Análisis de sentimiento

HR Analytics: Teoría y Práctica

http://pablohaya.com/contact

01/2023

Introducción

Vamos a ver un ejemplo más avanzado aunque manteniendo la simplicidad en cuanto a nuestra aproximación analítica.

Una manera de analizar el sentimiento que expresa un texto es asignar a cada palabra una polaridad, esto si, si es positiva, negativa o neutra. El sentimiento de una oración sería la suma de las aportaciones individuales de cada palabra.

El paquete `tidyverse` viene con tres lexicones en inglés de propósito general incorporados que asignan a cada palabra una polaridad.

- `AFINN`: las puntuaciones varían entre -5 y 5.
- `bing`: cada palabra puede ser positiva o negativa. Incluye también emociones
- `nrc`: cada palabra puede ser positiva o negativa.

La función `get_sentiments()` permite obtener cada uno de estos lexicones. Probar a visualizar el formato de los otros dos lexicones.

```
get_sentiments("afinn")
```

```
## # A tibble: 2,477 x 2
##    word       value
##    <chr>      <dbl>
##  1 abandon       -2
##  2 abandoned     -2
##  3 abandons      -2
##  4 abducted      -2
##  5 abduction     -2
##  6 abductions    -2
##  7 abhor         -3
##  8 abhorred      -3
##  9 abhorrent     -3
## 10 abhors        -3
## # ... with 2,467 more rows
```

Leemos de nuevo el *dataset*, y tokenizamos la columna summary

```
reviews <- read_csv(here("data/employee_reviews_10000.csv"))

tidy_reviews <- reviews %>%
  select(summary) %>%
  unnest_tokens(word, summary)

tidy_reviews
```

```
## # A tibble: 43,693 x 1
##    word
##    <chr>
## 1 better
## 2 for
## 3 the
## 4 short
## 5 term
## 6 head
```

Obtenemos todas las palabras etiquetadas como `joy` en el lexicon `nrc`

```
nrc_joy <- get_sentiments("nrc") %>%
  filter(sentiment == "joy")

nrc_joy
```

```
## # A tibble: 687 x 2
##   word        sentiment
##   <chr>       <chr>
## 1 absolution  joy
## 2 abundance   joy
## 3 abundant    joy
## 4 accolade    joy
## 5 accompaniment joy
## 6 accomplish  joy
## # ... with 681 more rows
```

Nos quedamos unicamente con aquellas palabras que coinciden con el conjunto de palabras que representan a la emoción `joy`, y visualizamos su frecuencia absoluta:

```
tidy_reviews %>%
  inner_join(nrc_joy) %>%
  count(word, sort = TRUE)
```

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

```
## # A tibble: 141 x 2
##    word            n
##    <chr>       <int>
##  1 good         1010
##  2 fun           117
##  3 pay           108
##  4 love           93
##  5 excellent      90
##  6 fulfillment    67
##  7 grow           67
```

A practicar

Ejercicio. Repetir los análisis con las columnas `pros` y `cons`.

Distribución sentimiento por compañia

Hasta ahora hemos contemplado análisis que tienen en cuenta todos los textos de una columna (`summary`) en nuestro ejemplo. Ahora vamos a realizar análisis por cada comentario. Es preciso guardar para cada palabra el identificador del comentario al que pertenece. En nuestro ejemplo, será el número de línea.

Además, vamos a incluir también la compañía a la que hace referencia el comentario.

```
tidy_reviews <- reviews %>%
  select(company, summary) %>%
  mutate(id_review = row_number()) %>%
  unnest_tokens(word, summary)

tidy_reviews
```

```
## # A tibble: 43,693 x 3
##    company id_review word
##    <chr>       <int> <chr>
##  1 apple           1 better
##  2 apple           1 for
##  3 apple           1 the
##  4 apple           1 short
##  5 apple           1 term
##  6 google          2 head
##  7 google          2 of
##  8 google          2 industry
##  9 amazon          3 overall
## 10 amazon          3 is
## # ... with 43,683 more rows
```

Vamos a realizar análisis de sentimiento empleando el lexicon `bing`

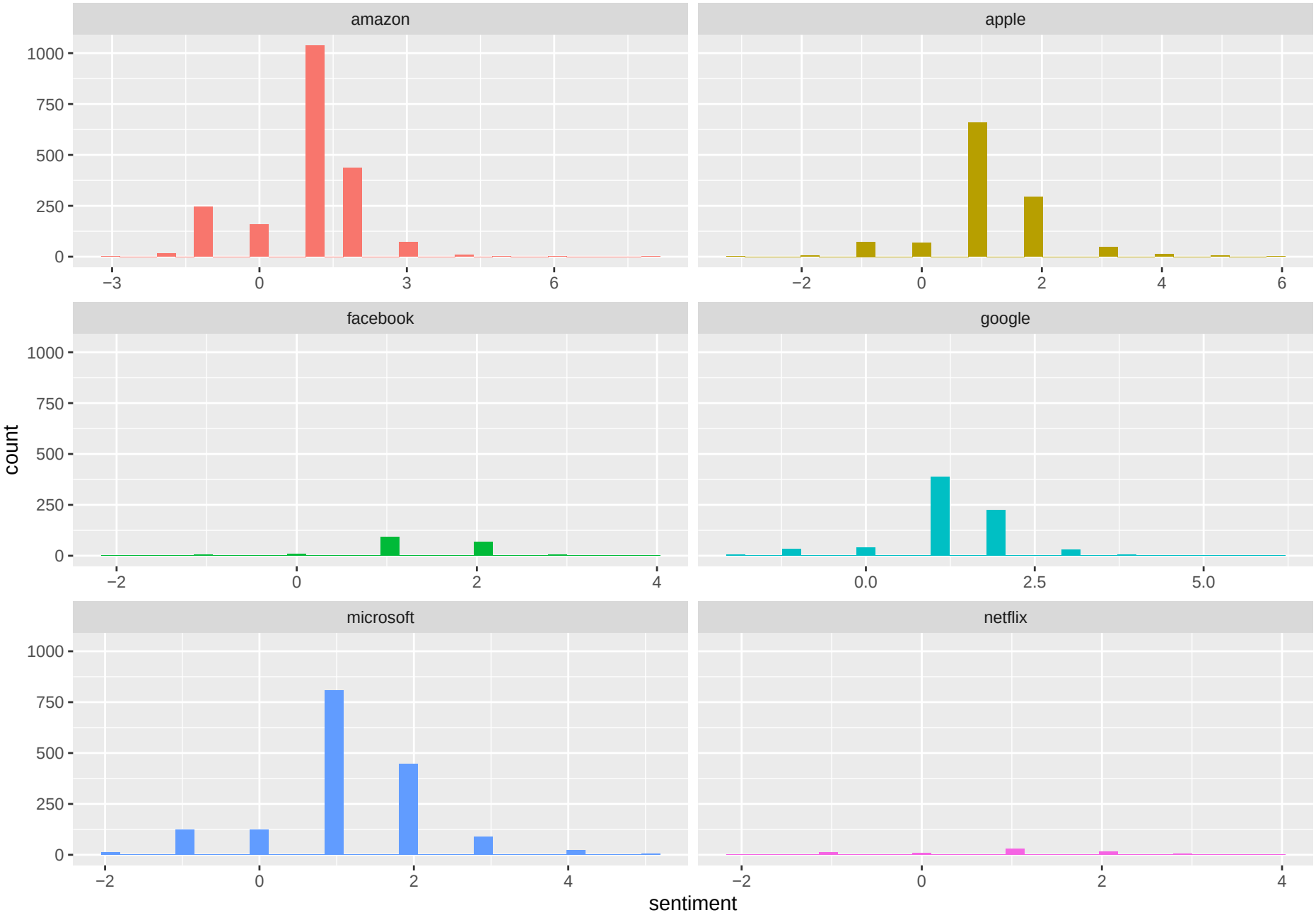
```
reviews_sentiment <- tidy_reviews %>%
  inner_join(get_sentiments("bing"))  %>%
  count(company, id_review, sentiment) %>%
  pivot_wider(names_from = sentiment,
              values_from = n,
              values_fill = 0) %>%
  mutate(sentiment = positive - negative)

reviews_sentiment
```

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

```
## # A tibble: 5,750 x 5
##    company id_review positive negative sentiment
##    <chr>       <int>    <int>    <int>     <int>
##  1 amazon          3        3        0         3
##  2 amazon          5        1        1         0
##  3 amazon          7        1        0         1
##  4 amazon          9        2        2         0
##  5 amazon         11        0        1        -1
##  6 amazon         14        2        0         2
##  7 amazon         15        2        1         1
##  8 amazon         23        2        0         2
##  9 amazon         24        0        1        -1
## 10 amazon         26        0        1        -1
## # ... with 5,740 more rows
```

Visualizamos los histogramas de frecuencia de palabras por cada compañía

```
ggplot(reviews_sentiment, aes(sentiment, fill = company)) +
  geom_histogram(show.legend = FALSE)  +
  facet_wrap(~company, ncol = 2, scales = "free_x")
```


amazon
apple
facebook
google
microsoft
netflix
count
sentiment
1000
750
500
250
0
−3
0
3
6
−2
0
2
4
6
−2
0
2
4
0.0
2.5
5.0
−2
0
2
4
−2
0
2
4

Palabras positivas y negativas más frecuentes

Analizamoslos términos positivos y negativos por separado

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

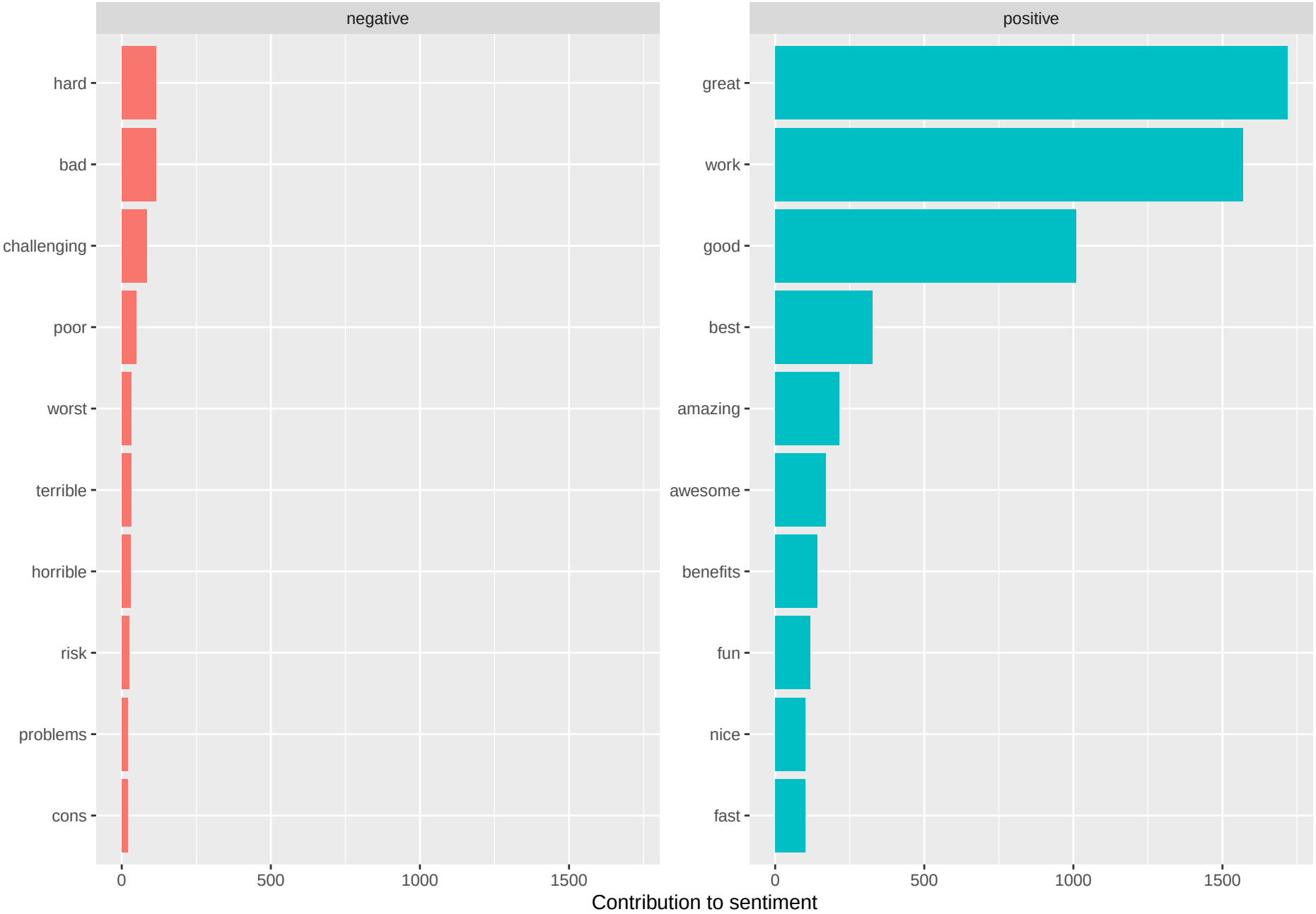
```
## Joining, by = "word"
```

```
bing_word_counts
```

```
## # A tibble: 693 x 3
##    word      sentiment      n
##    <chr>     <chr>      <int>
##  1 great     positive    1718
##  2 work      positive    1569
##  3 good      positive    1010
##  4 best      positive     326
```

La siguiente gráfica muestra como contribuyen los términos más frecuentes al sentimiento:

```
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)
```


negative
positive
hard
bad
challenging
poor
worst
terrible
horrible
risk
problems
cons
great
work
good
best
amazing
awesome
benefits
fun
nice
fast
0
500
1000
1500
Contribution to sentiment

Nubes de palabras

Es preciso instalar el paquete `wordcloud` que dependiendo de la versión de R puede ser un poco complejo. Probar primero:

- install.packages("slam")
- install.packages("wordcloud")

Si no funciona hay que instalar primero las Rtools

- Como instalar las RTools: https://cran.r-project.org/bin/windows/Rtools/rtools40.html

Recordar realizar el paso de actualizar el `PATH` y reiniciar `R`

Instalar el paquete `devtools`

- `install.packages("devtools")`

y finalmente instalar desde la consola de RStudio: `wordcloud`

- `devtools::install_github("ifellows/wordcloud")`

Una vez instalado el paquete, visualizar una nube de palabras es muy sencillo:

```
library(wordcloud)

tidy_reviews %>%
  anti_join(stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 50))
```

```
## Warning: package 'wordcloud' was built under R version 4.0.

## Warning: package 'RColorBrewer' was built under R version 4
```

Podemos incorporar palabras prohibidas que quisieramos eliminar

```
custom_stop_words <- bind_rows(tibble(
                                word = c("company"),
                                lexicon = c("propio")
                               ),
                               stop_words)
custom_stop_words
```

```
## # A tibble: 1,150 x 2
##   word    lexicon
##   <chr>   <chr>
## 1 company propio
## 2 a       SMART
## 3 a's     SMART
## 4 able    SMART
## 5 about   SMART
## 6 above   SMART
## # ... with 1,144 more rows
```

Mostramos la nueva nube de palabras habiendo eliminado company

```
tidy_reviews %>%
  anti_join(custom_stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 50))
```


engineer
warehouse
fast
management
associate
bad
senior
hard
software
learning
apple
amazon
job
benefits
sales
people
advisor
pay
excellent
learn
nice
love
customer
challenging
time
review
program
fun
support
balance
culture
google
team
career
retail
intern
smart
service
microsoft
specialist
opportunities
development
life
technical
awesome
amazing
environment
experience
business
manager

Mismo gráfico pero destacando los términos positivo y negativo

```
tidy_reviews %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("red", "green"),
                   max.words = 50)
```

negative

positive