initial eda

margaret

4/4/2022

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(spotifyr)
library(billboard)
library(tidytext)
```

```
##
## Attaching package: 'tidytext'

## The following object is masked from 'package:spotifyr':
##
##     tidy
```

```r
# access_token <- get_spotify_access_token()
```

```r
data1 <- spotify_track_data
data2 <- wiki_hot_100s
tracks <- read_csv("../data/track_info.csv")
```

```
## Rows: 3535 Columns: 29
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (17): href, id, name, preview_url, type, uri, album.album_type, album.hr...
## dbl  (5): disc_number, duration_ms, popularity, track_number, album.total_tr...
## lgl  (7): artists, available_markets, explicit, is_local, album.artists, alb...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
artists <- read_csv("../data/artist_info.csv")
```

```
## Rows: 1471 Columns: 11
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (6): href, id, name, type, uri, external_urls.spotify
## dbl (2): popularity, followers.total
## lgl (3): genres, images, followers.href
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```r
long_genres <- read_csv("../data/artist_genres.csv")
```

```
## Rows: 6681 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): artist_id, genres, href, name, type, uri, external_urls.spotify
## dbl (4): latest_release, med_release, popularity, followers.total
## lgl (2): images, followers.href
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```r
data <- full_join(
  spotify_track_data, wiki_hot_100s,
  by = c("artist_name" = "artist", "track_name" = "title")
)
```

```r
data2 <- data2 %>%
  mutate(
    featured = str_extract(title, "(?<=featuring ).*"),
    cleaned_track =
      ifelse(str_detect(title, "featuring"),
             str_extract(title, ".*( featuring*)"), title) %>%
      str_to_upper()
  )
data1 <- data1 %>%
  mutate(
    cleaned_track = track_name %>%
      str_replace("&", "and") %>%
      str_to_upper()
  )
data <- right_join(
  data1, data2,
  by = c("artist_name" = "artist", "cleaned_track" = "cleaned_track")
)
```

```r
data %>%
  drop_na(no) %>%
  group_by(artist_name) %>%
  summarize(
    mean_place = mean(as.numeric(no)),
    count = n(),
    most_recent = max(as.numeric(year.y))
  ) %>%
  filter(count > 5) %>%
  arrange(desc(count), desc(mean_place))
```

```
## Warning in mean(as.numeric(no)): NAs introduced by coercion

## Warning in mean(as.numeric(no)): NAs introduced by coercion

## Warning in mean(as.numeric(no)): NAs introduced by coercion

## Warning in mean(as.numeric(no)): NAs introduced by coercion

## Warning in mean(as.numeric(no)): NAs introduced by coercion
```

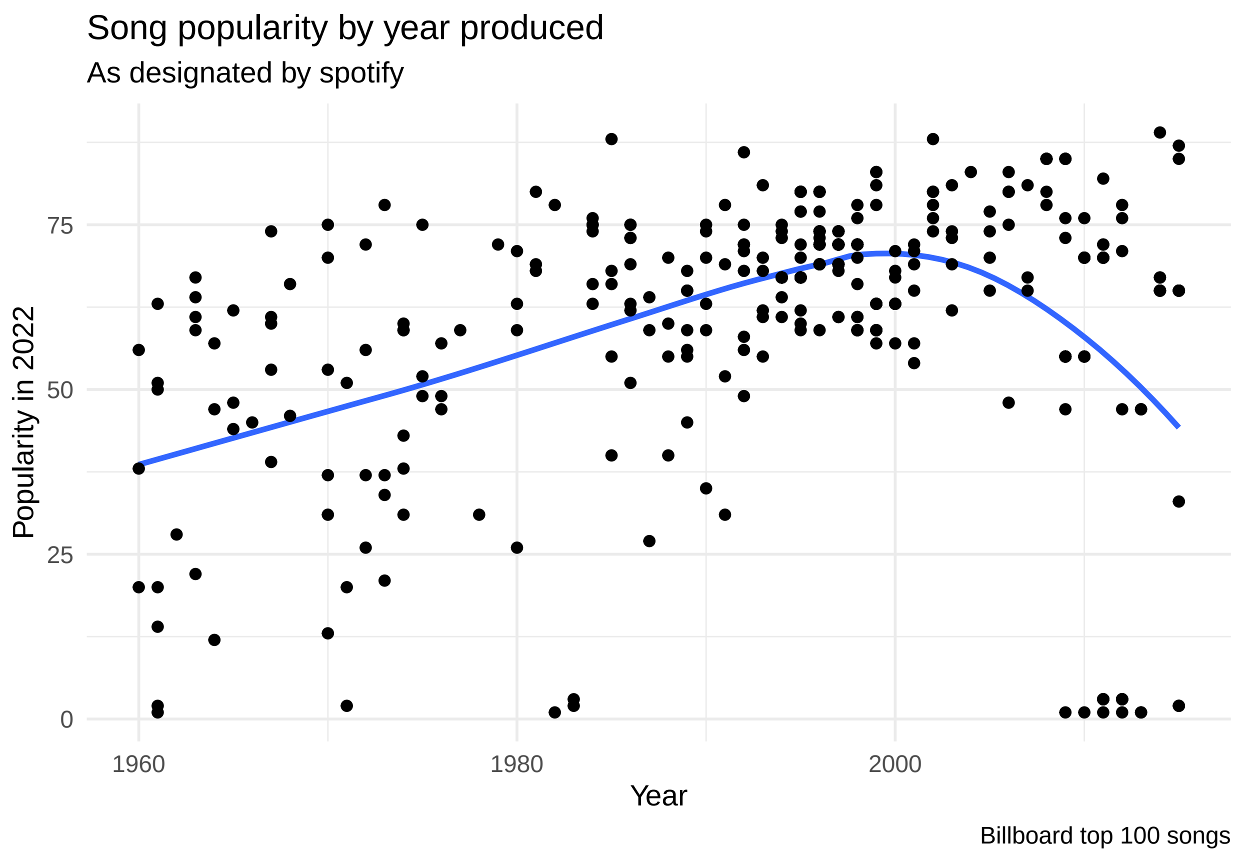
```
## # A tibble: 212 x 4
##    artist_name     mean_place count most_recent
##    <chr>                <dbl> <int>       <dbl>
##  1 Madonna               50.8    37        2006
##  2 Mariah Carey          30.2    28        2009
##  3 Elton John            45.6    27        1998
##  4 The Beatles           43.5    26        1976
##  5 Janet Jackson         36.8    26        2001
##  6 Taylor Swift          41.6    25        2016
##  7 Rihanna               42.1    24        2016
##  8 Kelly Clarkson        55.0    23        2015
##  9 Stevie Wonder         48.3    22        1986
## 10 Michael Jackson       47.4    22        2002
## # ... with 202 more rows
```

```
track_data <- inner_join(
  data,
  tracks,
  by = c("track_id" = "id")
)

big_data <- left_join(track_data, artists, by = c("artist_id" = "id"))
```

```
track_data %>%
 filter(as.numeric(no) < 11 & popularity != "0") %>%
  ggplot(
    aes(
      x = as.numeric(year.x), y = as.numeric(popularity)
    )
  ) +
  geom_smooth(se = F) +
  geom_point() +
  labs(
    x = "Year",
```

```r
    y = "Popularity in 2022",
    title = "Song popularity by year produced",
    subtitle = "As designated by spotify",
    caption = "Billboard top 100 songs"
  ) +
  theme_minimal()
```

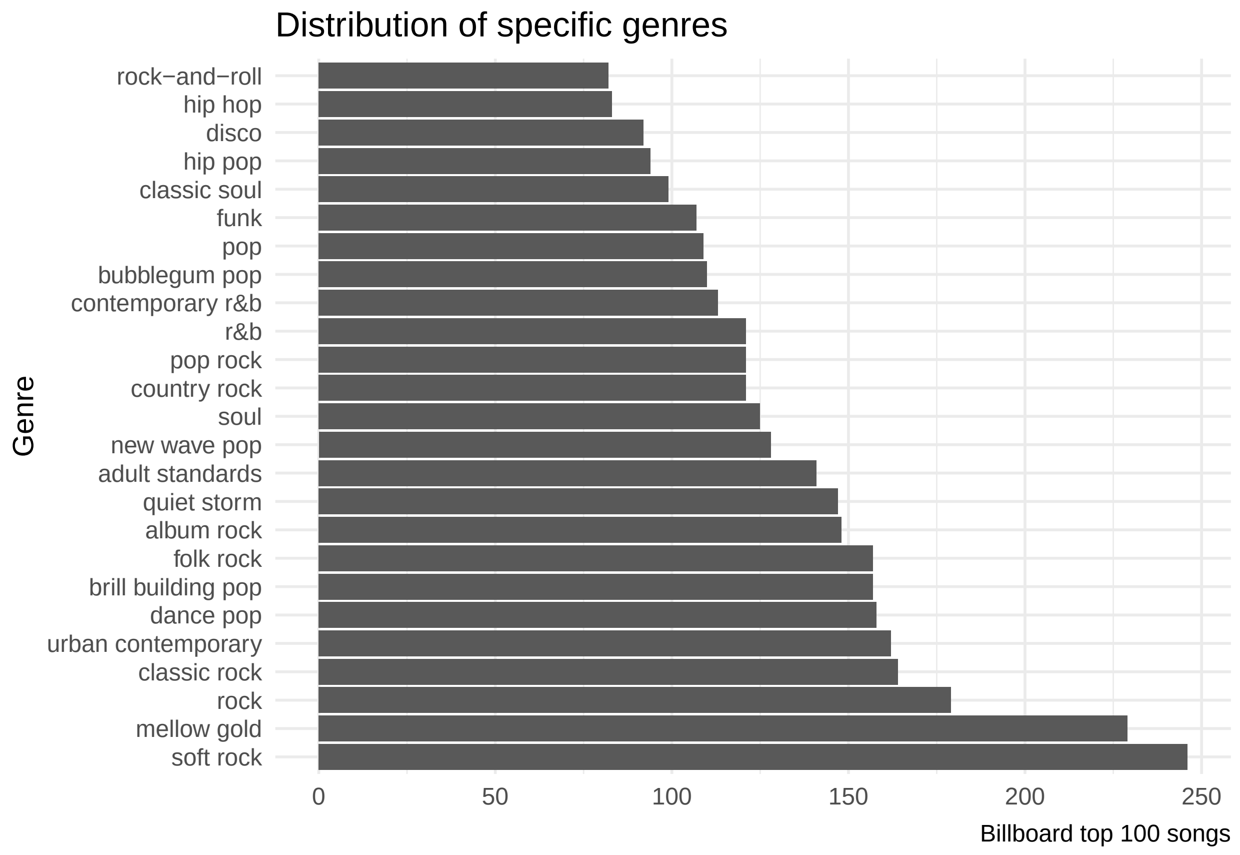
```
## Warning in mask$eval_all_filter(dots, env_filter): NAs introduced by coercion

## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



```r
artist_year <- track_data %>%
  group_by(artist_id) %>%
  summarize(
    latest_release = max(as.numeric(year.x)),
    med_release = median(as.numeric(year.x))
    ) %>%
  inner_join(artists, by = c("artist_id" = "id")) %>%
  group_by(latest_release)
```

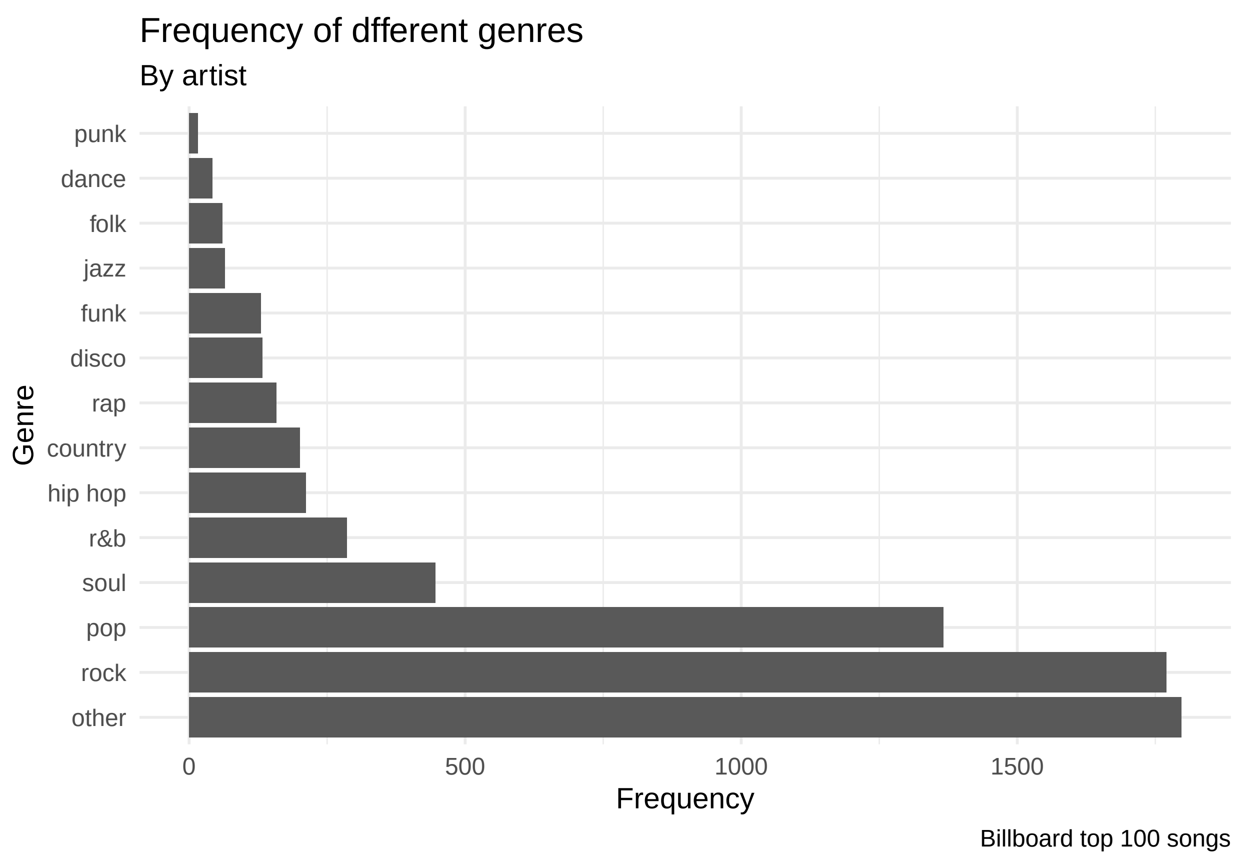
```r
long_genres %>%
  rename("genre" = "genres") %>%
  count(genre) %>%
  arrange(desc(n)) %>%
  slice(1:25) %>%
  ggplot(aes(y = fct_inorder(genre, n), x = n)) +
  geom_col() +
  labs(
    x = NULL,
    y = "Genre",
    title = "Distribution of specific genres",
    caption = "Billboard top 100 songs"
  ) +
  theme_minimal()
```

```
## Warning in if (is.na(ordered)) {: the condition has length > 1 and only the
## first element will be used

## Warning in if (ordered) "ordered": the condition has length > 1 and only the
## first element will be used

## Warning in if (is.na(ordered)) {: the condition has length > 1 and only the
## first element will be used

## Warning in if (ordered) "ordered": the condition has length > 1 and only the
## first element will be used
```

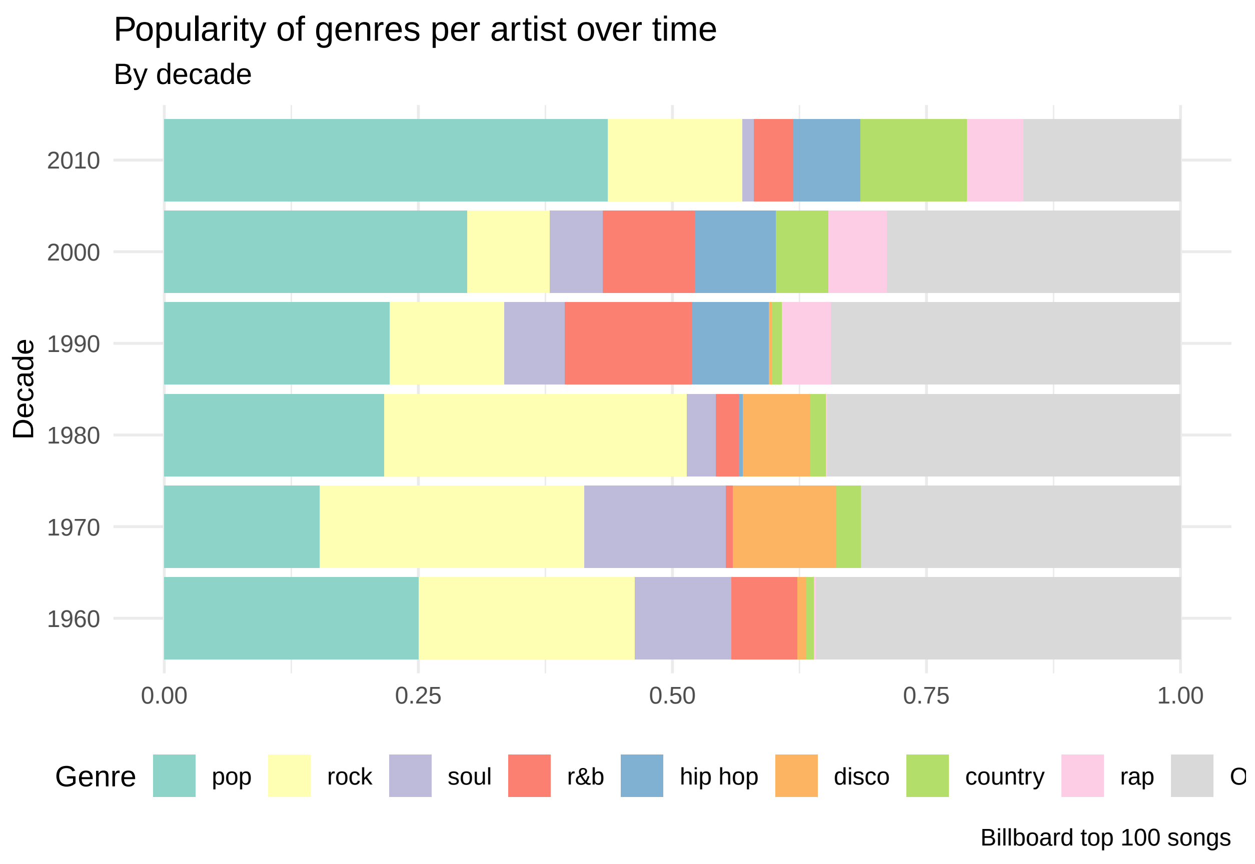


```r
long_genres %>%
  rename("genre" = "genres") %>%
  mutate(
    simple_genre = case_when(
      str_detect(genre, "pop") ~ "pop",
      str_detect(genre, "rock") ~ "rock",
      str_detect(genre, "r&b") | str_detect(genre, "rhythm and blues") ~ "r&b",
      str_detect(genre, "soul") ~ "soul",
      str_detect(genre, "country") ~ "country",
      str_detect(genre, "punk") ~ "punk",
      str_detect(genre, "jazz") ~ "jazz",
      str_detect(genre, "folk") ~ "folk",
      str_detect(genre, "rap") ~ "rap",
      str_detect(genre, "dance") ~ "dance",
      str_detect(genre, "disco") ~ "disco",
      str_detect(genre, "funk") ~ "funk",
      str_detect(genre, "hip hop") ~ "hip hop",
      TRUE ~ "other"
    )
  ) %>%
  count(simple_genre) %>%
  arrange(desc(n)) %>%
  ggplot(aes(y = fct_inorder(simple_genre, n), x = n)) +
  geom_col() +
  labs(
    x = "Frequency",
    y = "Genre",
    title = "Frequency of dfferent genres",
    subtitle = "By artist",
    caption = "Billboard top 100 songs"
  ) +
  theme_minimal()
```

```
## Warning in if (is.na(ordered)) {: the condition has length > 1 and only the
## first element will be used

## Warning in if (ordered) "ordered": the condition has length > 1 and only the
## first element will be used

## Warning in if (is.na(ordered)) {: the condition has length > 1 and only the
## first element will be used

## Warning in if (ordered) "ordered": the condition has length > 1 and only the
## first element will be used
```

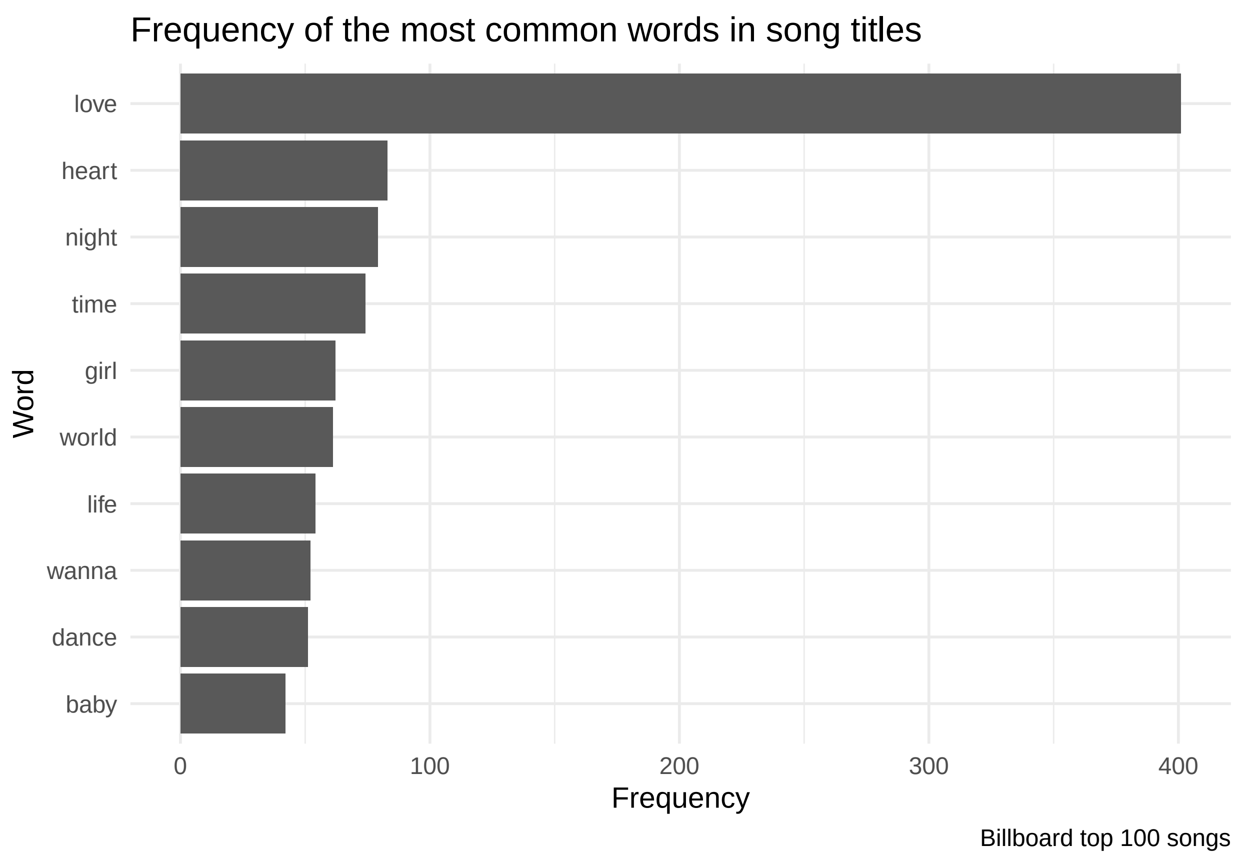


```r
long_genres %>%
  mutate(
    simple_genre = case_when(
      str_detect(genres, "pop") ~ "pop",
      str_detect(genres, "rock") ~ "rock",
      str_detect(genres, "r&b") | str_detect(genres, "rhythm and blues") ~ "r&b",
      str_detect(genres, "soul") ~ "soul",
      str_detect(genres, "country") ~ "country",
      str_detect(genres, "rap") ~ "rap",
      str_detect(genres, "disco") ~ "disco",
      str_detect(genres, "hip hop") ~ "hip hop",
      TRUE ~ "other"
    )
  ) %>%
  select(-genres) %>%
  distinct() %>%
  mutate(
    decade = floor(med_release/10)*10,
    simple_genre = fct_other(fct_infreq(simple_genre), drop = c("other"))
  ) %>%
  count(decade, simple_genre) %>%
  ggplot(aes(x = n, y = as.factor(decade), fill = fct_rev(simple_genre))) +
  geom_col(position = "fill") +
  scale_fill_brewer(palette = "Set3", direction = -1) +
  labs(
    x = NULL,
    y = "Decade",
    fill = "Genre",
    title = "Popularity of genres per artist over time",
    subtitle = "By decade",
    caption = "Billboard top 100 songs"
  ) +
  guides(fill = guide_legend(nrow = 1, reverse = T)) +
  theme_minimal() +
  theme(legend.position = "bottom", legend.direction = "horizontal")
```



```
track_data %>%
  select(year.x, artist_name, track_name, popularity, album.name) %>%
  unnest_tokens(word, track_name) %>%
  mutate(
    decade = as.factor(floor(as.numeric(year.x)/10)*10)
    ) %>%
  count(word, sort = TRUE) %>%
  anti_join(stop_words) %>%
  slice(1:10) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(
    y = "Word",
    x = "Frequency",
    title = "Frequency of the most common words in song titles",
    caption = "Billboard top 100 songs"
    ) +
  theme_minimal()
```

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

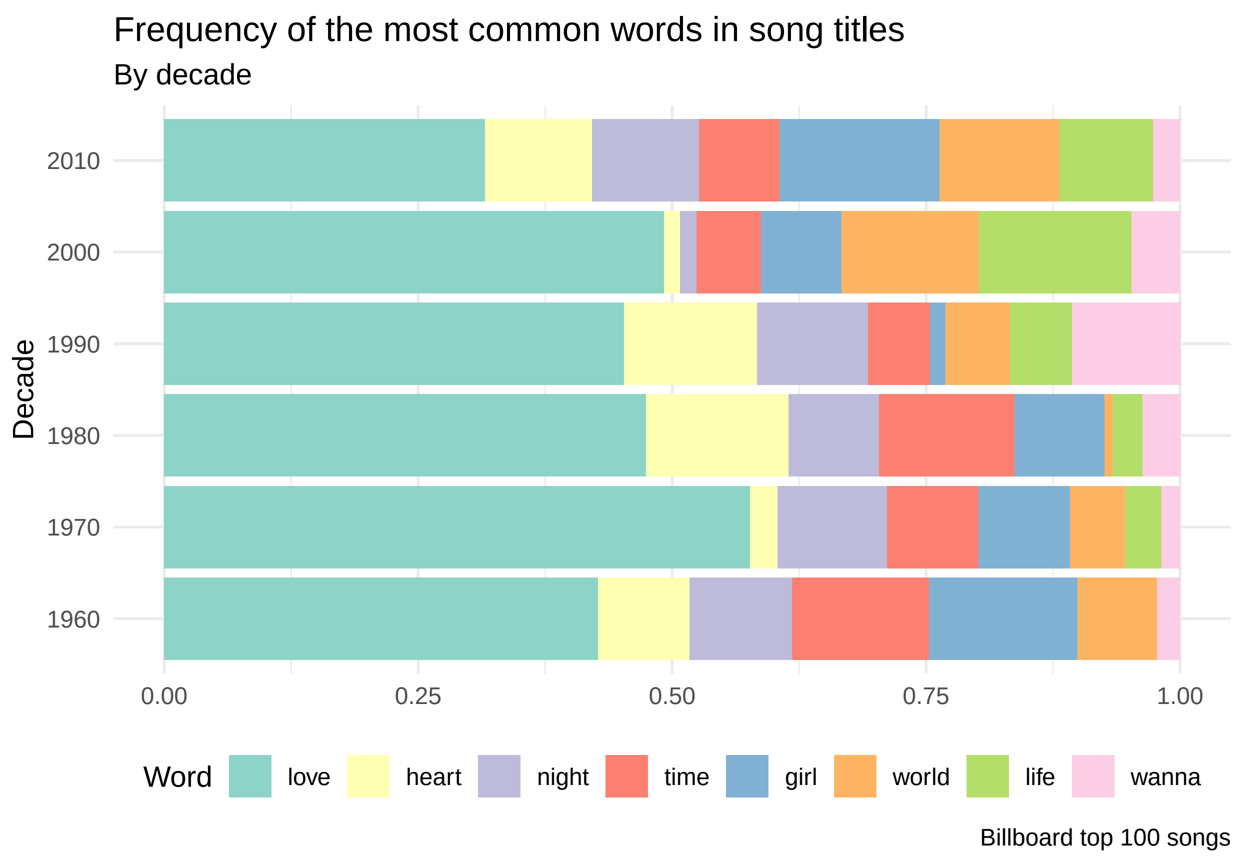


```
common_words <- track_data %>%
  select(year.x, artist_name, track_name, popularity, album.name) %>%
  unnest_tokens(word, track_name) %>%
  mutate(
    decade = as.factor(floor(as.numeric(year.x)/10)*10)
    ) %>%
  count(word, sort = TRUE) %>%
  anti_join(stop_words) %>%
  slice(1:8) %>%
  pull(word)
```

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

```
track_data %>%
  select(year.x, artist_name, track_name, popularity, album.name) %>%
  unnest_tokens(word, track_name) %>%
  mutate(
    decade = as.factor(floor(as.numeric(year.x)/10)*10),
    word = fct_infreq(word)
    ) %>%
  count(decade, word, sort = TRUE) %>%
  anti_join(stop_words) %>%
  filter(word %in% common_words) %>%
  ggplot(aes(x = n, y = decade, fill = fct_rev(word))) +
  geom_col(position = "fill") +
  scale_fill_brewer(palette = "Set3", direction = -1) +
  labs(
    y = "Decade",
    x = NULL,
    fill = "Word",
    title = "Frequency of the most common words in song titles",
    subtitle = "By decade",
    caption = "Billboard top 100 songs"
    ) +
  guides(fill = guide_legend(nrow = 1, reverse = T)) +
  theme_minimal() +
  theme(legend.position = "bottom", legend.direction = "horizontal")
```

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



```r
a <- track_data %>%
  filter( year.x == 1980, no == 1) %>%
  pull(track_name)

a[[1]]
```

```
## [1] "Call Me"
```