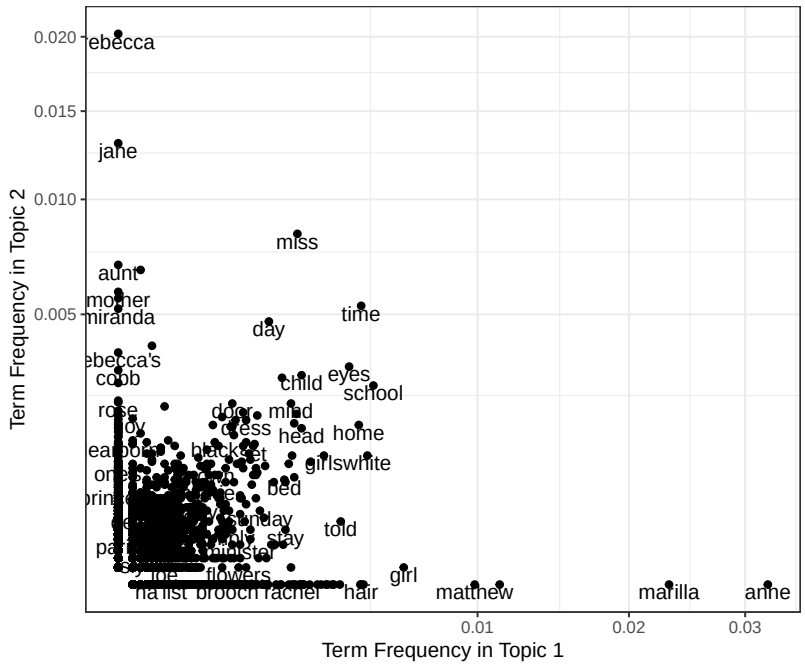


Figure 2: Comparison of term frequencies for the two topics.

So we may conclude that the books may have some similarities in language, but the stories are significantly different in terms of their sentiment.

Looking at the summaries of the two books, we find that both books desribe young girls and their coming-of-age stories, but that Anne and Rebecca are different in their personalities:

- *"Anne of Green Gables" is about Anne's personal growth, her imaginative and often dramatic nature, and the challenges she faces in building relationships, particularly with Marilla and Diana.*
- *"Rebecca of Sunnybrook Farm" focuses on Rebecca who is optimistic, and tends to bring out the best in everyone around her.*

Tutorial Question 2 - The function grep()

We so far only used the function grep() when extracting the text data from the files provided by Project Gutenberg. Specifically, we used it to identify the lines containing the word "EBOOK", signalling the beginning and end of the book. The following exercise will require you to use grep() to identify all lines which contain a specific phrase:

The Police of Utopia sent us data on burglaries which were reported between 2015 and 2021, including a short description providing information on the number of criminals and their victims. Victims are classified into six groups: "young single", "young couple", "middle-aged single", "middle-aged couple", "elderly single" and "elderly couple". The data are available in the file "UtopiaCrimes.csv". Extract the following information using the functions grep():

a) For which group of people did the police record the most burglaries?

We start by loading the date:

```
Crimes <- read.csv("UtopiaCrimes.csv")
```

Looking at the data, we see that the description states the group affected by the burglary. So we need for each group to extract the lines of text in which they are mentioned and count the number of occurrences. The function grep() extracts the indices of the lines and length() then counts them:

```
length( grep( Crimes$Description, pattern = "young single" ) )
```

```
## [1] 2126
```

```
length( grep( Crimes$Description, pattern = "young couple" ) )
```

```
## [1] 1488
```

```
length( grep( Crimes$Description, pattern = "middle-aged single" ) )
```

```
## [1] 3043
```

```
length( grep( Crimes$Description, pattern = "middle-aged couple" ) )
```

```
## [1] 2017
```

```
length( grep( Crimes$Description, pattern = "elderly single" ) )
```

```
## [1] 4410
```

```
length( grep( Crimes$Description, pattern = "elderly couple" ) )
```

```
## [1] 3429
```

The results show that the most burglaries were observed for the group "elderly single".

b) What is the proportion of burglaries that involved more than two criminals?

We need to extract the descriptions which state "Three criminals", and "More than 3 criminals". This can again be achhieved using grep() and length():

```
num_3_burglars <- length( grep( Crimes$Description, pattern = "Three Criminals" ) )
num_4_burglars <- length( grep( Crimes$Description, pattern = "More than 3 criminals" )
```

Finally, we calculate the proportion:

```
( num_3_burglars + num_4_burglars ) / nrow( Crimes )
```

```
## [1] 0.2440501
```

We find that 24.4% of burglaries were committed by a group of three or more people.

Homework Question - Sentiment Analysis vs Latent Dirichlet Allocation

So far we have used sentiment analysis to explore whether a statement has a positive or negative emotional intent. The purpose of this exercise is to apply sentiment analysis to another data set and to explore whether Latent Dirichlet Allocation (LDA) is able to identify differences in the language used for positive and negative reviews.

We will be working with customer reviews for British Airways. The reviews are stored in the file "British Airways Reviews.csv" and the following information is provided:

- **rating** - Score given by the customer (1="very poor", 10="very good")
- **country** - The country where the customer resides
- **review** - Written comment provided by the customer

Perform the following tasks using the techniques introduced in Chapter 3 of the lecture notes:

a) Derive a sentiment score for each review based on the written comments. Explore how the score you obtain compares with the numerical rating given by the customer.

b) Estimate a LDA model with $K = 2$ topics. Explore the relation between the proportions $\psi_{1,1}, \ldots, \psi_{N,1}$ and the numerical score given by the customer. In your analysis you should carefully consider which words to include in the analysis.

c) Based on your results in parts a) and b), discuss the performance of sentiment analysis and LDA in terms of identifying whether a review by a British Airways customer is more positive or more negative.

We start by loading the data and by creating an index for each review:

```
BA <- read.csv( "British Airways Reviews.csv" )
BA <- BA %>% mutate( Index=1:nrow(BA) )
```

To perform sentiment analysis, we will consider the AFINN sentiment lexicon:

```
AFINN <- read.csv("AFINN Sentiment Lexicon.csv")
```

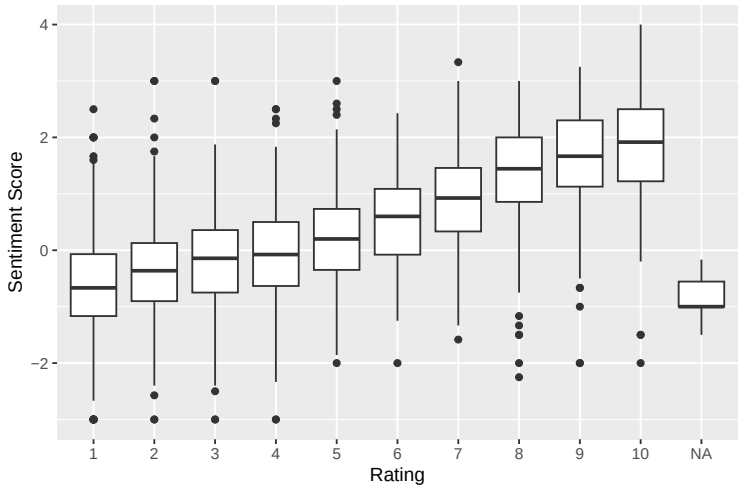
The first step is to split the reviews into individual words and assign each word its sentiment score:

```
BA_sentiment <- BA %>%
  unnest_tokens( word, review ) %>%
  inner_join( AFINN, by="word" )
```

For each review we calculate its average sentiment and use box plots to visualise the distribution of the calculated sentiment score for the different ratings:

```
BA_sentiment %>%
  group_by( factor(Index) ) %>%
  summarise( rating = rating[1], sentiment = mean( value ) ) %>%
  ggplot( aes( x=factor(rating), y=sentiment ) ) +
```

```
  geom_boxplot( ) +
  labs( x="Rating", y="Sentiment Score" )
```



We find that the calculated sentiment score increases with the rating, with reviews which received no rating recording the lowest median sentiment score. As such, the median sentiment score increases with the positivity of the review / customer satisfaction.

Let's move on the second part. The first task is to decide which words should be considered when estimating topics. While we should remove stop words when performing LDA, in this particular case we may want to keep the ones that represent emotional intent. One option is thus to define a new stop list which contains all the words from our standard stop list excluding the ones in the Bing sentiment lexicon (we use it because it inculdes more words than AFINN)

```
stop_words_sentiment <- stop_words %>% anti_join( Bing, by="word")
```

We are now ready to perform the steps required to perform LDA, beginning with deriving the document-term matrix:

```
BA_count <- BA %>%
  unnest_tokens( word, review ) %>%
  anti_join( stop_words_sentiment, by="word" ) %>%
  count( Index, word, sort = TRUE )
BA_dtm <- BA_count %>% cast_dtm( Index, word, n )
```

We can now fit the LDA model with $K = 2$ topics:

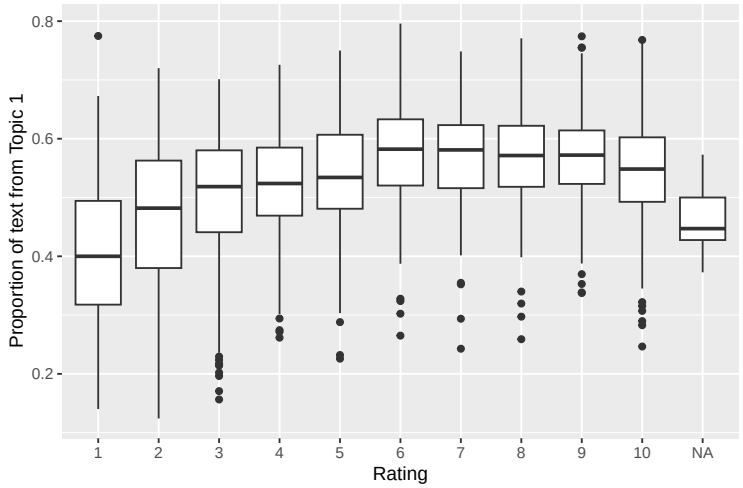
```
BA_LDA <- LDA( BA_dtm, k = 2, method = "Gibbs", control = list(seed=1234) )
```

Interest lies in the proportions $\psi_{1,1}, \ldots, \psi_{N,1}$ which we can extract using

```
BA_topics <- tidy( BA_LDA , matrix = "gamma" )
```

Finally, we visualize the relation between the proportions and the ratings using a box plot:

```
BA_topics %>%
  filter( topic == 1 ) %>%
  mutate( document = as.numeric(document) ) %>%
  full_join( BA, by = c("document"="Index") ) %>%
  ggplot( aes( x=factor(rating), y=gamma ) ) +
  geom_boxplot() + labs( x="Rating", y="Proportion of text from Topic 1" )
```



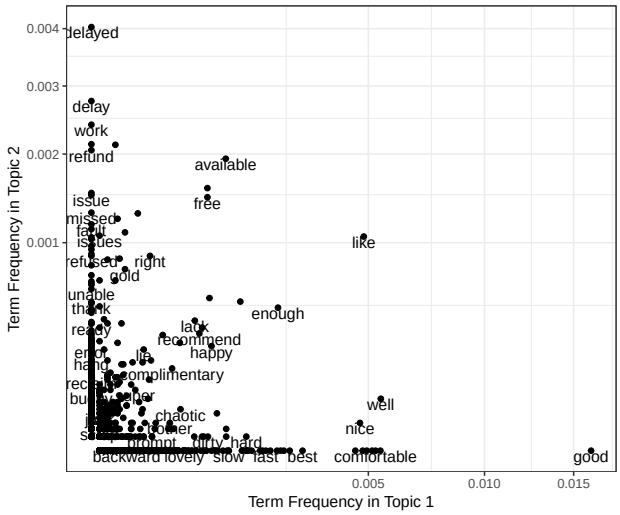
We find that reviews with a rating of 1 or 2 tend to feature less of Topic 1 than Topic 2, while the opposite applies for reviews receiving a rating of 3 or higher. The differences are small though and we see quite a variety in the proportions for all ratings, with proportions for each rating ranging between about 2-% and 80%.

When we compare the results for the sentiment analysis and LDA, we find that sentiment analysis performs best at identifying whether a review is positive or not. However, the performance is not stellar as even very positive reviews may be assigned a negative sentiment score. Consequently, the text data analysis tool box we introduced in this course can only capture the emotional intent of a customer review to a certain extent.

The weak performance of LDA is not very surprising, as there is no guarantee that the two estimated topics will be defined by whether the review uses a positive or negative wording. Reviews may use quite a few unique words, such as locations names. Let's have a look at the frequencies of the words in the Bing sentiment lexicon for the different topics:

```
tidy( BA_LDA , matrix = "beta" ) %>%
  inner_join( Bing, by=c("term"="word") ) %>%
  mutate( topic = case_when( topic==1 ~ "Topic1", topic==2 ~ "Topic2") ) %>%
  pivot_wider( names_from = topic, values_from = beta, values_fill = 0 ) %>%
  ggplot( aes(x=Topic1, y=Topic2) ) + geom_point() +
  geom_text( aes(label=term), check_overlap = TRUE, vjust=1 ) +
  coord_trans( x="sqrt", y="sqrt" ) + theme_bw() +
  labs( x="Term Frequency in Topic 1", y="Term Frequency in Topic 2" )
```

```
## Warning in inner_join(., Bing, by = c(term = "word")): Detected an unexpected many-to
## i Row 24613 of `x` matches multiple rows in `y`.
## i Row 2698 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```



We see that LDA did actually quite a good job: words we would assign with a positive sentiment are mainly in Topic 1, while words with a negative sentiment can mostly be found in Topic 2. As such, the fact that most of the estimated proportions being in the 40-60% range is down to the two topics having quite a lof of common words.