TAR Course Analysis

Ryan Christensen

Contents

Loading the packages and building the url

```
require(rvest)
require(tidytext)
require(tidyverse)
require(stringr)
require(tm)
require(topicmodels)
require(wordcloud)
```

```
urlBase <- sprintf("https://e-discoveryteam.com/tar-course/tar-course-")

target <- map(1:17, function(i) {
  if (i == 1) {
    paste(urlBase, i, "st-class/", sep = "")
  } else if (i == 2) {
    paste(urlBase, i, "nd-class/", sep = "")
  } else if ( i == 3) {
    paste(urlBase, i, "rd-class/", sep = "")
  } else {
    paste(urlBase, i, "th-class/", sep = "")
  }
})
```

Scraping the data

```
texts <- map(target, read_html)

text_list <- map(texts, function(i) {
  (html_text(html_nodes(i, ".entry")))
})
```

Chucking that "borrowed" data into a data_frame

Tokenizing the text puts it into a Tidy format of one term per row. This makes removing standard stop words easy with an `anti-join`

```
text_df <- data_frame(text = unlist(text_list), Article = 1:17)

text_df <- text_df %>%
  unnest_tokens(word, text)

text_df %>%
  anti_join(stop_words) %>%
  count(word, sort = TRUE)
```

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

```
## # A tibble: 3,918 x 2
##          word     n
##         <chr> <int>
##  1  documents   349
##  2     review   328
##  3     search   289
##  4   document   228
##  5     coding   217
##  6 predictive   208
##  7   training   205
##  8    machine   194
##  9       step   177
## 10  discovery   159
## # ... with 3,908 more rows
```

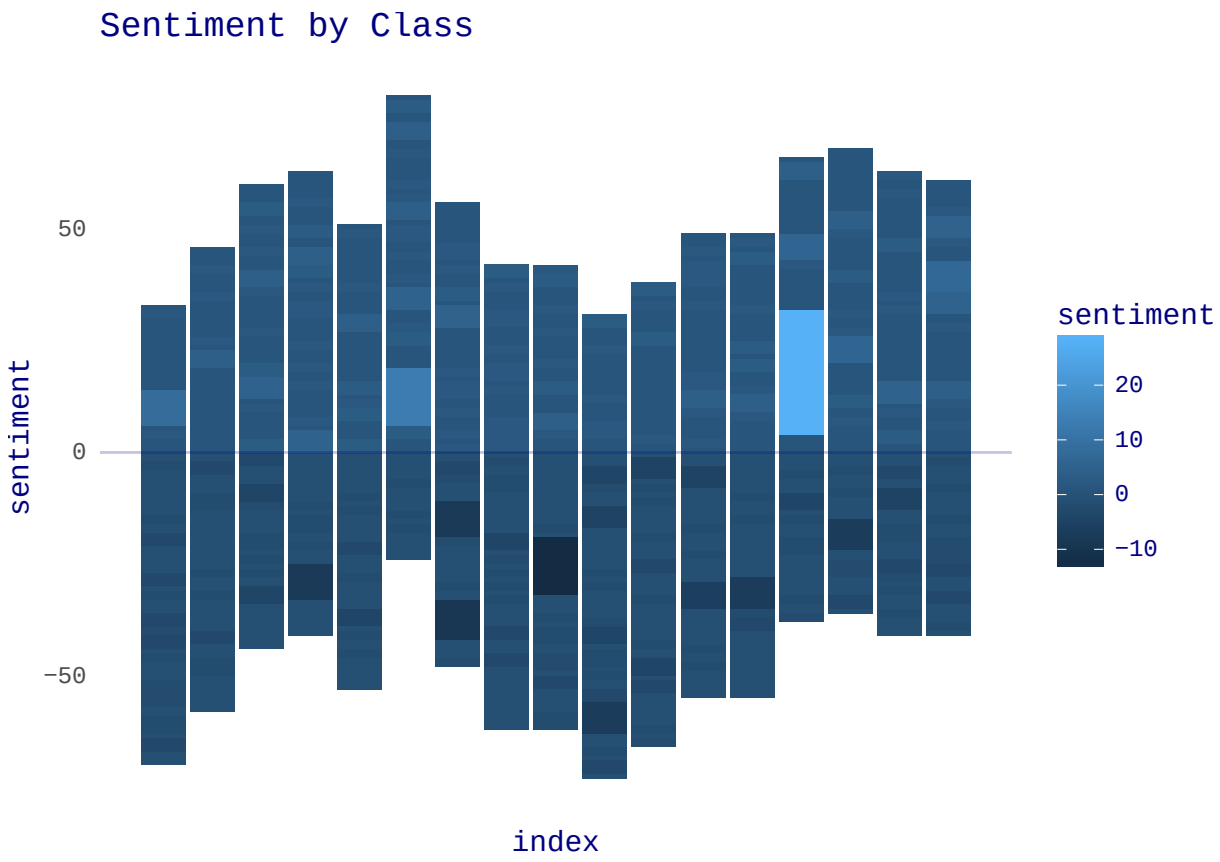
Taking a look at the authors state of mind

Simple sentiment analysis of the courses. Each class is chuncked together and plotted with each word's sentiment score colored by degree of positive or negative.

```
text_sentiment <- text_df %>%
  anti_join(stop_words) %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, index = row_number() %/% 104, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)
```

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

```
ggplot(text_sentiment, aes(index, sentiment, fill = sentiment)) +
  geom_col() +
  geom_hline(aes(yintercept = 0), alpha = 0.23, col = "navy") +
  theme(panel.background = element_blank(),
        text = element_text(family = "mono", color = "navy"),
        axis.ticks = element_blank(),
        axis.text.x = element_blank()) +
  ggtitle("Sentiment by Class ")
```



TF_IDF — Looking at what terms are most important to the overall text.

```
article_words <- text_df %>%
  count(Article, word, sort = TRUE)

total_words <- article_words %>%
  group_by(Article) %>%
  summarise(total = sum(n))

article_words <- left_join(article_words, total_words)
```

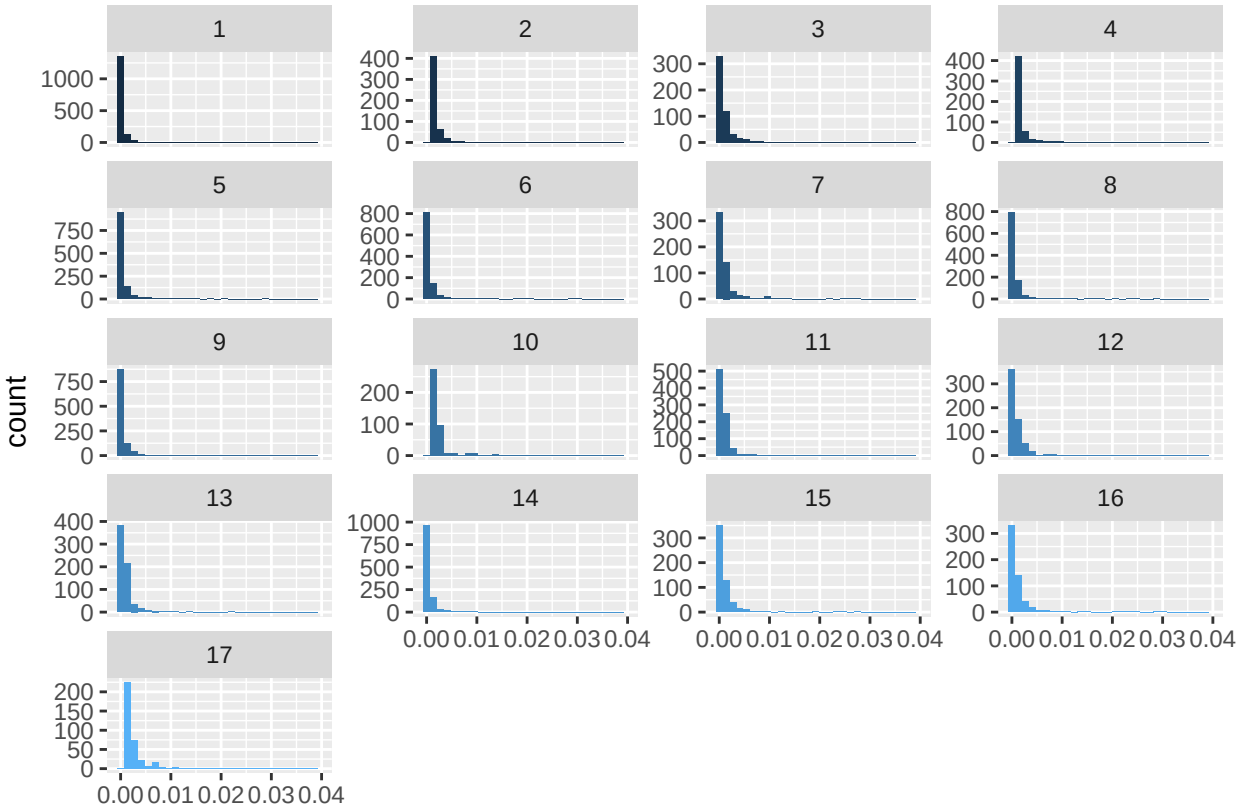
```
## Joining, by = "Article"
```

```
article_words
```

```
## # A tibble: 13,356 x 4
##    Article  word     n total
##      <int> <chr> <int> <int>
##  1       1   the   524  7938
##  2       6   the   312  4417
##  3      14   the   310  5383
##  4       1    of   302  7938
##  5       9   the   253  4542
##  6       5   the   250  4592
##  7       1   and   195  7938
##  8       8   the   188  3479
```

```
##  9       1    to   182  7938
## 10      14    of   160  5383
## # ... with 13,346 more rows
```

```
ggplot(article_words, aes(n/total, fill = Article)) +
  geom_histogram(show.legend = FALSE) +
  facet_wrap(~Article, ncol = 4, scales = "free_y") +
  xlim(NA, 0.04)
```




```
article_words <- article_words %>%
  bind_tf_idf(word, Article, n)
```

```
myStopWords <- data_frame(word = c("_____", "193", "502", "d", "95", "2.5", "2006", "california", "ralph
```

```
article_words %>%
  anti_join(myStopWords) %>%
  select(-total) %>%
  arrange(desc(tf_idf)) %>%
  top_n(20)
```

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

```
## Selecting by tf_idf
```

```
## # A tibble: 20 x 6
##    Article       word     n          tf       idf      tf_idf
##      <int>      <chr> <int>       <dbl>     <dbl>       <dbl>
##  1      13 confidence    28 0.011142061 2.1400662 0.023844748
##  2      13     sample    29 0.011539992 1.7346011 0.020017282
##  3      13      range    33 0.013131715 1.2237754 0.016070270
```

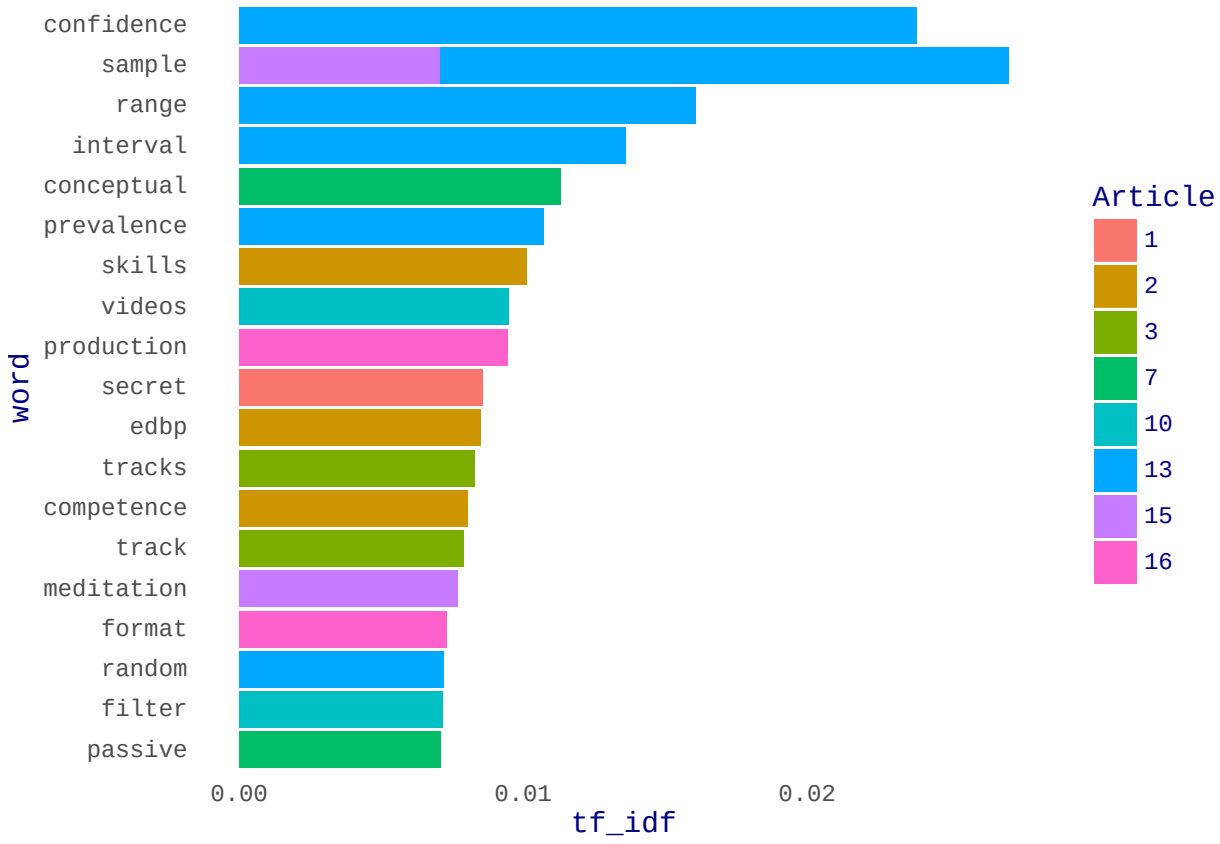
```
##  4      13   interval    16 0.006366892 2.1400662 0.013625570
##  5       7 conceptual     6 0.003994674 2.8332133 0.011317763
##  6      13 prevalence    22 0.008754477 1.2237754 0.010713514
##  7       2     skills    11 0.008264463 1.2237754 0.010113847
##  8      10     videos     3 0.003348214 2.8332133 0.009486205
##  9      16 production    23 0.014857881 0.6359888 0.009449446
## 10       1     secret    24 0.003023432 2.8332133 0.008566027
## 11       2       edbp     4 0.003005259 2.8332133 0.008514540
## 12       3     tracks     6 0.003880983 2.1400662 0.008305561
## 13       2 competence     5 0.003756574 2.1400662 0.008039317
## 14       3      track    10 0.006468305 1.2237754 0.007915753
## 15      15 meditation     4 0.002719239 2.8332133 0.007704183
## 16      16     format     4 0.002583979 2.8332133 0.007320965
## 17      13     random    24 0.009550338 0.7537718 0.007198776
## 18      10     filter     3 0.003348214 2.1400662 0.007165400
## 19       7    passive    12 0.007989348 0.8873032 0.007088974
## 20      15     sample     6 0.004078858 1.7346011 0.007075191
```

```
plot_article <- article_words %>%
  anti_join(myStopWords) %>%
  arrange(desc(tf_idf)) %>%
  mutate(word = factor(word, levels = rev(unique(word))))
```

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

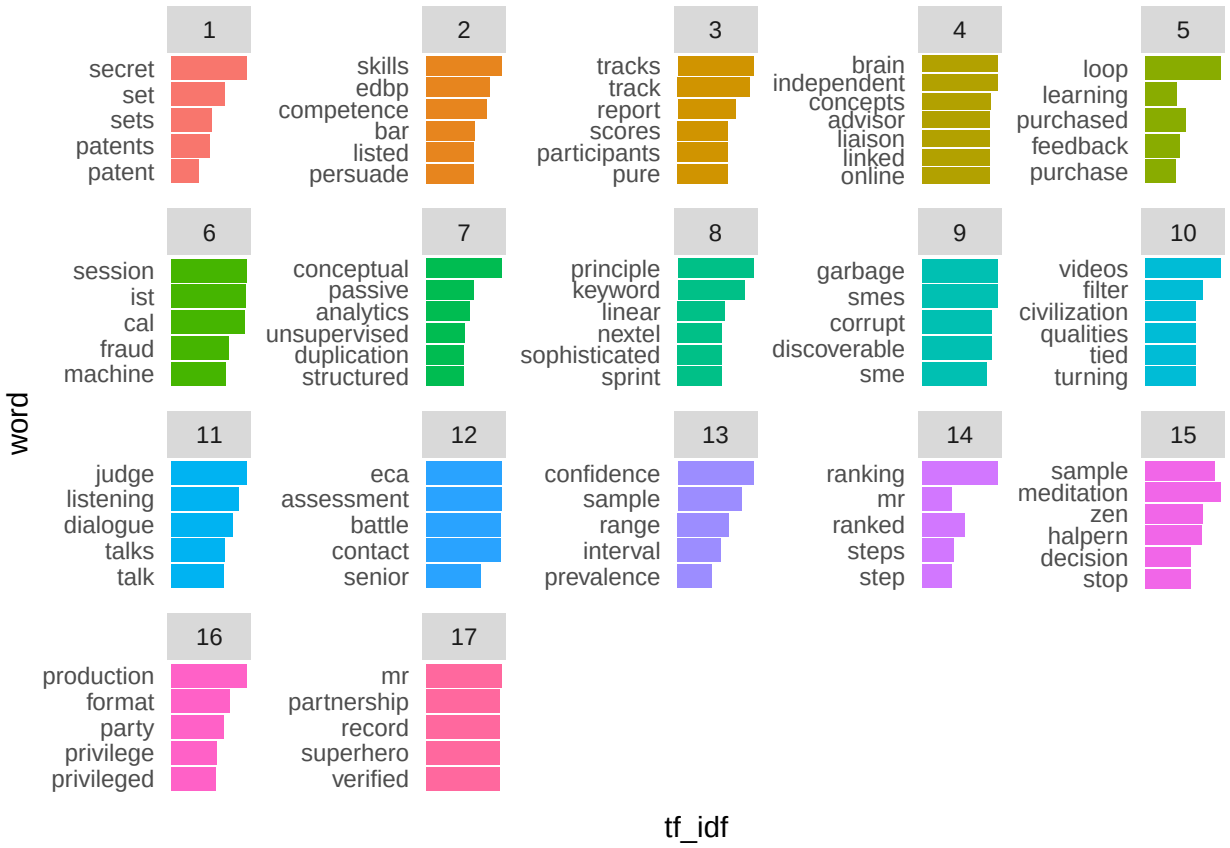
```
plot_article %>%
  top_n(20) %>%
  ggplot(aes(word, tf_idf, fill = factor(Article))) +
  geom_col()+
  scale_fill_discrete(guide = guide_legend(title = "Article")) +
  coord_flip() +
  theme(panel.background = element_blank(),
        text = element_text(family = "mono", color = "navy"),
        axis.ticks = element_blank())
```

```
## Selecting by tf_idf
```




```
plot_article %>%
  group_by(Article) %>%
  top_n(5) %>%
  ungroup() %>%
  ggplot(aes(word, tf_idf, fill = factor(Article))) +
    geom_col(show.legend = FALSE) +
    facet_wrap(~Article, ncol = 5, scales = "free") +
    coord_flip() +
    theme(panel.background = element_blank(),
          axis.text.x = element_blank(),
          axis.ticks = element_blank())
```

```
## Selecting by tf_idf
```


1
secret
set
sets
patents
patent
2
skills
edbp
competence
bar
listed
persuade
3
tracks
track
report
scores
participants
pure
4
brain
independent
concepts
advisor
liaison
linked
online
5
loop
learning
purchased
feedback
purchase
6
session
ist
cal
fraud
machine
7
conceptual
passive
analytics
unsupervised
duplication
structured
8
principle
keyword
linear
nextel
sophisticated
sprint
9
garbage
smes
corrupt
discoverable
sme
10
videos
filter
civilization
qualities
tied
turning
11
judge
listening
dialogue
talks
talk
12
eca
assessment
battle
contact
senior
13
confidence
sample
range
interval
prevalence
14
ranking
mr
ranked
steps
step
15
sample
meditation
zen
halpern
decision
stop
16
production
format
party
privilege
privileged
17
mr
partnership
record
superhero
verified
word
tf_idf