TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE TIJUANA

SUBDIRECCIÓN ACADÉMICA

DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN

NOMBRE DE LOS ALUMNOS:

RAYMUNDO HIRALES LAZARENO (N. CONTROL: 17212339)

GALAVIZ LONA OSCAR EDUARDO (N.CONTROL: 17212993)

Carrera: Ingeniería Informática

Semestre: 9no

MATERIA: Minería de datos

PROFESOR: JOSE CHRISTIAN ROMERO HERNANDEZ

TRABAJOS: Practica 1

FECHA: 1/11/21

```
#Hirales Lazareno Raymundo - 17212339
#Galaviz Lona Oscar Eduardo - 17212993

#Librerias requeridas
library( dplyr )
library( tidytext )
library( textdata )
library( ggplot2 )
library( RColorBrewer )
library( wordcloud )
library( reshape2 )
library( tidyverse )

# El archivo esta subido a un github para facilitar su carga dentro de RStudio
mlpURL <- "https://raw.githubusercontent.com/SmilodonCub/DATA607/master/my-little-pony-transcript/clean_dialog.csv"
mlp_df <- read.csv( mlpURL )
dim( mlp_df )

#
colnames( mlp_df )

#En este apartado exploraremos los sentimientos como una funcion de la narrativa de los subsecuentes episodios de MLP.
# La pregunta es como los sentimientos varian a lo largo del dialogo del episodio?
episodeLines <- mlp_df %>%
  group_by( title ) %>% #with respect to episode title:
  mutate( id = row_number()) %>% #add a new feature 'id' to enumerate each row of text
  group_by( title, id) %>% #with respect to episode title & line of text(id):
  rowwise() %>%
  summarise( lines = paste(dialog, collapse = "&&")) %>%
  #paste all episode lines together delimited by '&&'
  mutate( lines = str_split( lines, "&&") ) %>%
  #mutate lines to a list of lines
  unnest( lines ) %>% #unnest list of lines to one line per row
  unnest_tokens(word, lines) #one token/word per line
head(episodeLines)

#list of first 6 episode title
AllEpisodes <- unique(episodeLines$title)[1:6]

#perform an inner join with the bing lexicon
pony_sentiment <- episodeLines %>%
  filter( title %in% AllEpisodes ) %>% #subset for the first 8 episodes
  inner_join(get_sentiments("bing")) %>% #inner join with 'bing' lexicon
  #for each title, tally the sentiment score of the
  #tokens in increments of 10 lines
  count(title, index = id %/% 6, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)
head( pony_sentiment )
```

```
#Visualize sentiment over the course of episode narative:

colourCount = length(unique(pony_sentiment$title))
mycolors = colorRampPalette(brewer.pal(9, "PuRd"))(colourCount)

ggplot(pony_sentiment, aes(index, sentiment,color='black', fill = title)) +
  geom_point(show.legend = FALSE) +
  facet_wrap(~title, ncol = 2, scales = "free_x") +
  scale_fill_manual( values = mycolors) +
  labs( title = 'Sentiment Analysis', subtitle="Sentiment across episode
trajectory for the first 6 episodes")
```


```
#Proceso para comparar dos episodios para tener un analisis mas completo de la
narrativa
#combining 2 episodes to have a longer narative to analyze:
doubleEpisode <- c('A Canterlot Wedding - Part 1', 'A Canterlot Wedding - Part 2')
#a dataframe for the first episode:
doubleEpLines1 <- episodeLines %>%
  filter( title %in% doubleEpisode[1] )
addlines <- max(doubleEpLines1$id)
#a dataframe for the second that increments the line 'id'
doubleEpLines2 <- episodeLines %>%
  filter( title %in% doubleEpisode[2] ) %>%
  mutate( id = id + addlines )
#bind the two episodes to one dataframe
doubleEpLines <- rbind( doubleEpLines1, doubleEpLines2)

#inner join with 'afinn' and sum sentiment value for every 10 lines
afinn <- doubleEpLines %>%
  inner_join(get_sentiments("afinn")) %>%
  group_by(index = id %/% 10) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")
#inner join with 'bing' and 'nrc' also sum sentiment value for every 10 lines
bing_and_nrc <- bind_rows(doubleEpLines  %>%
                            inner_join(get_sentiments("bing")) %>%
                            mutate(method = "Bing et al."),
                          doubleEpLines  %>%
                            inner_join(get_sentiments("nrc") %>%
                                         filter(sentiment %in% c("positive",
                                                                 "negative"))) %>%
                            mutate(method = "NRC")) %>%
  count(method, index = id %/% 10, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)

#bind afinn, bing, and nrc data then visualize the narative sentiments
bind_rows(afinn,
          bing_and_nrc) %>%
```

```
  ggplot(aes(index, sentiment,color='black', fill = method)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~method, ncol = 1, scales = "free_y") +
  scale_fill_manual( values = mycolors[2:4]) +
  labs( title = 'Sentiment Analysis', subtitle="Sentiment for the same text with 3
different lexicons")

#This code demonstrates the differences in lexicon sentiment criterion:
nrc_sent <- get_sentiments("nrc") %>%
  filter(sentiment %in% c("positive",
                          "negative")) %>%
  count(sentiment)
nrc_ratio <- nrc_sent$n[1]/nrc_sent$n[2]

bing_sent <- get_sentiments("bing") %>%
  count(sentiment)
bing_ratio <- bing_sent$n[1]/bing_sent$n[2]

cat( 'The +/- ratio for nrc=', nrc_ratio, 'this < the +/- bing =',bing_ratio)
```


```
#Proceso para encontrar las palabras mas usadas de forma positiva y negativas
# Hacer un inner join para las etiquetas de sentimientos con toquen nos ayudara.
# aqui la manipulacion es usada para explorar las palabras usadas
#frequentemente para sentimientos positivos y negativos que aparecieron
#en los episodios my little pony
bing_word_counts <- episodeLines %>%
  group_by( word ) %>% #group with respect to word,
  summarise( n = n()) %>% #count a total for each words occurance
  inner_join(get_sentiments("bing")) %>% #join bing sentiments
  arrange( desc( n )) #arrange in descending order
head(bing_word_counts)

#Visualize the top 15 most frequent positive and negative words.

bing_word_counts %>%
  group_by(sentiment) %>%
  top_n(15,n) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n,color='black', fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Contribution to sentiment",
       x = NULL) +
  coord_flip() +
  scale_fill_manual( values = mycolors[3:4]) +
  labs( title = 'Most Common Words', subtitle="15 most frequent positive and
negative words that appear across all transcripts")

#There is a problem! In the top 10 negative words, ‘discord’ is listed.
```

```
#This is problematic, because Discord is the name of a villain in
#My Little Pony. We would like to add this nameto a list of stop words.
#Stop words are words that are to be excluded from further analysis.
#add 'discord' to the stop words...
custom_stop_words <- bind_rows(tibble(word = c("discord"),
                                      lexicon = c("custom")),
                               stop_words)
custom_stop_words
```


```
#Proceso para realizar nube de palabras
#basic word cloud
bing_word_counts %>%
  anti_join(stop_words) %>%
  with(wordcloud(word, n, max.words = 80,colors =
c("#F592AB","#BF408B","#B040BF","#8340BF")))

#word cloud that compares categorical tokens (positive vs negative sentiments)
bing_word_counts %>%
  inner_join(get_sentiments("bing")) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("cyan", "magenta"),
                   max.words = 100)
```


```
#Nuevo proceso para comparar a todos los personajes y sacar al mas positivo
ponies_top50 <- mlp_df %>%
  group_by( pony ) %>%
  summarise( count = n(), lines=paste(dialog, collapse="&&") ) %>%
  mutate(lines = str_split( lines, "&&")) %>%
  arrange( desc( count ) ) %>%
  top_n( 50, count ) %>%
  unnest( lines ) %>%
  unnest_tokens(word, lines)
unique(ponies_top50$pony)

#Acomodar
ponies_lineTally <- ponies_top50 %>%
  select( pony, count ) %>%
  group_by( pony ) %>%
  summarise( count = max(count) )
ponies_lineTally

#usa el get sentiment con la libreria syuzhey para construir los puntajes de
sentimientos
library( syuzhet )
```

```
ponies_top50$syuzhet <- get_sentiment(ponies_top50$word, method="syuzhet")
ponies_top50$bing <- get_sentiment(ponies_top50$word, method="bing")
ponies_top50$nrc <- get_sentiment(ponies_top50$word, method="nrc")

#junta a cada palabra con su respectivo pony y su calificacion
#group by pony and summarise the sums of the 3 lexicon scores
ponies_sentimentScores <- ponies_top50 %>%
  group_by( pony, count ) %>%
  summarise( syuzhetScore = sum( syuzhet ),
             bingScore = sum( bing ),
             nrcScore = sum( nrc ))
#normalize the scores to account for the number of lines delivered by each
character
ponies_sentimentScores <- ponies_sentimentScores %>%
  mutate( syuzhetScore = syuzhetScore/count,
          bingScore = bingScore/count,
          nrcScore = nrcScore/count)
summary( ponies_sentimentScores )


#pivot the data longer to facilitate plotting the distributions of scores by
lexicon
plotData <- ponies_sentimentScores %>%
  pivot_longer(cols = syuzhetScore:nrcScore, names_to = 'lexicon')
#visualize as box plot:
ggplot(plotData, aes(x=lexicon, y=value)) +
  geom_boxplot(color="#8340BF", fill="#BF408B", alpha=0.2,
               outlier.colour="#B040BF", outlier.fill="#B040BF", outlier.size=5) +
  labs( title = 'Lexicon Scores Compared:', subtitle="distribution of lexicon
scores normalized by lines delivered for top 50 characters")

colourCount = 15
mycolors = colorRampPalette(brewer.pal(50, "PuRd"))(colourCount)

plotData <- ponies_sentimentScores %>%
  arrange( desc( syuzhetScore )) %>%
  head( n = 15L )
ordered <- plotData$pony
ggplot(plotData, aes(x=pony, y=syuzhetScore, color='black',fill=factor(pony) )) +
  geom_bar( stat = 'identity' ) +
  scale_x_discrete( limits = rev(ordered)) +
  coord_flip() +
  scale_fill_manual(values = mycolors ) +
  theme(legend.position="none") +
  labs( title = 'Highest Syuzhet Score', subtitle = 'Top 15 ranked character
Syuzhet Scores normalized by #lines delivered')
```