Text Analytics

Anshu Singh

June 7, 2020

Getting the dataset

```
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 3.5.3
```

```
getwd()
```

```
## [1] "C:/Users/anshu/Desktop/UNCC Courses/Advanced Business Analytics/Homework2 - Text analysis, topic modelling, sentimen
tal analysis"
```

```
setwd("C:/Users/anshu/Desktop/UNCC Courses/Advanced Business Analytics/Homework2 - Text analysis, topic modelling, sentiment
al analysis")

data <- fread("psychcentral_data.csv", sep=",", header=T, strip.white = T, na.strings = c("NA","NaN","","?"))

colnames(data)
```

```
## [1] "row"       "q_subject" "q_content" "answers"
```

The four columns here are :

1. rows -> which is nothing but the index
2. q_subject -> headline of the topic discussed
3. q_content -> explanation on the topic given in q_subject
4. answers -> reply back from people on their topic discussion.

Looking inside the subject and the topics discussed on the forum.

```
row2 = data$q_subject
head(row2)
```

```
## [1] "Saying Goodbye For Now"
## [2] "Im really afraid of going to school"
## [3] "jealousy filled hatred"
## [4] "Is my friend stuck in a fantasy world"
## [5] "I have mind problems or something weird pls read and help"
## [6] "Shed a Light of Hope"
```

```
library("tidytext")
```

```
## Warning: package 'tidytext' was built under R version 3.5.3
```

```
library("dplyr")
```

```
## Warning: package 'dplyr' was built under R version 3.5.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Pre-processing Data for Analysis

Toeknization - A concept of breaking paragraphs and sentences into words.

After applying the unnest_tokens() for the tokenization, we will remove the stop words which is nothing but removing ‘a’, ‘an’, ‘the’, ‘that’ from our text since they does not add any value to our information. After that, I am counting the frquency of the words used in our texts to get an idea of what are the topics discussed.

Visualizing using ‘ggplot’ is important to get deeper knowledge of word count that is used and we can filter that out on the base of our requirements. Here I am using anything greater than 100 to pop up in my visualization.

Wordcloud - A powerful way of presenting the words that anybody can visualize and analyse the most used words which uses size and boldness to differentitate the word occurence.

Stemming - It’s a concept of removing all the similar occuring words with their root words. For example - converting believes, believed, believing into believe

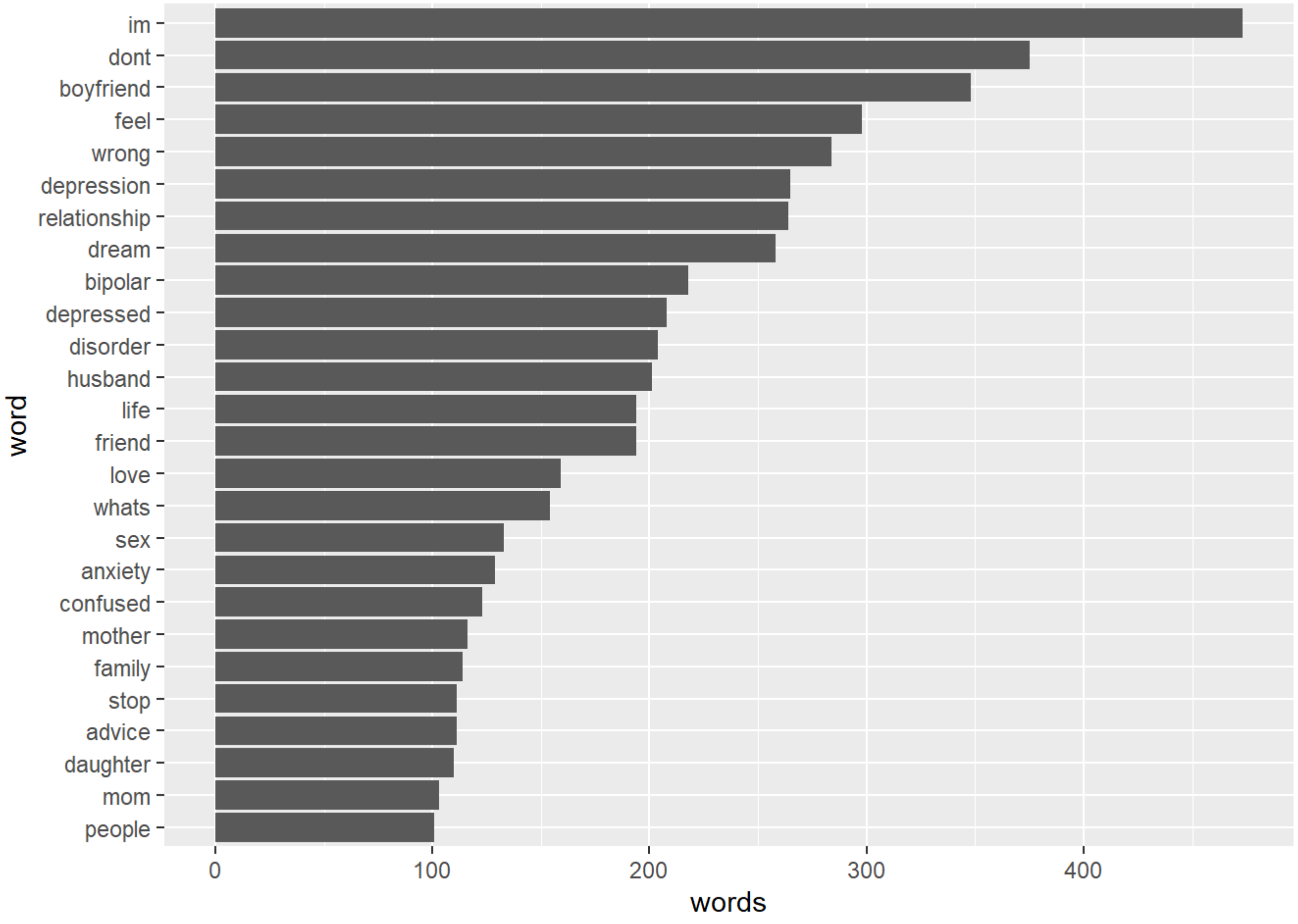
```
#Tokenizing
tidy_text <- data %>%
  unnest_tokens(word, q_subject)


tidy_text <- tidy_text %>%
  anti_join(stop_words)
```

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

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

```
## # A tibble: 4,829 x 2
##    word             n
##    <chr>        <int>
##  1 im             473
##  2 dont           375
##  3 boyfriend      348
##  4 feel           298
##  5 wrong          284
##  6 depression     265
##  7 relationship   264
##  8 dream          258
##  9 bipolar        218
## 10 depressed      208
## # ... with 4,819 more rows
```

```
library("ggplot2")

tidy_text %>%
  count(word, sort = TRUE) %>%
  filter(n > 100) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n,word)) +
  geom_bar(stat = "identity") +
  xlab("words")
```



```
library("wordcloud")
```

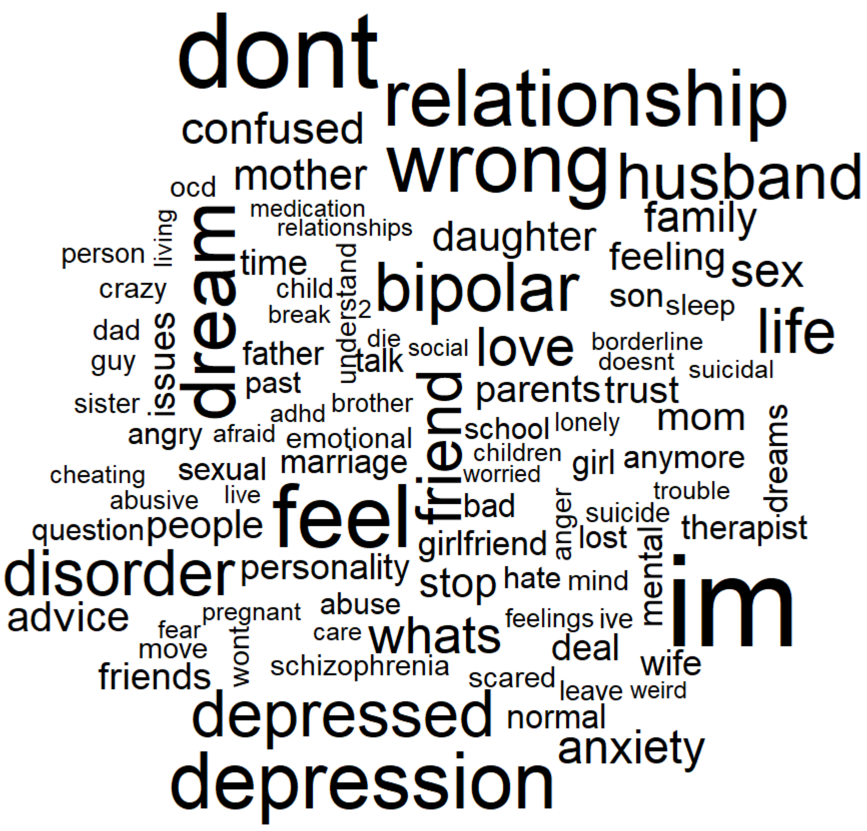
```
## Warning: package 'wordcloud' was built under R version 3.5.3
```

```
## Loading required package: RColorBrewer
```

```
tidy_text %>%
  anti_join(stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 100))
```

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

```
## Warning in wordcloud(word, n, max.words = 100): boyfriend could not be fit
## on page. It will not be plotted.
```



```
library("SnowballC")
library("tidytext")
library("dplyr")

#Stemming
tidy_text <- data %>%
  unnest_tokens(word, q_subject) %>%
  mutate(word = wordStem(word))

tidy_text <- tidy_text %>%
  anti_join(stop_words)
```

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

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

```
## # A tibble: 3,798 x 2
##    word             n
##    <chr>        <int>
##  1 thi            499
##  2 depress        490
##  3 im             473
##  4 feel           446
##  5 dont           375
##  6 boyfriend      373
##  7 dream          349
##  8 doe            346
##  9 relationship   299
## 10 friend         287
## # ... with 3,788 more rows
```

```
library("ggplot2")
library("dplyr")
library("tidyverse")
```

```
## Warning: package 'tidyverse' was built under R version 3.5.3
```

```
## -- Attaching packages ---------------------------------------------------------------------------------------------------
---------------------------------------- tidyverse 1.2.1 --
```

```
## v tibble  3.0.1     v purrr   0.3.4
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## Warning: package 'tibble' was built under R version 3.5.3
```

```
## Warning: package 'tidyr' was built under R version 3.5.3
```

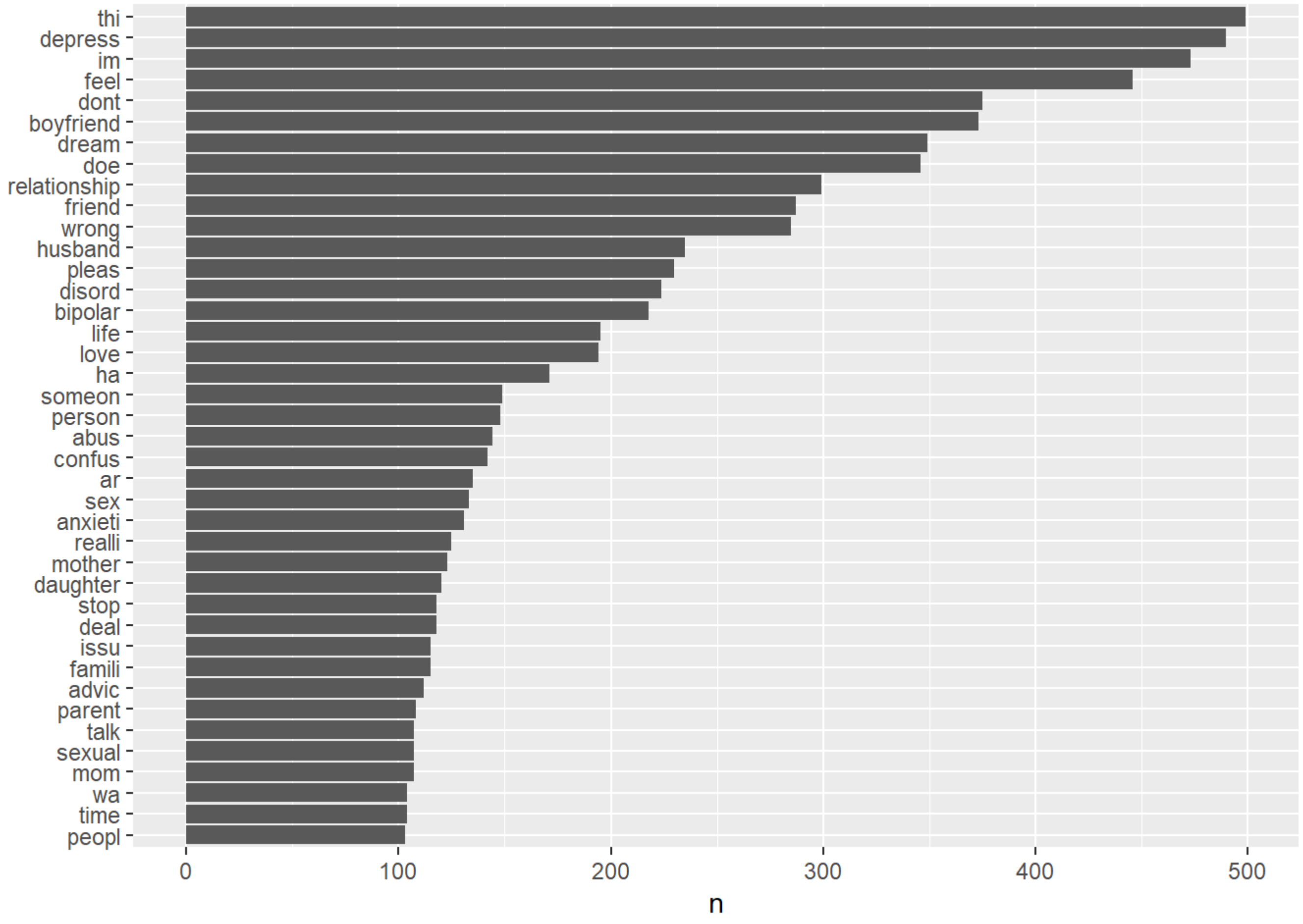
```
## Warning: package 'readr' was built under R version 3.5.3
```

```
## Warning: package 'purrr' was built under R version 3.5.3
```

```
## Warning: package 'forcats' was built under R version 3.5.3
```

```
## -- Conflicts -------------------------------------------------------------------------------------------------------
---------------------------------- tidyverse_conflicts() --
## x dplyr::between()   masks data.table::between()
## x dplyr::filter()    masks stats::filter()
## x dplyr::first()     masks data.table::first()
## x dplyr::lag()       masks stats::lag()
## x dplyr::last()      masks data.table::last()
## x purrr::transpose() masks data.table::transpose()
```

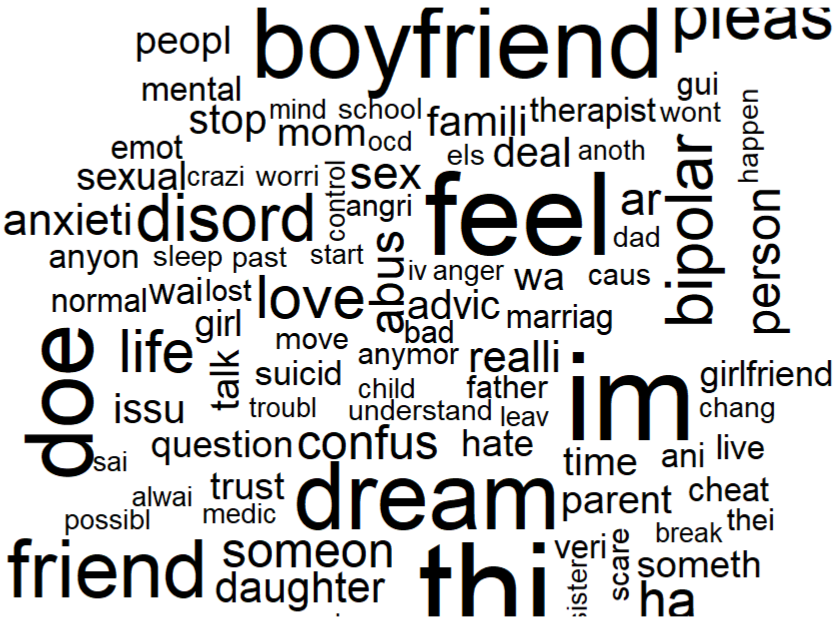
```
tidy_text %>%
  count(word, sort = TRUE) %>%
  filter(n > 100) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_bar(stat = "identity") +
  xlab(NULL) +
  coord_flip()
```



```
library("wordcloud")

tidy_text %>%
  anti_join(stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 100))
```

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

```
## Warning in wordcloud(word, n, max.words = 100): depress could not be fit on
## page. It will not be plotted.
```



Sentimental Analysis is giving emotion to the word - lets say positive words and negative words.

Here, I am using the world cloud to segregate the positive and negative emotions so that we don't have to manually seperate if from the previous wordcloud.

```
#Sentiment Analysis
library("reshape2")
```

```
## Warning: package 'reshape2' was built under R version 3.5.3
```

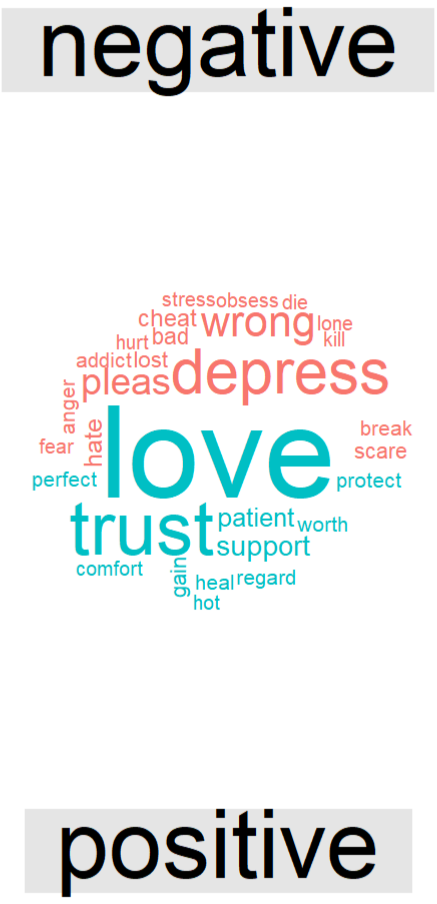
```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
## The following objects are masked from 'package:data.table':
##
##     dcast, melt
```

```
tidy_text %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("#F8766D", "#00BFC4"),
                 max.words = 30)
```

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



Now, since we have pretty good idea about the subject of the discussion, lets see if we can find something different in their explanation and discussion.

For that, i am using ‘q_content’ column from my dataset.

```
tidy_text <- data %>%
  unnest_tokens(word, q_content)
```

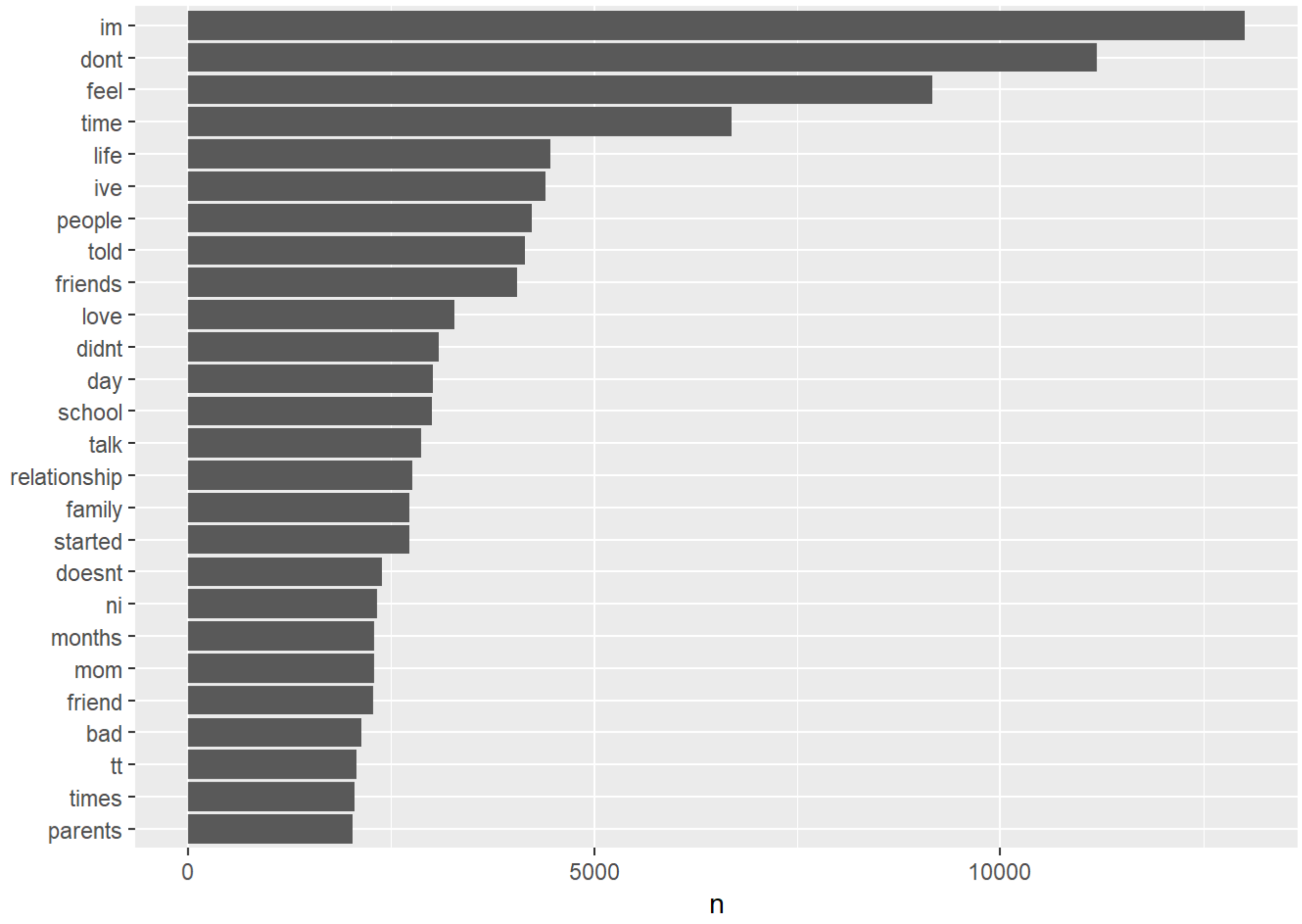
```
data(stop_words)

tidy_text <- tidy_text %>%
  anti_join(stop_words)
```

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

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

```
## # A tibble: 46,081 x 2
##    word        n
##    <chr>   <int>
##  1 im      13012
##  2 dont    11197
##  3 feel     9168
##  4 time     6697
##  5 life     4464
##  6 ive      4403
##  7 people   4233
##  8 told     4150
##  9 friends  4045
## 10 love     3281
## # ... with 46,071 more rows
```

```
library("ggplot2")


#visualizing word that appear more than 2000 times
tidy_text %>%
  count(word, sort = TRUE) %>%
  filter(n > 2000) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_bar(stat = "identity") +
  xlab(NULL) +
  coord_flip()
```



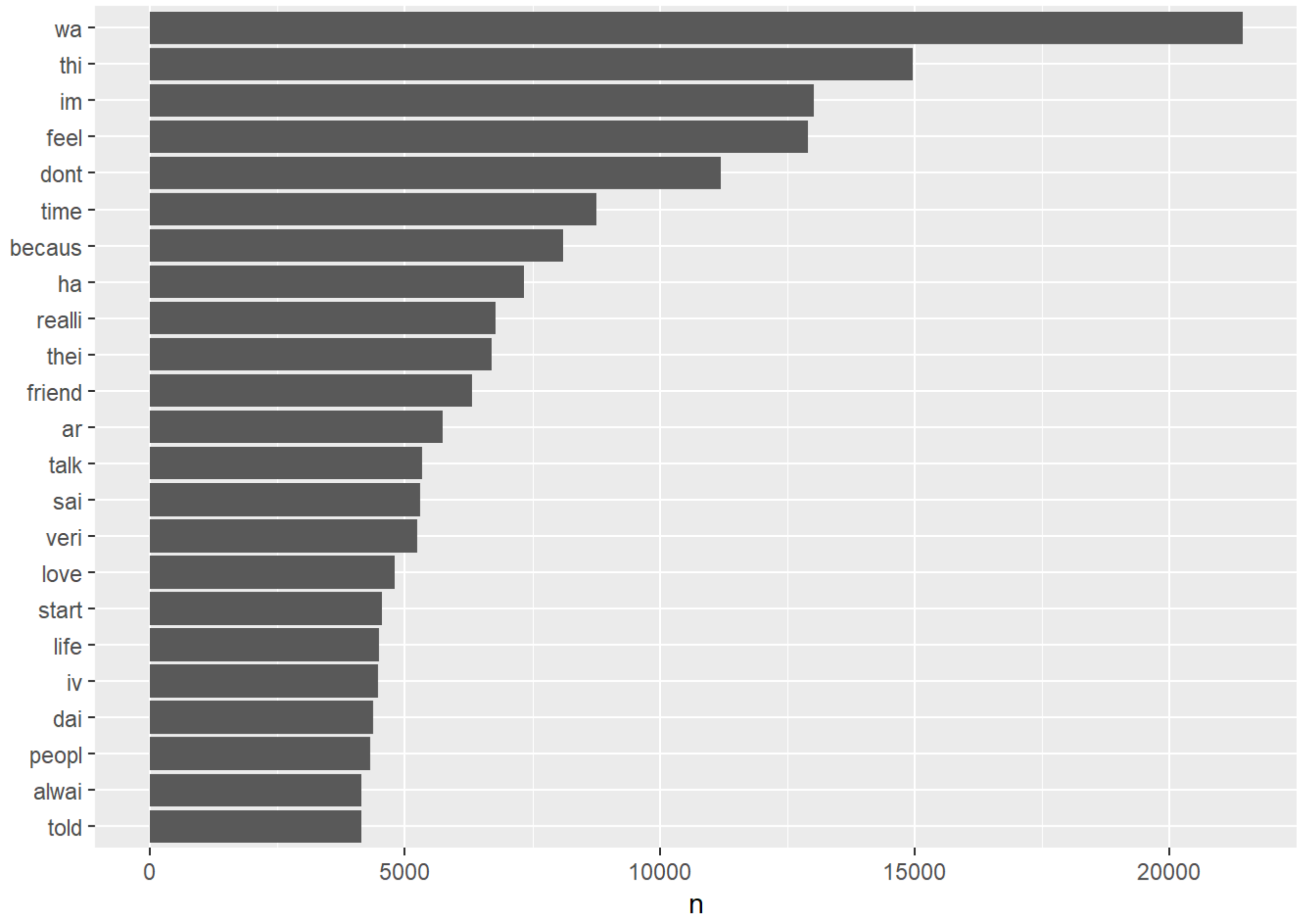
```
library("SnowballC")

#Stemming
tidy_text <- data %>%
  unnest_tokens(word, q_content) %>%
  mutate(word = wordStem(word))

data(stop_words)

tidy_text <- tidy_text %>%
  anti_join(stop_words)
```

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

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

```
## # A tibble: 36,404 x 2
##    word       n
##    <chr>  <int>
##  1 wa     21437
##  2 thi    14961
##  3 im     13016
##  4 feel   12905
##  5 dont   11197
##  6 time    8755
##  7 becaus  8104
##  8 ha      7340
##  9 realli  6780
## 10 thei    6698
## # ... with 36,394 more rows
```

```
#visualizing words
tidy_text %>%
  count(word, sort = TRUE) %>%
  filter(n > 4000) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_bar(stat = "identity") +
  xlab(NULL) +
  coord_flip()
```



```
sentiment <- tidy_text %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE)
```

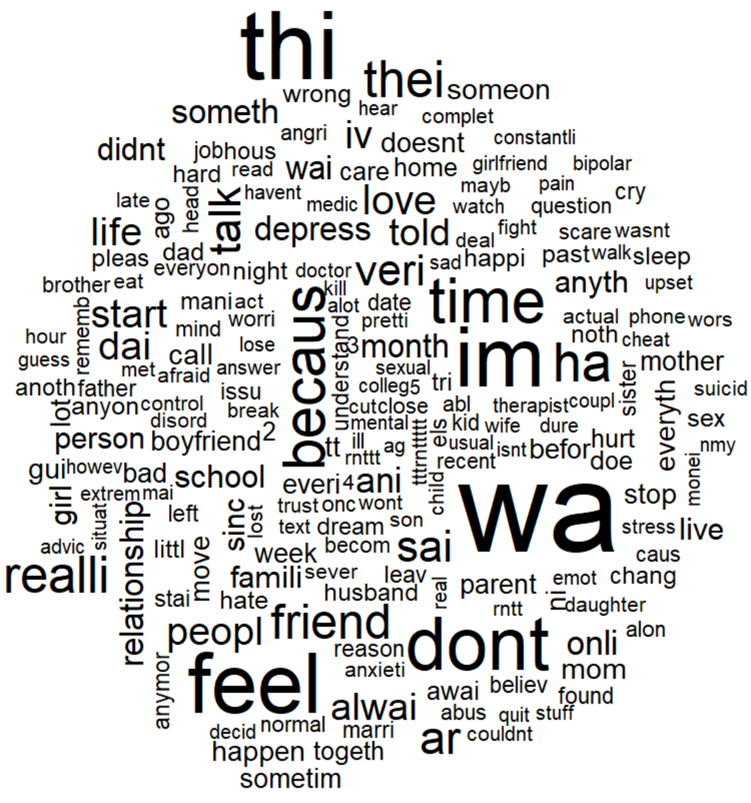
```
## Joining, by = "word"
```

```
head(sentiment)
```

```
## # A tibble: 6 x 3
##   word    sentiment     n
##   <chr>   <chr>     <int>
## 1 love    positive   4801
## 2 depress negative   3375
## 3 bad     negative   2133
## 4 hurt    negative   2000
## 5 wrong   negative   1715
## 6 hate    negative   1691
```

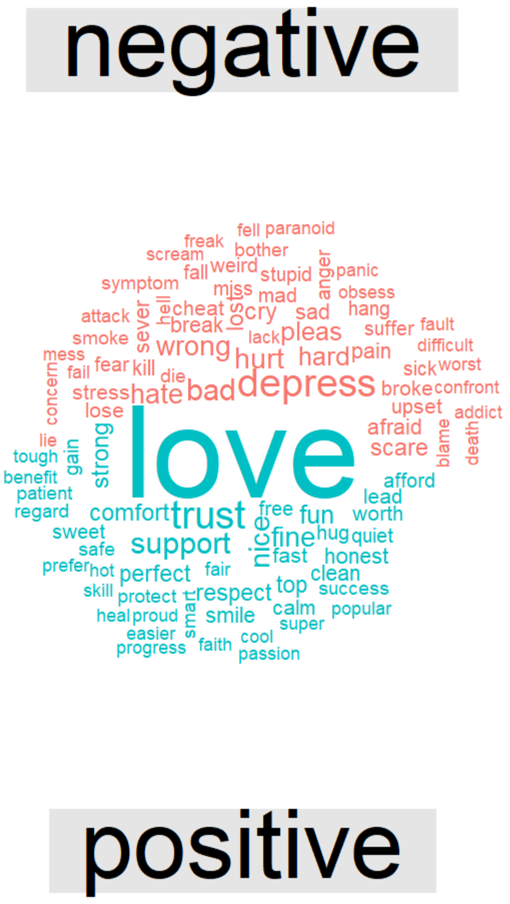
```
library("wordcloud")
tidy_text %>%
  anti_join(stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 200))
```

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



```
library("reshape2")
tidy_text %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("#F8766D", "#00BFC4"),
                   max.words = 100)
```

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



Now, working on the Answer column to find some insightful reply on the subject discussion.

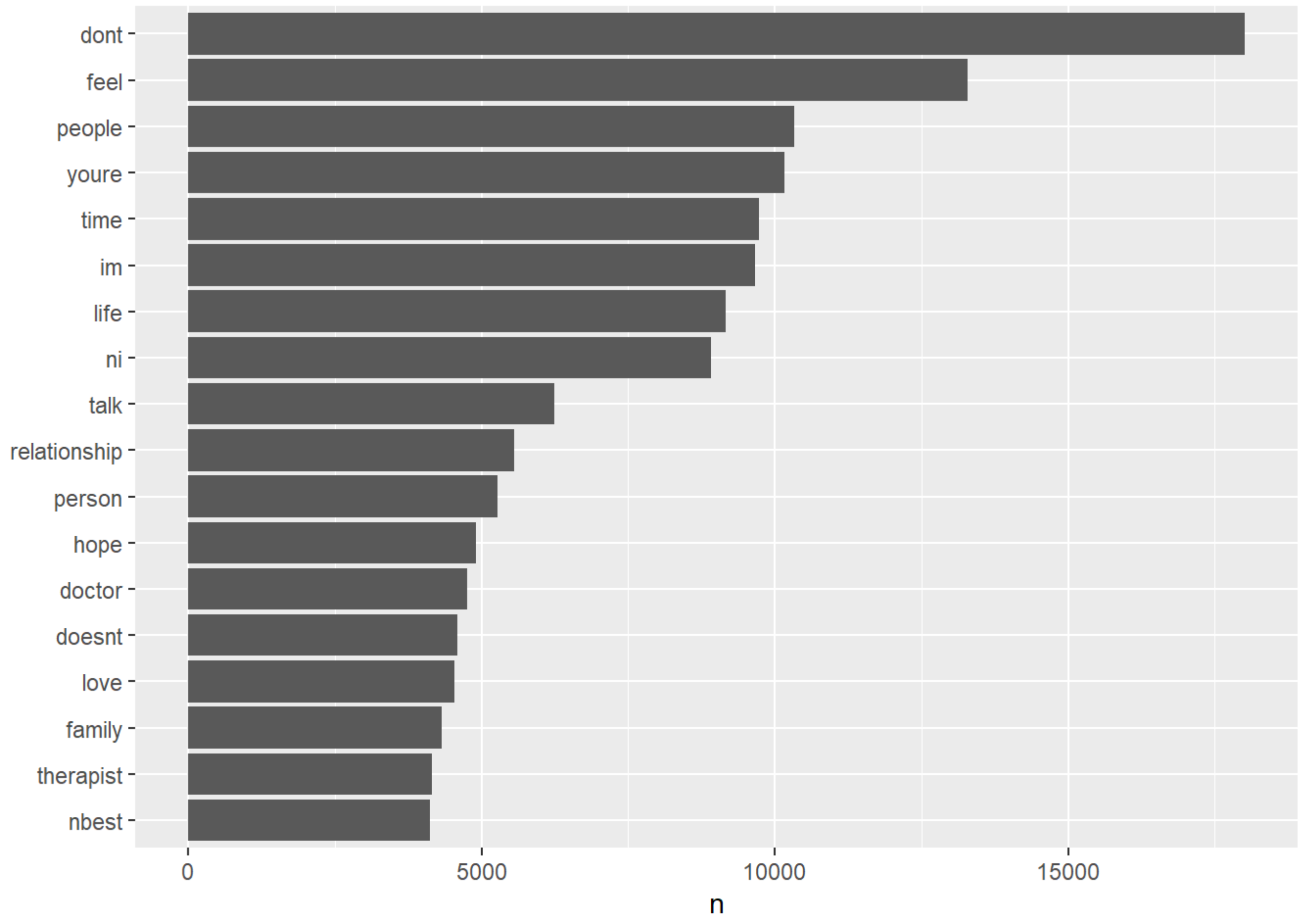
```
tidy_text <- data %>%
  unnest_tokens(word, answers)

tidy_text <- tidy_text %>%
  anti_join(stop_words)
```

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

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

```
## # A tibble: 54,645 x 2
##    word             n
##    <chr>        <int>
##  1 dont         18010
##  2 feel         13279
##  3 people       10334
##  4 youre        10162
##  5 time          9729
##  6 im            9664
##  7 life          9169
##  8 ni            8913
##  9 talk          6245
## 10 relationship  5557
## # ... with 54,635 more rows
```

```
tidy_text %>%
  count(word, sort = TRUE) %>%
  filter(n > 4000) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_bar(stat = "identity") +
  xlab(NULL) +
  coord_flip()
```



```
sentiment <- tidy_text %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE)
```

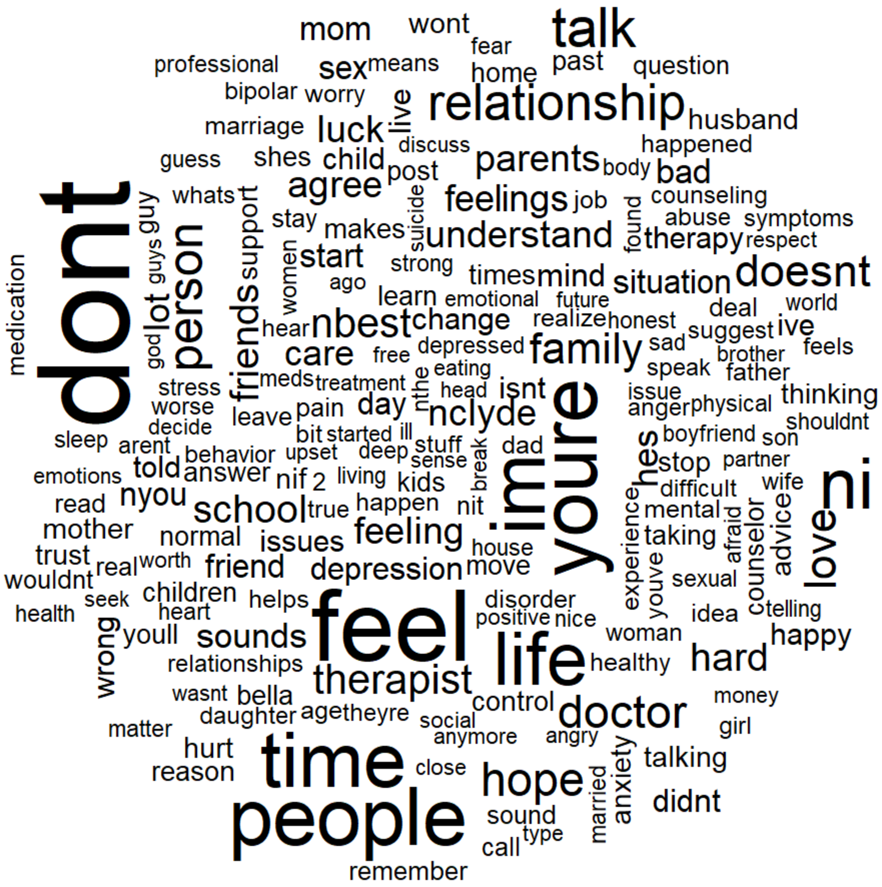
```
## Joining, by = "word"
```

```
head(sentiment)
```

```
## # A tibble: 6 x 3
##   word       sentiment     n
##   <chr>      <chr>     <int>
## 1 love       positive   4532
## 2 hard       negative   3659
## 3 luck       positive   3465
## 4 bad        negative   3079
## 5 depression negative   2647
## 6 wrong      negative   2473
```

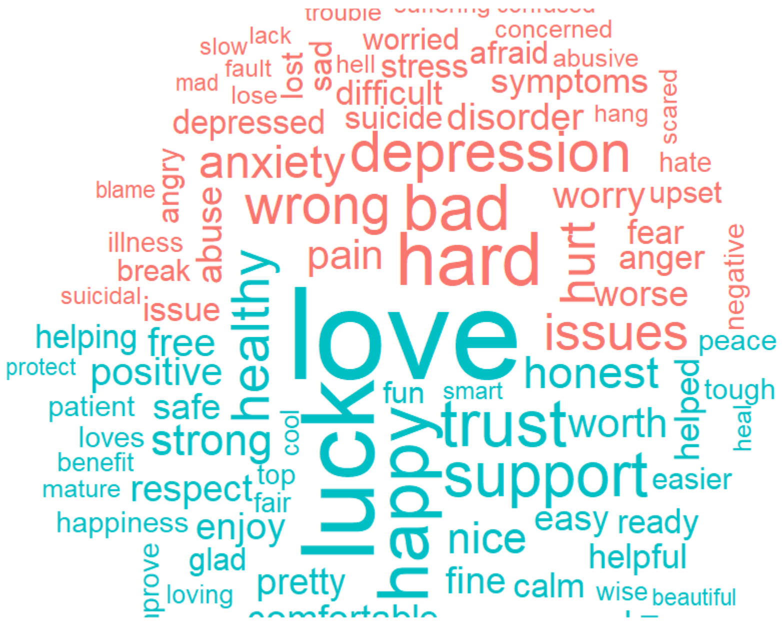
```
#WORDCLOUD

library("wordcloud")

tidy_text %>%
  anti_join(stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 200))
```

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



```
#COLOUR CODED WORDCLOUD
library("reshape2")


tidy_text %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("#F8766D", "#00BFC4"),
                   max.words = 100)
```

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



Topic Modeling using LDA (Latent Dirichlet Allocation) - It is a method of unsupervised classification of such documents, similar to clustering on numerical data, which finds natural group of items even when we are not sure what we are looking for.

Latent Dirichlet Allocation is a popular methof for fitting a topic model. It treats each document as a mixture of topics, and each topics as a mixture of words.

```
library(tm)
```

```
## Warning: package 'tm' was built under R version 3.5.3
```

```
## Loading required package: NLP
```

```
##
## Attaching package: 'NLP'
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
#library(RTextTools)
library(topicmodels)
```

```
## Warning: package 'topicmodels' was built under R version 3.5.3
```

```
library(slam)
```

```
## Warning: package 'slam' was built under R version 3.5.3
```

```
##
## Attaching package: 'slam'
```

```
## The following object is masked from 'package:data.table':
##
##     rollup
```

```
data <- data[1:1000,]
corpus <- Corpus(VectorSource(data$q_content), readerControl=list(language="en"))
dtm <- DocumentTermMatrix(corpus, control = list(stopwords = TRUE, minWordLength = 2, removeNumbers = TRUE, removePunctuatio
n = TRUE,  stemDocument = TRUE))
```

We use ldatuning for selecting number of k or topics using the code.

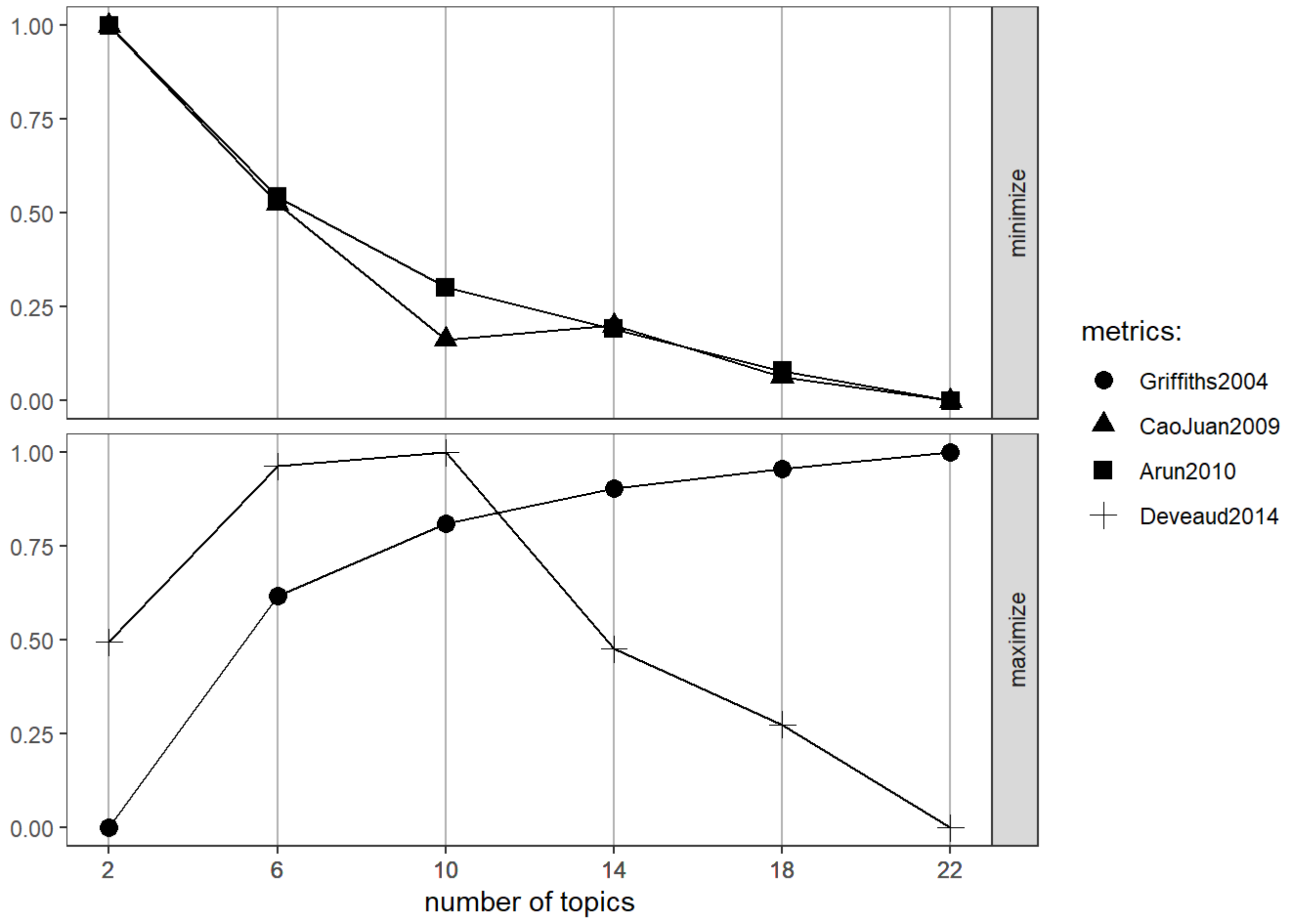
```
library(ldatuning)
```

```
## Warning: package 'ldatuning' was built under R version 3.5.3
```

```
result <- FindTopicsNumber(
  dtm,
  topics = seq(from = 2, to = 22, by = 4),
  metrics = c("Griffiths2004", "CaoJuan2009", "Arun2010", "Deveaud2014"),
  method = "Gibbs",
  control = list(seed = 77),
  mc.cores = 2L,
  verbose = TRUE
)
```

```
## fit models... done.
## calculate metrics:
##   Griffiths2004... done.
##   CaoJuan2009... done.
##   Arun2010... done.
##   Deveaud2014... done.
```

```
FindTopicsNumber_plot(result)
```



We can also start lda manually if we don’t want to use ldatuning package.

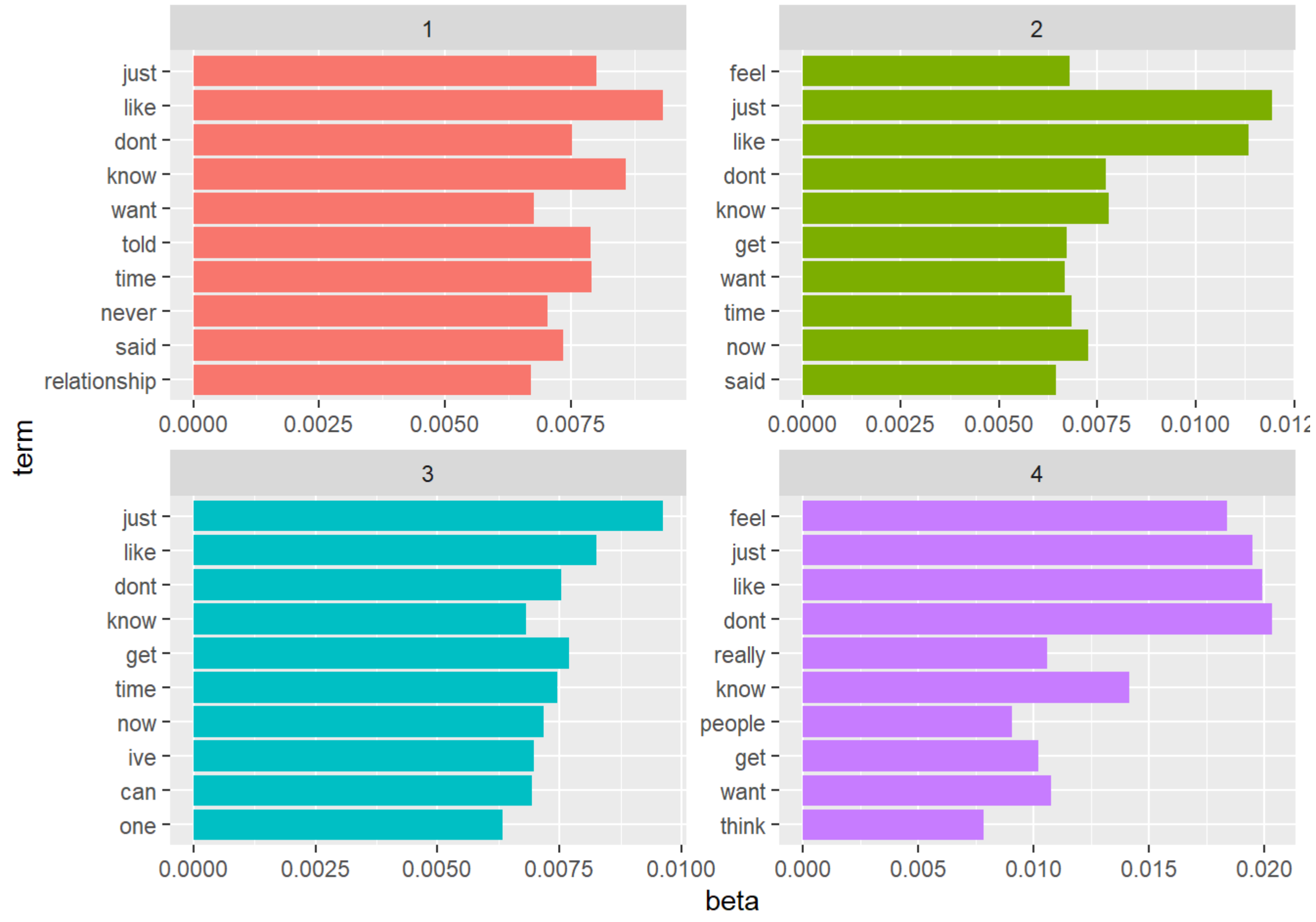
```
rowTotals <- apply(dtm , 1, sum)
dtm.new   <- dtm[rowTotals> 0, ]
lda <- LDA(dtm.new, k = 4)
```

```
lda_td <- tidy(lda)
lda_td
```

```
## # A tibble: 57,032 x 3
##    topic term          beta
##    <int> <chr>        <dbl>
##  1     1 aboven   2.69e- 5
##  2     2 aboven   6.80e-92
##  3     3 aboven   3.27e- 5
##  4     4 aboven   2.02e- 5
##  5     1 account  9.38e- 5
##  6     2 account  2.51e- 4
##  7     3 account  2.54e- 4
##  8     4 account  6.04e-17
##  9     1 actually 8.27e- 4
## 10     2 actually 5.65e- 4
## # ... with 57,022 more rows
```

```
top_terms <- lda_td %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)
```

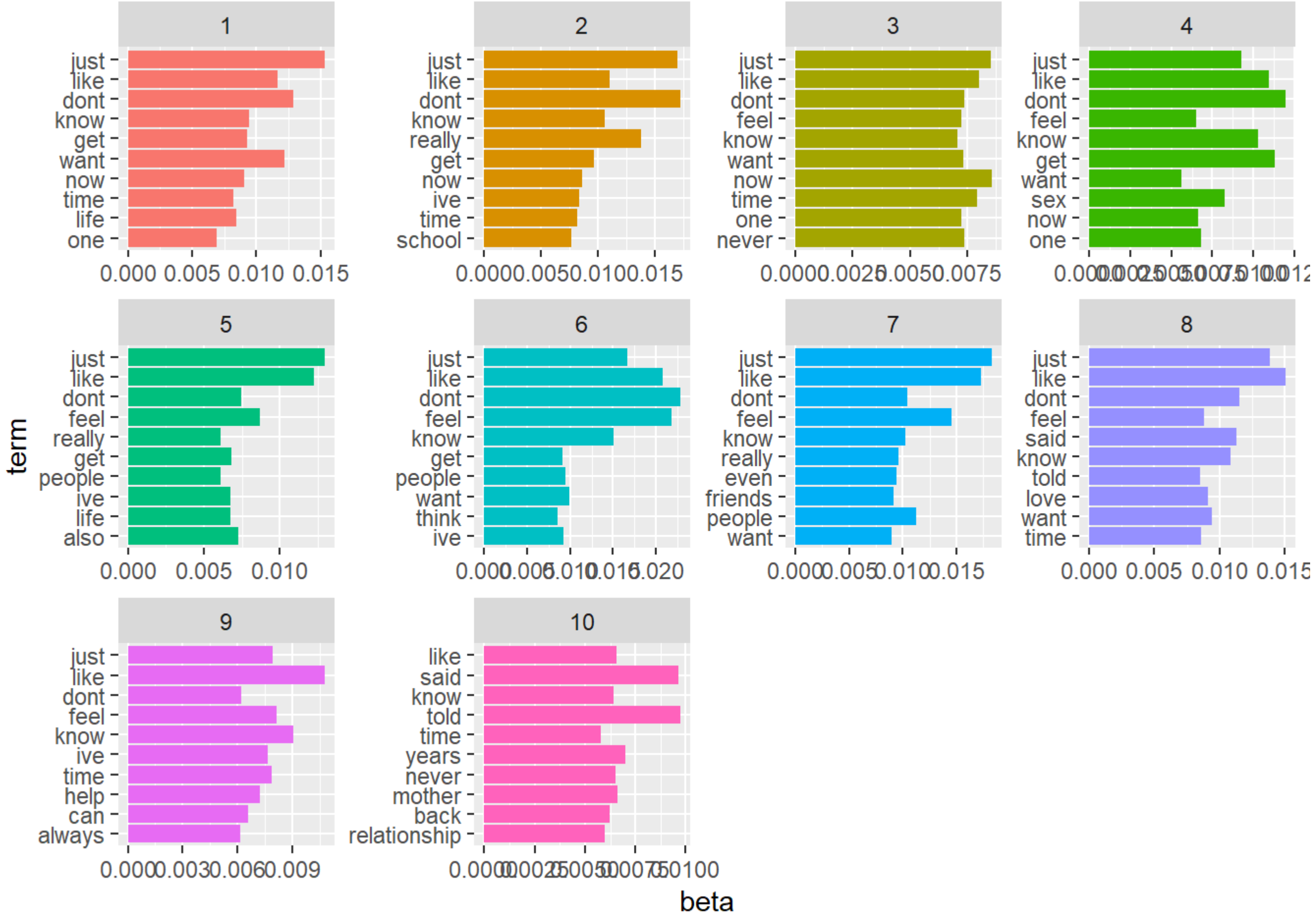
```
top_terms %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```



```
lda <- LDA(dtm.new, k = 10)
library(tidytext)
lda_td <- tidy(lda)
lda_td
```

```
## # A tibble: 142,580 x 3
##    topic term        beta
##    <int> <chr>      <dbl>
##  1     1 aboven 7.58e-  5
##  2     2 aboven 3.67e-185
##  3     3 aboven 2.36e-180
##  4     4 aboven 2.00e-181
##  5     5 aboven 1.66e-184
##  6     6 aboven 4.98e-183
##  7     7 aboven 7.10e-  5
##  8     8 aboven 3.31e-188
##  9     9 aboven 3.13e-182
## 10    10 aboven 6.66e-  5
## # ... with 142,570 more rows
```

```
top_terms <- lda_td %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

top_terms %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```

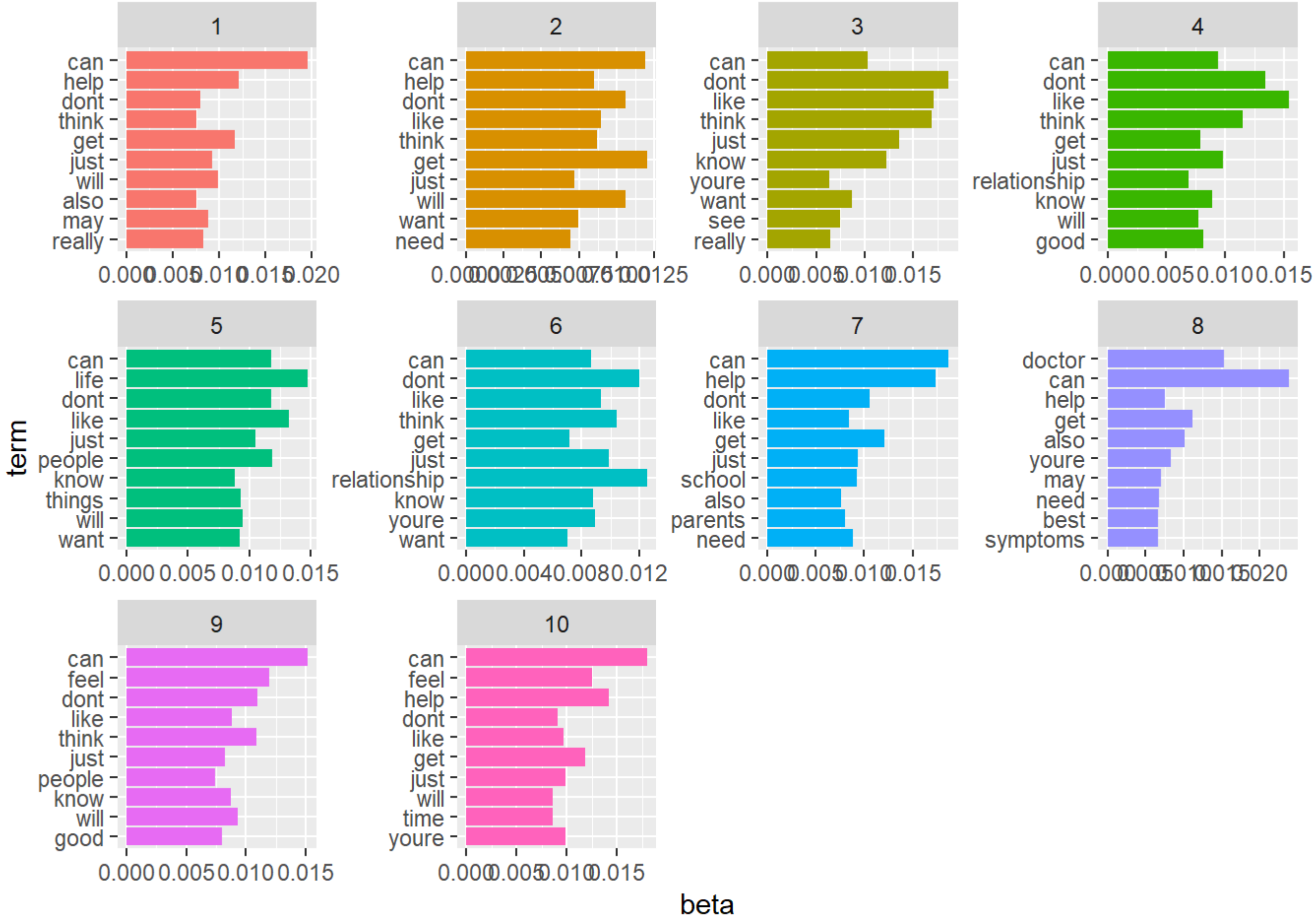


```
#doing same thing for answers
data <- data[1:1000,] # We perform LDA on the rows 1 through 1000 in the data.
corpus <- Corpus(VectorSource(data$answers), readerControl=list(language="en"))
dtm <- DocumentTermMatrix(corpus, control = list(stopwords = TRUE, minWordLength = 2, removeNumbers = TRUE, removePunctuatio
n = TRUE,  stemDocument = TRUE))
rowTotals <- apply(dtm , 1, sum) #Find the sum of words in each Document
dtm.new   <- dtm[rowTotals> 0, ] #remove all docs without words
lda <- LDA(dtm.new, k = 10)
```

```
lda_td <- tidy(lda)
lda_td
```

```
## # A tibble: 129,910 x 3
##    topic term             beta
##    <int> <chr>           <dbl>
##  1     1 actions 0.00000000557
##  2     2 actions 0.000533
##  3     3 actions 0.000341
##  4     4 actions 0.000224
##  5     5 actions 0.00000000664
##  6     6 actions 0.000223
##  7     7 actions 0.000342
##  8     8 actions 0.000219
##  9     9 actions 0.000693
## 10    10 actions 0.000244
## # ... with 129,900 more rows
```

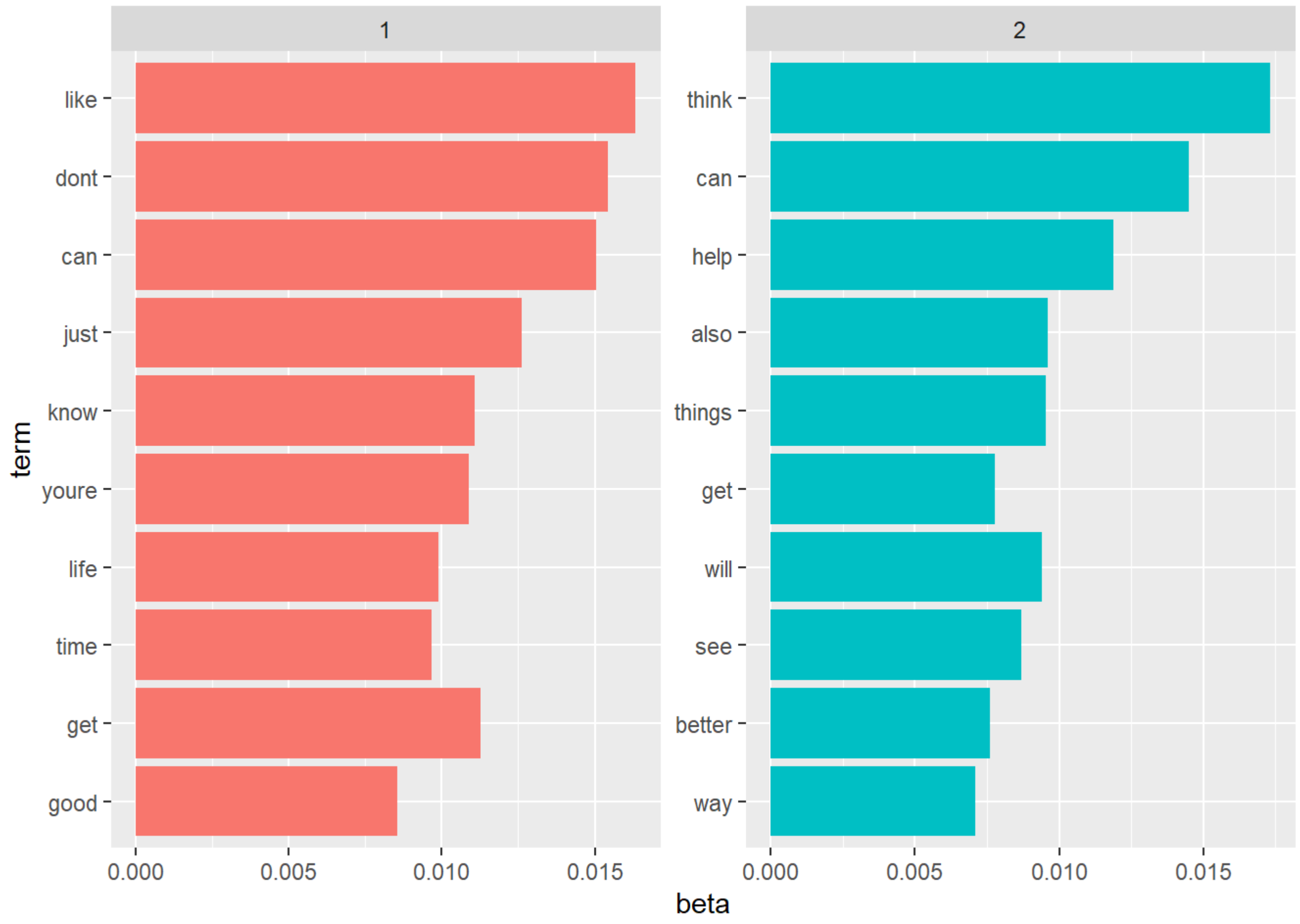
```
top_terms <- lda_td %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

top_terms %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```



```
#for K=2

lda <- LDA(dtm.new, k = 2) # k is the number of topics to be found.

library(tidytext)
lda_td <- tidy(lda)
lda_td
```

```
## # A tibble: 25,982 x 3
##    topic term           beta
##    <int> <chr>         <dbl>
##  1     1 actions  0.000456
##  2     2 actions  0.000117
##  3     1 activity 0.0000633
##  4     2 activity 0.000168
##  5     1 advice   0.00116
##  6     2 advice   0.00119
##  7     1 affected 0.000208
##  8     2 affected 0.0000917
##  9     1 also     0.00296
## 10     2 also     0.00961
## # ... with 25,972 more rows
```

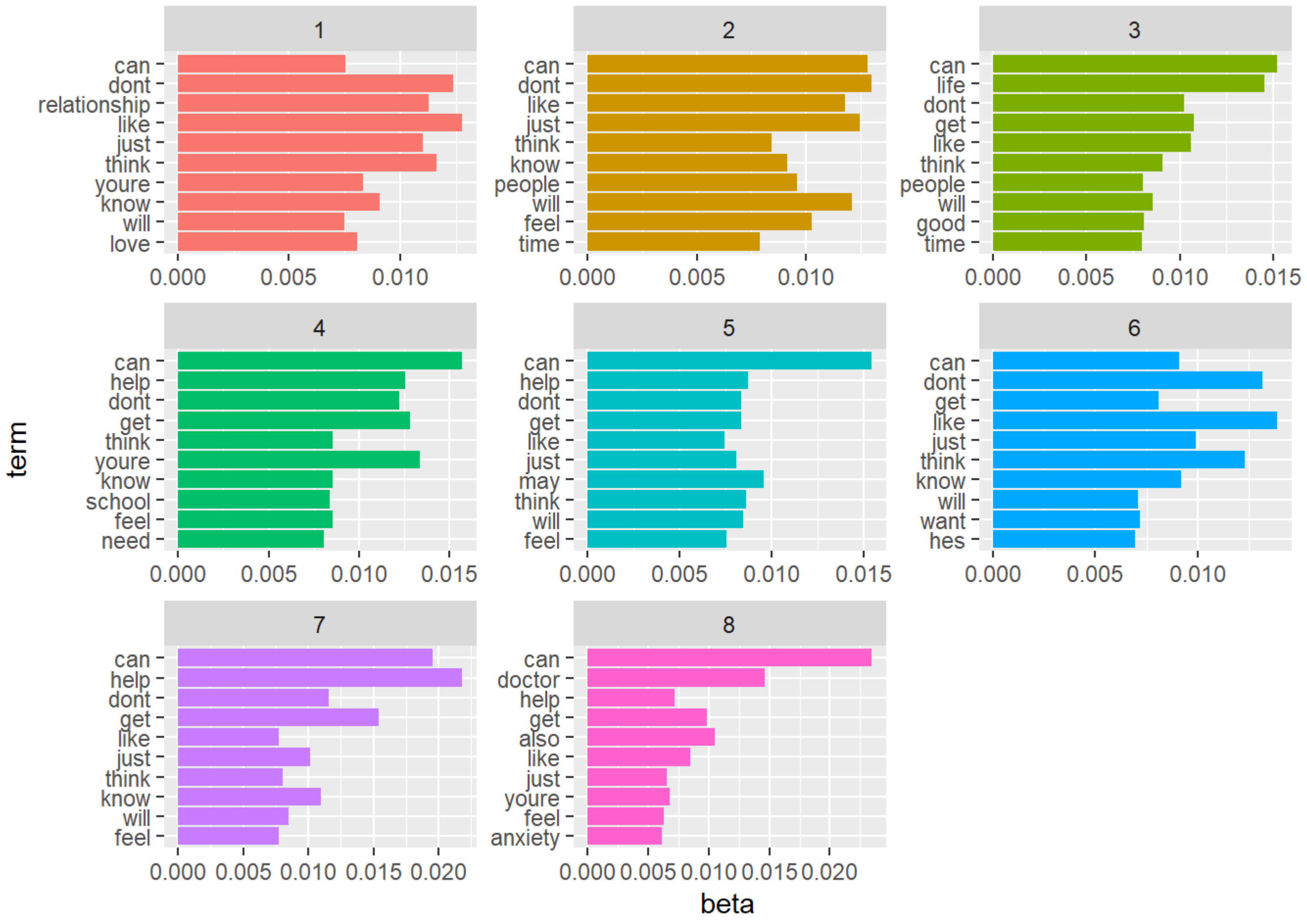
```
top_terms <- lda_td %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

top_terms %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```



```
#---------------------------------------
#for k=8
lda <- LDA(dtm.new, k = 8) # k is the number of topics to be found.

library(tidytext)
lda_td <- tidy(lda)
lda_td
```

```
## # A tibble: 103,928 x 3
##    topic term         beta
##    <int> <chr>       <dbl>
##  1     1 actions  5.09e- 8
##  2     2 actions  4.92e- 4
##  3     3 actions  3.86e-11
##  4     4 actions  5.96e- 4
##  5     5 actions  4.92e- 4
##  6     6 actions  3.63e- 4
##  7     7 actions  8.49e- 5
##  8     8 actions  2.76e- 4
##  9     1 activity 7.89e-19
## 10     2 activity 1.24e- 4
## # ... with 103,918 more rows
```

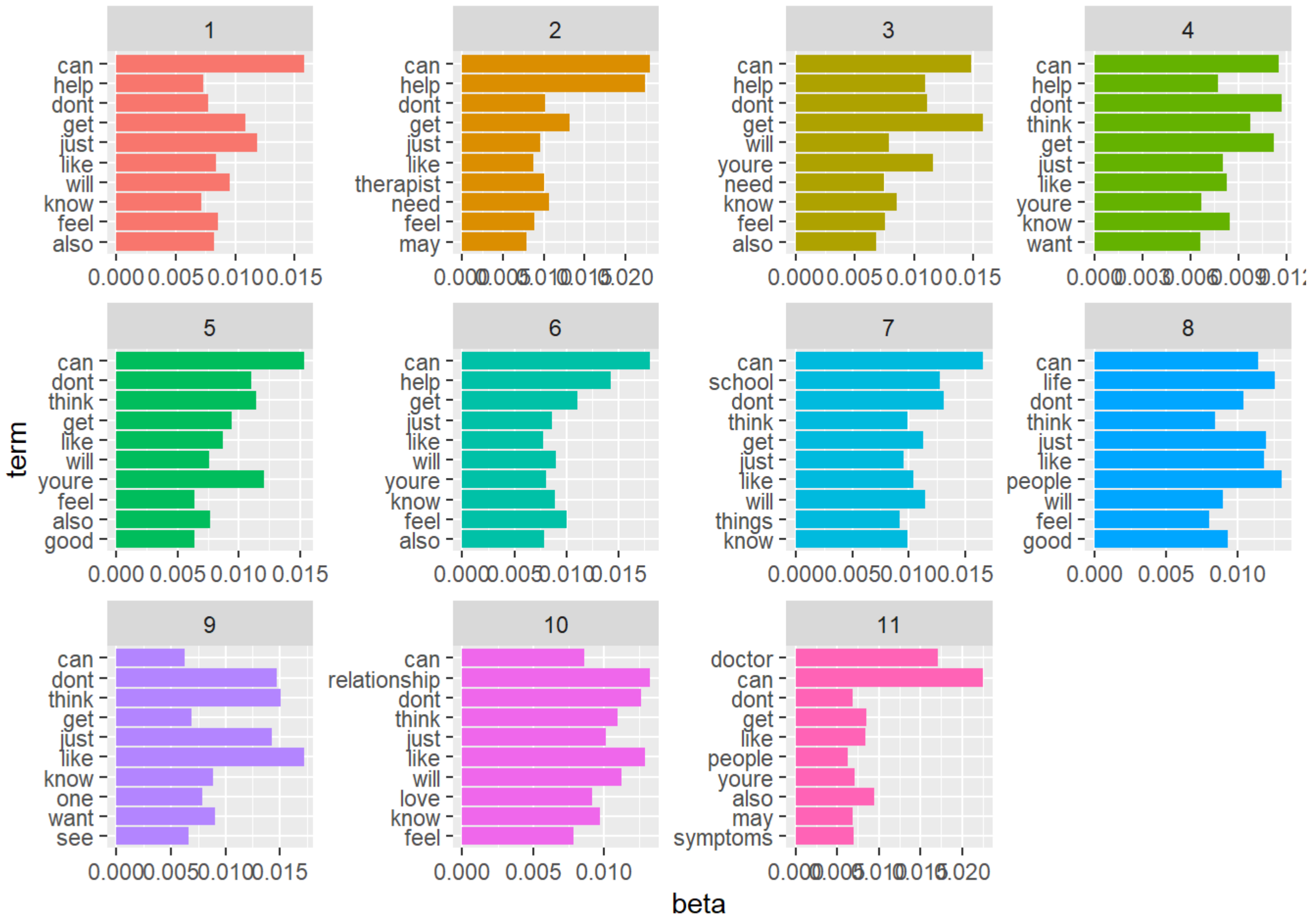
```
top_terms <- lda_td %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

top_terms %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```



```
#---------------------------------------
#for k=11
lda <- LDA(dtm.new, k = 11) # k is the number of topics to be found.

library(tidytext)
lda_td <- tidy(lda)
lda_td
```

```
## # A tibble: 142,901 x 3
##    topic term        beta
##    <int> <chr>      <dbl>
##  1     1 actions 1.93e- 4
##  2     2 actions 1.45e-15
##  3     3 actions 2.63e-15
##  4     4 actions 4.96e- 4
##  5     5 actions 9.30e- 4
##  6     6 actions 3.20e- 4
##  7     7 actions 1.98e- 4
##  8     8 actions 6.77e- 5
##  9     9 actions 9.00e- 5
## 10    10 actions 5.21e- 4
## # ... with 142,891 more rows
```

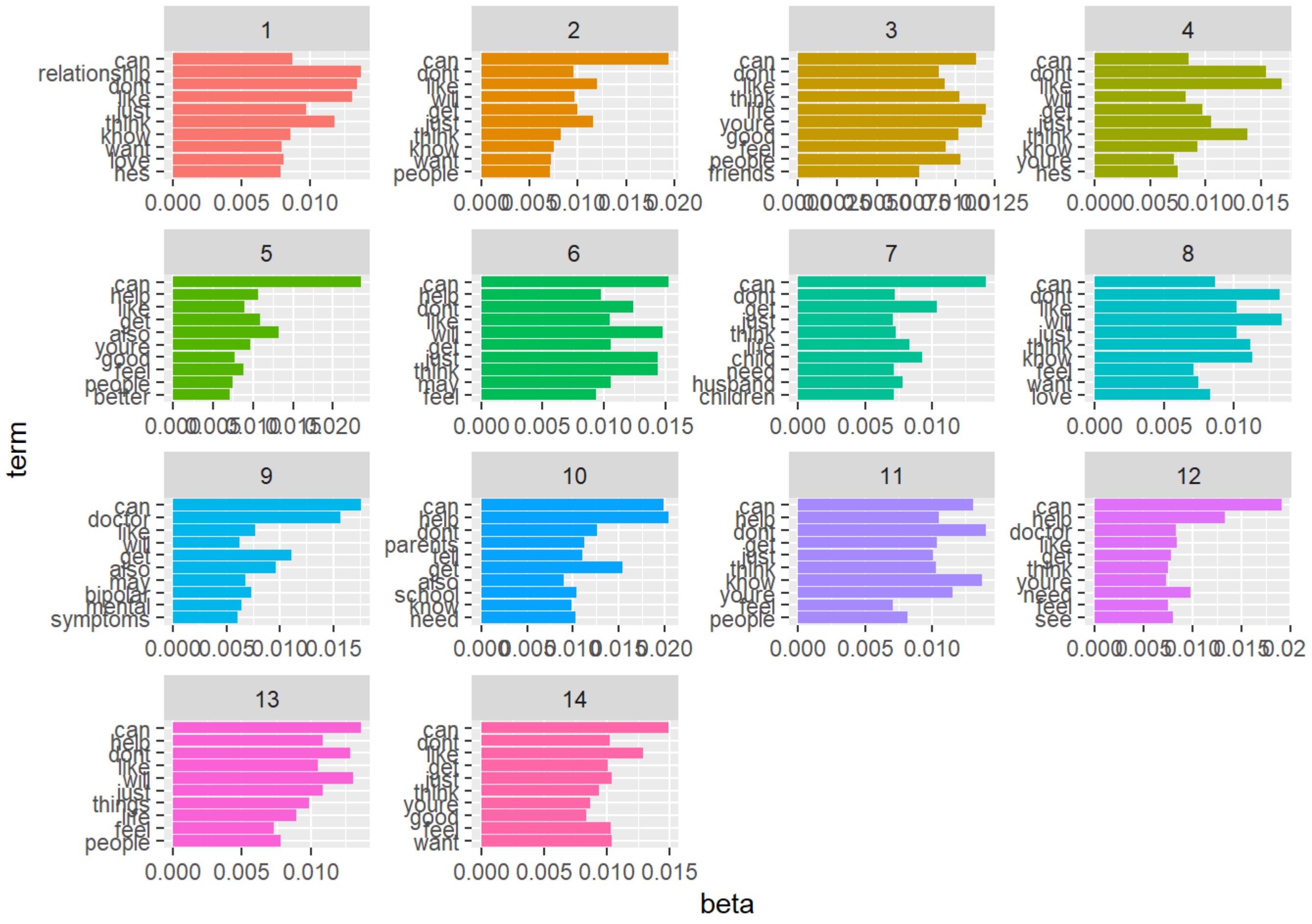
```
top_terms <- lda_td %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

top_terms %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```



```
#---------------------------------------
#for k=14
lda <- LDA(dtm.new, k = 14) # k is the number of topics to be found.

library(tidytext)
lda_td <- tidy(lda)
lda_td
```

```
## # A tibble: 181,874 x 3
##    topic term        beta
##    <int> <chr>      <dbl>
##  1     1 actions 3.80e- 9
##  2     2 actions 8.34e- 4
##  3     3 actions 1.62e- 4
##  4     4 actions 1.50e- 4
##  5     5 actions 1.08e-15
##  6     6 actions 5.79e- 4
##  7     7 actions 7.76e- 4
##  8     8 actions 6.52e- 4
##  9     9 actions 3.81e- 5
## 10    10 actions 4.49e- 4
## # ... with 181,864 more rows
```

```
top_terms <- lda_td %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

top_terms %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```



Final Thoughts :

1. The common words I can find here is “relationship”, “parents”, “school”, “husband”, “dream” and “depression” and the reason for their depression might be that they are not happy in their relationship or with their partners and parents or they are unhappy and stressed because of their dream (since they might not be able to achieve their goals) or their school life is not what they have expected to be and gets bullied which can be one of the cause.

2. Common words expressing their feelings are- depressed, anxiety, hate, worried, confused, wrong, feel that clearly shows what mood they were at the time they had posted this and what are the reasons behind that feeling.

3. People because of whom they are feeling like that - boyfriend, husband, friend, mother, family, people. These people are the reason for their mental stress and the way they are feeling because the words was repeated number of times (they can be directly or indirectly involved)

4. Some other words that came popping up in word cloud are - abuse marriage, scared, trust, schizophrenia, fear, alone, anger, girlfriend,children, wife, pregnant, ocd, suicidal and these words speak for themselves. I can connect to them right away with these words and it clearly expressed what they are going through.