Spotify Recommendations

Duncan Gates

30 November, 2020

Some quick cleaning

The dataset came from Kaggle but was still a little messy, not sure why the artists column came wrapped in ['name']. Additionally some date formatting was inconsistent (as always), I imputed -01-01 if there was no date after the year.

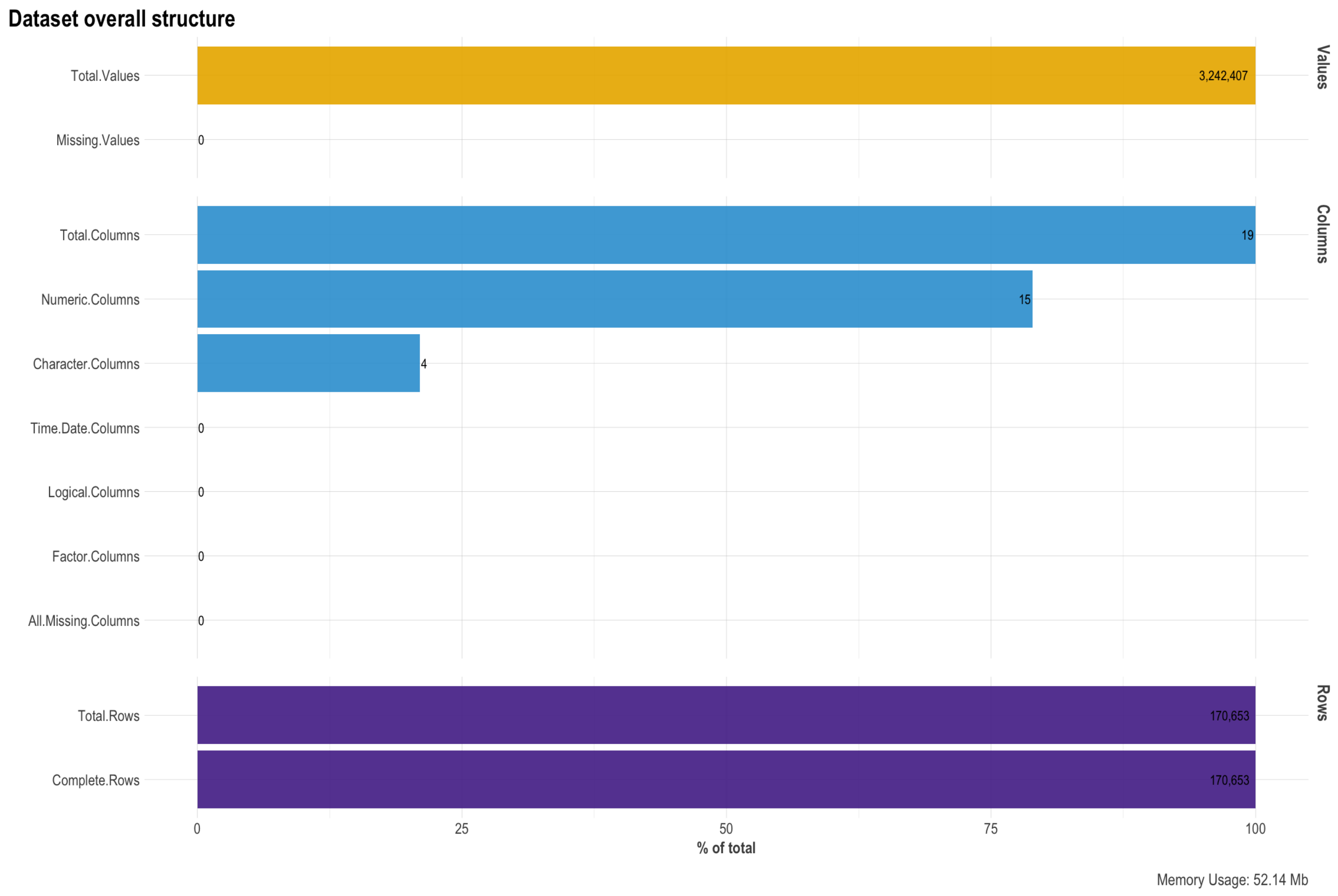
```
                    df2 <- df %>% mutate(artists = str_remove_all(artists, "\\['"),
                                         artists = str_remove_all(artists, "\\']"),
                                           artists = gsub(",", " and ", artists),
                            artists = str_remove_all(artists, "^'|'$"), # work on this a bit mo
                                                      re
                                decade = as.factor(floor(year/10)*10), # make decade column
                                                  year = as.factor(year),
                             release_date = ifelse(nchar(release_date) == 4, paste0(release_dat
                      e, "-01-01"), release_date)) # Make release_date a real date column
        # Also would be cool to try to recognize gender by name and make a dummy column
```

This one is for networking later.

```
df_genre2 <- df_genre %>% mutate(genres = str_remove_all(genres, "\\["), # Get rid of br
                                                 ackets
                                          genres = str_remove_all(genres, "\\]"),
                              genres = str_split(genres, ",")) %>%  # Split up genres by comma
                                                 unnest(genres) %>%
                            mutate(genres = str_remove_all(genres, "\\ '")) %>% # remove all sp
                                                ace appostrophes
                                   mutate(genres = str_remove_all(genres, "\\'")) %>% #remove a
                              ll appostrophes, could be a better filter than this
                            mutate(genres = str_remove_all(genres, "\"")) # Some quotes stuck a
                                       round on childrens music category
  # df_genre3 <- df_genre2 %>% pivot_wider(names_from = artists, values_from = genres)
```

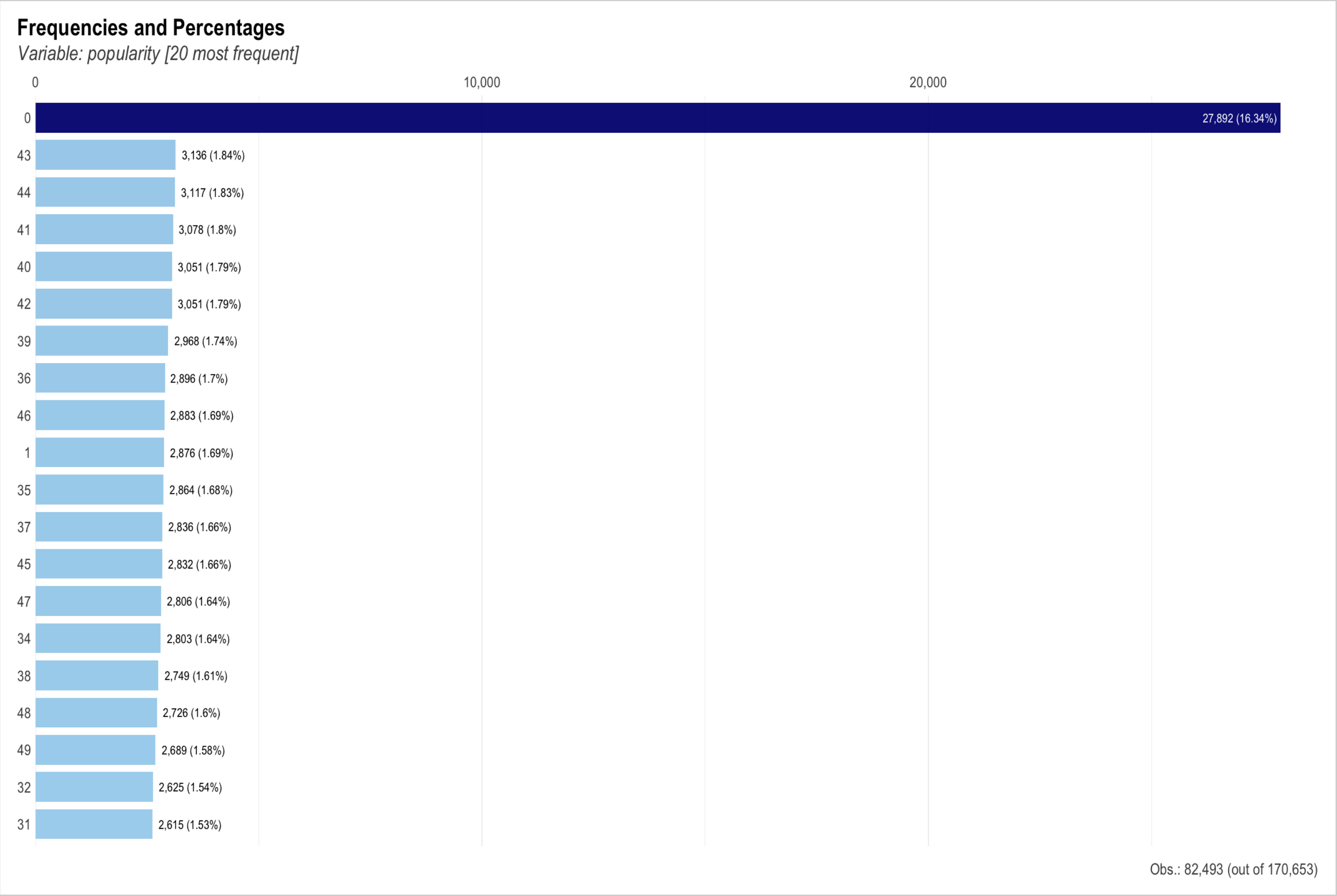
Looking At the Data Using Lares

```
df_str(df, return = "plot")
```



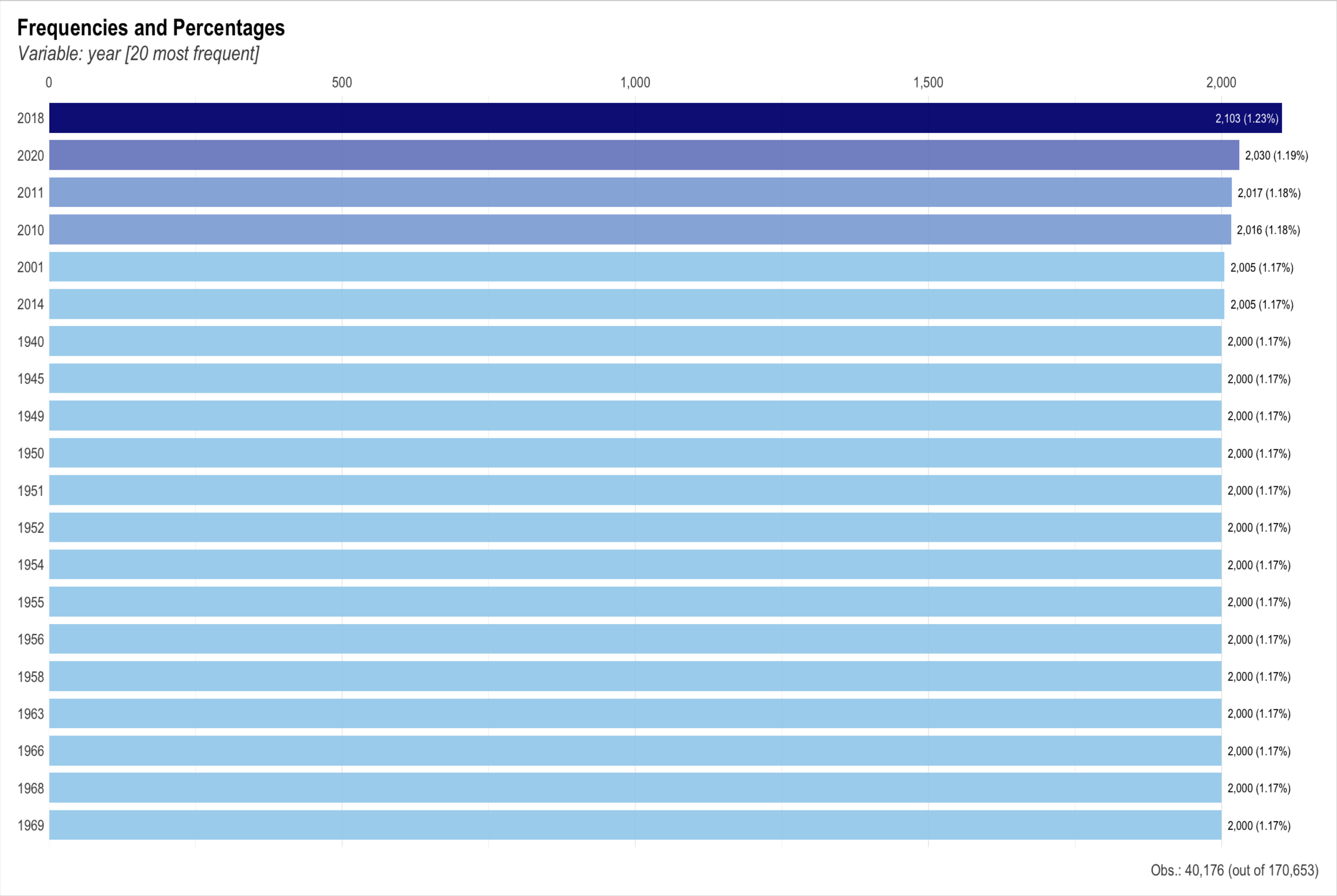
Look at percentages and cumulatives, looks like no popularity is very common!

```
df %>% freqs(popularity, plot = T)
```



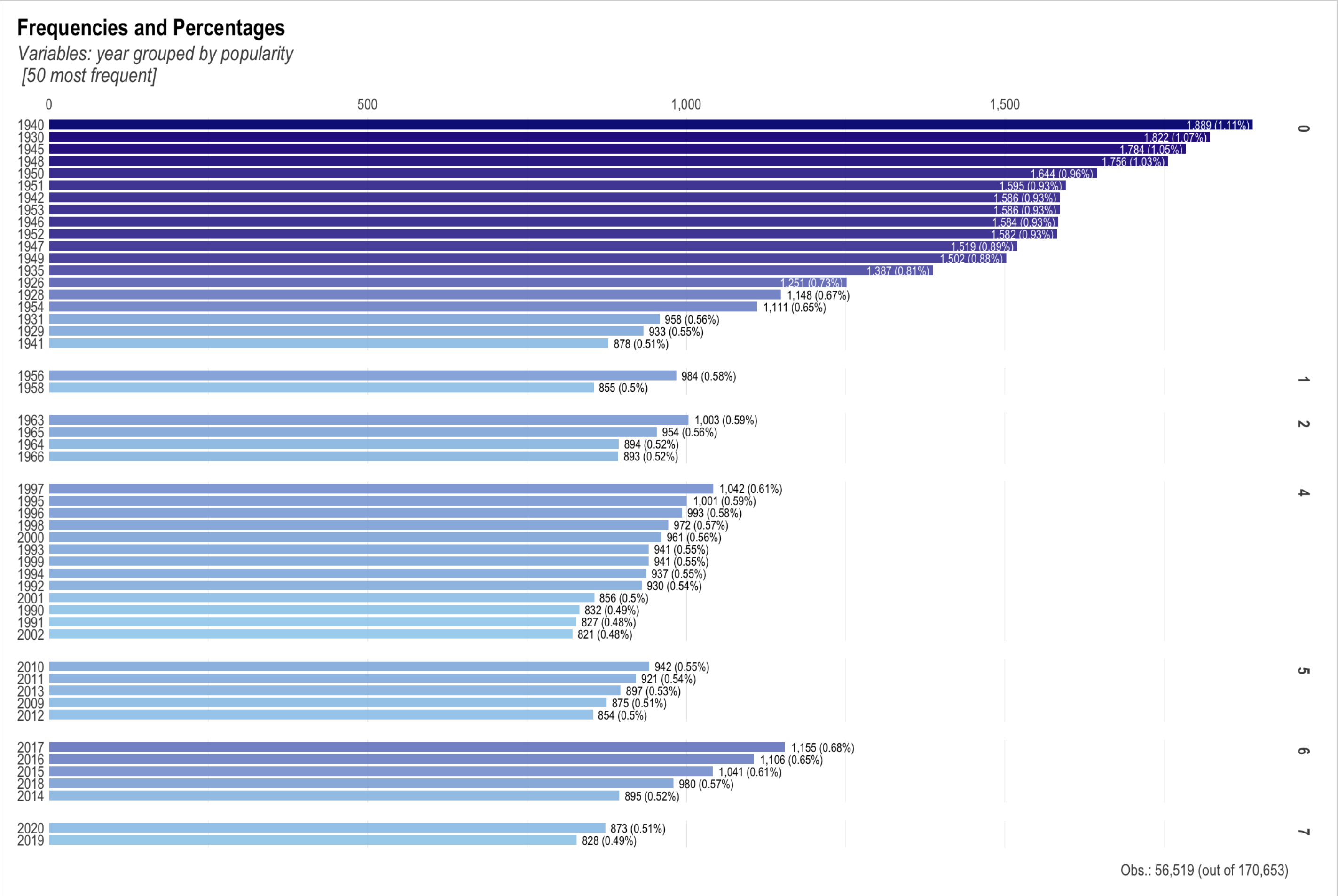
There is about the same amount of music every year in this dataframe

```
df %>% freqs(year, plot = T)
```



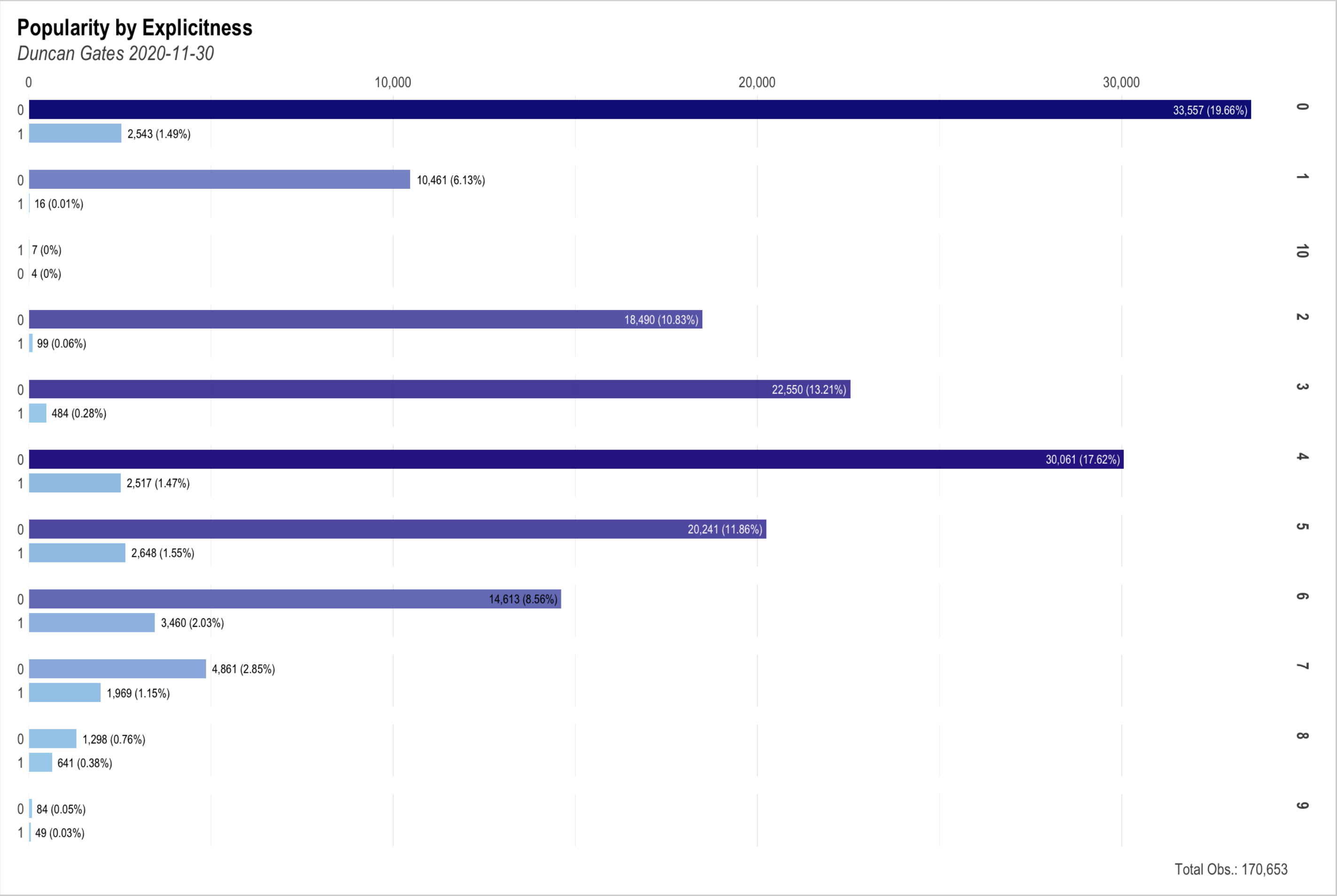
Basically newer stuff is more popular with little to no exceptions

```
df %>% mutate(popularity = round(popularity/10)) %>% freqs(popularity, year, plot = T, results = F, top = 50)
```



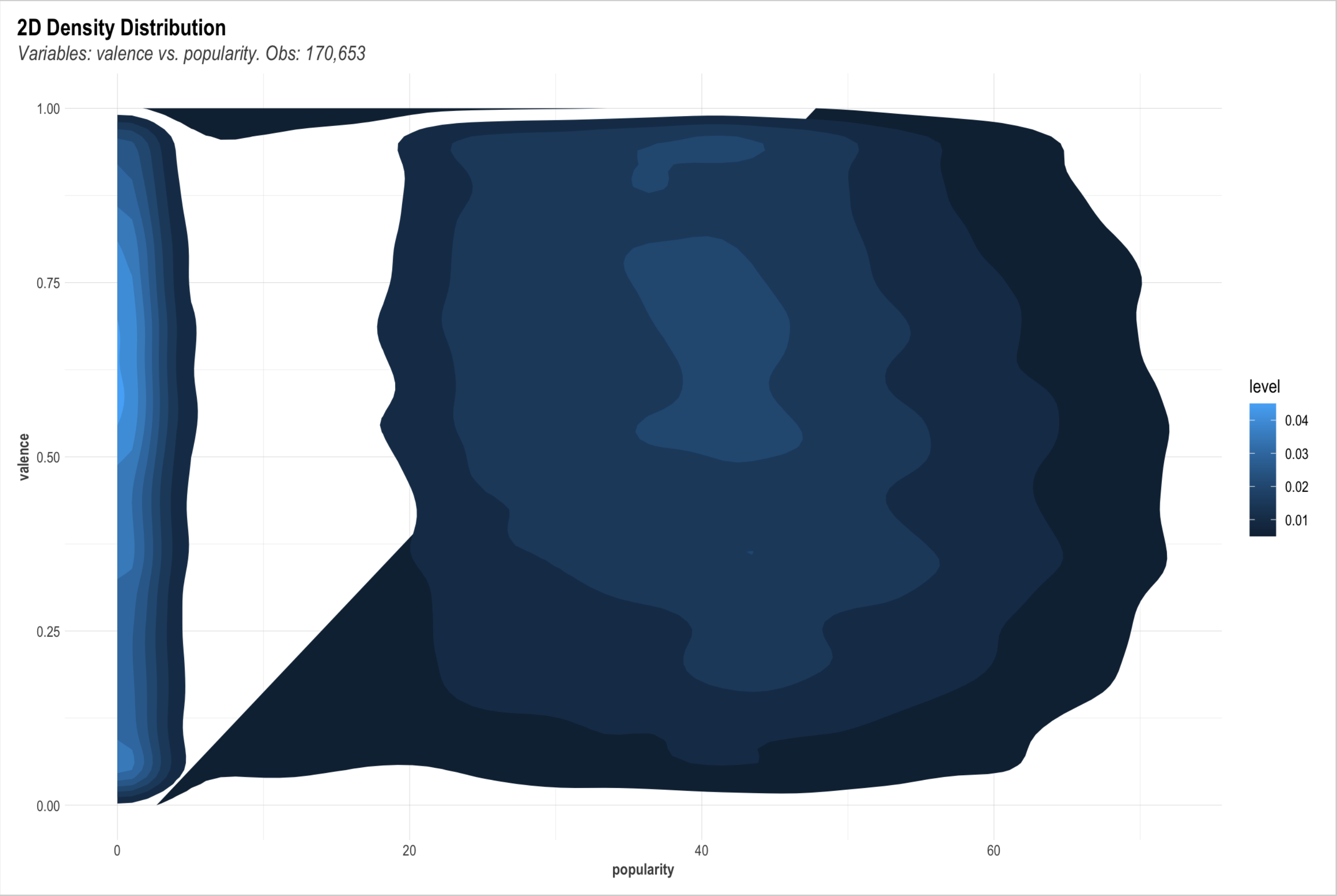
Looks like explicitness has a normal distribution compared to popularity

```
df %>%
mutate(explicit = as.factor(explicit),
   popularity = round(popularity/10)) %>%
freqs(popularity, explicit, plot = T,
   title = "Popularity by Explicitness",
subtitle = paste("Duncan Gates", Sys.Date()),
results = F)
```



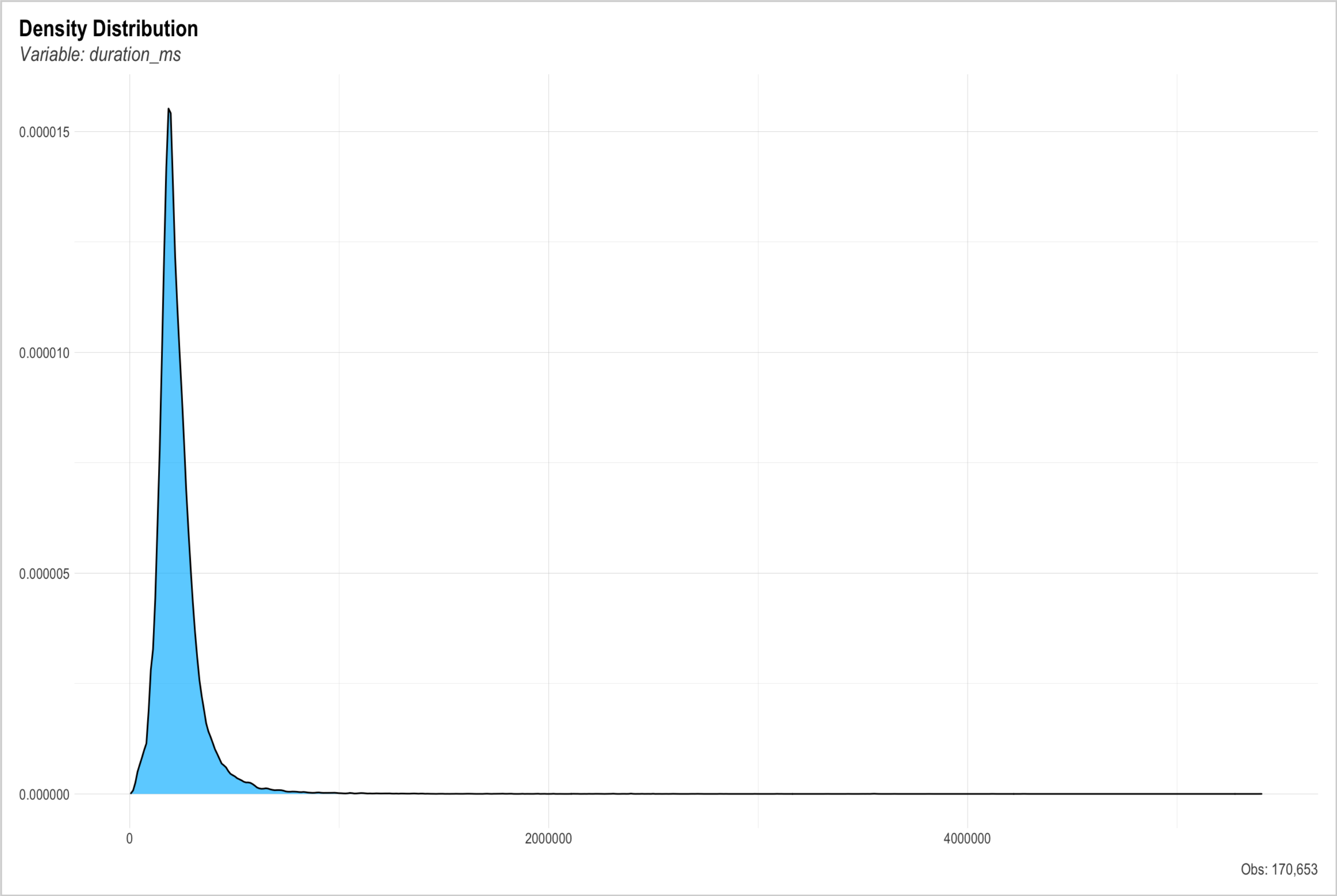
Now we check out the distribution, there’s some really cool stuff here

```
df %>% distr(popularity, valence) # Some really cool density plots can be done here
```



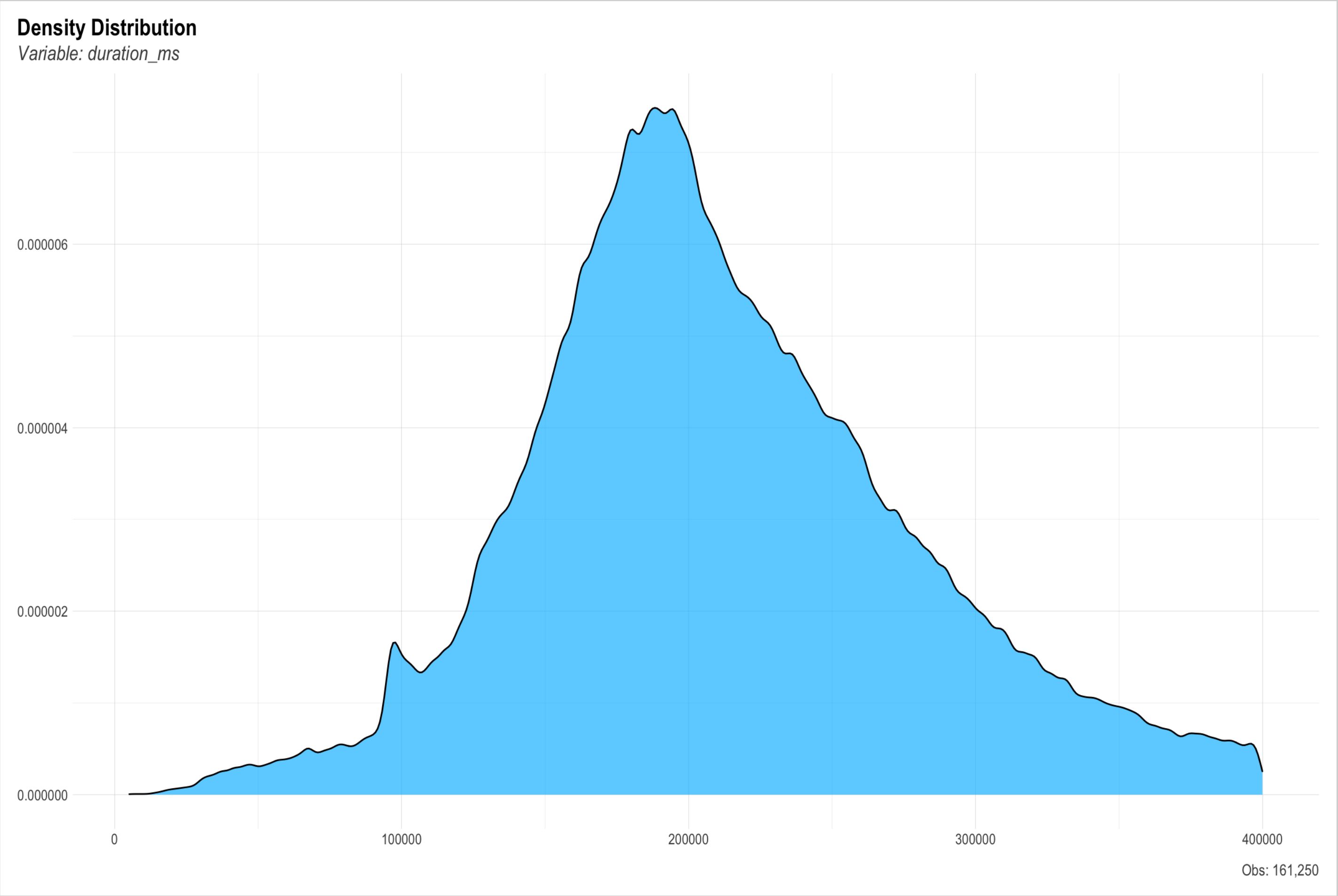
There’s also some really long songs out there…

```
df %>% distr(duration_ms)
```



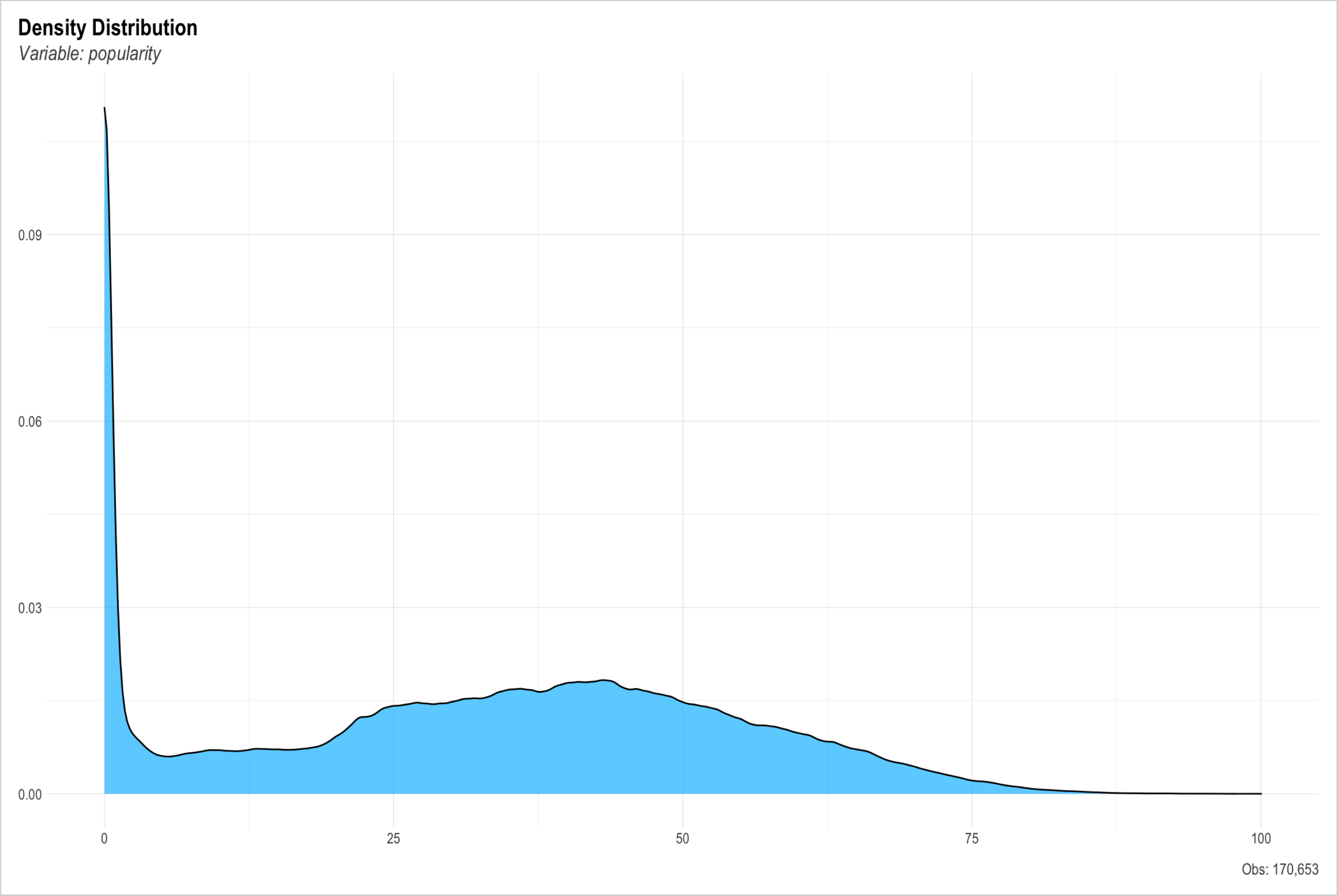
This looks more like the actual distribution

```
df %>%
filter(duration_ms < 400000) %>%
distr(duration_ms)
```



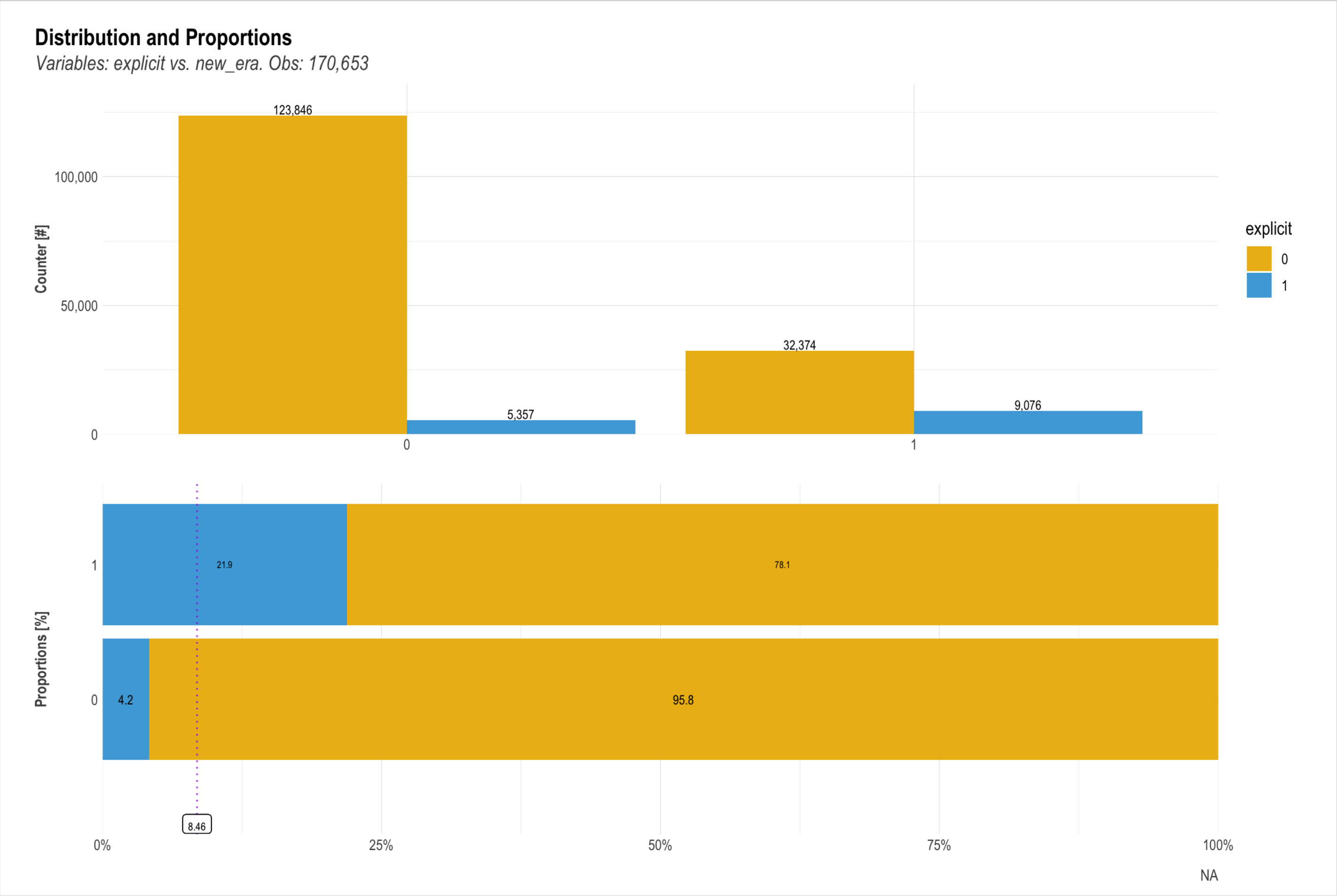
Very interesting distribution here

```
df %>% distr(popularity)
```

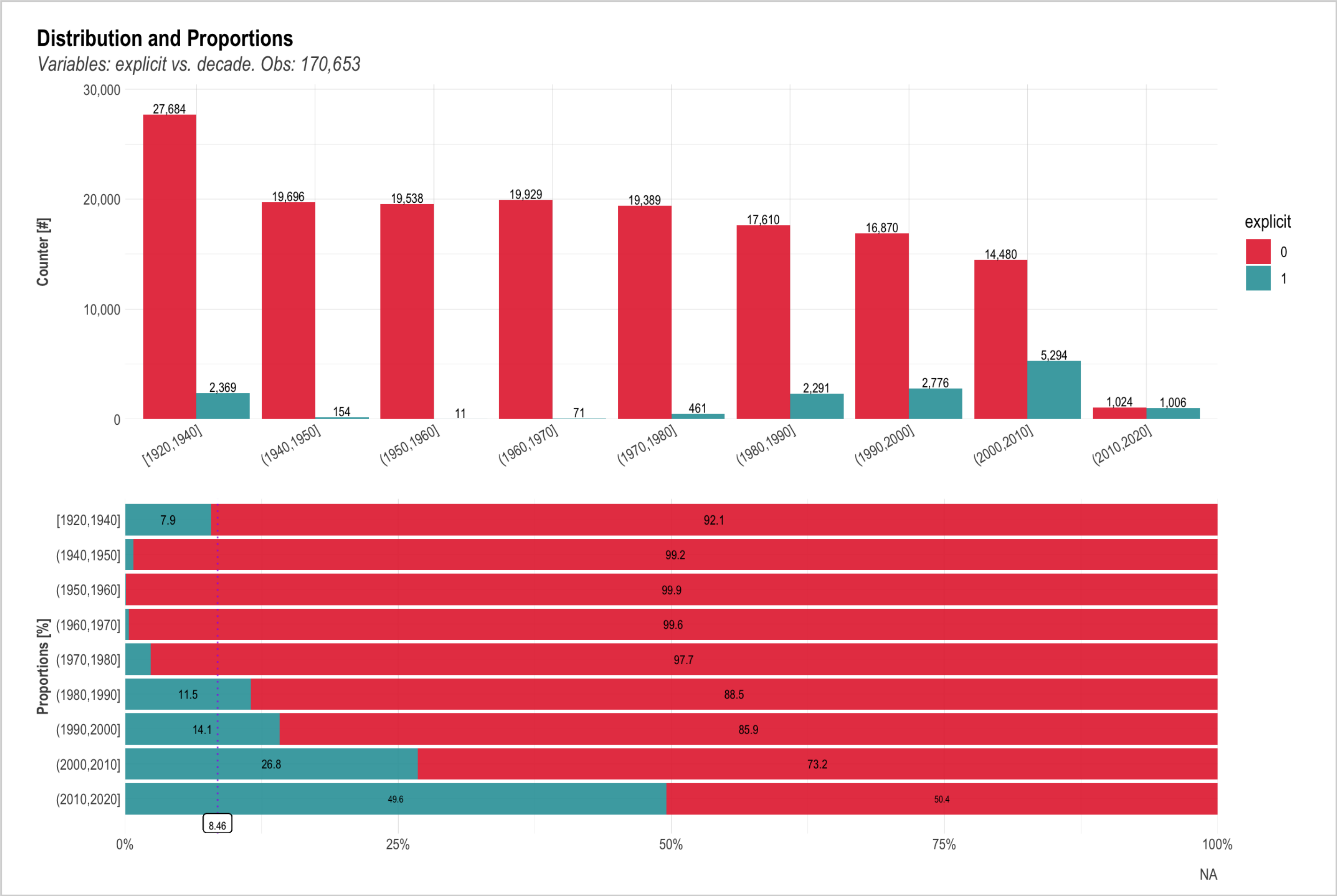


Looks like things are a lot more explicit in 2000-2020 as one might expect, would be interesting to see how when this starts, or what drives it. I also wonder what happened in 1920-1940?

```
            df %>%
  mutate(explicit = as.factor(explicit),
new_era = ifelse(year %in% c(2000:2020), 1, 0)) %>%
       distr(explicit, new_era)
```

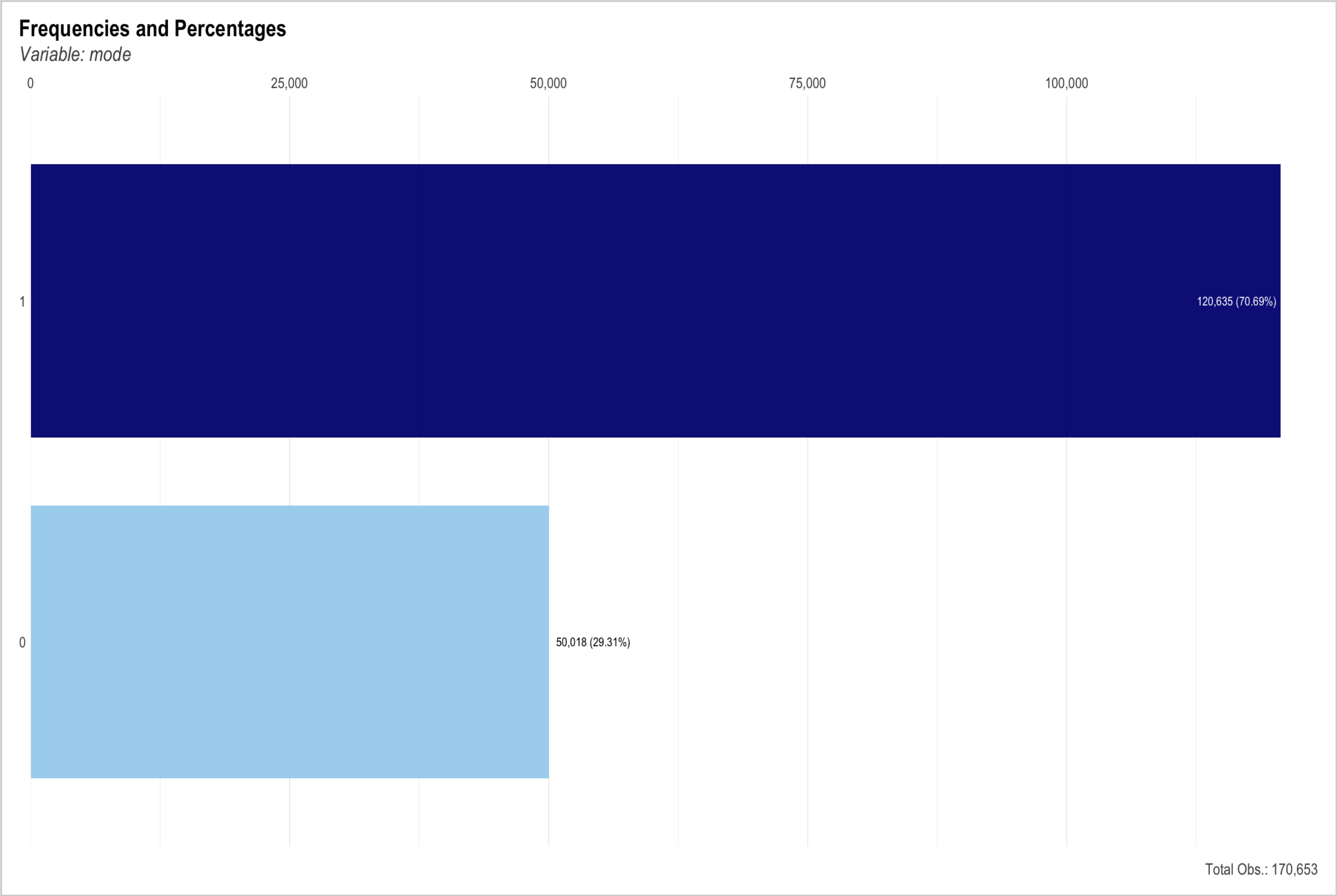


```
df %>%
mutate(decade = floor(year/10)*10) %>%
distr(explicit, decade, custom_colours = T, abc = T)
```



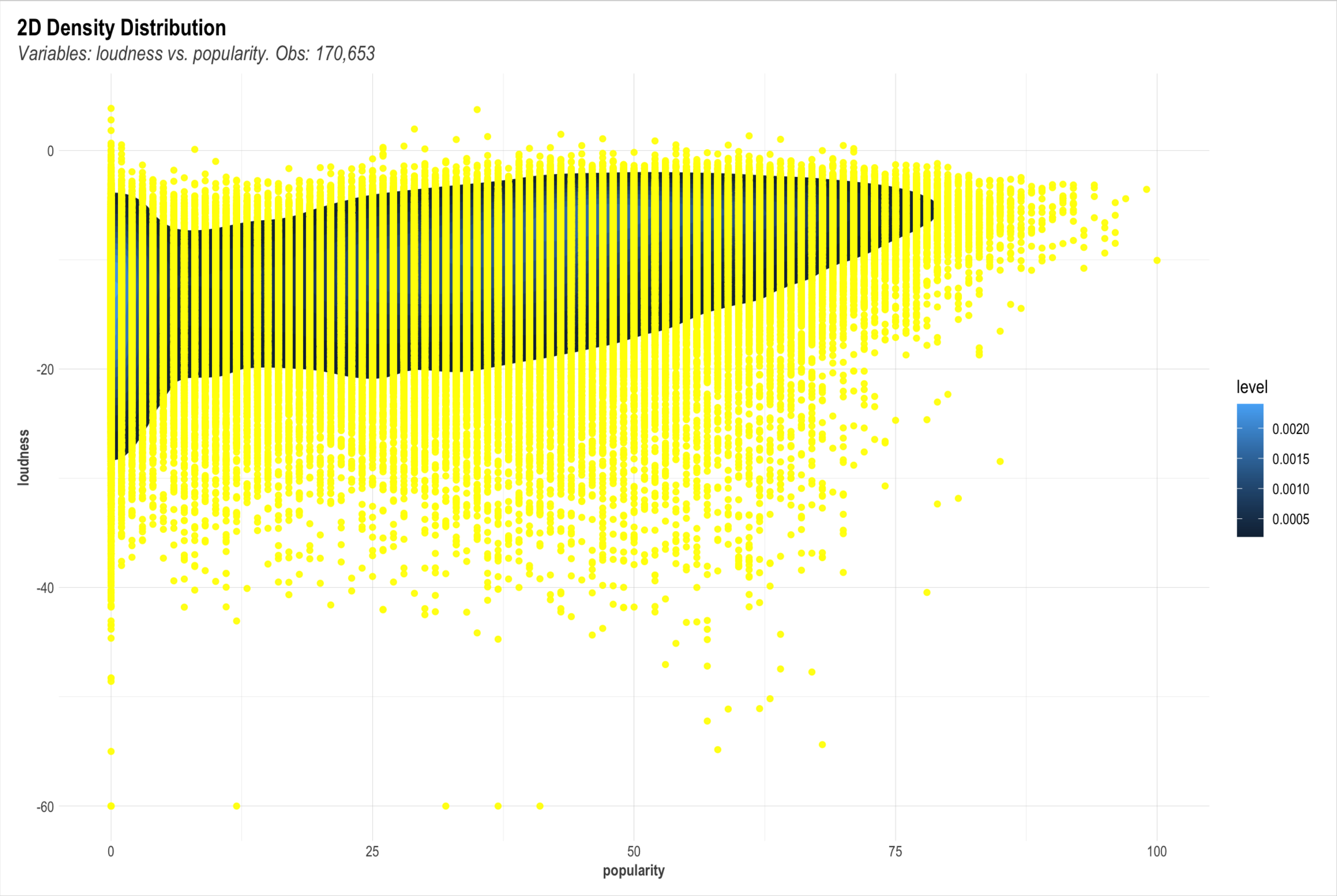
By the way mode is just whether the song is major or minor.

```
df %>% distr(mode, force = "char")
```



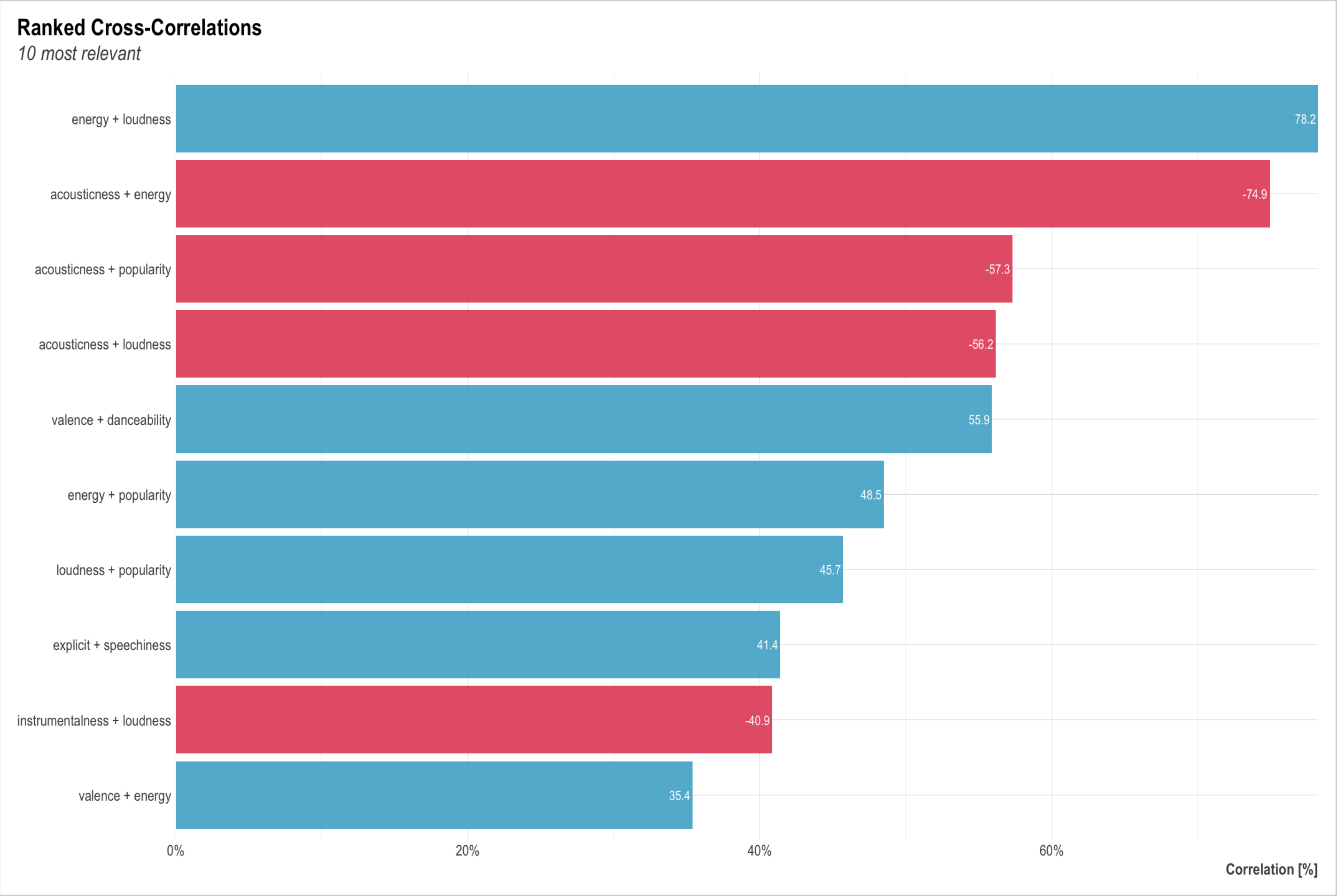
You can even use ggplot2!

```
df %>%
distr(popularity, loudness) + geom_point(color = "yellow")
```



Wouldn’t be data science without some random regressions, even more data science/machine learningy since the second one is a log odds table!

```
df %>%
select(-c(id, name, artists, year, release_date, key)) %>%
corr_cross(top = 10) # Look at top 10 correlations in the data, key messes with this i
dk why
```



```
table <- df %>%
select(-c(id, name, artists, year, release_date, key)) %>%
corr_var(popularity, logs = T, plot = F, top = 10)
table %>%
mutate(variables = paste(toupper(substr(variables, 1, 1)), substr(variables, 2, nchar(
as.character(variables))), sep = "")) %>%
mutate(corr = kableExtra::cell_spec(corr, "html", color = ifelse(corr > 0, "blue", "re
d")),
pvalue = kableExtra::cell_spec(pvalue, "html", color = ifelse(pvalue < 0.05, "g
reen", "red"))) %>%
kableExtra::kable(format = "html", escape = F) %>%
kableExtra::kable_styling("striped", full_width = F, position = "center")
```

variables	corr	pvalue
Popularity_log	0.890732	0
Acousticness	-0.573162	0
Acousticness_log	-0.55757	0
Energy_log	0.488822	0
Energy	0.485005	0
Loudness	0.457051	0
Instrumentalness_log	-0.300402	0

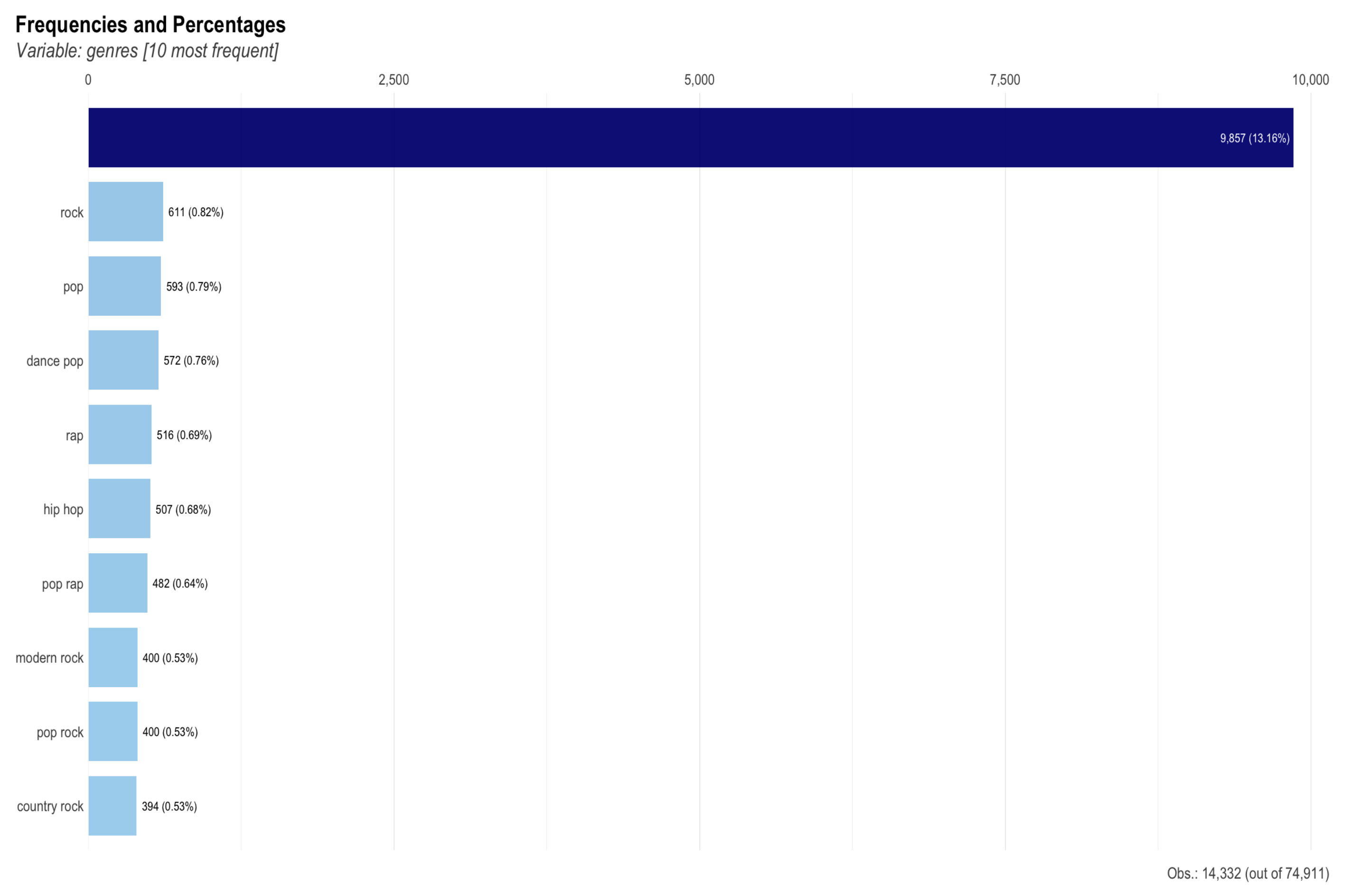
variables	corr	pvalue
Instrumentalness	-0.29675	0
Danceability	0.199606	0
Danceability_log	0.196287	0

```
# wow OHSE is pretty dope check this out with a better dataframe
```

Fun with reactable

Using lares one more time to get an idea of the data, there’s a lot of NA’s at first so I drop those and look again

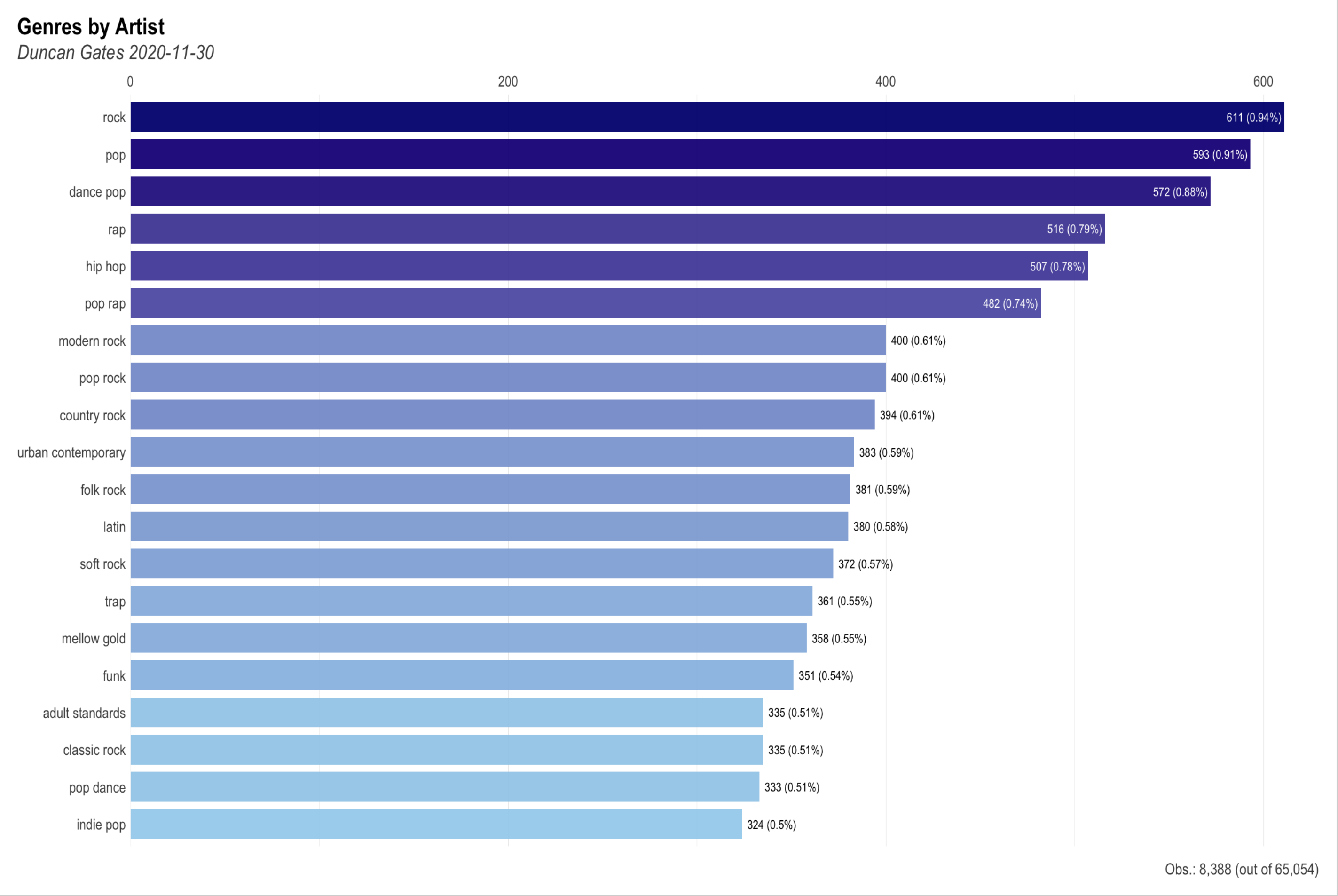
```
df_genre2 %>% distr(genres)
```



```
df_genre2 <- df_genre2 %>% na_if("") %>% na.omit %>%
                         mutate(genres = as.factor(genres))
```

What's it look like for genres now? Apparently in this dataframe there has been more rock than pop, not sure if that is actually the case (it does seem possible) or if its just the nature of this data.

```
df_genre2 %>% freqs(genres, plot = T,
         title = "Genres by Artist",
subtitle = paste("Duncan Gates", Sys.Date()),
                results = F)
```



```
df_genre2 %>%
  select(-count) %>%
  summer(genres) %>%
  mutate_if(is.numeric, funs(round)) %>%
  dplyr::rename(Count = n, `Duration` = duration_ms) %>%
  mutate(`Duration` = paste0(minute(seconds_to_period((`Duration`/(1000*Count)))),
                             ":",
                             dseconds(round(seconds_to_period((`Duration`/(1000*
Count))), digits = 2)))) %>% # Some disgusting lubridate here sorry
  rename_with(str_to_title) %>%
  mutate(Genres = str_to_title(Genres)) %>%
  mutate_at(vars(c("Acousticness", "Danceability", "Energy", "Instrumentalness", "Livene
ss", "Loudness", "Speechiness", "Tempo", "Valence", "Popularity", "Key", "Mode"
)), ~round((./Count), digits = 3)) %>%
  arrange(desc(Count)) %>%
  reactable(bordered = T,
            highlight = T,
            defaultColDef = colDef(align = "center",
                                   width = 150,
                                   footer = function(values = c("Count", "Acousticness",
"Danceability", "Energy", "Instrumentalness", "Liveness", "Loudness", "Speechin
ess", "Tempo", "Valence", "Popularity", "Key", "Mode")) {
                                     if (!is.numeric(values)) return()
                                     sparkline(values, type = "bar", width = 100, height = 30) # Can also do bo
xplots and line graphs
                                   }))
```

Genres	Count	Acousticness	Danceability	Duration
Rock	611	0.173	0.502	4:10.06s
Pop	593	0.258	0.631	3:37.06s
Dance Pop	572	0.182	0.652	3:49.23s
Rap	516	0.145	0.727	4:0.34s
Hip Hop	507	0.16	0.724	4:7.97s
Pop Rap	482	0.156	0.707	3:55.07s
Modern Rock	400	0.158	0.538	3:54.39s
Pop Rock	400	0.22	0.545	3:58.57s
Country Rock	394	0.36	0.553	3:59.78s
Urban Contemporary	383	0.245	0.661	4:26.81s

1–10 of 2978 rows Previous **1** 2 3 4 5 ... 298 Next

Let's filter down a bit and make things prettier

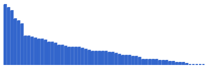
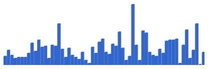
```r
make_color_pal <- function(colors, bias = 1) {
  get_color <- colorRamp(colors, bias = bias)
  function(x) rgb(get_color(x), maxColorValue = 255)
} # Make a color function
good_color <- make_color_pal(viridis::magma(n = 12), bias = 2)
# seq(0.1, 0.9, length.out = 12) %>%
#   good_color() %>%
#   scales::show_col() # This just shows the color palette generated

color_table <- df_genre2 %>%
  select(-count) %>%
  summer(genres) %>%
  filter(n > 200) %>% # Let's get the top 60 most popular genres
  mutate_if(is.numeric, funs(round)) %>%
  dplyr::rename(Count = n, `Duration` = duration_ms) %>%
  mutate(`Duration` = paste0(minute(seconds_to_period((`Duration`/(1000*Count)))),
                             ":",
                             dseconds(round(seconds_to_period((`Duration`/(1000*
    Count))), digits = 2)))) %>% # Some disgusting lubridate here sorry
  rename_with(str_to_title) %>%
  mutate(Genres = str_to_title(Genres)) %>%
  mutate_at(vars(c("Acousticness", "Danceability", "Energy", "Instrumentalness", "Livene
    ss", "Loudness", "Speechiness", "Tempo", "Valence", "Popularity", "Key", "Mode"
    )), ~round((./Count), digits = 3)) %>%
  arrange(desc(Count))
```

```
## Grouped by: 'genres'
```

```
## Joining, by = "genres"
```

```
color_table %>%
reactable(bordered = T,
          highlight = T,
          columns = list(
            Count = colDef(
              name = "Count",
              style = function(value) {
                value
normalized <- (value - min(color_table$Count)) / (max(color_table$Coun
t) - min(color_table$Count))
                color <- rev(good_color(normalized))
                list(background = color)
              }
            )
          ),
          defaultColDef = colDef(align = "center",
                                 width = 150,
                                 footer = function(values = c("Count", "Acousticness",
"Danceability", "Energy", "Instrumentalness", "Liveness", "Loudness", "Speechin
ess", "Tempo", "Valence", "Popularity", "Key", "Mode")) {
                                   if (!is.numeric(values)) return()
sparkline(values, type = "bar", width = 100, height = 30) # Can also do bo
xplots and line graphs
                                 }))
```

Genres	Count	Acousticness	Danceability	Duration
Rock	611	0.173	0.502	4:10.06s
Pop	593	0.258	0.631	3:37.06s
Dance Pop	572	0.182	0.652	3:49.23s
Rap	516	0.145	0.727	4:0.34s
Hip Hop	507	0.16	0.724	4:7.97s
Pop Rap	482	0.156	0.707	3:55.07s
Modern Rock	400	0.158	0.538	3:54.39s
Pop Rock	400	0.22	0.545	3:58.57s
Country Rock	394	0.36	0.553	3:59.78s
Urban Contemporary	383	0.245	0.661	4:26.81s
				

1–10 of 60 rows

Previous **1** 2 3 4 5 6 Next

Machine Learning???

Let’s load some networking libraries

```
library(ggraph)
library(igraph)
```

Now let’s make some central nodes for our network.

```
network_df <- df_genre2 %>% select(genres, artists, popularity)
network_graph <- network_df %>%
graph_from_data_frame() # From igraph

a <- grid::arrow(type = "closed", length = unit(.10, "inches"))
# ggraph(network_graph, layout = "fr") +
#   geom_edge_link() +
#   geom_node_point(color = "lightblue", size = 5) +
#   geom_node_text(aes(label = name), vjust = 1, hjust = 1, repel = TRUE) +
#   ggtitle("Bi-Gram Network for All Songs") +
#   theme_void()
```