Assignment 2

Simon

13/10/2021

```
### Load standardpackages
library(tidyverse) # Collection of all the good stuff like dplyr, ggplot2 ect.
library(magrittr) # For extra-piping operators (eg. %<>%)
library(textrecipes)
library(recipes)
library(stopwords)
library(tidytext)
library(readr)
library(SnowballC)
library(topicmodels)
```

Loader datasættet

```
data <- read_csv("https://raw.githubusercontent.com/simonmig10/M2-sds/main/twitter_hate_speech.csv") %>%
  rename(X1 = ...1)
```

```
## New names:
## * `` -> ...1
```

```
## Rows: 24783 Columns: 3
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): tweet
## dbl (2): ...1, class
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
data %>% as.tibble()
```

```
## Warning: `as.tibble()` was deprecated in tibble 2.0.0.
## Please use `as_tibble()` instead.
## The signature and semantics have changed, see `?as_tibble`.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

```
## # A tibble: 24,783 x 3
##       X1 class tweet
##    <dbl> <dbl> <chr>
##  1     0     2 "!!! RT @mayasolovely: As a woman you shouldn't complain about c~
##  2     1     1 "!!!!! RT @mleew17: boy dats cold...tyga dwn bad for cuffin dat ~
##  3     2     1 "!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby4life: You ever f~
##  4     3     1 "!!!!!!!!! RT @C_G_Anderson: @viva_based she look like a tranny"
##  5     4     1 "!!!!!!!!!!!!! RT @ShenikaRoberts: The shit you hear about me mi~
##  6     5     1 "!!!!!!!!!!!!!!!!!!\"@T_Madison_x: The shit just blows me..claim~
##  7     6     1 "!!!!!!\"@__BrighterDays: I can not just sit up and HATE on anot~
##  8     7     1 "!!!!&#8220;@selfiequeenbri: cause I'm tired of you big bitches ~
##  9     8     1 "\" &amp; you might not get ya bitch back &amp; thats that \""
## 10     9     1 "\" @rhythmixx_ :hobbies include: fighting Mariam\"\n\nbitch"
## # ... with 24,773 more rows
```

1. Preprocessing

Removing retweets and removing numbers from tweets.

```
data$tweet = data$tweet %>%
  str_remove_all("[0123456789]")

data %<>%
  filter(!(tweet %>% str_detect('RT')))  %>%
  rename(ID = X1)
```

Checking that retweets got removed

```
data %>% glimpse()
```

```
## Rows: 17,615
## Columns: 3
## $ ID    <dbl> 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 2~
## $ class <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ tweet <chr> "!!!!!!!!!!!!!!!!!!\"@T_Madison_x: The shit just blows me..claim~
```

Converting data to a tibble so it can be tokenized

```
data <- tibble(ID = data[[1]] %>% as.numeric(),
               text = data[[3]] %>% as.character(),
               labels = data[[2]] %>% as.logical())
```

tokenizing the data by tweets and stemming the words, so singular and plural words become the same.

```
data_tidy <- data %>%
  unnest_tokens(word, text, token = "tweets")
```

```
## Using `to_lower = TRUE` with `token = 'tweets'` may not preserve URLs.
```

```
  #text_tokens(word, stemmer = "en") %>%
```

Checking whether it worked

```
data_tidy %>% head(50)
```

```
## # A tibble: 50 x 3
##       ID labels word
##    <dbl> <lgl>  <chr>
##  1     5 TRUE   tmadisonx
##  2     5 TRUE   the
##  3     5 TRUE   shit
##  4     5 TRUE   just
##  5     5 TRUE   blows
##  6     5 TRUE   meclaim
##  7     5 TRUE   you
##  8     5 TRUE   so
##  9     5 TRUE   faithful
## 10     5 TRUE   and
## # ... with 40 more rows
```

It did, so now we count the words to check the most used words

```
data_tidy %>% count(word, sort = TRUE)
```

```
## # A tibble: 26,333 x 2
##    word      n
##    <chr> <int>
##  1 a      6352
##  2 bitch  5932
##  3 i      5499
##  4 the    5004
##  5 you    4100
##  6 to     3633
##  7 and    2803
##  8 my     2645
##  9 that   2536
## 10 in     2103
## # ... with 26,323 more rows
```

It shows that irrelevant words like “i” and “a” are commonly present, so these needs to be removed.

Hashtags are being removed and so are other typical twitter specific stuff like http and so on. Words shorter than 3 letters are also removed and words occuring less than 100 times. Lastly the data is antijoined with stopwords and the tidying proces is done.

```
# preprocessing
data_tidy %<>%
  filter(!(word %>% str_detect('@'))) %>% # remove hashtags and mentions
  filter(!(word %>% str_detect('^amp|^http|^t\\.co'))) %>% # Twitter specific stuff
#  mutate(word = word %>% str_remove_all('[^[:alnum:]]')) %>% ## remove all special characters
  filter(str_length(word) > 2 ) %>% # Remove words with less than  3 characters
```

```
  group_by(word) %>%
  filter(n() > 100) %>% # remove words occuring less than 100 times
  ungroup() %>%
  anti_join(stop_words, by = 'word') %>%  # remove stopwords
  mutate(word = wordStem(word))
```

TF IDF'

The TF_IDF score of each word is now being added to the tidied data. The TF-IDF (term frequency-inverse document frequency) is a measure that evaluates how relevant a word is to a document in a collection of documents.

This is done by multiplying two metrics: how many times a word appears in a document, and the inverse document frequency of the word across a set of documents.

```
# TFIDF weights
data_tidy %<>%
  add_count(ID, word) %>%
  bind_tf_idf(term = word,
              document = ID,
              n = n)
```

Now the scores are showed.

```
# TFIDF topwords
data_tidy %>%
  count(word, wt = tf_idf, sort = TRUE) %>%
  head(10)
```

```
## # A tibble: 10 x 2
##    word      n
##    <chr> <dbl>
##  1 bitch 3012.
##  2 hoe   2595.
##  3 pussi 1950.
##  4 trash 1449.
##  5 fuck  1296.
##  6 dont  1062.
##  7 ass   1005.
##  8 nigga  990.
##  9 bird   985.
## 10 lol    932.
```

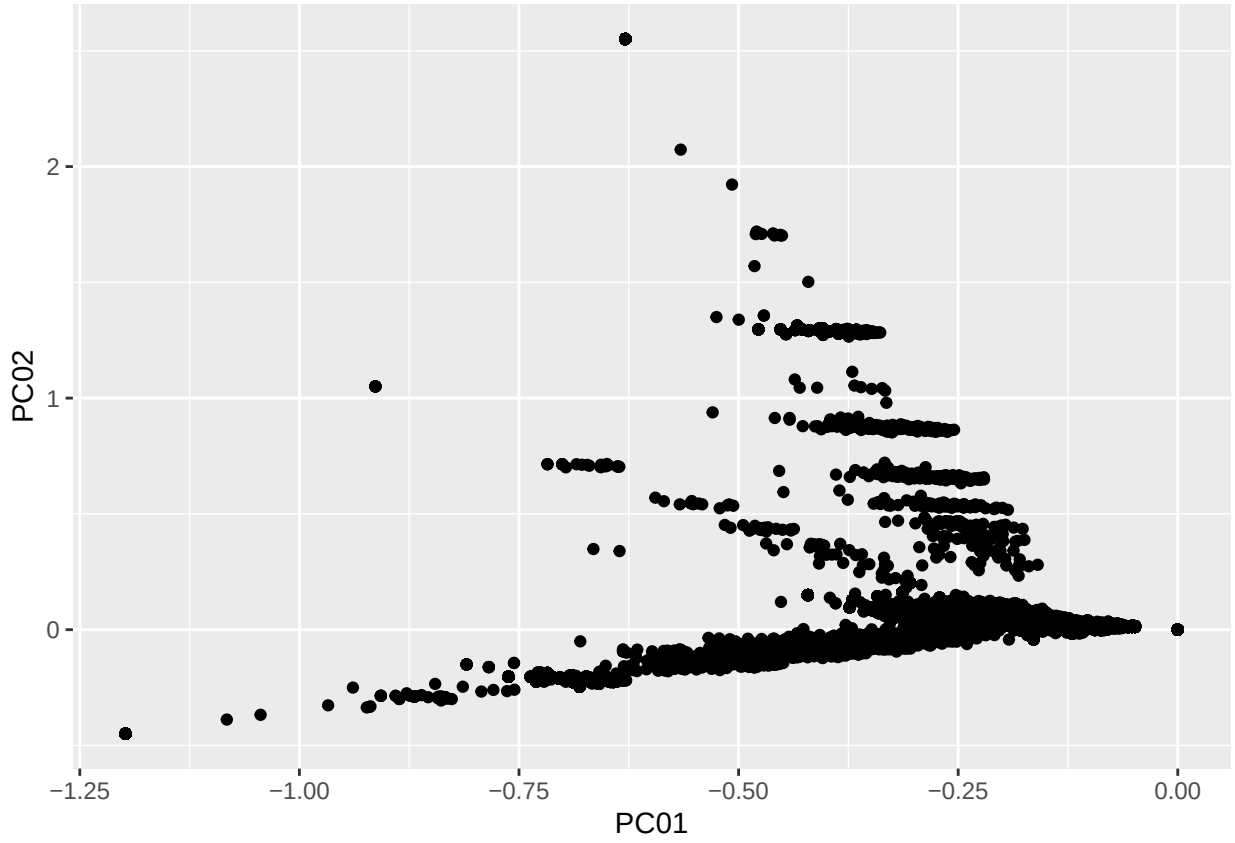
And here the ten words with the highest TF_IDF displayed.

Dimensionality reduction

Creating a recipe

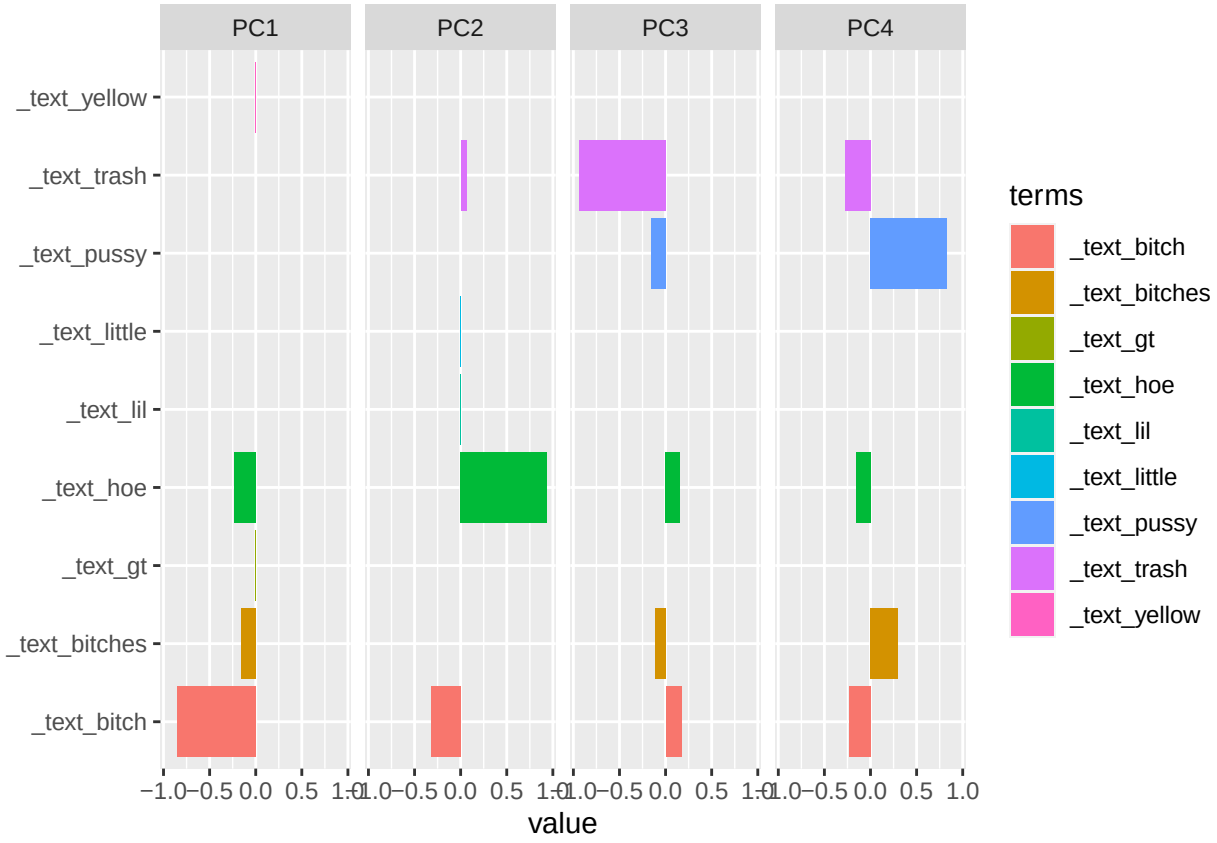
A recipe is created to dimensionality reduce the data and weighting them by TD-IDF scores.

```r
recipe_base <- data %>%
  select(ID, text) %>%
  # BAse recipe starts
  recipe(~.) %>%
  update_role(ID, new_role = "ID") %>% # Update role of ID
  step_tokenize(text, token = 'words') %>% # tokenize
  step_stopwords(text, keep = FALSE) %>% # remove stopwords
  step_untokenize(text) %>% # Here we now have to first untokenize
  step_tokenize(text, token = "ngrams", options = list(n = 1, n_min = 1)) %>% # and tokenize again
  step_tokenfilter(text, min_times = 25) %>%
  prep()

recipe_pca <- recipe_base %>% # tokenize
  step_tfidf(text, prefix = '') %>% # TFIDF weighting --> so different from the above.
  step_pca(all_predictors(), num_comp = 10) %>% # PCA
  prep()

#Plot 1
recipe_pca %>% juice() %>%
  ggplot(aes(x = PC01, y = PC02)) +
  geom_point()
```




```r
#Plot 2
recipe_pca %>%
  tidy(7) %>%
```

```
  filter(component %in% paste0("PC", 1:4)) %>%
  group_by(component) %>%
    arrange(desc(value)) %>%
    slice(c(1:2, (n()-2):n())) %>%
  ungroup() %>%
  mutate(component = fct_inorder(component)) %>%
  ggplot(aes(value, terms, fill = terms)) +
  geom_col(show.legend = TRUE) +
  facet_wrap(~component, nrow = 1) +
  labs(y = NULL)
```



These plots are not just awesome they also show how well some of the highest scored words are explained by each principal component.

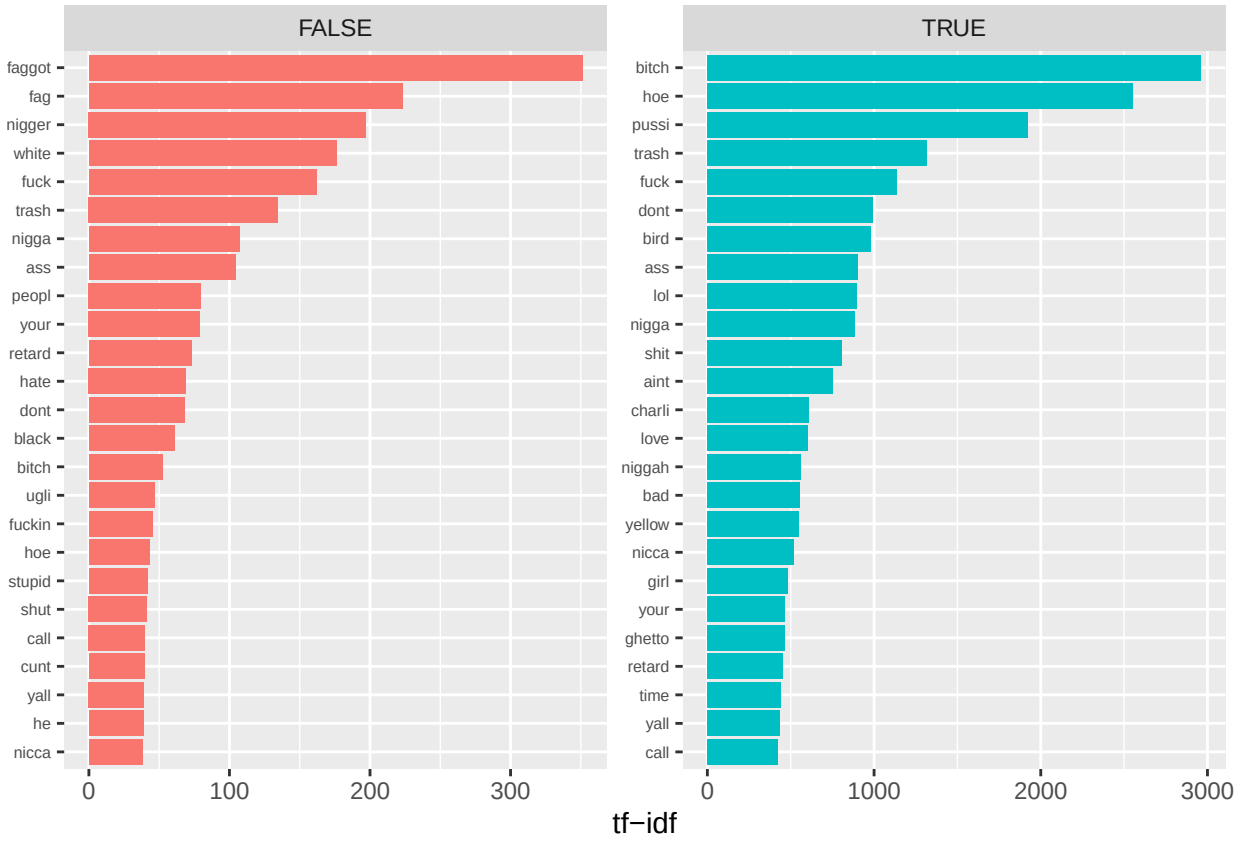
2. Explore and compare the 2 "classes of interest" - hate speech vs offensive language

Can you see differences by using simple count-based approaches?

```
labels_words <- data_tidy %>%
  group_by(labels) %>%
  count(word, wt = tf_idf, sort = TRUE, name = "tf_idf") %>%
```

```
  slice(1:25) %>%
  ungroup()
```

```
labels_words %>%
  mutate(word = reorder_within(word, by = tf_idf, within = labels)) %>%
  ggplot(aes(x = word, y = tf_idf, fill = labels)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "tf-idf") +
  facet_wrap(~labels, ncol = 2, scales = "free") +
  coord_flip() +
  scale_x_reordered() +
  theme(axis.text.y = element_text(size = 6))
```



The result shows that the words with the highest TD-IDF score in the hate speech basket is “faggot”, “fag” and “nigger” where as the words with the highest TD-IDF in the offensive language basket is “bitch”, “hoe” and “pussi”. This shows that hate speech is associated with racism and homophobia, where as offensive language is more cussing and bad mouthing.

##Can you identify themes (aka clusters / topics) that are specific for one class or another?

LDA-topic modelling is used to separate the dataset into topics for hate speech and offensive language respectively.

First the tidy data is extracted and split into two datasets

```
lda_data = data_tidy %>%
  select(ID, labels, word, n, tf_idf)
```

```
off = lda_data %>%
  filter(labels == TRUE) %>%
  as.tibble()

hate = lda_data %>%
  filter(labels == FALSE) %>%
  as.tibble()
```

Then the data is made into a document-term matrix

```
text_dtm1 <- hate %>%
  cast_dtm(document = ID, term = word, value = n)

text_dtm2 <- off %>%
  cast_dtm(document = ID, term = word, value = n)
```

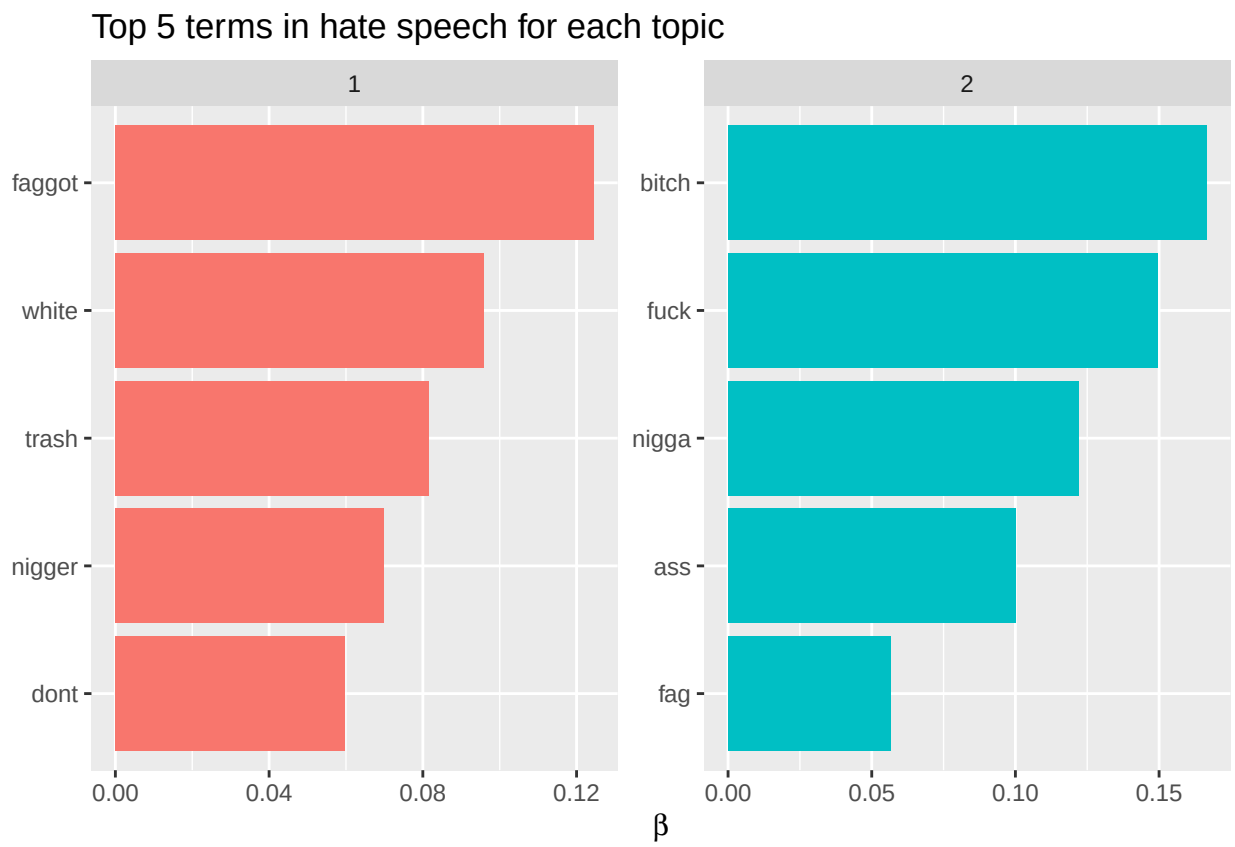
The topics are created using the "Gibbs" method and there are only being created two topics for each category

```
text_lda1 <- text_dtm1 %>%
  LDA(k = 2, method = "Gibbs",
      control = list(seed = 1337))

text_lda2 <- text_dtm2 %>%
  LDA(k = 2, method = "Gibbs",
      control = list(seed = 1337))
```

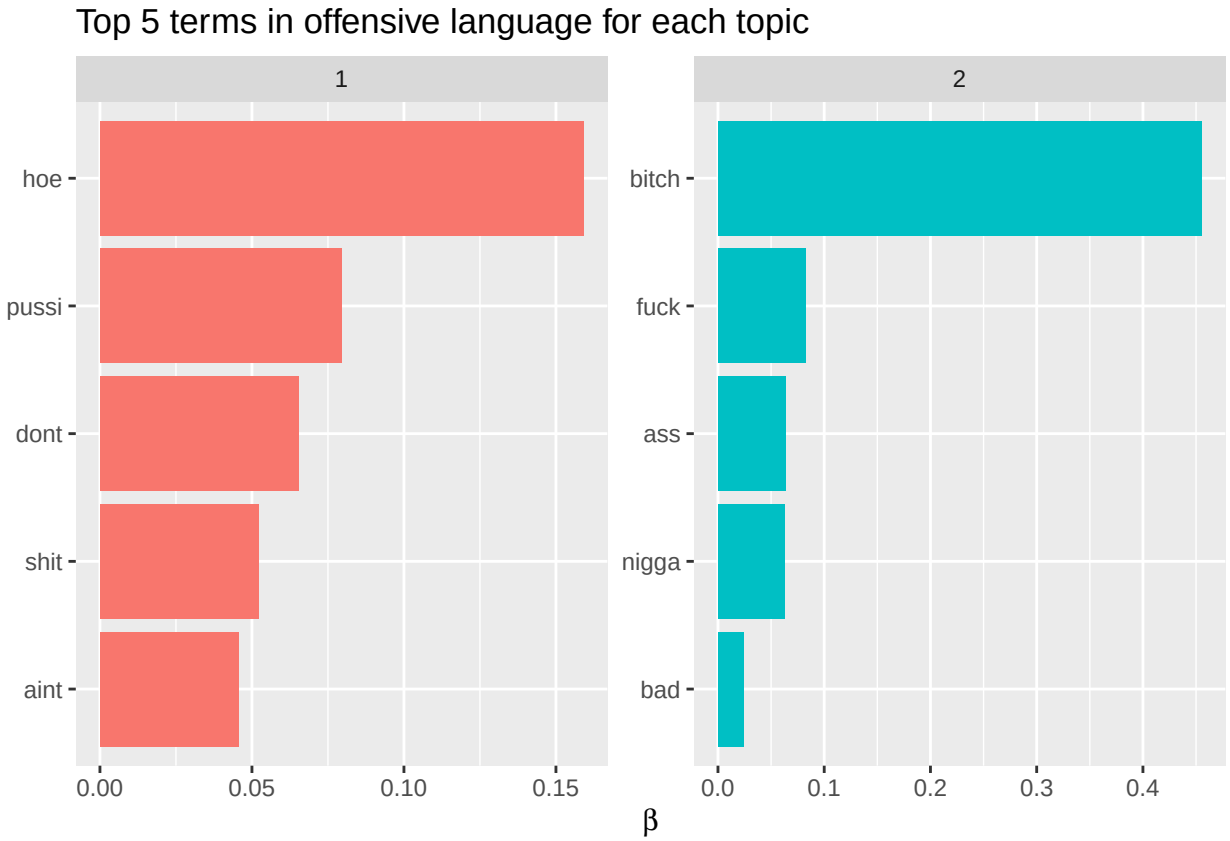
We want the Beta parameter as i shows the probability that a word occurs in a certain topic. Therefore, looking at the top probability words of a topic often gives us a good intuition regarding its properties, which is done in two plots below.

```
lda_beta1 <- text_lda1 %>%
  tidy(matrix = "beta")

lda_beta2 <- text_lda2 %>%
  tidy(matrix = "beta")

lda_beta1 %>%
  # slice
  group_by(topic) %>%
  arrange(topic, desc(beta)) %>%
  slice(1:5) %>%
  ungroup() %>%
  # visualize
  mutate(term = reorder_within(term, beta, topic)) %>%
  group_by(topic, term) %>%
  arrange(desc(beta)) %>%
  ungroup() %>%
  ggplot(aes(term, beta, fill = as.factor(topic))) +
  geom_col(show.legend = FALSE) +
  coord_flip() +
  scale_x_reordered() +
```

```
  labs(title = "Top 5 terms in hate speech for each topic",
       x = NULL, y = expression(beta)) +
  facet_wrap(~ topic, ncol = 3, scales = "free")
```




```
lda_beta2 %>%
  # slice
  group_by(topic) %>%
  arrange(topic, desc(beta)) %>%
  slice(1:5) %>%
  ungroup() %>%
  # visualize
  mutate(term = reorder_within(term, beta, topic)) %>%
  group_by(topic, term) %>%
  arrange(desc(beta)) %>%
  ungroup() %>%
  ggplot(aes(term, beta, fill = as.factor(topic))) +
  geom_col(show.legend = FALSE) +
  coord_flip() +
  scale_x_reordered() +
  labs(title = "Top 5 terms in offensive language for each topic",
       x = NULL, y = expression(beta)) +
  facet_wrap(~ topic, ncol = 3, scales = "free")
```



The two above plots kinda shows a similar picture as the one showcased earlier. It seems like the two topics are not separated by any particular measure such as racism and homophobia in the case of hate speech nor seems the topic in offensive language to be separated by anything specific.