Data Mining and Analysis Using R

Prabhani Gunasekera

1/28/2022

Introduction

This data mining and analysis exercise was carried out for the COVID-19 misinformation data set, which had initially been compiled for a research project at the Centre for Machine Learning and Health at Carnegie Mellon University by Shahan Ali Memon and Kathleen M. Carley. It contains Twitter data of about 4000 tweets around misinformation related to coronavirus pandemic.The data set had 3 columns- status ID, text and annotation. The annotations had been added manually based on the categories identified by the researchers.

Task A - Text Mining

```
#install.packages("tm")
#install.packages("textclean")
#install.packages("textstem")
#install.packages("SnowballC")
#install.packages("wordcloud")
#install.packages("RColorBrewer")
#install.packages("ggplot2")
```

Initially, the text mining package ‘tm’ and other required functions such as, SnowballC, RColorBrewer and ggplot2 were installed. However, when knitting the document these lines of codes were commented as it gives an error otherwise.

The required libraries were called before compiling the codes.

```
library(tm) #Text mining
```

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

```
library(SnowballC) #Required for stemming
library(textstem)
```

```
## Loading required package: koRpus.lang.en
```

```
## Loading required package: koRpus
```

```
## Loading required package: sylly
```

```
## For information on available language packages for 'koRpus', run
##
##   available.koRpus.lang()
##
## and see ?install.koRpus.lang()
```

```
##
## Attaching package: 'koRpus'
```

```
## The following object is masked from 'package:tm':
##
##     readTagged
```

```
library(RColorBrewer) #Required for color palettes
library(ggplot2) #Required for plotting
```

```
##
## Attaching package: 'ggplot2'
```

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

```
library(wordcloud) #Required to create the word cloud
library(wordcloud2) #Required to create a shaped word cloud with better visualisation
```

At first, the provided COVID_19_Twitter_Misinformation csv dataset was read to the R environment and stored.

```
# Read the csv file
filePath <- "CMU-MisCOV19.csv"
coviddata <- read.csv(filePath, header = TRUE)
#coviddata
```

Next, it was required to clean this data set so that we remove any clutter that would distort the true picture of the data set content and thereby change the sentiments and information extracted from the data set. For this purpose we only require to clean the text column. Hence, initially only the text column was separately extracted.

```
covidtw <- coviddata$text
#covidtw
```

1.1 Cleaning the Data

It was noted that there are foreign language characters. Hence, they are removed before loading the data as a corpus as it was easier to handle them in the normal character vector.

```
#remove other characters than english letters
covidtw <- stringr::str_replace_all(covidtw,"[^a-zA-Z0-9[:punct:]\\s']","")
#covidtw
```

In the above code all other characters excluding simple and capital letters from A to Z, numbers from 0 to 9 and punctuation marks were removed from the text.

Thereafter, the data were loaded as a corpus named “tweets” to carry out further data cleaning and analysis.

```
tweets <- Corpus(VectorSource(covidtw))
#inspect(tweets)
```

Using the inspect() function the first 1000 tweets were inspected to obtain an idea of the data set in terms of unwanted terms, phrases, word forms, punctuation, links etc.

Before carrying out further cleaning, a list of available transformations were obtained so that the relevant ones could be chosen.

```
getTransformations()
```

```
## [1] "removeNumbers"     "removePunctuation" "removeWords"
## [4] "stemDocument"      "stripWhitespace"
```

Afterwards, a range of functions were used to clean the data.

Firstly, the content_transformer function was used to remove unwanted twitter web links and new line character.

```
toSpace <- content_transformer(function (x , pattern ) gsub(pattern, " ", x))
tweets <- tm_map(tweets, toSpace, "\\s?(f|ht)(tp)(s?)(://)([^\\.]*)[\\.|/](\\S*)")
```

```
## Warning in tm_map.SimpleCorpus(tweets, toSpace, "\\s?(f|ht)(tp)(s?)(://)([^\
## \.]*)[\\.|/](\\S*)"): transformation drops documents
```

```
tweets <- tm_map(tweets, toSpace, "\n")
```

```
## Warning in tm_map.SimpleCorpus(tweets, toSpace, "\n"): transformation drops
## documents
```

Once the above were removed the text were transformed to lower case and numbers, punctuation, remaining special characters found, English stop words and extra white spaces present were removed. Finally, lemmatisation was carried out to keep only the base form of the words. Here, lemmatizing was used instead of stemming as stemming tends to leave only the stem of the words which in certain instances removed the meaningful form of the word. At the end of cleaning exercise, the data were inspected again to check if the cleaning had been performed to a satisfactory level at a glance.

```
# Convert the text to lower case
tweets <- tm_map(tweets, content_transformer(tolower))
```

```
## Warning in tm_map.SimpleCorpus(tweets, content_transformer(tolower)):
## transformation drops documents
```

```
# Remove numbers
tweets <- tm_map(tweets, removeNumbers)
```

```
## Warning in tm_map.SimpleCorpus(tweets, removeNumbers): transformation drops
## documents
```

```
# Remove punctuations
tweets <- tm_map(tweets, removePunctuation)
```

```
## Warning in tm_map.SimpleCorpus(tweets, removePunctuation): transformation drops
## documents
```

```
removeSpecialChars <- function(x) gsub("[^a-zA-Z0-9 ]","",x)
tweets <- tm_map(tweets, removeSpecialChars)
```

```
## Warning in tm_map.SimpleCorpus(tweets, removeSpecialChars): transformation drops
## documents
```

```
# Remove english common stopwords
tweets <- tm_map(tweets, removeWords, stopwords("english"))
```

```
## Warning in tm_map.SimpleCorpus(tweets, removeWords, stopwords("english")):
## transformation drops documents
```

```
# Eliminate extra white spaces
tweets <- tm_map(tweets, stripWhitespace)
```

```
## Warning in tm_map.SimpleCorpus(tweets, stripWhitespace): transformation drops
## documents
```

```
# Text stemming using lematisation
tweets <- tm_map(tweets, textstem::lemmatize_strings)
```

```
## Warning in tm_map.SimpleCorpus(tweets, textstem::lemmatize_strings):
## transformation drops documents
```

```
#inspect(tweets)
```

#1.2 Preliminary Knowledge Extraction and Visualisation

A data-set of terms were obtained using the document-term matrix. The function TermDocumentMatrix() from text mining package was used as follows to transform the data into a data frame to carry out the analysis. Then, a data frame with frequency of each word sorted in the descending order was obtained so that a information can be extracted on the most frequent terms that appeared in the tweets.

```
dtm <- TermDocumentMatrix(tweets)
m <- as.matrix(dtm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
head(d, 10)
```

word <chr>	freq <dbl>

	word <chr>	freq <dbl>
covid	covid	2177
coronavirus	coronavirus	1785
cure	cure	654
people	people	575
bleach	bleach	516
immune	immune	445
trump	trump	407
get	get	407
will	will	390
can	can	369

1-10 of 10 rows

It was seen that “covid” and “coronavirus” were the most frequent terms. This is expected as the tweets were obviously about the virus. The appearance of “cure” suggests that most of these tweets could have had misinformation on a cure for the virus, and “bleach” appeared to be a highly discussed topic in this regard as well. An interesting finding is that “trump” appeared as the 7th highest frequent word. It could be that the tweets discussed the former president Trump’s decisions or comments. Or maybe the word appeared in its actual meaning as well. To explore this further, other associated words with these frequent words were also checked. However, before going to that step the list of words that appeared more than 100 times and a plot of the most frequent 15 terms were explored to obtain a better picture of the frequency of the words.

```
findFreqTerms(dtm, lowfreq = 100)
```

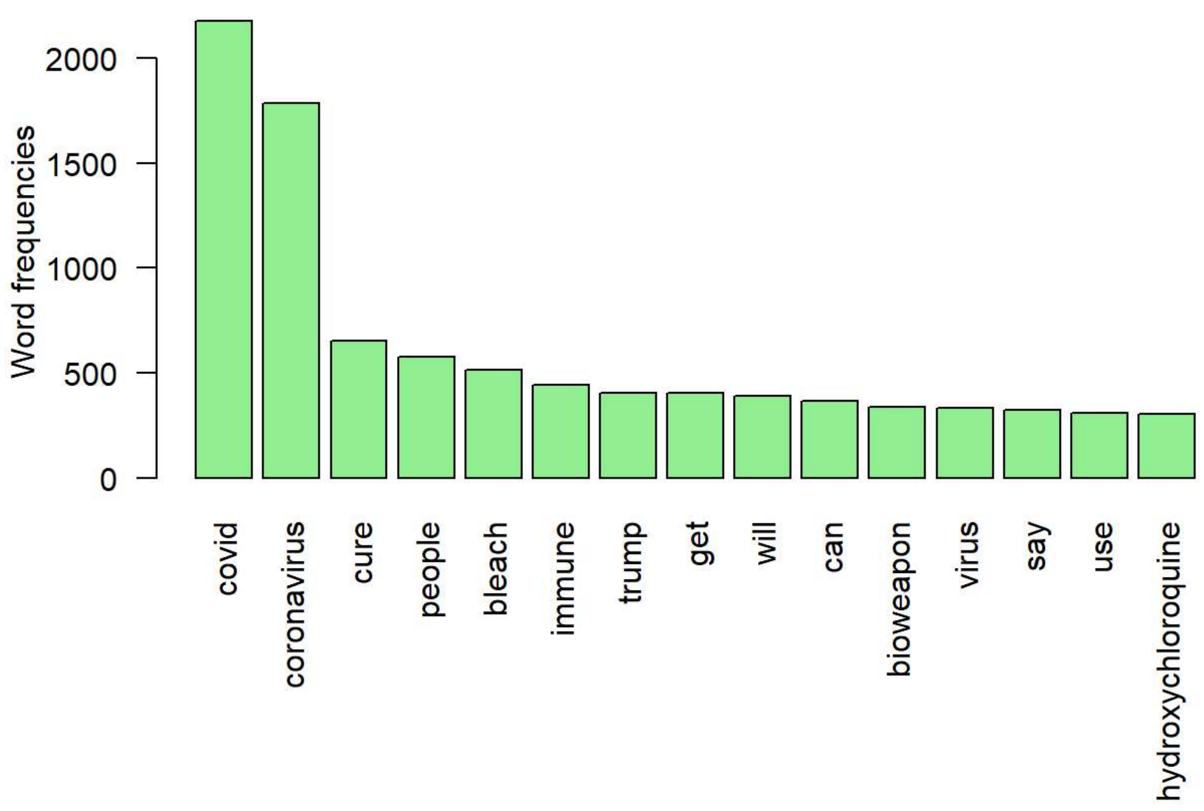
```
##  [1] "coronavirus"        "new"                "amp"
##  [4] "day"                "even"               "hand"
##  [7] "time"               "trump"              "stay"
## [10] "work"               "also"               "covid"
## [13] "fake"               "get"                "hydroxychloroquine"
## [16] "one"                "virus"              "china"
## [19] "death"              "health"             "believe"
## [22] "people"             "think"              "protect"
## [25] "wear"               "make"               "news"
## [28] "claim"              "cure"               "drink"
## [31] "garlic"             "water"              "like"
## [34] "cant"               "keep"               "via"
## [37] "vaccine"            "will"               "bioweapon"
## [40] "use"                "case"               "cause"
## [43] "good"               "know"               "system"
## [46] "spread"             "can"                "test"
## [49] "conspiracy"         "give"               "theory"
## [52] "world"              "prevent"            "say"
## [55] "oil"                "now"                "pandemic"
## [58] "find"               "take"               "tell"
## [61] "kill"               "just"               "stop"
## [64] "drug"               "treat"              "come"
## [67] "eat"                "help"               "immune"
## [70] "try"                "bleach"             "still"
## [73] "mask"               "please"             "see"
## [76] "need"               "doesnt"             "youre"
## [79] "fuck"               "want"               "may"
## [82] "treatment"          "malaria"
```

There were 83 words that had a frequency of more than 100 in the data set. Words such as, conspiracy and bioweapon seem to be related to one topic, whereas cure, garlic and hydroxychloroquine give the idea of another different topic that was discussed in tweets. At a glance, the above words indicate that the covid misinformation tweets carried a few main topics.

For better illustration, the frequency of the first 15 frequent words were plotted and figure 1.1 below.

```
par(mar=c(10,5,3,2))
barplot(d[1:15,]$freq, las = 2, names.arg = d[1:15,]$word,
        col ="lightgreen", main ="Figure 1.2.1 - Most Frequent Words in the Tweets",
        ylab = "Word frequencies")
```

Figure 1.2.1 - Most Frequent Words in the Tweets



In addition to the previously seen top 10 words, it was seen that bioweapon and hydroxychloroquine were part of top 15 as well. This gives a faint idea about the frequency of the top most tweeted topics as well. Using the previously created document term matrix, the associations of the frequent terms were explored as follows.

As mentioned previously, it was interesting to find the association or the correlation of the frequent terms with other terms. For this findAssocs() function was used.

The below R code finds the words associated with “cure” in the tweets.

```
findAssocs(dtm, terms = "cure", corlimit = 0.2)
```

```
## $cure
## hydroxychloroquine
##               0.29
```

It was seen that the 3rd most frequent word “cure” is only correlated with one word “hydroxychloroquine” and that is only a very weak positive association. This suggests that perhaps the tweets were discussing a possible cure but only a few tweets specifically discussed “hydroxychloroquine” in relation to a cure.

Next the association with the term “bioweapon” was checked.

```
findAssocs(dtm, terms = "bioweapon", corlimit = 0.2)
```

```
## $bioweapon
##    china  chinese    wuhan      lab engineer
##     0.33     0.30     0.27     0.23     0.23
```

It was seen that China was weakly correlated with the term bioweapon indicating the presence of some tweets relating China with a bioweapon. However, bioweapon was separately discussed as well.

```
findAssocs(dtm, terms = "bleach", corlimit = 0.2)
```

```
## $bleach
##  drink inject
##   0.41   0.31
```

The above weak associations with the word ‘bleach’ suggested low presence of comments around drinking or injecting bleach related to coronavirus.

Finally, as part of preliminary visualisations, word clouds were drawn depicting the significance of the words in the Twitter data set.

Figure 1.2.2 shows a basic word cloud drawn using wordcloud().

```
set.seed(1234)
wordcloud(words = d$word, freq = d$freq, min.freq = 1,
          max.words=200, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(12, "Paired"))
```

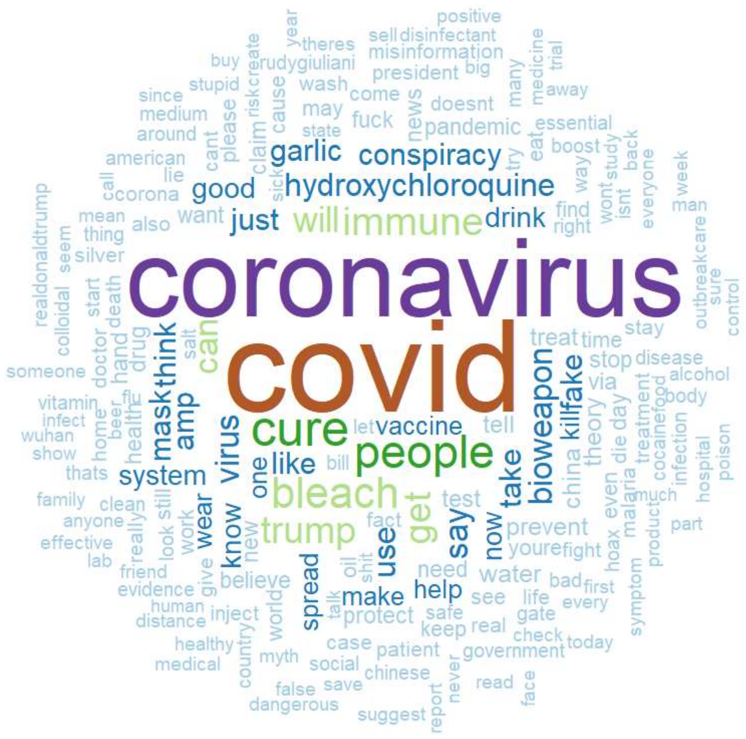
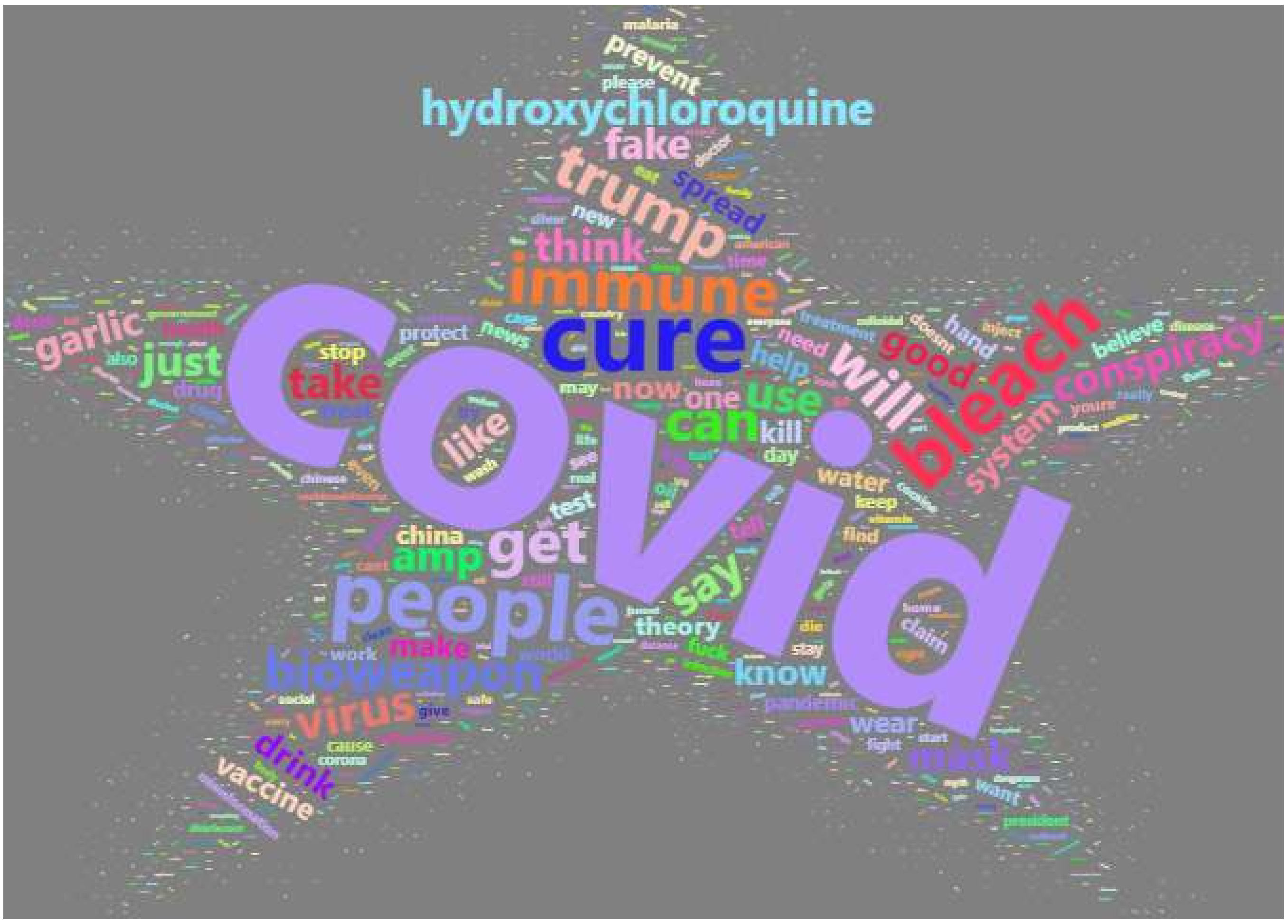


Figure 1.2.2 - Word cloud of 200 most significant words

A more prominent figure of word cloud(figure 1.2.3) in ‘star’ shape was drawn using wordcloud2(). Reference: https://cran.r-project.org/web/packages/wordcloud2/wordcloud2.pdf (https://cran.r-project.org/web/packages/wordcloud2/wordcloud2.pdf)

```
wordcloud2(d,  size = 1, color = "random-light", backgroundColor = "grey", shape = 'star')
```



```
Figure 1.2.3 - Word cloud of the tweets
```

```r
library(knitr)
#install.packages("tinytex")
#tinytex::install_tinytex()
```

Task B - Sensitivity Analysis

There were further packages that required to be installed to carry out sensitivity analysis.They are installed below. When knitting these were commented.

```r
#install.packages("tidytext")
#install.packages("textdata")
```

The below libraries were loaded before carrying out the analysis.

```r
library(dplyr)
```

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

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

```
library(stringr)
library(tidytext)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.6     v purrr   0.3.4
## v tidyr   1.1.4     v forcats 0.5.1
## v readr   2.1.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x ggplot2::annotate() masks NLP::annotate()
## x dplyr::filter()     masks stats::filter()
## x dplyr::lag()        masks stats::lag()
## x readr::tokenize()   masks koRpus::tokenize()
```

```
library(ggplot2)
library(tidyr)
library(textdata)
```

Previously text mining had been carried out using a corpus. However, it was required to combine the cleaned text from before with the initial data set with annotations. Therefore, the corpus was converted to a dataframe and then combined with the original dataset as below.

```
tweetsdf <- data.frame(cleanedtext = sapply(tweets, as.character), stringsAsFactors = FALSE)
#converting corpus to dataframe
covidtw2 <- cbind(coviddata,tweetsdf) #combined data set with cleaned text column
```

```
options(max.print = 100000)
```

Next, it was required to split the text into tokens, which in this case was words, so that sentiment analysis could be carried out.The below chunk of code performs the tokenisation.

```
tweets_tokens <- covidtw2 %>%
  mutate(cleanedtext=as.character(covidtw2$cleanedtext)) %>%
  unnest_tokens(word, cleanedtext)
tweets_tokens
```

status_id
<int>
0
0
0

status_id <int>
0
0
0
0
0
0
0

1-10 of 10,000 rows | 1-1 of 4 columns Previous **1** 2 3 4 5 6 ... 1000 Next

To obtain an overall idea about the frequency of the tokens/words, the count function was used. This gave the same output as before when the count was taken from document term matrix where covid and coronavirus were the first and second most frequent terms respectively.

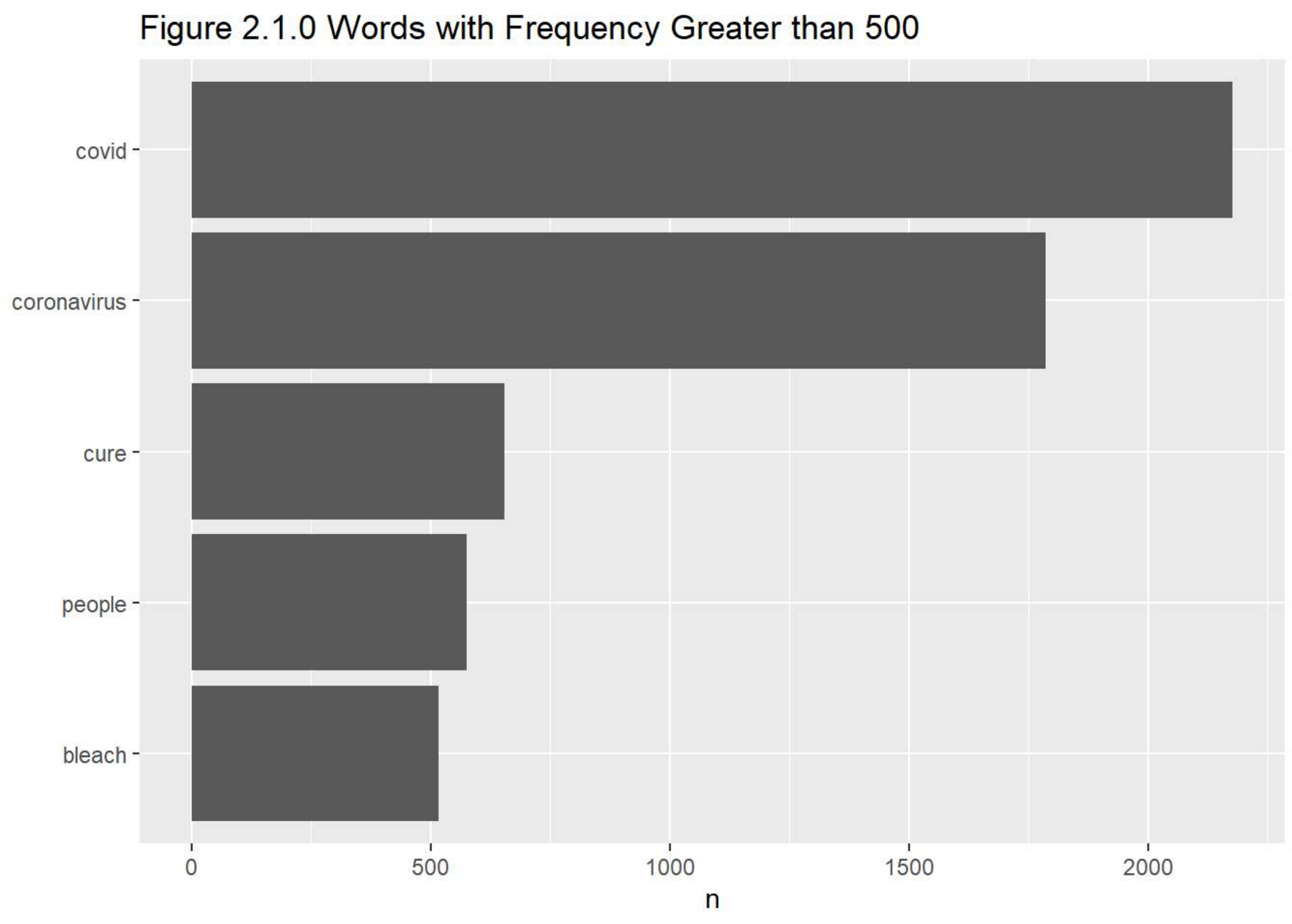
```
tweets_tokens %>%
  count(word, sort = TRUE)
```

word <chr>	n <int>
covid	2177
coronavirus	1785
cure	654
people	575
bleach	516
immune	445
get	407
trump	407
will	390
can	369

1-10 of 9,310 rows Previous **1** 2 3 4 5 6 ... 931 Next

In the previous section it was explored as to which list of words appeared more than 100 times. Here, it a horizontal bar chart of words which appeared more than 500 times was produced.As per below figure 2.1.0 the list comprised of words generally related to the pandemic except for ‘bleach’. It indicated that out of the topics discussed in the tweets something related to bleach had been mostly tweeted about.

```
tweets_tokens %>%
  count(word, sort = TRUE) %>%
  filter(n > 500) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  ggtitle('Figure 2.1.0 Words with Frequency Greater than 500') +
  geom_col() +
  xlab(NULL) +
  coord_flip()
```



Before the proper sentiment analysis was carried out, the sentiment lexicons in the tidyverse package were reviewed as follows.

```
sentiments
```

word <chr>	**sentiment** <chr>
2-faces	negative
abnormal	negative
abolish	negative
abominable	negative
abominably	negative
abominate	negative

word <chr>	sentiment <chr>
abomination	negative
abort	negative
aborted	negative
aborts	negative

1-10 of 6,786 rows Previous **1** 2 3 4 5 6 ... 679 Next

While the sentiments were broadly classified as positive or negative the sentiment words could also be used in the analysis.

First, ‘angry’ words in the tweets were chosen to be explored. The below code identified the overall most common anger related words in the tweets.

```
nrc_anger <- get_sentiments("nrc") %>%
  filter(sentiment == "anger")

tweets_tokens %>%
  inner_join(nrc_anger) %>%
  count(word, sort = TRUE)
```

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

word <chr>	n <int>
treat	148
death	102
lie	92
fight	91
disease	83
bad	81
hoax	76
shit	58
poison	52
lose	42

1-10 of 343 rows Previous **1** 2 3 4 5 6 ... 35 Next

As per the output, it was seen that treat, death, lie, fight and disease topped the list in terms of frequency.

Since the data set had annotations, it was important to identify the most common anger words with respect to certain separate annotations. It was decided to check the anger words of ‘Conspiracy’ and ‘Fake Cure’ annotated tweets.

The below code explored the anger words in ‘Conspiracy’ related tweets.

```r
nrc_anger <- get_sentiments("nrc") %>%
  filter(sentiment == "anger")

tweets_tokens %>%
  filter(annontation == "conspiracy") %>%
  inner_join(nrc_anger) %>%
  count(word, sort = TRUE)
```

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

word <chr>	**n** <int>
death	24
attack	20
lie	17
poison	17
bad	15
hoax	15
evil	12
money	12
fear	11
force	10

1-10 of 136 rows Previous **1** 2 3 4 5 6 ... 14 Next

With respect to ‘Conspiracy’, death and attack were the most frequent anger words.

Next the most common anger words in ‘fake cure’ related tweets were explored.

```r
nrc_anger <- get_sentiments("nrc") %>%
  filter(sentiment == "anger")

tweets_tokens %>%
  filter(annontation == "fake cure") %>%
  inner_join(nrc_anger) %>%
  count(word, sort = TRUE)
```

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

word <chr>	**n** <int>
treat	4
demand	3
disease	3

word <chr>	n <int>
death	2
scare	2
awful	1
bee	1
censor	1
combat	1
dying	1

1-10 of 19 rows Previous **1** 2 Next

As per the above output, there were no words with very high frequency. However, treat and demand were the most common.

Thereafter, the overall positive and negative sentiments were analysed using the ‘bing’ lexicon.

```
tweets_tokens %>%
  inner_join(get_sentiments("bing")) %>% # pull out only sentiment words
  count(sentiment) %>% # gets the counts of positive & negative words
  spread(sentiment, n, fill = 0) %>% # make data wide rather than narrow
  mutate(sentiment = positive - negative)
```

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

negative <dbl>	positive <dbl>	sentiment <dbl>
5305	4033	-1272

1 row

It was seen that the overall sentiment of the tweets was negative as the negatives outweighed the positives.

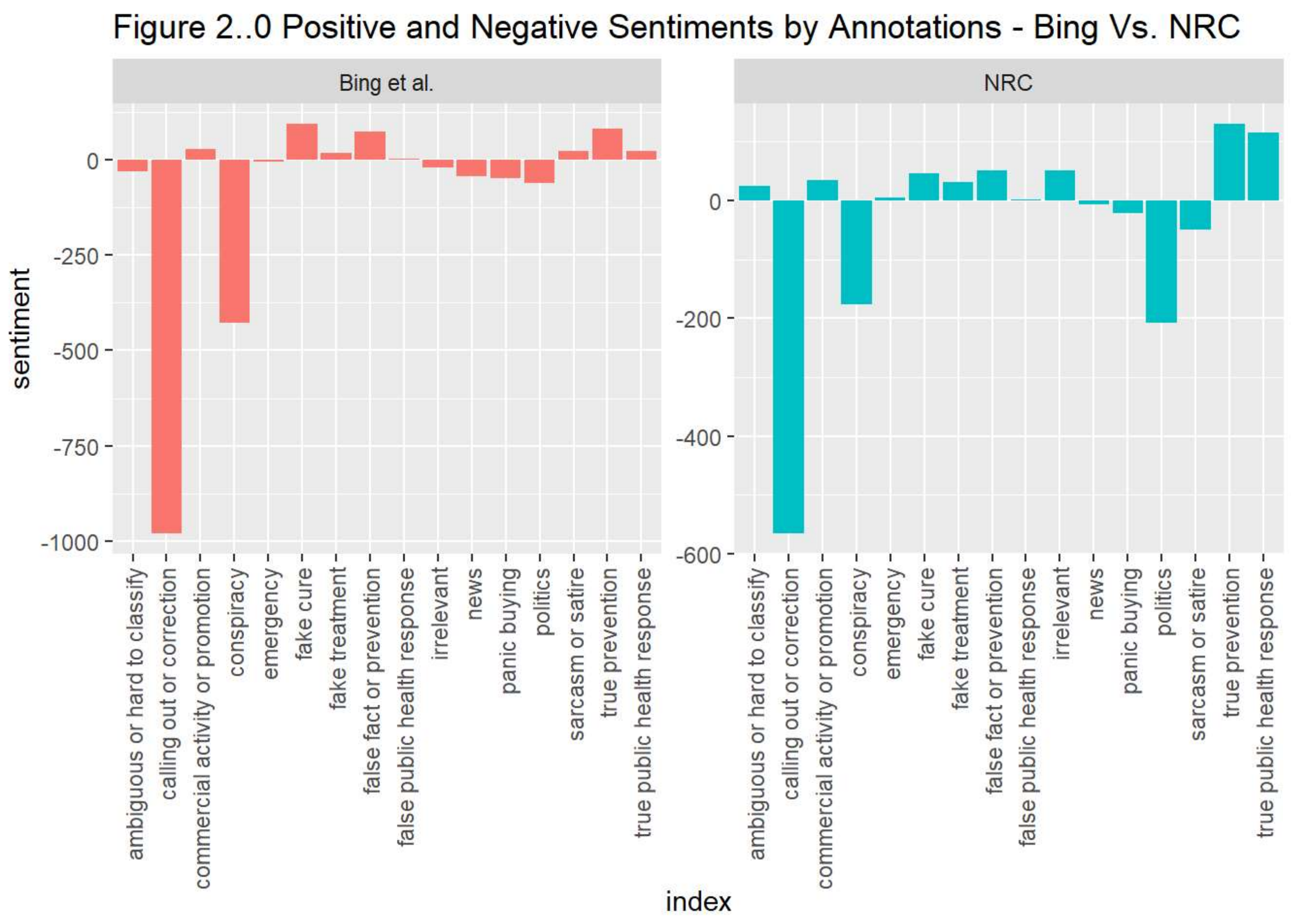
Next, it was attempted to compare the scores by annotations using Bing and NRC Emotion lexicon.

```
bing_and_nrc <- bind_rows(tweets_tokens %>%
      inner_join(get_sentiments("bing")) %>%
        mutate(method = "Bing et al."),
  tweets_tokens %>%
      inner_join(get_sentiments("nrc") %>%
        filter(sentiment %in% c("positive", "negative"))) %>%
        mutate(method = "NRC")) %>%
            count(method, index = annontation, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)
```

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

In order to illustrate the results, a plot of the above output was produced using the below code. Figure 2.2.0 gives this plot.

```
bing_and_nrc %>%
  ggplot(aes(index, sentiment, fill = method)) +
  geom_col(show.legend = FALSE) +
  ggtitle('Figure 2..0 Positive and Negative Sentiments by Annotations - Bing Vs. NRC') +
  theme(axis.text.x = element_text(angle=90, vjust=.5, hjust=1))+
  facet_wrap(~method, ncol = 2, scales = "free_y")
```



Task C - Topic Modelling

Initially, the required libraries were called.

```
library(topicmodels)
library(stats)
library(reshape2)
```

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

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

The word counts by annotations were explored here.

```
word_counts <- tweets_tokens %>%
  anti_join(stop_words) %>%
  count(annontation, word, sort = TRUE) %>%
  ungroup()
```

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

```
word_counts
```

annontation <chr>	word <chr>	n <int>
calling out or correction	covid	805
calling out or correction	coronavirus	593
calling out or correction	people	294
conspiracy	coronavirus	290
politics	trump	283
conspiracy	covid	277
conspiracy	bioweapon	231
calling out or correction	cure	229
politics	covid	224
sarcasm or satire	covid	202

1-10 of 10,000 rows   Previous **1** 2 3 4 5 6 ... 1000 Next

Thereafter, the data set was added to a document term matrox using the below code.

```
annontations_dtm <- word_counts %>%
  cast_dtm(annontation, word, n)
annontations_dtm
```

```
## <<DocumentTermMatrix (documents: 16, terms: 8988)>>
## Non-/sparse entries: 17858/125950
## Sparsity           : 88%
## Maximal term length: 198
## Weighting          : term frequency (tf)
```

Next, the LDA model was run with 3 topics.

```
annontations_lda <- LDA(annontations_dtm, k = 3, control = list(seed = 1234))
annontations_lda
```

```
## A LDA_VEM topic model with 3 topics.
```

```
annontations_topics <- tidy(annontations_lda, matrix = "beta")
annontations_topics
```

topic <int>	term <chr>	beta <dbl>
1	covid	4.699126e-02
2	covid	4.264250e-02
3	covid	4.380880e-02
1	coronavirus	3.615842e-02
2	coronavirus	4.386528e-02
3	coronavirus	3.022005e-02
1	people	1.701930e-02
2	people	7.569858e-03
3	people	6.343893e-03
1	trump	1.534682e-02

1-10 of 10,000 rows Previous **1** 2 3 4 5 6 ... 1000 Next

The top 5 terms in each of the 3 topics were analysed and listed using the below code.

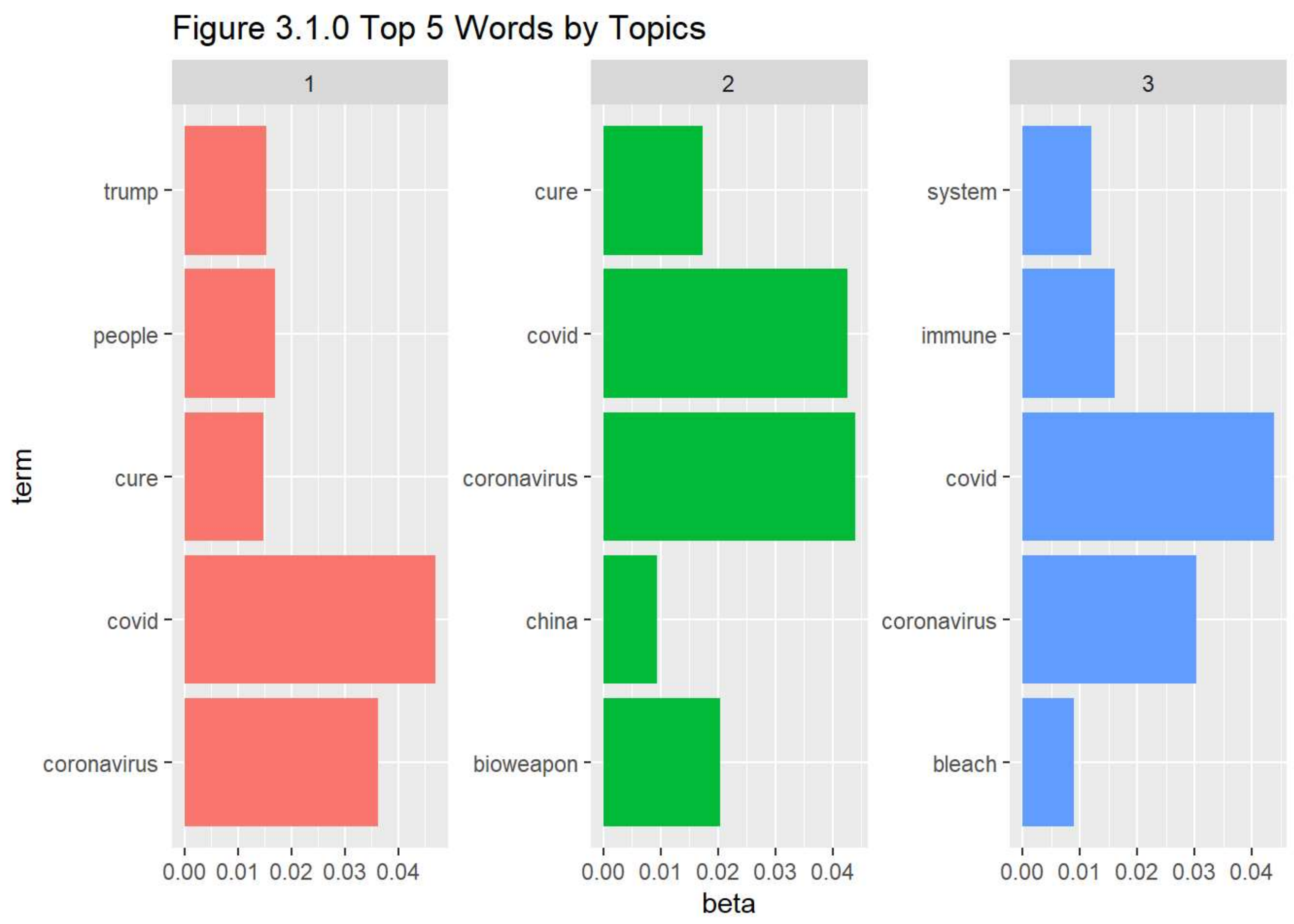
```
top_terms <- annontations_topics %>%
  group_by(topic) %>%
  top_n(5, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)
top_terms
```

topic <int>	term <chr>	beta <dbl>
1	covid	0.046991256
1	coronavirus	0.036158423
1	people	0.017019305
1	trump	0.015346822
1	cure	0.014828191
2	coronavirus	0.043865281
2	covid	0.042642499
2	bioweapon	0.020428962
2	cure	0.017325150
2	china	0.009351765

1-10 of 15 rows Previous **1** 2 Next

These were graphically represented by a plots as follows in figure 3.1.0.

```
top_terms %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  ggtitle('Figure 3.1.0 Top 5 Words by Topics') +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```



It showed clear separation in top terms amongst the different topics.

```
annontations_gamma <- tidy(annontations_lda, matrix = "gamma")
annontations_gamma
```

document <chr>	topic <int>	gamma <dbl>
calling out or correction	1	9.999947e-01
conspiracy	1	6.113157e-06
politics	1	9.999871e-01
sarcasm or satire	1	3.140910e-01
false fact or prevention	1	1.113980e-05
true prevention	1	1.734694e-05
fake cure	1	3.970219e-05

document <chr>	topic <int>	gamma <dbl>
true public health response	1	1.942357e-05
ambiguous or hard to classify	1	1.212034e-01
news	1	4.012573e-05

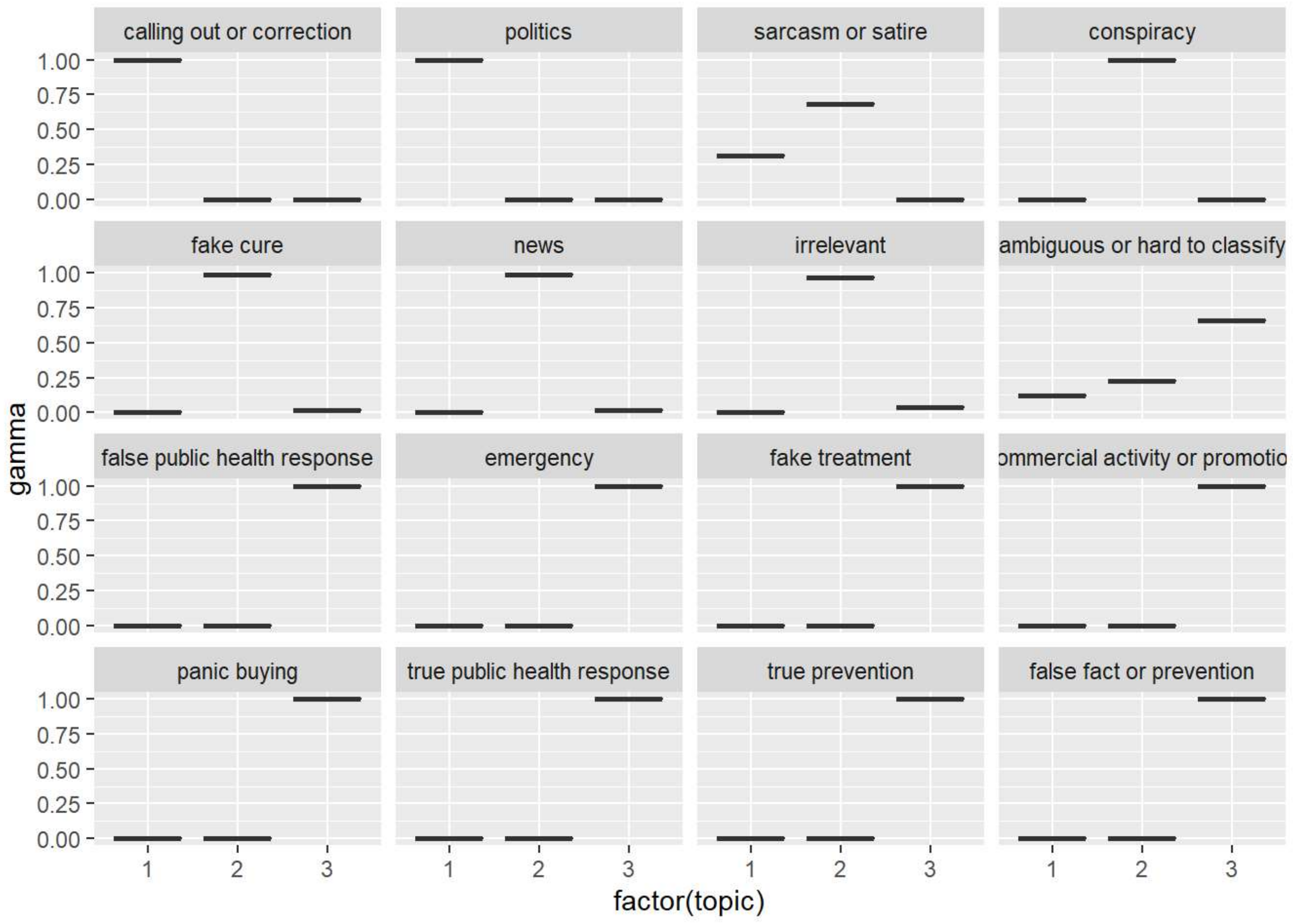
1-10 of 48 rows   Previous **1** 2 3 4 5 Next

```
annontations_gamma <- annontations_gamma %>%
  separate(document, "annontation", sep = "_", convert = TRUE)
annontations_gamma
```

annontation <chr>	topic <int>	gamma <dbl>
calling out or correction	1	9.999947e-01
conspiracy	1	6.113157e-06
politics	1	9.999871e-01
sarcasm or satire	1	3.140910e-01
false fact or prevention	1	1.113980e-05
true prevention	1	1.734694e-05
fake cure	1	3.970219e-05
true public health response	1	1.942357e-05
ambiguous or hard to classify	1	1.212034e-01
news	1	4.012573e-05

1-10 of 48 rows   Previous **1** 2 3 4 5 Next

```
# reorder titles in order of topic 1, topic 2 and topic 3 before plotting
annontations_gamma %>%
  mutate(title = reorder(annontation, gamma * topic)) %>%
  ggplot(aes(factor(topic), gamma)) +
  geom_boxplot() +
  facet_wrap(~ title)
```



```
annontation_classifications <- annontations_gamma %>%
  group_by(annontation) %>%
  top_n(1, gamma) %>%
  ungroup()
annontation_classifications
```

annontation <chr>	topic <int>	gamma <dbl>
calling out or correction	1	0.9999947
politics	1	0.9999871
conspiracy	2	0.9999878
sarcasm or satire	2	0.6858996
fake cure	2	0.9849003
news	2	0.9847673
irrelevant	2	0.9633245
false fact or prevention	3	0.9999777
true prevention	3	0.9999653
true public health response	3	0.9999612

1-10 of 16 rows Previous **1** 2 Next

As per above output the annontations were separated to the 3 broad topics.