Análisis de Sentimientos en las películas de Spielberg

Adquisición de Datos

Primero tenemos que cargar las librerías necesarias.

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

## The following objects are masked from 'package:reshape':
##
##     colsplit, melt, recast

##
## Attaching package: 'ggplot2'

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

## Loading required package: tcltk

## Warning in fun(libname, pkgname): couldn't connect to display ":0"
```

Ahora podemos utilizar la función `Corpus` del paquete `tm`, que genera automáticamente un corpus a partir de un directorio con un archivo por documento.

```
camino <- file.path('datos/peliculas')
docs   <- Corpus(DirSource(camino), readerControl = list(language = 'spa'))
```

Preprocesado

Una vez hemos creado el corpus, vamos a limpiarlo eliminando números, símbolos, espacios y stopwords. Para ello usamos la función `tm_map`.

```
corpus_clean <- tm_map(docs, removePunctuation, preserve_intra_word_dashes = TRUE)
corpus_clean <- tm_map(corpus_clean, content_transformer(tolower))
corpus_clean <- tm_map(corpus_clean, removeWords, stopwords('es'))
corpus_clean <- tm_map(corpus_clean, content_transformer(stripWhitespace))
corpus_clean <- tm_map(corpus_clean, removeNumbers)
```

Vectorizado

Una vez hemos limpiado los textos, tenemos que convertir las listas de palabras que representan cada película a vectores numéricos para poder analizarlas. Vamos a usar la función `DocumentTermMatrix` que tiene como columnas cada una de las palabras y como filas cada documento (película). De esta forma, en la fila i y columna j aparecerá el número de veces que la palabra j está en el documento i.

```
dtm <- DocumentTermMatrix(corpus_clean, control =  list(stopwords=FALSE, wordLengths=c(0, Inf)))
```

Análisis de Sentimiento

Ahora vamos a generar dos nuevas `DocumentTermMatrix`, una para las palabras positivas y otra para las negativas.

```
camino        <- file.path('datos/diccionarios/positive_words_es.txt')
listapositiva <- read.csv(camino,header=F,quote="",fileEncoding="Latin1")
listapositiva <- as.matrix(listapositiva)
listapositiva <- as.vector(listapositiva)

ppositivas.tdm<- DocumentTermMatrix(corpus_clean, control = list(tokenize    = 'word',
                                                                 dictionary  = listapositiva,
                                                                 stopwords   = FALSE,
                                                                 wordLengths = c(0, Inf),
                                                                 stemming    = 'spanish'))
ppositivas.tdm
```

```
## <<DocumentTermMatrix (documents: 11, terms: 1555)>>
## Non-/sparse entries: 2817/14288
## Sparsity           : 84%
## Maximal term length: 19
## Weighting          : term frequency (tf)
```

```
camino        <- file.path('datos/diccionarios/negative_words_es.txt')
listanegativa <- read.csv(camino,header=F,quote="",fileEncoding="Latin1")
listanegativa <- as.matrix(listanegativa)
listanegativa <- as.vector(listanegativa)

pnegativas.tdm<- DocumentTermMatrix(corpus_clean, control = list(tokenize    = 'word',
                                                                 dictionary  = listanegativa,
                                                                 stopwords   = FALSE,
                                                                 wordLengths = c(0, Inf),
                                                                 stemming    = 'spanish'))
pnegativas.tdm
```

```
## <<DocumentTermMatrix (documents: 11, terms: 2720)>>
## Non-/sparse entries: 2611/27309
## Sparsity           : 91%
## Maximal term length: 19
## Weighting          : term frequency (tf)
```

Podemos ahora resumir toda esta información en una única tabla con el conteo de palabras positivas, negativas y neutras de cada una de las películas.

```
positivas <- apply(ppositivas.tdm,1,sum)
negativas <- apply(pnegativas.tdm,1,sum)
num_total <- apply(dtm,1,sum)
```

```
total.df  <- data.frame(positive = positivas,
                        negative = negativas,
                        neutral  = num_total - positivas - negativas)
total.df
```

```
##                                     positive negative neutral
## Always.txt                               554      498    6195
## Atrapame si puedes.txt                   729      371    5981
## el mundo perdido Jurassic Park.txt       646      575    7281
## En busca del arca perdida.txt           1104      687   10881
## E.T., el extraterrestre.txt              914      604    9557
## Indiana Jones y la Ultima Cruzada.txt    822      463    7311
## La lista de Schindler.txt                798      661    8457
## Munich.txt                               719      707    8201
## Parque Jurasico.txt                      811      498    8124
## Salvar al soldado Ryan.txt               769      645    8374
## Tiburon.txt                              829      690    8578
```

Para poder comparar las películas entre sí, vamos a normalizar los valores, dividiendo entre el total de palabras del documento. Además vamos a generar un score, definido como la diferencia entre el porcentaje de palabras positivas y negativas (así, películas con scores positivos tienen más palabras positivas que negativas y viceversa).

```
scores <- as.data.frame(apply(total.df, 2, function(x)(100 * x / num_total)))
scores$score <- scores$positive - scores$negative
scores$movie <- substr(rownames(scores),1,nchar(rownames(scores))-4)
scores
```

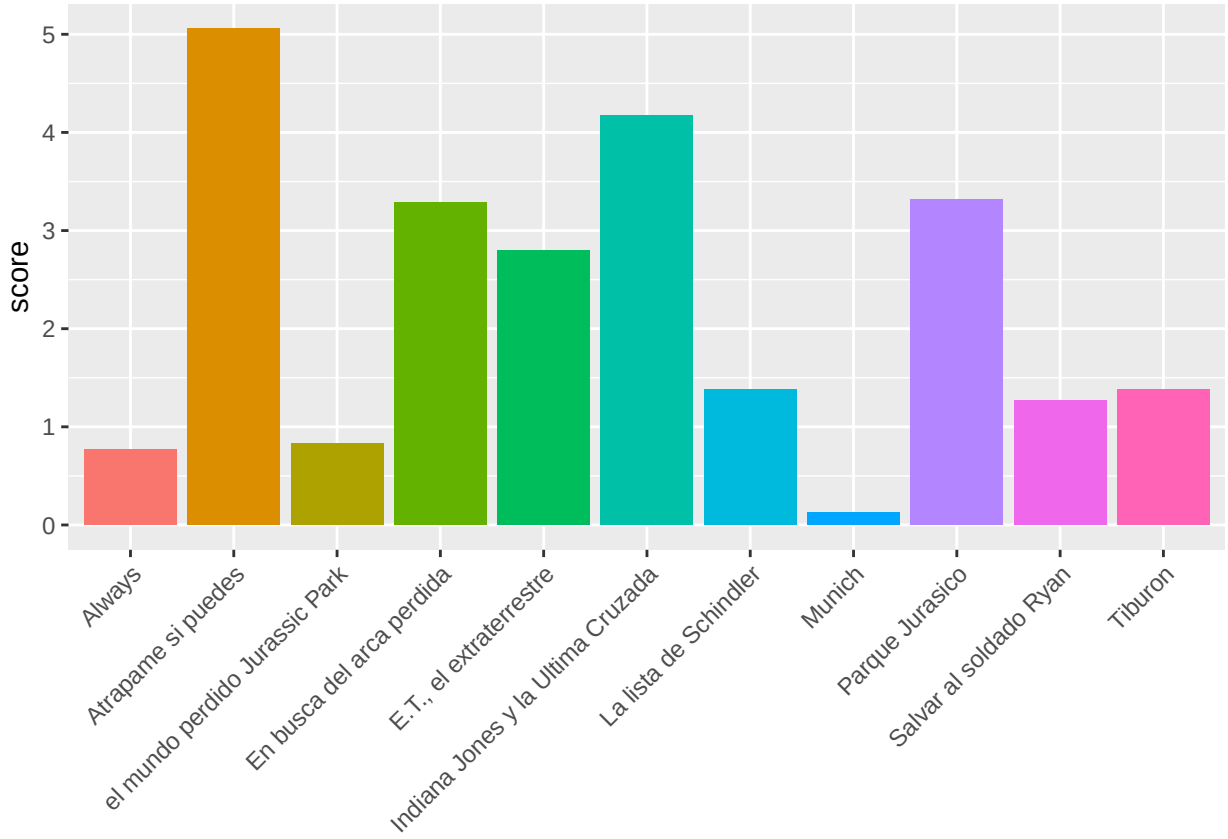
```
##                                      positive negative  neutral
## Always.txt                           7.644543 6.871809 85.48365
## Atrapame si puedes.txt              10.295156 5.239373 84.46547
## el mundo perdido Jurassic Park.txt   7.598212 6.763115 85.63867
## En busca del arca perdida.txt        8.712121 5.421402 85.86648
## E.T., el extraterrestre.txt          8.252822 5.453725 86.29345
## Indiana Jones y la Ultima Cruzada.txt 9.562587 5.386226 85.05119
## La lista de Schindler.txt            8.047600 6.665994 85.28641
## Munich.txt                           7.468578 7.343929 85.18749
## Parque Jurasico.txt                  8.597477 5.279338 86.12318
## Salvar al soldado Ryan.txt           7.856559 6.589702 85.55374
## Tiburon.txt                          8.210360 6.833713 84.95593
##                                         score
## Always.txt                          0.7727335
## Atrapame si puedes.txt              5.0557831
## el mundo perdido Jurassic Park.txt  0.8350976
## En busca del arca perdida.txt       3.2907197
## E.T., el extraterrestre.txt         2.7990971
## Indiana Jones y la Ultima Cruzada.txt 4.1763611
## La lista de Schindler.txt           1.3816055
## Munich.txt                          0.1246494
## Parque Jurasico.txt                 3.3181385
## Salvar al soldado Ryan.txt          1.2668574
## Tiburon.txt                         1.3766465
##                                                                  movie
```

```
## Always.txt                                                    Always
## Atrapame si puedes.txt                                Atrapame si puedes
## el mundo perdido Jurassic Park.txt       el mundo perdido Jurassic Park
## En busca del arca perdida.txt                 En busca del arca perdida
## E.T., el extraterrestre.txt                     E.T., el extraterrestre
## Indiana Jones y la Ultima Cruzada.txt Indiana Jones y la Ultima Cruzada
## La lista de Schindler.txt                      La lista de Schindler
## Munich.txt                                                    Munich
## Parque Jurasico.txt                                  Parque Jurasico
## Salvar al soldado Ryan.txt                  Salvar al soldado Ryan
## Tiburon.txt                                                  Tiburon
```

Análisis

Una vez hemos limpiado los datos y generado una puntuación de sentimiento por película, vamos a analizar los datos resultantes. En primer lugar vamos a ver las puntuaciones que hemos obtenido por película.

```
ggplot(scores, aes(x = movie, y = score, fill = movie)) +
  geom_col() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        legend.position = "none",
        axis.title.x = element_blank())
```

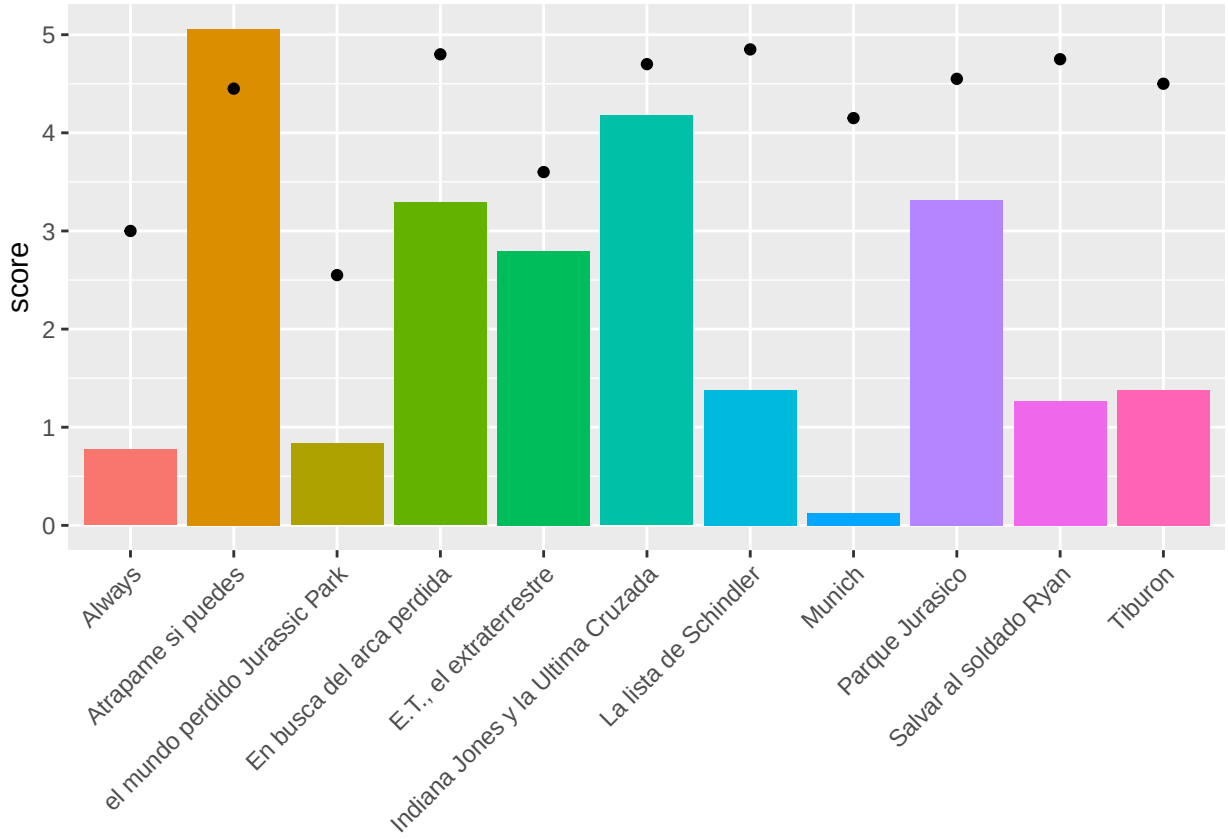


Para tener algo de contexto, al gráfico anterior le vamos a añadir las puntuaciones de las películas en la página web Rotten Tomatoes.

```
rotten_tomatoes <- data.frame('movie' = c('Always', 'Atrapame si puedes', 'el mundo perdido Jurassic Pa
                                          'E.T., el extraterrestre', 'Indiana Jones y la Ultima Cruzada
                                          'Munich', 'Parque Jurasico', 'Salvar al soldado Ryan', 'Tibur
                              'rt_score' = c(6.0, 8.9, 5.1, 9.6, 7.2, 9.4, 9.7, 8.3, 9.1, 9.5, 9.0))

scores <- merge(scores, rotten_tomatoes, 'movie')
scores$rt_score <- scores$rt_score / 2
```

```
ggplot(scores, aes(x = movie, y = score, fill = movie)) +
  geom_col() +
  geom_point(aes(y = rt_score)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        legend.position = "none",
        axis.title.x = element_blank())
```



Vemos que hay cierta relación entre las puntuaciones de Rotten Tomatoes y nuestra puntuación de sentimiento, pero en algunos casos como La lista de Schindler, Munich o E.T. las diferencias son grandes. Esto puede deberse a que las palabras que se emplean en las críticas sean tradicionalmente negativas o positivas (como guerra o conflicto en las primeras), aunque no lo sean en el contexto. Vamos a ver el coeficiente de correlación entre ambas puntuaciones.

```
print(paste0('Correlación entre scores: ', cor(scores$score, scores$rt_score)))
```

```
## [1] "Correlación entre scores: 0.415416172907056"
```

```
print(paste0('Correlación entre positivas y RT score: ', cor(scores$positive, scores$rt_score)))
```

```
## [1] "Correlación entre positivas y RT score: 0.45130044015036"
```

```
print(paste0('Correlación entre negativas y RT score: ', cor(scores$negative, scores$rt_score)))
```

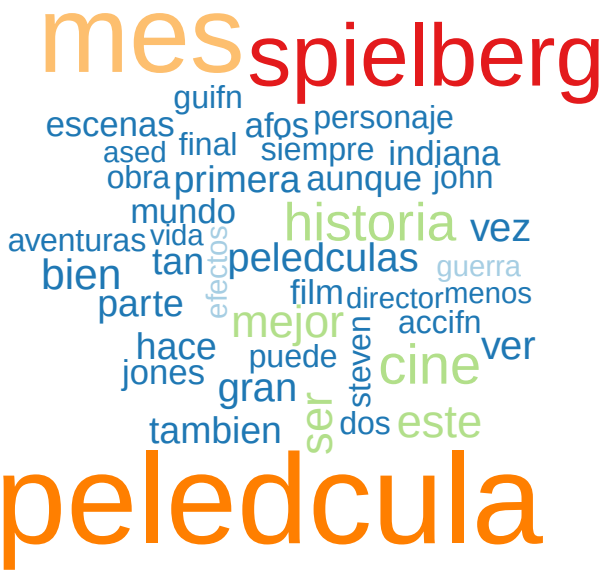
```
## [1] "Correlación entre negativas y RT score: -0.331879995904806"
```

Parece que nuestra hipótesis es correcta, ya que vemos tres cosas:

1. Existe correlación (0.42) entre nuestra puntuación y la de Rotten Tomatoes.
2. Hay una correlación ligeramente más fuerte (0.45) entre el porcentaje de palabras positivas y la puntuación de Rotten Tomatoes.
3. La correlación entre palabras negativas y puntuación de Rotten Tomatoes es débil (-0.33), lo que refuerza la hipótesis de que estamos tratando como negativas palabras que no lo son en este contexto.

Para entender mejor qué palabras están marcando la diferencia entre positivas y negativas vamos a generar nubes de palabras. Primero, una nube con todas las palabras del corpus.

```
m <- as.matrix(TermDocumentMatrix(corpus_clean))
m <- sort(rowSums(m), decreasing = TRUE)
colorPalette <- brewer.pal(8, "Paired")
wordcloud(names(m), m, min.freq = 200, colors=colorPalette)
```



Lo primero que nos llama la atención es que la palabra más repetída es 'peledcul', que es claramente un typo al escribir 'película'. Lo siguiente que vemos es que hay muchas palabras, como 'Spielberg' o 'historia' que deberíamos considerar stopwords en nuestro contexto.

Vamos a ver ahora la nube de palabras positivas.

```
m <- as.matrix(TermDocumentMatrix(corpus_clean, control = list(tokenize    = 'word',
                                                               dictionary  = listapositiva,
                                                               stopwords   = FALSE,
                                                               wordLengths = c(0, Inf),
                                                               stemming    = 'spanish')))
m <- sort(rowSums(m), decreasing = TRUE)
colorPalette <- brewer.pal(8, "Paired")
wordcloud(names(m), m, min.freq = 30, colors=colorPalette)
```

Parece que en general todas tienen bastante sentido. Veamos las negativas.

```
m <- as.matrix(TermDocumentMatrix(corpus_clean, control = list(tokenize    = 'word',
                                                               dictionary  = listanegativa,
                                                               stopwords   = FALSE,
                                                               wordLengths = c(0, Inf),
                                                               stemming    = 'spanish')))
m <- sort(rowSums(m), decreasing = TRUE)
colorPalette <- brewer.pal(8, "Paired")
wordcloud(names(m), m, min.freq = 20, colors=colorPalette)
```

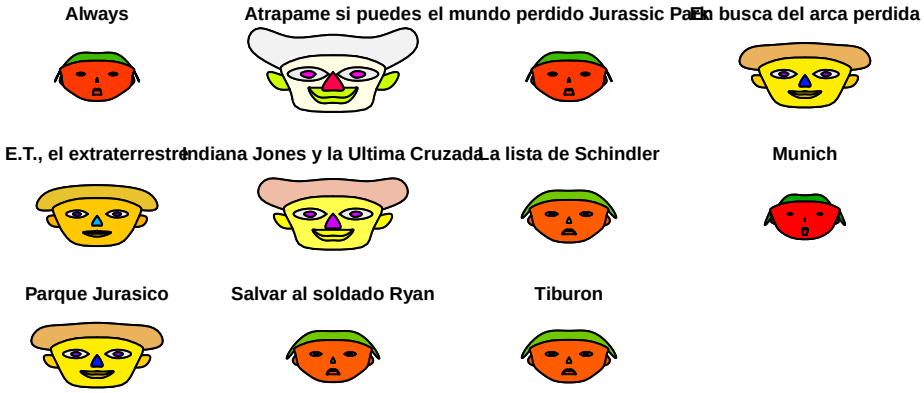
```
## Warning in wordcloud(names(m), m, min.freq = 20, colors = colorPalette):
```

```
## casi could not be fit on page. It will not be plotted.
```

Aquí vemos alguna inconsistencia que puede explicar la baja correlación, como 'terror' o 'soldado'.

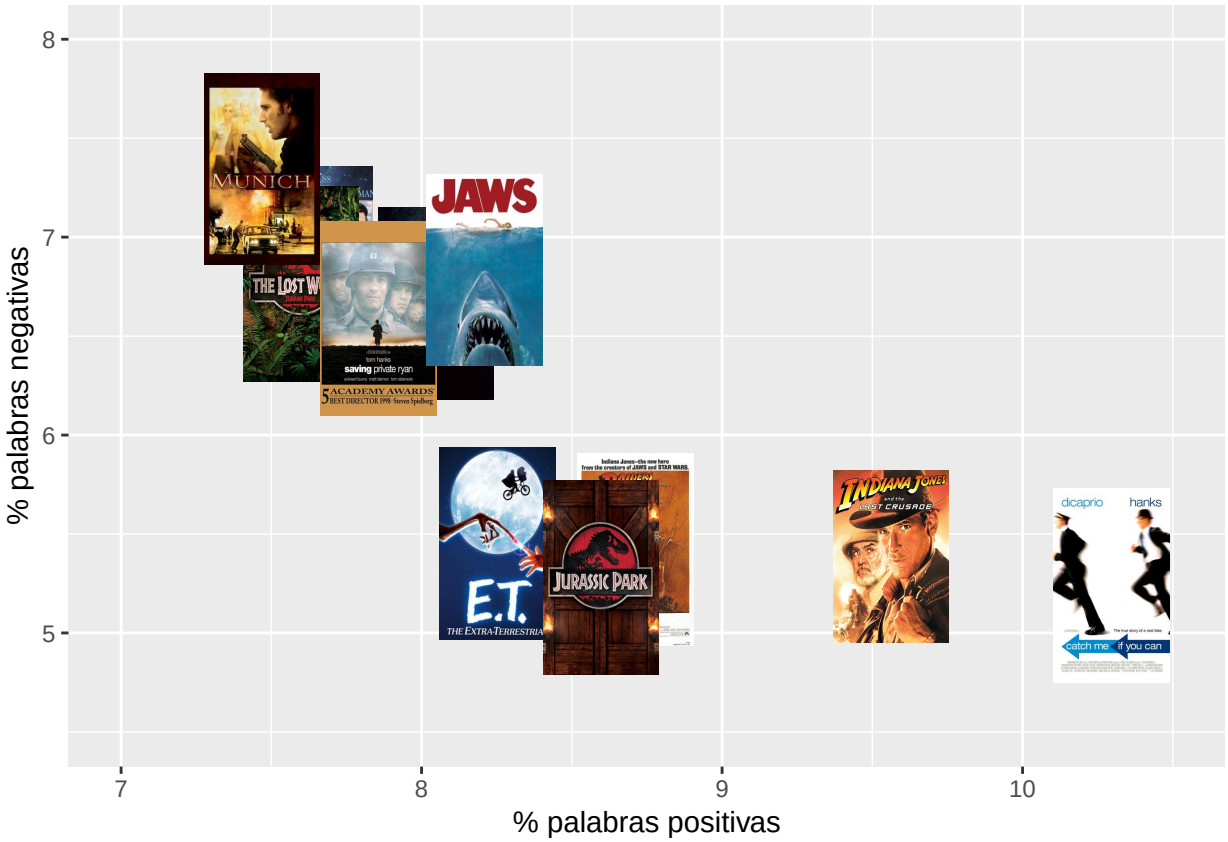
Vamos a terminar haciendo un plot de Chernoff Faces, para ver cómo de contentos están los personajes después de ver cada película (y por los memes, claro).

```
faces(scores[5:5] , labels = scores$movie, cex=0.9, print.info = FALSE)
```



Ya estamos! Siempre está bien una visualización chula, así que vamos a intentar una con las carátulas de las películas según su porcentaje de palabras positivas y negativas.

```
scores$image <- paste0('datos/imagenes/', scores$movie, '.jpg')
ggplot(scores, aes(x=positive, y=negative, image=image)) +
  geom_image(asp=1.7, size=0.1) + ylim(c(4.5, 8)) + xlim(7, 10.5) +
  xlab('% palabras positivas') + ylab('% palabras negativas')
```


8
7
6
5
% palabras negativas
7
8
9
10
% palabras positivas
MUNICH
JAWS
THE LOST W
saving private ryan
5 ACADEMY AWARDS
BEST DIRECTOR 1998 Steven Spielberg
E.T.
THE EXTRA-TERRESTRIAL
JURASSIC PARK
INDIANA JONES
THE LAST CRUSADE
dicaprio
hanks
catch me if you can