Visualizations

Aaron Dantzler

2023-08-04

```
# 1. Load packages:
library(tidytext) # contains sentiment lexicons
```

```
## Warning: package 'tidytext' was built under R version 4.3.1
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.1

## Warning: package 'ggplot2' was built under R version 4.3.1

## Warning: package 'tidyr' was built under R version 4.3.1

## Warning: package 'stringr' was built under R version 4.3.1

## Warning: package 'forcats' was built under R version 4.3.1

## Warning: package 'lubridate' was built under R version 4.3.1

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
#for corpus prep
library(stringr)
library(tm)
```

```
## Warning: package 'tm' was built under R version 4.3.1
## Loading required package: NLP
##
## Attaching package: 'NLP'
##
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(stm)
```

```
## stm v1.3.6 successfully loaded. See ?stm for help.
##  Papers, resources, and other materials at structuraltopicmodel.com
```

```
library(quanteda)
```

```
## Package version: 3.3.1
## Unicode version: 13.0
## ICU version: 69.1
## Parallel computing: 8 of 8 threads used.
## See https://quanteda.io for tutorials and examples.
##
## Attaching package: 'quanteda'
##
## The following object is masked from 'package:tm':
##
##     stopwords
##
## The following objects are masked from 'package:NLP':
##
##     meta, meta<-
```

```
library(textdata)
```

```
## Warning: package 'textdata' was built under R version 4.3.1
```

```
#for visualizaiton
library(ggplot2)
library(psych)
```

```
## Warning: package 'psych' was built under R version 4.3.1
```

```
##
## Attaching package: 'psych'
##
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
library(clipr)
```

```
## Welcome to clipr. See ?write_clip for advisories on writing to the clipboard in R.
```

```r
# Set working directory and load data
data <- read.csv("D:\\Princeton\\BSPL\\norms_sent_afinn.csv")
```

```r
new_df <- data

frq_topic <- c(new_df$frq_topic_t1, new_df$frq_topic_t2, new_df$frq_topic_t3)

stacked_df <- data.frame(frq_topic)

stacked_df$frq <- c(new_df$frq_t1, new_df$frq_t2, new_df$frq_t3)
```

```r
time1 <- c(1)
num_repetitions <- 616
time1 <- rep(time1, times = num_repetitions)

time2 <- c(2)
time2 <- rep(time2, times = num_repetitions)

time3 <- c(3)
time3 <- rep(time3, times = num_repetitions)

time <- c(time1, time2, time3)

stacked_df$time <- time
```

```r
stacked_df$prolific <- c(new_df$prolific, new_df$prolific, new_df$prolific)
stacked_df$control <- c(new_df$control, new_df$control, new_df$control)
stacked_df$treatment <- c(new_df$treatment, new_df$treatment, new_df$treatment)
```

```r
stacked_df$treated <- ifelse(((data$control == "climate") &
                              (data$frq_topic_t1 == 1 | data$frq_topic_t1 == 2)),
                            0,
                      ifelse(((data$control == "health") &
                              (data$frq_topic_t1 == 4 | data$frq_topic_t1 == 5)),
                            0,
                       ifelse(((data$control == "politics") &
                              (data$frq_topic_t1 == 5 | data$frq_topic_t1 == 6)),
                            0, 1)))
```

```r
stacked_df$evidence <- ifelse((stacked_df$treated == 1) &
                                (stacked_df$treatment == "evidence"), 1, 0)

stacked_df$normevidence <- ifelse((stacked_df$treated == 1) &
                                (stacked_df$treatment == "normevidence"), 1, 0)

stacked_df$norm <- ifelse((stacked_df$treated == 1) &
                                (stacked_df$treatment == "norm"), 1, 0)
```

```r
stacked_df$doc_id <- 1:nrow(stacked_df)
```

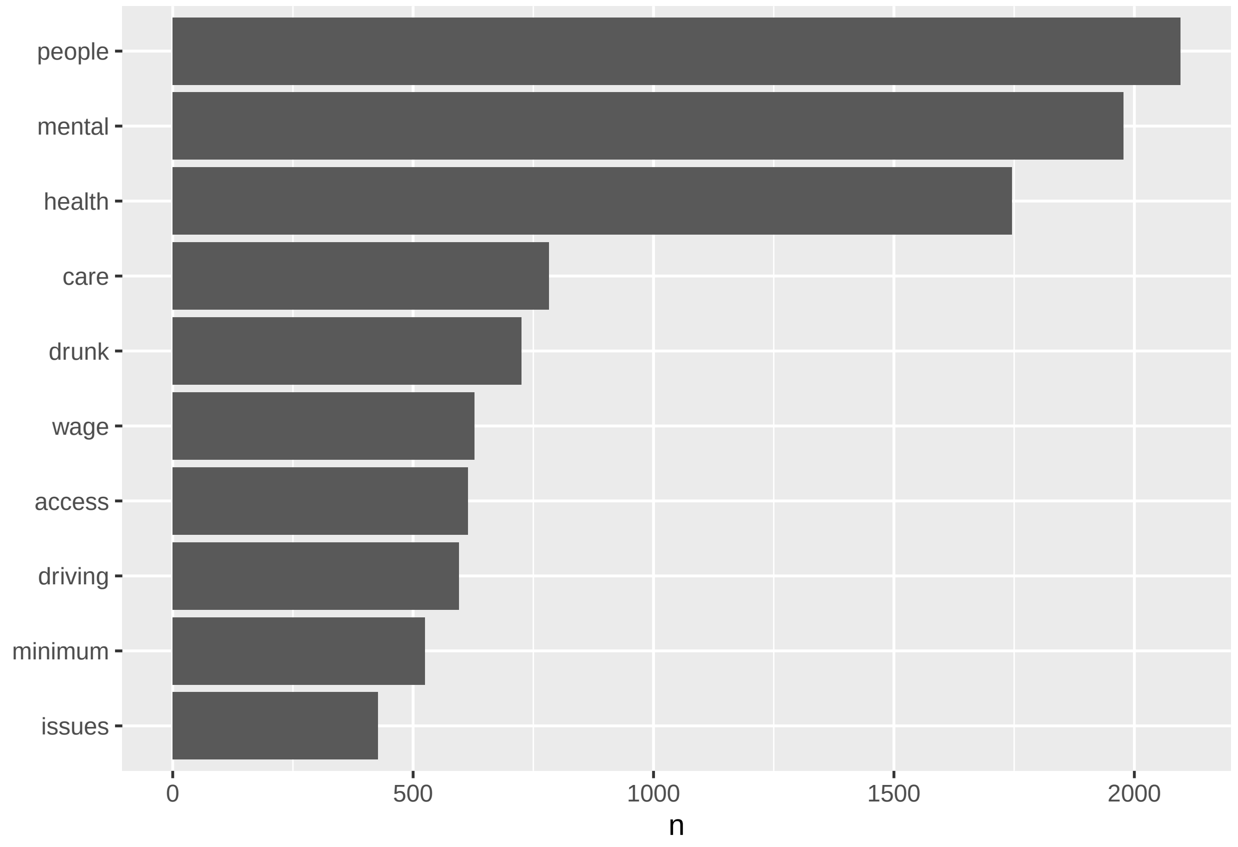
```
docs_df <- subset(stacked_df, select = c(frq, frq_topic, time, prolific, control, treatment, treated, e

colnames(docs_df)[colnames(docs_df) == "frq"] <- "text"
```

All Data

```
data(stop_words)

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

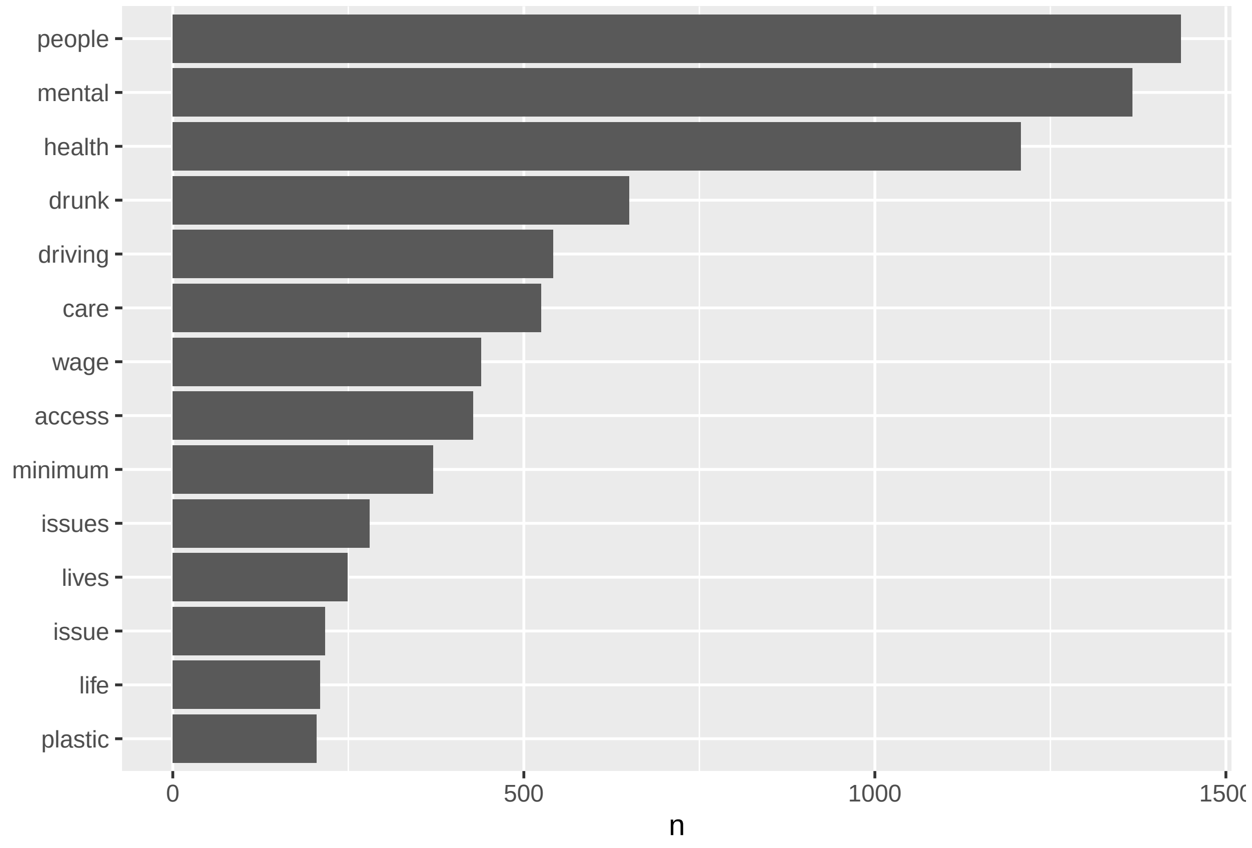
```
## Joining with `by = join_by(word)`
```

```
library(ggplot2)

tidy %>%
  count(word, sort = TRUE) %>%
  filter(n > 400) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```

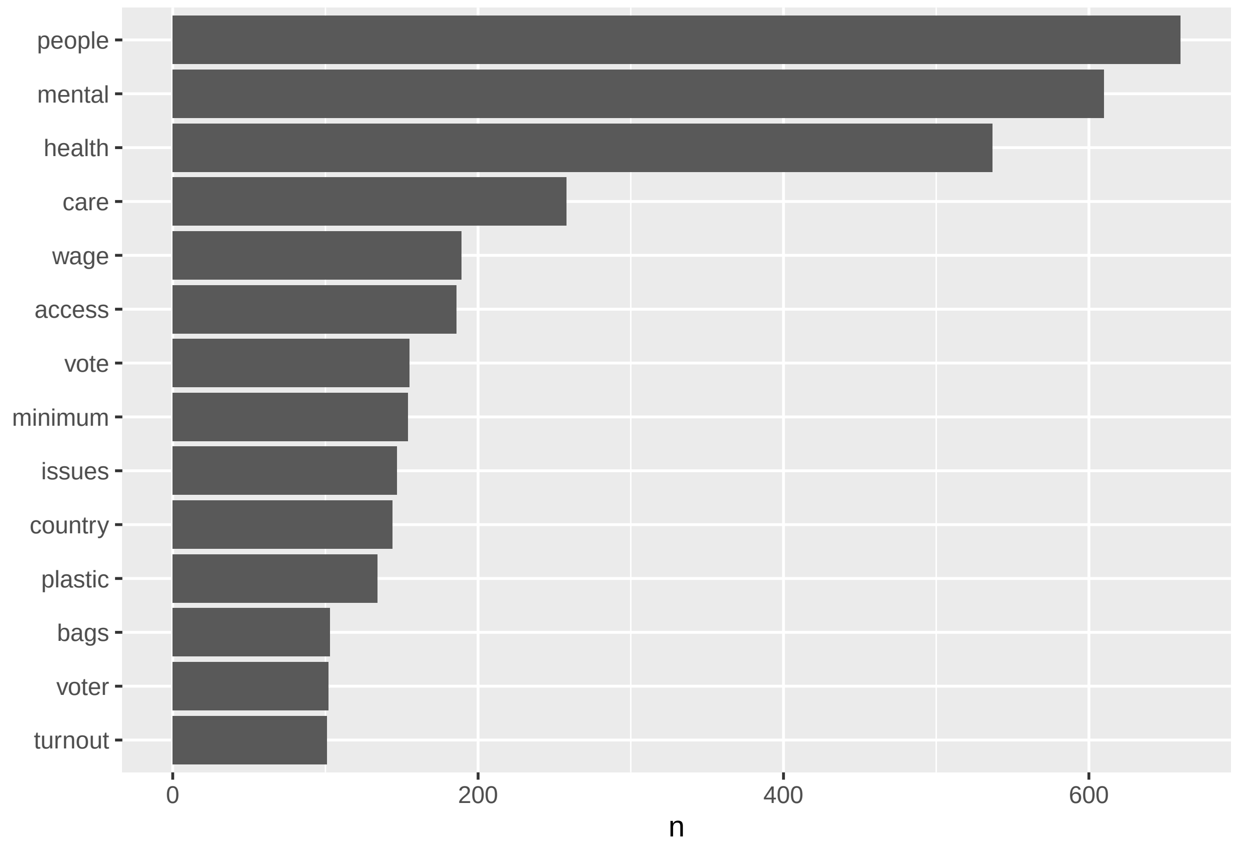


```r
tidy_treated <- subset(tidy, treated == 1)
tidy_untreated <- subset(tidy, treated == 0)
```

```r
library(ggplot2)

tidy_treated %>%
  count(word, sort = TRUE) %>%
  filter(n > 200) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```



```r
tidy_untreated %>%
  count(word, sort = TRUE) %>%
  filter(n > 100) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```

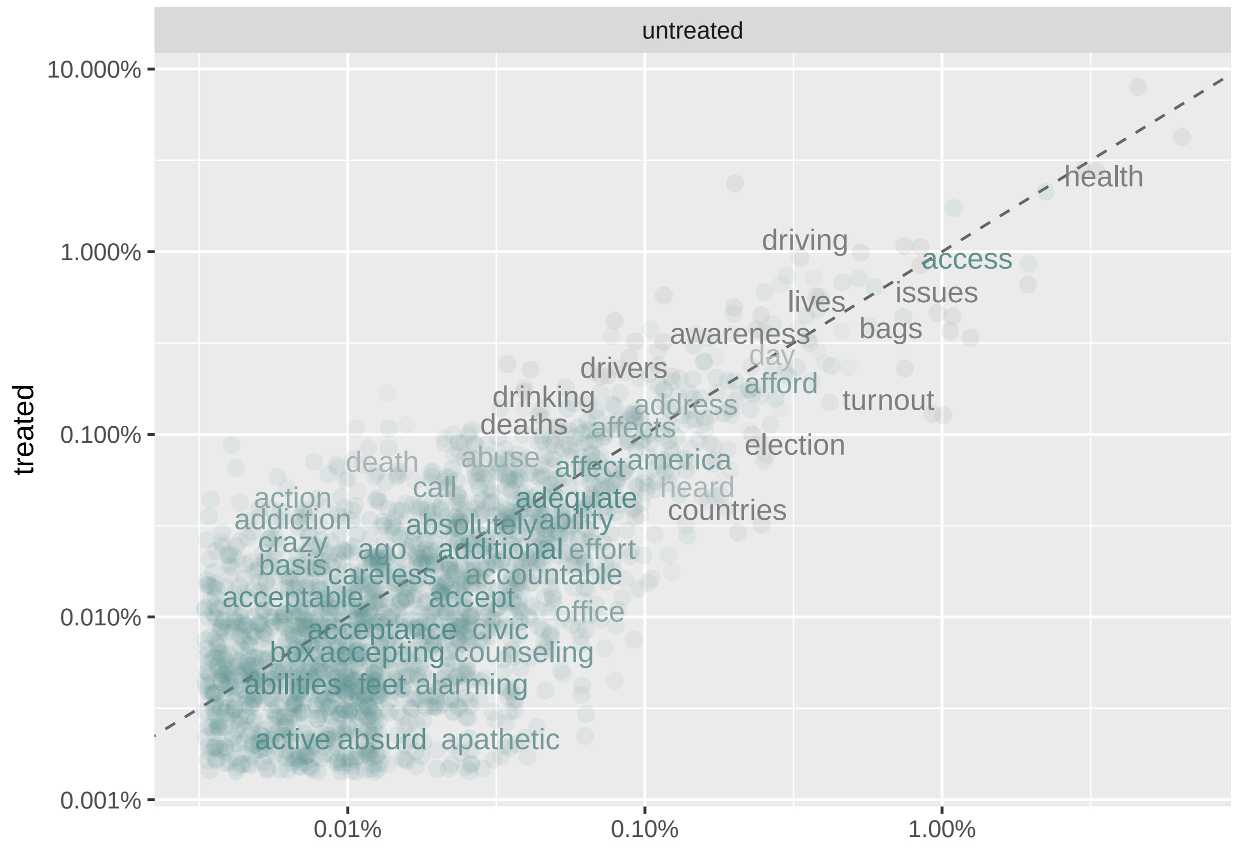


```r
library(scales)
```

```
## Warning: package 'scales' was built under R version 4.3.1

##
## Attaching package: 'scales'

## The following objects are masked from 'package:psych':
##
##     alpha, rescale

## The following object is masked from 'package:purrr':
##
##     discard

## The following object is masked from 'package:readr':
##
##     col_factor
```

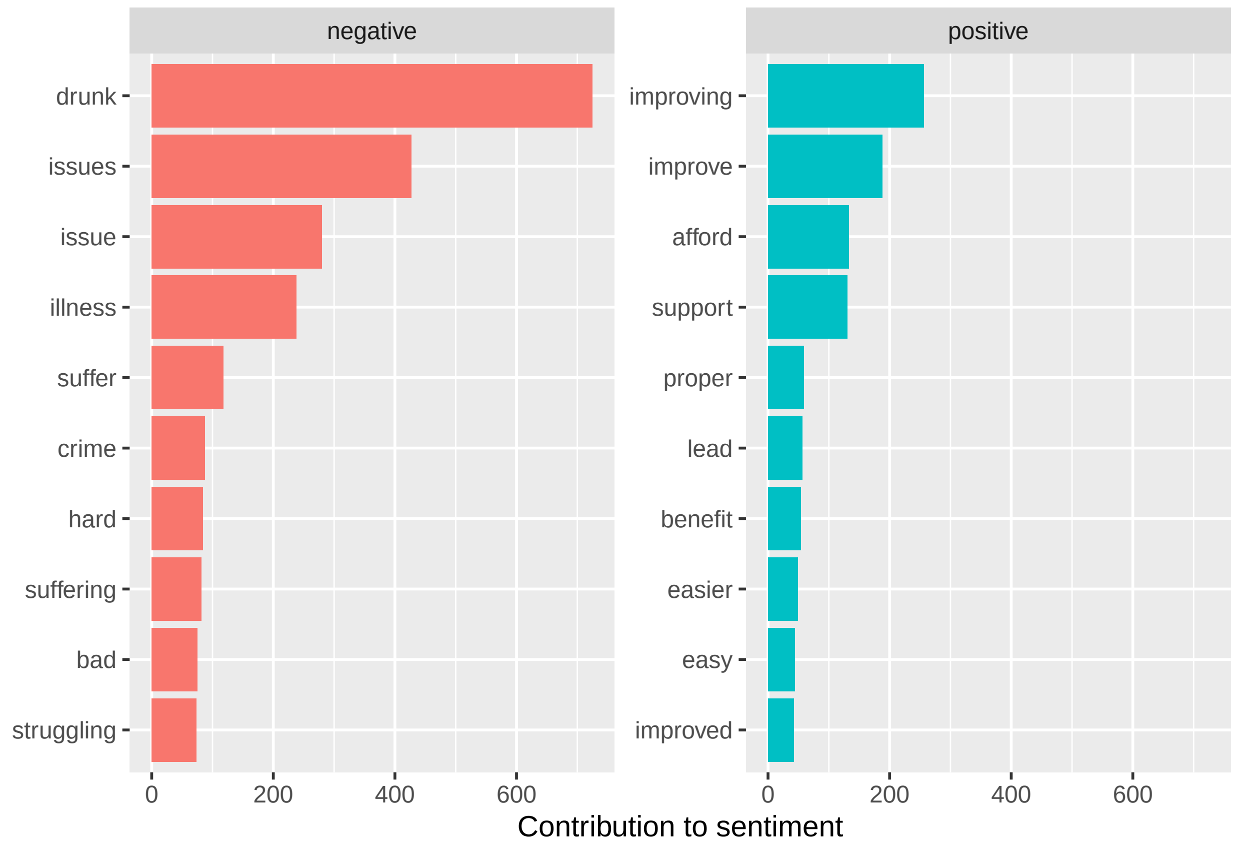
```r
# expect a warning about rows with missing values being removed
ggplot(frequency, aes(x = proportion, y = `treated`,
                      color = abs(`treated` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  scale_color_gradient(limits = c(0, 0.001),
                       low = "darkslategray4", high = "gray75") +
  facet_wrap(~treatment, ncol = 2) +
  theme(legend.position="none") +
  labs(y = "treated", x = NULL)
```

```
## Warning: Removed 3688 rows containing missing values ('geom_point()').

## Warning: Removed 3689 rows containing missing values ('geom_text()').
```

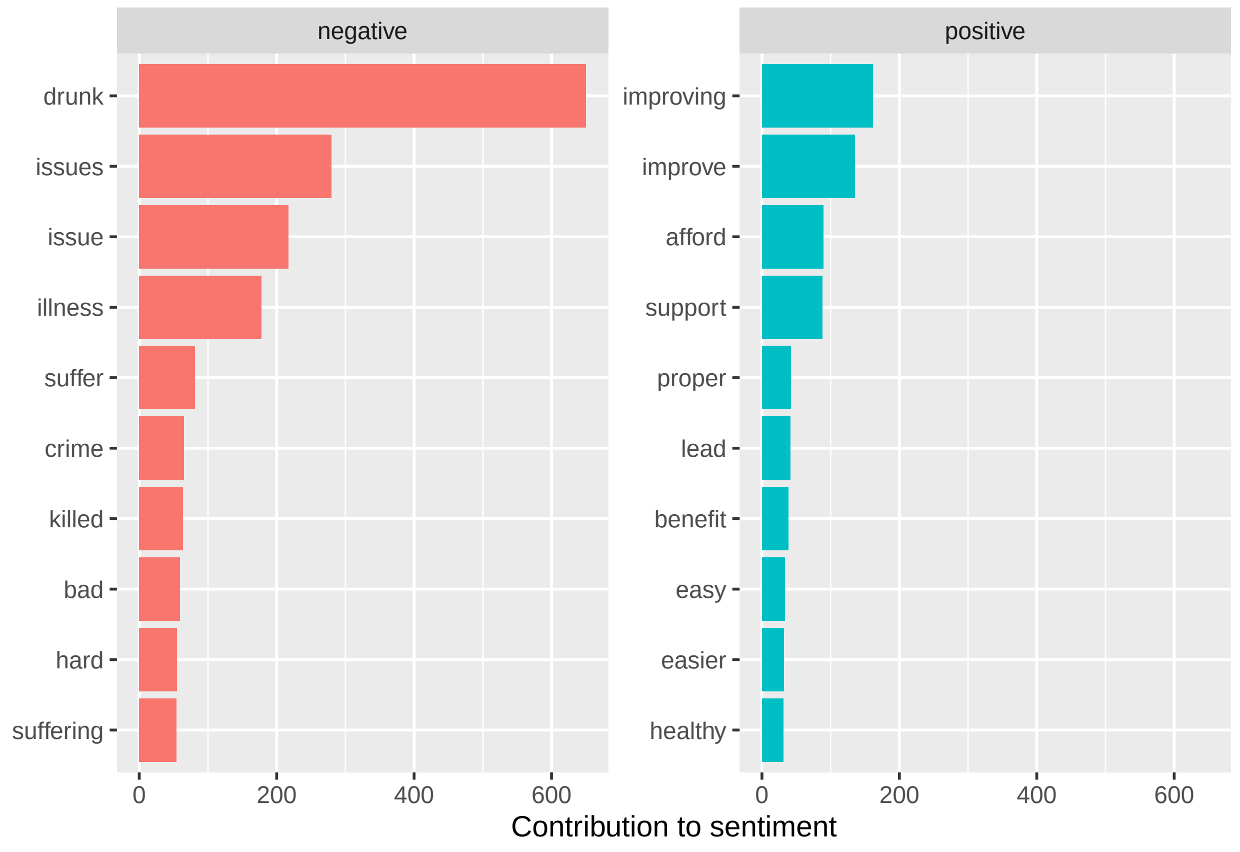


Words above the line are associated more with treated. Below associated with untreated.

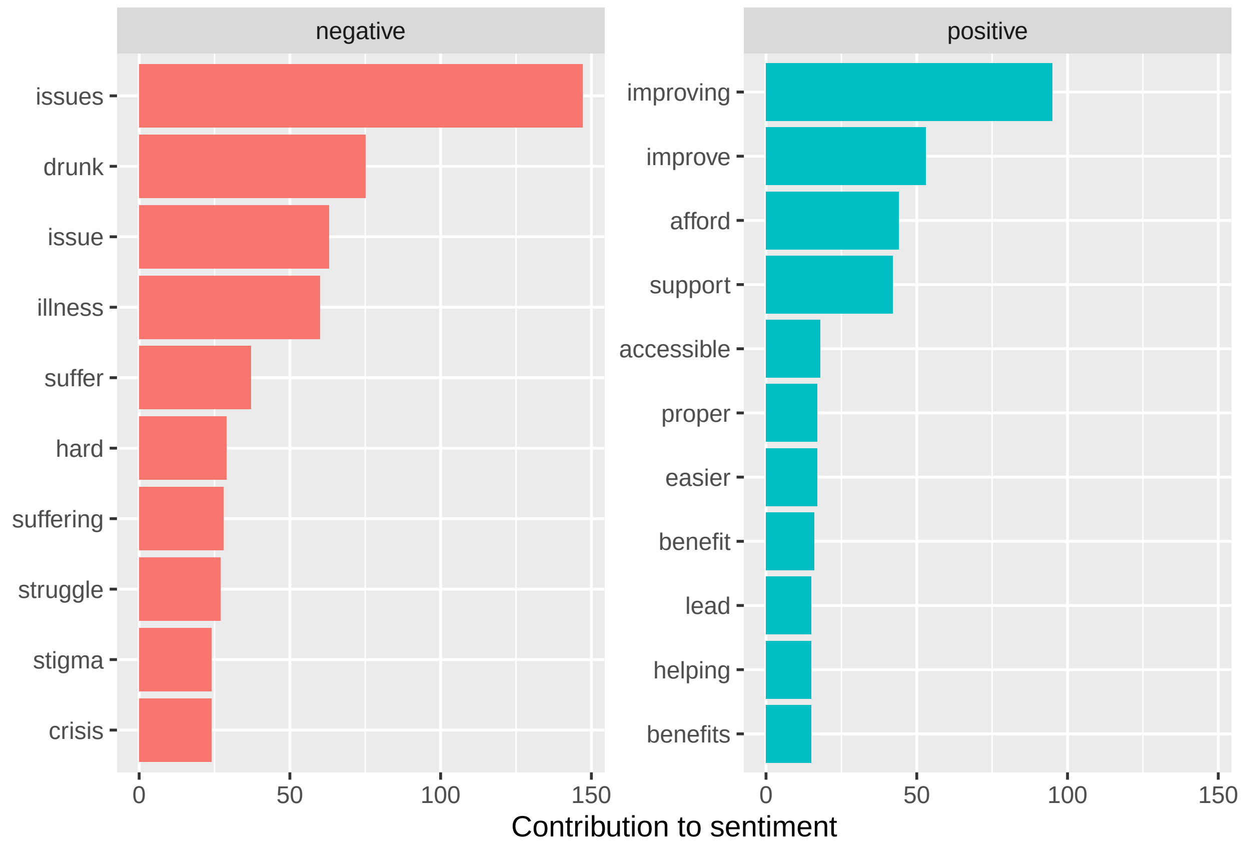
```
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```



```
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```



```r
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```



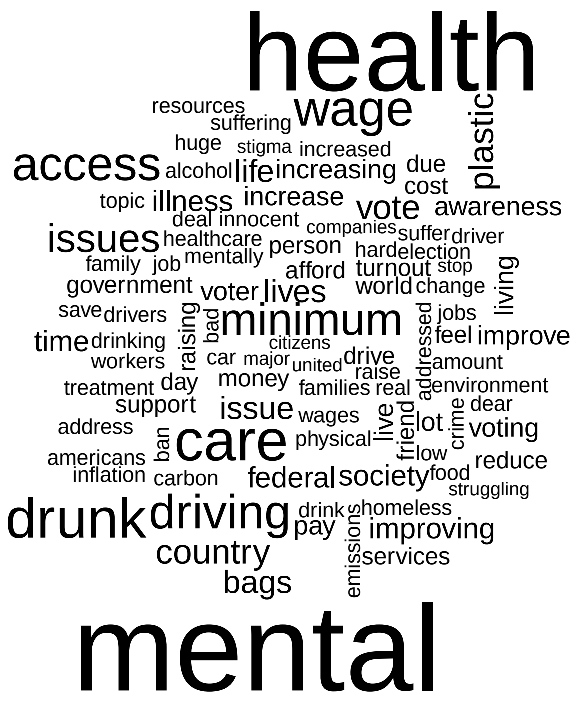
```
library(wordcloud)
```

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

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

```
## Joining with ‘by = join_by(word)‘
```

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

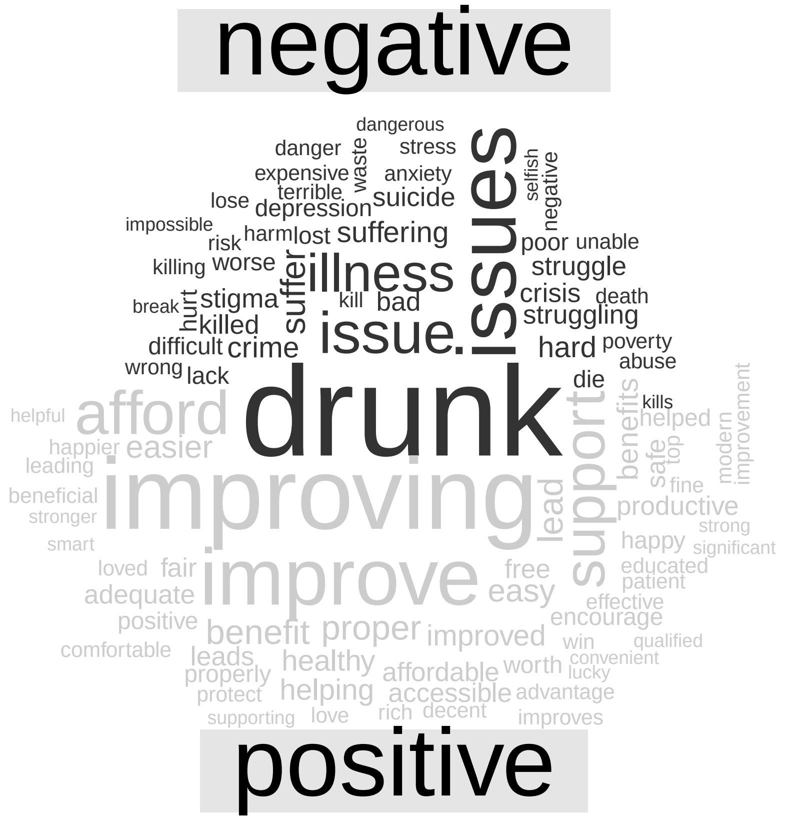


```r
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 4.3.1

##
## Attaching package: 'reshape2'

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

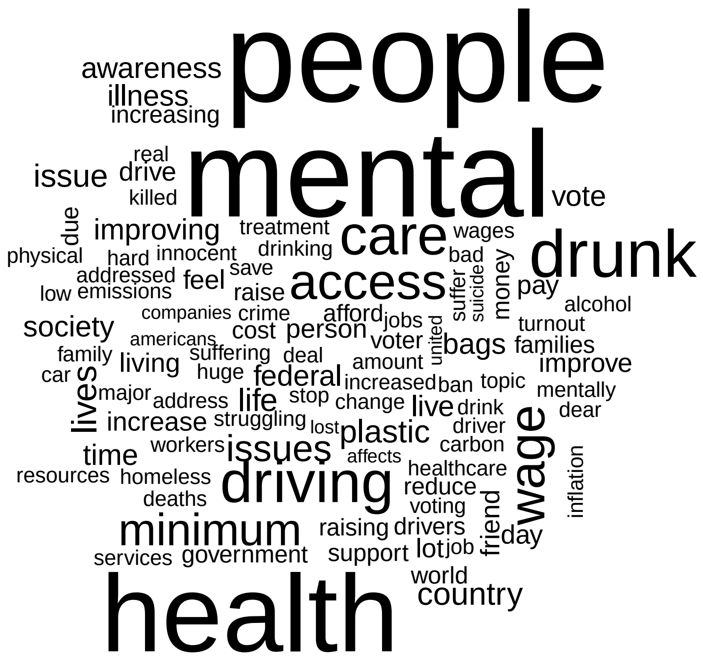
```r
tidy %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

```
## Joining with `by = join_by(word)`

## Warning in inner_join(., get_sentiments("bing")): Detected an unexpected many-to-many relationship b
## i Row 20039 of `x` matches multiple rows in `y`.
## i Row 3621 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.
```



```
library(wordcloud)

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

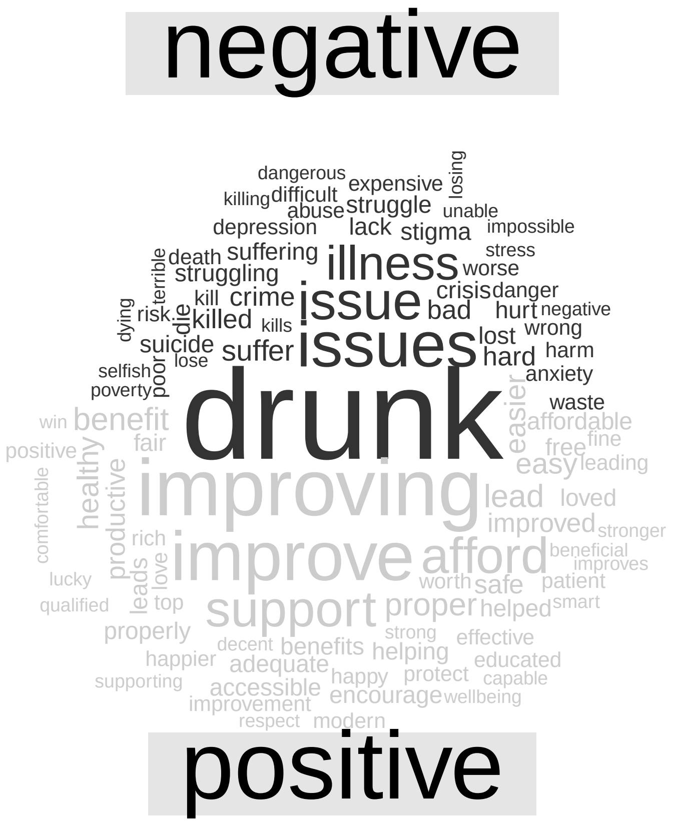
```
## Joining with 'by = join_by(word)'
```



```
library(reshape2)

tidy_treated %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

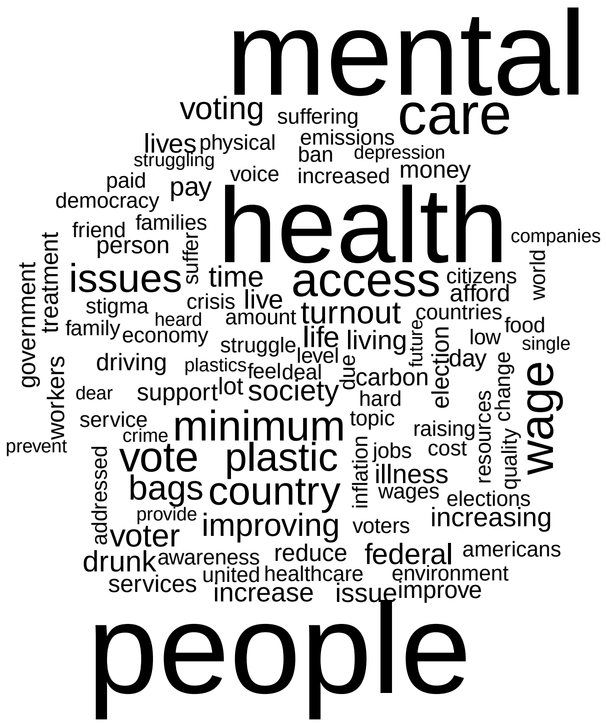
```
## Joining with ‘by = join_by(word)‘
```

```
## Warning in inner_join(., get_sentiments("bing")): Detected an unexpected many-to-many relationship b
## i Row 13807 of ‘x‘ matches multiple rows in ‘y‘.
## i Row 3621 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.
```



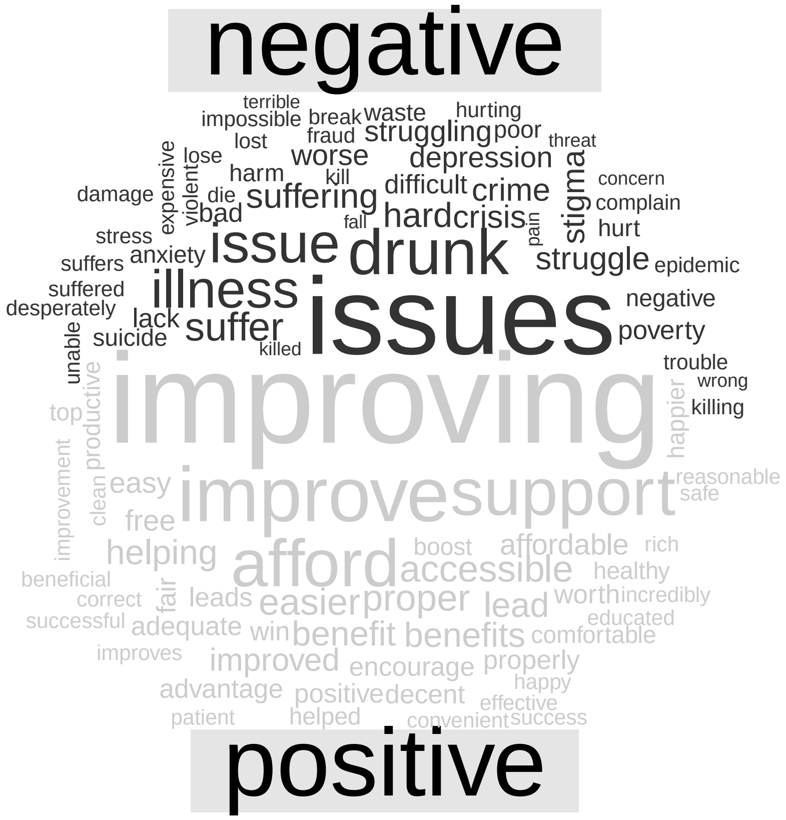
```r
library(wordcloud)

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

```
## Joining with `by = join_by(word)`
```



```
library(reshape2)

tidy_untreated %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

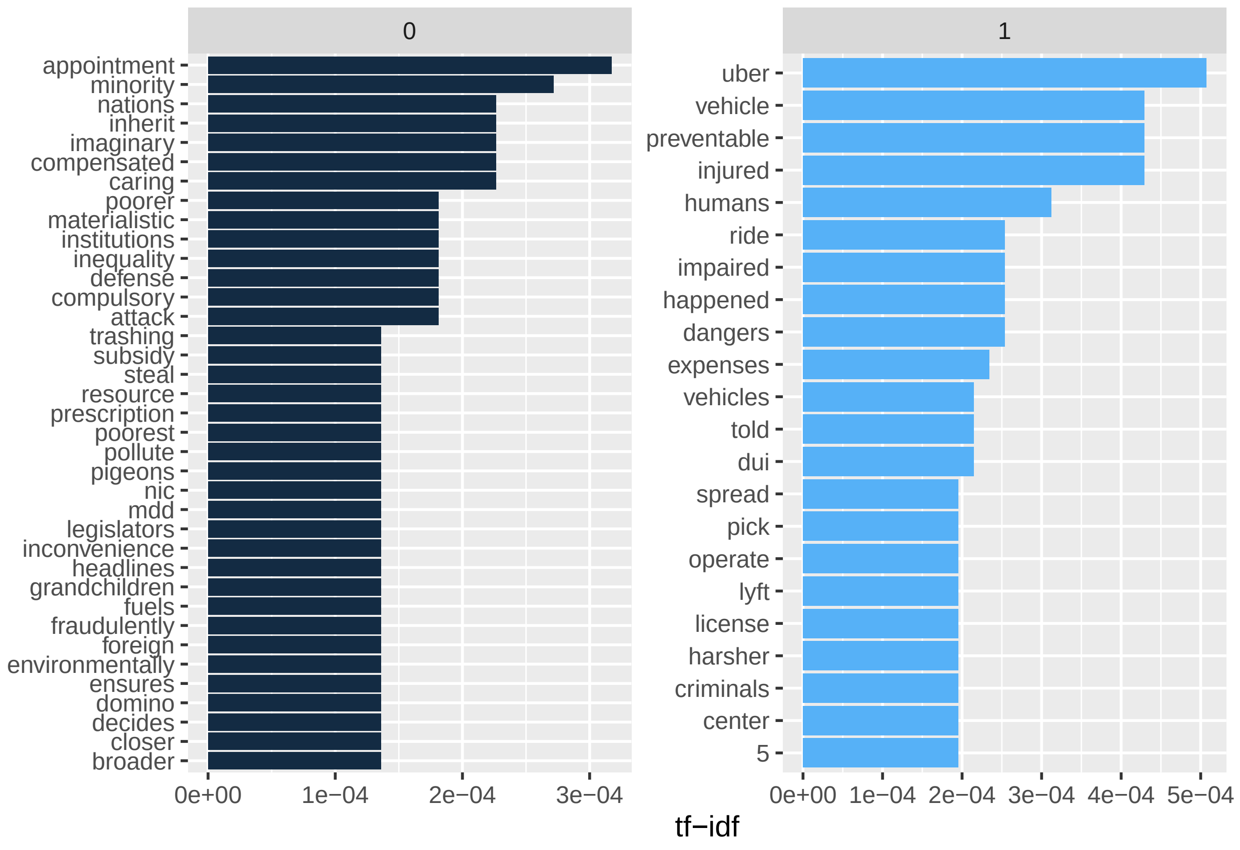
```
## Joining with 'by = join_by(word)'
```



tf_idf, term frequency and inverse document frequency tells us the words that are important for a document but are not important for the corpus as a whole. Filters out common words that many documents use.

```
tidy_words <- tidy %>%
  count(treated, word, sort = TRUE)
```

```
library(forcats)

tidy_tf_idf %>%
  group_by(treated) %>%
  slice_max(tf_idf, n = 15) %>%
  ungroup() %>%
  ggplot(aes(tf_idf, fct_reorder(word, tf_idf), fill = treated)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~treated, ncol = 2, scales = "free") +
  labs(x = "tf-idf", y = NULL)
```

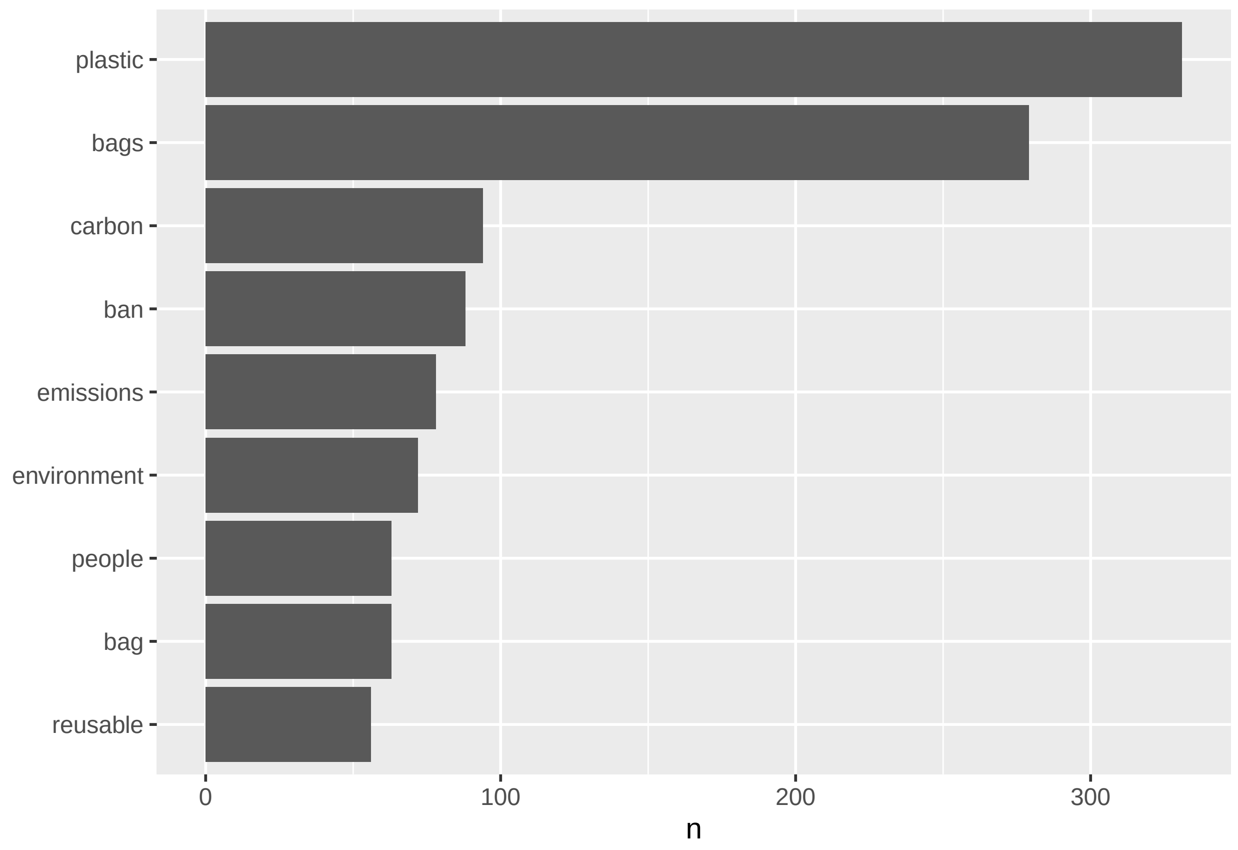


Climate Only

```
data(stop_words)

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

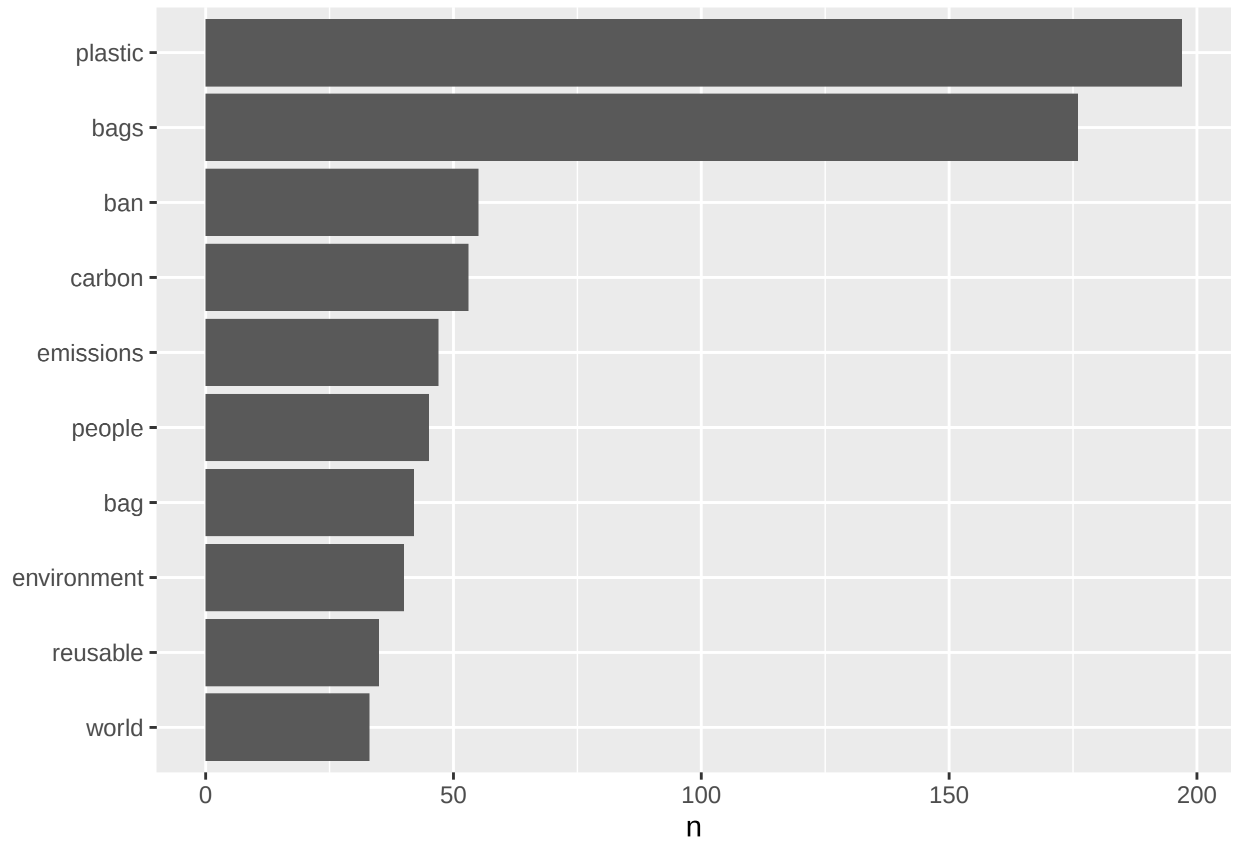
```
## Joining with ‘by = join_by(word)‘
```

```
library(ggplot2)

tidy %>%
  count(word, sort = TRUE) %>%
  filter(n > 50) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```

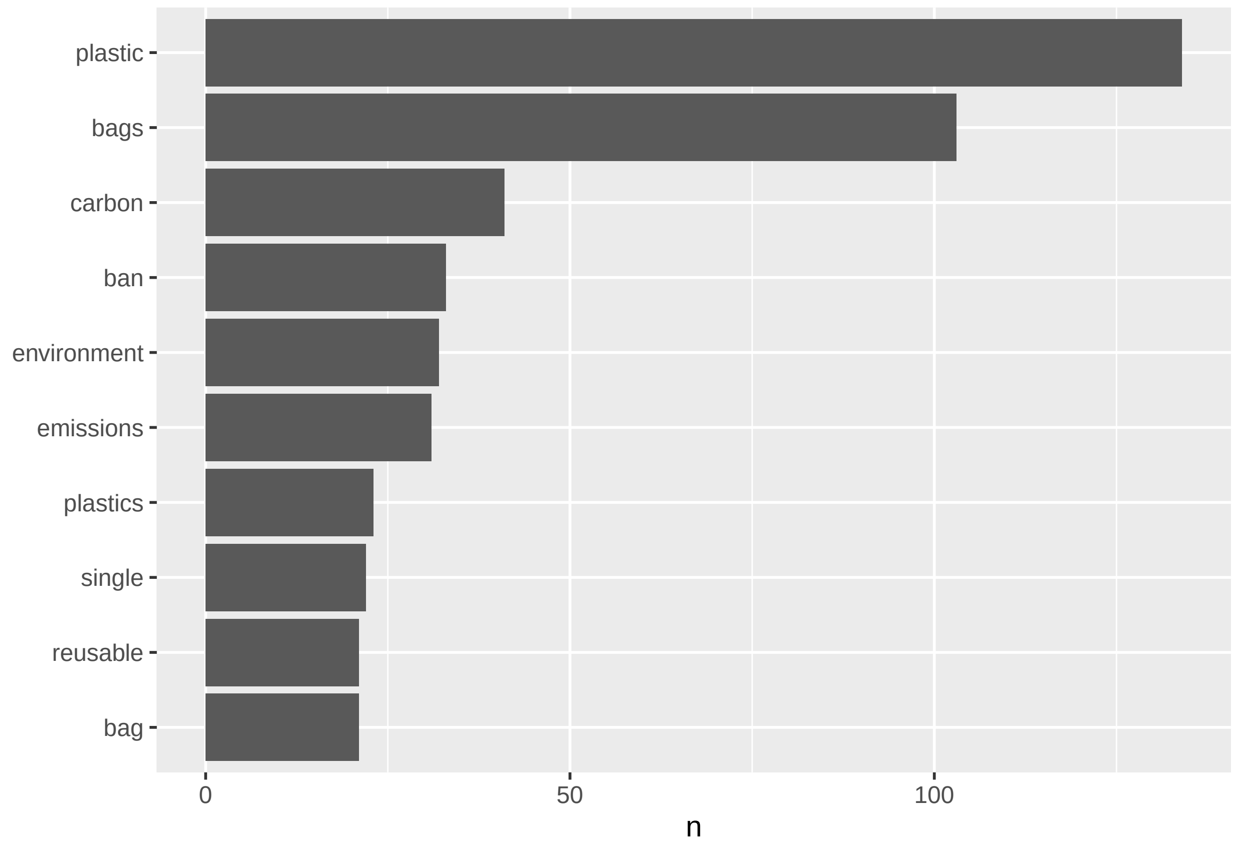


```
tidy_treated <- subset(tidy, treated == 1)
tidy_untreated <- subset(tidy, treated == 0)
```

```
library(ggplot2)

tidy_treated %>%
  count(word, sort = TRUE) %>%
  filter(n > 30) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```

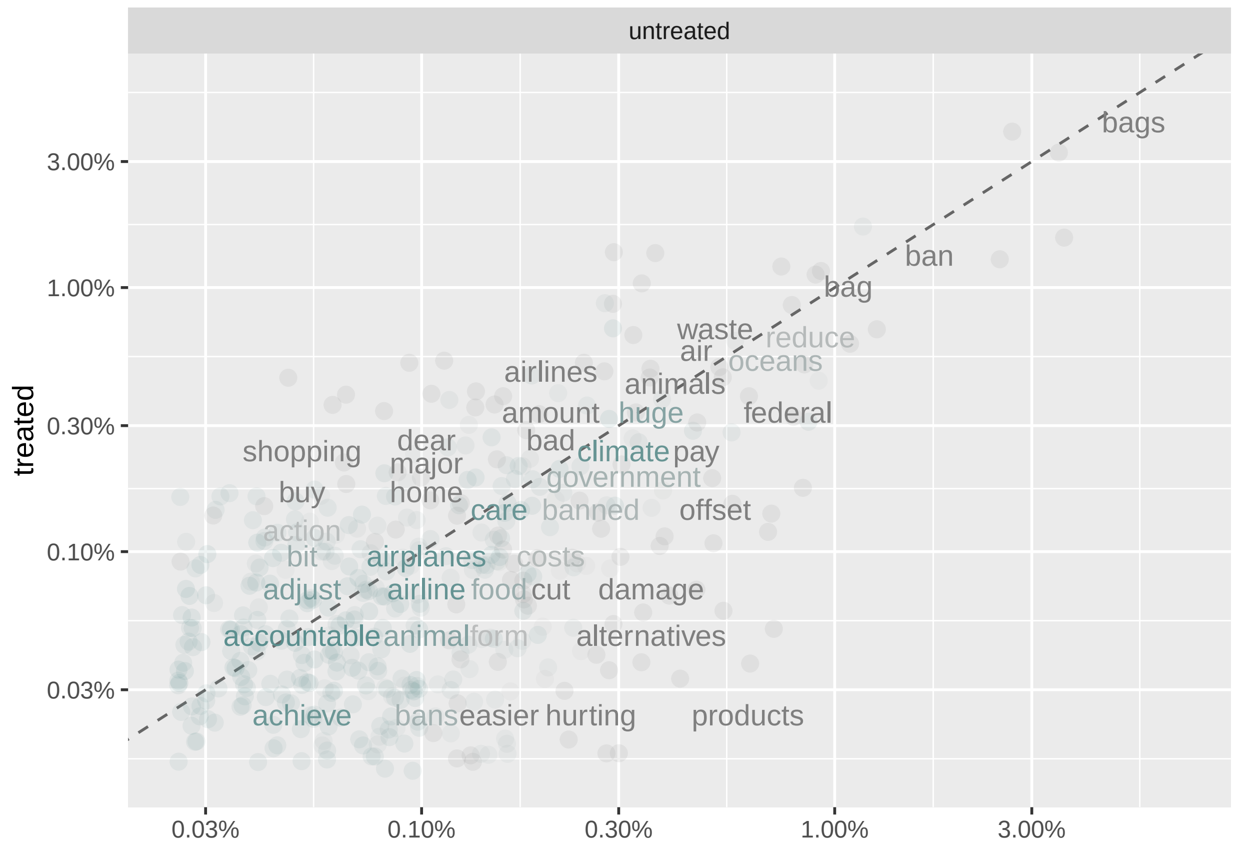


```
tidy_untreated %>%
  count(word, sort = TRUE) %>%
  filter(n > 20) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```



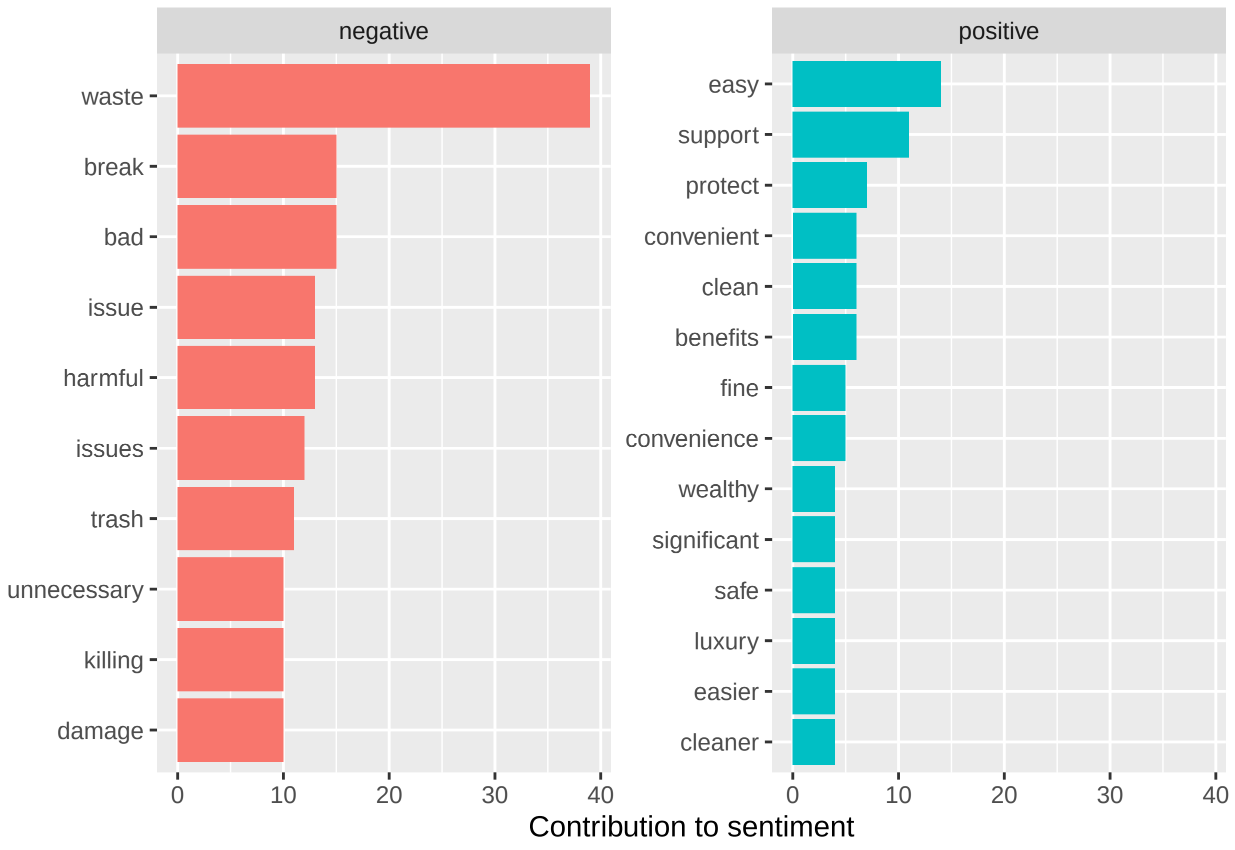
```
library(scales)

# expect a warning about rows with missing values being removed
ggplot(frequency, aes(x = proportion, y = `treated`,
                      color = abs(`treated` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  scale_color_gradient(limits = c(0, 0.001),
                       low = "darkslategray4", high = "gray75") +
  facet_wrap(~treatment, ncol = 2) +
  theme(legend.position="none") +
  labs(y = "treated", x = NULL)
```

```
## Warning: Removed 1181 rows containing missing values (‘geom_point()‘).

## Warning: Removed 1182 rows containing missing values (‘geom_text()‘).
```

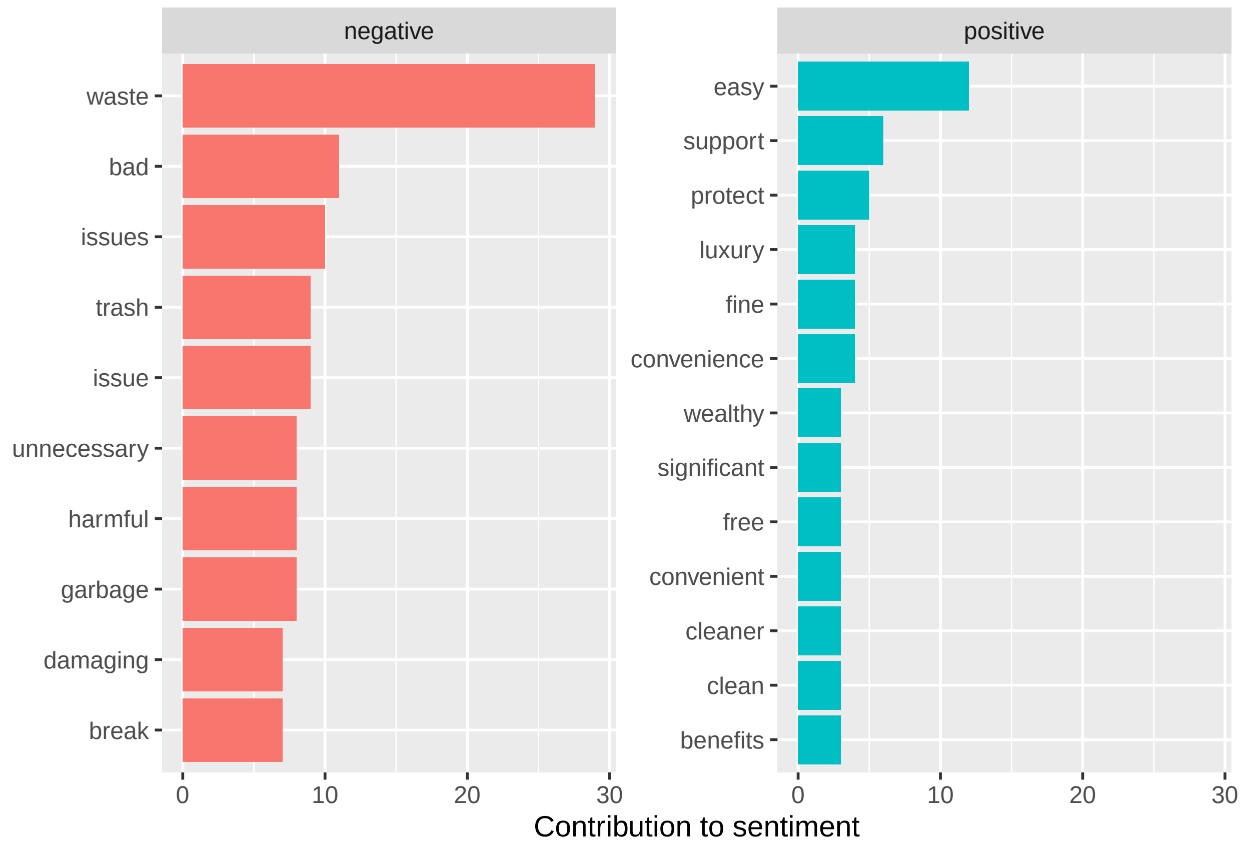


Words above the line are associated more with treated. Below associated with untreated.

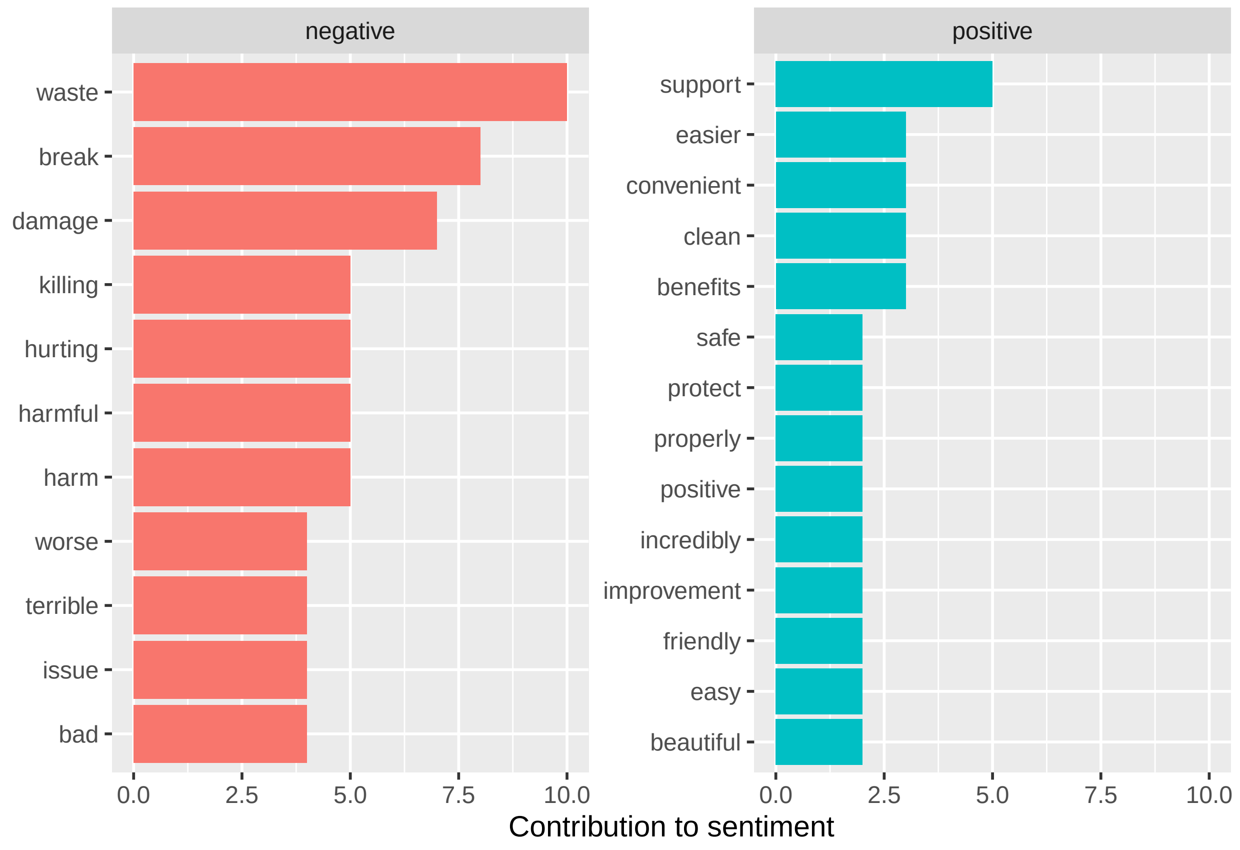
```
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```



```
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```

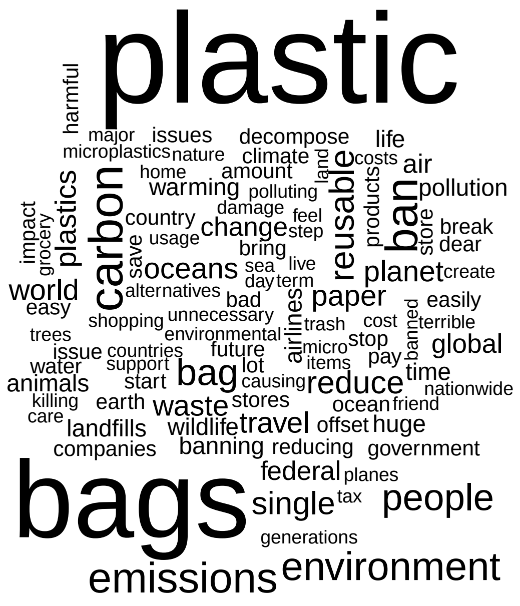


```
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```



```
library(wordcloud)

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

```
## Joining with ‘by = join_by(word)‘
```



```
library(reshape2)

tidy %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

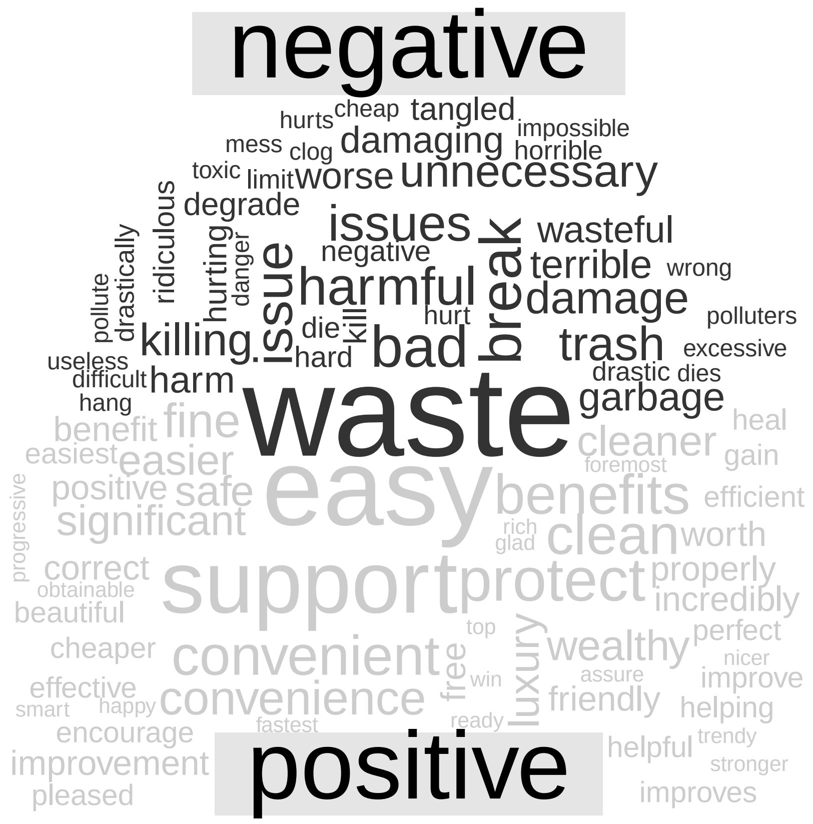
```
## Joining with ‘by = join_by(word)‘
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): progress could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): sustainable could not be fit on page. It will not be plotted.
```

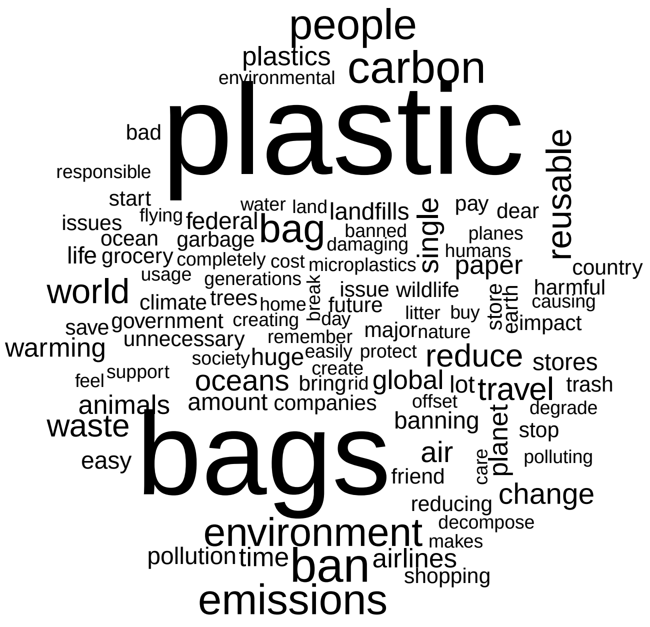
```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): improvements could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): sustainability could not be fit on page. It will not be plotted.
```



```
library(wordcloud)

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

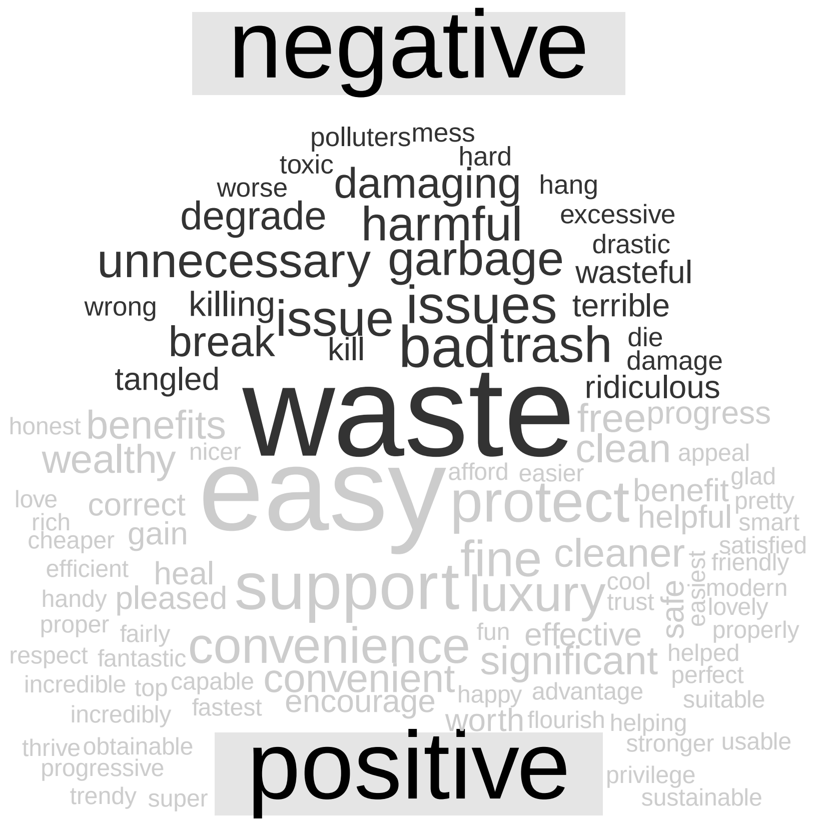
```
## Joining with 'by = join_by(word)'
```



```
library(reshape2)

tidy_treated %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

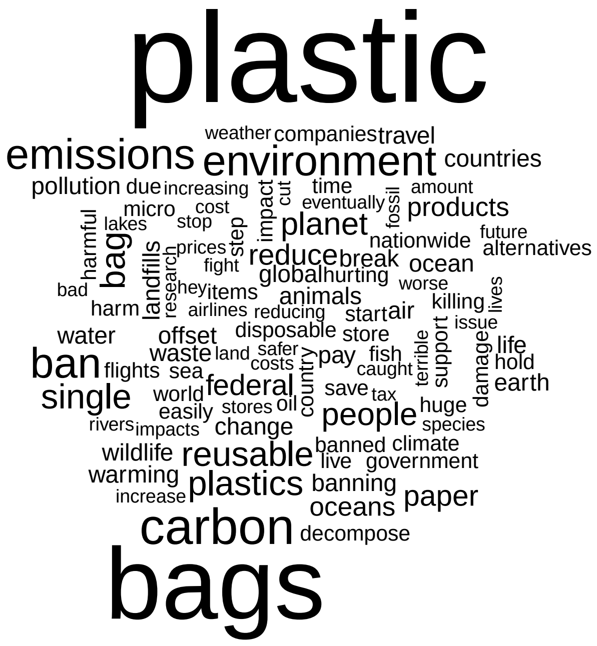
```
## Joining with ‘by = join_by(word)‘
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): useable could not be fit on page. It will not be plotted.
```



```r
library(wordcloud)

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

```
## Joining with ‘by = join_by(word)‘
```



```r
library(reshape2)

tidy_untreated %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

```
## Joining with ‘by = join_by(word)‘
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): improvement could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): convenience could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): efficiently could not be fit on page. It will not be plotted.
```

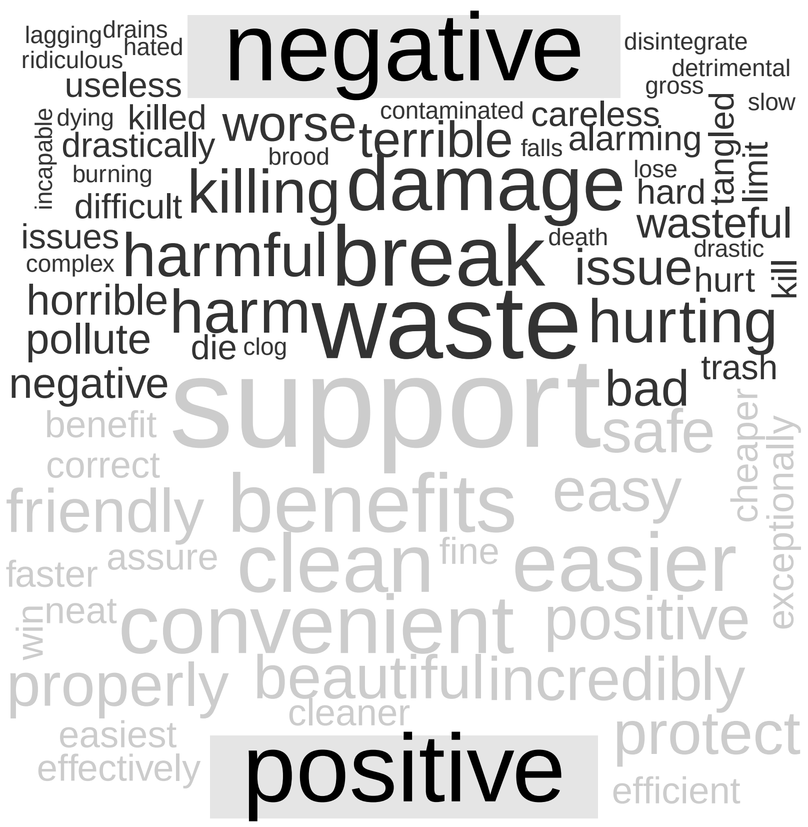
```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): foremost could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): helping could not be fit on page. It will not be plotted.
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): improve could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): improvements could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): improves could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): perfect could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): preferably could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): significant could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): smarter could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): sufficient could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): sustainable could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): wealthy could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): worth could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): inconvenience could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): unnecessary could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): irreparable could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): irreversible could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): strangle could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): suffering could not be fit on page. It will not be plotted.
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): suffocate could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): unable could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): worrisome could not be fit on page. It will not be plotted.
```

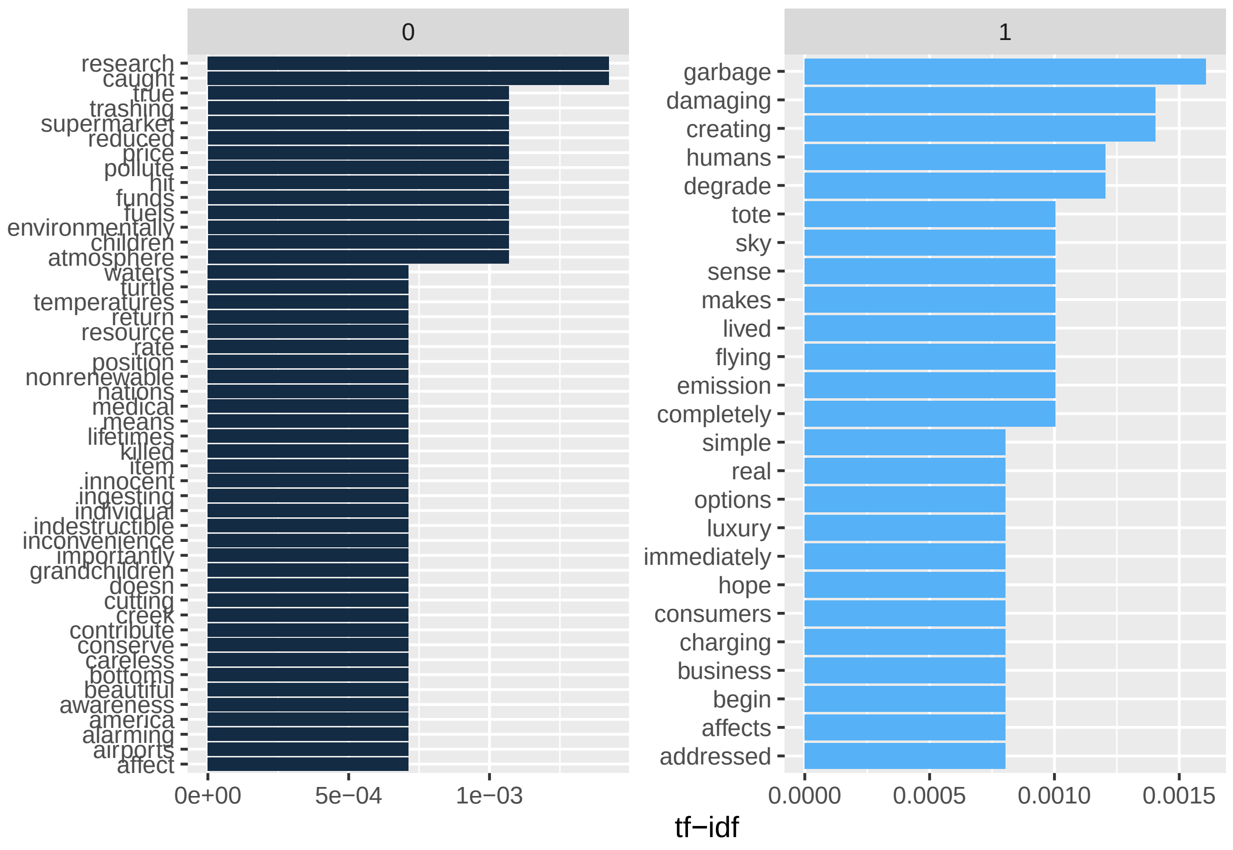
```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): worry could not be fit on page. It will not be plotted.
```



tf_idf, term frequency and inverse document frequency tells us the words that are important for a document but are not important for the corpus as a whole. Filters out common words that many documents use.

```r
tidy_words <- tidy %>%
  count(treated, word, sort = TRUE)
```

```
library(forcats)

tidy_tf_idf %>%
  group_by(treated) %>%
  slice_max(tf_idf, n = 15) %>%
  ungroup() %>%
  ggplot(aes(tf_idf, fct_reorder(word, tf_idf), fill = treated)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~treated, ncol = 2, scales = "free") +
  labs(x = "tf-idf", y = NULL)
```

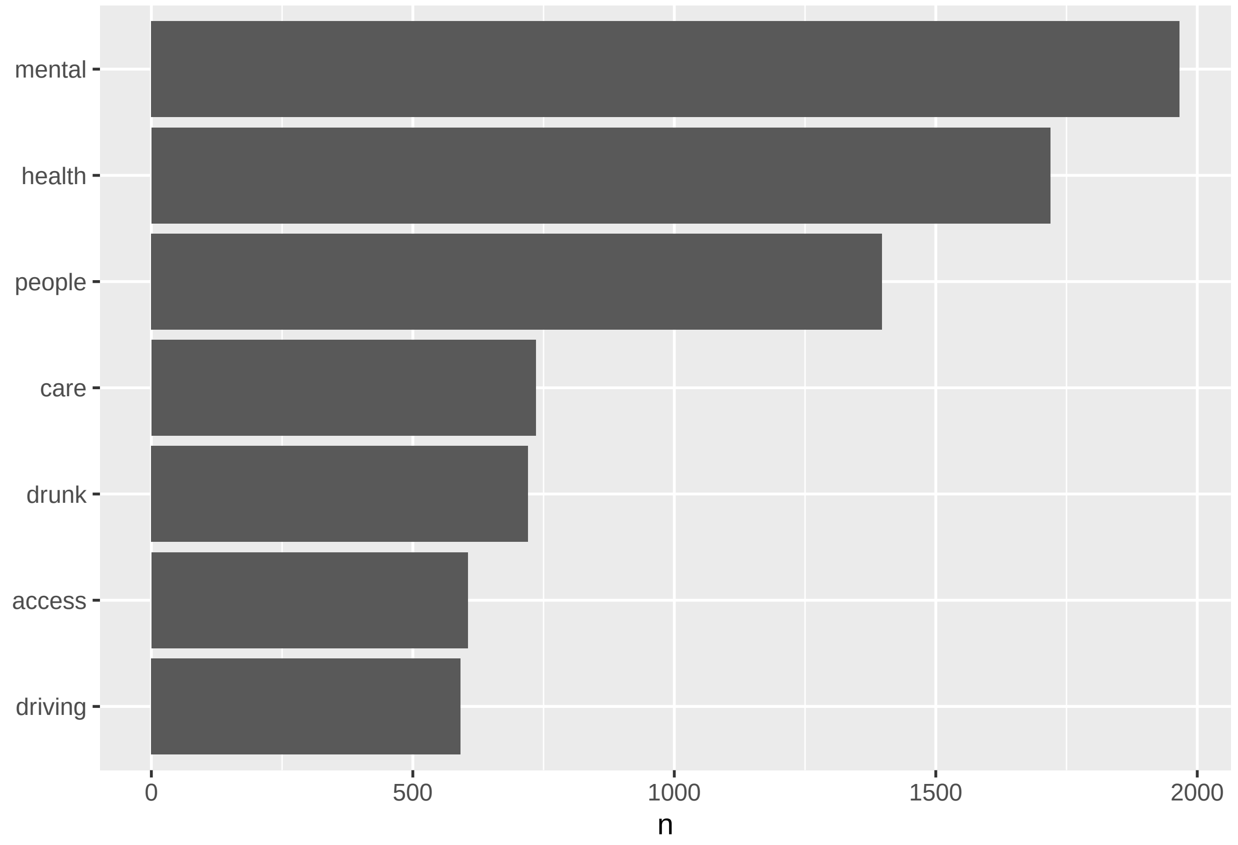


Health Only

```
data(stop_words)

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

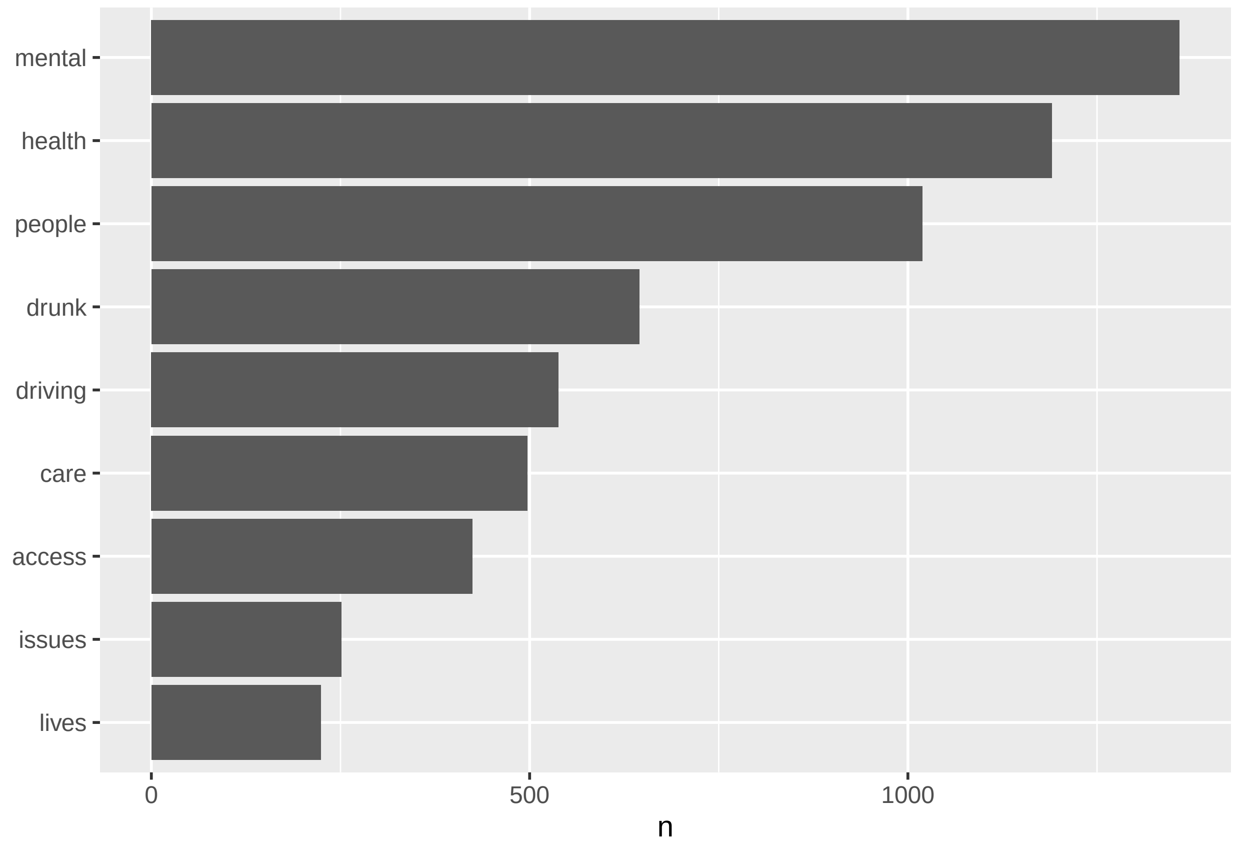
```
## Joining with ‘by = join_by(word)‘
```

```
library(ggplot2)

tidy %>%
  count(word, sort = TRUE) %>%
  filter(n > 400) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```

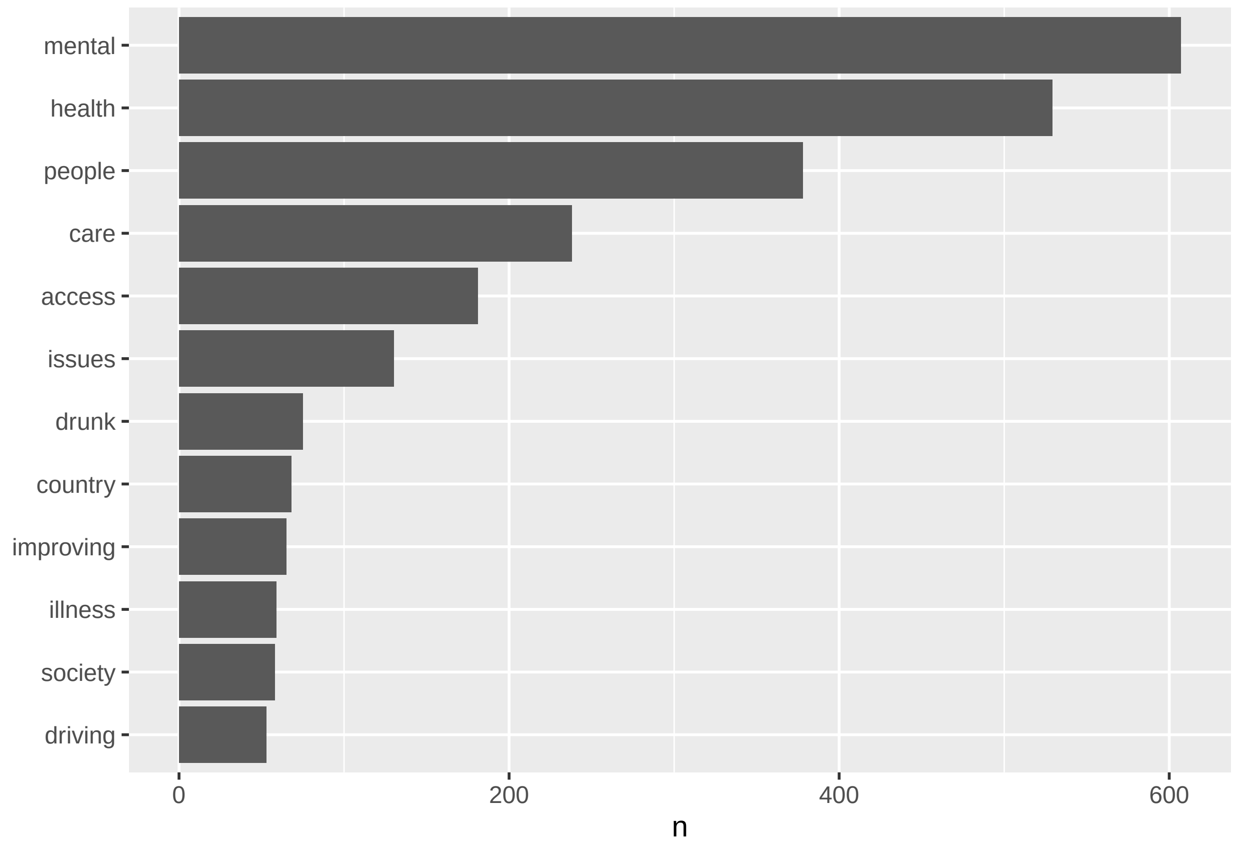


```
tidy_treated <- subset(tidy, treated == 1)
tidy_untreated <- subset(tidy, treated == 0)
```

```
library(ggplot2)

tidy_treated %>%
  count(word, sort = TRUE) %>%
  filter(n > 200) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```



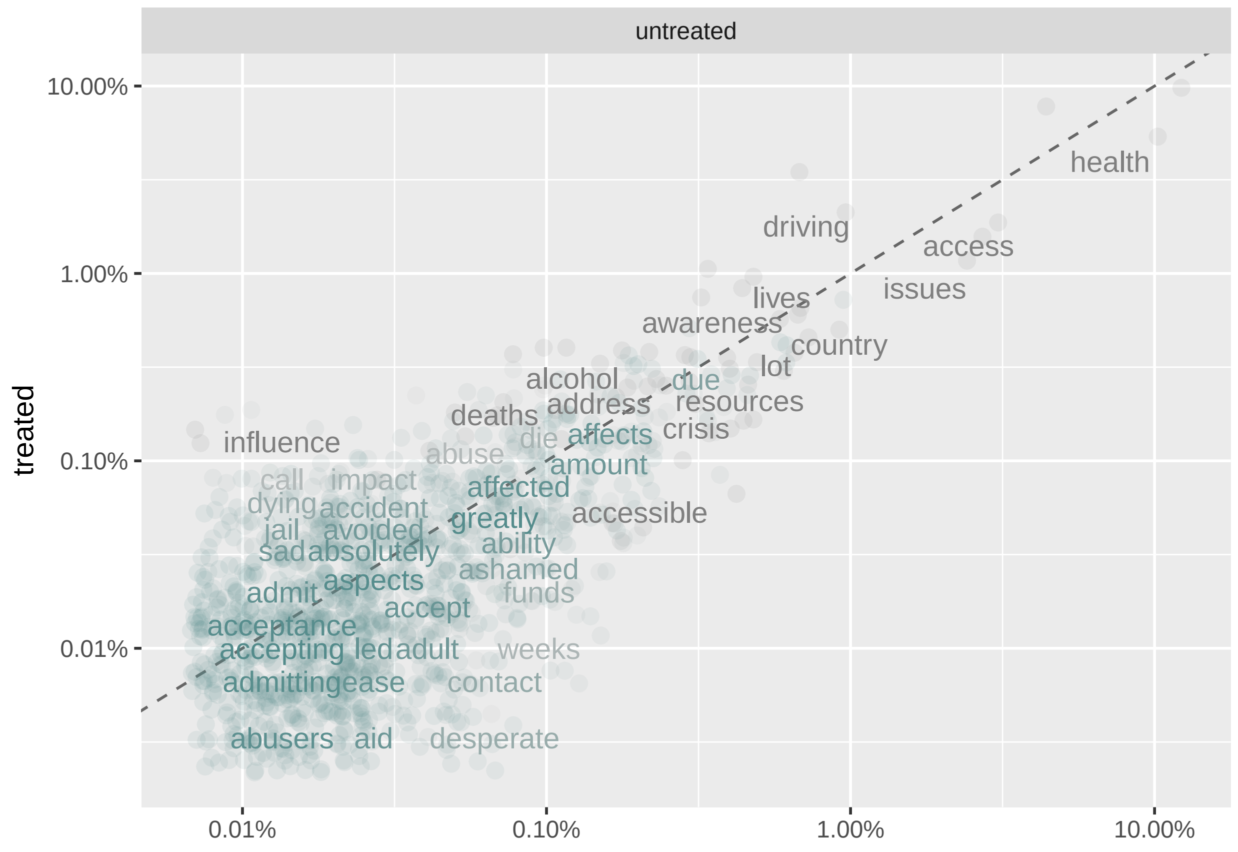
```
tidy_untreated %>%
  count(word, sort = TRUE) %>%
  filter(n > 50) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```



```r
library(scales)

# expect a warning about rows with missing values being removed
ggplot(frequency, aes(x = proportion, y = `treated`,
                      color = abs(`treated` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  scale_color_gradient(limits = c(0, 0.001),
                       low = "darkslategray4", high = "gray75") +
  facet_wrap(~treatment, ncol = 2) +
  theme(legend.position="none") +
  labs(y = "treated", x = NULL)
```

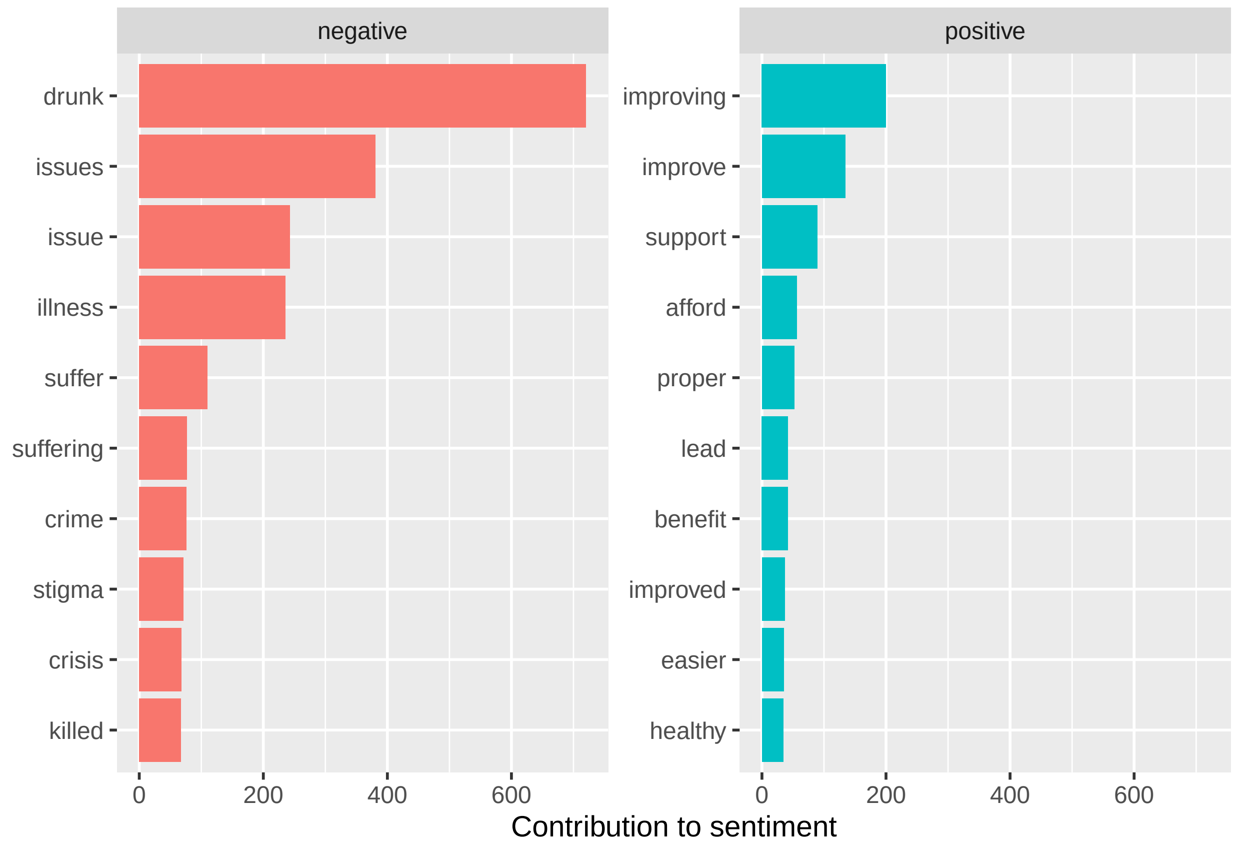
```
## Warning: Removed 2729 rows containing missing values (‘geom_point()‘).
```

```
## Warning: Removed 2730 rows containing missing values (‘geom_text()‘).
```

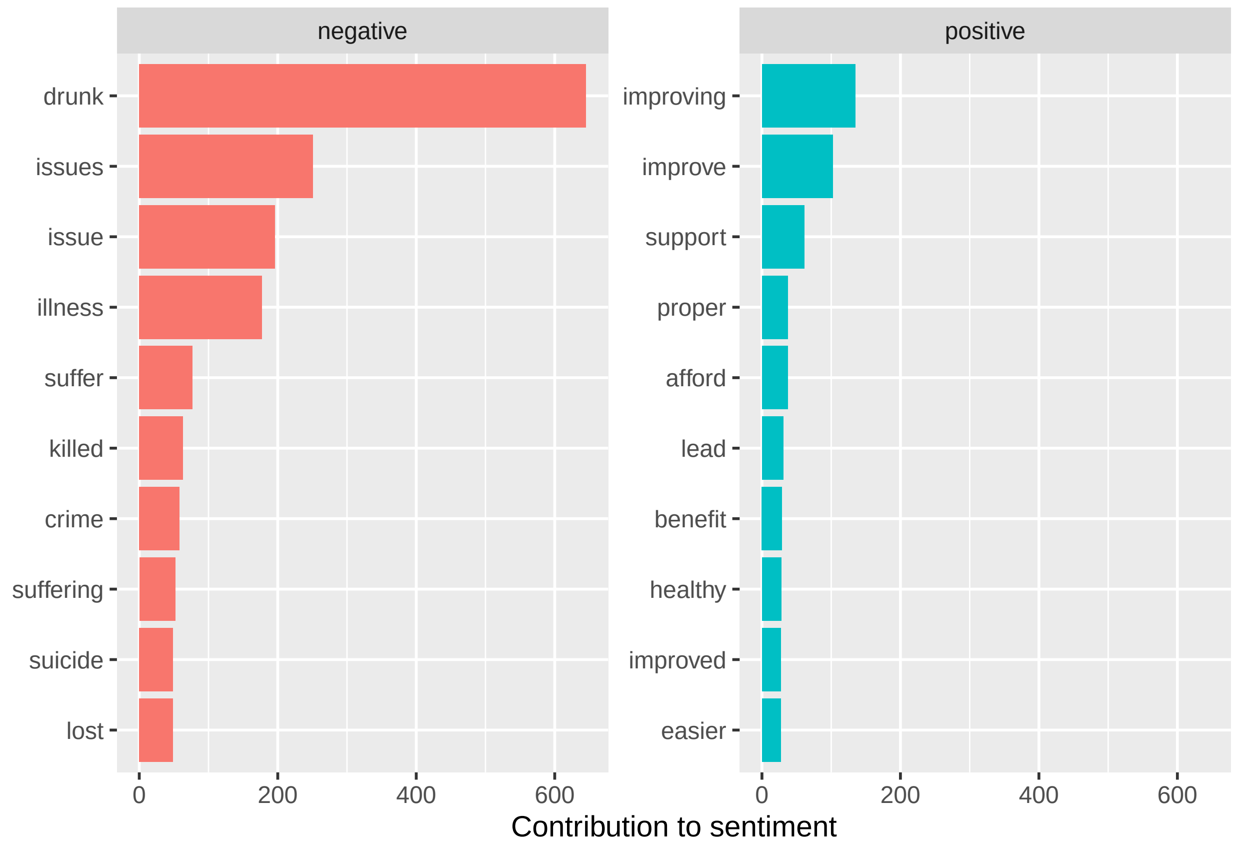


Words above the line are associated more with treated. Below associated with untreated.

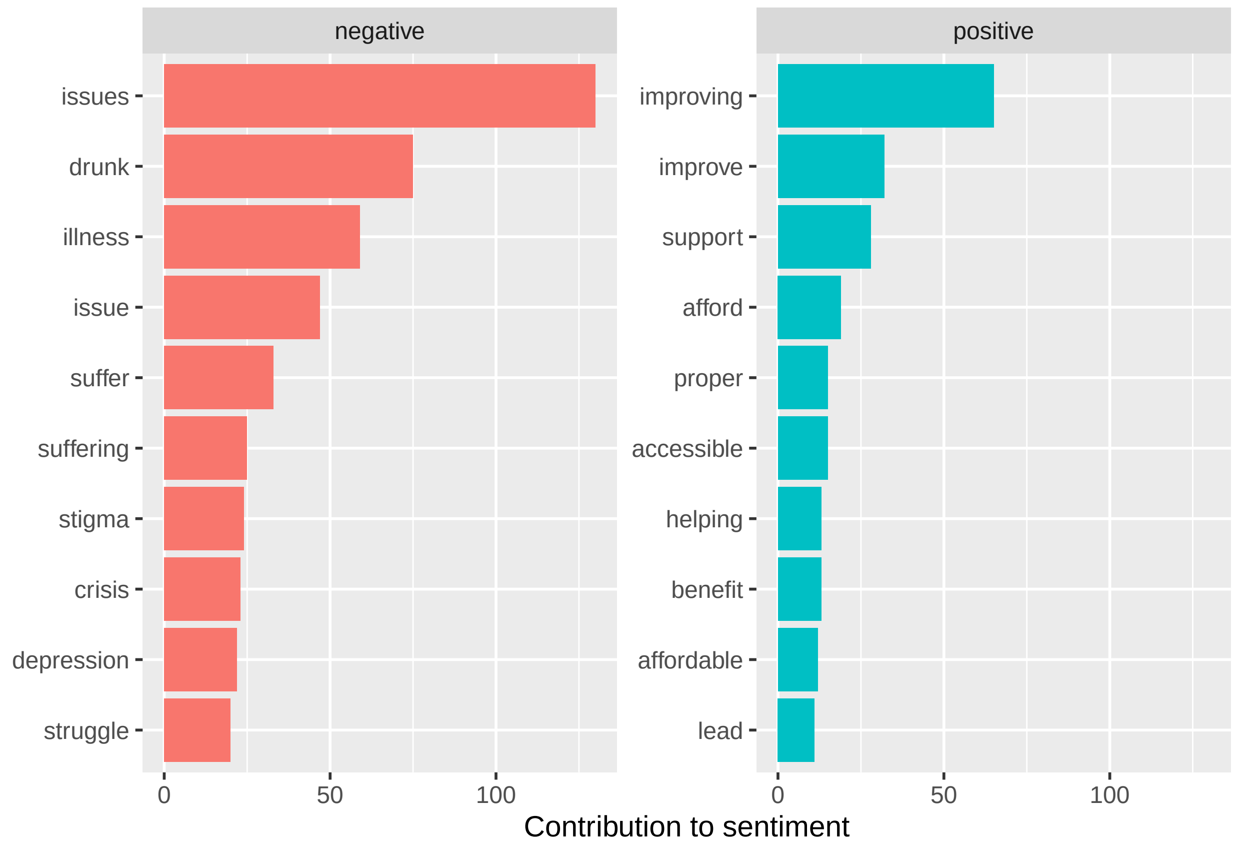
```r
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```



```
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```

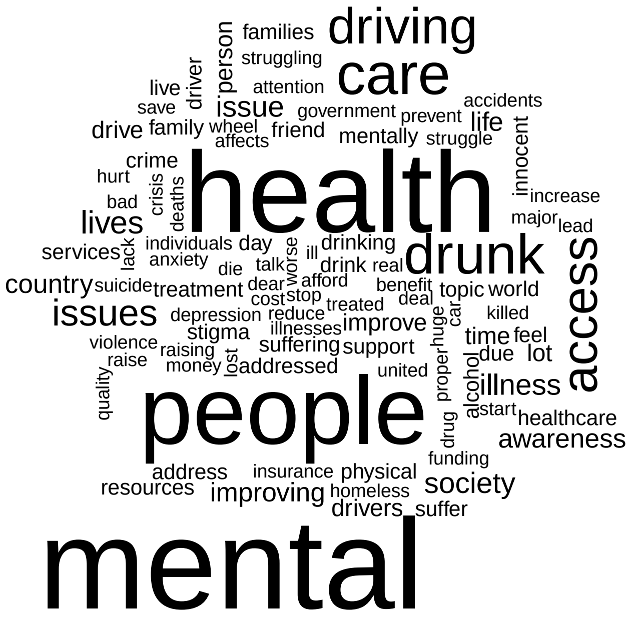


```
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```



```
library(wordcloud)

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

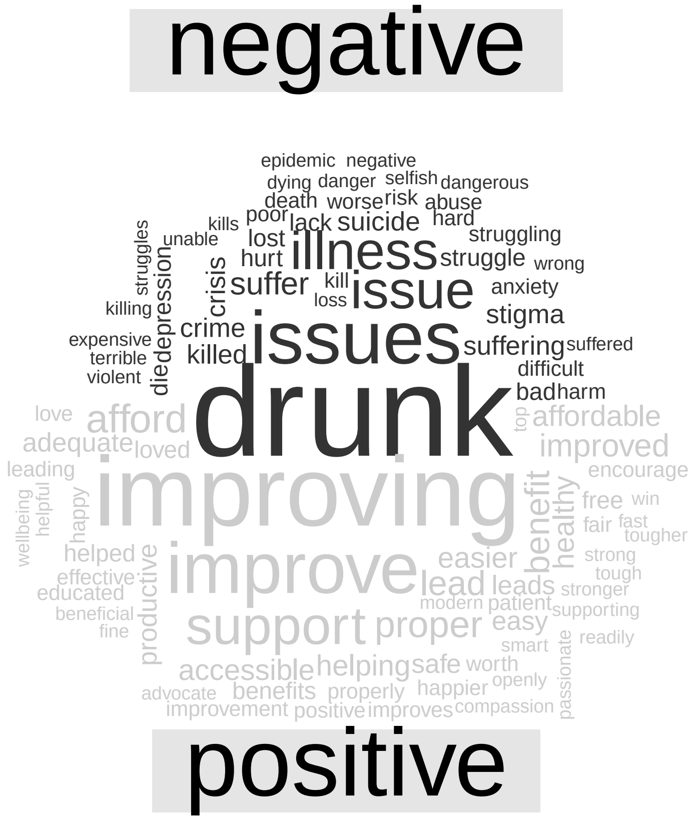
```
## Joining with ‘by = join_by(word)‘
```



```
library(reshape2)

tidy %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

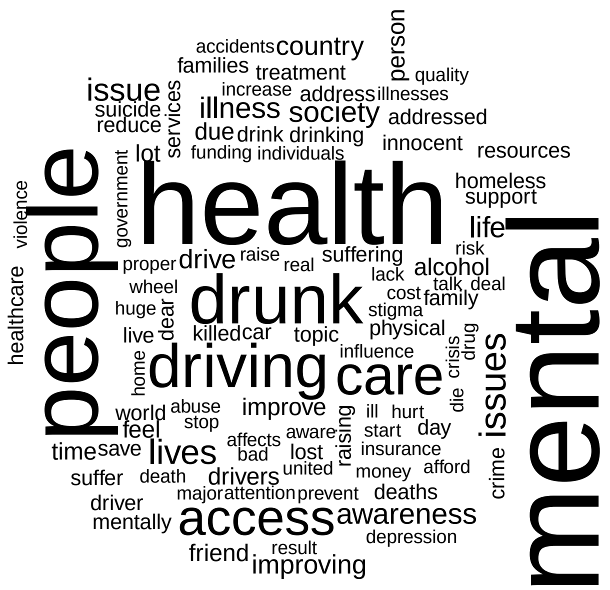
```
## Joining with 'by = join_by(word)'
```

```
## Warning in inner_join(., get_sentiments("bing")): Detected an unexpected many-to-many relationship be
## i Row 12364 of 'x' matches multiple rows in 'y'.
## i Row 3621 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.
```



```r
library(wordcloud)

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

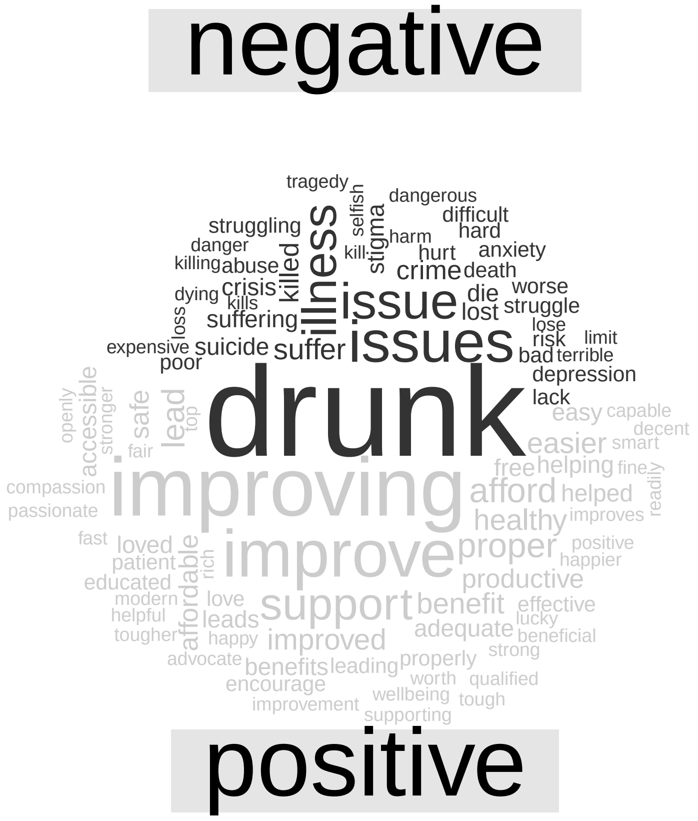
```
## Joining with ‘by = join_by(word)‘
```



```
library(reshape2)

tidy_treated %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

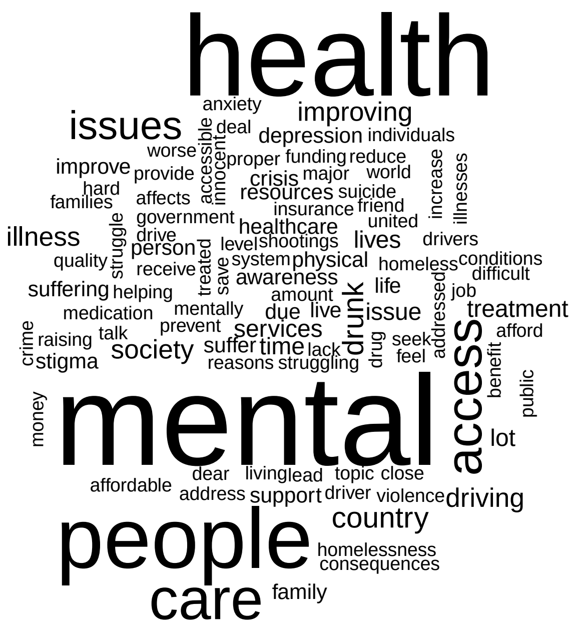
```
## Joining with ‘by = join_by(word)‘
```

```
## Warning in inner_join(., get_sentiments("bing")): Detected an unexpected many-to-many relationship b
## i Row 9677 of ‘x‘ matches multiple rows in ‘y‘.
## i Row 3621 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.
```



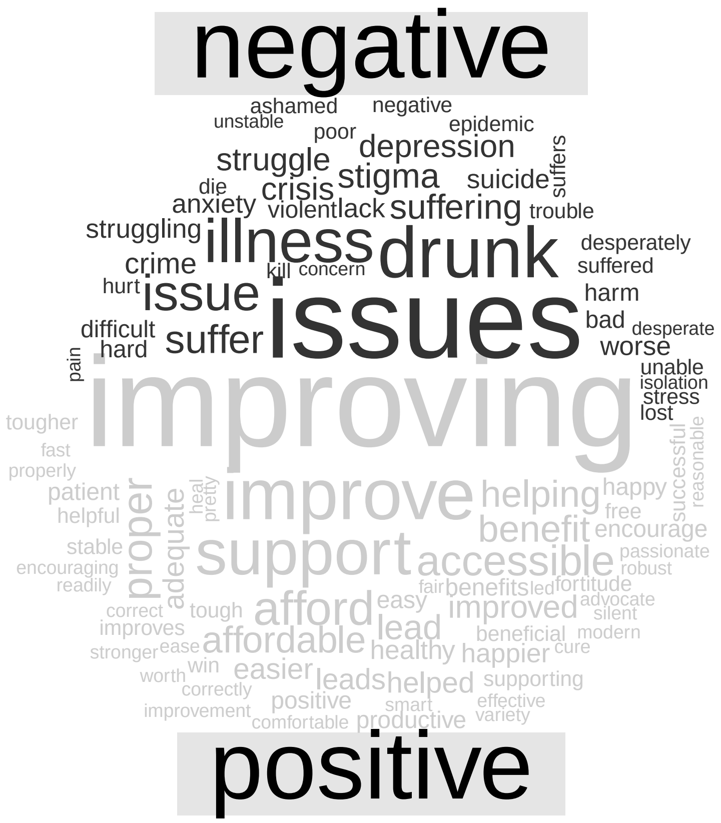
```
library(wordcloud)

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

```
## Joining with ‘by = join_by(word)‘
```



```
library(reshape2)

tidy_untreated %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

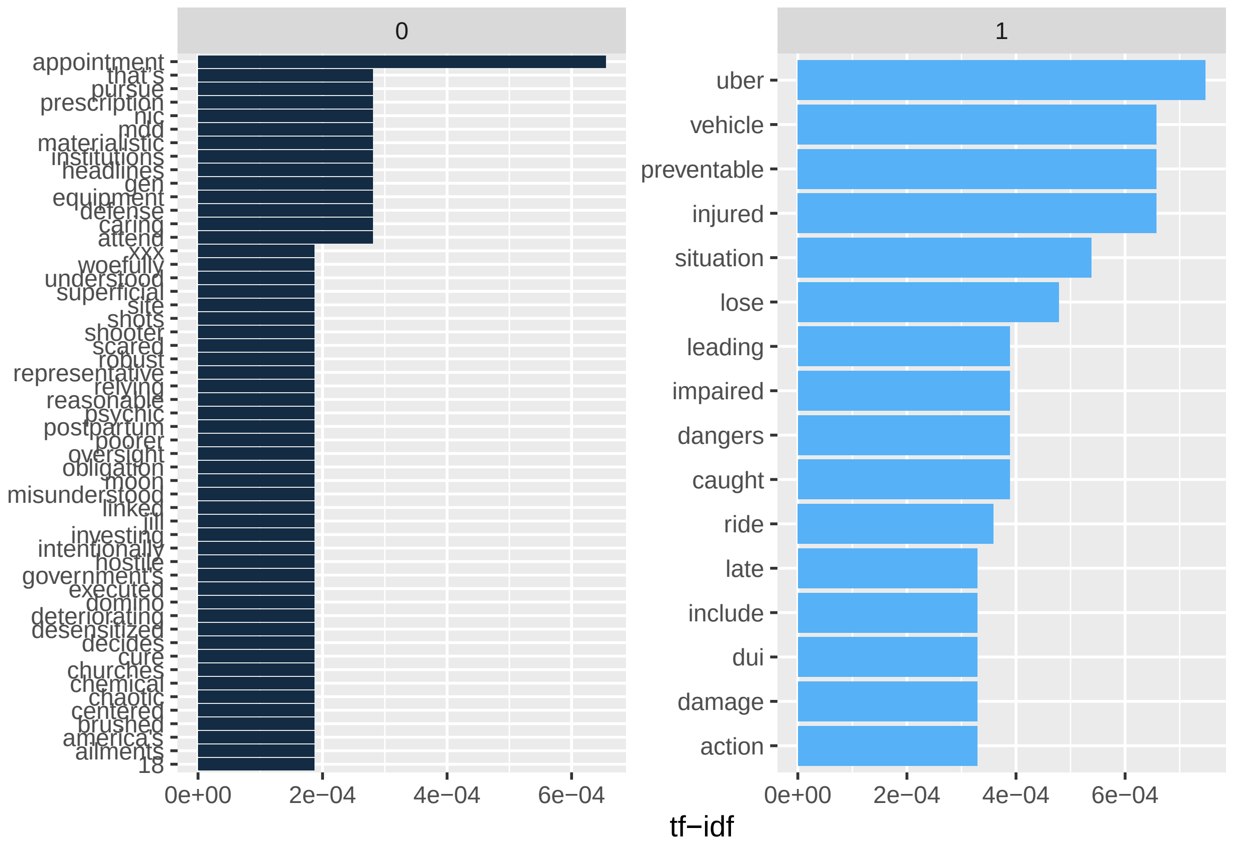
```
## Joining with `by = join_by(word)`
```



tf_idf, term frequency and inverse document frequency tells us the words that are important for a document but are not important for the corpus as a whole. Filters out common words that many documents use.

```r
tidy_words <- tidy %>%
  count(treated, word, sort = TRUE)
```

```r
library(forcats)

tidy_tf_idf %>%
  group_by(treated) %>%
  slice_max(tf_idf, n = 15) %>%
  ungroup() %>%
  ggplot(aes(tf_idf, fct_reorder(word, tf_idf), fill = treated)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~treated, ncol = 2, scales = "free") +
  labs(x = "tf-idf", y = NULL)
```

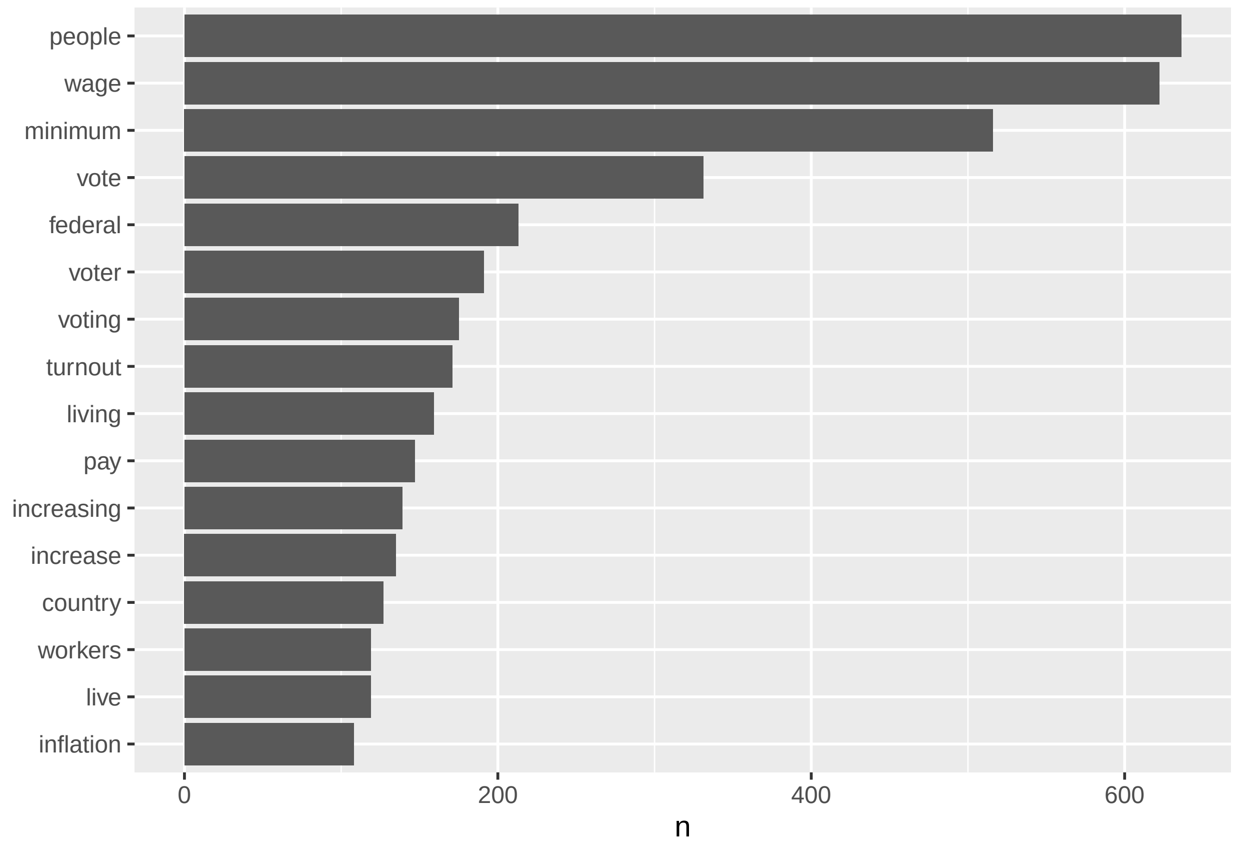


Politics Only

```
data(stop_words)

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

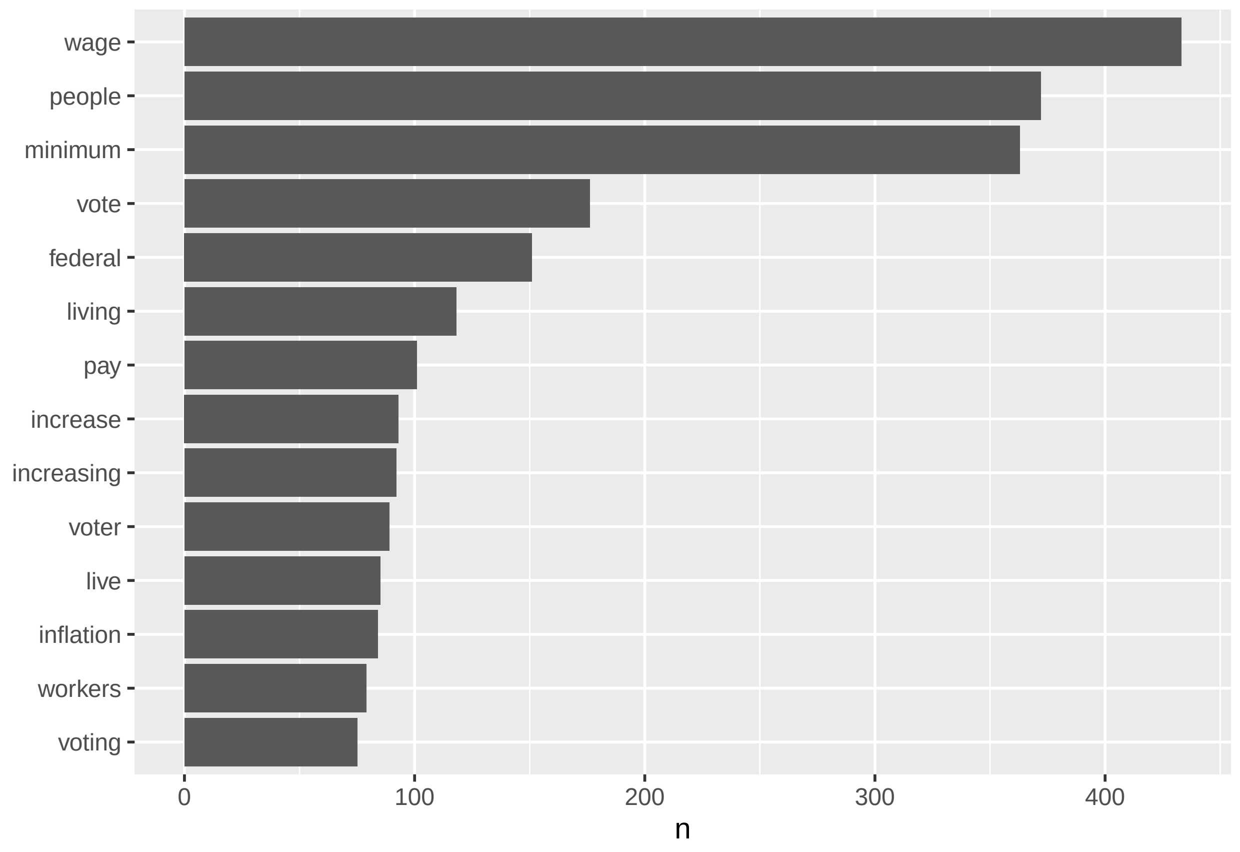
```
## Joining with ‘by = join_by(word)‘
```

```
library(ggplot2)

tidy %>%
  count(word, sort = TRUE) %>%
  filter(n > 100) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```

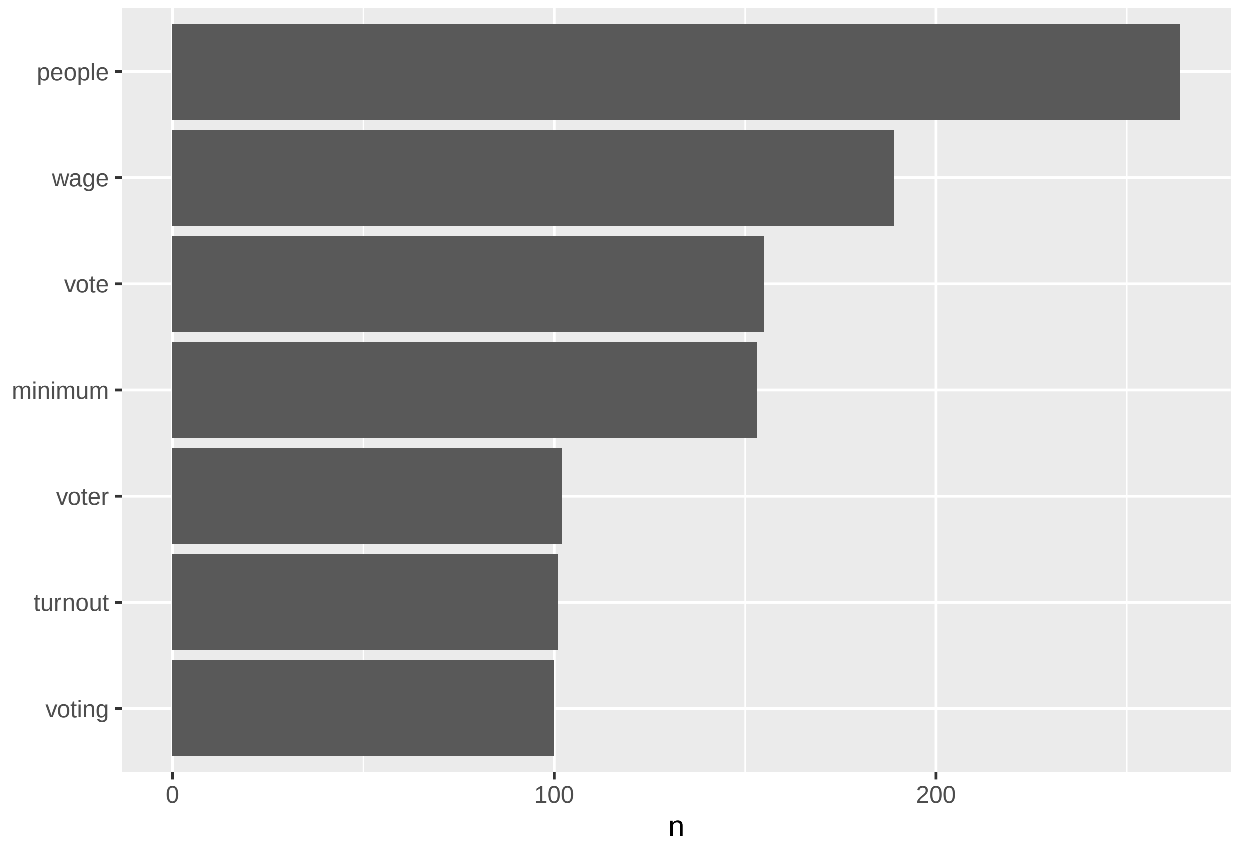


```r
tidy_treated <- subset(tidy, treated == 1)
tidy_untreated <- subset(tidy, treated == 0)
```

```r
library(ggplot2)

tidy_treated %>%
  count(word, sort = TRUE) %>%
  filter(n > 70) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```

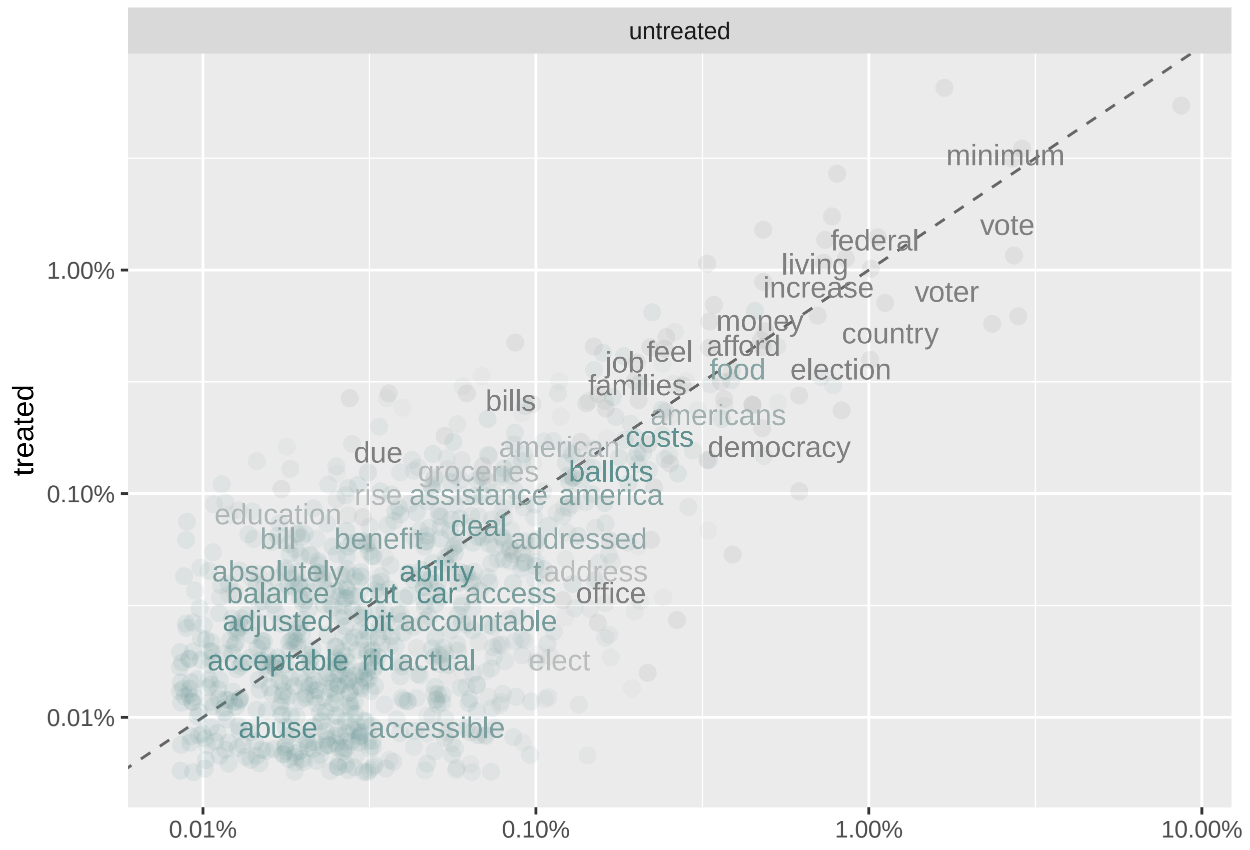


```
tidy_untreated %>%
  count(word, sort = TRUE) %>%
  filter(n > 70) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```



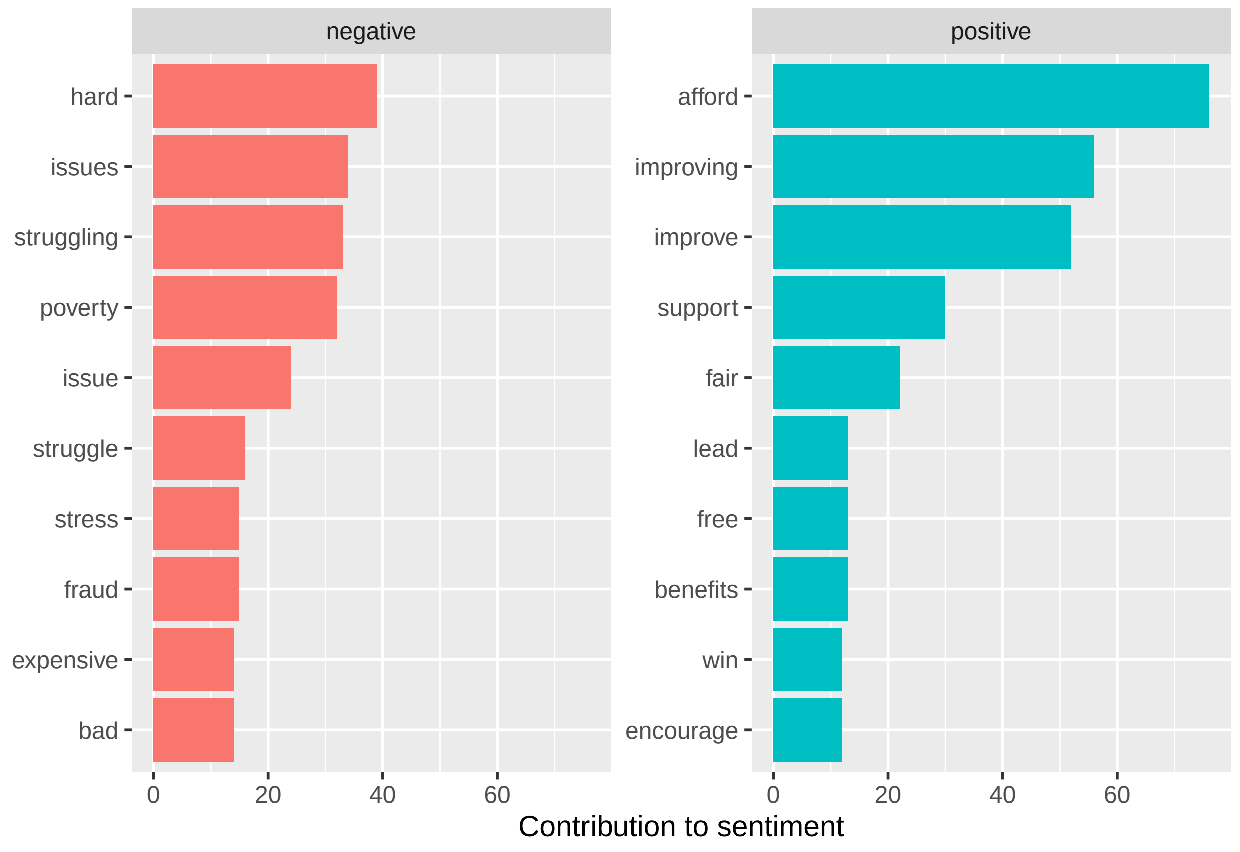
```
library(scales)

# expect a warning about rows with missing values being removed
ggplot(frequency, aes(x = proportion, y = `treated`,
                      color = abs(`treated` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  scale_color_gradient(limits = c(0, 0.001),
                       low = "darkslategray4", high = "gray75") +
  facet_wrap(~treatment, ncol = 2) +
  theme(legend.position="none") +
  labs(y = "treated", x = NULL)
```

```
## Warning: Removed 1852 rows containing missing values (‘geom_point()‘).

## Warning: Removed 1853 rows containing missing values (‘geom_text()‘).
```

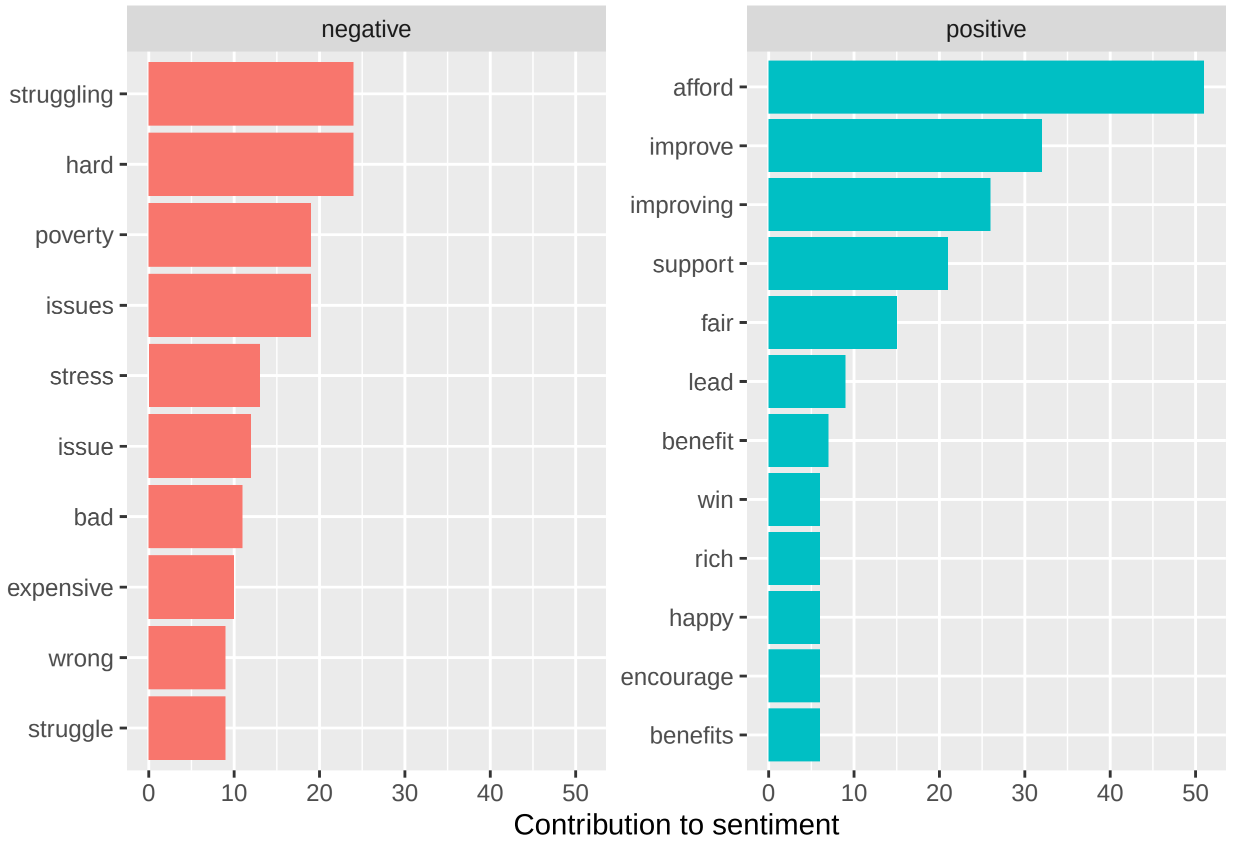


Words above the line are associated more with treated. Below associated with untreated.

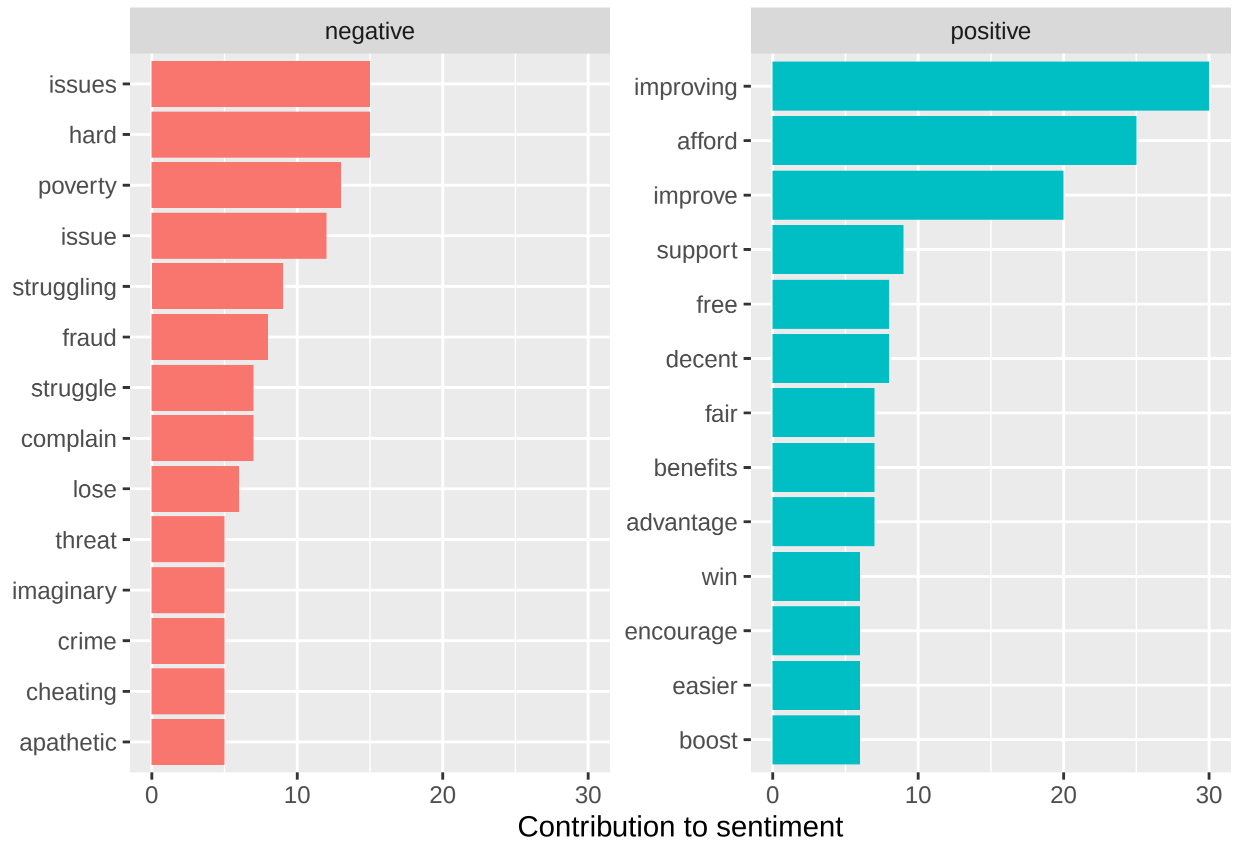
```
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```



```
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```



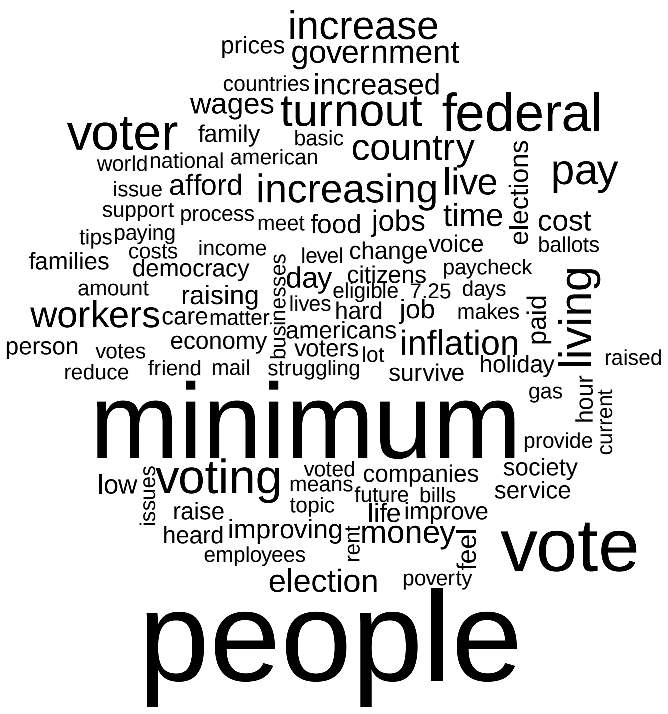
```r
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```



```
library(wordcloud)

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

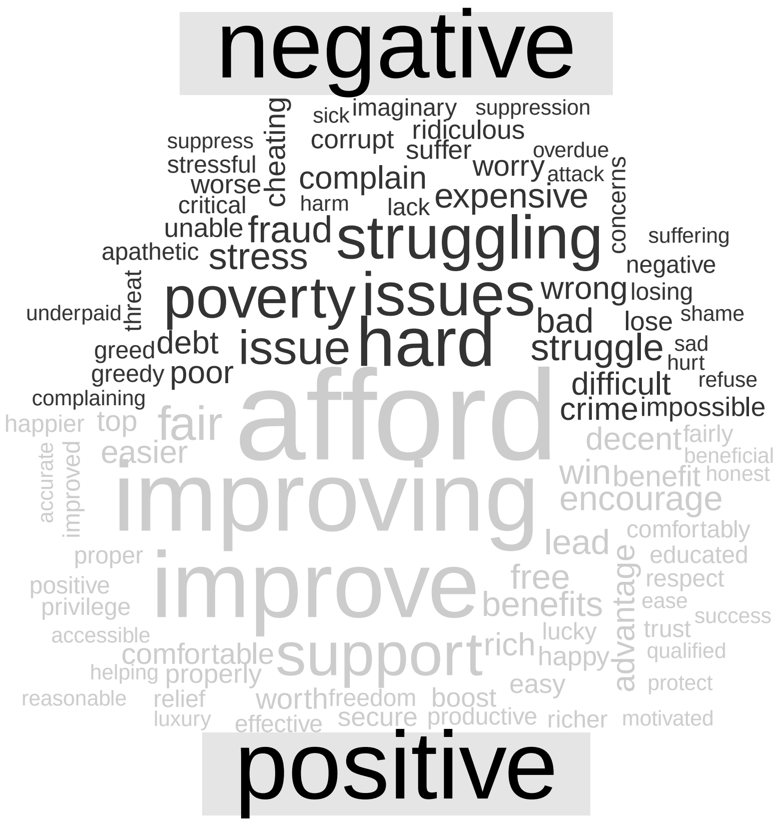
```
## Joining with 'by = join_by(word)'
```

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



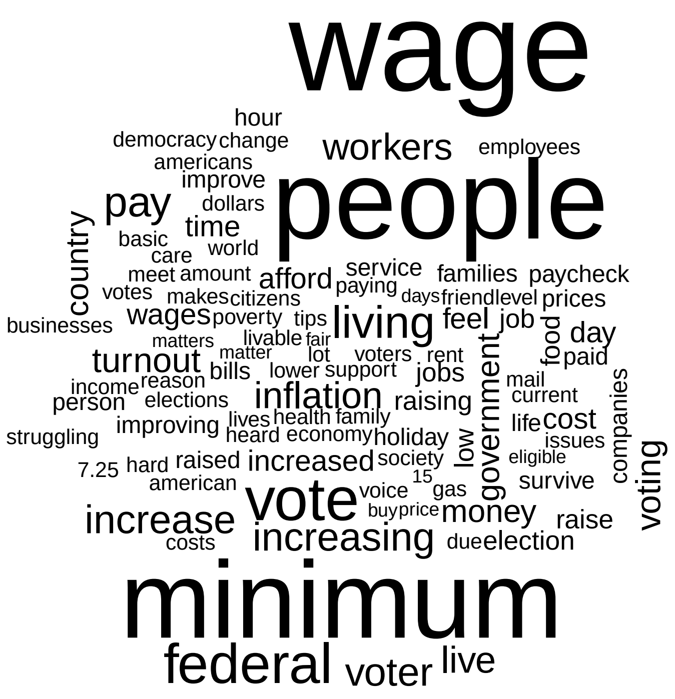
```r
library(reshape2)

tidy %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

```
## Joining with `by = join_by(word)`
```



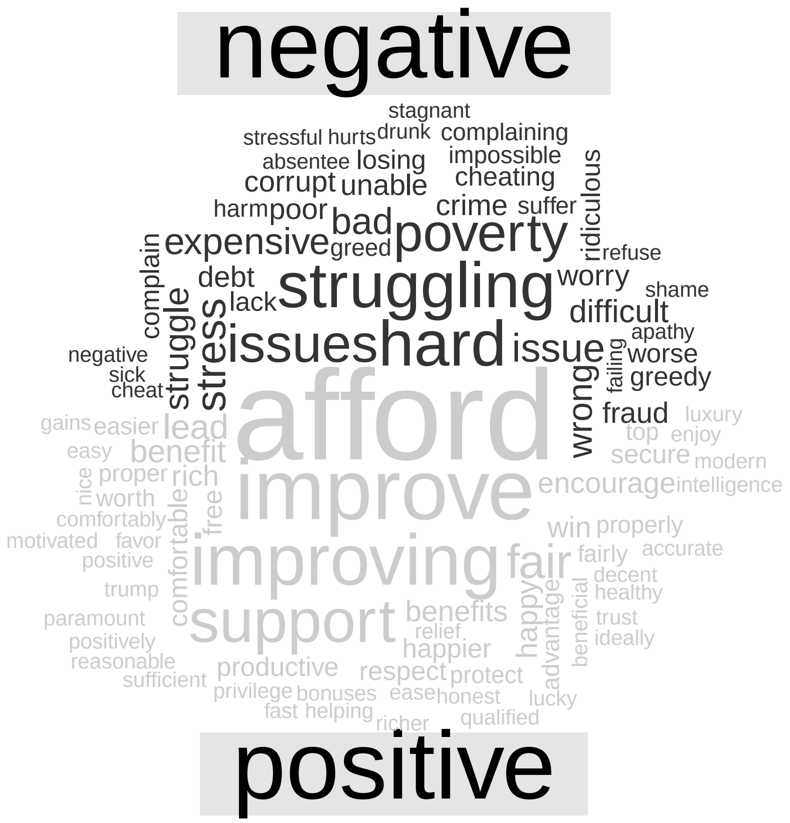
```r
library(wordcloud)

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

```
## Joining with `by = join_by(word)`
```



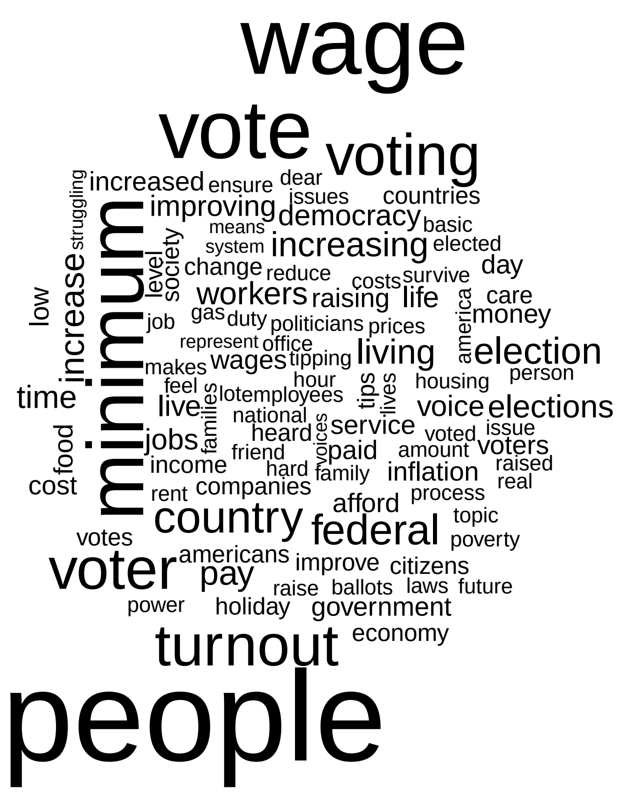
```
library(reshape2)

tidy_treated %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

```
## Joining with ‘by = join_by(word)‘
```



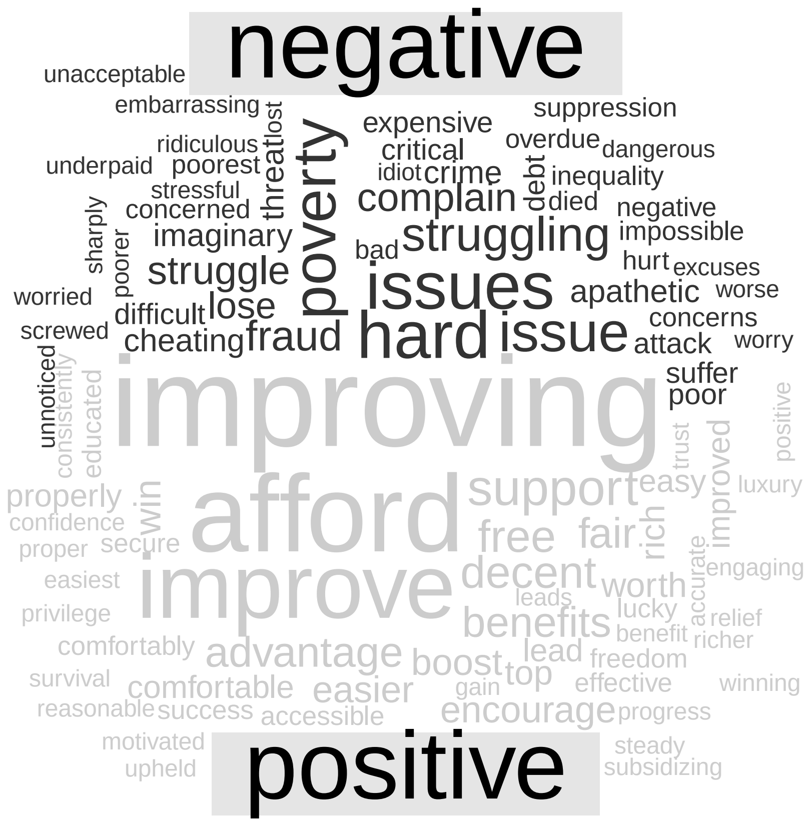
```r
library(wordcloud)

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

```
## Joining with ‘by = join_by(word)‘
```



```r
library(reshape2)

tidy_untreated %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

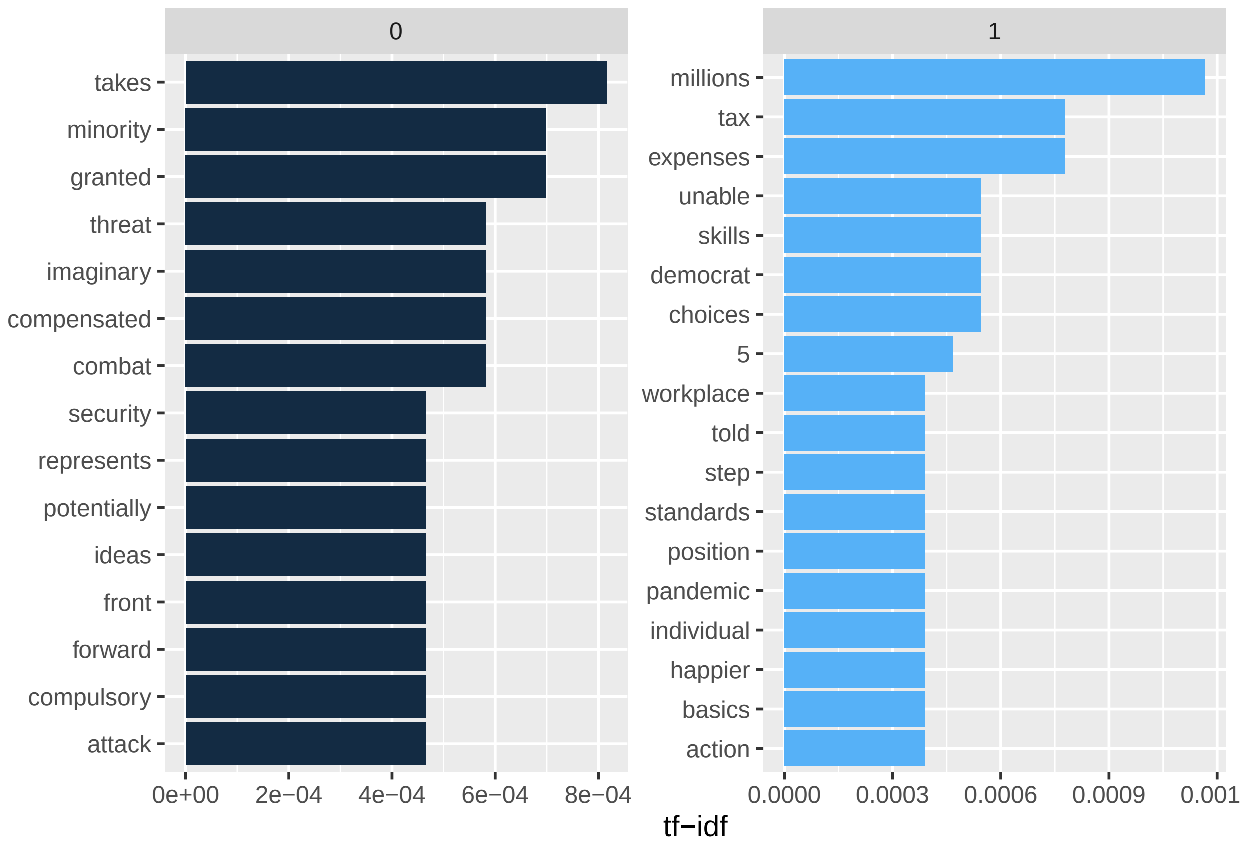
```
## Joining with ‘by = join_by(word)‘
```



tf_idf, term frequency and inverse document frequency tells us the words that are important for a document but are not important for the corpus as a whole. Filters out common words that many documents use.

```r
tidy_words <- tidy %>%
  count(treated, word, sort = TRUE)
```

```r
library(forcats)

tidy_tf_idf %>%
  group_by(treated) %>%
  slice_max(tf_idf, n = 15) %>%
  ungroup() %>%
  ggplot(aes(tf_idf, fct_reorder(word, tf_idf), fill = treated)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~treated, ncol = 2, scales = "free") +
  labs(x = "tf-idf", y = NULL)
```


0
1
takes
minority
granted
threat
imaginary
compensated
combat
security
represents
potentially
ideas
front
forward
compulsory
attack
millions
tax
expenses
unable
skills
democrat
choices
5
workplace
told
step
standards
position
pandemic
individual
happier
basics
action
0e+00
2e−04
4e−04
6e−04
8e−04
0.0000
0.0003
0.0006
0.0009
0.001
tf−idf