analysis

```
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   2.1.1     v forcats 0.5.1

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

```
library(nnet)
library(broom)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(RColorBrewer)
library(knitr)
library(patchwork)
```

RESEARCH QUESTION/DESCRIPTION OF DATA AND RESPONSE:

Humans can infer the genre of a song based on its audio features. That is, it is pretty easy for humans to understand the difference between a rock song and a classical song. Thus there must be some relationship between the audio features of a song and its genre. For computers, this task is much more trivial because it requires inferring the human perceptible audio features of a song based on a very high dimensional space (audio files are represented by long lists of numbers in computers). Spotify has algorithms which are designed to quantify human perceptible audio features.

This statistical analysis is designed to model and predict the genre of a Spotify track (response) based on its audio feature values (predictors) that are determined by Spotify algorithms. Based on the assumption that a songs genre is defined by its human perceptible audio features, if we find evidence that a relationship exists between Spotify's audio features and a genre, then this would be evidence to support that Spotify's audio features capture the human perception of audio.

An important note is that we cannot assume all human perceptible musical features are dependent on genre. This project assumes some of the audio features included in the dataset are measures of human percptible musical features which are dependednt on genres.

Response: The response is a categorical variable describing the genre of a track. The genre of a track is determined by Spotify and was retreived through the Spotify API for this dataset. The genres of this dataset include classical, electronic dance music (edm_dance), country, hip-hop, metal, punk, pop, and rock. Each track in the data set is associated with one genre through the variable “genre”. It is important to note that these are not all the genres of Spotify tracks. Spotify only allows an API user to retreive a subset of the genres. I selected these genres based on what I determined to be “popular” and mainstream.

Predictor variables: The predictor varibles are features desrcribing different audio characteristics of a song. These audo characteristic features of a track are determined by spotify and was retreived through the Spotify API for this dataset. The audio features in this dataset include danceability, energy, key, loudness, mode, speechiness, acousticness, intrumentalness, liveness, valence, tempo, track duration (duration_ms), and time signature (time_signature). These are the varaibles which will be used as predictor variables for a track genre. For more detailed information on the predictors, see the “README” file in the data folder. It would be very useful to understand more about how these audio feature measures were calculated and the techincal details toSpotify’s algorithms but I could not find any documentation on this.

EXPLORITY DATA ANALYSIS

```
track_data <- read.csv('data/spotify_tracksV2.csv')
track_data <- track_data %>% mutate(duration_ms = duration_ms / 1000) %>% filter_all(any_vars(! is.na(.
track_data <- rename(track_data,duration_sec = duration_ms)
```

First we parse the data and do a bit of cleaning. We want to convert the duration from milliseconds to seconds so it is more readable. We also want to remove any observations with missing values. An observation may have missing values if the API response timed out during the data scraping. Therfor I determined that a missing value for an observation is indepednent from any of the variables in the dataset and can be safely removed from the dataset.

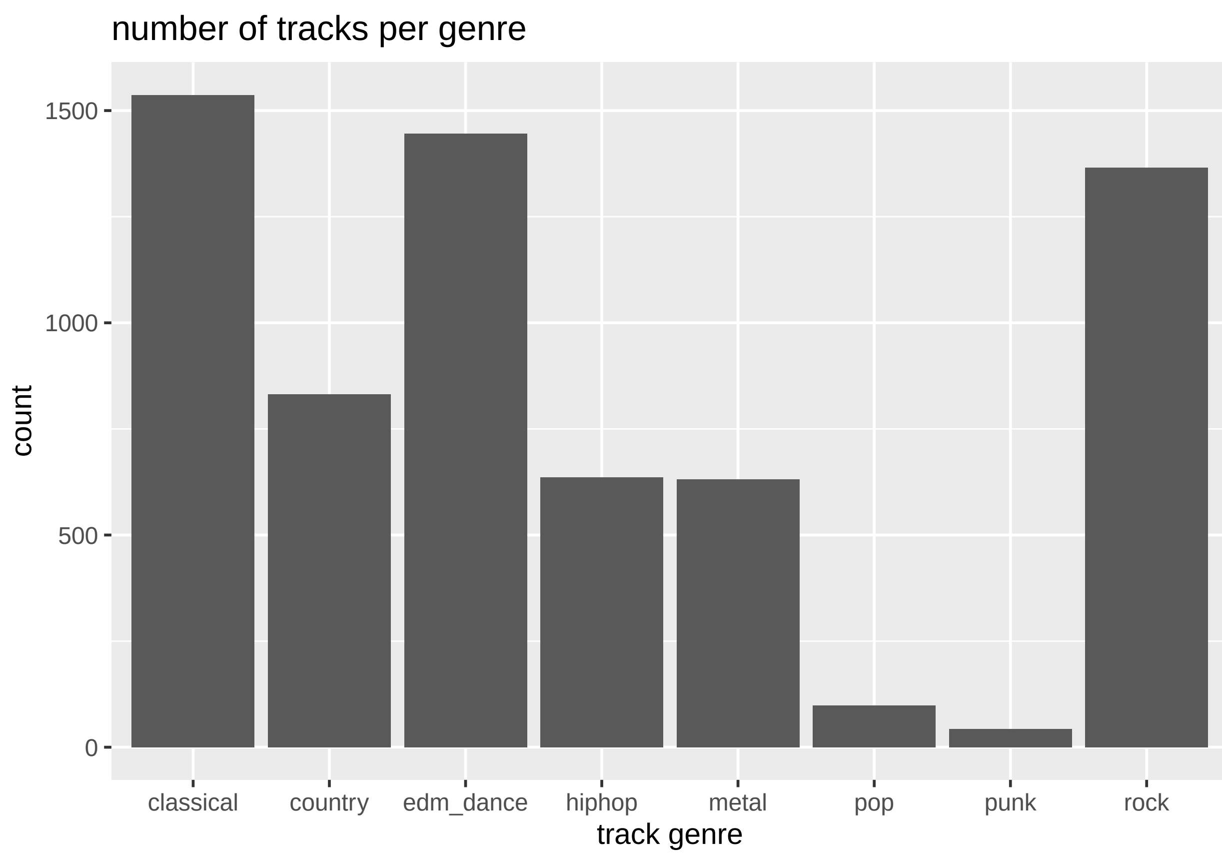
data_fill_fac is just mutating to factor variables.

```
data_fill_fac <- track_data %>% filter_all(any_vars(! is.na(.))) %>% mutate(genre = as.factor(genre)) %
```

EDA RESPONSE VARIABLE:

Next we plot the distribution of the resposne, the track genre.

```
p_genre <- ggplot(track_data, aes(x=genre)) + geom_bar() + labs (y = "count", x = "track genre",
                                                                title = "number of tracks per genre")
p_genre
```

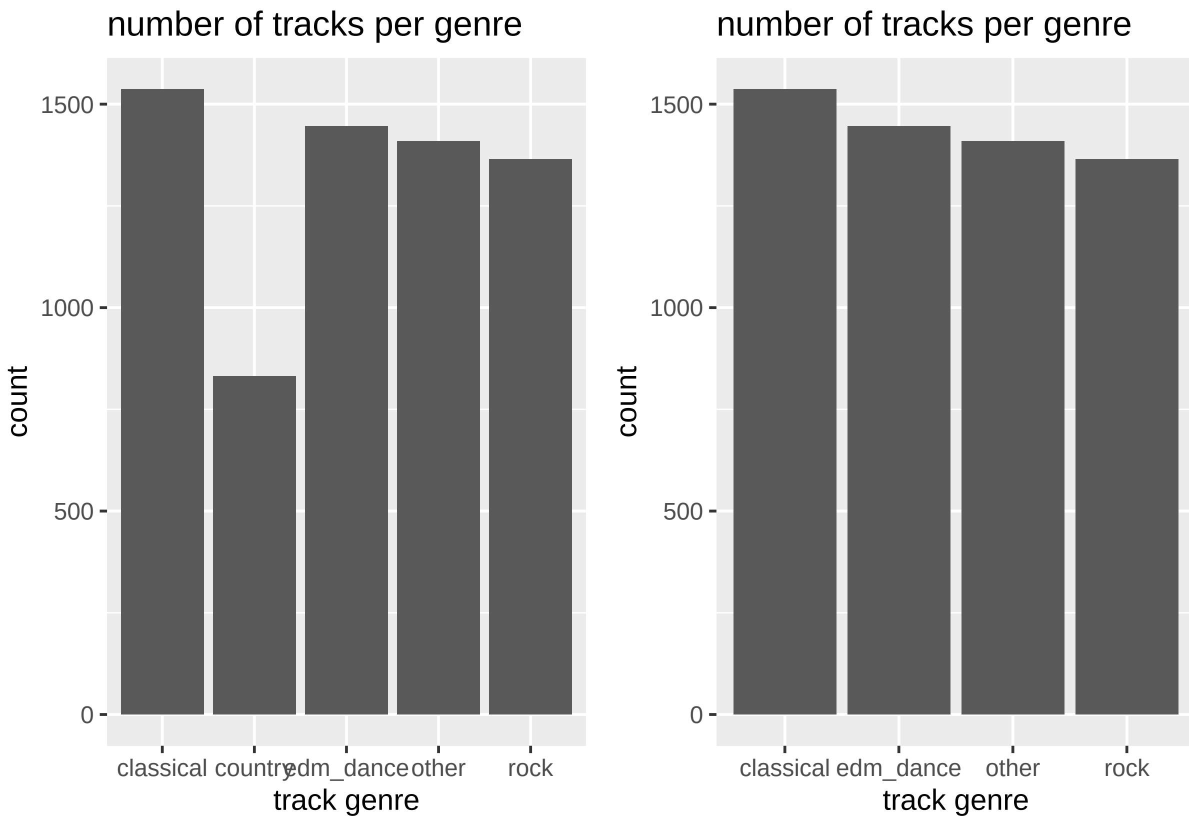


We can observe that there is a relatively high amount of observations with the genre rock, classical, edm_dance. Obervations with country, hip-hop and metal genres, have a smaller amount of observations and observations with pop and punk genres have very few observations compared to the rest. This would pose an issue when conducting a multinomial analsysis because there is a lot of variation in the genre counts. Particularly, the model would have a difficult time predicting pop and punk genres.

Furthermore, having 8 response categories is generally poor practice for multinomial modeling as multinomial modeling is not complex enough to perform accurate predictions for more than 5 response categories.

Because of this, we should mutate our response varaible to have 5 or less more equally distributed categories.

```
#g1 combines hiphop, metal, pop, and punk
g1 <- track_data %>% mutate(genre = ifelse(genre == "hiphop", "other", ifelse(genre == "metal", "other"

#g2 is the same as g1 but without country
g2 <- g1 %>% filter(genre != "country")



p_g1 <- ggplot(g1, aes(x=genre)) + geom_bar() + labs (y = "count", x = "track genre",
                                                        title = "number of tracks per genre")

p_g2 <- ggplot(g2, aes(x=genre)) + geom_bar() + labs (y = "count", x = "track genre",
                                                        title = "number of tracks per genre")
p_g1 + p_g2
```



The choose to create two more dataframes for seperate analysis. The first one called "g1" in the code does not change observations with classical, country, edm_dance, or rock genres but combines obsevations with all other genres into a new category for genre called "other". We can observe that this dataframe has a much more equal distribution of genres and only has 5 categories. This is a much better response for a multinomial model. However, we can still observe that there is still much less observations with country as the genre.

In the second dataframe called "g2" in the code we drop observations with country genre to make an equal distribution of genre counts. We can observe from the bar graph that this creates a very even amount of genres across the dataset. However, this choice is questionable because we are loosing information by dropping these values. Therefor I will first conduct analysis with "g1" and only consider using "g2" if I find that the multinomial model of "g1" is failing due to the lack of observations with country as the genre. I may also create another data frame like "g2" only instead of dropping country I add it to the "other" genre. This may be a better choice because it avoids loosing information.

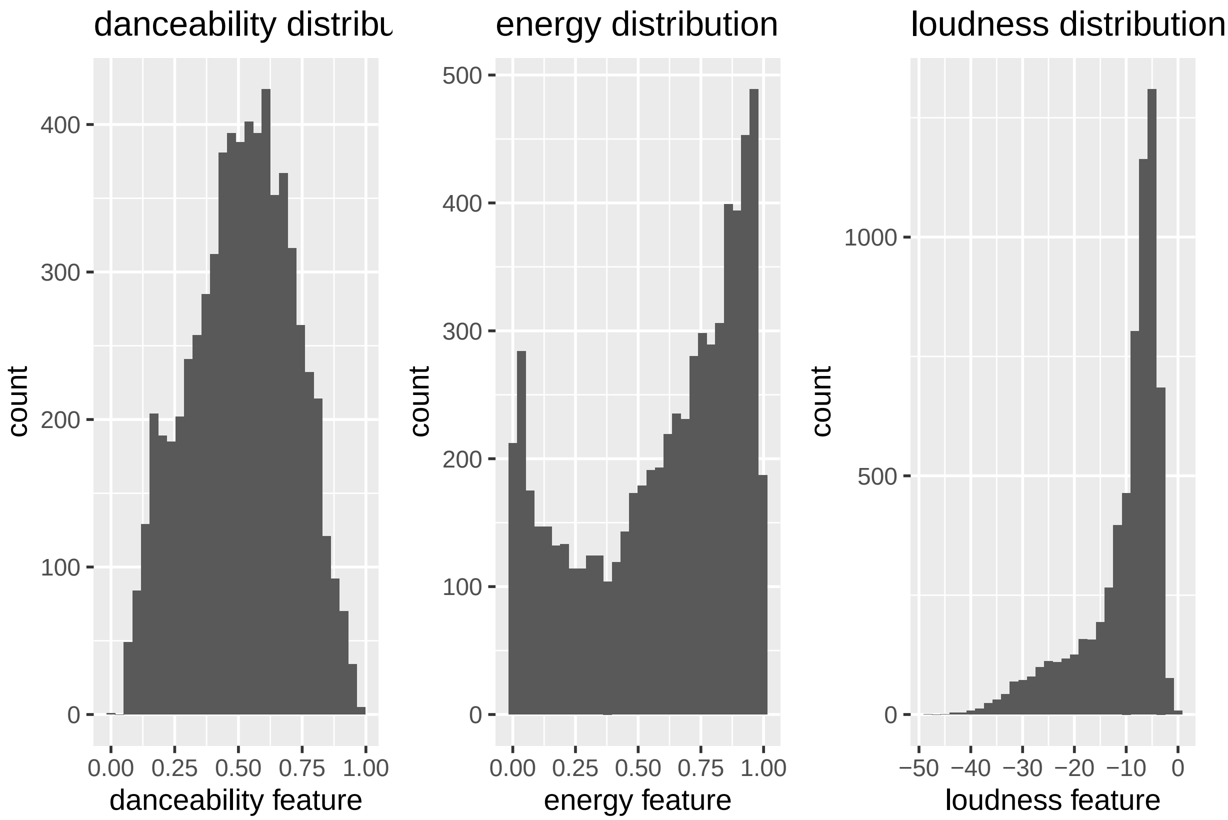
EDA PREDICTOR VARIABLES:

Lets plot all continuous predictor variables.

```
p_dance <- ggplot(data_fill_fac, aes(x=danceability)) + geom_histogram() +
  labs(y = "count", x = "danceability feature",
       title = "danceability distribution")

p_energy <- ggplot(data_fill_fac, aes(x=energy)) + geom_histogram() +
  labs(y = "count", x = "energy feature",
       title = "energy distribution")

p_loud <- ggplot(data_fill_fac, aes(x=loudness)) + geom_histogram() +
```

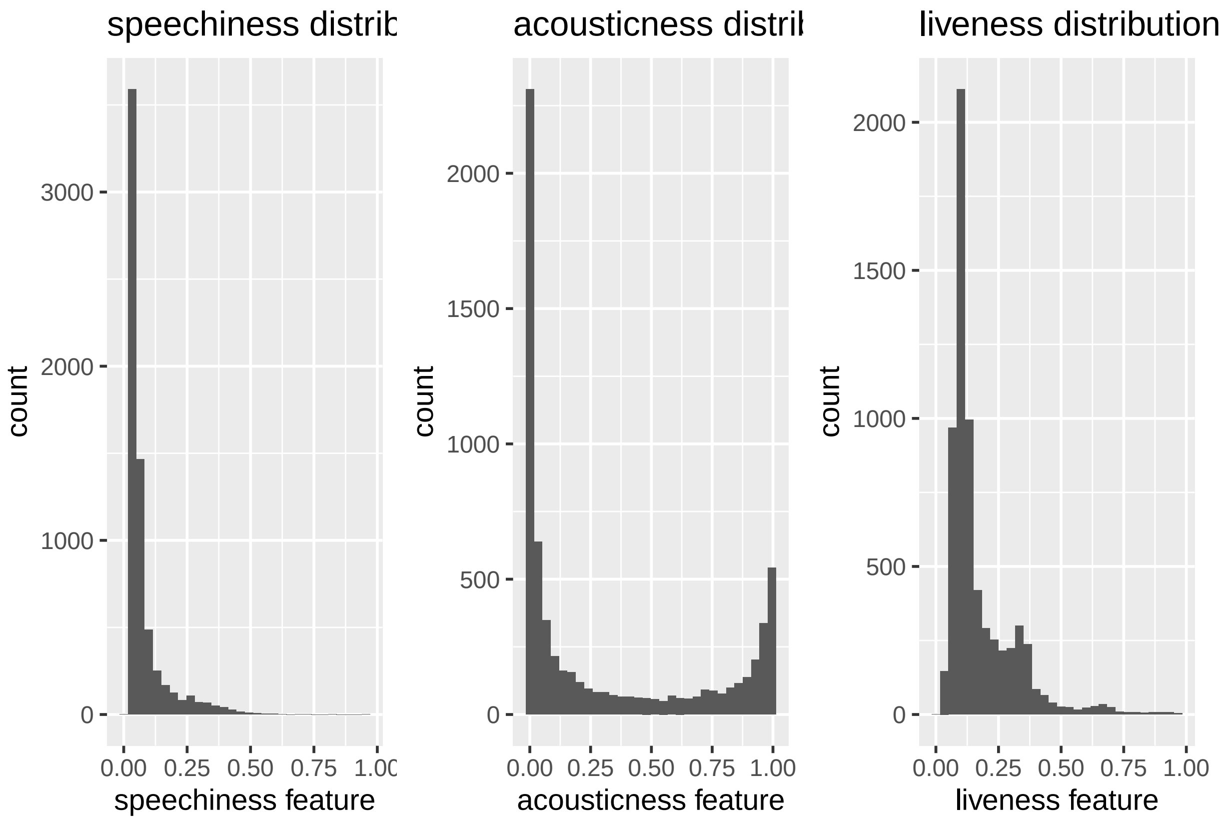
```r
  labs(y = "count", x = "loudness feature",
       title = "loudness distribution")

p_speech <- ggplot(data_fill_fac, aes(x=speechiness)) + geom_histogram() +
  labs(y = "count", x = "speechiness feature",
       title = "speechiness distribution")

p_acoustic <- ggplot(data_fill_fac, aes(x=acousticness)) + geom_histogram() +
  labs(y = "count", x = "acousticness feature",
       title = "acousticness distribution")

p_live <- ggplot(data_fill_fac, aes(x=liveness)) + geom_histogram() +
  labs(y = "count", x = "liveness feature",
       title = "liveness distribution")

p_valence <- ggplot(data_fill_fac, aes(x=valence)) + geom_histogram() +
  labs(y = "count", x = "valence feature",
       title = "valence distribution")

p_tempo <- ggplot(data_fill_fac, aes(x=tempo)) + geom_histogram() +
  labs(y = "count", x = "track tempo",
       title = "tempo distribution")

p_duration <- ggplot(data_fill_fac, aes(x=duration_sec)) + geom_histogram() +
  labs(y = "count", x = "track duration (MS)",
       title = "duration distribution")

p_dance + p_energy + p_loud
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



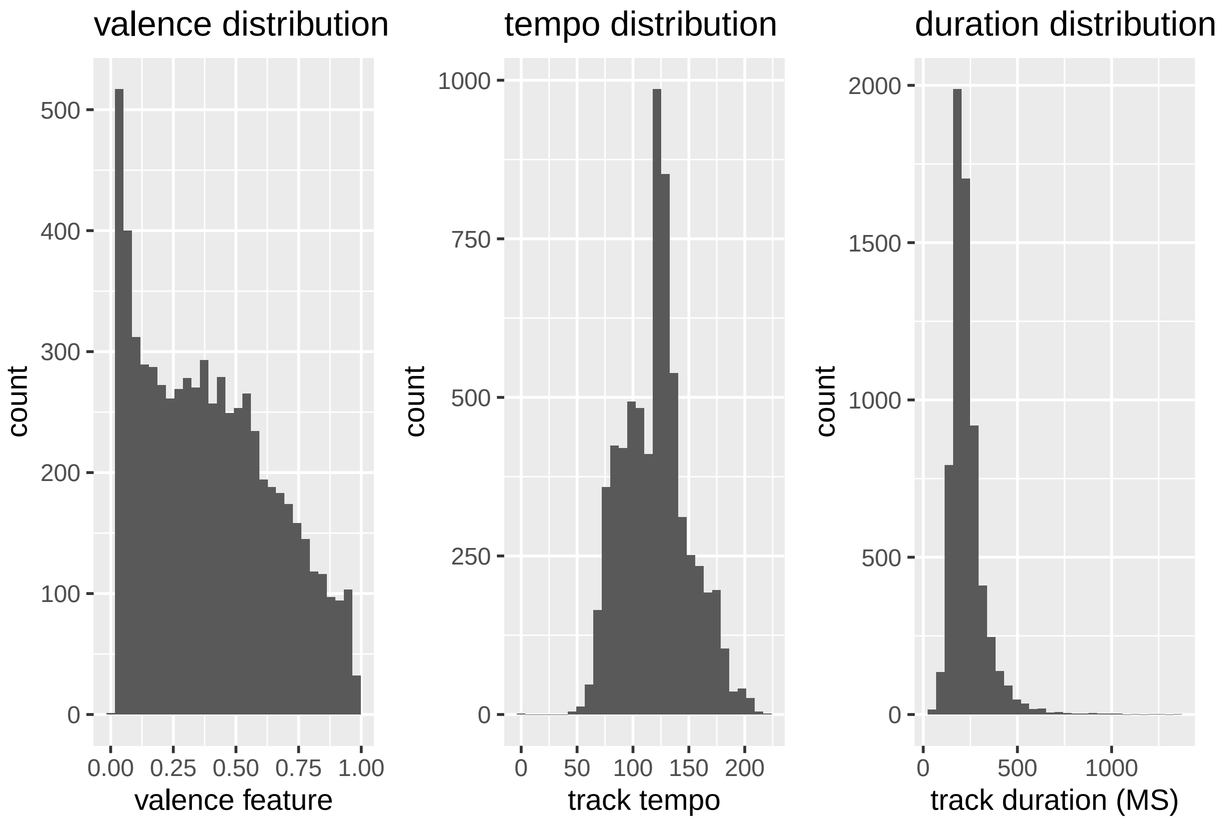

```
p_speech + p_acoustic + p_live
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```




```
p_valence + p_tempo + p_duration
```

```
## 'stat_bin()' using 'bins = 30'. Pick better value with 'binwidth'.
## 'stat_bin()' using 'bins = 30'. Pick better value with 'binwidth'.
## 'stat_bin()' using 'bins = 30'. Pick better value with 'binwidth'.
```




```
summary(data_fill_fac)
```

```
##   track.name        artists.names      artists.ids        track.popularity
##  Length:6588        Length:6588        Length:6588        Min.   :  0.00
##  Class :character   Class :character   Class :character   1st Qu.: 34.00
##  Mode  :character   Mode  :character   Mode  :character   Median : 47.00
##                                                           Mean   : 44.41
##                                                           3rd Qu.: 59.00
##                                                           Max.   :100.00
##
##   artist.ids          track.id         release.date        danceability
##  Length:6588        Length:6588        Length:6588        Min.   :0.0000
##  Class :character   Class :character   Class :character   1st Qu.:0.3700
##  Mode  :character   Mode  :character   Mode  :character   Median :0.5240
##                                                           Mean   :0.5131
##                                                           3rd Qu.:0.6660
##                                                           Max.   :0.9810
##
##      energy             key         loudness       mode      speechiness
##  Min.   :0.00086   2      : 753   Min.   :-47.917   0:2442   Min.   :0.0000
##  1st Qu.:0.34200   7      : 748   1st Qu.:-12.705   1:4146   1st Qu.:0.0366
##  Median :0.67550   0      : 663   Median : -7.566            Median :0.0466
##  Mean   :0.59545   1      : 663   Mean   :-10.429            Mean   :0.0788
##  3rd Qu.:0.86700   9      : 639   3rd Qu.: -5.314            3rd Qu.:0.0775
##  Max.   :0.99800   4      : 577   Max.   :  0.352            Max.   :0.9550
```

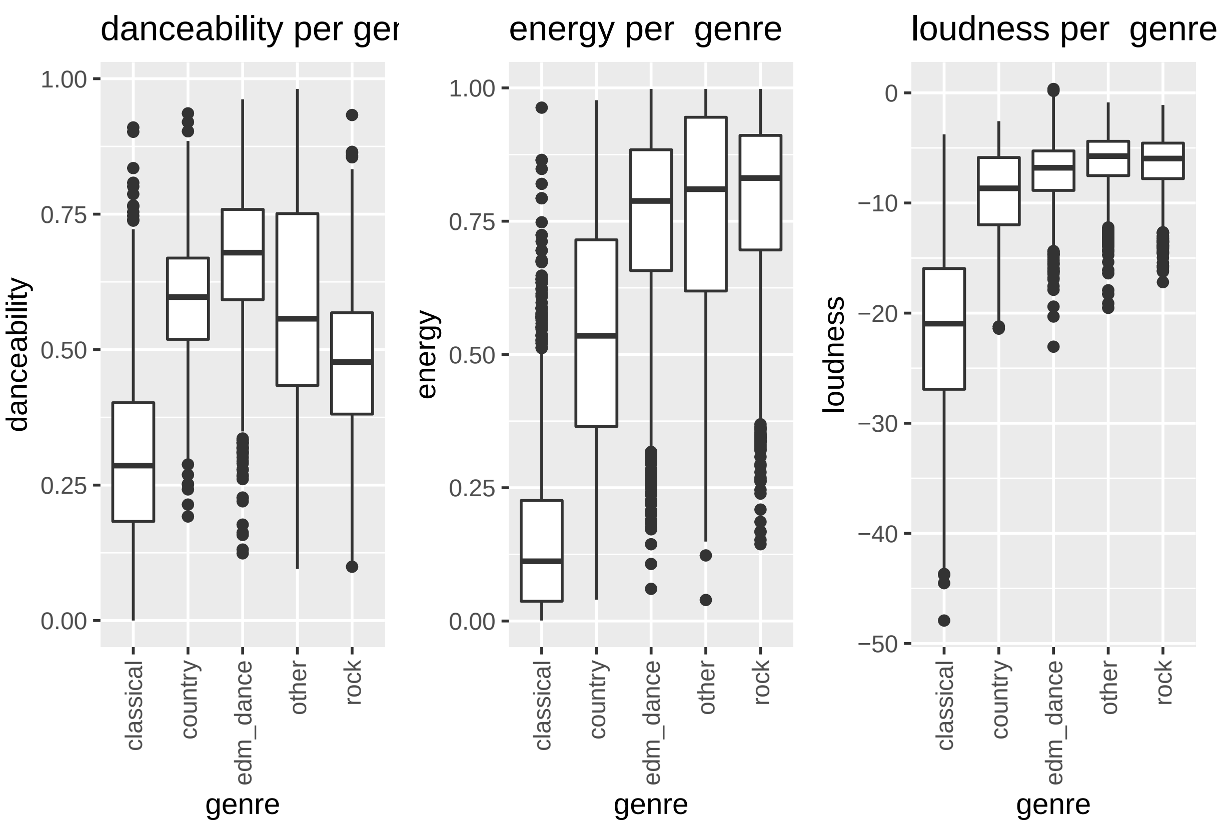
```
##                      (Other):2545
##    acousticness       instrumentalness       liveness          valence
##   Min.   :0.0000012   Min.   :0.0000000   Min.   :0.0119   Min.   :0.0000
##   1st Qu.:0.0043475   1st Qu.:0.0000036   1st Qu.:0.0930   1st Qu.:0.1670
##   Median :0.0850500   Median :0.0046950   Median :0.1180   Median :0.3730
##   Mean   :0.3210833   Mean   :0.2850233   Mean   :0.1747   Mean   :0.3955
##   3rd Qu.:0.7210000   3rd Qu.:0.7490000   3rd Qu.:0.2170   3rd Qu.:0.5870
##   Max.   :0.9960000   Max.   :0.9840000   Max.   :0.9790   Max.   :0.9820
##
##       tempo            type                 id              duration_sec
##   Min.   :  0.00   Length:6588        Length:6588        Min.   :  38.49
##   1st Qu.: 97.85   Class :character   Class :character   1st Qu.: 175.83
##   Median :122.89   Mode  :character   Mode  :character   Median : 211.00
##   Mean   :120.39                                         Mean   : 230.45
##   3rd Qu.:137.42                                         3rd Qu.: 258.45
##   Max.   :220.15                                         Max.   :1344.96
##
##   time_signature      follow               pop           num.artist
##   Min.   :0.000   Min.   :       11   Min.   :  0.00   Min.   : 1.000
##   1st Qu.:4.000   1st Qu.:    24891   1st Qu.: 49.00   1st Qu.: 1.000
##   Median :4.000   Median :   179304   Median : 66.00   Median : 1.000
##   Mean   :3.895   Mean   :  2060257   Mean   : 82.15   Mean   : 1.493
##   3rd Qu.:4.000   3rd Qu.:  1127706   3rd Qu.:100.00   3rd Qu.: 2.000
##   Max.   :5.000   Max.   :142499402   Max.   :652.00   Max.   :17.000
##
##         genre           t_ID
##   classical:1537   Min.   :   1
##   edm_dance:1446   1st Qu.:1648
##   rock     :1365   Median :3294
##   country  : 831   Mean   :3294
##   hiphop   : 636   3rd Qu.:4941
##   metal    : 631   Max.   :6588
##   (Other)  : 142
```

We can observe a wide range of distributions for the predictor variables. Some predictors like danceability and tempo seem to roughly follow a normal distribution. Acousticness however seems to roughly follow a bimodal distribution. Some plots, particularly the speechiness plot do not show a lot of variation. It seems the vast majority of observations have a speechiness of around 0.

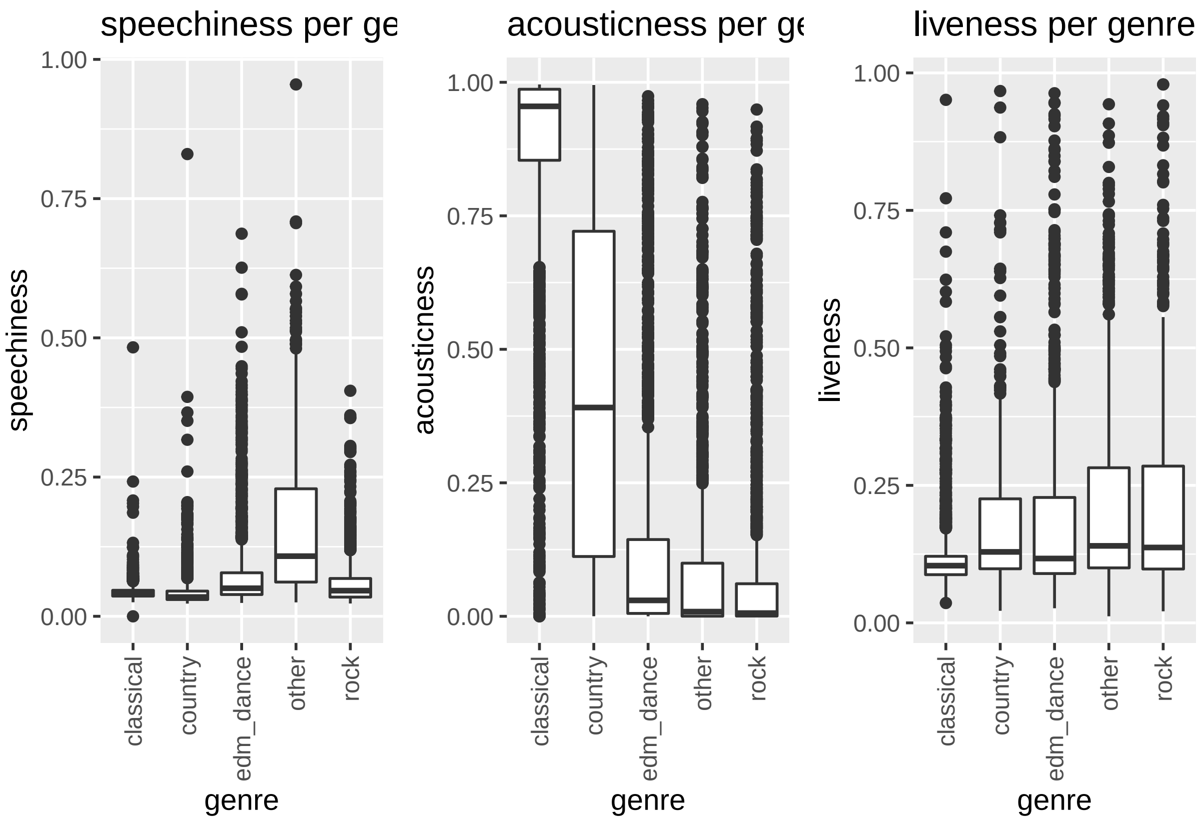
EDA CONTINOUS PREDICTORS VS GENRE

```
bg1_dance <- ggplot(g1, aes(genre, danceability)) + geom_boxplot() +labs (title = "danceability per genr

bg1_energy <- ggplot(g1, aes(genre, energy)) + geom_boxplot() +labs (title = "energy per  genre")+ theme

bg1_loud <- ggplot(g1, aes(genre, loudness)) + geom_boxplot() +labs (title = "loudness per  genre")+ the

bg1_speech <- ggplot(g1, aes(genre, speechiness)) + geom_boxplot() +labs (title = "speechiness per genre

bg1_acoust <- ggplot(g1, aes(genre, acousticness)) + geom_boxplot() +labs (title = "acousticness per gen

bg1_live <- ggplot(g1, aes(genre, liveness)) + geom_boxplot() +labs (title = "liveness per genre")+ them

bg1_val <- ggplot(g1, aes(genre, valence)) + geom_boxplot() +labs (title = "valence per genre")+ theme(a
```

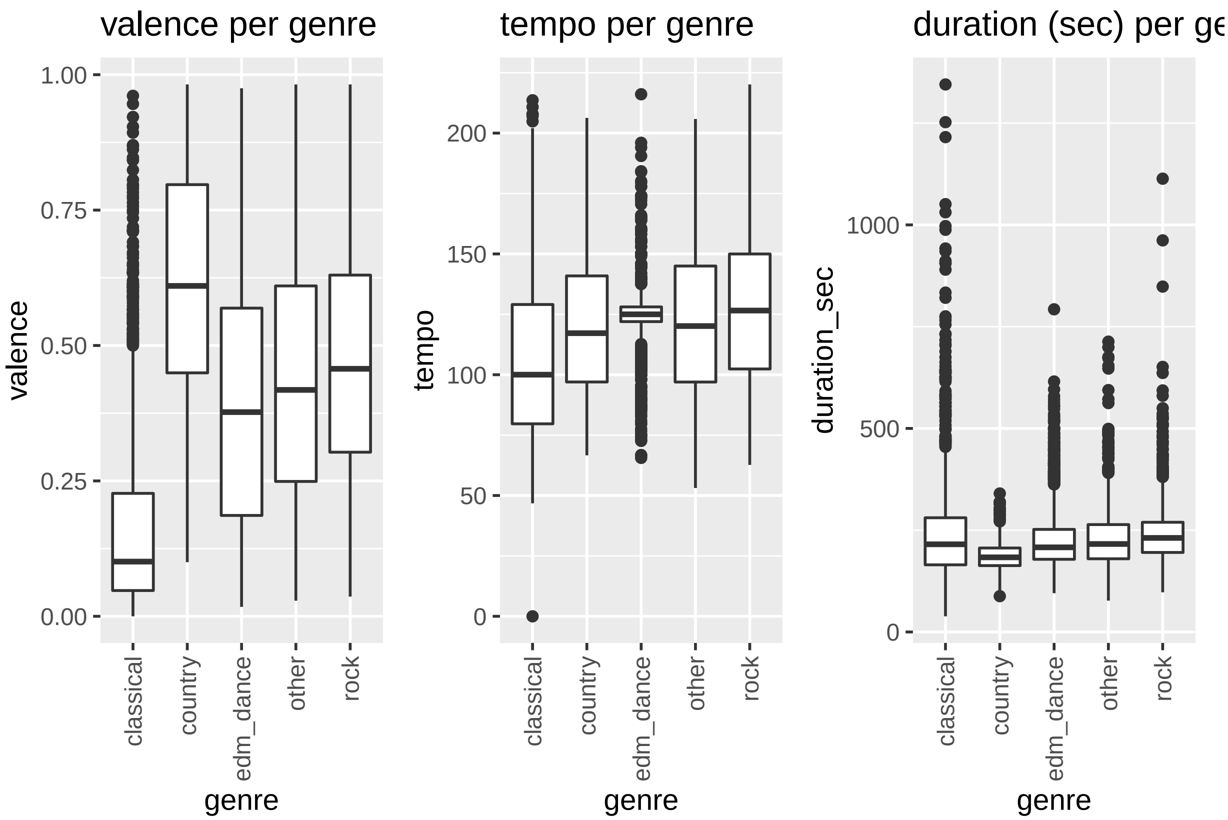
```
bg1_tempo <- ggplot(g1, aes(genre, tempo)) + geom_boxplot() +labs (title = "tempo per genre")+ theme(ax
bg1_dur <- ggplot(g1, aes(genre, duration_sec)) + geom_boxplot() +labs (title = "duration (sec) per gen

bg1_dance + bg1_energy + bg1_loud
```




```
bg1_speech + bg1_acoust + bg1_live
```




```
bg1_val + bg1_tempo + bg1_dur
```

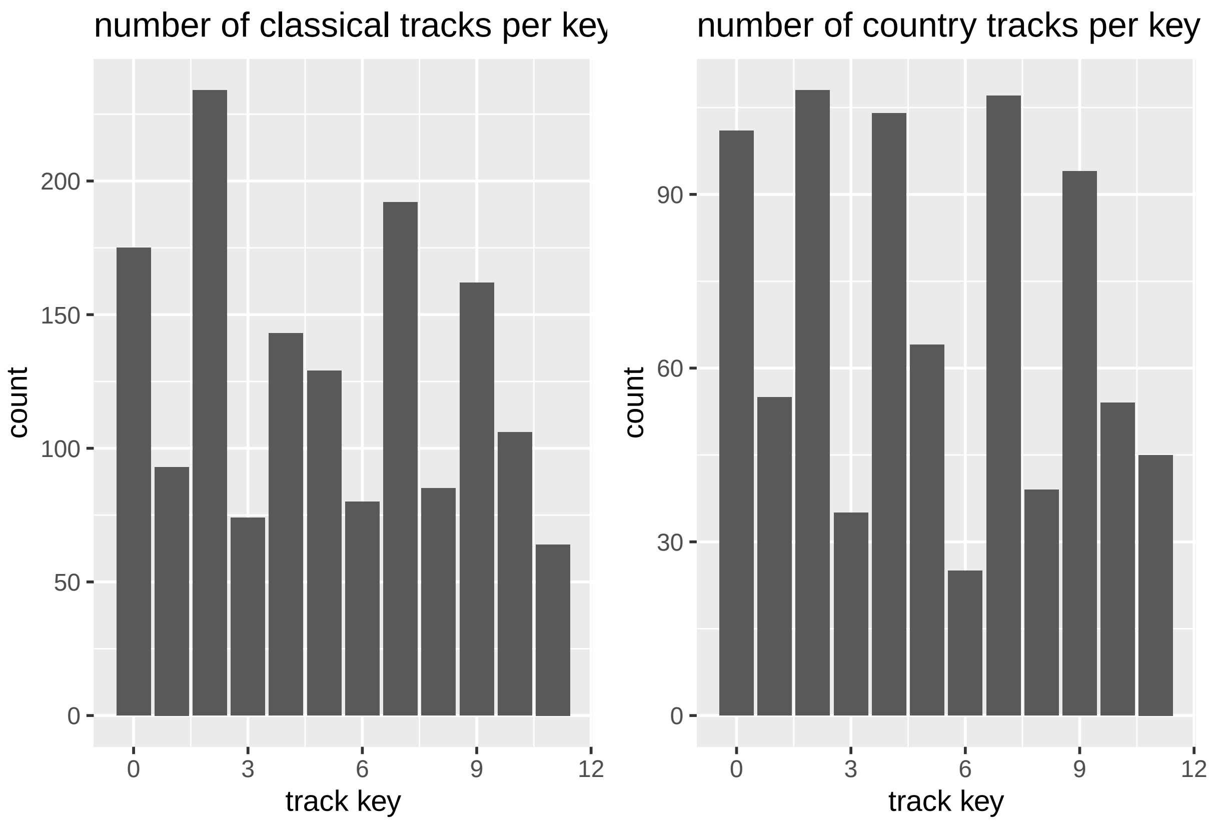


Here we plotted the continous predictor values per genre. Speechiness liveness and duration do not show that much variability between genres. Another interesting thing is that classical has a much different median for many of the predictors. This may suggest that classical would be a good baseline category. The most promising predictors based on these plots seem to be energy and danceability based on their variation between genres.

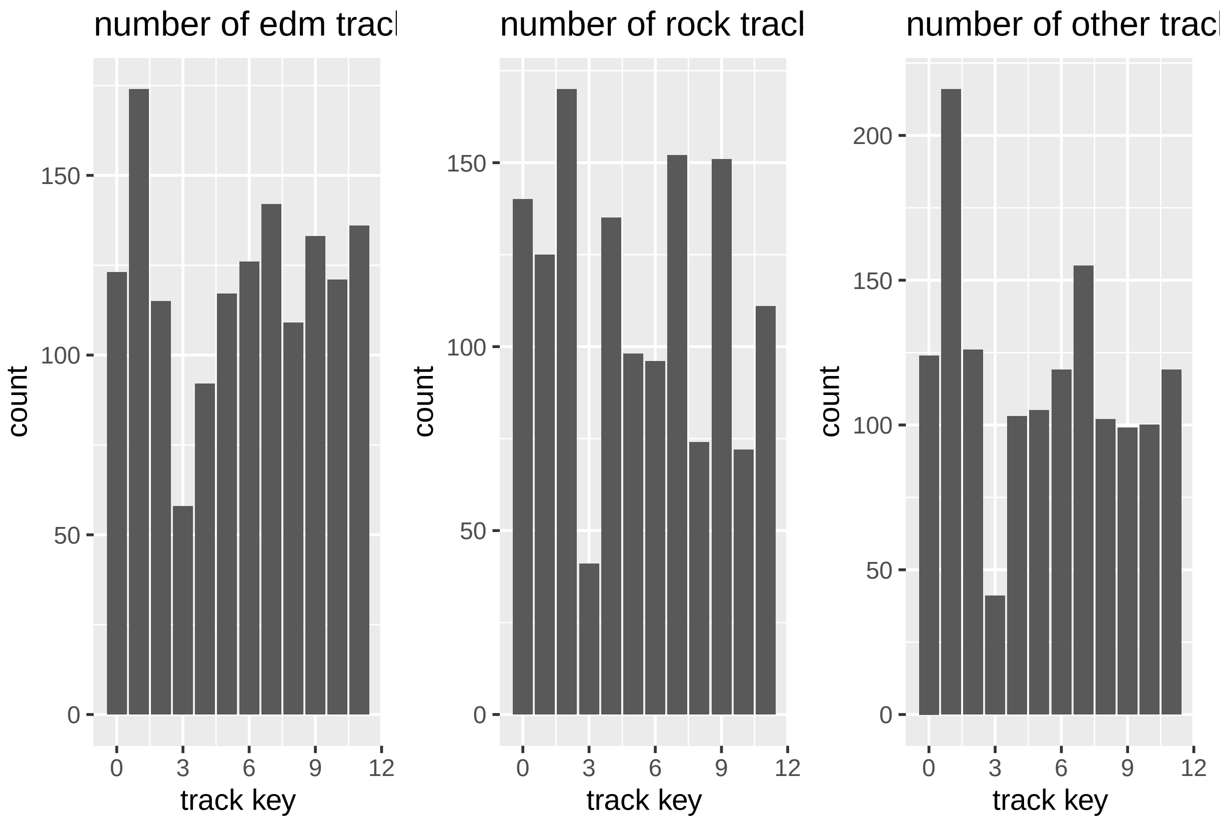
[TODO: EDA ON CATEGORICAL PREDICTORS]

```
g_clas <- g1 %>% filter(genre == "classical" )
g_country <- g1 %>% filter(genre == "country" )
g_edm <- g1 %>% filter(genre == "edm_dance" )
g_rock <- g1 %>% filter(genre == "rock" )
g_other <- g1 %>% filter(genre == "other" )

pkey_clas <- ggplot(g_clas, aes(x=key)) + geom_bar() + labs (y = "count", x = "track key",
                                                              title = "number of classical tracks per key

pkey_country <- ggplot(g_country, aes(x=key)) + geom_bar() + labs (y = "count", x = "track key",
                                                              title = "number of country tracks per key")

pkey_edm <- ggplot(g_edm, aes(x=key)) + geom_bar() + labs (y = "count", x = "track key",
                                                              title = "number of edm tracks per key")

pkey_rock <- ggplot(g_rock, aes(x=key)) + geom_bar() + labs (y = "count", x = "track key",
                                                              title = "number of rock tracks per key")

pkey_other <- ggplot(g_other, aes(x=key)) + geom_bar() + labs (y = "count", x = "track key",
```

```
                                                        title = "number of other tracks per key")

pkey_clas + pkey_country
```



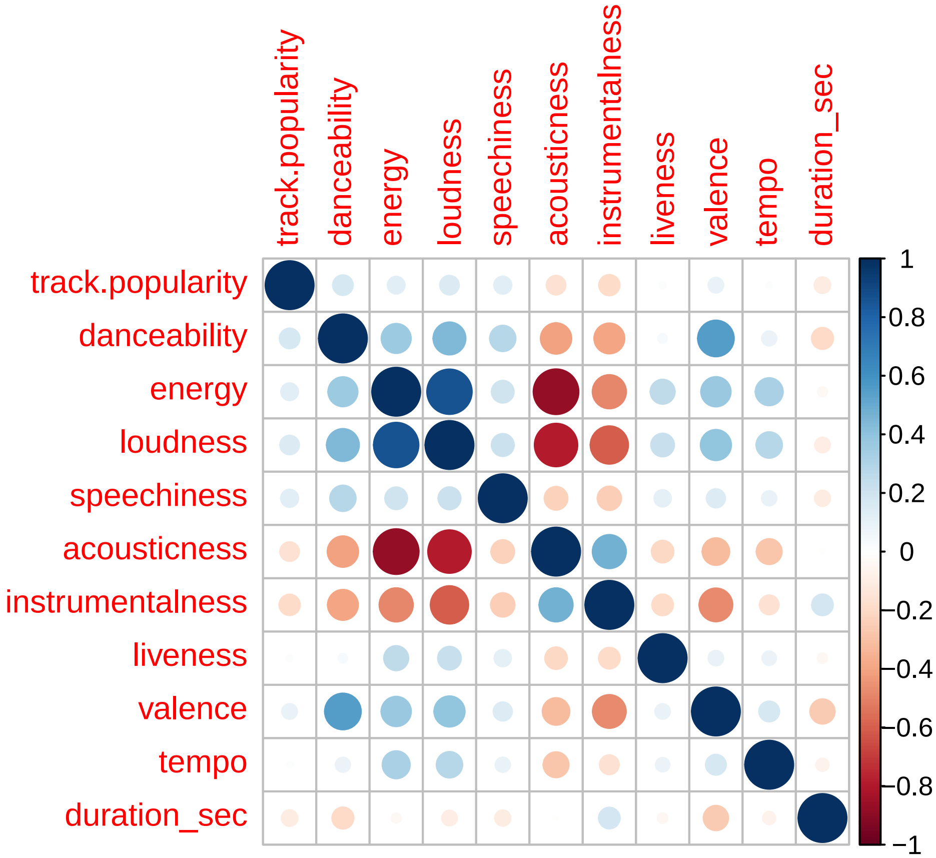

```
pkey_edm + pkey_rock + pkey_other
```



EDA INDEPEDNENT COVARIATES

Lets next determine if there are any possible covariates that are dependent from eachother. To do this, we will create a linear correlation plot of the continuous predictor variables.

```
ntype_data <- subset(track_data, select = -c(track.name, artists.names, artists.ids, artist.ids, track.i
corrplot(cor(ntype_data))
```



We can observe that for the most part, there is not much correlation between predictor variables. However, acousticness appears to be highly correlated with energy and loudness. This may cause noise in our model. However, before determining if I should drop any predictor variables I should use an ANOVA test. This correlation plot suggests that I may want to run an anova test without acousticness in the reduced model.

MODELING:

Finally, lets create a multinomial model with all original predictor variables.

```
g1 <- g1 %>%  mutate(genre = as.factor(genre)) %>% mutate(key = as.factor(key)) %>% mutate(mode = as.fa


full_model <- multinom(genre ~ danceability + energy + loudness + speechiness + acousticness + instrume
```

```
## # weights:  120 (92 variable)
## initial  value 10602.976967
## iter  10 value 8584.861247
## iter  20 value 5816.989315
## iter  30 value 5031.729575
## iter  40 value 4833.859614
## iter  50 value 4765.265875
## iter  60 value 4716.201636
## iter  70 value 4704.444742
## iter  80 value 4698.811143
## iter  90 value 4694.969253
## iter 100 value 4694.668567
## final  value 4694.668567
## stopped after 100 iterations
```

```r
tidy(full_model, conf.int = TRUE, exponentiate = FALSE) %>%
  kable(digits = 3, format = "markdown")
```

y.level	term	estimate	std.error	statistic	p.value	conf.low	conf.high
country	(Intercept)	2.045	0.535	3.824	0.000	0.997	3.093
country	danceability	4.586	0.376	12.205	0.000	3.849	5.322
country	energy	-0.272	0.404	-0.672	0.502	-1.064	0.521
country	loudness	0.207	0.029	7.170	0.000	0.150	0.264
country	speechiness	-8.827	0.055	-160.445	0.000	-8.934	-8.719
country	acousticness	-2.981	0.365	-8.158	0.000	-3.697	-2.265
country	instrumentalness	-7.552	0.540	-13.988	0.000	-8.610	-6.494
country	liveness	3.940	0.299	13.176	0.000	3.354	4.527
country	valence	3.709	0.335	11.068	0.000	3.052	4.366
country	tempo	0.000	0.003	0.002	0.999	-0.006	0.006
country	duration_sec	-0.007	0.001	-5.253	0.000	-0.009	-0.004
country	key1	-0.281	0.362	-0.775	0.438	-0.991	0.429
country	key2	-0.559	0.295	-1.894	0.058	-1.138	0.020
country	key3	0.122	0.426	0.287	0.774	-0.712	0.956
country	key4	0.231	0.341	0.678	0.497	-0.437	0.899
country	key5	-0.734	0.345	-2.127	0.033	-1.411	-0.058
country	key6	0.170	0.227	0.748	0.455	-0.275	0.614
country	key7	-0.560	0.297	-1.889	0.059	-1.142	0.021
country	key8	-0.900	0.383	-2.348	0.019	-1.651	-0.149
country	key9	0.183	0.322	0.568	0.570	-0.449	0.815
country	key10	-0.631	0.372	-1.695	0.090	-1.360	0.099
country	key11	0.779	0.452	1.723	0.085	-0.107	1.665
country	mode1	2.204	0.235	9.363	0.000	1.743	2.665
edm_dance	(Intercept)	-7.805	0.447	-17.443	0.000	-8.681	-6.928
edm_dance	danceability	14.314	0.322	44.456	0.000	13.683	14.945
edm_dance	energy	6.504	0.351	18.545	0.000	5.817	7.192
edm_dance	loudness	0.191	0.031	6.232	0.000	0.131	0.251
edm_dance	speechiness	3.303	0.473	6.986	0.000	2.376	4.229
edm_dance	acousticness	-2.147	0.346	-6.209	0.000	-2.824	-1.469
edm_dance	instrumentalness	-2.160	0.290	-7.460	0.000	-2.727	-1.592
edm_dance	liveness	4.552	0.227	20.013	0.000	4.106	4.997
edm_dance	valence	-2.687	0.303	-8.854	0.000	-3.282	-2.092
edm_dance	tempo	0.007	0.003	2.211	0.027	0.001	0.014
edm_dance	duration_sec	0.003	0.001	3.087	0.002	0.001	0.005
edm_dance	key1	0.169	0.345	0.491	0.624	-0.507	0.846
edm_dance	key2	-0.683	0.297	-2.304	0.021	-1.265	-0.102
edm_dance	key3	0.707	0.430	1.646	0.100	-0.135	1.550
edm_dance	key4	-0.584	0.339	-1.724	0.085	-1.247	0.080
edm_dance	key5	-0.190	0.332	-0.572	0.567	-0.842	0.461
edm_dance	key6	1.218	0.162	7.514	0.000	0.900	1.536
edm_dance	key7	-0.622	0.292	-2.131	0.033	-1.194	-0.050
edm_dance	key8	-0.022	0.365	-0.061	0.952	-0.737	0.693
edm_dance	key9	0.281	0.318	0.884	0.377	-0.342	0.904
edm_dance	key10	-0.666	0.369	-1.804	0.071	-1.389	0.058
edm_dance	key11	0.789	0.441	1.790	0.073	-0.075	1.653
edm_dance	mode1	0.020	0.212	0.094	0.925	-0.395	0.435
other	(Intercept)	-1.160	0.404	-2.868	0.004	-1.952	-0.367
other	danceability	6.751	0.283	23.849	0.000	6.196	7.306

y.level	term	estimate	std.error	statistic	p.value	conf.low	conf.high
other	energy	3.739	0.327	11.424	0.000	3.097	4.380
other	loudness	0.286	0.032	9.077	0.000	0.224	0.348
other	speechiness	13.678	0.397	34.495	0.000	12.900	14.455
other	acousticness	-4.255	0.381	-11.165	0.000	-5.002	-3.508
other	instrumentalness	-4.746	0.317	-14.951	0.000	-5.368	-4.124
other	liveness	4.129	0.207	19.901	0.000	3.722	4.535
other	valence	-0.631	0.304	-2.074	0.038	-1.228	-0.035
other	tempo	-0.005	0.003	-1.528	0.126	-0.012	0.001
other	duration_sec	0.007	0.001	7.793	0.000	0.005	0.009
other	key1	0.298	0.345	0.865	0.387	-0.378	0.974
other	key2	-0.816	0.296	-2.754	0.006	-1.396	-0.235
other	key3	0.308	0.435	0.709	0.478	-0.544	1.160
other	key4	-0.467	0.339	-1.376	0.169	-1.131	0.198
other	key5	-0.478	0.337	-1.418	0.156	-1.139	0.183
other	key6	1.029	0.151	6.791	0.000	0.732	1.326
other	key7	-0.490	0.292	-1.676	0.094	-1.063	0.083
other	key8	-0.268	0.369	-0.726	0.468	-0.991	0.455
other	key9	-0.075	0.322	-0.232	0.817	-0.706	0.557
other	key10	-0.744	0.370	-2.011	0.044	-1.469	-0.019
other	key11	0.615	0.441	1.396	0.163	-0.249	1.479
other	mode1	0.444	0.216	2.056	0.040	0.021	0.866
rock	(Intercept)	0.778	0.394	1.973	0.048	0.005	1.550
rock	danceability	1.552	0.290	5.345	0.000	0.983	2.122
rock	energy	3.288	0.320	10.285	0.000	2.661	3.914
rock	loudness	0.218	0.029	7.422	0.000	0.160	0.275
rock	speechiness	-0.976	0.593	-1.648	0.099	-2.138	0.185
rock	acousticness	-4.885	0.367	-13.303	0.000	-5.605	-4.165
rock	instrumentalness	-4.824	0.306	-15.760	0.000	-5.423	-4.224
rock	liveness	4.248	0.199	21.300	0.000	3.857	4.639
rock	valence	2.604	0.300	8.673	0.000	2.016	3.193
rock	tempo	-0.003	0.003	-1.051	0.293	-0.010	0.003
rock	duration_sec	0.006	0.001	7.512	0.000	0.005	0.008
rock	key1	0.026	0.345	0.075	0.940	-0.651	0.702
rock	key2	-0.327	0.283	-1.155	0.248	-0.882	0.228
rock	key3	0.192	0.424	0.452	0.651	-0.640	1.024
rock	key4	0.054	0.327	0.165	0.869	-0.587	0.696
rock	key5	-0.380	0.329	-1.154	0.249	-1.026	0.266
rock	key6	0.902	0.151	5.960	0.000	0.605	1.199
rock	key7	-0.497	0.287	-1.732	0.083	-1.059	0.065
rock	key8	-0.549	0.367	-1.496	0.135	-1.269	0.170
rock	key9	0.313	0.311	1.007	0.314	-0.296	0.921
rock	key10	-0.698	0.366	-1.905	0.057	-1.415	0.020
rock	key11	1.027	0.435	2.361	0.018	0.174	1.879
rock	mode1	0.994	0.214	4.644	0.000	0.574	1.413

KEY:

We can observe that some paramaters have a high p value particularly ones that are ascociated with key. If a tracks key is not useful for humans to determine a songs track, then this model accurately reflects that. However, if a tracks key can be used by humans to help determine the genre, then this is either an inaccuracy in the model, the data, or Spotify's alrgorithim for determining a tracks key.

Looking at the key more closely, we can see that the genres of other, rock, and edm_dance have a low p-value for key6. Key 6 correspondes to the key of f-sharp. Maybe this would be a good place to ask an expert in the field to interpret why a key6 would have a low p value for edm_dance, other, and rock.

From my understanding of music theory, a key is the set of notes used in the song determined by the base note and whether the song is major or minor. However, Spotify's key feature only is the base note and does not include whether it is major or minor. The Spotify's key feature would more accurately be named "picth class". It is important to consider this when interpreting the key feature.
See https://developer.spotify.com/documentation/web-api/reference/#/operations/get-several-audio-features

DANCEABLE:

If I told you a track was very "danceable", you would probably think that it is more likely for the track to belong to the electronic dance music genre than the classical genre. This is reflected in the model which shows that the danceability of a track increases its log odds of being an electronic dance music track as oppose to a classical track by 14.31

TEMPO:

Aside from the key, another alarmingly large p-value is the tempo. Particularly, in country and rock. Other response categories have a much lower p-value for tempo, however the confidence interval is close to 0. If tempo is independent from genre then this model accurately reflects this. However, if tempo is dependent on a songs genre then there is either an issue with the model, the data, or Spotify's algorithm for determining tempo.

SPEECHINESS:

There is also some interesting model outputs for speechiness. For some genres, speechiness is behaving like we would generall expect it to. Specifically, the other and edm_dance genres have low p-values with confidence intervals far from 0. This is what we would expect because classical tracks should have relatively very low levels of speech.

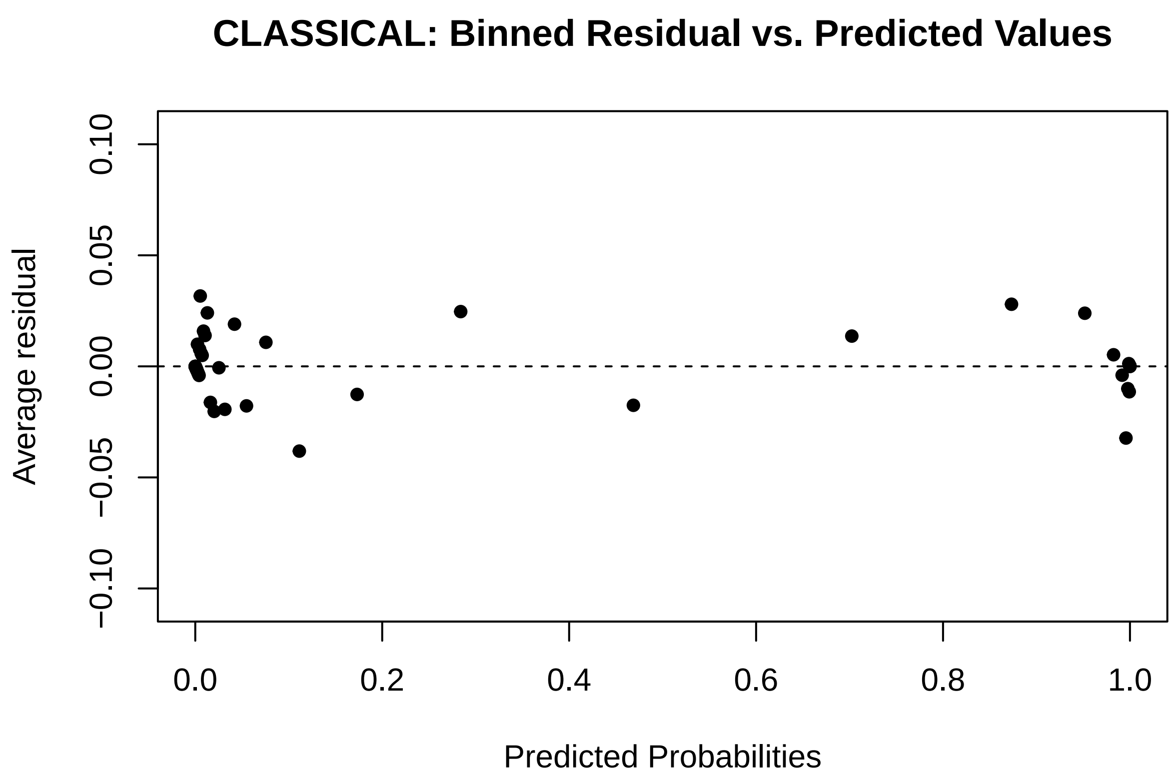
However, according to the model a higher speechiness value correlates to a higher liklehood that a given track is of the classical genre as oppose to the country genre. The p value is low and confidence interval far from 0 for this paramater.

```r
pred_probs <- as_tibble(predict(full_model, type = "probs")) %>%
                        mutate(t_ID = 1:n())

full_m_aug <- inner_join(g1, pred_probs,
                           by = "t_ID") %>%
  select(t_ID, everything())

full_m_aug <- full_m_aug %>%
  mutate(pred_gen = predict(full_model, type = "class"))

residuals <- as_tibble(residuals(full_model)) %>%  #calculate residuals
  setNames(paste('resid.', names(.), sep = "")) %>% #update column names
  mutate(t_ID = 1:n()) #add obs number


full_m_aug <- inner_join(full_m_aug, residuals, by = "t_ID") %>%
  select(t_ID, everything())

conf_m <- full_m_aug %>%
  count(genre, pred_gen, .drop = FALSE) %>%
  pivot_wider(names_from = pred_gen, values_from = n)
```

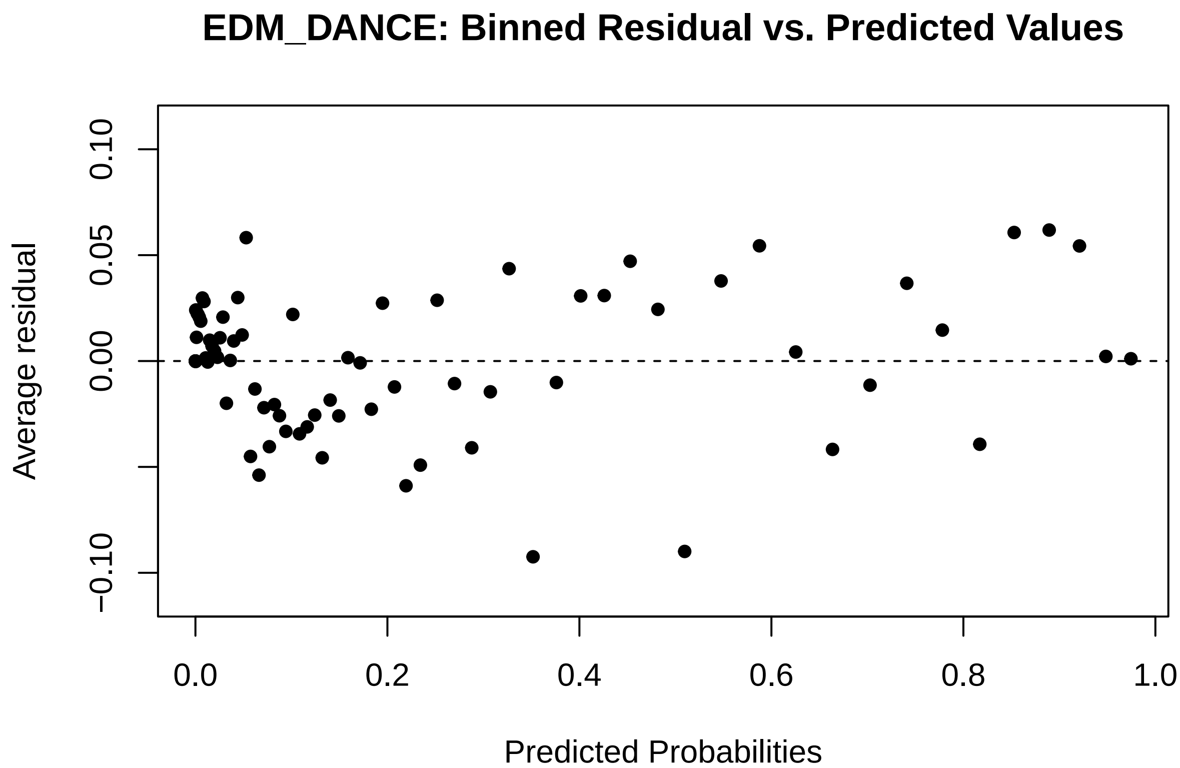
```
conf_m
```

```
## # A tibble: 5 x 6
##   genre     classical country edm_dance other  rock
##   <fct>         <int>   <int>     <int> <int> <int>
## 1 classical      1458      27        32     2    18
## 2 country          22     543        38    22   206
## 3 edm_dance        29      70      1034   185   128
## 4 other             9      56       248   762   334
## 5 rock             11     118       133   177   926
```

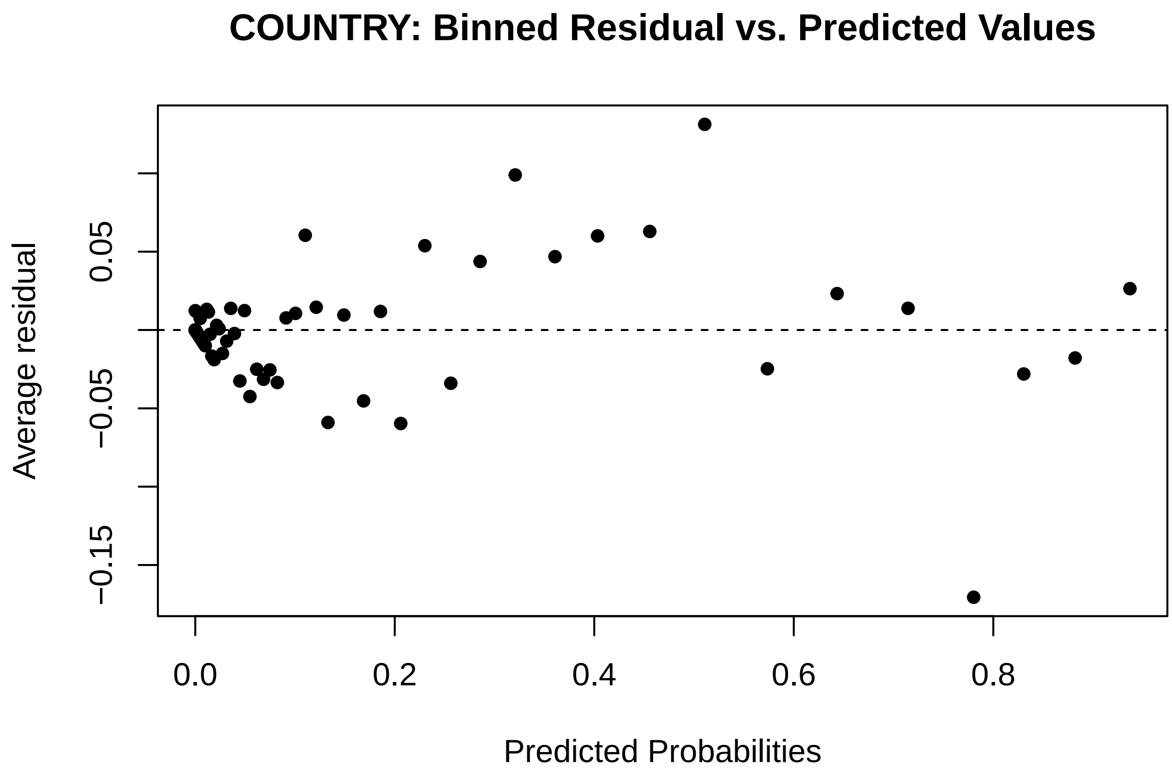
```
nbins <- sqrt(nrow(full_m_aug))

arm::binnedplot(x = full_m_aug$classical, y = full_m_aug$resid.classical,
                xlab = "Predicted Probabilities",
                main = "CLASSICAL: Binned Residual vs. Predicted Values",
                col.int = FALSE)
```



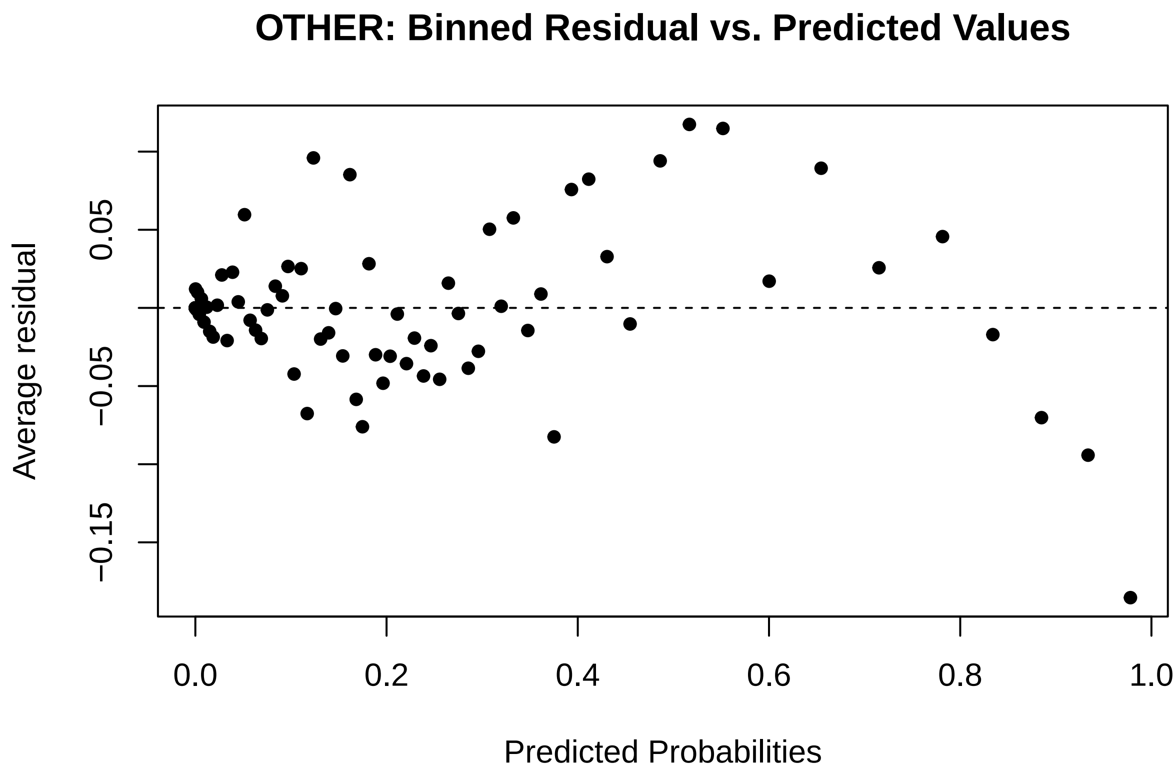

```
arm::binnedplot(x = full_m_aug$edm_dance, y = full_m_aug$resid.edm_dance,
                xlab = "Predicted Probabilities",
                main = "EDM_DANCE: Binned Residual vs. Predicted Values",
                col.int = FALSE)
```



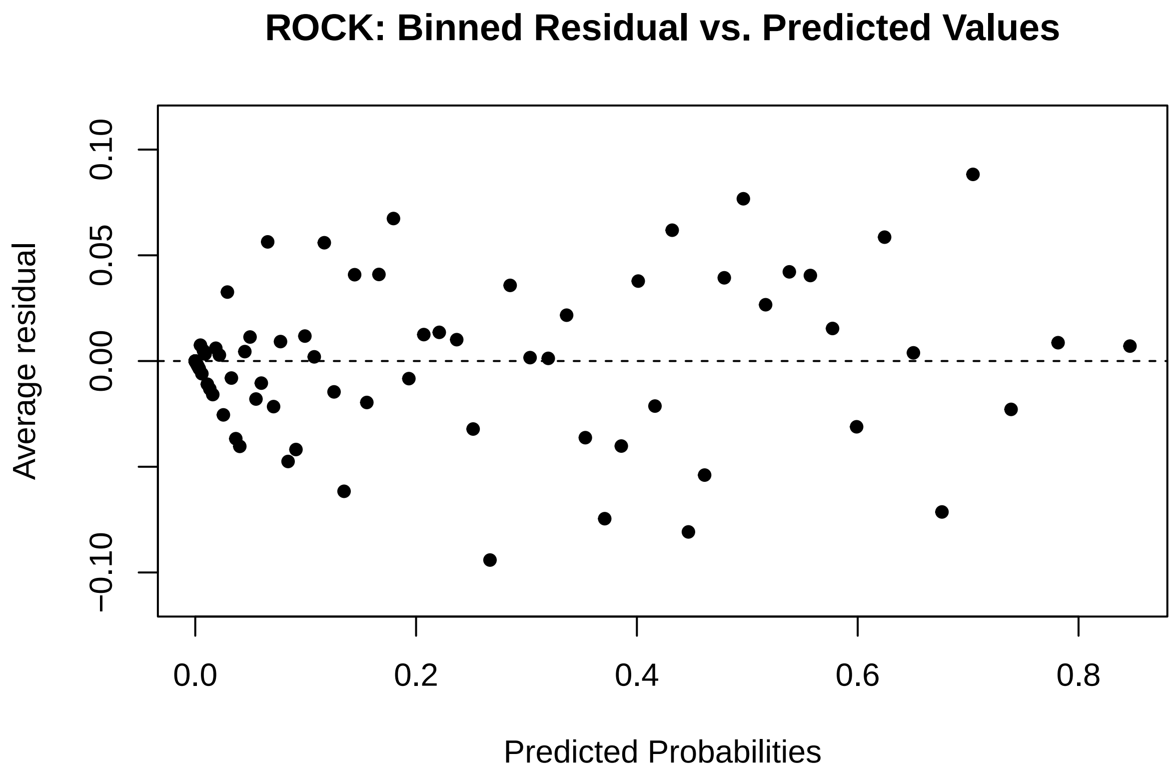

```
arm::binnedplot(x = full_m_aug$country, y = full_m_aug$resid.country,
                xlab = "Predicted Probabilities",
                main = "COUNTRY: Binned Residual vs. Predicted Values",
                col.int = FALSE)
```



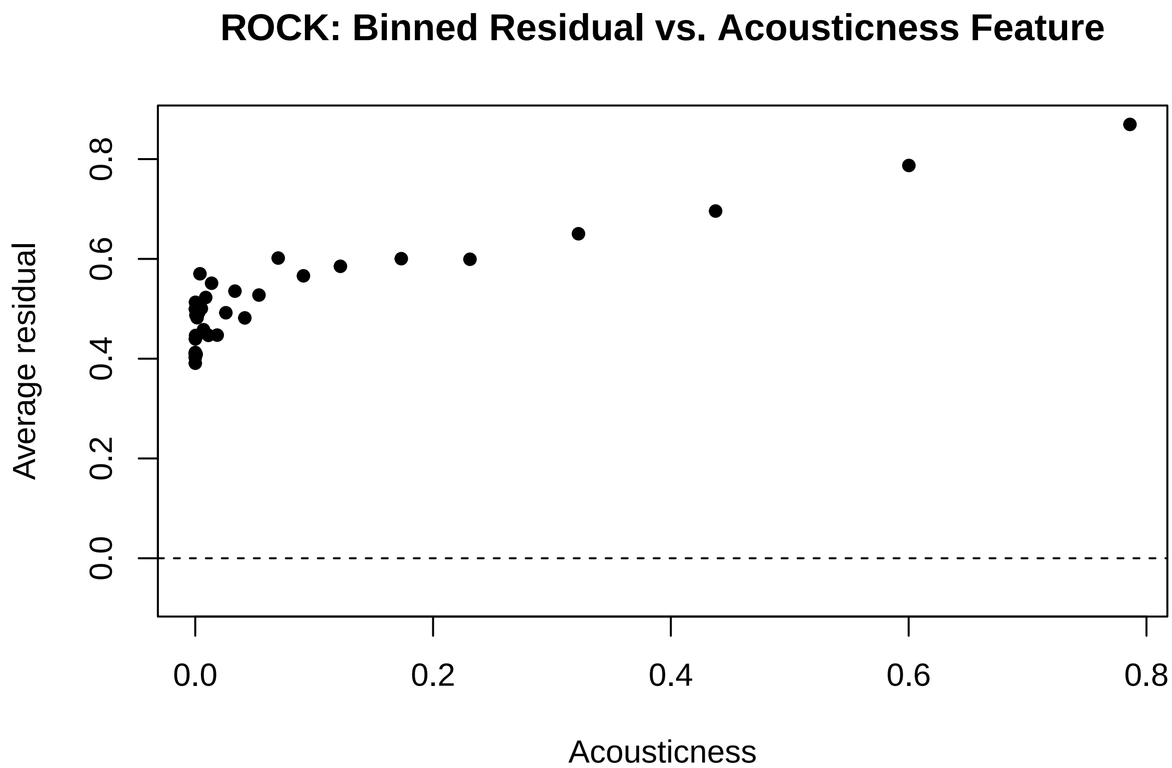

```
arm::binnedplot(x = full_m_aug$other, y = full_m_aug$resid.other,
                xlab = "Predicted Probabilities",
                main = "OTHER: Binned Residual vs. Predicted Values",
                col.int = FALSE)
```



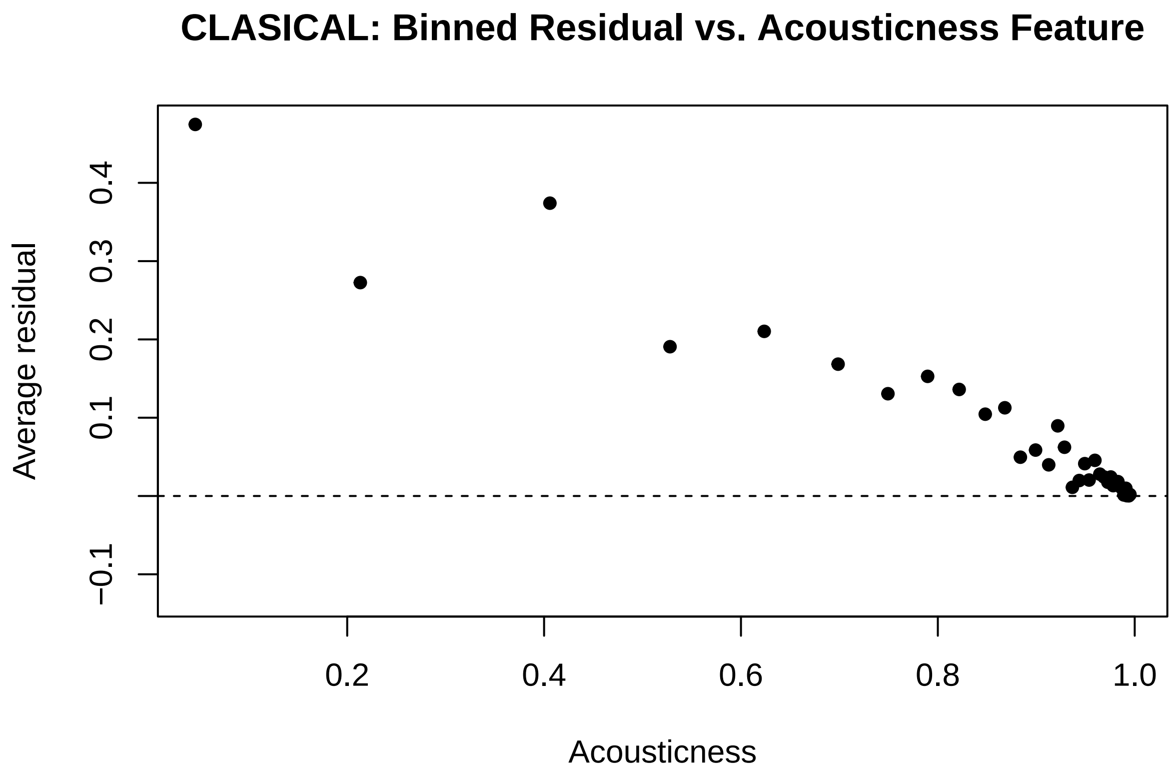

```
arm::binnedplot(x = full_m_aug$rock, y = full_m_aug$resid.rock,
                xlab = "Predicted Probabilities",
                main = "ROCK: Binned Residual vs. Predicted Values",
                col.int = FALSE)
```



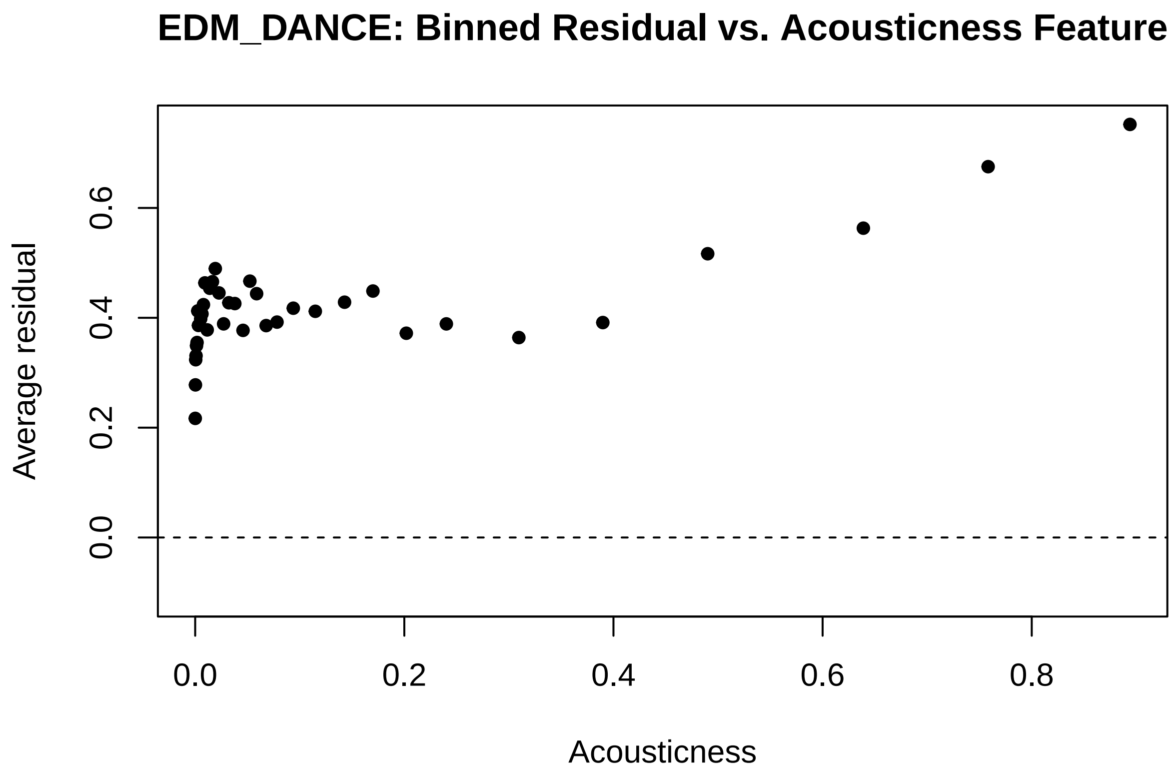

```
full_m_aug_rock <- full_m_aug %>% filter(genre == "rock")
nbins <- sqrt(nrow(full_m_aug_rock))
arm::binnedplot(x = full_m_aug_rock$acousticness, y = full_m_aug_rock$resid.rock,
                xlab = "Acousticness",
                main = "ROCK: Binned Residual vs. Acousticness Feature",
                col.int = FALSE)
```



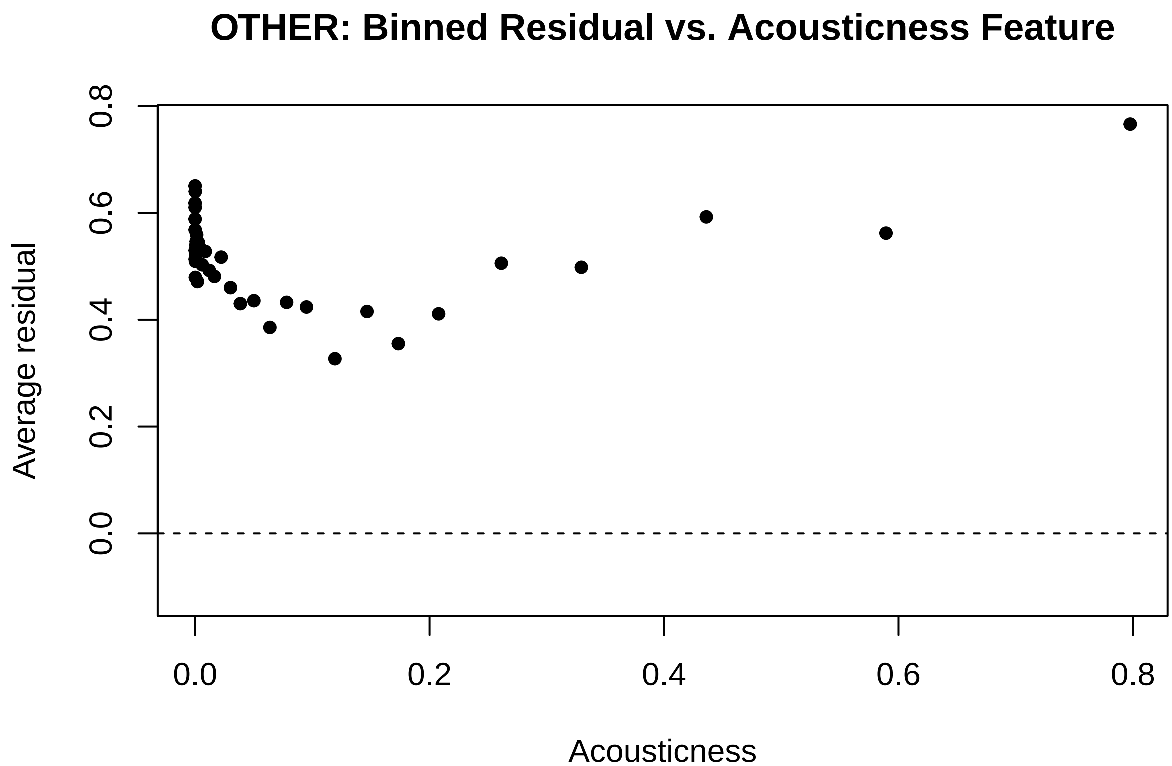

```
full_m_aug_class <- full_m_aug %>% filter(genre == "classical")
nbins <- sqrt(nrow(full_m_aug_class))
arm::binnedplot(x = full_m_aug_class$acousticness, y = full_m_aug_class$resid.classical,
                xlab = "Acousticness",
                main = "CLASICAL: Binned Residual vs. Acousticness Feature",
                col.int = FALSE)
```



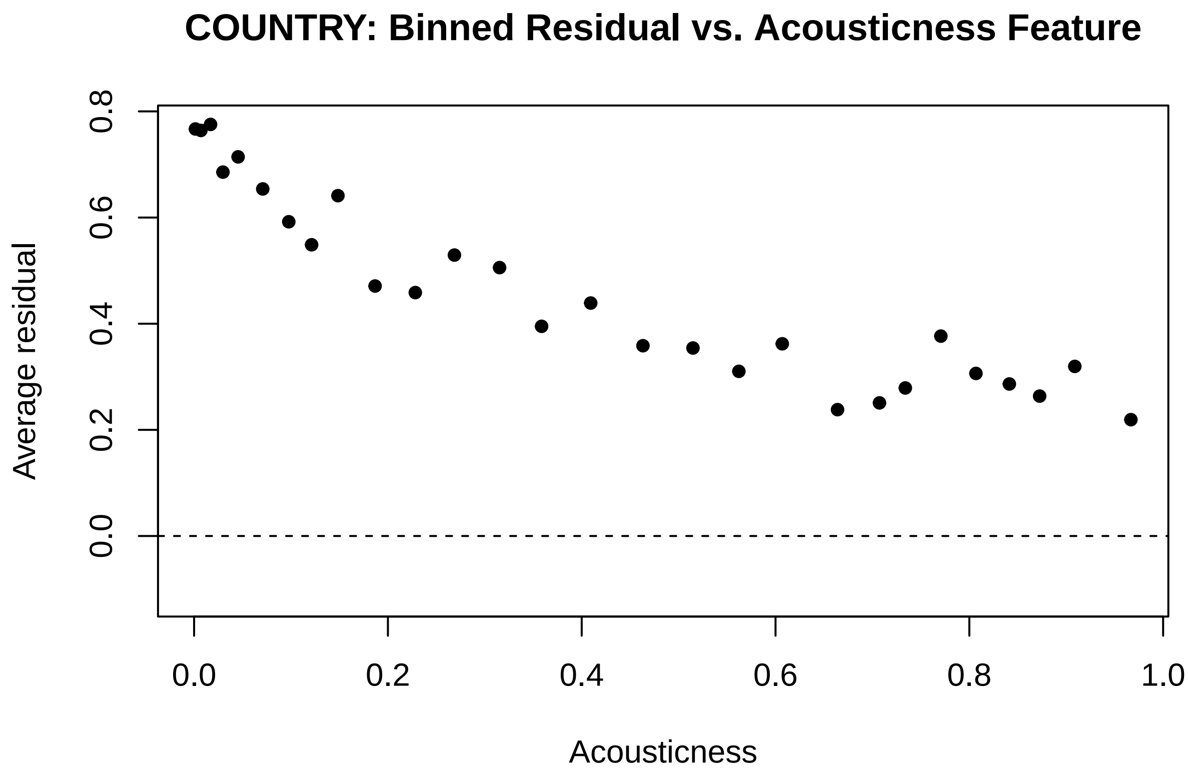

```
full_m_aug_edm <- full_m_aug %>% filter(genre == "edm_dance")
nbins <- sqrt(nrow(full_m_aug_edm))
arm::binnedplot(x = full_m_aug_edm$acousticness, y = full_m_aug_edm$resid.edm_dance,
                xlab = "Acousticness",
                main = "EDM_DANCE: Binned Residual vs. Acousticness Feature",
                col.int = FALSE)
```



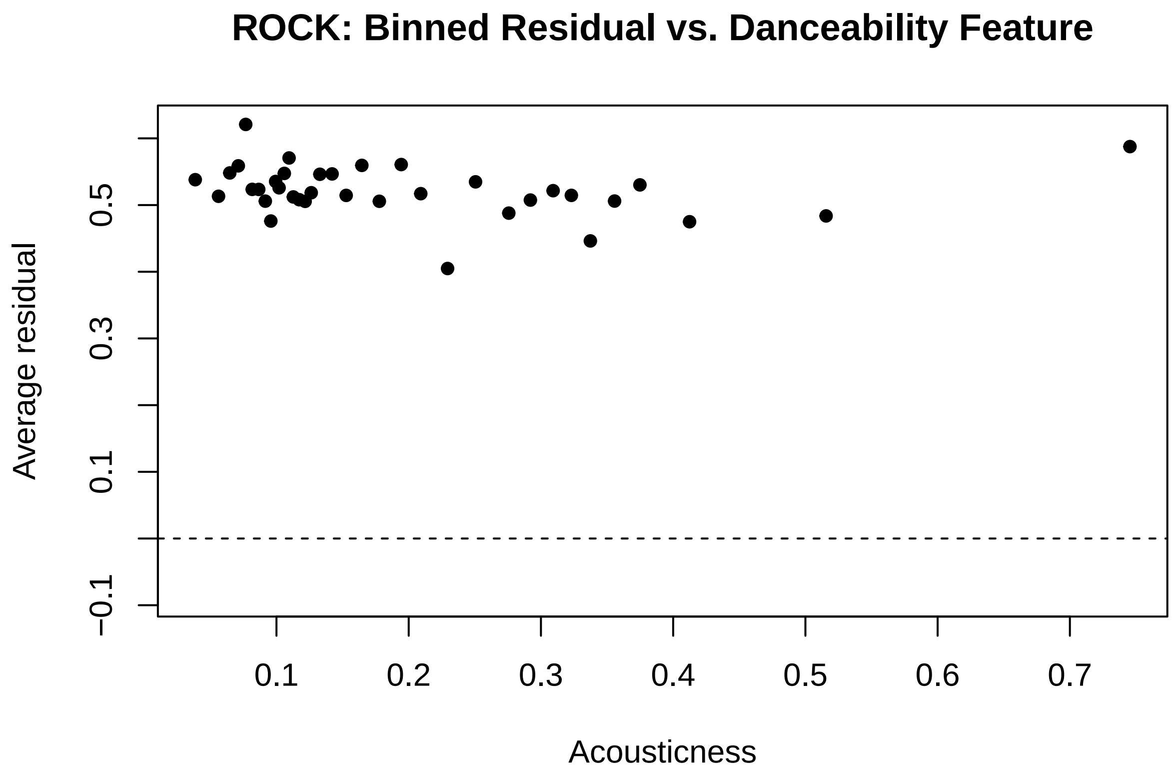

```
full_m_aug_other <- full_m_aug %>% filter(genre == "other")
nbins <- sqrt(nrow(full_m_aug_other))
arm::binnedplot(x = full_m_aug_other$acousticness, y = full_m_aug_other$resid.other,
                xlab = "Acousticness",
                main = "OTHER: Binned Residual vs. Acousticness Feature",
                col.int = FALSE)
```



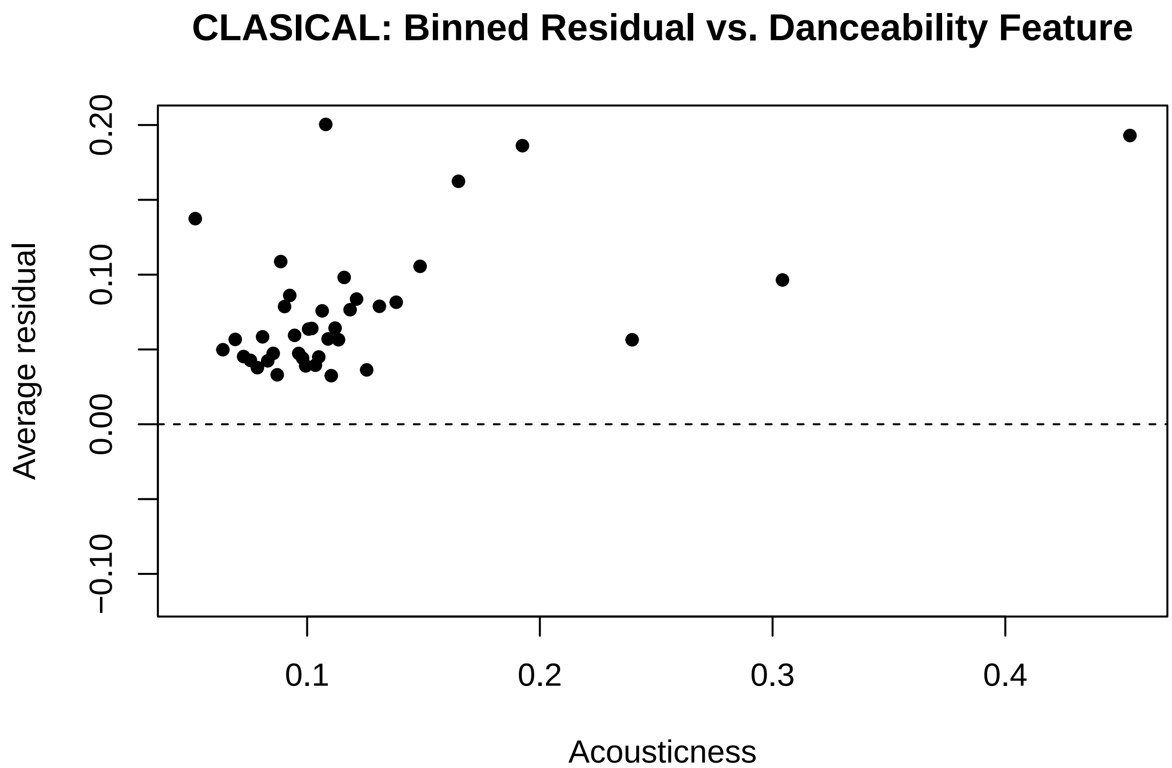

```
full_m_aug_country <- full_m_aug %>% filter(genre == "country")
nbins <- sqrt(nrow(full_m_aug_country))
arm::binnedplot(x = full_m_aug_country$acousticness, y = full_m_aug_country$resid.country,
                xlab = "Acousticness",
                main = "COUNTRY: Binned Residual vs. Acousticness Feature",
                col.int = FALSE)
```



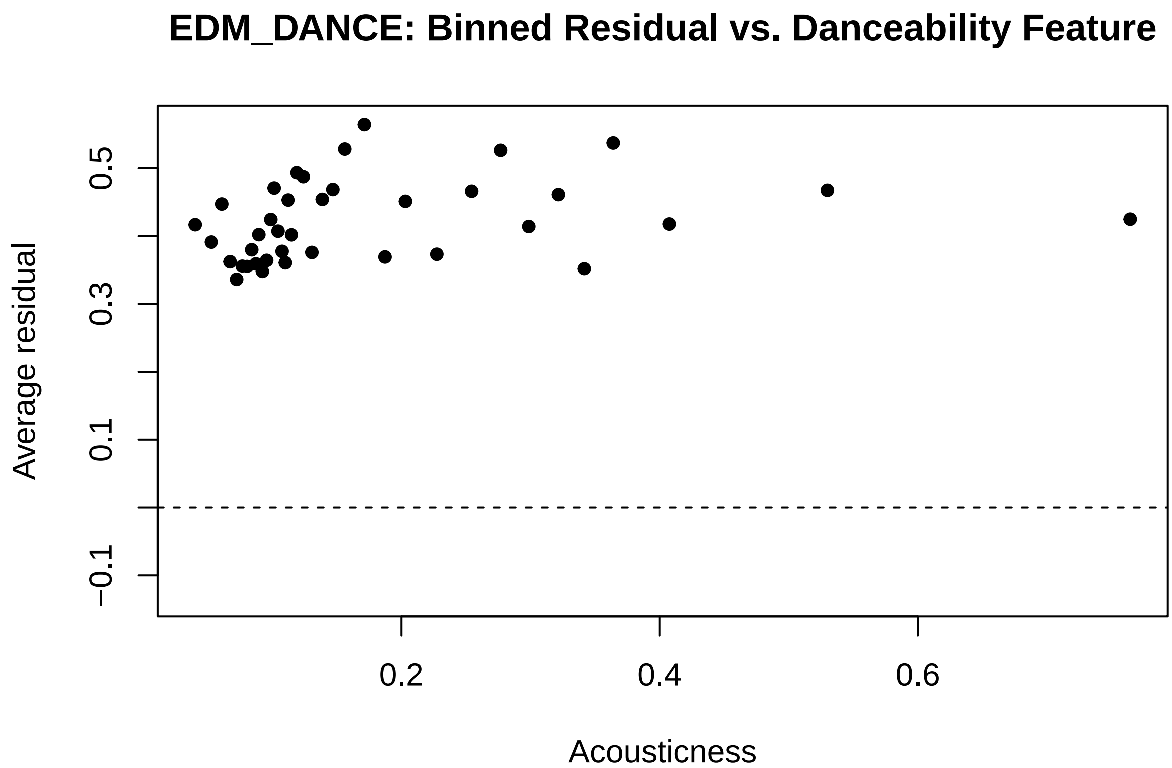

```
full_m_aug_rock <- full_m_aug %>% filter(genre == "rock")
nbins <- sqrt(nrow(full_m_aug_rock))
arm::binnedplot(x = full_m_aug_rock$liveness, y = full_m_aug_rock$resid.rock,
                xlab = "Acousticness",
                main = "ROCK: Binned Residual vs. Danceability Feature",
                col.int = FALSE)
```



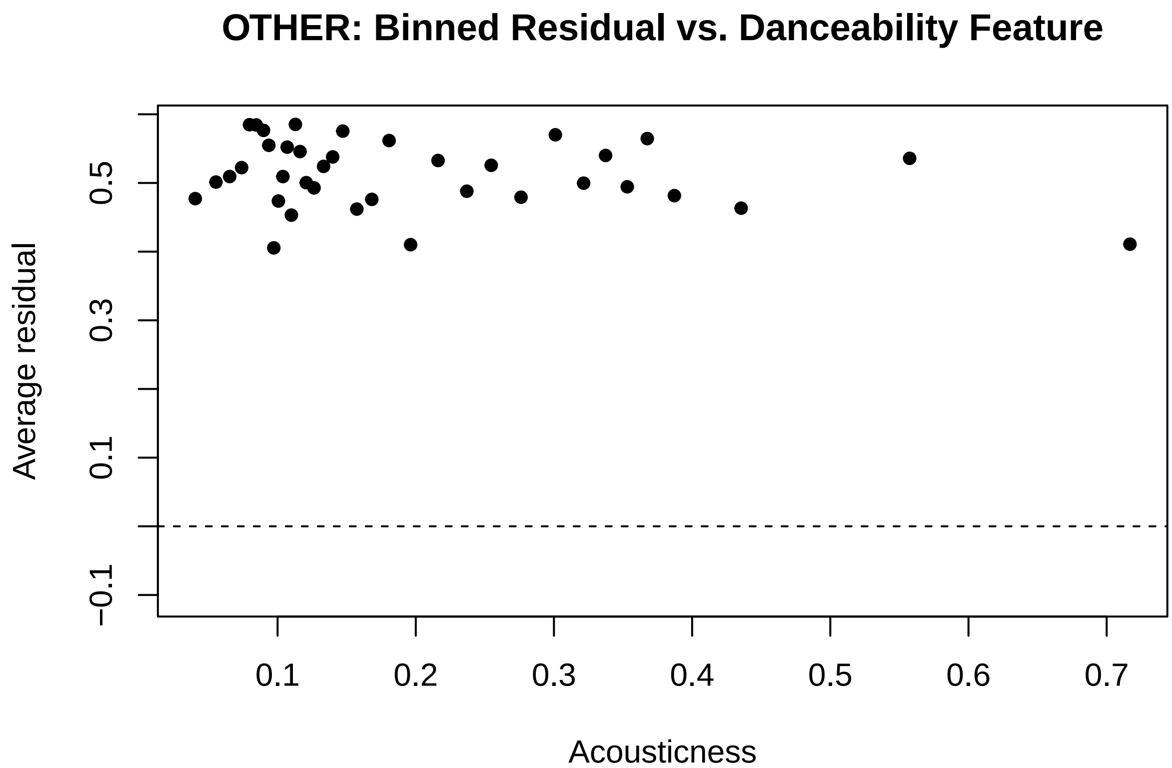

```
full_m_aug_class <- full_m_aug %>% filter(genre == "classical")
nbins <- sqrt(nrow(full_m_aug_class))
arm::binnedplot(x = full_m_aug_class$liveness, y = full_m_aug_class$resid.classical,
                xlab = "Acousticness",
                main = "CLASICAL: Binned Residual vs. Danceability Feature",
                col.int = FALSE)
```



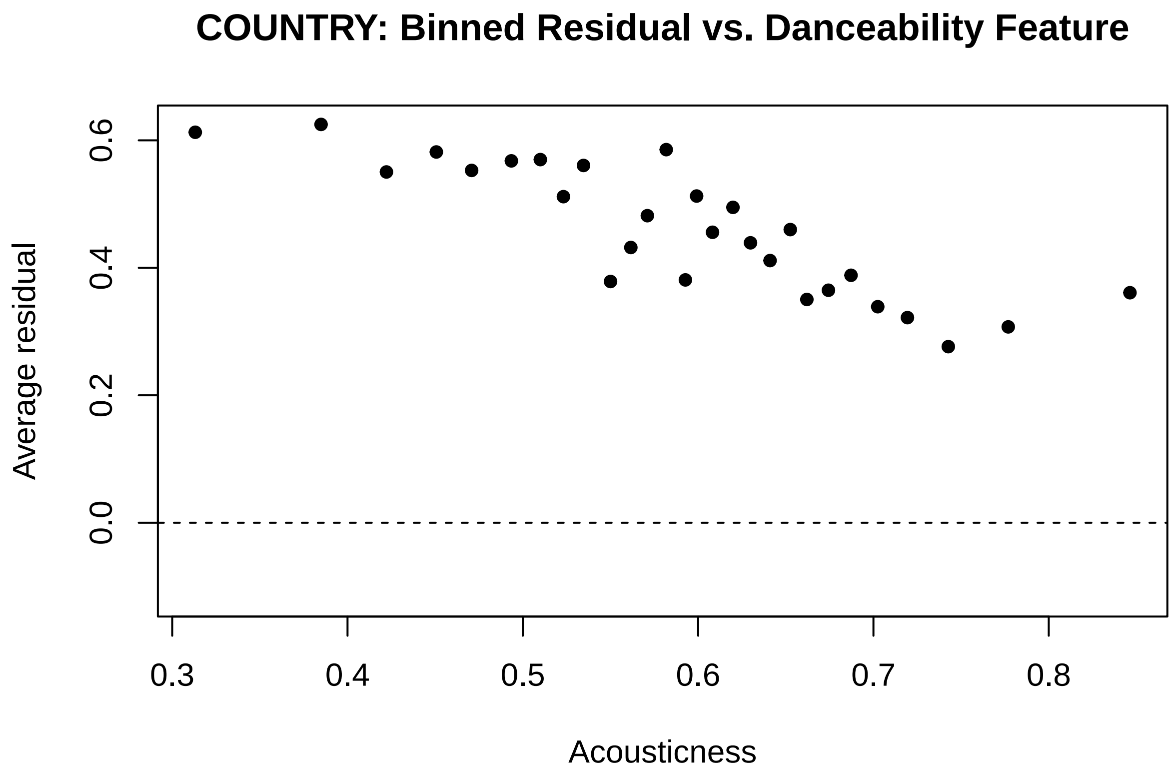

```r
full_m_aug_edm <- full_m_aug %>% filter(genre == "edm_dance")
nbins <- sqrt(nrow(full_m_aug_edm))
arm::binnedplot(x = full_m_aug_edm$liveness, y = full_m_aug_edm$resid.edm_dance,
                xlab = "Acousticness",
                main = "EDM_DANCE: Binned Residual vs. Danceability Feature",
                col.int = FALSE)
```




```
full_m_aug_other <- full_m_aug %>% filter(genre == "other")
nbins <- sqrt(nrow(full_m_aug_other))
arm::binnedplot(x = full_m_aug_other$liveness, y = full_m_aug_other$resid.other,
                xlab = "Acousticness",
                main = "OTHER: Binned Residual vs. Danceability Feature",
                col.int = FALSE)
```




```
full_m_aug_country <- full_m_aug %>% filter(genre == "country")
nbins <- sqrt(nrow(full_m_aug_country))
arm::binnedplot(x = full_m_aug_country$danceability, y = full_m_aug_country$resid.country,
                xlab = "Acousticness",
                main = "COUNTRY: Binned Residual vs. Danceability Feature",
                col.int = FALSE)
```


COUNTRY: Binned Residual vs. Danceability Feature
Average residual
0.0
0.2
0.4
0.6
0.3
0.4
0.5
0.6
0.7
0.8
Acousticness