```
# EPPS 6302 Group Project Analysis File
# By: Genna Campain, Ryan Gordon, Su Lin Goh, Michelle Kim

# Library
library(spotifyr)
library(magrittr)
library(geniusr)
library(dplyr)
library(tidyverse)
library(tidytext)
library(textdata)
library(stringr)
library(readr)

# Data loading and editing
load("~/Desktop/Fall 2022/Data methods/Group Project/lyrics_data.Rda")
final_df$num <- seq.int(from = 1, to = 250)
final_df$year <- 2018
final_df$year[which(final_df$num > 50 & final_df$num < 101)] <- 2019
final_df$year[which(final_df$num > 100 & final_df$num < 151)] <- 2020
final_df$year[which(final_df$num > 150 & final_df$num < 201)] <- 2021
final_df$year[which(final_df$num > 200)] <- 2022


# Paste all lyrics for year into one line of dataframe
df2 <- final_df %>%
  group_by(year) %>%
  summarise(col = paste(lyrics, collapse=" "))

# Text analysis
sp_stop_words <- spanish_stopwords <- read.table("~/Desktop/Fall 2022/Data methods/Group
Project/spanish_stopwords.txt", quote="\"", comment.char="") %>%
  rename(word = V1)
count_result <- data.frame()
afinn_result <- data.frame()
bing_and_nrc_result <- data.frame()
for(j in 2018:2022) {
  text <- df2$col[which(df2$year == j)]
  tibble <- tibble(line = 1, text = text)
  tidy_lyrics <- unnest_tokens(tibble, word, text)
  data(stop_words)
  tidy_lyrics <- tidy_lyrics %>%
    anti_join(stop_words) %>%
    anti_join(sp_stop_words)
  # Count of common words
  count <- tidy_lyrics %>%
    count(word, sort = TRUE) %>%
    mutate(year = j)
  count_result <- rbind(count_result, count)
  # Sentiment analysis using afinn
  afinn <- get_sentiments("afinn")
  afinn_text <- tidy_lyrics %>%
    inner_join(afinn) %>%
    summarise(sentiment = sum(value)) %>%
    mutate(method = "AFINN") %>%
    mutate(year = j)
  afinn_result <- rbind(afinn_result, afinn_text)
  # Sentiment analysis using bing and nrc
  bing_and_nrc <- bind_rows(
    tidy_lyrics %>%
      inner_join(get_sentiments("bing")) %>%
      mutate(method = "Bing et al."),
    tidy_lyrics %>%
      inner_join(get_sentiments("nrc") %>%
                   filter(sentiment %in% c("positive",
                                           "negative"))
      ) %>%
      mutate(method = "NRC")) %>%
    count(method, sentiment) %>%
    pivot_wider(names_from = sentiment,
                values_from = n,
                values_fill = 0) %>%
    mutate(sentiment = positive - negative) %>%
    mutate(year = j)
  bing_and_nrc_result <- rbind(bing_and_nrc_result, bing_and_nrc)
}

# Pull top 100 words of each year
top100words <- count_result %>%
  group_by(year) %>%
  mutate(rown = row_number()) %>%
  ungroup()
top100words <- top100words[which(top100words$rown <= 100), ]

## Replace profanity with placeholder letters
top100words$word[top100words$word == 'shit'] <- 's***'
top100words$word[top100words$word == 'nigga'] <- 'n****'
top100words$word[top100words$word == 'niggas'] <- 'n*****'
```

```
top100words$word[top100words$word == 'bitch'] <- 'b****'
top100words$word[top100words$word == 'fuck'] <- 'f***'
top100words$word[top100words$word == 'ass'] <- 'a**'
top100words$word[top100words$word == 'pussy'] <- 'p****'
top100words$word[top100words$word == 'fuckin'] <- 'f*****'
top100words$word[top100words$word == 'mothafuckin'] <- 'm****f*****'

## Word cloud with top 50 words
library(wordcloud2)
library(htmlwidgets)
words2018 <- top100words[which(top100words$year == 2018), ] %>%
  select(word, n)
words2019 <- top100words[which(top100words$year == 2019), ] %>%
  select(word, n)
words2020 <- top100words[which(top100words$year == 2020), ] %>%
  select(word, n)
words2021 <- top100words[which(top100words$year == 2021), ] %>%
  select(word, n)
words2022 <- top100words[which(top100words$year == 2022), ] %>%
  select(word, n)
hw1 = wordcloud2(words2018, shape = 'triangle')
hw2 = wordcloud2(words2019, shape = 'triangle')
hw3 = wordcloud2(words2020, shape = 'triangle')
hw4 = wordcloud2(words2021, shape = 'triangle')
hw5 = wordcloud2(words2022, shape = 'triangle')
saveWidget(hw1,"1.html",selfcontained = F)
webshot::webshot("1.html","1.png",vwidth = 1992, vheight = 1744, delay =10)
saveWidget(hw2,"2.html",selfcontained = F)
webshot::webshot("2.html","2.png",vwidth = 1992, vheight = 1744, delay =10)
saveWidget(hw3,"3.html",selfcontained = F)
webshot::webshot("3.html","3.png",vwidth = 1992, vheight = 1744, delay =10)
saveWidget(hw4,"4.html",selfcontained = F)
webshot::webshot("4.html","4.png",vwidth = 1992, vheight = 1744, delay =10)
saveWidget(hw5,"5.html",selfcontained = F)
webshot::webshot("5.html","5.png",vwidth = 1992, vheight = 1744, delay =10)

# Identify which words are not being matched with lexicon
text <- df2$col
tibble <- tibble(line = 1, text = text)
tidy_lyrics <- unnest_tokens(tibble, word, text)
data(stop_words)
tidy_lyrics <- tidy_lyrics %>%
  anti_join(stop_words) %>%
  anti_join(sp_stop_words)
```

```
# Pull out words not matched to afinn
afinn <- get_sentiments("afinn")
not_matched_afinn <- tidy_lyrics %>%
  anti_join(afinn) %>%
    unique()
# Pull out words not matched to bing and nrc
not_matched_bing <- tidy_lyrics %>%
    anti_join(get_sentiments("bing")) %>%
    unique()
not_matched_nrc <- tidy_lyrics %>%
    anti_join(get_sentiments("nrc"))%>%
    unique()
# Spanish language analysis
pos_sp <- read_csv("isol/positivas_mejorada.csv", col_names = FALSE) %>%
  mutate(score = 1)
neg_sp <- read_csv("isol/negativas_mejorada.csv", col_names = FALSE) %>%
  mutate(score = -1)
sp_sent <- rbind(pos_sp, neg_sp) %>%
  rename(word = X1)
# sp_text <- tidy_lyrics %>%
  # inner_join(sp_sent) %>%
  # summarise(sentiment = sum(score))

## Save as CSV files
setwd("~/Desktop/Fall 2022/Data methods/Group Project")
write.csv(not_matched_afinn,"Not matched afinn.csv", row.names = FALSE)
write.csv(not_matched_bing_and_nrc,"Not matched bingnrc.csv", row.names = FALSE)

# Combine bing, spanish and self-created sentiment values
bing_sent <- get_sentiments("bing") %>%
  rename(score = sentiment)
bing_sent$score[bing_sent$score == 'positive'] <- 1
bing_sent$score[bing_sent$score == 'negative'] <- -1
our_sent <- read_excel("Not matched bingnrc.xlsx") %>%
  rename(score = "...3") %>%
  na.omit() %>%
  select(word, score)
mix_sent <- rbind(bing_sent, sp_sent) %>%
  rbind(our_sent)
mix_sent$score <- as.numeric(mix_sent$score)

bing_sp_sent <- rbind(bing_sent, sp_sent)
bing_sp_sent$score <- as.numeric(bing_sp_sent$score)
```

```
# Pull out words not matched to new lexicons
not_matched_spb <- tidy_lyrics %>%
  anti_join(bing_sp_sent)%>%
  unique()
not_matched_mix <- tidy_lyrics %>%
  anti_join(mix_sent)%>%
  unique()

# Analyze with new values
mix_result <- data.frame()
for(j in 2018:2022) {
  text <- df2$col[which(df2$year == j)]
  tibble <- tibble(line = 1, text = text)
  tidy_lyrics <- unnest_tokens(tibble, word, text)
  data(stop_words)
  tidy_lyrics <- tidy_lyrics %>%
    anti_join(stop_words) %>%
    anti_join(sp_stop_words)
  # Sentiment analysis using self-created lexicon
  mix_text <- tidy_lyrics %>%
    inner_join(mix_sent) %>%
    summarise(sentiment = sum(score)) %>%
    mutate(method = "mixed") %>%
    mutate(year = j)
  mix_result <- rbind(mix_result, mix_text)
}

bing_sp_result <- data.frame()
for(j in 2018:2022) {
  text <- df2$col[which(df2$year == j)]
  tibble <- tibble(line = 1, text = text)
  tidy_lyrics <- unnest_tokens(tibble, word, text)
  data(stop_words)
  tidy_lyrics <- tidy_lyrics %>%
    anti_join(stop_words) %>%
    anti_join(sp_stop_words)
  # Sentiment analysis using self-created lexicon
  bing_sp_text <- tidy_lyrics %>%
    inner_join(bing_sp_sent) %>%
    summarise(sentiment = sum(score)) %>%
    mutate(method = "bing_sp") %>%
    mutate(year = j)
  bing_sp_result <- rbind(bing_sp_result, bing_sp_text)
}
```

```r
## Create visualizations
bing <- bing_and_nrc_result[which(bing_and_nrc_result$method == "Bing et al."), ] %>%
  select(sentiment, year) %>%
  mutate(method = "Bing")
nrc <- bing_and_nrc_result[which(bing_and_nrc_result$method == "NRC"), ]  %>%
  select(sentiment, year) %>%
  mutate(method = "NRC")
afinn <- afinn_result %>%
  select(sentiment, year, method)
bing_sp <- bing_sp_result %>%
  select(sentiment, year) %>%
  mutate(method = "Bing_sp")
mix <- mix_result %>%
  select(sentiment, year) %>%
  mutate(method = "Mix")
big_df <- rbind(bing, nrc) %>%
  rbind(afinn) %>%
  rbind(bing_sp) %>%
  rbind(mix) %>%
  rename(Year = year, Method = method)
big_df$Year <- as.factor(big_df$Year)
save(big_df, file="all sentiments combined.Rda")

### library(ggplot2)
### library(RColorBrewer)
### library(ggthemes)
ggplot(big_df,
       aes(x = Year,
           y = sentiment,
           fill = Method)) +
  geom_bar(stat = "identity",
           position = "dodge") +
  theme_minimal() +
  labs(x = "Year", y = "Sentiment Score") +
  scale_fill_brewer(palette = "Set1")

# Calculate summary statistics for audio analysis
load("~/Desktop/Fall 2022/Data methods/Group Project/playlist audio features.Rda")
sum_songs_ana <- songs_ana %>%
  group_by(year) %>%
  summarise(m_dance = mean(danceability),
            m_valence = mean(valence),
            m_acoustic = mean(acousticness))
```

```
sum_songs_ana$year <- as.factor(sum_songs_ana$year)

songs_ana$year <- as.factor(songs_ana$year)

ggplot(songs_ana, aes(valence, fill = year)) +
  geom_density() +
  facet_grid(year ~ .) +
  theme_minimal() +
  theme(legend.position = "none") +
  scale_fill_brewer(palette = "Set1") +
  labs(x = "Valence", y = "Density")

ggplot(songs_ana, aes(danceability, fill = year)) +
  geom_density() +
  facet_grid(year ~ .) +
  theme_minimal() +
  theme(legend.position = "none") +
  scale_fill_brewer(palette = "Set1") +
  labs(x = "Danceability", y = "Density")

ggplot(songs_ana, aes(acousticness, fill = year)) +
  geom_density() +
  facet_grid(year ~ .) +
  theme_minimal() +
  theme(legend.position = "none") +
  scale_fill_brewer(palette = "Set1") +
  labs(x = "Acousticness", y = "Density")

df1 <- sum_songs_ana %>%
  select(year, m_dance) %>%
  mutate(Characteristic = "Danceability") %>%
  rename(Value = m_dance)
df2 <- sum_songs_ana %>%
  select(year, m_valence) %>%
  mutate(Characteristic = "Valence") %>%
  rename(Value = m_valence)
df3 <- sum_songs_ana %>%
  select(year, m_acoustic) %>%
  mutate(Characteristic = "Acousticness") %>%
  rename(Value = m_acoustic)
sum_songs_ana <- rbind(df1, df2) %>%
  rbind(df3)
sum_songs_ana$Year <- as.factor(sum_songs_ana$year)
```

```
ggplot(sum_songs_ana,
      aes(x = Year,
          y = Value,
          fill = Characteristic)) +
  geom_bar(stat = "identity",
          position = "dodge") +
  theme_minimal() +
  labs(x = "Year", y = "Value") +
  scale_fill_brewer(palette = "Set1")

# Unique words from each year
count_result <- data.frame()
for(j in 2018:2022) {
  text <- df2$col[which(df2$year == j)]
  tibble <- tibble(line = 1, text = text)
  tidy_lyrics <- unnest_tokens(tibble, word, text)
  data(stop_words)
  tidy_lyrics <- tidy_lyrics %>%
    anti_join(stop_words) %>%
    anti_join(sp_stop_words)

  # Count of common words
  count <- tidy_lyrics %>%
    count(word, sort = TRUE) %>%
    mutate(year = j)
  count_result <- rbind(count_result, count)
}
words2018 <- count_result[which(count_result$year == 2018), ] %>%
  select(word)
words2019 <- count_result[which(count_result$year == 2019), ] %>%
  select(word)
words2020 <- count_result[which(count_result$year == 2020), ] %>%
  select(word)
words2021 <- count_result[which(count_result$year == 2021), ] %>%
  select(word)
words2022 <- count_result[which(count_result$year == 2022), ] %>%
  select(word)

u_words2020 <- words2020 %>%
  anti_join(words2018) %>%
  anti_join(words2019) %>%
  anti_join(words2021) %>%
  anti_join(words2022) %>%
  inner_join(count_result[which(count_result$year == 2020), ]) %>%
```

```r
  select(word, n)
u_words2020 <- u_words2020[1:100,]
u_words2020$word[u_words2020$word == 'shit'] <- 's***'
u_words2020$word[u_words2020$word == 'nigga'] <- 'n****'
u_words2020$word[u_words2020$word == 'niggas'] <- 'n*****'
u_words2020$word[u_words2020$word == 'bitch'] <- 'b****'
u_words2020$word[u_words2020$word == 'fuck'] <- 'f***'
u_words2020$word[u_words2020$word == 'ass'] <- 'a**'
u_words2020$word[u_words2020$word == 'pussy'] <- 'p****'
u_words2020$word[u_words2020$word == 'fuckin'] <- 'f*****'
u_words2020$word[u_words2020$word == 'mothafuckin'] <- 'm****f*****'
hw7 = wordcloud2(u_words2020, shape = 'triangle')
saveWidget(hw7,"7.html",selfcontained = F)
webshot::webshot("7.html","7.png",vwidth = 1992, vheight = 1744, delay =10)



# Analyzing covid-specific songs
text <- final_df2$col
tibble <- tibble(line = 1, text = text)
tidy_lyrics <- unnest_tokens(tibble, word, text)
data(stop_words)
tidy_lyrics <- tidy_lyrics %>%
  anti_join(stop_words)
# Count of common words
count <- tidy_lyrics %>%
  count(word, sort = TRUE)
# Sentiment analysis using afinn
afinn <- get_sentiments("afinn")
afinn_text <- tidy_lyrics %>%
  inner_join(afinn) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")
# Sentiment analysis using bing and nrc
bing_and_nrc <- bind_rows(
  tidy_lyrics %>%
    inner_join(get_sentiments("bing")) %>%
    mutate(method = "Bing et al."),
  tidy_lyrics %>%
    inner_join(get_sentiments("nrc") %>%
             filter(sentiment %in% c("positive",
                                     "negative"))
    ) %>%
    mutate(method = "NRC")) %>%
  count(method, sentiment) %>%
  pivot_wider(names_from = sentiment,
            values_from = n,
            values_fill = 0) %>%
  mutate(sentiment = positive - negative)
## word cloud
## Replace profanity with placeholder letters
count$word[count$word == 'shit'] <- 's***'
count$word[count$word == 'nigga'] <- 'n****'
count$word[count$word == 'niggas'] <- 'n*****'
count$word[count$word == 'bitch'] <- 'b****'
count$word[count$word == 'fuck'] <- 'f***'
count$word[count$word == 'ass'] <- 'a**'
count$word[count$word == 'pussy'] <- 'p****'
count$word[count$word == 'fuckin'] <- 'f*****'
count$word[count$word == 'mothafuckin'] <- 'm****f*****'
hw6 = wordcloud2(count, shape = 'triangle')
saveWidget(hw6,"6.html",selfcontained = F)
webshot::webshot("6.html","6.png",vwidth = 1992, vheight = 1744, delay =10)

# Analyze counts of words not being matched
not_matched_d <- c("AFINN", 5846, "Bing", 5680, "NRC", 5421, "Bing_sp", 5502, "Mix", 4618)
not_matched <- matrix(not_matched_d, nrow = 5, ncol = 2, byrow = TRUE) %>%
  as.data.frame()
colnames(not_matched) <- c("Lexicon", "Unmatched")
not_matched$total <- 37093
not_matched$Unmatched <- as.numeric(not_matched$Unmatched)
not_matched$pct_unmatched <- (not_matched$Unmatched/not_matched$total)*100
ggplot(not_matched,
      aes(x = Lexicon,
         y = pct_unmatched,
         fill = Lexicon)) +
  geom_bar(stat = "identity",
         position = "dodge", show.legend = FALSE) +
  theme_minimal() +
  labs(x = "Lexicon", y = "Percent of Words Unmatched") +
  scale_fill_brewer(palette = "Set1")
```