El texto más parecido a *Castillos* de Amanda Miguel es…
El muchacho de los ojos tristes de Jeanette

Una valor más alto indica una correlación mayor. *Ramito de Violetas* es la más débil del grupo.

De izquiera a derecha: palabras en común de *Castillos* con *Ramito…*, *El muchacho…* y *Piedras Rodantes*. Mayor tamaño indica mayor frecuencia. La nube de *El muchacho* es más representativa de los temas de *Castillos*. En cambio, "piedras" en «Las **piedras** rodando se encuentran…» no transmite lo mismo que «…rompió mi ilusión con **piedras**».

Se hizo un análisis extrayendo las palabras de las canciones, eliminando *stop words*.
La correlación entre dos textos está dada por el coeficiente de Phi, que da valores más bajos cuando hay más palabras en solo uno de los textos, que cuando coincide la aparición de palabras. Estandarizando, da un valor entre -1 y 1 (en el Anexo B se explica con una matriz de confusión). A diferencia del coeficiente de Pearson, el 0 no significa que no hay correlación; en este caso el 0 es una correlación media.
En las nubes de palabras se muestra las palabras en común que aparecen una sola vez.

Es muy importante conocer el contexto de los datos y considerar diferentes perspectivas para dar una respuesta. Tener disponibles herramientas como las nubes de palabras y correlaciones da seguridad a la respuesta de la interrogante: basarse solo en un análisis de sentimiento daría una respuesta distinta (ver anexo).
Aunque esta tarea sea técnicamente basada en texto, es escencial escuchar las canciones para tomar una decisión adecuada. Así, se agotarán las técnicas de *data mining* que den sentido y justifiquen las razones por las cuales se puede decir que *Castillos* tiene un tono similar a *El muchacho*, basado en la repetición y contexto de las palabras extraídas.

Guía rápida: análisis de texto de canciones en R.

Paso 1: Preparar paqueterías

```
library(genius) #bajar canciones
library(tidyverse) #manipulación de datos
library(tidytext) #text mining
library(tm) #text mining: stop words,
library(wordcloud) #nubes de palabras
library(widyr) #correlaciones entre textos
```

Paso 2: Bajar canciones. Eliminar NA's si es necesario.

```
textoA <- genius_lyrics("amanda miguel", "castillos")
textoB <- genius_lyrics("mi banda el mexicano", "ramito de violetas")
textoC <- genius_lyrics("Jeanette", "El Muchacho de los Ojos Tristes")[-1,]
textoD <- genius_lyrics("El Tri", "Las Piedras Rodantes")
```

Paso 3: Extraer palabras en una columna llamada `word` a partir de la columna `lyric` del paso 1.

```
textos_palabras <- textos %>%
                   unnest_tokens(word, lyric)
```

Paso 4: Eliminar *stop words* con la lista en español del paquete `tm`. Agregar a la lista palabras innecesarias posteriormente.

```
custom_stop_words <- bind_rows(stop_words,
                               data_frame(word = stopwords("spanish"),
                                          lexicon = "custom"),
                               data_frame(word = c("tururu", "turututuru",
                                                   "turutututururu", "quién",
                                                   "asi", "cada", "mas"),
                                          lexicon = "custom"))

tidy_textos <- textos_palabras %>%
  anti_join(custom_stop_words)
```

Paso 5: Contar la frecuencia de palabras y ordenar descendentemente.

```
tidy_count <- tidy_textos %>%
              count(track_title, word, sort = TRUE)
```

Paso 6: Calcular correlaciones de cada canción con *Castillos*

```
textos_cors <- tidy_textos %>%
  pairwise_cor(track_title, word, sort = TRUE) %>%
  filter(item1 == 'Castillos')
```

Paso 7: Pasar la cuenta de palabras a TermDocumentMatrix y después a matriz para graficar la nube de palabras

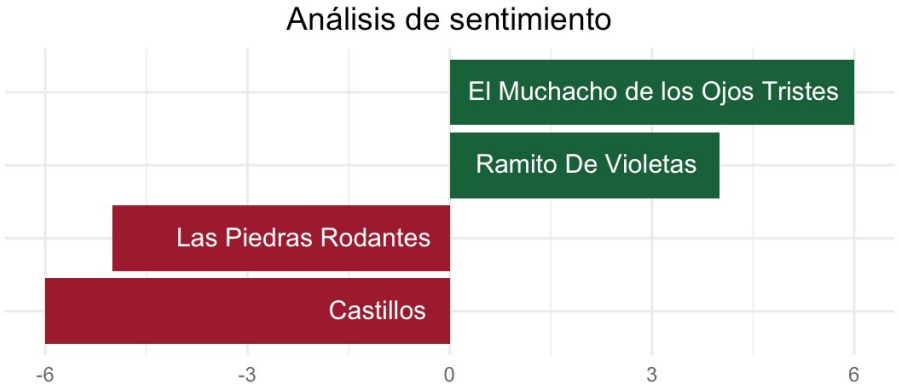
```
tidy_tdm <- tidy_count %>%
            cast_tdm(word, track_title, n)
tidy_matrix <- as.matrix(tidy_tdm)
```

Paso 8: Graficar nube de palabras en común de *Castillos* con cada una de las canciones. Asignar sombras de color similares. Desactivar orden aleatorio para que la más frecuente quede al centro.

```
commonality.cloud(tidy_matrix[,c(4,1)], # muchacho ojos tristes
                  colors = c('#15ef32', '#10b526', '#0e821d', '#084210'),
                  random.order = FALSE)
commonality.cloud(tidy_matrix[,c(4,2)], # piedras rodantes
                  colors = c('#ff77ab', '#c64d7c', '#99345a','#5e1531'),
                  random.order = FALSE)
commonality.cloud(tidy_matrix[,c(4,3)], # ramitos violetas
                  colors = c('#32c1ff', '#299cce', '#217da5', '#11475e'),
                  random.order = FALSE)
```

Anexo A: *Sentiment Analysis*

Se tenía pensado agregar los resultados del Sentiment Analysis para las canciones:



En el análisis se califican las palabras como positivas y negativas, las primeras se suman y las segundas se restan para dar un balance del sentimiento de la canción.

La idea era mostrar una herramienta más que podría usarse para tomar la decisión de qué canción se parece más a *Castillos*. Sin embargo, el espacio limitado implicó reducir las gráficas a mostrar, y se eliminó porque podría ser perjudicial para la justificación de la respuesta. NO QUIERE DECIR que la respuesta sea incorrecta, sino que una explicación detallada era necesaria, pero el espacio era insuficiente.

La canción *El muchacho…* es una canción triste, pero la repetición de palabras como "sonrisa", "amor" y "besos" le da un balance positivo, que hace que sea la más positiva de las cuatro canciones, al otro extremo de la canción objetivo, *Castillos.* Asimismo, *Las Piedras Rodantes* no es tan negativa como *Ramito de violetas*, pero la selección de diccionario de sentimiento puede haber influido en el resultado.

Es necesario agregar que para este análisis tuvo que buscarse la traducción de las canciones al inglés. Se verificó manualmente que el texto trasmitiera la misma idea, con números iguales de versos y palabras suficientes y necesarias.

Anexo B: *Coeficiente de Phi*

	Has word Y	No word Y	Total
Has word X	n_{11}	n_{10}	$n_{1\cdot}$
No word X	n_{01}	n_{00}	$n_{0\cdot}$
Total	$n_{\cdot 1}$	$n_{\cdot 0}$	n

$$\phi = \frac{n_{11}n_{00} - n_{10}n_{01}}{\sqrt{n_{1\cdot}n_{0\cdot}n_{\cdot 1}n_{\cdot 0}}}$$

Fuente:
https://bookdown.org/Maxine/tidy-text-mining/counting-and-correlating-pairs-of-words-with-widyr.html