MATHS 7107 Data Taming Assignment Final Report

Chang Dong

2023-04-24

Appendix

```r
pacman::p_load(skimr, tidyverse, tidymodels, themis, recipes, dials,kknn, vip,forcats,caret,MASS,discrim
```

1.1 Data import

```r
spotify_songs_origin <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/m
```

```
## Rows: 32833 Columns: 23
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (10): track_id, track_name, track_artist, track_album_id, track_album_na...
## dbl (13): track_popularity, danceability, energy, key, loudness, mode, speec...
##
## 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
skim(spotify_songs_origin)
```

Table 1: Data summary

Name	spotify_songs_origin
Number of rows	32833
Number of columns	23
Column type frequency:	
character	10
numeric	13
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
track_id	0	1	22	22	0	28356	0
track_name	5	1	1	144	0	23449	0
track_artist	5	1	2	69	0	10692	0
track_album_id	0	1	22	22	0	22545	0
track_album_name	5	1	1	151	0	19743	0
track_album_release_date	0	1	4	10	0	4530	0

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
playlist_name	0	1	6	120	0	449	0
playlist_id	0	1	22	22	0	471	0
playlist_genre	0	1	3	5	0	6	0
playlist_subgenre	0	1	4	25	0	24	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
track_popularity	0	1	42.48	24.98	0.00	24.00	45.00	62.00	100.00	
danceability	0	1	0.65	0.15	0.00	0.56	0.67	0.76	0.98	
energy	0	1	0.70	0.18	0.00	0.58	0.72	0.84	1.00	
key	0	1	5.37	3.61	0.00	2.00	6.00	9.00	11.00	
loudness	0	1	-6.72	2.99	-46.45	-8.17	-6.17	-4.64	1.27	
mode	0	1	0.57	0.50	0.00	0.00	1.00	1.00	1.00	
speechiness	0	1	0.11	0.10	0.00	0.04	0.06	0.13	0.92	
acousticness	0	1	0.18	0.22	0.00	0.02	0.08	0.26	0.99	
instrumentalness	0	1	0.08	0.22	0.00	0.00	0.00	0.00	0.99	
liveness	0	1	0.19	0.15	0.00	0.09	0.13	0.25	1.00	
valence	0	1	0.51	0.23	0.00	0.33	0.51	0.69	0.99	
tempo	0	1	120.88	26.90	0.00	99.96	121.98	133.92	239.44	
duration_ms	0	1	225799.81	59834.01	4000.00	187819.00	216000.00	253585.00	517810.00	

1.2 Data Cleanning Method

These have n_unique more than 1000 should be considered as Text instead of Categorical data (“track_id”,“track_name”, “track_artist”, “track_album_id”, “track_album_name”), , “track_album_release_date” is not included, because we know it is a time series should be numerical.

```
text_variables <- c("track_id", "track_name", "track_artist", "track_album_id", "track_album_name")

spotify_songs <- spotify_songs_origin %>%
  dplyr::select(-dplyr::any_of(text_variables))
```

Categorical Data (factor)

```
categorical_variables <- c("playlist_name", "playlist_id", "playlist_genre", "playlist_subgenre")

spotify_songs <- spotify_songs %>%
  mutate(across(all_of(categorical_variables), as.factor))
```

Numerical Data (numerical)

```
numerical_variables <- c("track_popularity","danceability", "energy", "key",
                         "loudness","mode", "speechiness","acousticness",
                         "instrumentalness", "liveness", "valence",
                         "tempo", "duration_ms")

spotify_songs <- spotify_songs %>%
  mutate(across(all_of(numerical_variables), as.numeric))

spotify_songs <- spotify_songs %>%
```

```
  mutate(track_album_release_year = as.numeric(format(as.Date(track_album_release_date, format = "%Y-%m-
  dplyr::select(-track_album_release_date)

numerical_variables <- c(numerical_variables, "track_album_release_year")
```

```
skim(spotify_songs)
```

Table 4: Data summary

Name	spotify_songs
Number of rows	32833
Number of columns	18
Column type frequency:	
factor	4
numeric	14
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
playlist_name	0	1	FALSE	449	Ind: 308, 202: 247, Per: 244, Har: 219
playlist_id	0	1	FALSE	471	4Jk: 247, 37i: 198, 6Kn: 195, 3xM: 189
playlist_genre	0	1	FALSE	6	edm: 6043, rap: 5746, pop: 5507, r&b: 5431
playlist_subgenre	0	1	FALSE	24	pro: 1809, sou: 1675, ind: 1672, lat: 1656

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
track_popularity	0	1.00	42.48	24.98	0.00	24.00	45.00	62.00	100.00	
danceability	0	1.00	0.65	0.15	0.00	0.56	0.67	0.76	0.98	
energy	0	1.00	0.70	0.18	0.00	0.58	0.72	0.84	1.00	
key	0	1.00	5.37	3.61	0.00	2.00	6.00	9.00	11.00	
loudness	0	1.00	-6.72	2.99	-46.45	-8.17	-6.17	-4.64	1.27	
mode	0	1.00	0.57	0.50	0.00	0.00	1.00	1.00	1.00	
speechiness	0	1.00	0.11	0.10	0.00	0.04	0.06	0.13	0.92	
acousticness	0	1.00	0.18	0.22	0.00	0.02	0.08	0.26	0.99	
instrumentalness	0	1.00	0.08	0.22	0.00	0.00	0.00	0.00	0.99	
liveness	0	1.00	0.19	0.15	0.00	0.09	0.13	0.25	1.00	
valence	0	1.00	0.51	0.23	0.00	0.33	0.51	0.69	0.99	
tempo	0	1.00	120.88	26.90	0.00	99.96	121.98	133.92	239.44	
duration_ms	0	1.00	225799.81	59834.01	4000.00	187819.00	216000.00	253585.00	517810.00	
track_album_release_year	1886	0.94	2012.20	10.40	1957.00	2010.00	2017.00	2019.00	2020.00	

```r
spotify_songs <- spotify_songs %>%
  drop_na()

skim(spotify_songs)
```

Table 7: Data summary

Name	spotify_songs
Number of rows	30947
Number of columns	18
Column type frequency:	
factor	4
numeric	14
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
playlist_name	0	1	FALSE	449	Ind: 299, 202: 247, Per: 212, Har: 211
playlist_id	0	1	FALSE	471	4Jk: 247, 6Kn: 195, 37i: 190, 3xM: 189
playlist_genre	0	1	FALSE	6	edm: 5969, rap: 5471, pop: 5303, r&b: 5094
playlist_subgenre	0	1	FALSE	24	pro: 1760, ind: 1647, lat: 1573, neo: 1547

Variable type: numeric

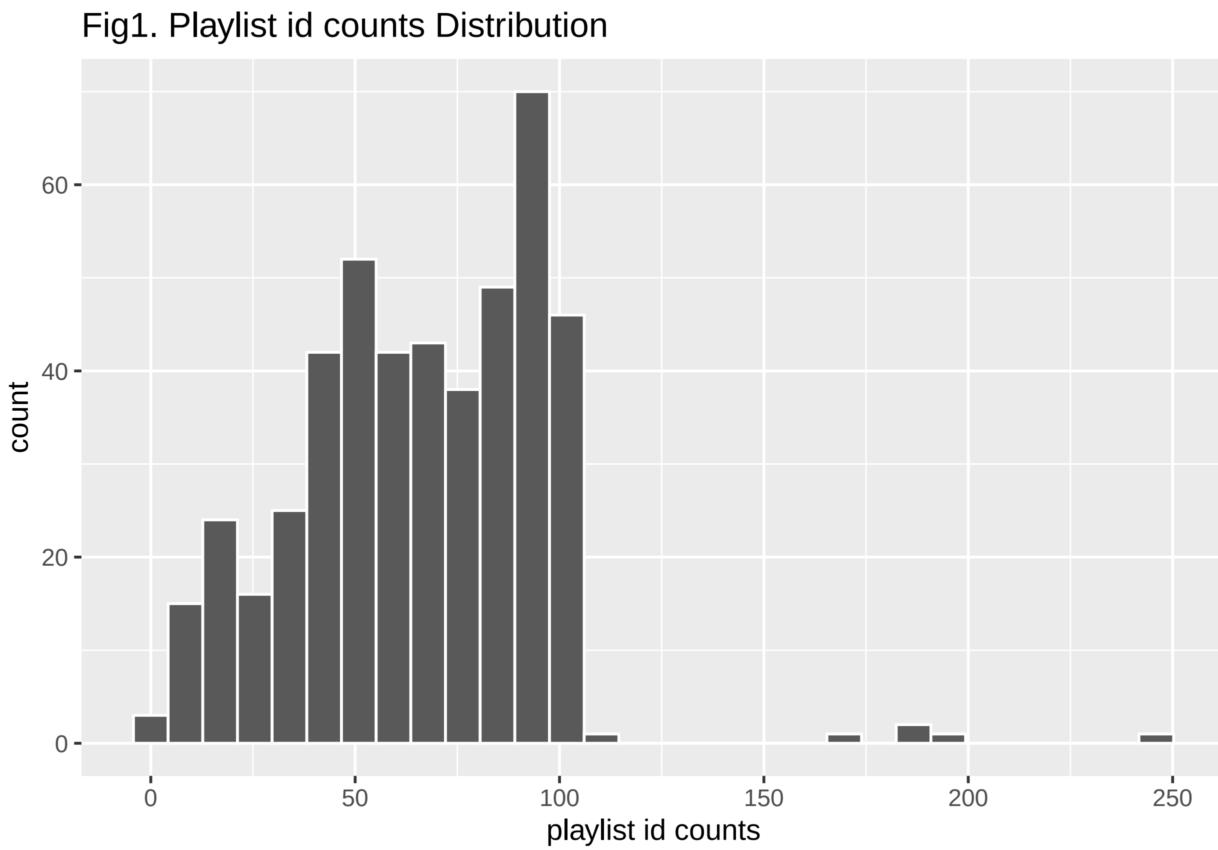
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
track_popularity	0	1	42.75	24.96	0.00	25.00	45.00	62.00	100.00	
danceability	0	1	0.66	0.14	0.00	0.57	0.67	0.76	0.98	
energy	0	1	0.70	0.18	0.00	0.58	0.72	0.84	1.00	
key	0	1	5.37	3.61	0.00	2.00	6.00	9.00	11.00	
loudness	0	1	-6.64	2.95	-46.45	-8.07	-6.09	-4.61	1.27	
mode	0	1	0.56	0.50	0.00	0.00	1.00	1.00	1.00	
speechiness	0	1	0.11	0.10	0.00	0.04	0.06	0.13	0.92	
acousticness	0	1	0.18	0.22	0.00	0.02	0.08	0.26	0.99	
instrumentalness	0	1	0.09	0.23	0.00	0.00	0.00	0.01	0.99	
liveness	0	1	0.19	0.15	0.00	0.09	0.13	0.25	1.00	
valence	0	1	0.51	0.23	0.00	0.33	0.51	0.69	0.99	
tempo	0	1	120.94	26.85	0.00	99.97	122.00	133.52	239.44	
duration_ms	0	1	223950.10	59113.89	4000.00	186750.00	214400.00	251133.00	517810.00	
track_album_release_year	0	1	2012.20	10.40	1957.00	2010.00	2017.00	2019.00	2020.00	

1.3 Exploratory Data Analysis(EDA) Method

Categorical variable

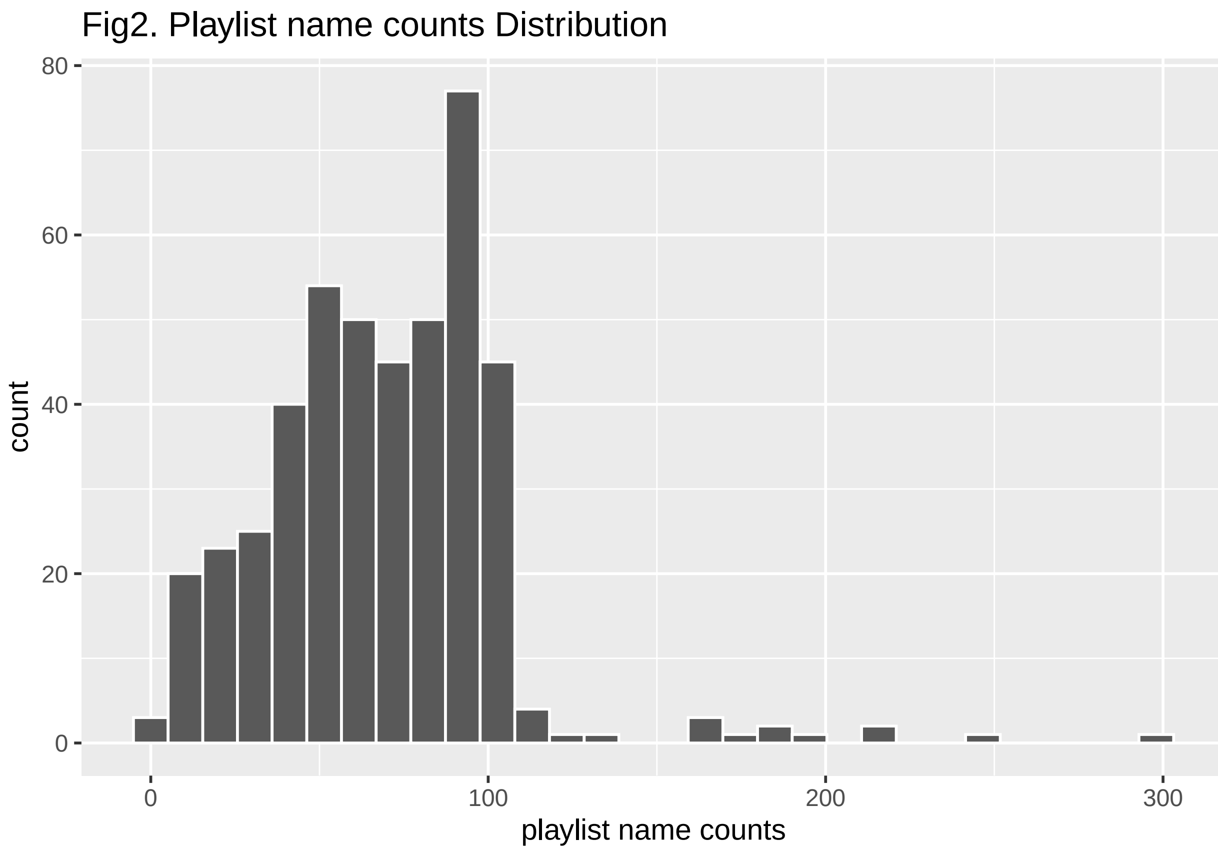
playlist_id

```
playlist_id_counts <- spotify_songs %>%
  count(playlist_id) %>%
  arrange(desc(n))
playlist_id_counts %>% ggplot(aes(x= n)) + geom_histogram(bins = 30, color = "white") +
  ggtitle("Fig1. Playlist id counts Distribution") +xlab("playlist id counts") +ylab("count")
```



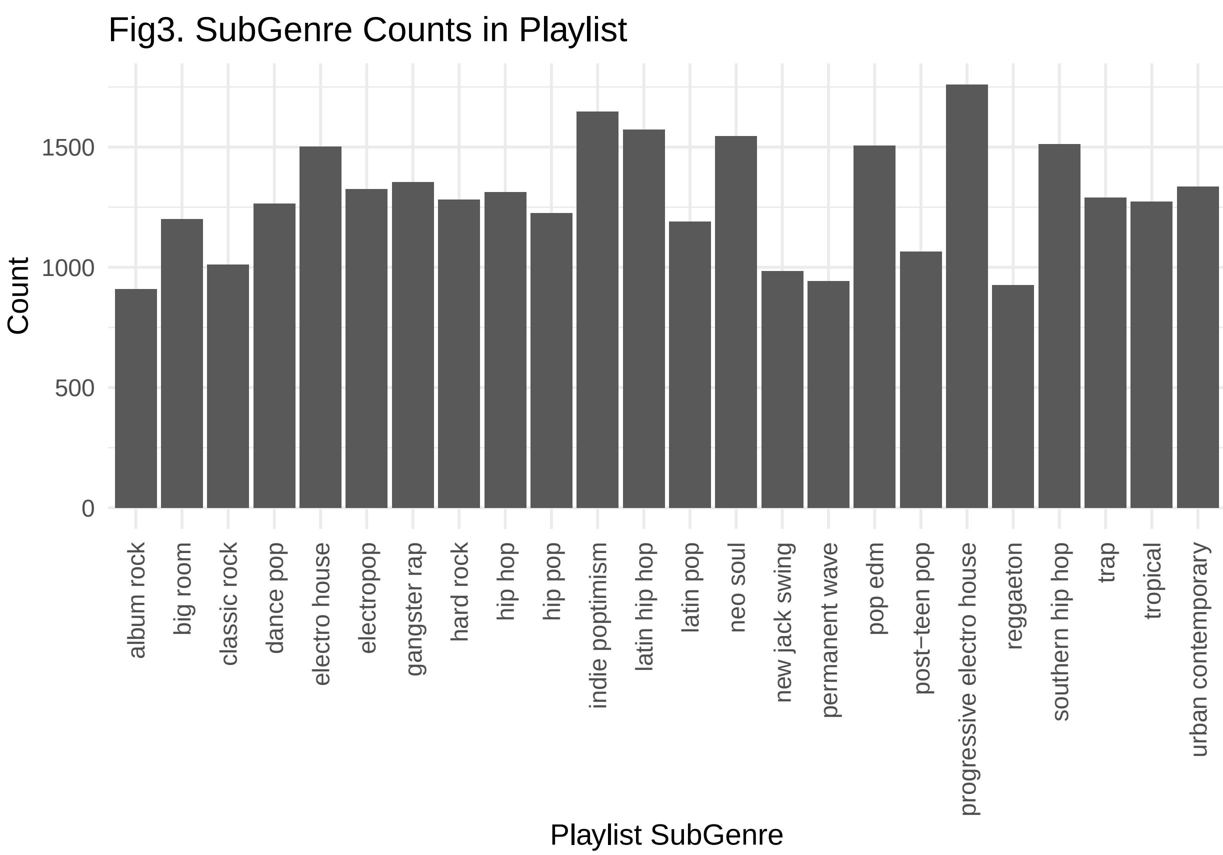
playlist_name

```
playlist_name_counts <- spotify_songs %>%
  count(playlist_name) %>%
  arrange(desc(n))
playlist_name_counts %>% ggplot(aes(x= n)) + geom_histogram(bins = 30, color = "white") +
  ggtitle("Fig2. Playlist name counts Distribution") +xlab("playlist name counts") +ylab("count")
```



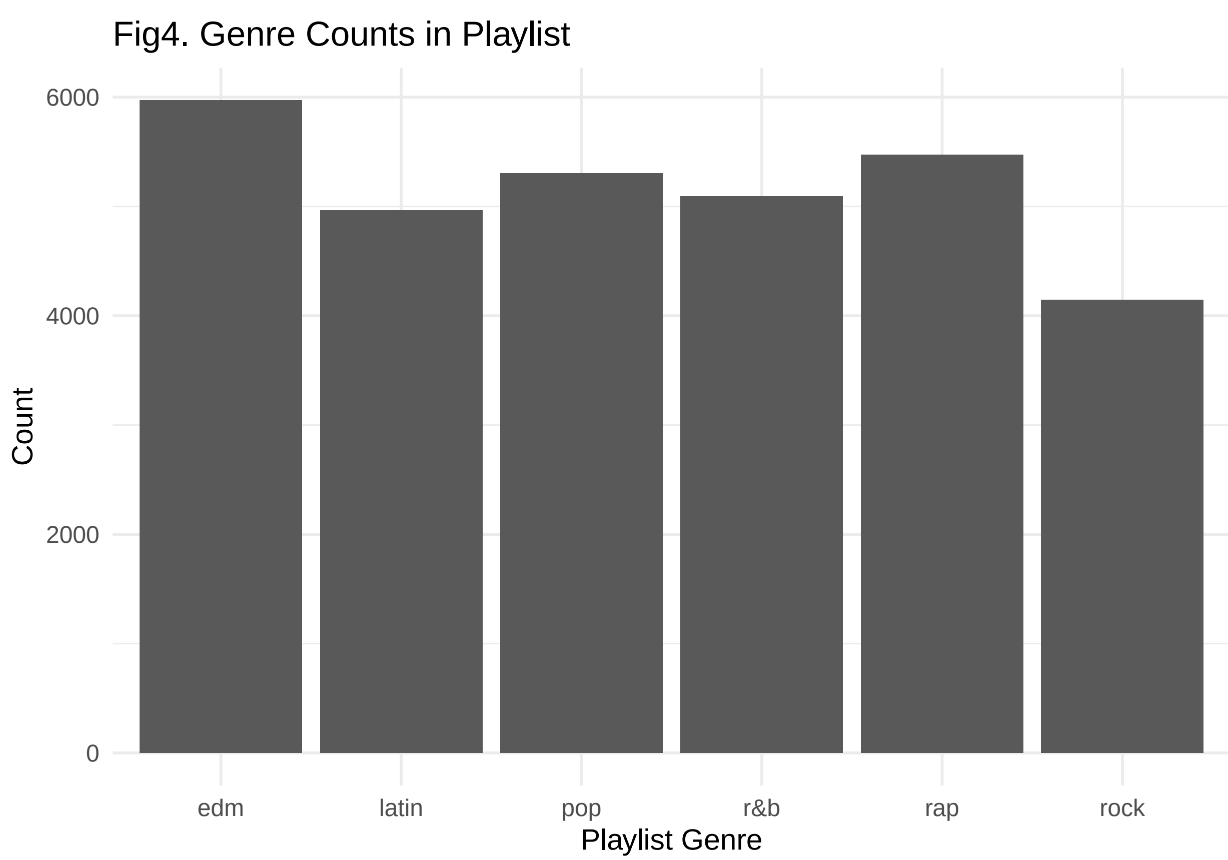
playlist_subgenre

```r
playlist_subgenre_counts <- spotify_songs %>%
  count(playlist_subgenre) %>%
  arrange(desc(n))

playlist_subgenre_counts %>%
  ggplot(aes(x = playlist_subgenre, y = n)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1)) +
  labs(title = "Fig3. SubGenre Counts in Playlist",
       x = "Playlist SubGenre",
       y = "Count")
```



playlist_genre (Response variable)

```
playlist_genre_counts <- spotify_songs %>%
  count(playlist_genre) %>%
  arrange(desc(n))

playlist_genre_counts %>%
  ggplot(aes(x = playlist_genre, y = n)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(title = "Fig4. Genre Counts in Playlist",
       x = "Playlist Genre",
       y = "Count")
```




```r
chisq_test <- function(variable_name) {
  contingency_table <- spotify_songs %>%
    count(!!sym(variable_name), playlist_genre) %>%
    spread(key = playlist_genre, value = n, fill = 0) %>%
    column_to_rownames(var= variable_name)

  chi_square_test <- chisq.test(contingency_table)
  cat(variable_name)
  print(chi_square_test)
}

for (variable in setdiff(categorical_variables, "playlist_genre")) {
  chisq_test(variable)
}
```

Chi^2-Test (categorical data versus categorical data)

```
## Warning in chisq.test(contingency_table): Chi-squared approximation may be
## incorrect

## playlist_name
##  Pearson's Chi-squared test
##
## data:  contingency_table
## X-squared = 152976, df = 2240, p-value < 2.2e-16

## Warning in chisq.test(contingency_table): Chi-squared approximation may be
```

```
## incorrect
```

```
## playlist_id
##  Pearson's Chi-squared test
##
## data:  contingency_table
## X-squared = 152976, df = 2350, p-value < 2.2e-16
##
## playlist_subgenre
##  Pearson's Chi-squared test
##
## data:  contingency_table
## X-squared = 154735, df = 115, p-value < 2.2e-16
```

```r
spotify_songs <- spotify_songs %>%
  dplyr::select(-setdiff(categorical_variables, "playlist_genre"))
```

Numerical variable

Compile a standard function for EDA

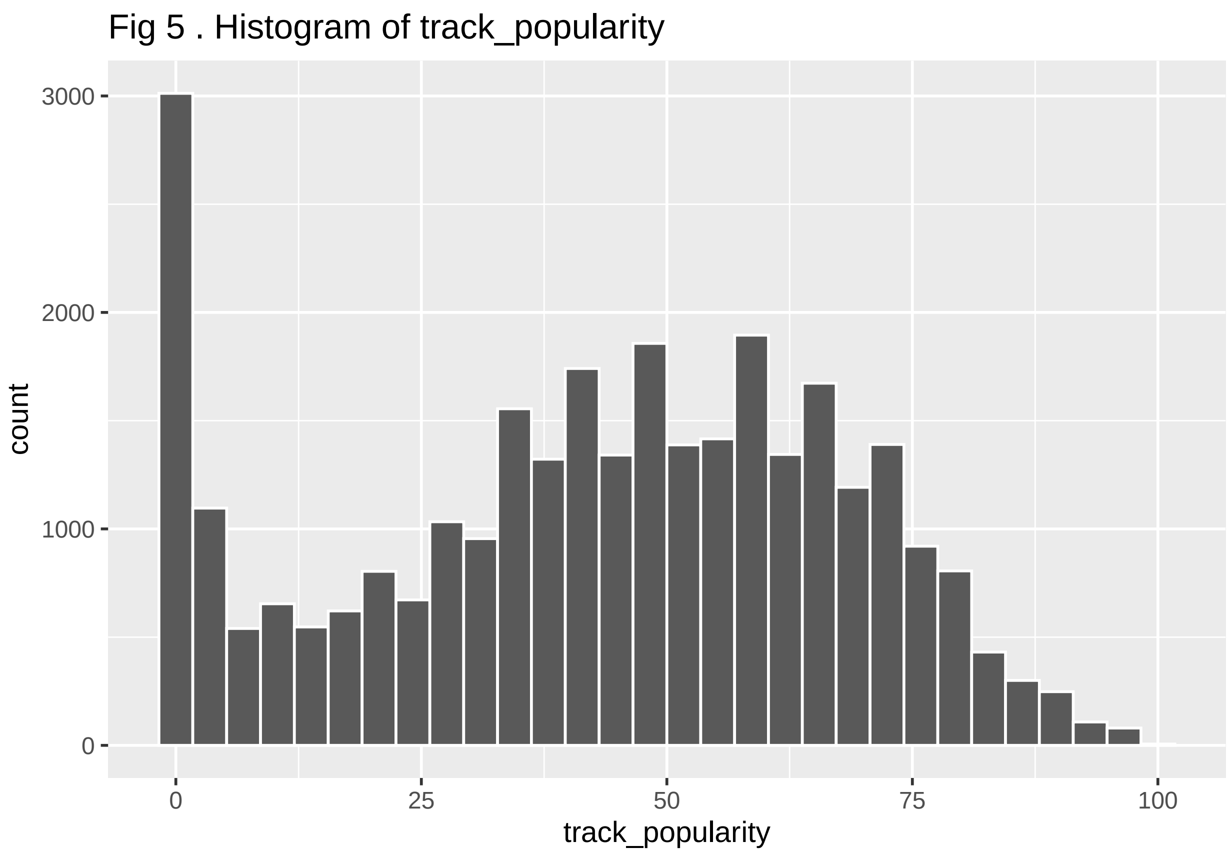
- plot a histogram to check the distribution of each numerical variable
- plot a box plot to check the distribution of each numerical variable in different genres
- Anova test to check the distribution differences in different genres

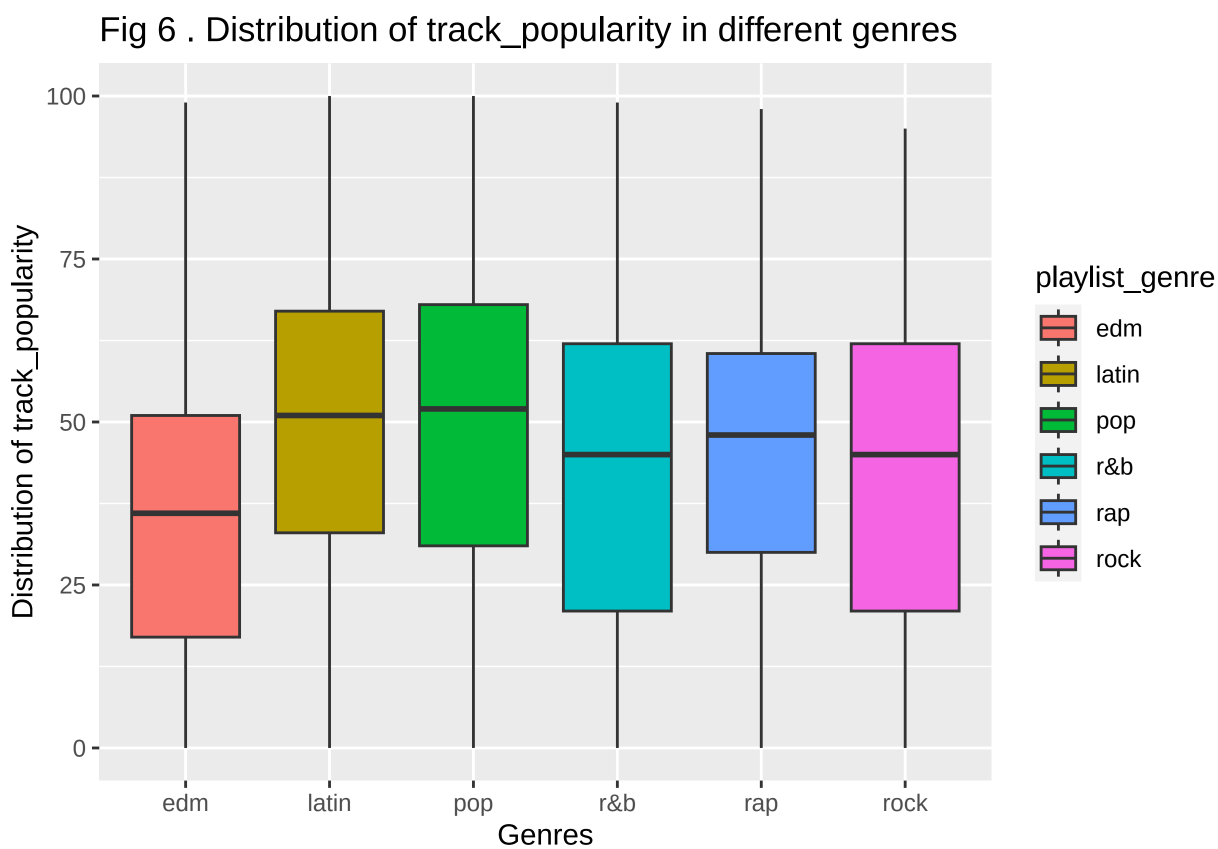
```r
analyze_variable_hist <- function(variable_name) {

  # histogram
  idx = 4 + 2*which(numerical_variables == variable_name)-1
  histo_gram <- spotify_songs %>% ggplot(aes_string(variable_name)) +
    geom_histogram(bins = 30,color="white") +
    ggtitle(paste("Fig",idx,". Histogram of", variable_name)) +
    xlab(variable_name) + ylab("count")
  print(histo_gram)

}

analyze_variable_box <- function(variable_name) {
# boxplot
  idx = 4 + 2*which(numerical_variables == variable_name)
  box_plot <- spotify_songs %>% ggplot(aes_string(x = "playlist_genre",
                                                  y = variable_name,
                                                  fill = "playlist_genre")) +
    geom_boxplot() + ggtitle(paste("Fig",idx,". Distribution of", variable_name, "in different genres")
    xlab("Genres") + ylab(paste("Distribution of", variable_name))
  print(box_plot)
}

analyze_variable_ANOVA <- function(variable_name) {
  #ANOVA
  avg_by_genre <- tapply(spotify_songs[[variable_name]], spotify_songs$playlist_genre, mean)
  print(avg_by_genre)

  formula_str <- paste(variable_name, "~ playlist_genre")
  anova_result <- aov(as.formula(formula_str), data = spotify_songs)
  print(summary(anova_result))
```

```
  tukey_result <- TukeyHSD(anova_result)
  print(tukey_result)
}
```

```
for (variable in numerical_variables) {
  cat("\n")
  cat("\n")
  analyze_variable_hist(variable)
  cat("\n")
  cat("\n")
  analyze_variable_box(variable)
  cat("\n")
  cat(variable)
  cat("\n")
  analyze_variable_ANOVA (variable)
  print("NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>")
}
```

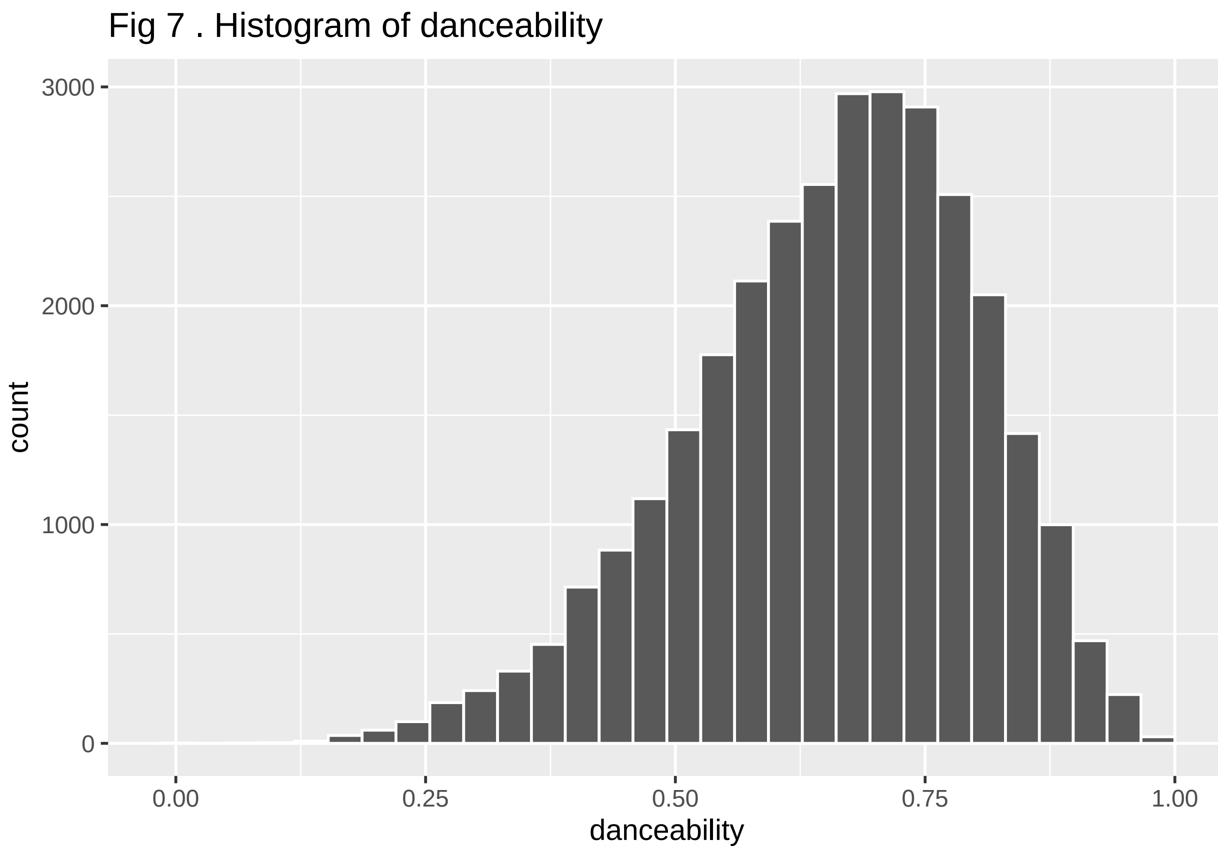
```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation ideoms with `aes()`
```

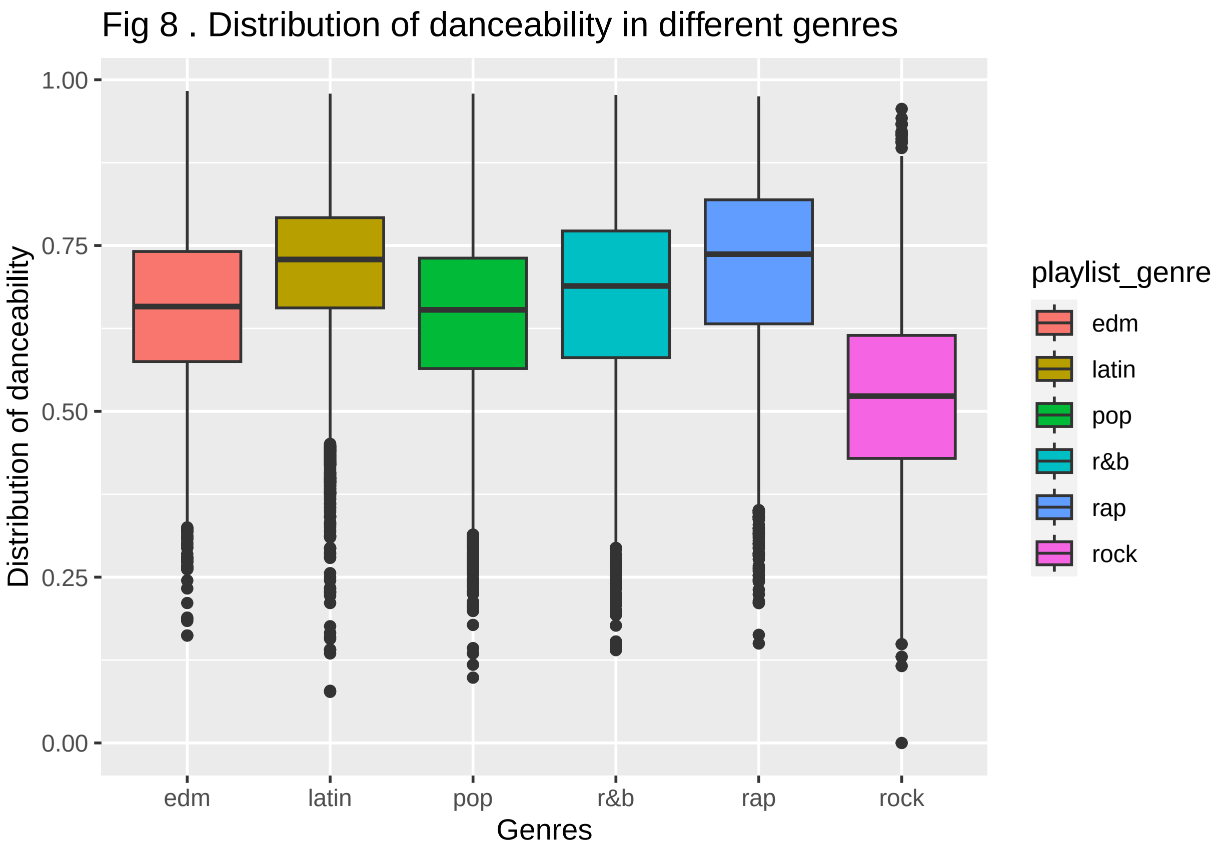





```
##
## track_popularity
##      edm    latin      pop      r&b      rap     rock
## 34.93885 47.56679 48.07147 41.81017 43.62128 41.42247
##                   Df   Sum Sq Mean Sq F value Pr(>F)
## playlist_genre     5   645473  129095   214.4 <2e-16 ***
## Residuals      30941 18627072     602
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                  diff        lwr        upr     p adj
## latin-edm  12.6279435 11.2847705 13.9711166 0.0000000
## pop-edm    13.1326183 11.8131651 14.4520714 0.0000000
## r&b-edm     6.8713181  5.5376085  8.2050277 0.0000000
## rap-edm     8.6824251  7.3737436  9.9911066 0.0000000
## rock-edm    6.4836233  5.0701367  7.8971099 0.0000000
## pop-latin   0.5046747 -0.8762588  1.8856082 0.9039405
## r&b-latin  -5.7566255 -7.1511871 -4.3620638 0.0000000
## rap-latin  -3.9455185 -5.3161635 -2.5748734 0.0000000
## rock-latin -6.1443202 -7.6153624 -4.6732780 0.0000000
## r&b-pop    -6.2613002 -7.6330308 -4.8895695 0.0000000
## rap-pop    -4.4501932 -5.7976020 -3.1027843 0.0000000
```

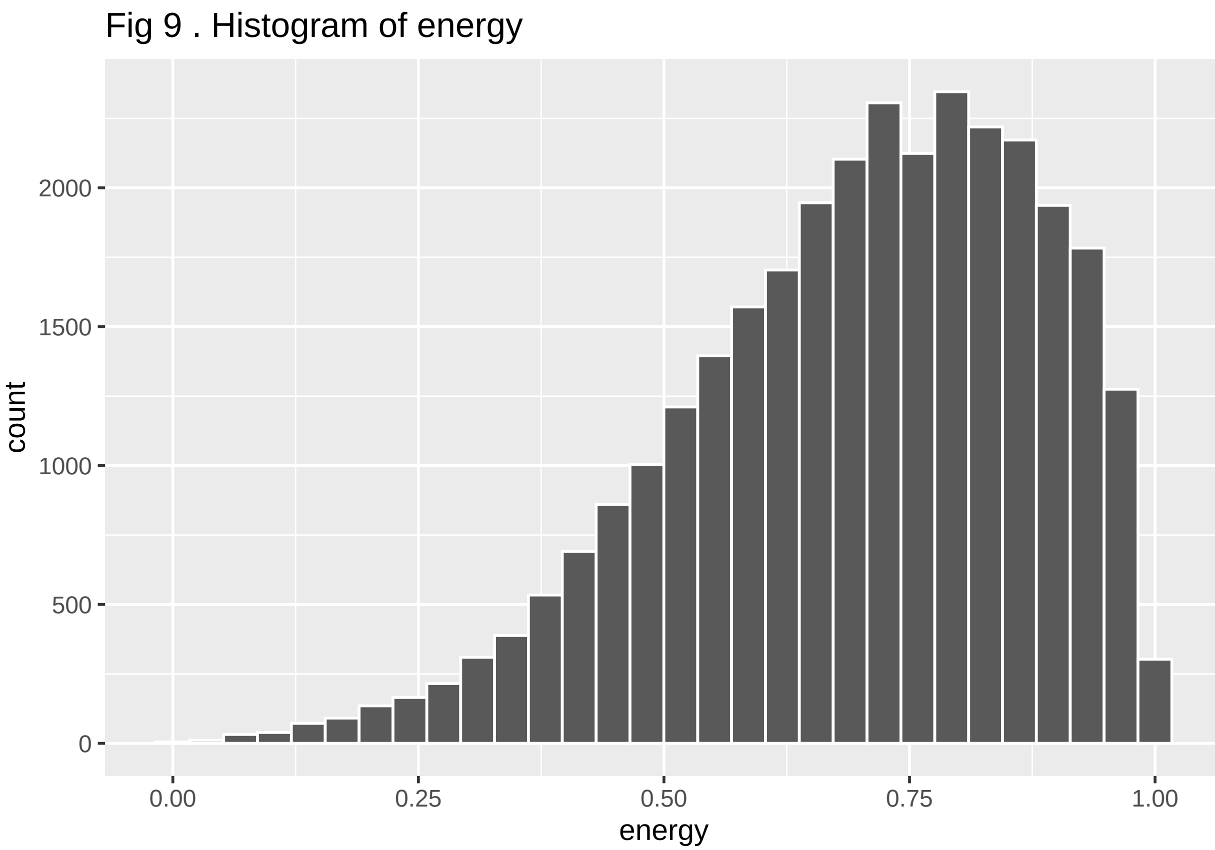
```
## rock-pop   -6.6489949 -8.0984113 -5.1995785 0.0000000
## rap-r&b     1.8111070  0.4497344  3.1724796 0.0020783
## rock-r&b   -0.3876947 -1.8501012  1.0747117 0.9747460
## rock-rap   -2.1988017 -3.6384192 -0.7591843 0.0001952
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

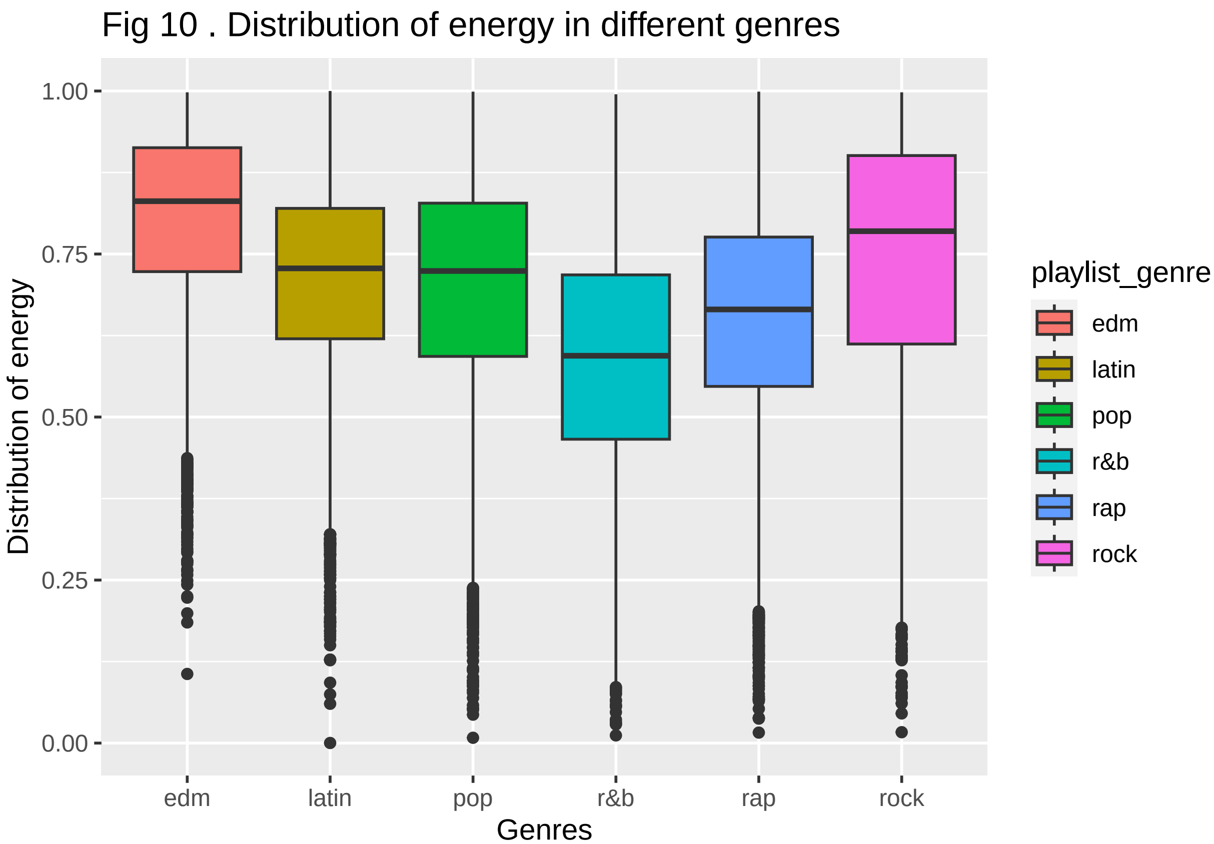





```
##
## danceability
##       edm     latin       pop       r&b       rap      rock
## 0.6546405 0.7134672 0.6406509 0.6697148 0.7175184 0.5201989
##                    Df Sum Sq Mean Sq F value Pr(>F)
## playlist_genre     5  115.7  23.149    1364 <2e-16 ***
## Residuals      30941  525.3   0.017
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                     diff          lwr          upr     p adj
## latin-edm    0.058826742  0.051694070  0.065959414 0.0000000
## pop-edm     -0.013989618 -0.020996330 -0.006982906 0.0000002
## r&b-edm      0.015074287  0.007991868  0.022156705 0.0000000
## rap-edm      0.062877894  0.055928383  0.069827405 0.0000000
## rock-edm    -0.134441537 -0.141947596 -0.126935478 0.0000000
## pop-latin   -0.072816359 -0.080149551 -0.065483168 0.0000000
## r&b-latin   -0.043752455 -0.051158016 -0.036346894 0.0000000
## rap-latin    0.004051152 -0.003227405  0.011329709 0.6077803
## rock-latin  -0.193268278 -0.201079976 -0.185456581 0.0000000
## r&b-pop      0.029063904  0.021779583  0.036348226 0.0000000
## rap-pop      0.076867512  0.069712346  0.084022677 0.0000000
```

```
## rock-pop   -0.120451919 -0.128148776 -0.112755062 0.0000000
## rap-r&b     0.047803607  0.040574290  0.055032925 0.0000000
## rock-r&b   -0.149515823 -0.157281662 -0.141749985 0.0000000
## rock-rap   -0.197319431 -0.204964253 -0.189674609 0.0000000
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

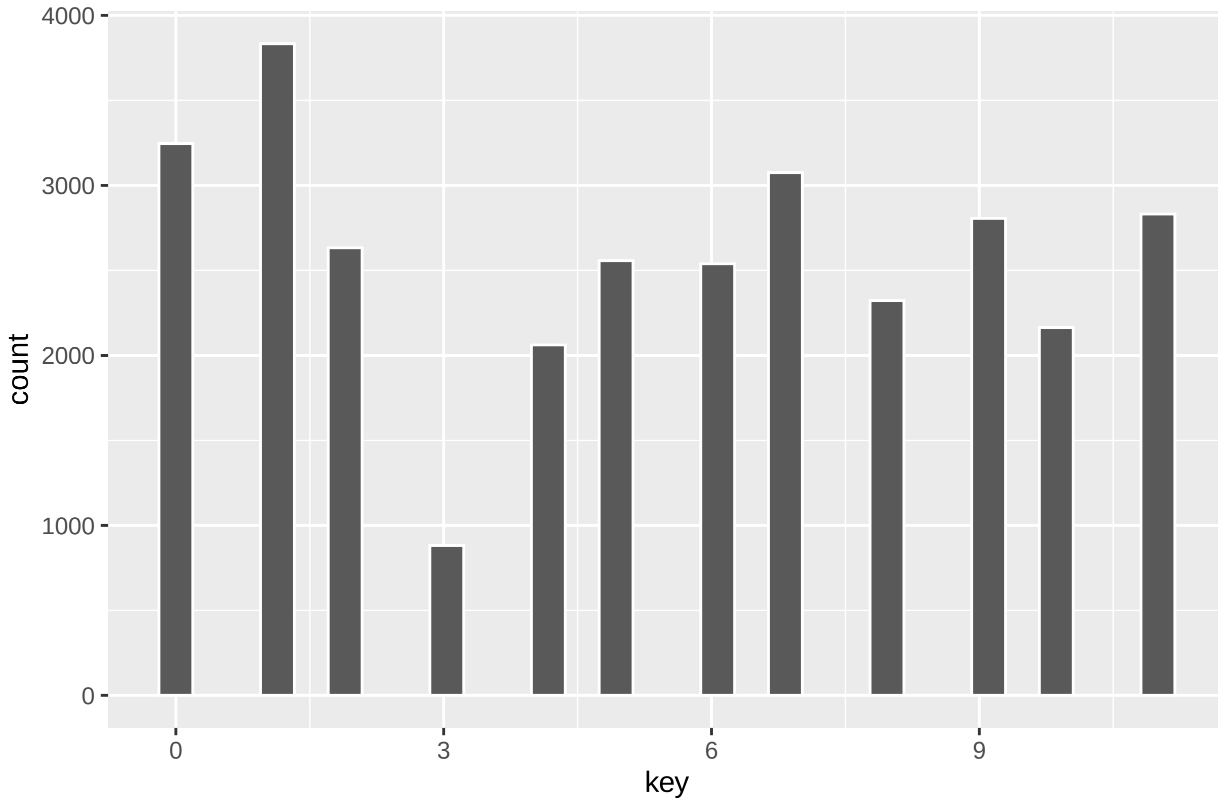


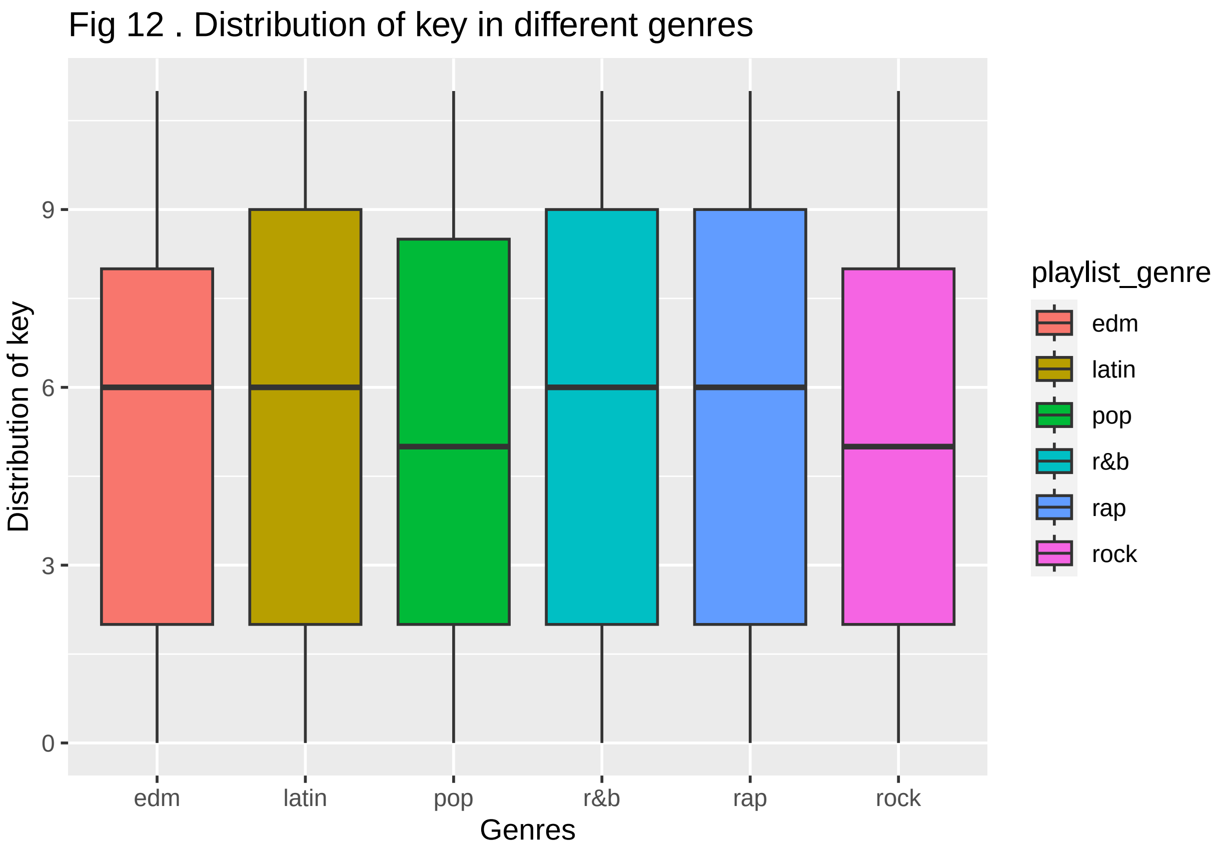



```
##
## energy
##       edm     latin       pop       r&b       rap      rock
## 0.8029283 0.7076550 0.6983240 0.5889020 0.6502701 0.7385113
##                   Df Sum Sq Mean Sq F value Pr(>F)
## playlist_genre     5  146.1  29.212    1045 <2e-16 ***
## Residuals      30941  864.6   0.028
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                   diff         lwr           upr     p adj
## latin-edm  -0.09527329 -0.10442416 -0.0861224264 0.0000000
## pop-edm    -0.10460426 -0.11359353 -0.0956149974 0.0000000
## r&b-edm    -0.21402625 -0.22311265 -0.2049398600 0.0000000
## rap-edm    -0.15265824 -0.16157412 -0.1437423549 0.0000000
## rock-edm   -0.06441699 -0.07404689 -0.0547870819 0.0000000
## pop-latin  -0.00933097 -0.01873910  0.0000771546 0.0533869
## r&b-latin  -0.11875296 -0.12825393 -0.1092519889 0.0000000
## rap-latin  -0.05738494 -0.06672297 -0.0480469105 0.0000000
## rock-latin  0.03085631  0.02083428  0.0408783312 0.0000000
## r&b-pop    -0.10942199 -0.11876742 -0.1000765634 0.0000000
## rap-pop    -0.04805397 -0.05723370 -0.0388742455 0.0000000
```

```
## rock-pop    0.04018728  0.03031259  0.0500619677 0.0000000
## rap-r&b     0.06136802  0.05209316  0.0706428778 0.0000000
## rock-r&b    0.14960927  0.13964608  0.1595724573 0.0000000
## rock-rap    0.08824125  0.07843332  0.0980491802 0.0000000
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

Fig 11 . Histogram of key

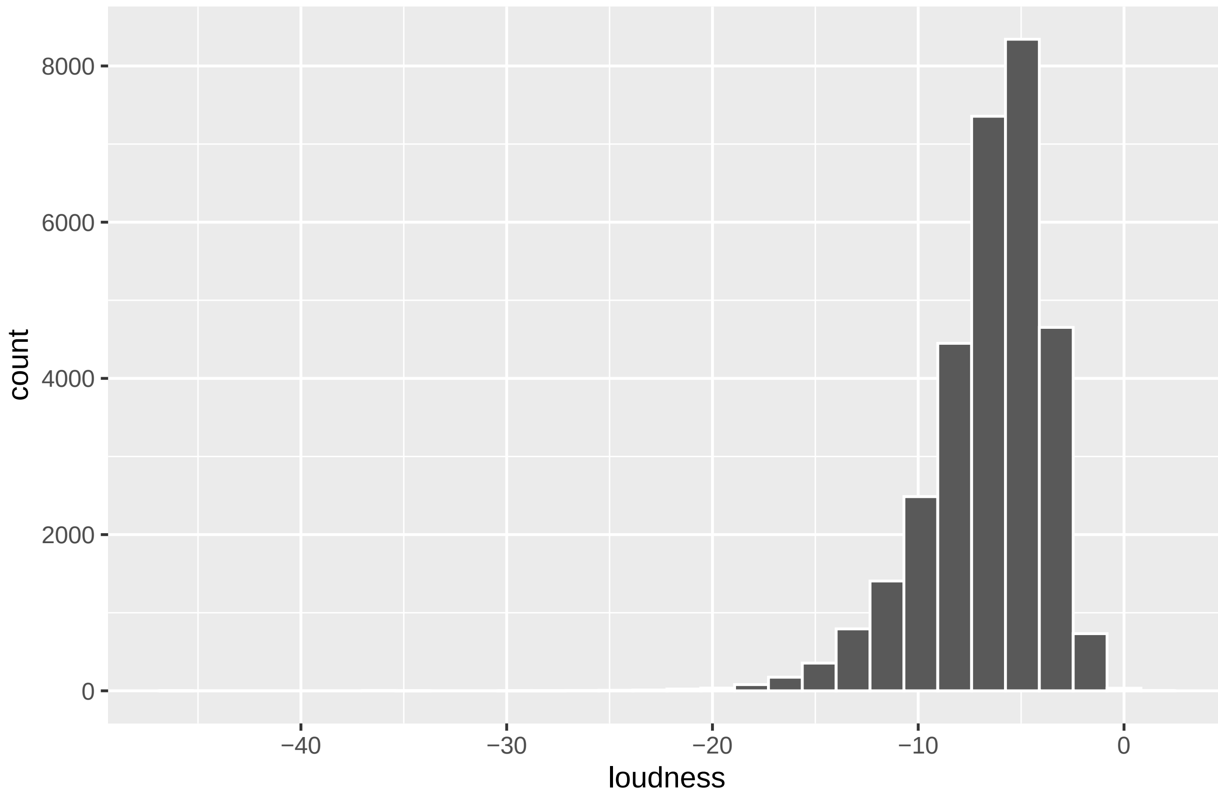


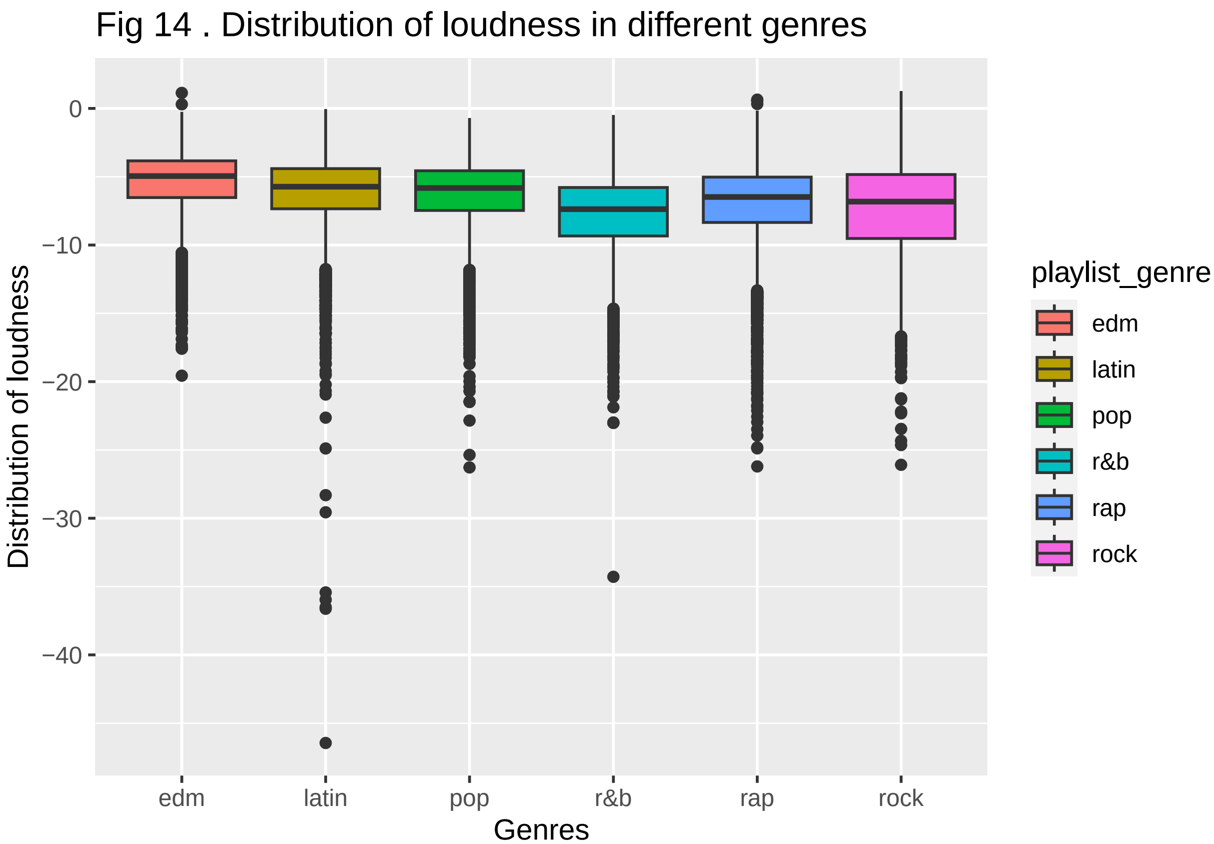



```
##
## key
##      edm    latin      pop      r&b      rap     rock
## 5.351148 5.481161 5.301150 5.390459 5.455675 5.203279
##                  Df Sum Sq Mean Sq F value  Pr(>F)
## playlist_genre     5    246   49.22    3.77 0.00206 **
## Residuals      30941 403958   13.06
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                    diff         lwr         upr     p adj
## latin-edm   0.13001299 -0.06778771  0.32781369 0.4188392
## pop-edm    -0.04999730 -0.24430492  0.14431031 0.9778599
## r&b-edm     0.03931177 -0.15709531  0.23571885 0.9929300
## rap-edm     0.10452778 -0.08819355  0.29724912 0.6345533
## rock-edm   -0.14786812 -0.35602345  0.06028722 0.3282987
## pop-latin  -0.18001030 -0.38337174  0.02335115 0.1175059
## r&b-latin  -0.09070122 -0.29606959  0.11466714 0.8075283
## rap-latin  -0.02548521 -0.22733154  0.17636112 0.9992140
## rock-latin -0.27788111 -0.49451229 -0.06124993 0.0034988
## r&b-pop     0.08930907 -0.11269712  0.29131526 0.8068408
## rap-pop     0.15452509 -0.04389939  0.35294957 0.2287320
```

```
## rock-pop   -0.09787081 -0.31131730  0.11557567 0.7815225
## rap-r&b     0.06521602 -0.13526482  0.26569685 0.9396816
## rock-r&b   -0.18717988 -0.40253934  0.02817957 0.1309320
## rock-rap   -0.25239590 -0.46439936 -0.04039244 0.0090504
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

Fig 13 . Histogram of loudness






```
##
## loudness
##       edm     latin       pop       r&b       rap      rock
## -5.411837 -6.222340 -6.300329 -7.807318 -7.013525 -7.410332
##                   Df Sum Sq Mean Sq F value Pr(>F)
## playlist_genre     5  20647    4129   514.2 <2e-16 ***
## Residuals      30941 248461       8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                   diff        lwr         upr     p adj
## latin-edm  -0.81050326 -0.9656307 -0.65537584 0.000000
## pop-edm    -0.88849278 -1.0408807 -0.73610485 0.000000
## r&b-edm    -2.39548156 -2.5495160 -2.24144711 0.000000
## rap-edm    -1.60168884 -1.7528327 -1.45054497 0.000000
## rock-edm   -1.99849539 -2.1617435 -1.83524724 0.000000
## pop-latin  -0.07798952 -0.2374780  0.08149897 0.731055
## r&b-latin  -1.58497830 -1.7460407 -1.42391586 0.000000
## rap-latin  -0.79118558 -0.9494858 -0.63288533 0.000000
## rock-latin -1.18799213 -1.3578876 -1.01809670 0.000000
## r&b-pop    -1.50698878 -1.6654144 -1.34856316 0.000000
## rap-pop    -0.71319606 -0.8688127 -0.55757944 0.000000
```

```
## rock-pop   -1.11000261 -1.2774004 -0.94260481 0.000000
## rap-r&b     0.79379272  0.6365634  0.95102206 0.000000
## rock-r&b    0.39698617  0.2280881  0.56588423 0.000000
## rock-rap   -0.39680655 -0.5630726 -0.23054046 0.000000
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

Fig 15 . Histogram of mode

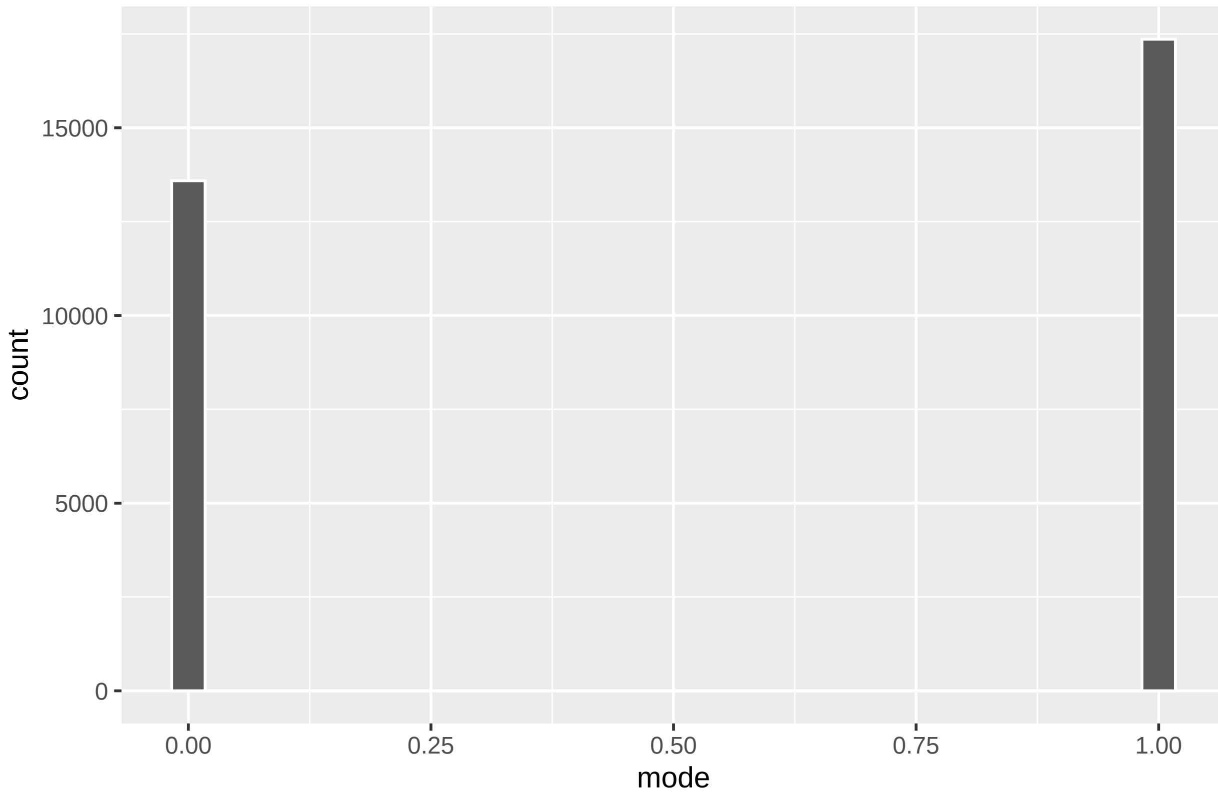
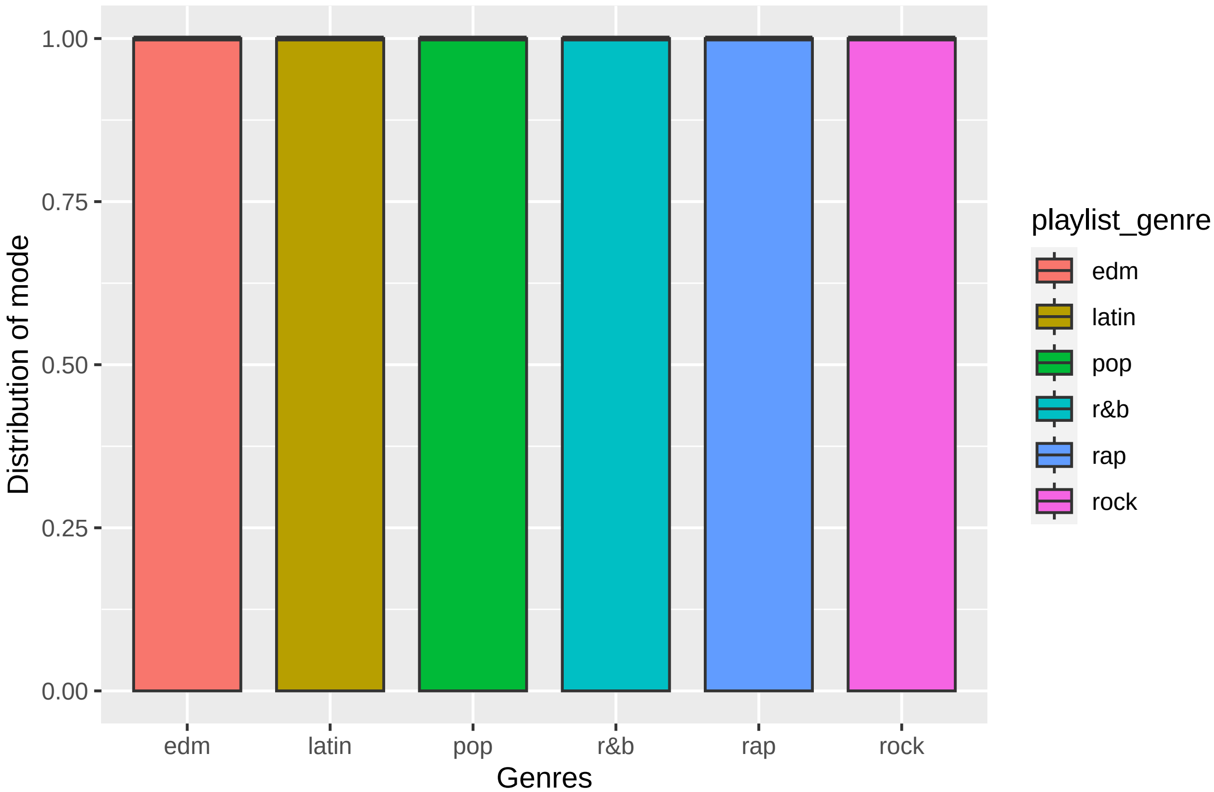


Fig 16 . Distribution of mode in different genres




```
##
## mode
##       edm     latin       pop       r&b       rap      rock
## 0.5191824 0.5617570 0.5855176 0.5188457 0.5189179 0.6956836
##                   Df Sum Sq Mean Sq F value Pr(>F)
## playlist_genre     5    108  21.519   88.61 <2e-16 ***
## Residuals      30941   7514   0.243
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                     diff          lwr         upr     p adj
## latin-edm   4.257456e-02  0.015596958  0.06955216 0.0001005
## pop-edm     6.633519e-02  0.039834002  0.09283638 0.0000000
## r&b-edm    -3.367418e-04 -0.027124270  0.02645079 1.0000000
## rap-edm    -2.645117e-04 -0.026549350  0.02602033 1.0000000
## rock-edm    1.765012e-01  0.148111338  0.20489103 0.0000000
## pop-latin   2.376063e-02 -0.003975389  0.05149665 0.1421710
## r&b-latin  -4.291130e-02 -0.070921040 -0.01490156 0.0001838
## rap-latin  -4.283907e-02 -0.070368446 -0.01530970 0.0001344
## rock-latin  1.339266e-01  0.104380776  0.16347247 0.0000000
## r&b-pop    -6.667193e-02 -0.094223109 -0.03912075 0.0000000
## rap-pop    -6.659970e-02 -0.093662378 -0.03953702 0.0000000
```

```
## rock-pop    1.101660e-01  0.081054500  0.13927749 0.0000000
## rap-r&b     7.223008e-05 -0.027270908  0.02741537 1.0000000
## rock-r&b    1.768379e-01  0.147465526  0.20621033 0.0000000
## rock-rap    1.767657e-01  0.147851012  0.20568038 0.0000000
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

Fig 17 . Histogram of speechiness

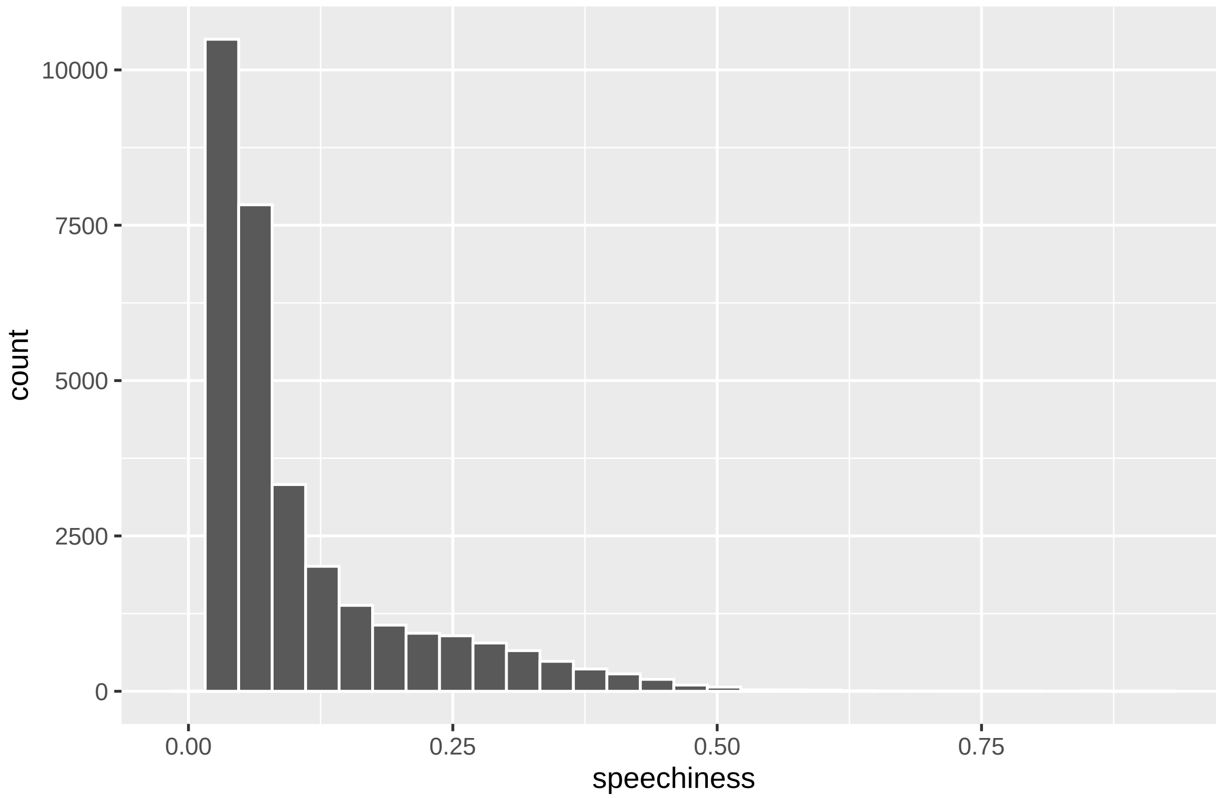
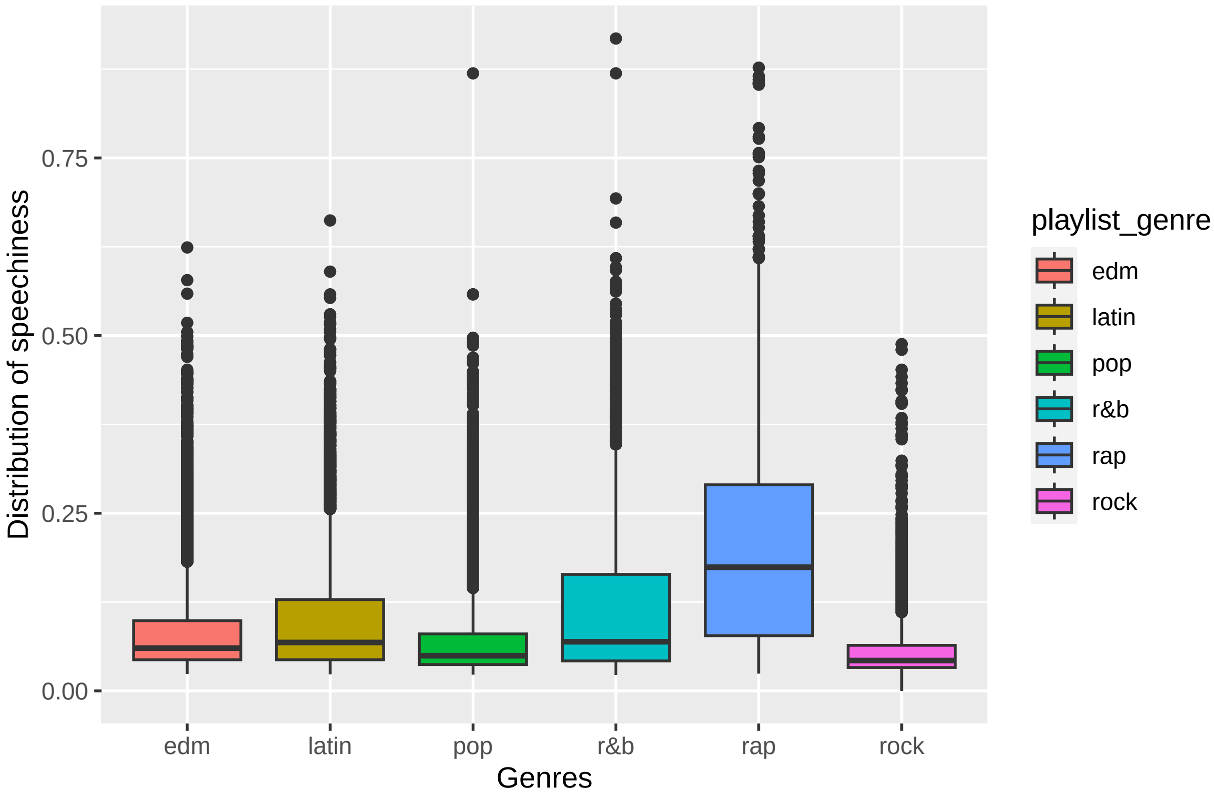


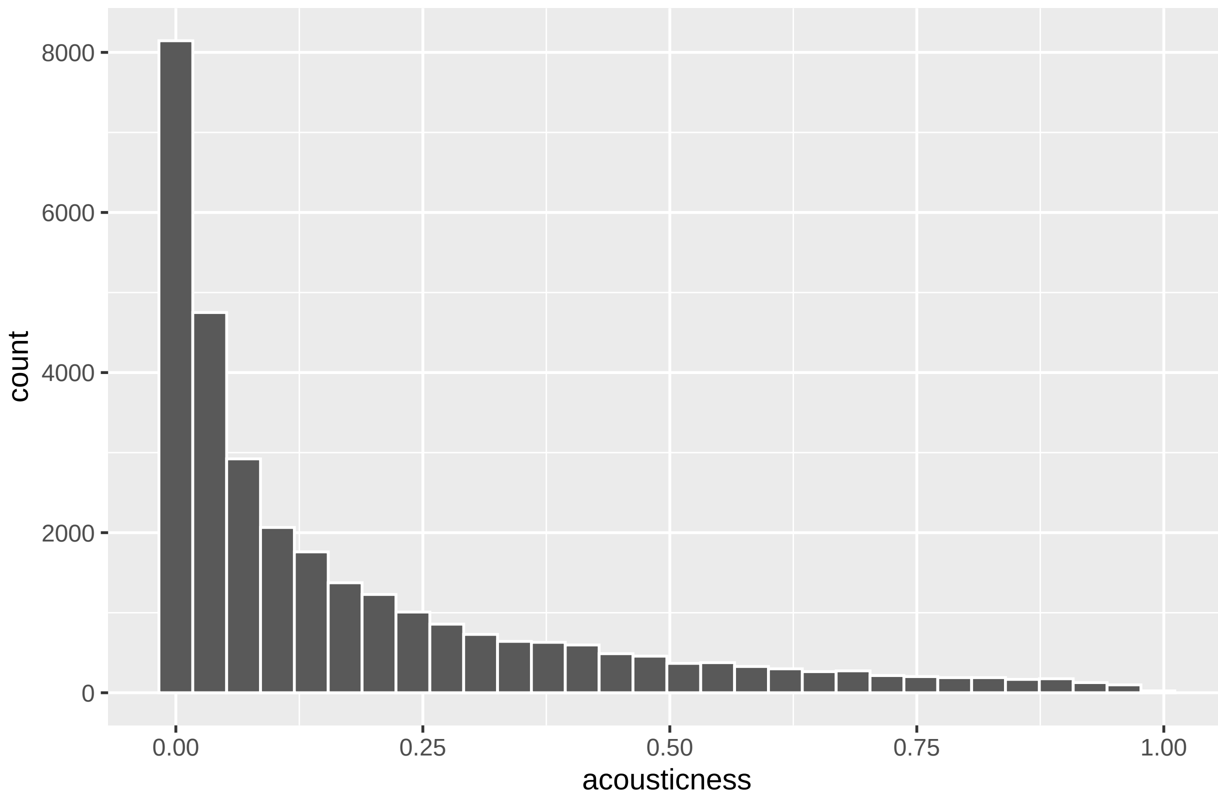
Fig 18 . Distribution of speechiness in different genres

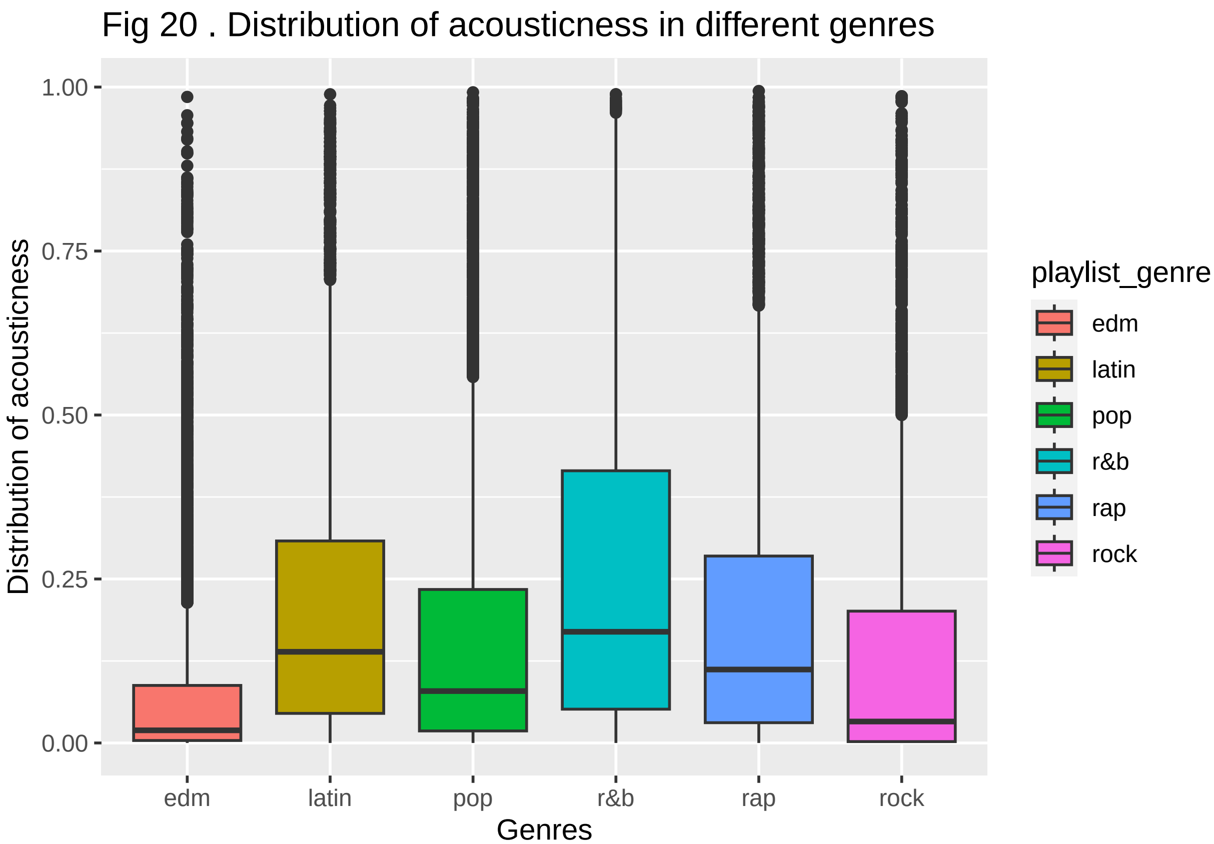



```
##
## speechiness
##        edm      latin        pop        r&b        rap       rock
## 0.08687212 0.10327175 0.07482399 0.11890188 0.19613449 0.05865602
##                    Df Sum Sq Mean Sq F value Pr(>F)
## playlist_genre     5  61.81  12.362    1478 <2e-16 ***
## Residuals      30941 258.76   0.008
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                   diff         lwr         upr p adj
## latin-edm   0.01639963  0.01139338  0.02140587     0
## pop-edm    -0.01204814 -0.01696597 -0.00713030     0
## r&b-edm     0.03202976  0.02705879  0.03700073     0
## rap-edm     0.10926237  0.10438468  0.11414006     0
## rock-edm   -0.02821611 -0.03348442 -0.02294779     0
## pop-latin  -0.02844776 -0.03359475 -0.02330078     0
## r&b-latin   0.01563013  0.01043236  0.02082791     0
## rap-latin   0.09286274  0.08775410  0.09797138     0
## rock-latin -0.04461573 -0.05009857 -0.03913290     0
## r&b-pop     0.04407790  0.03896521  0.04919058     0
## rap-pop     0.12131050  0.11628847  0.12633254     0
```

```
## rock-pop   -0.01616797 -0.02157020 -0.01076574     0
## rap-r&b     0.07723261  0.07215853  0.08230668     0
## rock-r&b   -0.06024587 -0.06569652 -0.05479522     0
## rock-rap   -0.13747847 -0.14284418 -0.13211276     0
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

Fig 19 . Histogram of acousticness

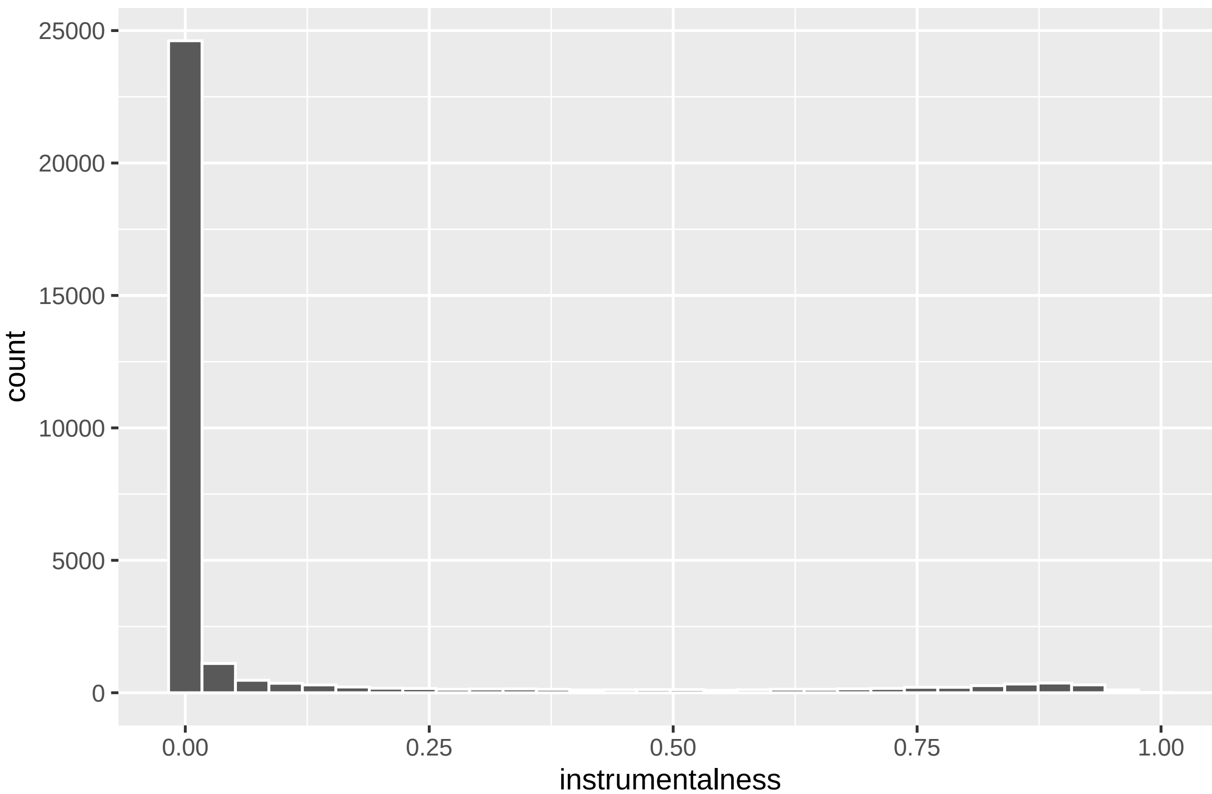


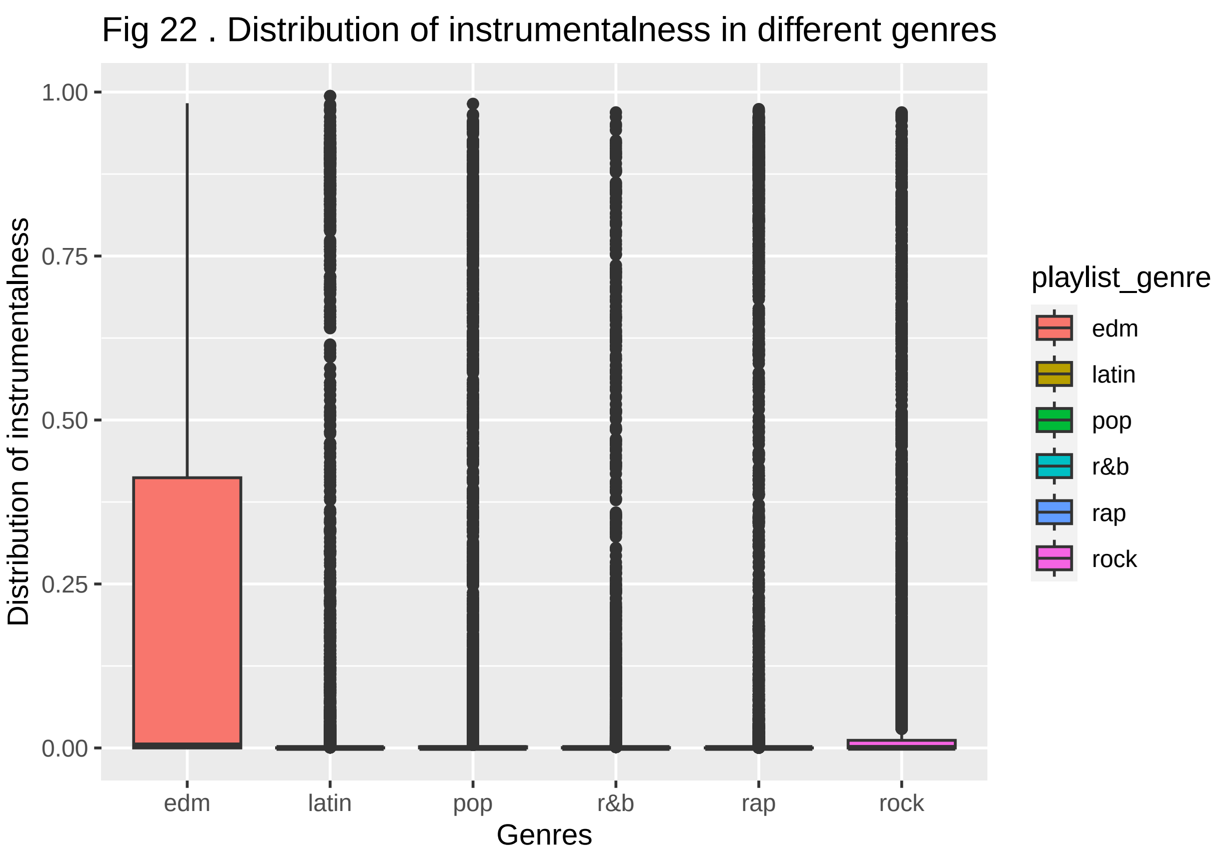



```
##
## acousticness
##       edm     latin       pop       r&b       rap      rock
## 0.0816723 0.2110486 0.1730262 0.2629757 0.1959889 0.1400041
##                   Df Sum Sq Mean Sq F value Pr(>F)
## playlist_genre     5  105.3  21.070     468 <2e-16 ***
## Residuals      30941 1392.9   0.045
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                   diff         lwr          upr     p adj
## latin-edm   0.12937634  0.11776115  0.140991532 0.0000000
## pop-edm     0.09135387  0.07994379  0.102763938 0.0000000
## r&b-edm     0.18130339  0.16977004  0.192836749 0.0000000
## rap-edm     0.11431655  0.10299963  0.125633477 0.0000000
## rock-edm    0.05833177  0.04610854  0.070555005 0.0000000
## pop-latin  -0.03802247 -0.04996420 -0.026080746 0.0000000
## r&b-latin   0.05192705  0.03986748  0.063986631 0.0000000
## rap-latin  -0.01505979 -0.02691254 -0.003207029 0.0039839
## rock-latin -0.07104457 -0.08376552 -0.058323620 0.0000000
## r&b-pop     0.08994953  0.07808738  0.101811672 0.0000000
## rap-pop     0.02296269  0.01131087  0.034614509 0.0000003
```

```
## rock-pop   -0.03302209 -0.04555603 -0.020488156 0.0000000
## rap-r&b    -0.06698684 -0.07875941 -0.055214266 0.0000000
## rock-r&b   -0.12297162 -0.13561789 -0.110325351 0.0000000
## rock-rap   -0.05598478 -0.06843398 -0.043535581 0.0000000
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

Fig 21 . Histogram of instrumentalness






```
##
## instrumentalness
##        edm      latin        pop        r&b        rap       rock
## 0.21881824 0.04495271 0.05858493 0.02913776 0.07864474 0.06558976
##                   Df Sum Sq Mean Sq F value Pr(>F)
## playlist_genre     5  136.1  27.222   575.3 <2e-16 ***
## Residuals      30941 1464.0   0.047
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                    diff          lwr           upr     p adj
## latin-edm  -0.173865530 -0.185773460 -0.1619575999 0.0000000
## pop-edm    -0.160233306 -0.171930947 -0.1485356657 0.0000000
## r&b-edm    -0.189680485 -0.201504516 -0.1778564528 0.0000000
## rap-edm    -0.140173500 -0.151775645 -0.1285713561 0.0000000
## rock-edm   -0.153228478 -0.165759774 -0.1406971814 0.0000000
## pop-latin   0.013632224  0.001389527  0.0258749198 0.0188423
## r&b-latin  -0.015814955 -0.028178471 -0.0034514381 0.0036354
## rap-latin   0.033692030  0.021540546  0.0458435136 0.0000000
## rock-latin  0.020637052  0.007595496  0.0336786089 0.0000950
## r&b-pop    -0.029447178 -0.041608286 -0.0172860705 0.0000000
## rap-pop     0.020059806  0.008114323  0.0320052890 0.0000252
```

```
## rock-pop     0.007004829 -0.005845004  0.0198546614 0.6293906
## rap-r&b      0.049506984  0.037437706  0.0615762630 0.0000000
## rock-r&b     0.036452007  0.023487011  0.0494170031 0.0000000
## rock-rap    -0.013054977 -0.025817937 -0.0002920174 0.0414948
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

Fig 23 . Histogram of liveness

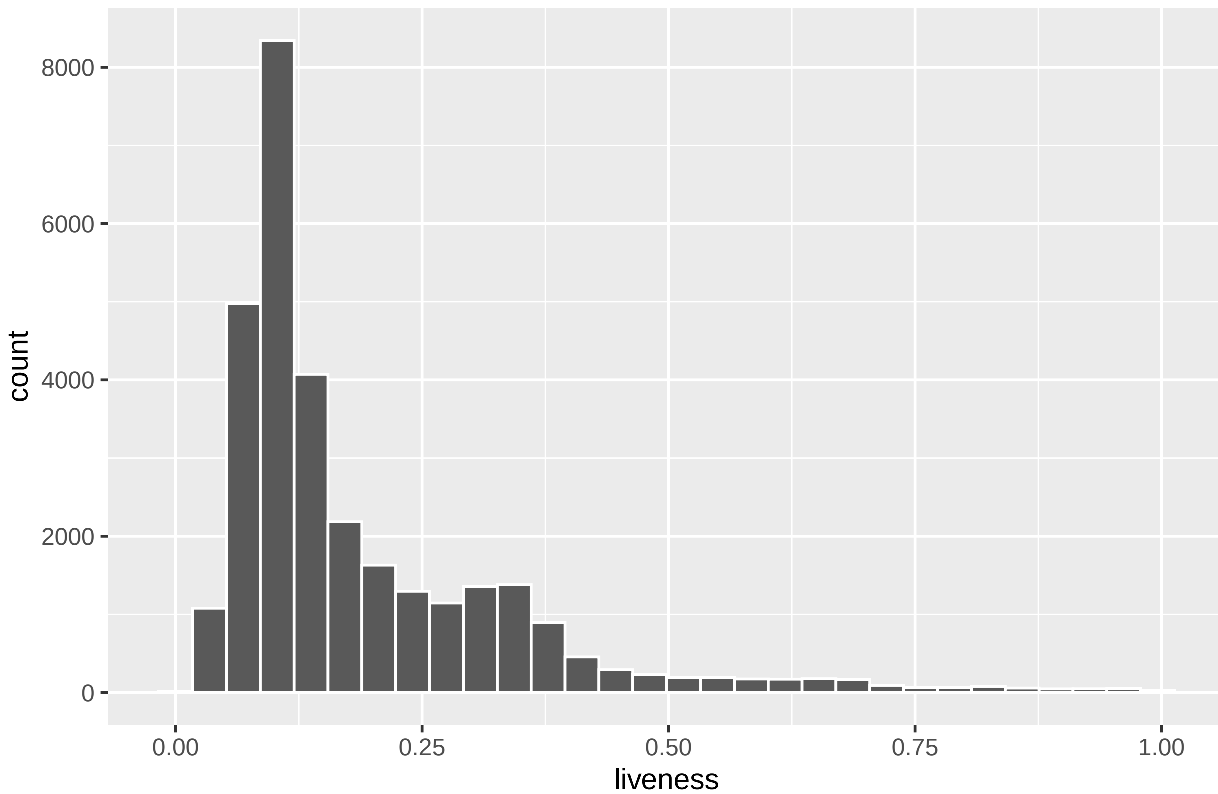
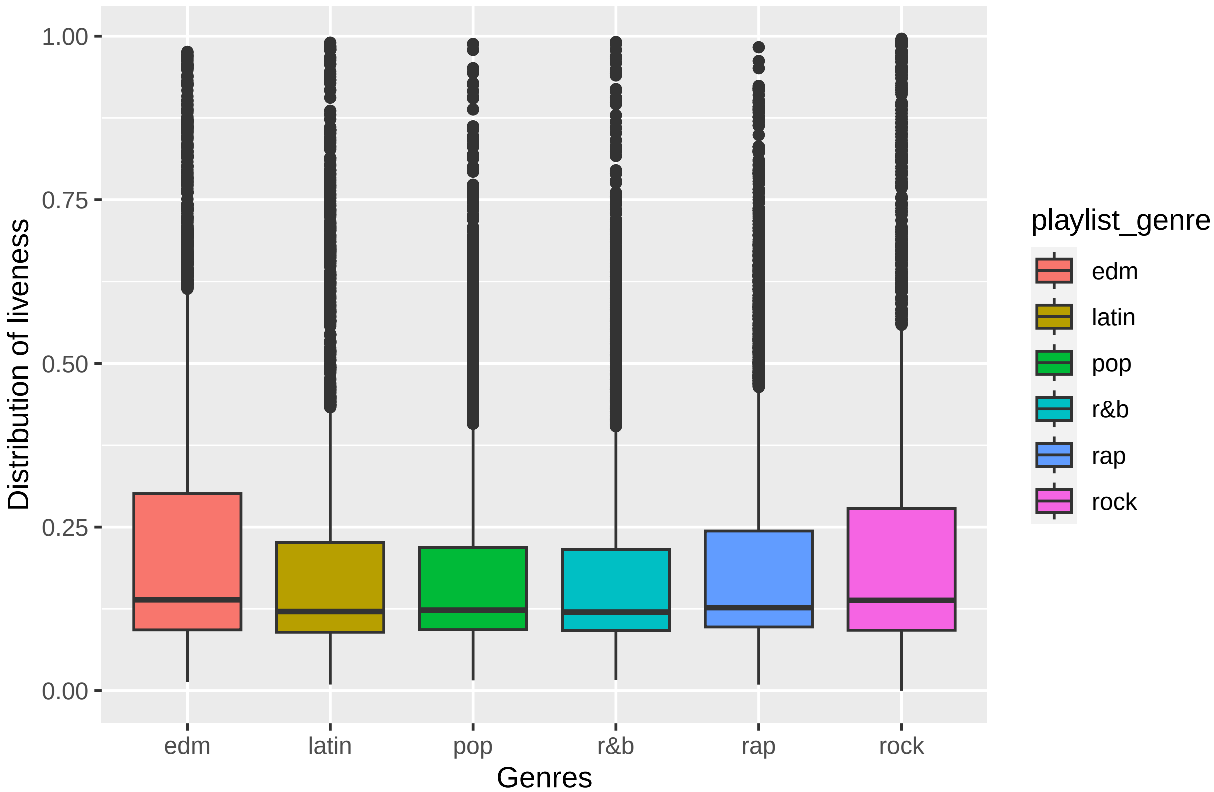


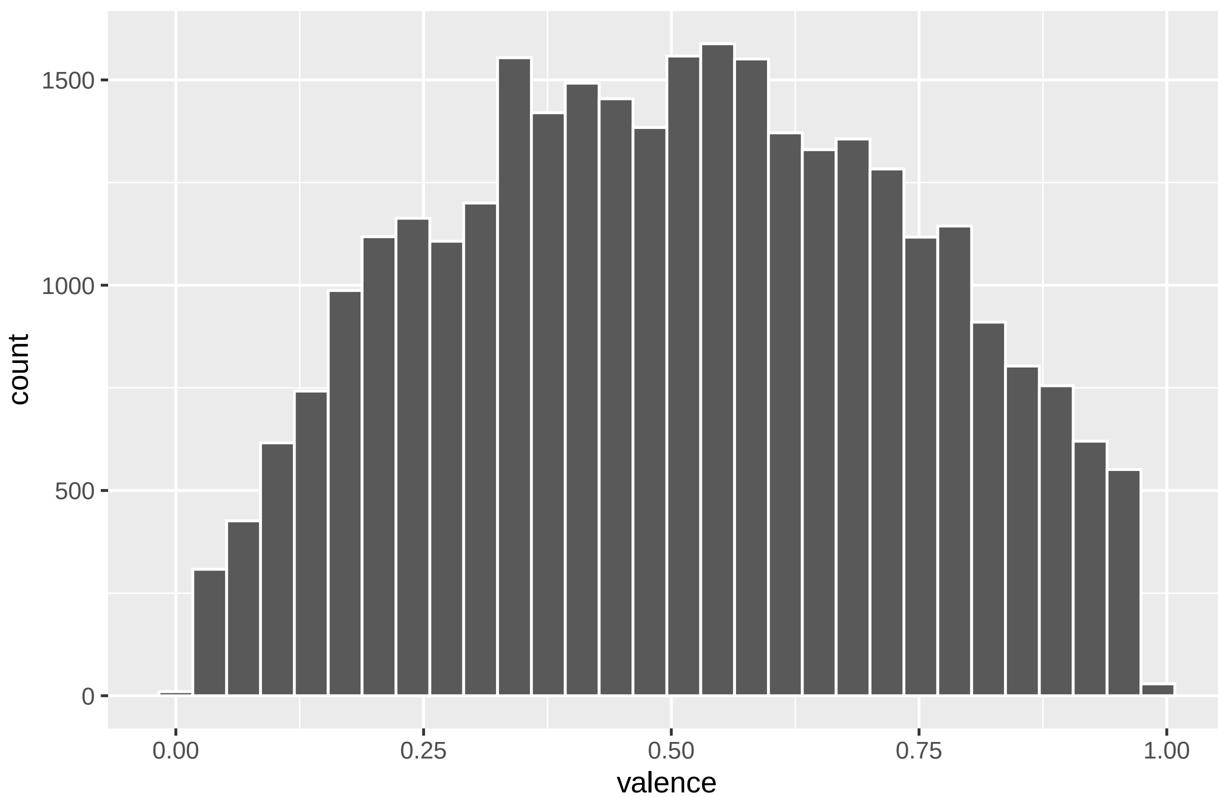
Fig 24 . Distribution of liveness in different genres

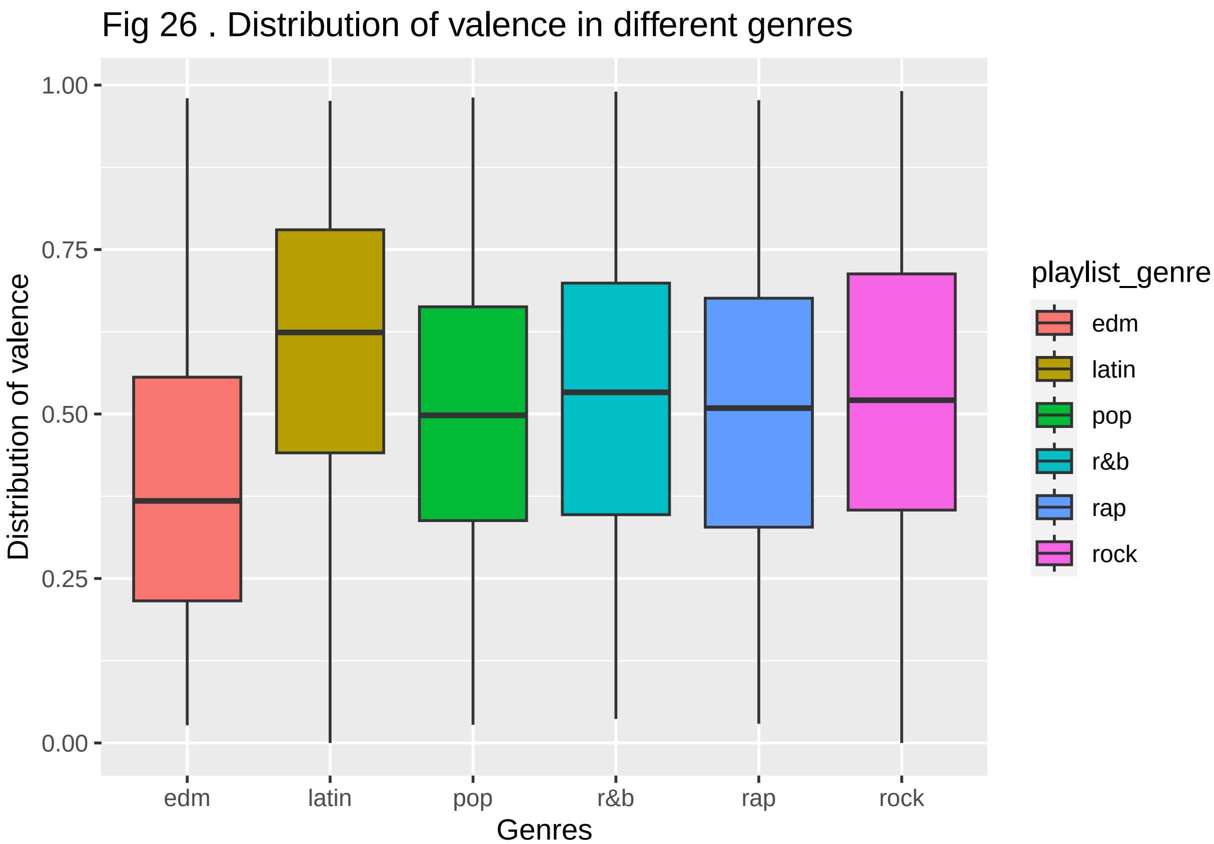



```
##
## liveness
##       edm     latin       pop       r&b       rap      rock
## 0.2125270 0.1803575 0.1767067 0.1749040 0.1899724 0.2045341
##                    Df Sum Sq Mean Sq F value Pr(>F)
## playlist_genre      5    6.5  1.2929   55.04 <2e-16 ***
## Residuals       30941  726.9  0.0235
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                    diff          lwr           upr     p adj
## latin-edm  -0.032169489 -0.040559929 -0.0237790481 0.0000000
## pop-edm    -0.035820312 -0.044062580 -0.0275780433 0.0000000
## r&b-edm    -0.037622962 -0.045954287 -0.0292916373 0.0000000
## rap-edm    -0.022554632 -0.030729613 -0.0143796511 0.0000000
## rock-edm   -0.007992933 -0.016822604  0.0008367368 0.1022232
## pop-latin  -0.003650823 -0.012277143  0.0049754965 0.8341357
## r&b-latin  -0.005453474 -0.014164925  0.0032579772 0.4761509
## rap-latin   0.009614857  0.001052806  0.0181769074 0.0172714
## rock-latin  0.024176555  0.014987351  0.0333657595 0.0000000
## r&b-pop    -0.001802650 -0.010371482  0.0067661814 0.9910974
## rap-pop     0.013265680  0.004848780  0.0216825805 0.0001033
```

```
## rock-pop    0.027827378  0.018773265  0.0368814923 0.0000000
## rap-r&b     0.015068330  0.006564202  0.0235724586 0.0000066
## rock-r&b    0.029630029  0.020494770  0.0387652880 0.0000000
## rock-rap    0.014561698  0.005568796  0.0235546010 0.0000579
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

Fig 25 . Histogram of valence

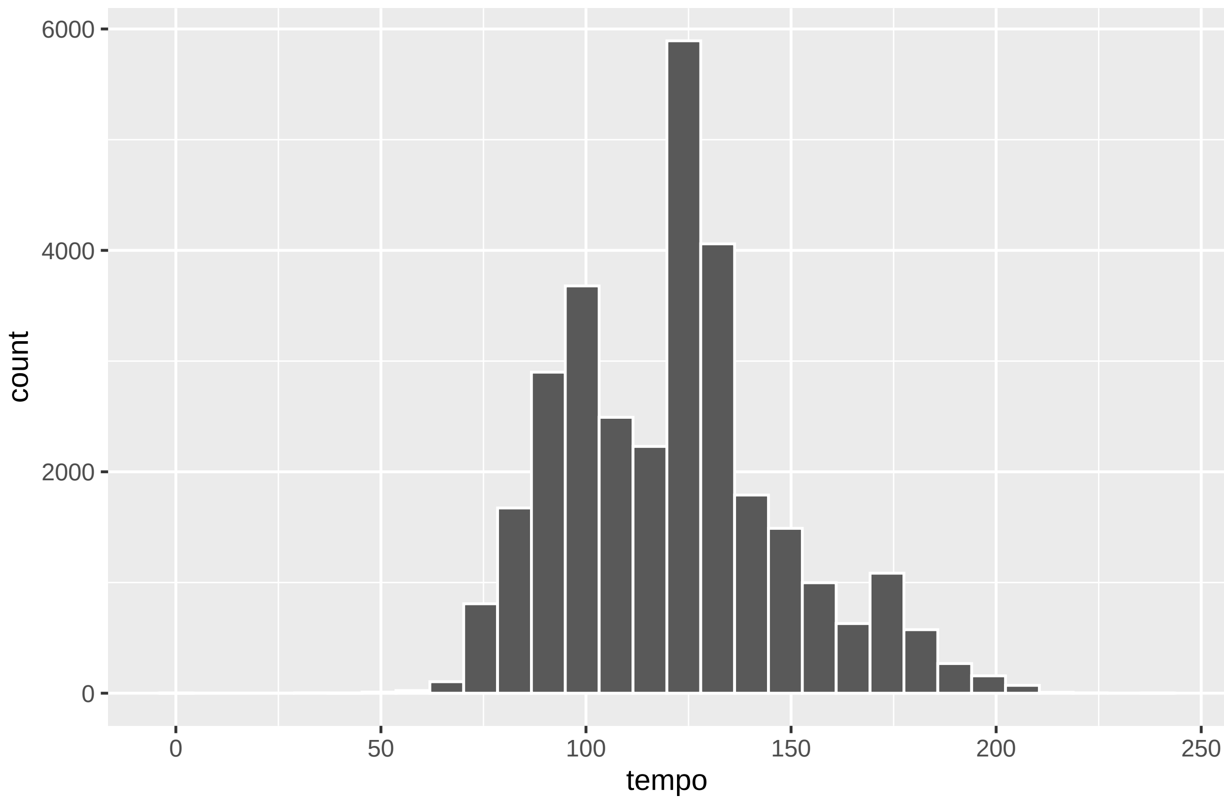


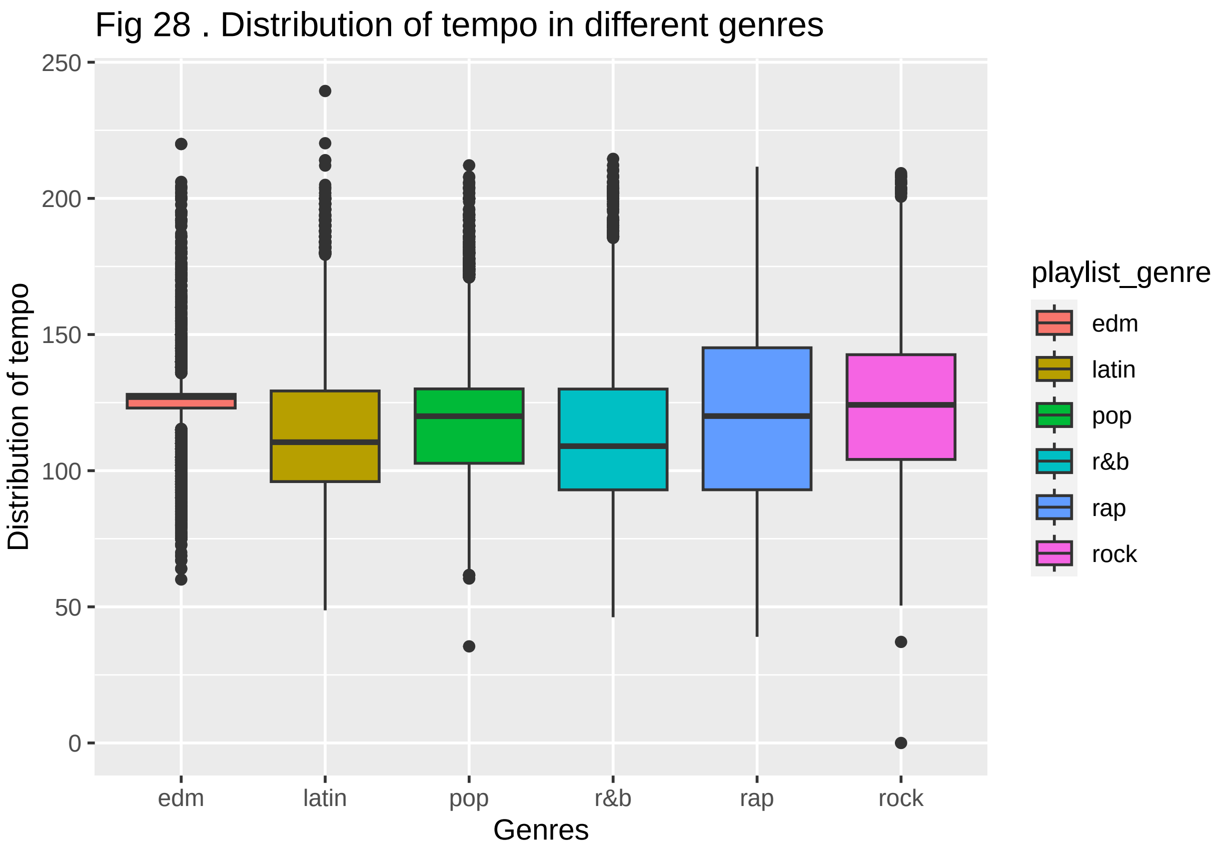



```
##
## valence
##       edm     latin       pop       r&b       rap      rock
## 0.3984501 0.6027073 0.5004120 0.5236551 0.5000706 0.5311139
##                   Df Sum Sq Mean Sq F value Pr(>F)
## playlist_genre     5    120   24.00   477.1 <2e-16 ***
## Residuals      30941   1556    0.05
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                     diff          lwr         upr     p adj
## latin-edm   0.2042571750  0.191979625  0.21653472 0.0000000
## pop-edm     0.1019619220  0.089901189  0.11402265 0.0000000
## r&b-edm     0.1252050148  0.113013968  0.13739606 0.0000000
## rap-edm     0.1016204632  0.089658191  0.11358274 0.0000000
## rock-edm    0.1326637807  0.119743516  0.14558405 0.0000000
## pop-latin  -0.1022952530 -0.114917960 -0.08967255 0.0000000
## r&b-latin  -0.0790521602 -0.091799437 -0.06630488 0.0000000
## rap-latin  -0.1026367118 -0.115165375 -0.09010805 0.0000000
## rock-latin -0.0715933943 -0.085039758 -0.05814703 0.0000000
## r&b-pop     0.0232430927  0.010704507  0.03578168 0.0000019
## rap-pop    -0.0003414588 -0.012657727  0.01197481 0.9999996
```

```
## rock-pop     0.0307018586  0.017453170  0.04395055 0.0000000
## rap-r&b     -0.0235845516 -0.036028458 -0.01114065 0.0000010
## rock-r&b     0.0074587659 -0.005908661  0.02082619 0.6051412
## rock-rap     0.0310433175  0.017884198  0.04420244 0.0000000
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

Fig 27 . Histogram of tempo

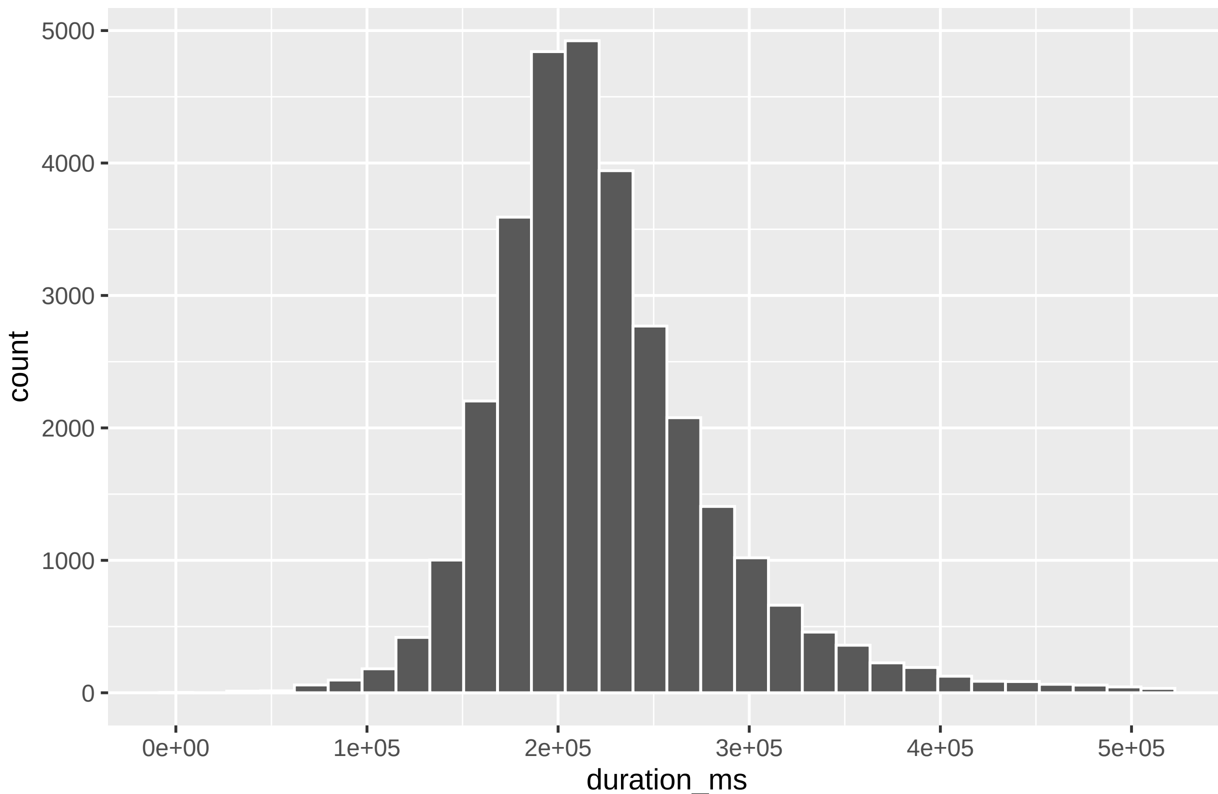


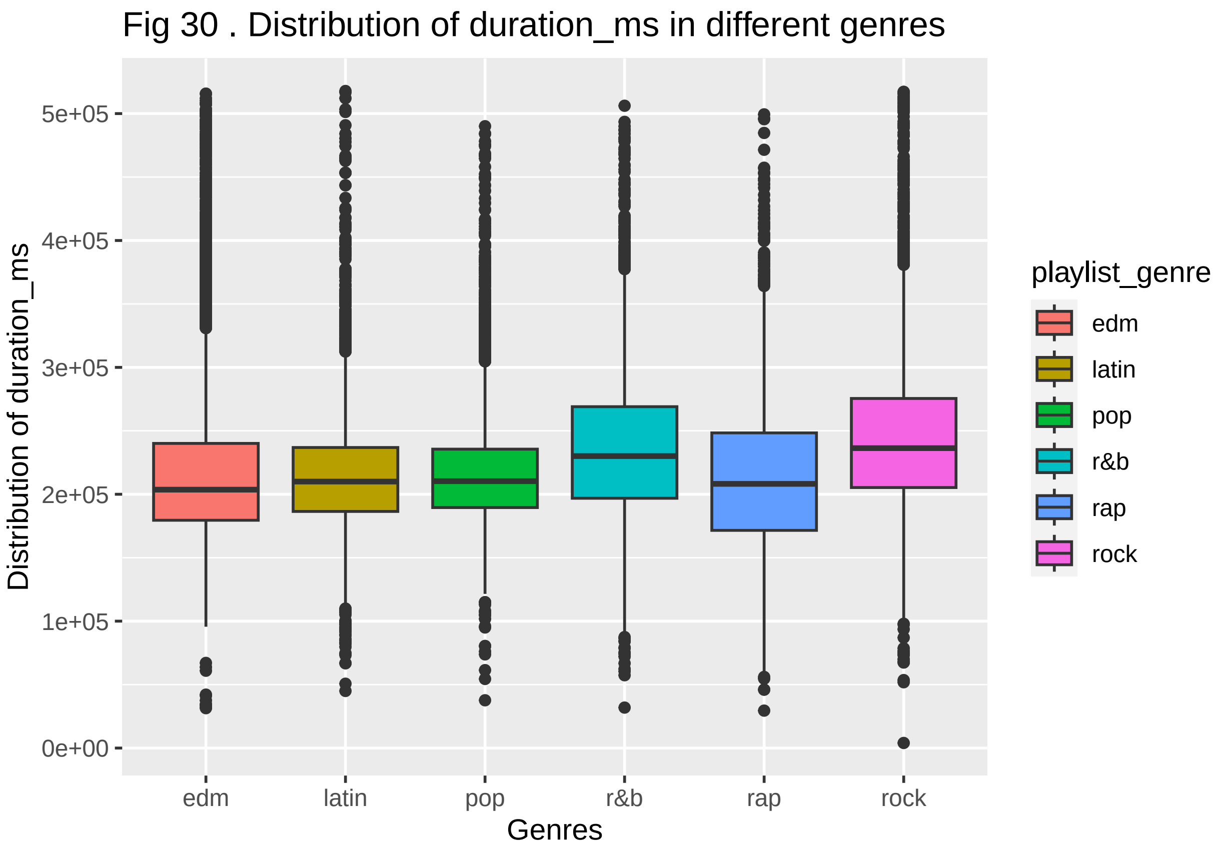



```
##
## tempo
##      edm    latin      pop      r&b      rap     rock
## 125.7464 118.7115 120.4868 114.2492 121.1452 125.2136
##                    Df   Sum Sq Mean Sq F value Pr(>F)
## playlist_genre      5   467636   93527   132.5 <2e-16 ***
## Residuals       30941 21841480     706
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                    diff         lwr         upr     p adj
## latin-edm    -7.0348422  -8.4892986  -5.5803857 0.0000000
## pop-edm      -5.2595762  -6.6883475  -3.8308049 0.0000000
## r&b-edm     -11.4971479 -12.9413568 -10.0529390 0.0000000
## rap-edm      -4.6011697  -6.0182768  -3.1840625 0.0000000
## rock-edm     -0.5328016  -2.0633971   0.9977939 0.9206764
## pop-latin     1.7752660   0.2799206   3.2706113 0.0093529
## r&b-latin    -4.4623057  -5.9724083  -2.9522032 0.0000000
## rap-latin     2.4336725   0.9494680   3.9178770 0.0000437
## rock-latin    6.5020406   4.9091209   8.0949602 0.0000000
## r&b-pop      -6.2375717  -7.7229517  -4.7521917 0.0000000
## rap-pop       0.6584065  -0.8006366   2.1174497 0.7929058
```

```
## rock-pop     4.7267746   3.1572725   6.2962767 0.0000000
## rap-r&b      6.8959782   5.4218144   8.3701420 0.0000000
## rock-r&b    10.9643463   9.3807779  12.5479147 0.0000000
## rock-rap     4.0683681   2.5094767   5.6272594 0.0000000
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

Fig 29 . Histogram of duration_ms






```
##
## duration_ms
##      edm    latin      pop      r&b      rap     rock
## 221647.9 215760.6 216827.5 236093.9 211773.5 247319.9
##                   Df    Sum Sq   Mean Sq F value Pr(>F)
## playlist_genre     5 4.461e+12 8.922e+11   266.2 <2e-16 ***
## Residuals      30941 1.037e+14 3.351e+09
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                   diff        lwr        upr     p adj
## latin-edm    -5887.289  -9056.156  -2718.422 0.0000018
## pop-edm      -4820.402  -7933.309  -1707.496 0.0001484
## r&b-edm      14445.948  11299.407  17592.489 0.0000000
## rap-edm      -9874.436 -12961.930  -6786.943 0.0000000
## rock-edm     25671.992  22337.238  29006.746 0.0000000
## pop-latin     1066.887  -2191.067   4324.840 0.9380135
## r&b-latin    20333.237  17043.131  23623.342 0.0000000
## rap-latin    -3987.148  -7220.828   -753.467 0.0058983
## rock-latin   31559.281  28088.740  35029.823 0.0000000
## r&b-pop      19266.350  16030.108  22502.591 0.0000000
## rap-pop      -5054.034  -8232.895  -1875.174 0.0000860
```

```
## rock-pop     30492.394  27072.874  33911.915 0.0000000
## rap-r&b     -24320.384 -27532.189 -21108.580 0.0000000
## rock-r&b     11226.045   7775.877  14676.212 0.0000000
## rock-rap     35546.429  32150.026  38942.832 0.0000000
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

Fig 31 . Histogram of track_album_release_year

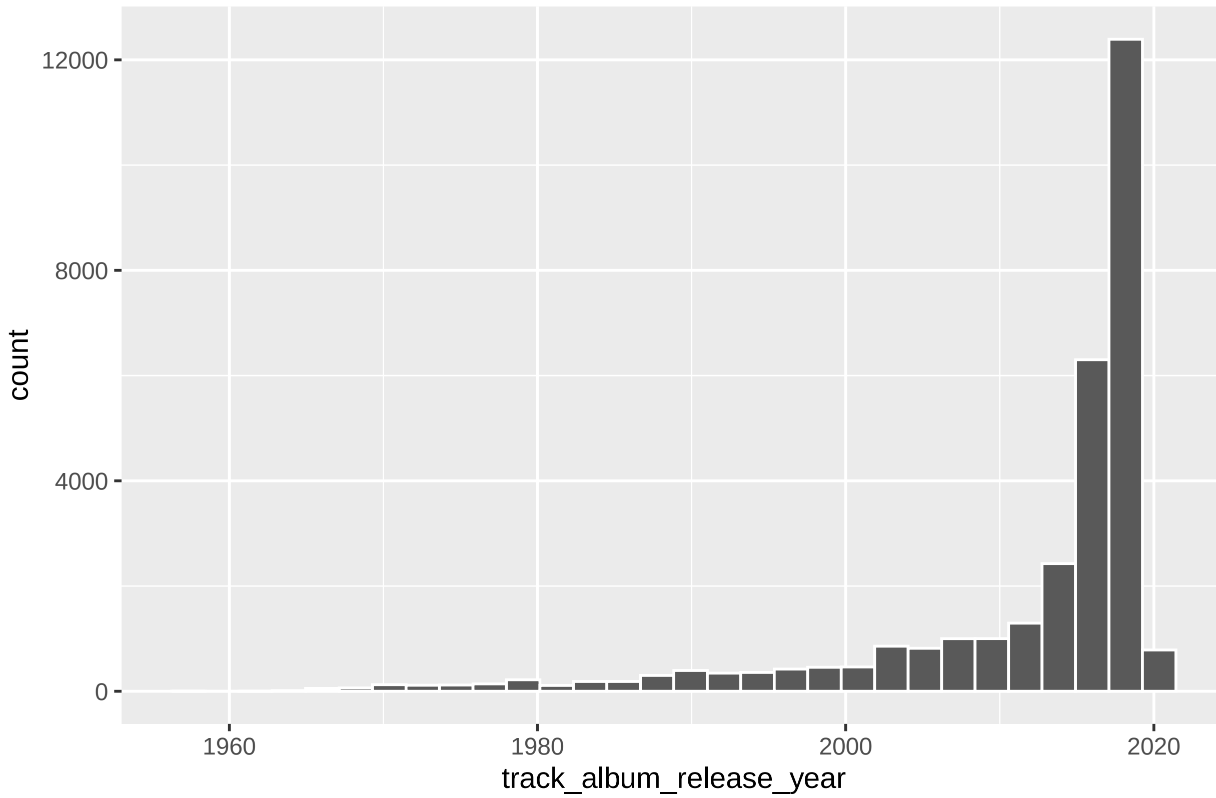
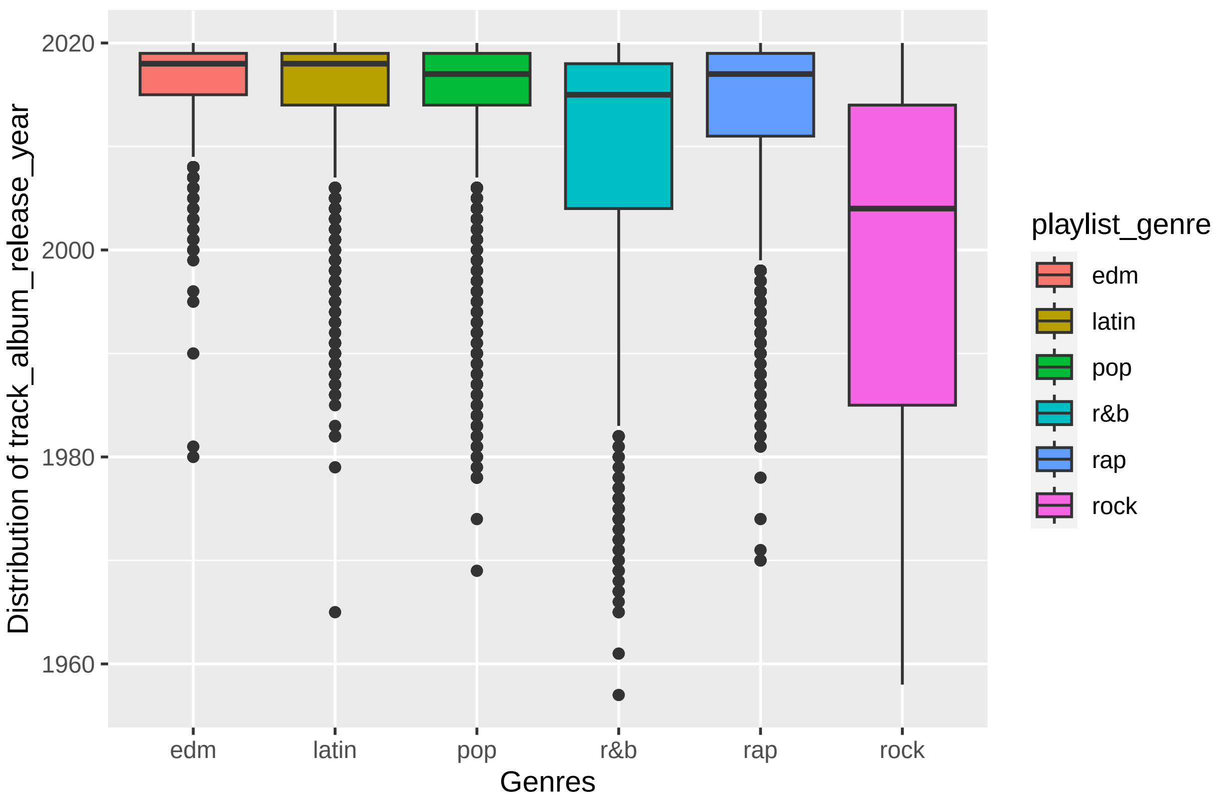


Fig 32 . Distribution of track_album_release_year in different genres

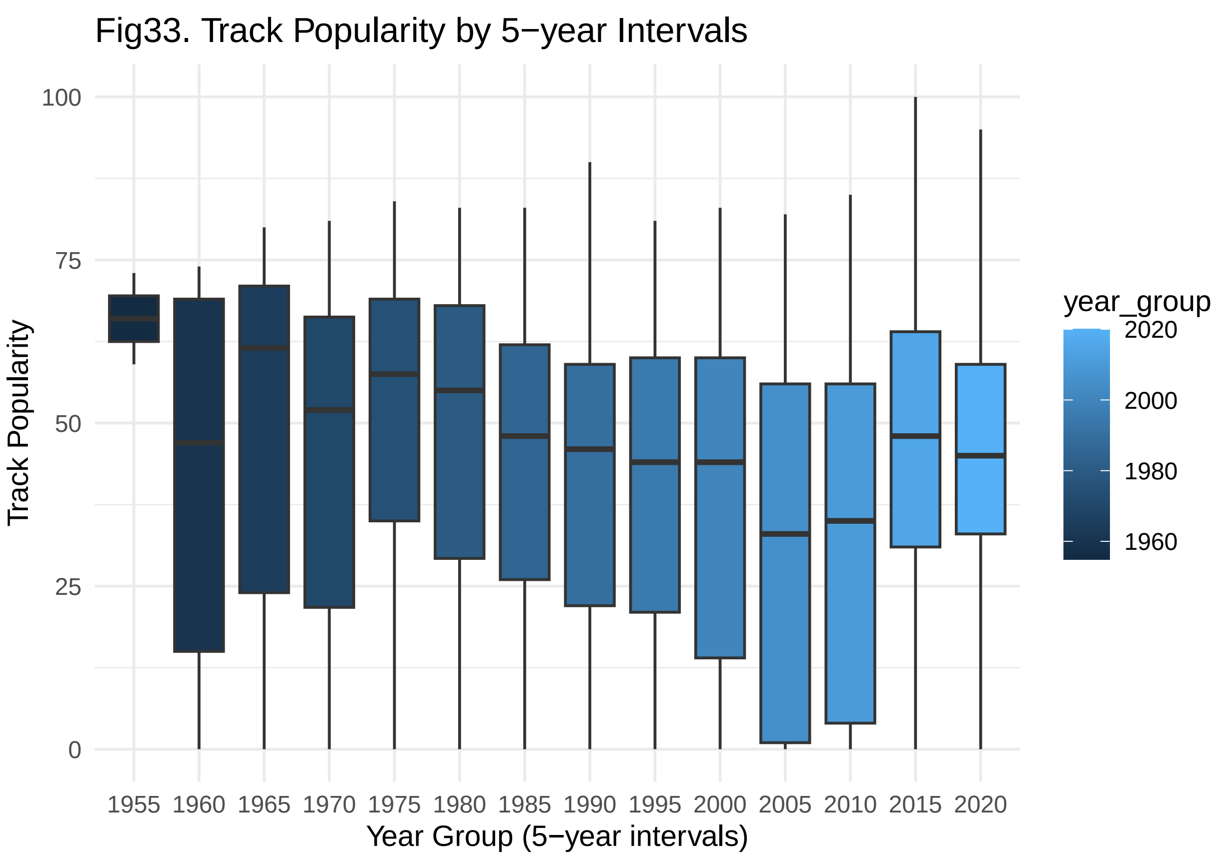



```
##
## track_album_release_year
##      edm    latin      pop      r&b      rap     rock
## 2016.780 2015.176 2014.843 2010.361 2013.411 1999.331
##                   Df  Sum Sq Mean Sq F value Pr(>F)
## playlist_genre     5  918246  183649    2341 <2e-16 ***
## Residuals      30941 2427434      78
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = as.formula(formula_str), data = spotify_songs)
##
## $playlist_genre
##                   diff         lwr         upr     p adj
## latin-edm   -1.6037595  -2.0886385  -1.1188805 0.0000000
## pop-edm     -1.9365664  -2.4128826  -1.4602501 0.0000000
## r&b-edm     -6.4188497  -6.9003124  -5.9373869 0.0000000
## rap-edm     -3.3686033  -3.8410310  -2.8961755 0.0000000
## rock-edm   -17.4487799 -17.9590418 -16.9385181 0.0000000
## pop-latin   -0.3328069  -0.8313173   0.1657034 0.4005061
## r&b-latin   -4.8150902  -5.3185202  -4.3116602 0.0000000
## rap-latin   -1.7648438  -2.2596401  -1.2700475 0.0000000
## rock-latin -15.8450205 -16.3760596 -15.3139813 0.0000000
## r&b-pop     -4.4822833  -4.9774714  -3.9870952 0.0000000
## rap-pop     -1.4320369  -1.9184450  -0.9456288 0.0000000
```

```
## rock-pop   -15.5122135 -16.0354459 -14.9889812 0.0000000
## rap-r&b      3.0502464   2.5587975   3.5416954 0.0000000
## rock-r&b   -11.0299302 -11.5578519 -10.5020086 0.0000000
## rock-rap   -14.0801767 -14.5998716 -13.5604817 0.0000000
##
## [1] "NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>NEXT>>>>"
```

How does track popularity change over time

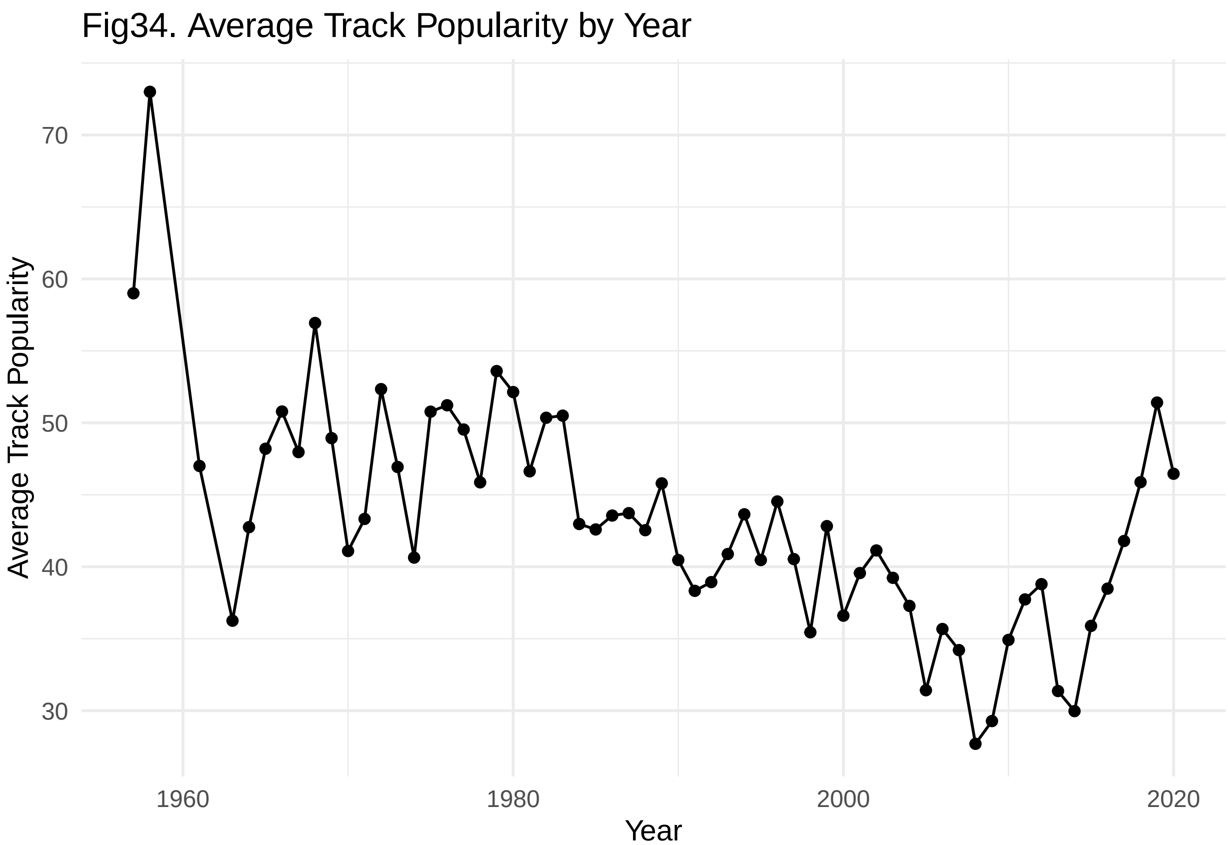
```
spotify_songs <- spotify_songs %>%
  mutate(year_group = 5 * (track_album_release_year %/% 5))

spotify_songs %>%
  ggplot(aes(x = factor(year_group),
             y = track_popularity,
             fill = year_group)) +
  geom_boxplot() +
  xlab("Year Group (5-year intervals)") +
  ylab("Track Popularity") +
  ggtitle("Fig33. Track Popularity by 5-year Intervals") +
  theme_minimal()
```



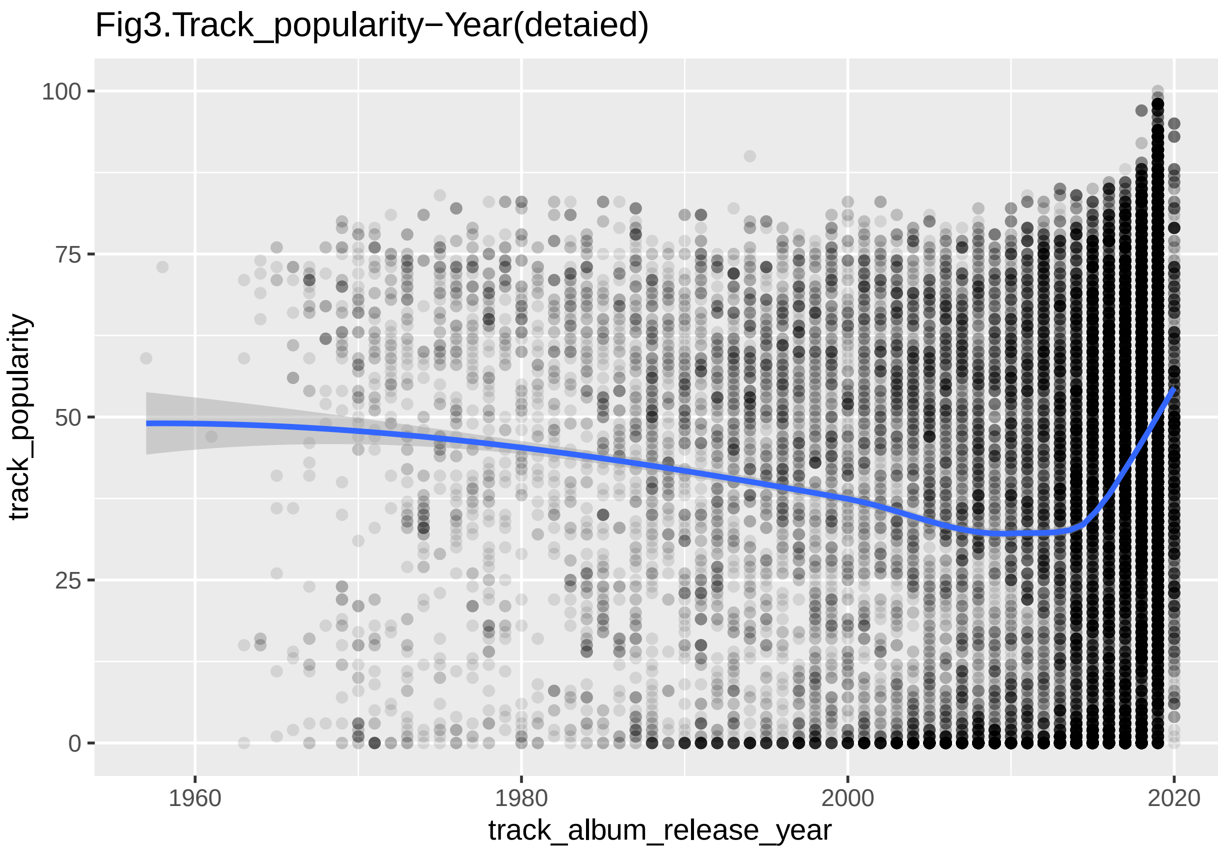

```
spotify_songs <- spotify_songs %>%
  dplyr::select(-year_group)
```

```
spotify_songs %>%
  group_by(track_album_release_year) %>%
  summarize(avg_popularity = mean(track_popularity)) %>%
```

```
ggplot(aes(x = track_album_release_year, y = avg_popularity)) +
  geom_point() +
  geom_line() +
  xlab("Year") +
  ylab("Average Track Popularity") +
  ggtitle("Fig34. Average Track Popularity by Year") +
  theme_minimal()
```




```
spotify_songs %>% ggplot(aes(x = track_album_release_year, y =track_popularity)) +
  geom_point(alpha = 0.1) +
  geom_smooth( formula = y ~ x, method = "loess") +
  ggtitle("Fig3.Track_popularity-Year(detaied)")
```




```r
linear_model <- lm(track_popularity ~track_album_release_year,
                   data = spotify_songs)

summary(linear_model)
```

```
##
## Call:
## lm(formula = track_popularity ~ track_album_release_year, data = spotify_songs)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -43.998 -18.037   2.524  19.364  56.162
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -279.38805   27.39308  -10.20   <2e-16 ***
## track_album_release_year    0.16009    0.01361   11.76   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 24.9 on 30945 degrees of freedom
## Multiple R-squared:  0.004449,   Adjusted R-squared:  0.004417
## F-statistic: 138.3 on 1 and 30945 DF,  p-value: < 2.2e-16
```

PCA

```r
spotify_songs_recipe_PCA <- recipe( playlist_genre ~ ., data = spotify_songs ) %>%
  step_normalize(all_predictors() ) %>% # Normalise our predictors
```

```r
  step_pca(all_predictors() ) # Do the PCA.
spotify_songs_recipe_PCA
```

```
##
## -- Recipe ------------------------------------------------------------------
##
## -- Inputs
## Number of variables by role
## outcome:    1
## predictor: 14
##
## -- Operations
## * Centering and scaling for: all_predictors()
## * PCA extraction with: all_predictors()
```

```r
spotify_songs_prepped <- spotify_songs_recipe_PCA %>% prep()
tidy(spotify_songs_prepped)
```

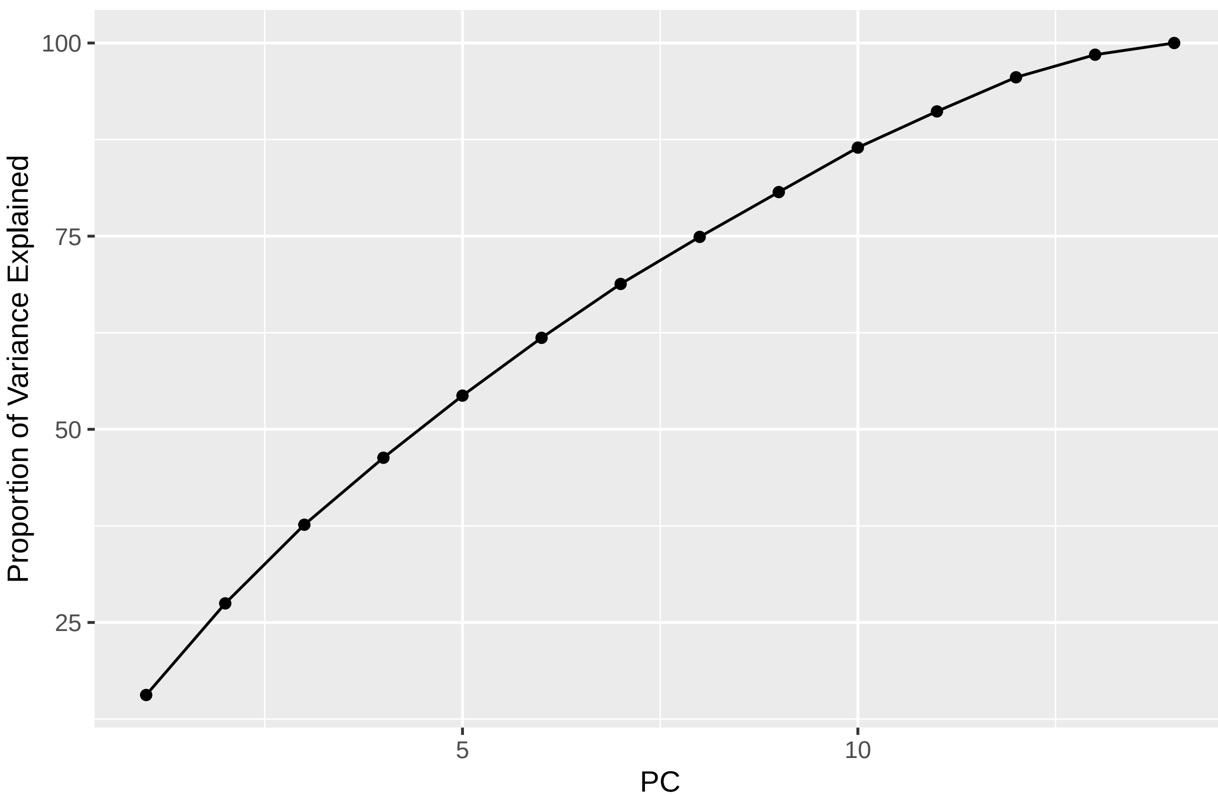
```
## # A tibble: 2 x 6
##   number operation type      trained skip  id
##    <int> <chr>     <chr>     <lgl>   <lgl> <chr>
## 1      1 step      normalize TRUE    FALSE normalize_LQOKm
## 2      2 step      pca       TRUE    FALSE pca_3KvVU
```

```r
sdev <- spotify_songs_prepped$steps[[2]]$res$sdev
ve <- sdev^2 / sum(sdev ^2)

variance_explained <- ve * 100

sorted_var_explained <- sort(variance_explained, decreasing = TRUE)

cumulative_var_explained <- cumsum(sorted_var_explained)
PC_CUM <- data.frame(PC = 1:length(cumulative_var_explained),
                     cumulative_var_explained = cumulative_var_explained)
PC_CUM %>% filter(cumulative_var_explained > 95)
```

```
##   PC cumulative_var_explained
## 1 12                 95.56142
## 2 13                 98.48400
## 3 14                100.00000
```

```r
PC_CUM %>% ggplot(aes(x = PC , y = cumulative_var_explained)) + geom_point() + geom_line() +
  ggtitle("Fig36. Proportion of Total Variance Explained by PC") + xlab("PC") +
  ylab("Proportion of Variance Explained")
```

Fig36. Proportion of Total Variance Explained by PC



1.4 Modeling

Sampling to reduce data set size

```
set.seed(1897402)
songs_per_genre <- 1000

spotify_songs_sampled  <- spotify_songs %>%
  group_by(playlist_genre) %>%
  sample_n(songs_per_genre) %>%
  ungroup()

skim(spotify_songs_sampled)
```

Table 10: Data summary

Name	spotify_songs_sampled
Number of rows	6000
Number of columns	15
Column type frequency:	
factor	1
numeric	14
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
playlist_genre	0	1	FALSE	6	edm: 1000, lat: 1000, pop: 1000, r&b: 1000

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
track_popularity	0	1	42.77	24.69	0.00	25.00	46.00	62.00	98.00	
danceability	0	1	0.65	0.15	0.08	0.56	0.67	0.76	0.97	
energy	0	1	0.70	0.18	0.02	0.58	0.72	0.84	1.00	
key	0	1	5.42	3.63	0.00	2.00	6.00	9.00	11.00	
loudness	0	1	-6.70	3.02	-36.51	-8.17	-6.14	-4.60	0.64	
mode	0	1	0.56	0.50	0.00	0.00	1.00	1.00	1.00	
speechiness	0	1	0.11	0.10	0.02	0.04	0.06	0.13	0.86	
acousticness	0	1	0.18	0.22	0.00	0.02	0.08	0.26	0.98	
instrumentalness	0	1	0.08	0.22	0.00	0.00	0.00	0.00	0.99	
liveness	0	1	0.19	0.15	0.01	0.09	0.13	0.25	1.00	
valence	0	1	0.51	0.23	0.00	0.33	0.52	0.70	0.98	
tempo	0	1	120.79	27.19	52.65	99.90	121.04	134.03	220.25	
duration_ms	0	1	224650.09	59104.81	31893.00	187510.25	214323.50	252233.25	517125.00	
track_album_release_year	0	1	2011.48	11.26	1958.00	2009.00	2016.00	2019.00	2020.00	

Split Train and Test

```
set.seed(1897402)
spotify_songs_split <- initial_split(spotify_songs_sampled,
                                     strata = playlist_genre)
spotify_songs_split
```

```
## <Training/Testing/Total>
## <4500/1500/6000>
```

```
spotify_songs_train <- training(spotify_songs_split)
spotify_songs_test <- testing(spotify_songs_split)
head(spotify_songs_train)
```

```
## # A tibble: 6 x 15
##   track_pop~1 playl~2 dance~3 energy   key loudn~4  mode speec~5 acous~6 instr~7
##         <dbl> <fct>     <dbl>  <dbl> <dbl>   <dbl> <dbl>   <dbl>   <dbl>   <dbl>
## 1          29 edm       0.478  0.904     0  -5.41      0  0.0648 0.00221 0
## 2          48 edm       0.355  0.993     5  -0.627     0  0.221  0.0108  1.13e-1
## 3          71 edm       0.687  0.707     4  -6.19      1  0.0328 0.0536  9.29e-5
## 4          55 edm       0.891  0.794     3  -3.44      0  0.0526 0.383   1.99e-4
## 5          46 edm       0.755  0.83      8  -6.34      0  0.124  0.104   6.6 e-1
## 6          37 edm       0.893  0.748     7  -5.60      0  0.051  0.311   2.59e-5
## # ... with 5 more variables: liveness <dbl>, valence <dbl>, tempo <dbl>,
## #   duration_ms <dbl>, track_album_release_year <dbl>, and abbreviated variable
## #   names 1: track_popularity, 2: playlist_genre, 3: danceability, 4: loudness,
## #   5: speechiness, 6: acousticness, 7: instrumentalness
```

```r
spotify_songs_train %>% count(playlist_genre) %>%
  mutate(percent = prop.table(n) * 100)
```

```
## # A tibble: 6 x 3
##   playlist_genre     n percent
##   <fct>          <int>   <dbl>
## 1 edm              750    16.7
## 2 latin            750    16.7
## 3 pop              750    16.7
## 4 r&b              750    16.7
## 5 rap              750    16.7
## 6 rock             750    16.7
```

```r
spotify_songs_test %>% count(playlist_genre)  %>%
  mutate(percent = prop.table(n) * 100)
```

```
## # A tibble: 6 x 3
##   playlist_genre     n percent
##   <fct>          <int>   <dbl>
## 1 edm              250    16.7
## 2 latin            250    16.7
## 3 pop              250    16.7
## 4 r&b              250    16.7
## 5 rap              250    16.7
## 6 rock             250    16.7
```

Preprocessing

```r
spotify_songs_rcp <- recipe(playlist_genre ~ ., data = spotify_songs_train) %>%
  step_zv(all_predictors()) %>%
  step_normalize(all_predictors()) %>%
  step_corr(all_predictors()) %>%
  prep()

spotify_songs_rcp
```

```
##

## -- Recipe ----------------------------------------------------------------------

##

## -- Inputs

## Number of variables by role

## outcome:    1
## predictor: 14

##

## -- Training information

## Training data contained 4500 data points and no incomplete rows.

##

## -- Operations
```

```
## * Zero variance filter removed: <none> | Trained

## * Centering and scaling for: track_popularity, danceability, ... | Trained

## * Correlation filter on: <none> | Trained
```

```
spotify_train_prep <- juice(spotify_songs_rcp)

spotify_train_prep %>% head()
```

```
## # A tibble: 6 x 15
##   track_~1 dance~2 energy     key loudn~3   mode speec~4 acous~5 instr~6 liven~7
##      <dbl>   <dbl>  <dbl>   <dbl>   <dbl>  <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1   -0.554  -1.16  1.13   -1.47     0.428 -1.14   -0.417  -0.795  -0.366   1.42
## 2    0.214  -1.99  1.61   -0.0991   2.00  -1.14    1.10   -0.757   0.144   0.996
## 3    1.14    0.244 0.0521 -0.374    0.172  0.878  -0.727  -0.567  -0.365  -0.193
## 4    0.496   1.62  0.527  -0.649    1.08  -1.14   -0.535   0.893  -0.365  -0.707
## 5    0.133   0.701 0.724   0.725    0.122 -1.14    0.156  -0.344   2.61   -0.667
## 6   -0.231   1.63  0.276   0.451    0.367 -1.14   -0.551   0.574  -0.365   1.17
## # ... with 5 more variables: valence <dbl>, tempo <dbl>, duration_ms <dbl>,
## #   track_album_release_year <dbl>, playlist_genre <fct>, and abbreviated
## #   variable names 1: track_popularity, 2: danceability, 3: loudness,
## #   4: speechiness, 5: acousticness, 6: instrumentalness, 7: liveness
```

```
spotify_test_prep <- bake(spotify_songs_rcp, new_data = spotify_songs_test)
spotify_test_prep %>% head()
```

```
## # A tibble: 6 x 15
##   track_~1 dance~2 energy     key loudn~3   mode speec~4 acous~5 instr~6 liven~7
##      <dbl>   <dbl>  <dbl>   <dbl>   <dbl>  <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1   0.577   -0.570  0.380  0.176   0.185  -1.14   -0.253  -0.335  -0.366  -0.673
## 2  -0.554    0.876  1.06  -0.374   0.0944 -1.14    0.602  -0.633   0.932  -0.671
## 3   0.0923   0.607  0.467 -0.374   0.632   0.878  -0.643  -0.739  -0.366   0.730
## 4  -1.08     0.439  1.52   1.55    1.07   -1.14    0.602  -0.613  -0.197   1.24
## 5  -1.64     0.432  0.560 -1.20    0.403  -1.14   -0.507  -0.759  -0.366   0.840
## 6   1.14    -2.14   1.57  -0.0991  1.27   -1.14   -0.500  -0.660   3.10    0.236
## # ... with 5 more variables: valence <dbl>, tempo <dbl>, duration_ms <dbl>,
## #   track_album_release_year <dbl>, playlist_genre <fct>, and abbreviated
## #   variable names 1: track_popularity, 2: danceability, 3: loudness,
## #   4: speechiness, 5: acousticness, 6: instrumentalness, 7: liveness
```

```
set.seed(1897402)
spotify_cv <- vfold_cv(
 data = spotify_train_prep,
 v = 5,
 strata = playlist_genre)

spotify_cv %>%
  slice( 1 ) %>%
 pull( splits )
```

```
## [[1]]
## <Analysis/Assess/Total>
## <3600/900/4500>
```

```
set.seed(1897402)
spotify_bootstrap <- bootstraps(spotify_train_prep, times = 10,  strata = playlist_genre )
```

###LDA

```r
lda_model <- discrim_linear(mode = "classification") %>%
  set_engine("MASS")

spotify_resamples <- fit_resamples(
  object = lda_model,
  preprocessor = recipe(playlist_genre ~ ., data = spotify_train_prep),
  resamples = spotify_bootstrap,
  control = control_resamples(save_pred = T)
)

prediction_lda <- spotify_resamples %>% collect_predictions()
metrics_lda <- spotify_resamples %>% collect_metrics()

lda_10cfm <- prediction_lda %>%
  group_by(id) %>%
  conf_mat(truth = playlist_genre, estimate = .pred_class) %>%
  pull(conf_mat)

lda_boost_result <- spotify_resamples %>%
  collect_predictions() %>%
  group_by(id)

confusionMatrix(lda_boost_result$.pred_class, as_factor(lda_boost_result$playlist_genre))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  edm latin  pop  r&b  rap rock
##      edm   1550   277  357  103  288  191
##      latin  324  1282  554  381  428  123
##      pop    561   434 1139  413  246  456
##      r&b    114   240  325  869  325  191
##      rap    183   365  162  634 1405   23
##      rock    18   111  205  357   81 1798
##
## Overall Statistics
##
##                Accuracy : 0.4871
##                  95% CI : (0.4794, 0.4947)
##     No Information Rate : 0.1685
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3846
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: edm Class: latin Class: pop Class: r&b Class: rap
## Sensitivity             0.56364      0.47324    0.41539    0.31520    0.50667
## Specificity             0.91165      0.86888    0.84678    0.91313    0.90051
## Pos Pred Value          0.56038      0.41462    0.35057    0.42103    0.50685
## Neg Pred Value          0.91271      0.89367    0.87915    0.86933    0.90044
```

```
## Prevalence                 0.16654      0.16405     0.16605     0.16696     0.16793
## Detection Rate             0.09387      0.07764     0.06898     0.05263     0.08508
## Detection Prevalence       0.16750      0.18725     0.19675     0.12499     0.16787
## Balanced Accuracy          0.73764      0.67106     0.63108     0.61416     0.70359
##                         Class: rock
## Sensitivity                  0.6463
## Specificity                  0.9438
## Pos Pred Value               0.6996
## Neg Pred Value               0.9294
## Prevalence                   0.1685
## Detection Rate               0.1089
## Detection Prevalence         0.1556
## Balanced Accuracy            0.7950
```

KNN

```r
knn_model <- nearest_neighbor(
  mode = "classification",
  neighbors = tune()
) %>%
  set_engine("kknn")

k_grid <- grid_regular(
  levels = 20,
  neighbors(range = c(1, 100))) %>%
  as_tibble()

spotify_knn_tune <- tune_grid(
  object = knn_model,
  resamples = spotify_bootstrap,
  grid = k_grid,
  preprocessor = recipe(playlist_genre ~., data = spotify_train_prep),
)


best.k <- spotify_knn_tune %>%
  select_best("accuracy")

best.k
```

```
## # A tibble: 1 x 2
##   neighbors .config
##       <int> <chr>
## 1        89 Preprocessor1_Model18
```

```r
knn_model_best <-
  nearest_neighbor(mode = "classification",
                   neighbors = best.k$neighbors) %>%
  set_engine( "kknn" )

knn_model_best
```

```
## K-Nearest Neighbor Model Specification (classification)
##
## Main Arguments:
```

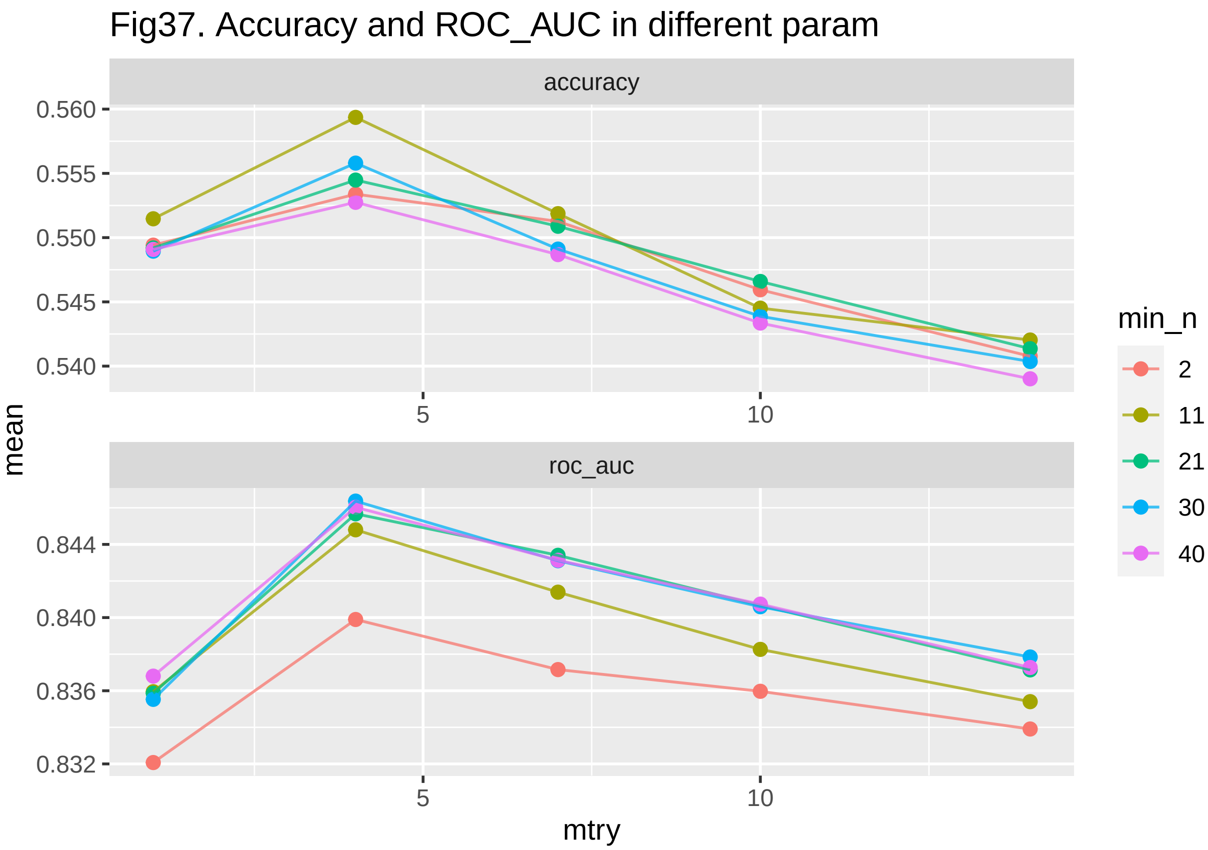
```
##   neighbors = best.k$neighbors
##
## Computational engine: kknn
```

```
spotify_knn_tune %>% collect_metrics()
```

```
## # A tibble: 40 x 7
##    neighbors .metric  .estimator  mean     n std_err .config
##        <int> <chr>    <chr>      <dbl> <int>   <dbl> <chr>
##  1         1 accuracy multiclass 0.417    10 0.00317 Preprocessor1_Model01
##  2         1 roc_auc  hand_till  0.650    10 0.00188 Preprocessor1_Model01
##  3         6 accuracy multiclass 0.432    10 0.00321 Preprocessor1_Model02
##  4         6 roc_auc  hand_till  0.742    10 0.00226 Preprocessor1_Model02
##  5        11 accuracy multiclass 0.451    10 0.00382 Preprocessor1_Model03
##  6        11 roc_auc  hand_till  0.764    10 0.00275 Preprocessor1_Model03
##  7        16 accuracy multiclass 0.462    10 0.00432 Preprocessor1_Model04
##  8        16 roc_auc  hand_till  0.776    10 0.00290 Preprocessor1_Model04
##  9        21 accuracy multiclass 0.471    10 0.00444 Preprocessor1_Model05
## 10        21 roc_auc  hand_till  0.783    10 0.00283 Preprocessor1_Model05
## # ... with 30 more rows
```

RF

```
rf_spec <- rand_forest(mode = "classification",
                       trees = 100,
                       mtry = tune(),
                       min_n = tune()) %>%
  set_engine("ranger", importance = "permutation")

rf_grid <- grid_regular(
  finalize( mtry(),
            spotify_train_prep%>%
              dplyr::select( -playlist_genre ) ),
  min_n(),
  levels = 5 )

set.seed(1897402)
doParallel::registerDoParallel() # This makes macs run a little faster
spotify_rf_tune <- tune_grid(object = rf_spec,
                             preprocessor = recipe(playlist_genre ~ ., data = spotify_train_prep),
                             resamples = spotify_bootstrap,
                             grid = rf_grid)

spotify_rf_tune %>%
 collect_metrics() %>%
  mutate( min_n = as.factor( min_n ) ) %>%
  ggplot( aes( x = mtry, y = mean, colour = min_n ) ) +
  geom_point( size = 2 ) +
  geom_line( alpha = 0.75 ) +
  ggtitle("Fig37. Accuracy and ROC_AUC in different param")+
  facet_wrap( ~ .metric, scales = "free", nrow = 3 )
```




```r
best_rf_acc <- select_best(spotify_rf_tune, "accuracy" )
best_rf_acc
```

```
## # A tibble: 1 x 3
##    mtry min_n .config
##   <int> <int> <chr>
## 1     4    11 Preprocessor1_Model07
```

```r
rf_model_best <- finalize_model( rf_spec, best_rf_acc )
rf_model_best
```

```
## Random Forest Model Specification (classification)
##
## Main Arguments:
##   mtry = 4
##   trees = 100
##   min_n = 11
##
## Engine-Specific Arguments:
##   importance = permutation
##
## Computational engine: ranger
```

1.5 Model Evaluation Method

standard model build function

```r
model_build_function <- function(model) {
  model.cv <- fit_resamples(
    object = model,
    preprocessor = recipe(playlist_genre ~ . ,
                          data = spotify_train_prep),
    resamples = spotify_cv,
    control = control_resamples(save_pred = T))

  model.metrics <- model.cv %>%
    collect_metrics()

  model.prediction <- model.cv %>%
    collect_predictions()

  return(list(metrics = model.metrics, prediction = model.prediction))
}
```

```r
genre = c(".pred_edm",".pred_latin",".pred_pop",".pred_r&b",".pred_rap",".pred_rock" )
genrename = c("edm","latin","pop","r&b","rap","rock" )

ROC_plot <- function(prediction,i,idx,model_name){

  index = idx+i
  predictions <-
  prediction %>%
  mutate(.pred_other = 1 - across(genre[i])) %>%
  mutate(playlist_genre = ifelse(playlist_genre == genrename[i],genrename[i], "other")) %>%
  mutate(playlist_genre = factor(playlist_genre, levels = c(genrename[i], "other"))) %>%
  mutate(.pred_class = ifelse(.pred_class == genrename[i],genrename[i],"other"))

  plot <- predictions %>% group_by( id ) %>%
  roc_curve( truth = as_factor(playlist_genre), estimate = genre[i] ) %>%
    ggplot(aes(x= 1-specificity, y = sensitivity)) + geom_point(alpha = 0.2, color = "red") +
    ggtitle(paste("Fig",index,". ROC curve of ", genrename[i],"for ", model_name)) + theme_minimal()
 return(plot)
}

AUC_value <- function(prediction,i){

  predictions <-
  prediction %>%
  mutate(.pred_other = 1 - across(genre[i])) %>%
  mutate(playlist_genre = ifelse(playlist_genre == genrename[i],genrename[i], "other")) %>%
  mutate(playlist_genre = factor(playlist_genre, levels = c(genrename[i], "other"))) %>%
  mutate(.pred_class = ifelse(.pred_class == genrename[i],genrename[i],"other"))

  roc_data <- predictions %>% group_by( id ) %>%
  roc_curve( truth = as_factor(playlist_genre), estimate = genre[i])
 return(roc_data)
}
```

LDA

```
lda.cv.model <- model_build_function(lda_model)
lda.prediction  <- lda.cv.model$prediction

lda.cv.model$metrics
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy multiclass 0.489     5 0.00338 Preprocessor1_Model1
## 2 roc_auc  hand_till  0.806     5 0.00204 Preprocessor1_Model1
```

```
lda.cfm <- confusionMatrix(lda.prediction$.pred_class, as_factor(lda.prediction$playlist_genre))
lda.cfm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction edm latin pop r&b rap rock
##      edm   418    72  94  31  78   53
##      latin  87   360 147 103 108   31
##      pop   159   120 313 109  69  123
##      r&b    30    64  93 238  82   53
##      rap    52   102  45 171 389    6
##      rock    4    32  58  98  24  484
##
## Overall Statistics
##
##                Accuracy : 0.4893
##                  95% CI : (0.4746, 0.5041)
##     No Information Rate : 0.1667
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3872
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: edm Class: latin Class: pop Class: r&b Class: rap
## Sensitivity             0.55733       0.4800    0.41733    0.31733    0.51867
## Specificity             0.91253       0.8731    0.84533    0.91413    0.89973
## Pos Pred Value          0.56032       0.4306    0.35050    0.42500    0.50850
## Neg Pred Value          0.91156       0.8936    0.87885    0.87005    0.90335
## Prevalence              0.16667       0.1667    0.16667    0.16667    0.16667
## Detection Rate          0.09289       0.0800    0.06956    0.05289    0.08644
## Detection Prevalence    0.16578       0.1858    0.19844    0.12444    0.17000
## Balanced Accuracy       0.73493       0.6765    0.63133    0.61573    0.70920
##                      Class: rock
## Sensitivity               0.6453
## Specificity               0.9424
## Pos Pred Value            0.6914
## Neg Pred Value            0.9300
## Prevalence                0.1667
```