CS-GY 6923 MACHINE LEARNING

Professor: Dr. Raman Kannan

Project 1: Exploratory Data Analysis

Author: Dharu Piraba Muguntharaman

Overview of the Dataset:

The dataset chosen is called the music genre dataset. This dataset is taken from Kaggle website: https://www.kaggle.com/datasets/vicsuperman/prediction-of-music-genre. This dataset contains about 10 different genres of music and different features like loudness, tempo, key are used to classify into appropriate genres.

The dataset contains 17 features(independent variables) and 1 target variable(dependent variable). There are a total of 50,005 observations. The dependent variable has about 10 different genres, hence this is a classification problem. Also, the number of features is greater than two, it is a multi-class classification problem. Out of the 17 features, 11 features have numerical values and remaining 6 features contain categorical values. These categorical data are transformed into numerical values by one hot encoding in the later part. The dataset contains null values and missing values which will be dealt in the data processing part.

The columns of the dataset consists of 18 fields which are representative of a particular song in the dataset. The columns are as follows :

1. **instance_id** : Serial number of the song in the dataset.
2. **artist_name** : Name of the artist of the song.
3. **track_name** : Title of the song.
4. **popularity** : An arbitrary score assigned to the song in the range of 0-100 with 100 being most popular and 0 being least.
5. **acousticness** : This value describes how acoustic a song is. A score of 1.0 means the song is most likely to be an acoustic one.
6. **danceability** : Danceability describes how suitable a track is for dancing based on a combination of musical elements. A value of 0.0 is least danceable and 1.0 is most danceable
7. **duration_ms** : Is the duration in milliseconds of the song.
8. **energy** : Represents how energetic the song is. The range of this field is between [0-1] with 1 being song with highest energy and 0 with lowest.

9. **instrumentalness** : This value represents the amount of vocals in the song. The closer it is to 1.0, the more instrumental the song is
10. **key:**Key of a piece is the group of pitches, or scale, that forms the basis of a music composition.
11. **liveness** : This value describes the probability that the song was recorded with a live audience.
12. **loudness** : Column representing how loud the song is.
13. **mode** : Major and Minor scales that the song is based upon.
14. **speechiness** : Speechiness detects the presence of spoken words in a track.
15. **tempo** : Speed at which the song is being played.
16. **obtained_date** : The date at which the song metadata was retrieved.
17. **valence** : A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive.
18. **music_genre** : The actual category to which the song belongs. This is our target variable.

Loading the Dataset:

The dataset is loaded from a csv file called music_genre.csv and it was loaded into a variable called music_data.

```
music_data=read.csv('/Users/dharupirabamuguntharaman/Downloads/music_genre.csv',na.strings=c("NA", "NULL"))
```

Figure 1: Loading data

To check the correctness of loaded data head and tail commands are used. The head command displays the first five rows of the dataset. The tail command displays the last five rows of the dataset.

```
> head(music_data)
  instance_id          artist_name           track_name popularity acousticness danceability duration_ms energy instrumentalness key
1       32894             Röyksopp Röyksopp's Night Out         27      0.00468        0.652          -1  0.941          0.79200  A#
2       46652 Thievery Corporation     The Shining Path         31      0.01270        0.622      218293  0.890          0.95000   D
3       30097       Dillon Francis            Hurricane         28      0.00306        0.620      215613  0.755          0.01180  G#
4       62177             Dubloadz               Nitro         34      0.02540        0.774      166875  0.700          0.00253  C#
5       24907          What So Not     Divide & Conquer         32      0.00465        0.638      222369  0.587          0.90900  F#
6       89064           Axel Boman                Hello         47      0.00523        0.755      519468  0.731          0.85400   D
  liveness loudness  mode speechiness            tempo obtained_date valence music_genre
1    0.115   -5.201 Minor      0.0748          100.889         4-Apr   0.759  Electronic
2    0.124   -7.043 Minor      0.0300 115.00200000000001         4-Apr   0.531  Electronic
3    0.534   -4.617 Major      0.0345          127.994         4-Apr   0.333  Electronic
4    0.157   -4.498 Major      0.2390          128.014         4-Apr   0.270  Electronic
5    0.157   -6.266 Major      0.0413          145.036         4-Apr   0.323  Electronic
6    0.216  -10.517 Minor      0.0412                ?         4-Apr   0.614  Electronic
```

Figure 2: Data entries displayed by head command

```
> tail(music_data)
      instance_id       artist_name              track_name popularity acousticness danceability duration_ms energy instrumentalness key
50000       28408      Night Lovell             Barbie Doll         56      0.13300        0.849      237667  0.660         7.96e-06   C
50001       58878             BEXEY               GO GETTA         59      0.03340        0.913          -1  0.574         0.00e+00  C#
50002       43557         Roy Woods      Drama (feat. Drake)         72      0.15700        0.709      251860  0.362         0.00e+00   B
50003       39767            Berner Lovin' Me (feat. Smiggz)         51      0.00597        0.693      189483  0.763         0.00e+00   D
50004       57944         The-Dream        Shawty Is Da Shit         65      0.08310        0.782      262773  0.472         0.00e+00   G
50005       63470 Naughty By Nature          Hip Hop Hooray         67      0.10200        0.862      267267  0.642         0.00e+00  F#
      liveness loudness  mode speechiness             tempo obtained_date valence music_genre
50000    0.296   -7.195 Major      0.0516             99.988         4-Apr   0.629     Hip-Hop
50001    0.119   -7.022 Major      0.2980  98.02799999999999         4-Apr   0.330     Hip-Hop
50002    0.109   -9.814 Major      0.0550 122.04299999999999         4-Apr   0.113     Hip-Hop
50003    0.143   -5.443 Major      0.1460            131.079         4-Apr   0.395     Hip-Hop
50004    0.106   -5.016 Minor      0.0441  75.88600000000001         4-Apr   0.354     Hip-Hop
50005    0.272  -13.652 Minor      0.1010  99.20100000000001         4-Apr   0.765     Hip-Hop
>
```

Figure 3: Data entries displayed by tail command

The dim command is performed to identify the total number of rows and columns in the dataset. The output returns 50,005 and 18 which is in accordance with the total of the actual observations in the dataset.

```
> dim(music_data)
[1] 50005    18
```

Figure 4: Total Dimensions of the dataset

To verify the correctness of the columns loaded, the names command is used. The names command returns the names of the columns or features located in the dataset. Similarly, the length command is used to identify the total number of columns in the dataset. This dataset contains a total of 18 columns.

```
> names(music_data)
 [1] "instance_id"      "artist_name"      "track_name"       "popularity"       "acousticness"     "danceability"     "duration_ms"
 [8] "energy"           "instrumentalness" "key"              "liveness"         "loudness"         "mode"             "speechiness"
[15] "tempo"            "obtained_date"    "valence"          "music_genre"
> length(names(music_data))
[1] 18
```

Figure 5: Names of all columns and length of the dataset

The which command is used here to identify the position of our target variable – music_genre. This command returns 18 which indicates that the target variable is the last column of the dataset.

```
> which(names(music_data)=='music_genre')
[1] 18
```

Figure 6: Last column of the dataset is identified as target variable

Here, omit command is used to discard any rows containing null values.

```
> music_data=na.omit(music_data)
> dim(music_data)
[1] 50000    18
```

Figure 7: Dimensions of the dataset reduced due to removing null values

Label Analysis:

The type of genre of music of each data frame is given at the last column of the dataset. Since, the dataset contains a large number of obsevations and the target variable is categorical, classification algorithms must be used.

To check if the type of classification problem, the number of categories of the target variable is used. If the target variable has more than two categories it is a multi class classification problem. To identify this, the length command is used. The length returns the count of the number of classes in the given variable. Here, target variable is passed to length command and the output is checked using a if statement. Since, the dataset contains 10 different classes, it is a multi-class classification problem.

```
> lbl=table(music_data$music_genre)
> label_names=names(lbl)
> label_count=as.numeric(lbl)
> label_names
 [1] "Alternative" "Anime"       "Blues"       "Classical"   "Country"     "Electronic"  "Hip-Hop"     "Jazz"        "Rap"
[10] "Rock"
> label_count
 [1] 5000 5000 5000 5000 5000 5000 5000 5000 5000 5000
> print(ifelse(length(lbl)==2,"Binary Classification","Multiclass Classification"))
[1] "Multiclass Classification"
```

Figure 8: Dataset contains multi class target variables

Imbalance class Identification:

An imbalanced classification problem is an example of a classification problem where the distribution of examples across the known classes is biased or skewed. This results in models that have poor predictive performance, specifically for the minority class. Hence, it is necessary to check the dataset for imbalance classification.

A function called Imbalance is created to determine if the dataset is imbalanced or not. This function takes the dataframe and checks the labels for imbalanced classes.

```
> Imbalance=function(data,labels,numUniqueLabels){
+   rows=nrow(data)
+   ideal_rows=rows*1/numUniqueLabels
+   flag=0
+   for (i in 1:numUniqueLabels){
+     if(labels[[i]]<ideal_rows*1/(numUniqueLabels/2)|labels[[i]]>ideal_rows*(numUniqueLabels/2) ){
+       print("Imbalanced class")
+       print(names(labels)[i])
+       flag=flag+1
+     }
+   }
+   if (flag==0){
+     print("No Imbalance class found in the dataset")
+   }
+ }
> Imbalance(music_data,lbl,10)
[1] "No Imbalance class found in the dataset"
>
```

Figure 9: Dataset found to have a balanced class

From the above output, it can be seen that the dataset if free from imbalances. All the 10 genres have 5000 observations each.

Checking for null values in the Dataset:

One if the important steps in data cleaning is identify missing values in the dataset and handle them. Missing data will lead to errors in the performance of the training model. Hence, it is necessary to check for null values and treat them.

The function Find_null checks if the dataset contains any null values. From, the output it is seen that, the dataset is free from null values as well.

```
>
> Find_null=function(data){
+   if(is.null(data)){
+     print("Dataset contains null values")
+   }else{
+     print("Dataset contains no null values")
+   }
+ }
> Find_null(music_data)
[1] "Dataset contains no null values"
>
```

Figure 10: Dataset has no null values

Dataset summary:

The summary command is called to provide a general summary of the dataset. The table summarises the mean, median, minimum value, maximum value, 1st quartile and 3rd quartile values of each of the features. This is useful in understanding about the general characteristics of the numerical features. For categorical features, this is not useful.

```
> summary(music_data)
  instance_id    artist_name        track_name         popularity     acousticness     danceability     duration_ms
 Min.   :20002   Length:50000       Length:50000       Min.   : 0.00   Min.   :0.0000   Min.   :0.0596   Min.   :    -1
 1st Qu.:37974   Class :character   Class :character   1st Qu.:34.00   1st Qu.:0.0200   1st Qu.:0.4420   1st Qu.: 174800
 Median :55914   Mode  :character   Mode  :character   Median :45.00   Median :0.1440   Median :0.5680   Median : 219281
 Mean   :55888                                         Mean   :44.22   Mean   :0.3064   Mean   :0.5582   Mean   : 221253
 3rd Qu.:73863                                         3rd Qu.:56.00   3rd Qu.:0.5520   3rd Qu.:0.6870   3rd Qu.: 268612
 Max.   :91759                                         Max.   :99.00   Max.   :0.9960   Max.   :0.9860   Max.   :4830606
     energy        instrumentalness       key             liveness         loudness          mode          speechiness
 Min.   :0.000792   Min.   :0.000000   Length:50000       Min.   :0.00967   Min.   :-47.046   Length:50000       Min.   :0.02230
 1st Qu.:0.433000   1st Qu.:0.000000   Class :character   1st Qu.:0.09690   1st Qu.:-10.860   Class :character   1st Qu.:0.03610
 Median :0.643000   Median :0.000158   Mode  :character   Median :0.12600   Median : -7.277   Mode  :character   Median :0.04890
 Mean   :0.599755   Mean   :0.181601                      Mean   :0.19390   Mean   : -9.134                      Mean   :0.09359
 3rd Qu.:0.815000   3rd Qu.:0.155000                      3rd Qu.:0.24400   3rd Qu.: -5.173                      3rd Qu.:0.09853
 Max.   :0.999000   Max.   :0.996000                      Max.   :1.00000   Max.   :  3.744                      Max.   :0.94200
    tempo          obtained_date        valence       music_genre
 Length:50000       Length:50000       Min.   :0.0000   Length:50000
 Class :character   Class :character   1st Qu.:0.2570   Class :character
 Mode  :character   Mode  :character   Median :0.4480   Mode  :character
                                       Mean   :0.4563
                                       3rd Qu.:0.6480
                                       Max.   :0.9920
>
```

Figure 11: Summary of the dataset

Removing un-useful features:

From the above summary, it can be seen that the features instance_id, artist_name , track_name and obtained_date can be removed since they don't hold any helpful .

```
> music_data=music_data[-c(1,2,3,16)]
> dim(music_data)
[1] 50000    14
```

Figure 12: Certain columns are removed

After removing these columns, the dataset has about 50,000 rows and 14 columns.

```
> summary(music_data)
   popularity     acousticness     danceability     duration_ms          energy         instrumentalness       key
 Min.   : 0.00   Min.   :0.0000   Min.   :0.0596   Min.   :     -1   Min.   :0.000792   Min.   :0.000000   Length:50000
 1st Qu.:34.00   1st Qu.:0.0200   1st Qu.:0.4420   1st Qu.: 174800   1st Qu.:0.433000   1st Qu.:0.000000   Class :character
 Median :45.00   Median :0.1440   Median :0.5680   Median : 219281   Median :0.643000   Median :0.000158   Mode  :character
 Mean   :44.22   Mean   :0.3064   Mean   :0.5582   Mean   : 221253   Mean   :0.599755   Mean   :0.181601
 3rd Qu.:56.00   3rd Qu.:0.5520   3rd Qu.:0.6870   3rd Qu.: 268612   3rd Qu.:0.815000   3rd Qu.:0.155000
 Max.   :99.00   Max.   :0.9960   Max.   :0.9860   Max.   :4830606   Max.   :0.999000   Max.   :0.996000
    liveness          loudness          mode           speechiness         tempo              valence       music_genre
 Min.   :0.00967   Min.   :-47.046   Length:50000       Min.   :0.02230   Length:50000       Min.   :0.0000   Length:50000
 1st Qu.:0.09690   1st Qu.:-10.860   Class :character   1st Qu.:0.03610   Class :character   1st Qu.:0.2570   Class :character
 Median :0.12600   Median : -7.277   Mode  :character   Median :0.04890   Mode  :character   Median :0.4480   Mode  :character
 Mean   :0.19390   Mean   : -9.134                      Mean   :0.09359                      Mean   :0.4563
 3rd Qu.:0.24400   3rd Qu.: -5.173                      3rd Qu.:0.09853                      3rd Qu.:0.6480
 Max.   :1.00000   Max.   :  3.744                      Max.   :0.94200                      Max.   :0.9920
>
```

Figure 13: Summary of the new dataset

Dataset Visualization:

Dataset visualization is an important tool in understanding the data behavior and constructing machine learning models. All the features are plotted against different music genres to understand each.

The popularity feature tells the popularity of a song. From the below plot, it can be seen that Rap, Hip hop and Rock are more popular compared to other genres. Also, a lot of outliers are found in all the genres with Jazz having the most outliers.

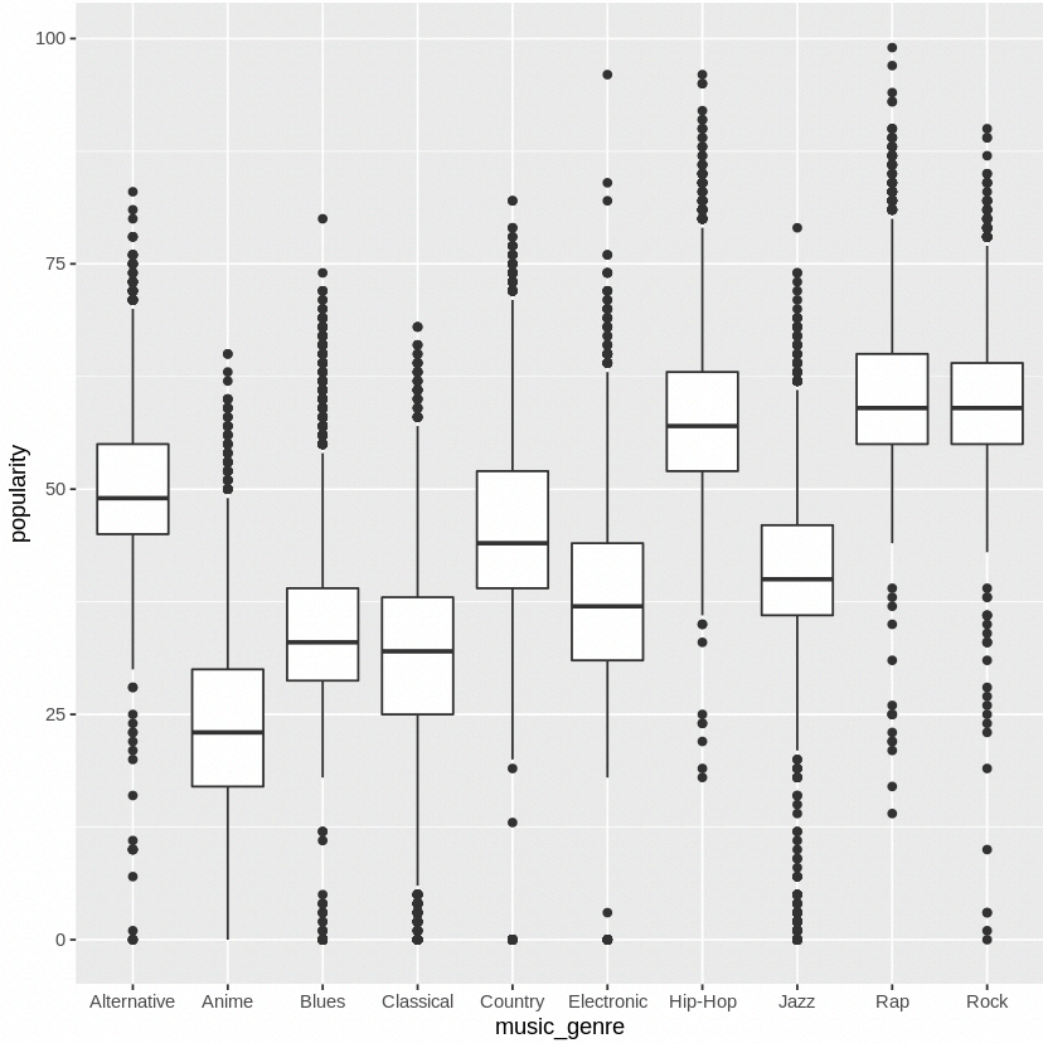

Figure 14: Popularity vs music genres

The acousticness feature is used to determine if a song is more acoustic or not. From the plot below, it could be seen that Classical music has the most outlier. Jazz is found to be more acoustic compared to other genres followed by anime.

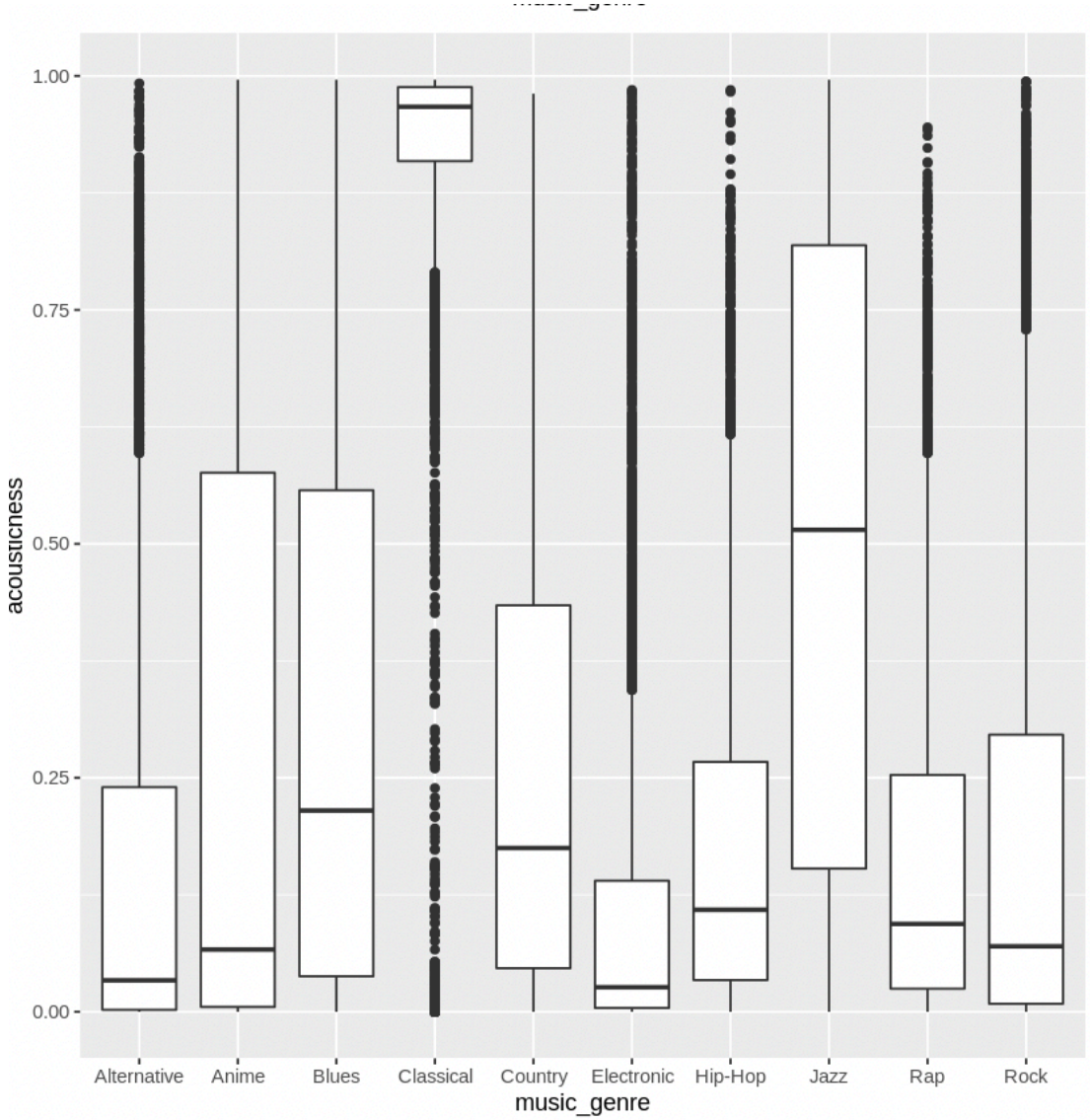

Figure 14: Acousticness vs music genres

The danceability feature is used to determine if a song is more suited for dance or not. From the plot below, it could be seen that Hip hop and Rap are found to be more danceable. Classical is least danceable.

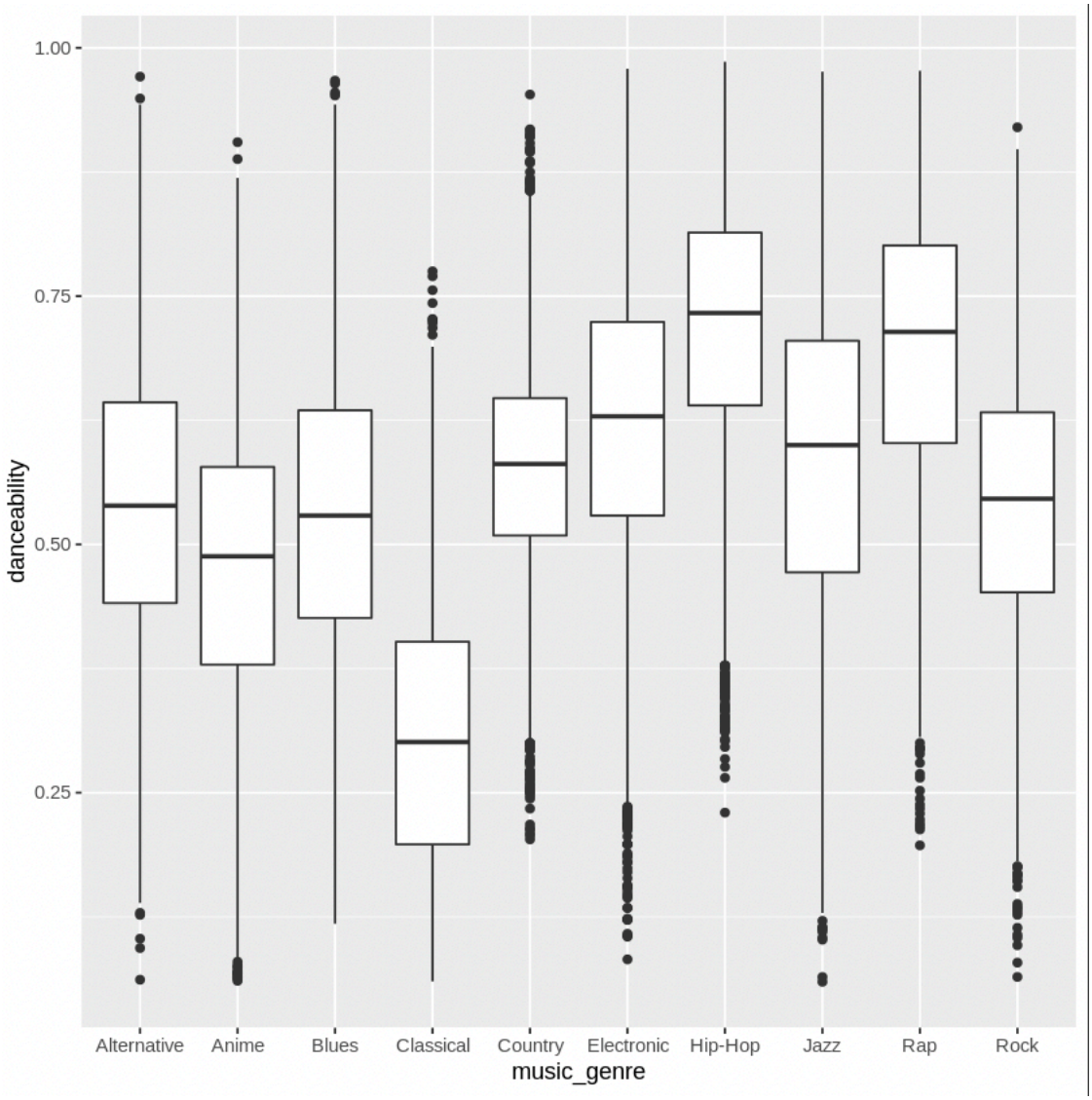

Figure 15: Danceability vs music genres

The duration_ms feature is used to measure the duration of a song. From the plot below, it could be seen that almost all genres have the same duration of songs. Classical and Electronic genres have some outliers.

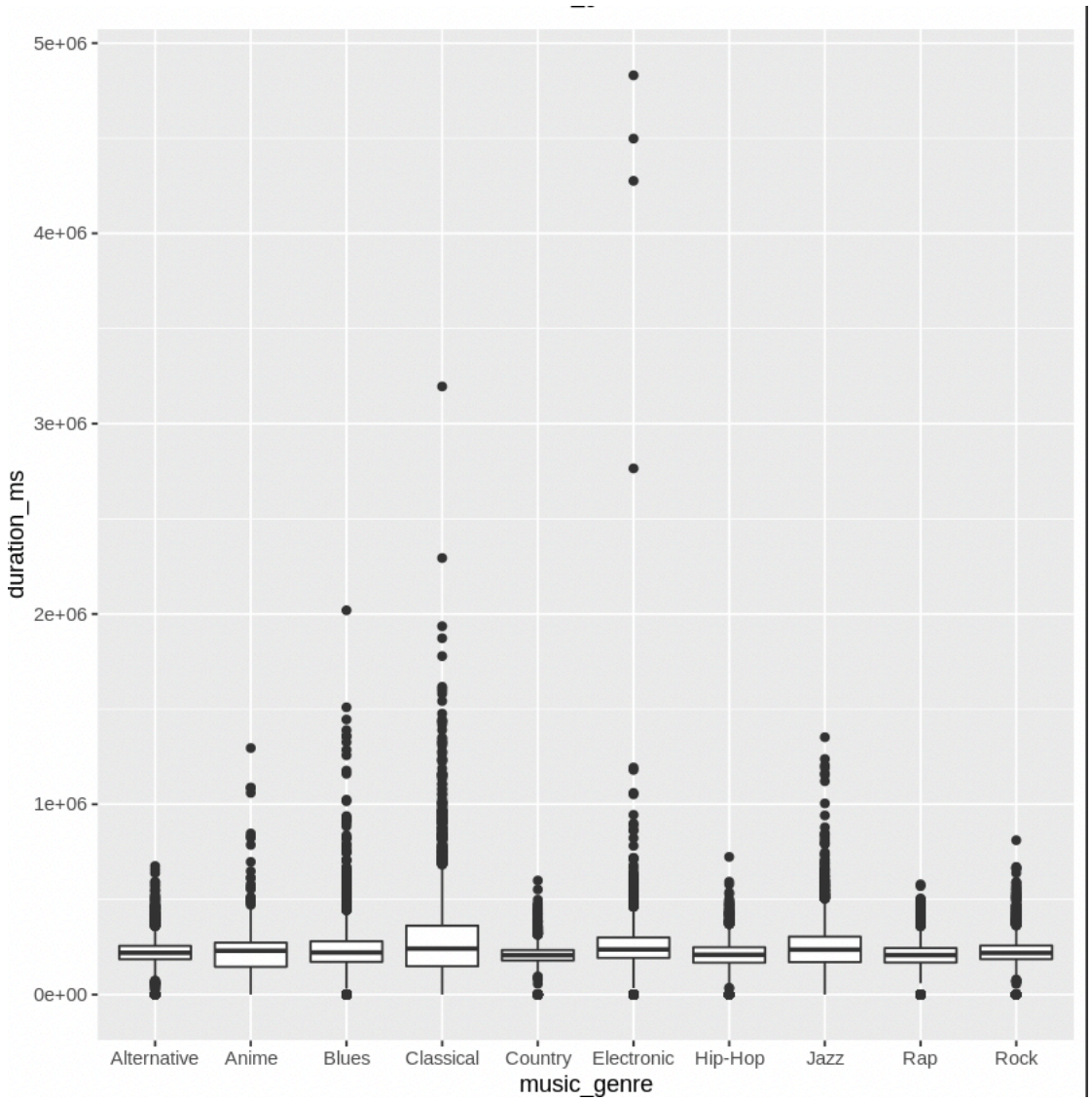

Figure 16: Duration vs music genres

The energy feature is used to determine if a song is more intense or energetic or not. From the plot below, it could be seen that Hip hop and Rap are found to be similar in energy. Classical music is least energetic. Anime genre is found to have higher energy.

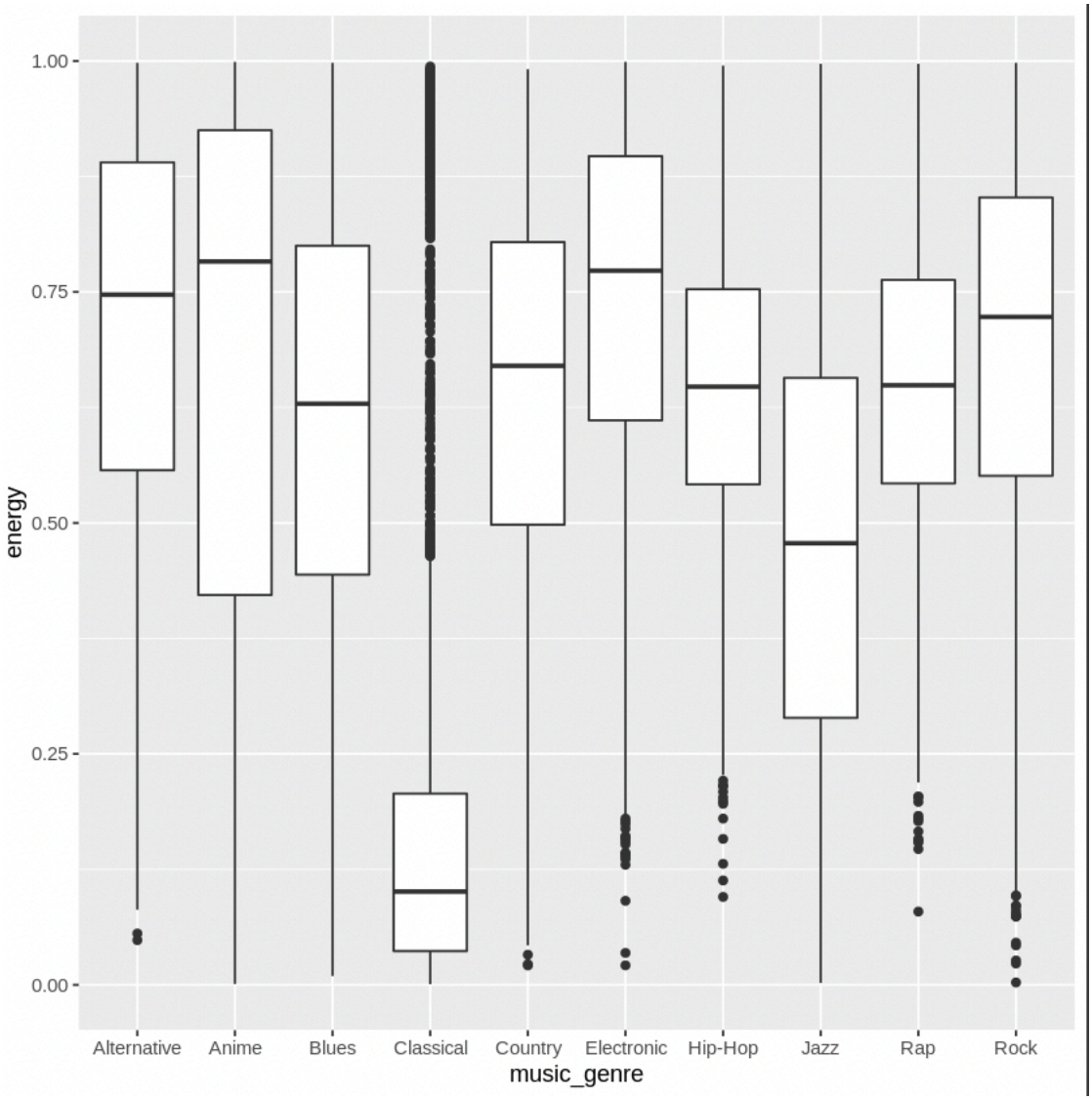

Figure 17: Energy vs music genres

The instrumental feature is used to determine if a song has more vocals or instruments. From the plot below, it could be seen that a lot genres have null values. This means that a lot of genres have more vocals compared to genres. Classical music is found to have more instruments.

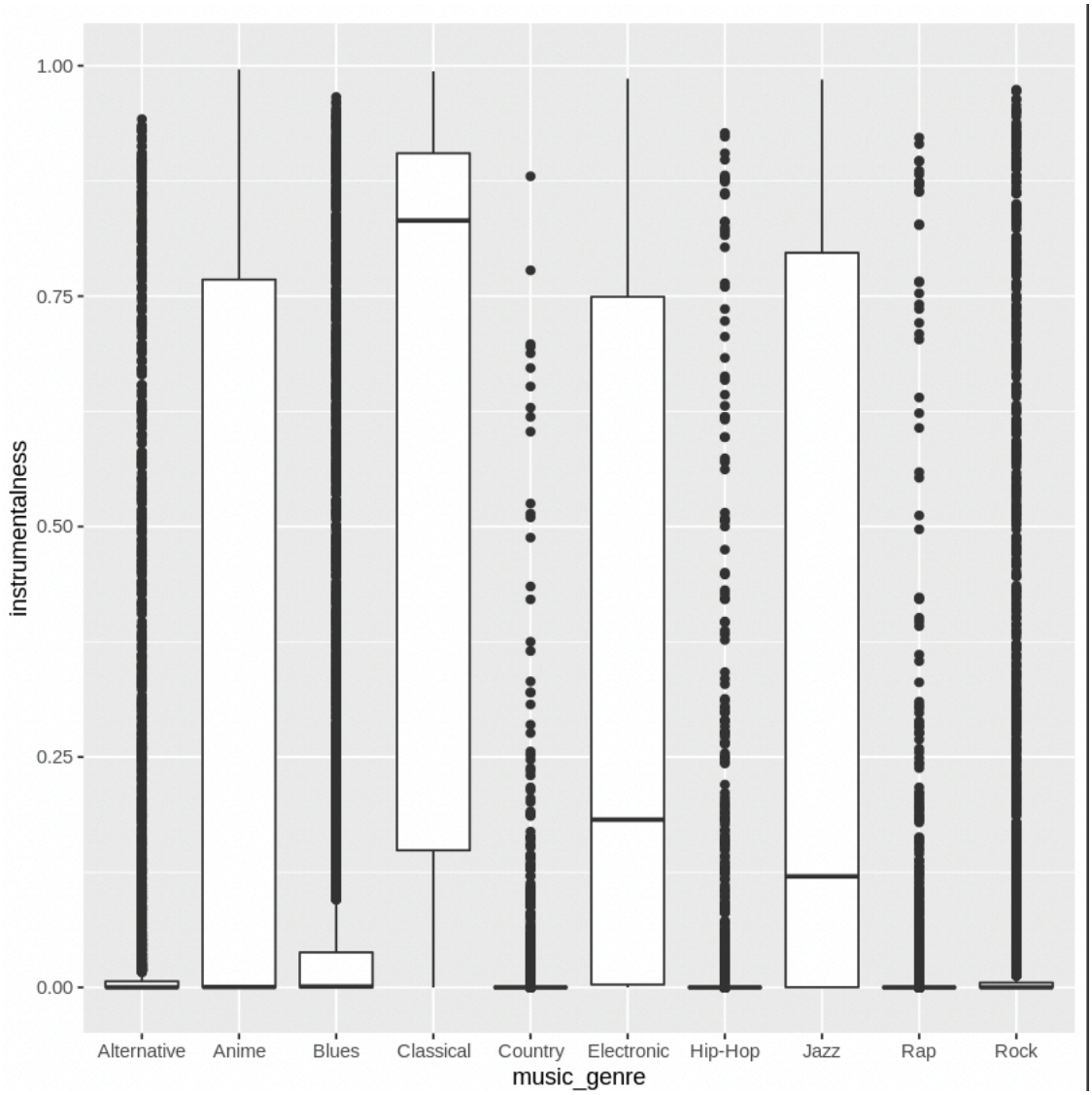

Figure 18: Instrumentalness vs music genres

The key feature is used to denote the presence of keys in each song. From the plot below, it could be seen that it's a categorical feature and hence no vital information could be inferred from the plot. It is necessary to convert them to numerical data.

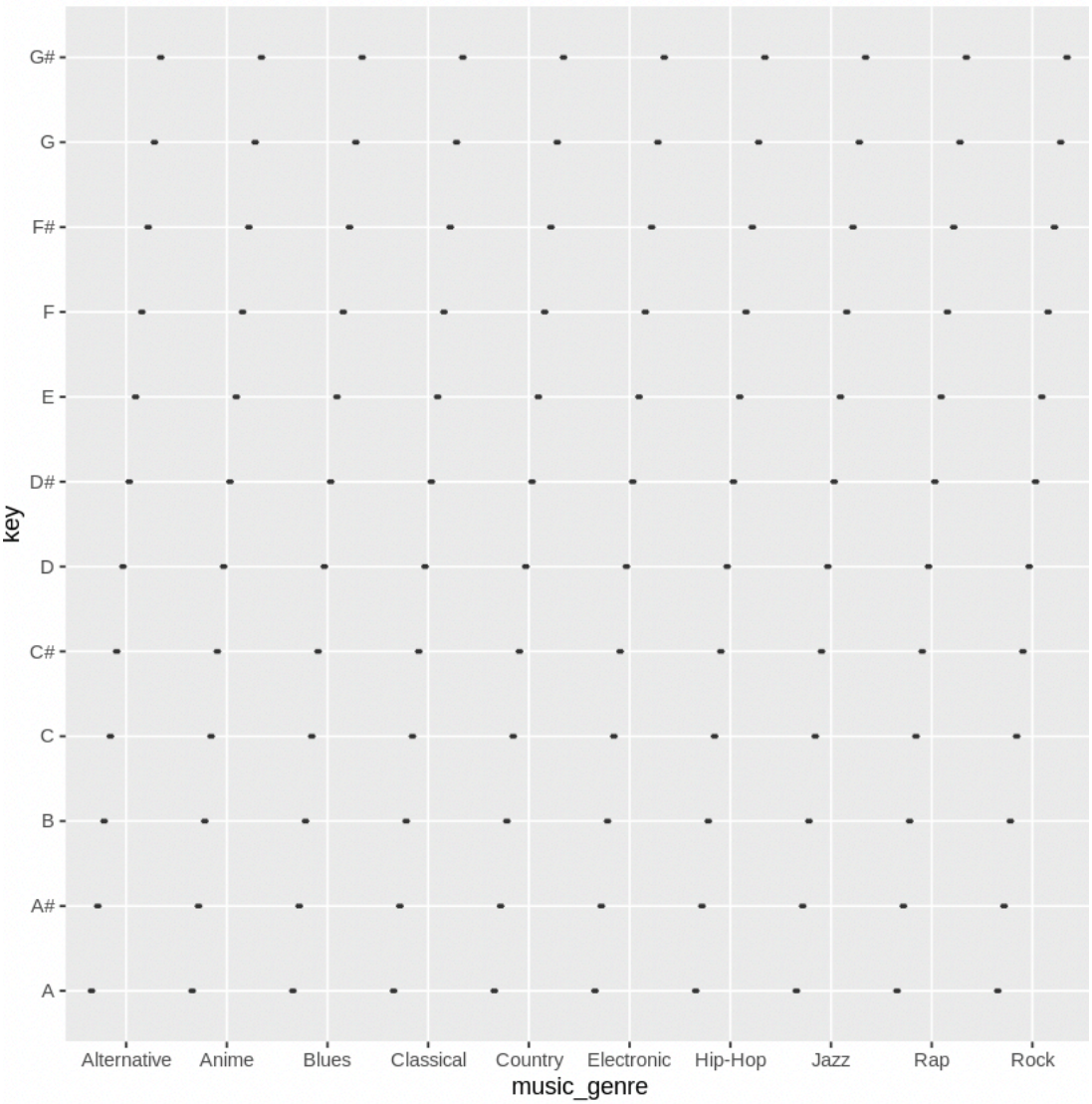

Figure 19: Key vs music genres

The liveness feature is used to determine if audience were present during the song performance. From the plot below, it could be seen that a lot genres have outliers. Majority of the songs from all genres have been performed without an audience.

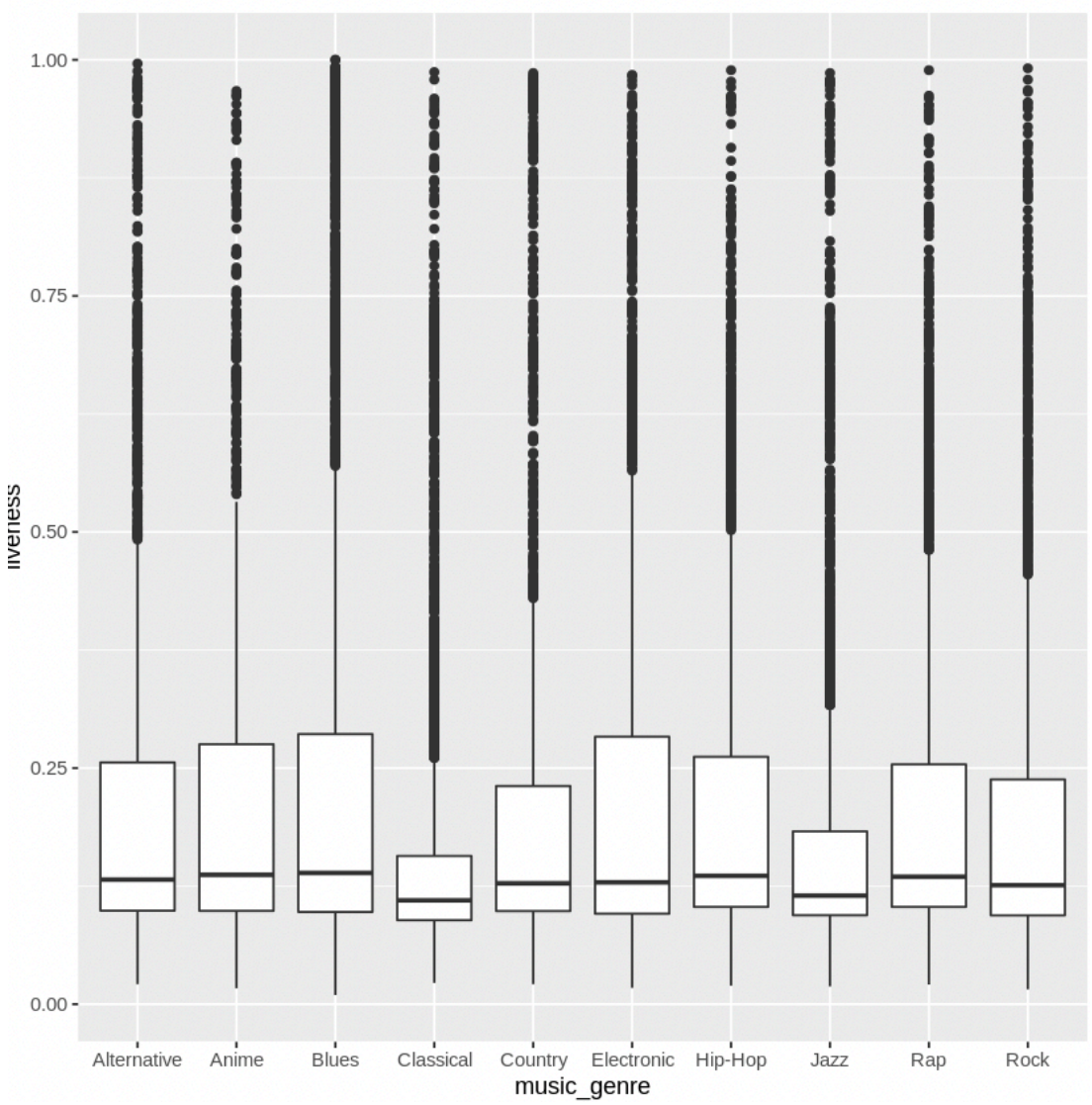

Figure 20: Liveness vs music genres

The loudness feature is used to determine if a song is loud or not. From the plot below, it could be seen that a lot genres have quieter songs. Classical music genre has a much lower loudness level compared to all other genres. Hip hop and Rap have the same loudness level.

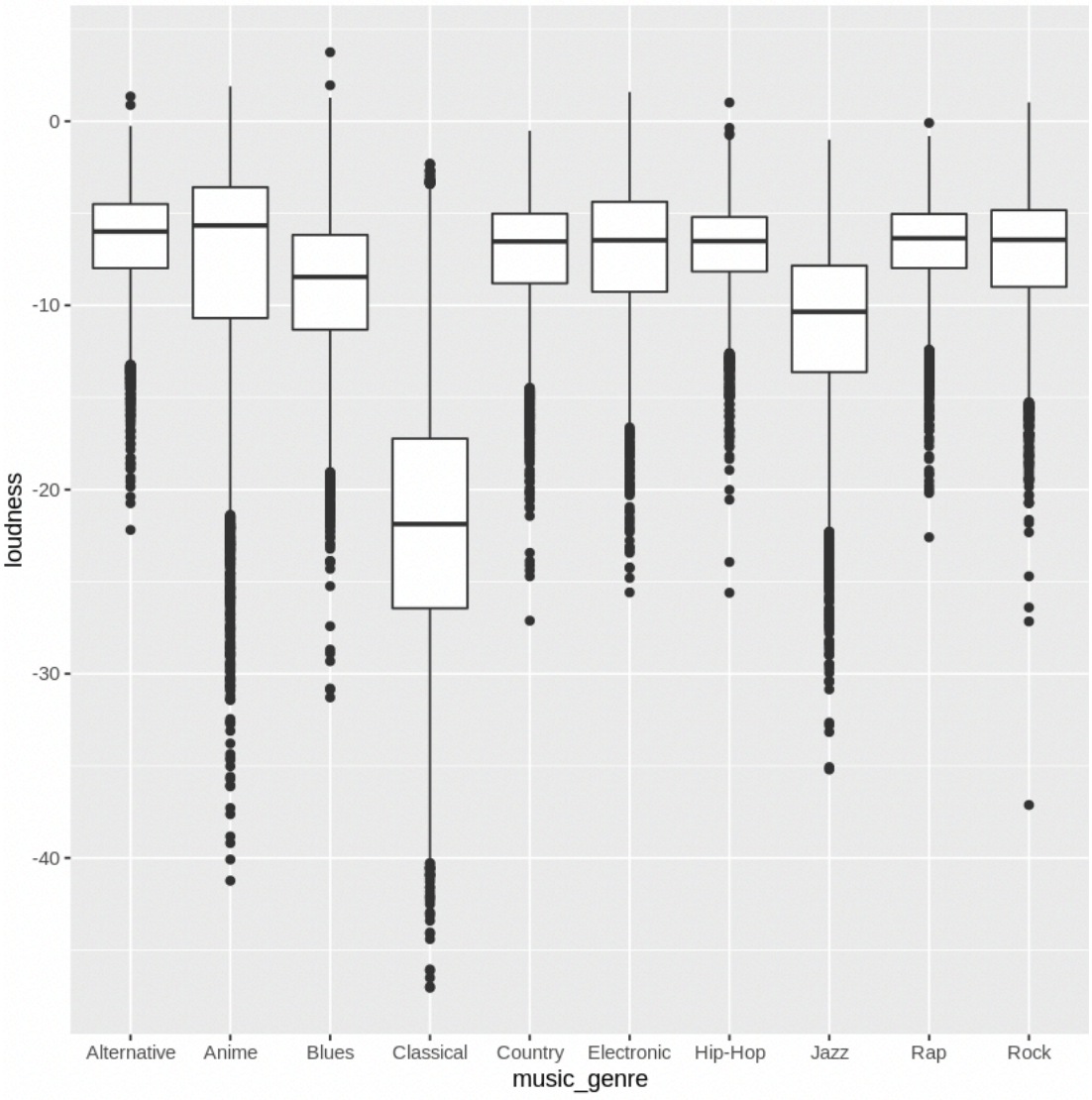

Figure 21: Loudness vs music genres

The mode feature is used to denote the two modes – major and minor. From the plot below, it could be seen that it's a categorical feature and hence no vital information could be inferred from the plot. It is necessary to convert them to numerical data.

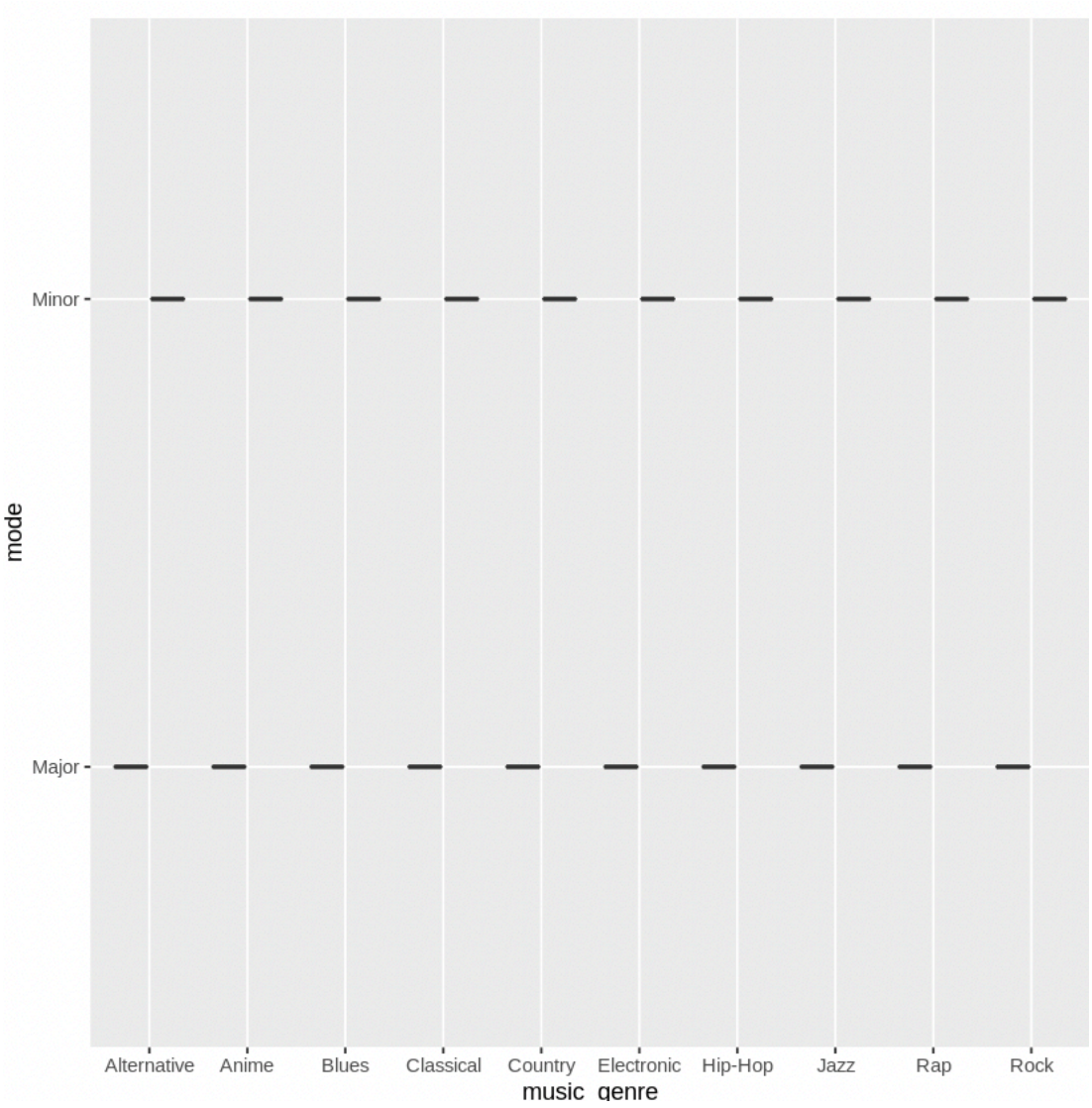

Figure 22: Mode vs music genres

The speechiness feature is used to determine if a song contains more words or not. From the plot below, it could be seen that a lot genres have almost no words. Classical music genre and Country music have almost zero level compared to all other genres. Hip hop and Rap have the most words in a song.

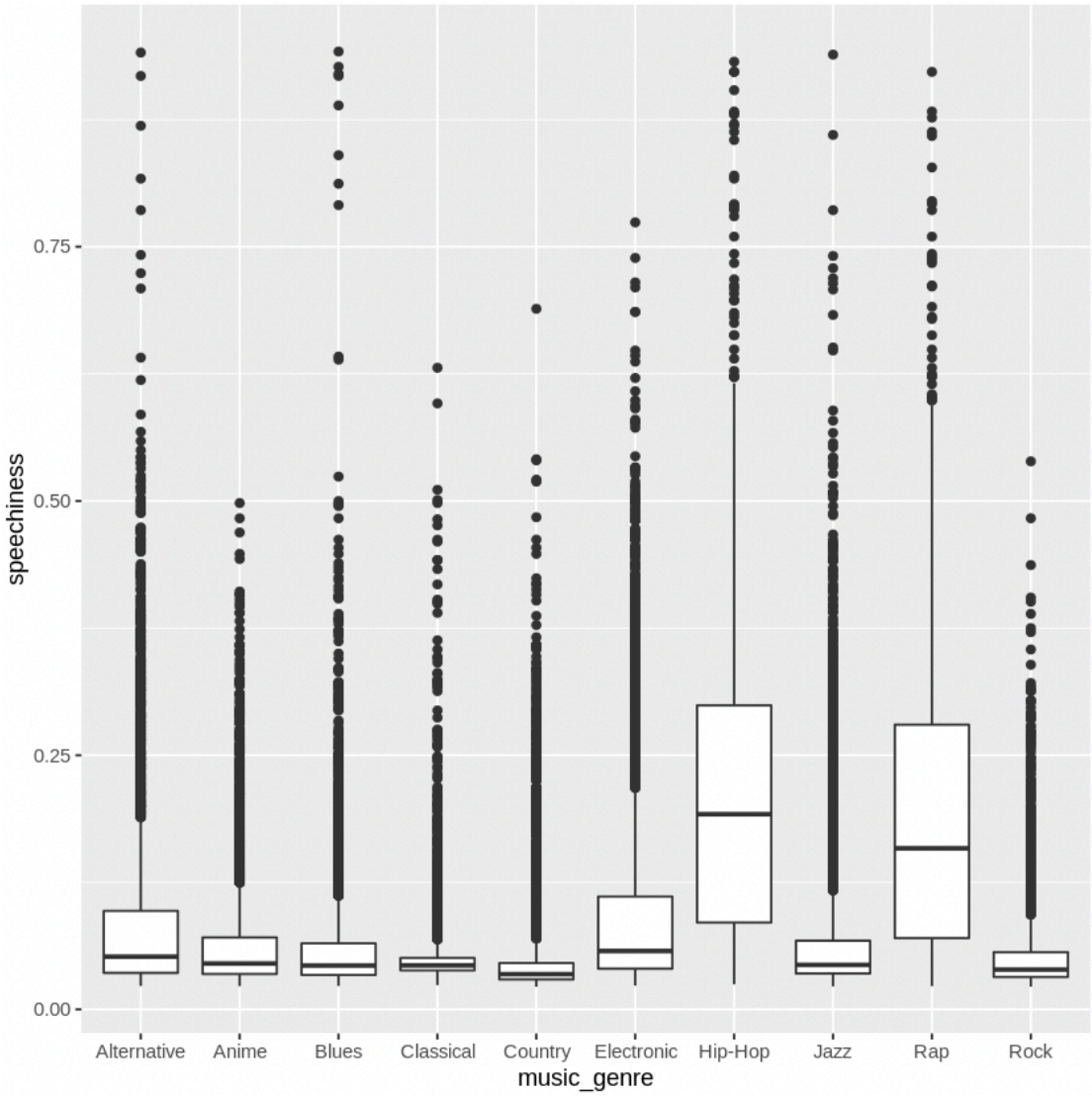

Figure 23: Speechiness vs music genres

The tempo feature is used to measure the beats per minute of the song. From the plot below, it could be seen that it's contains missing values and hence no vital information could be inferred from the plot. It is necessary to convert them to numerical data.

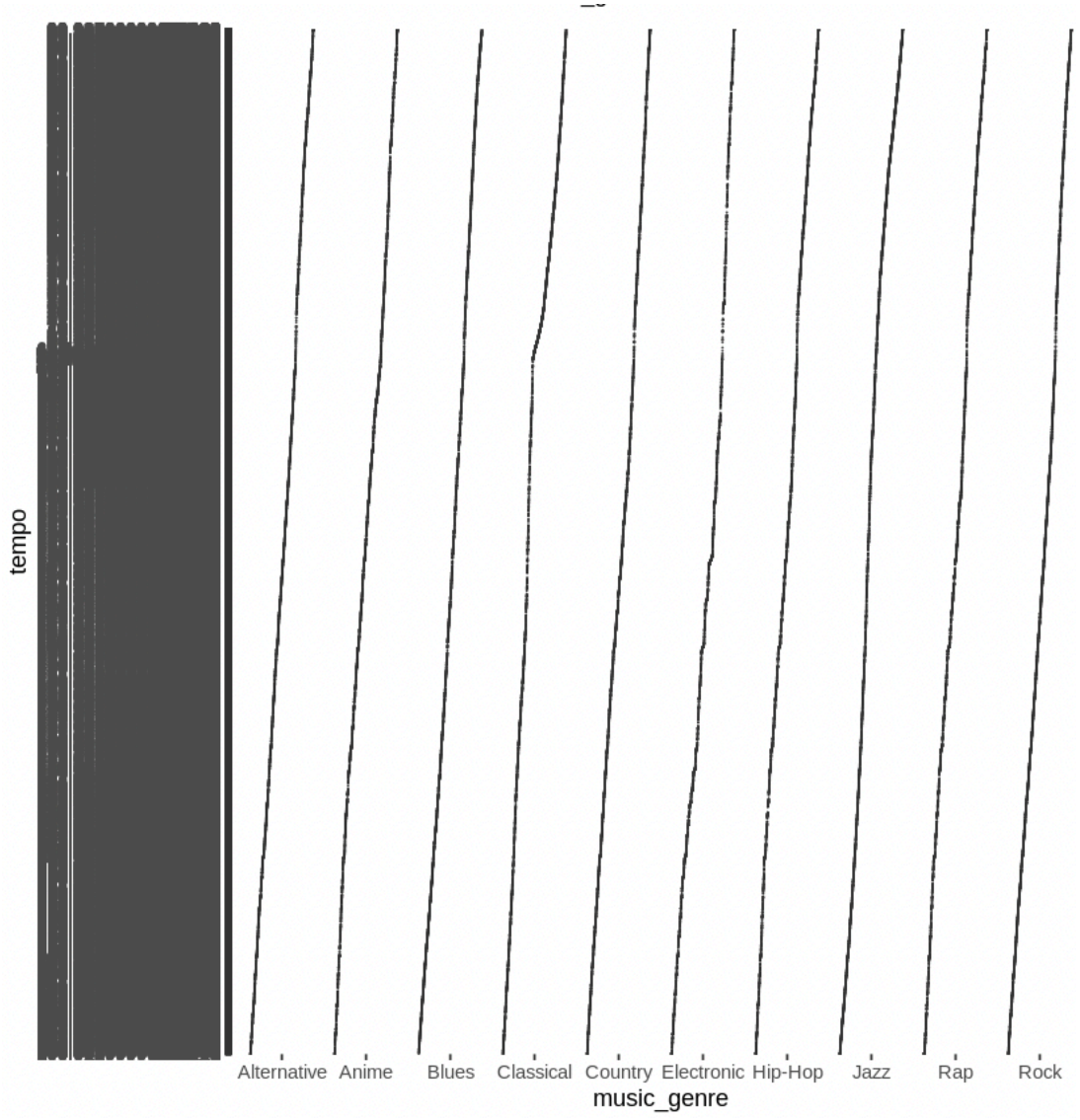

Figure 24: Tempo vs music genres

The valence feature is used to measure the positivity of a song. From the plot below, it could be seen that only Classical music has outliers. Almost all genres have a similar box plot.

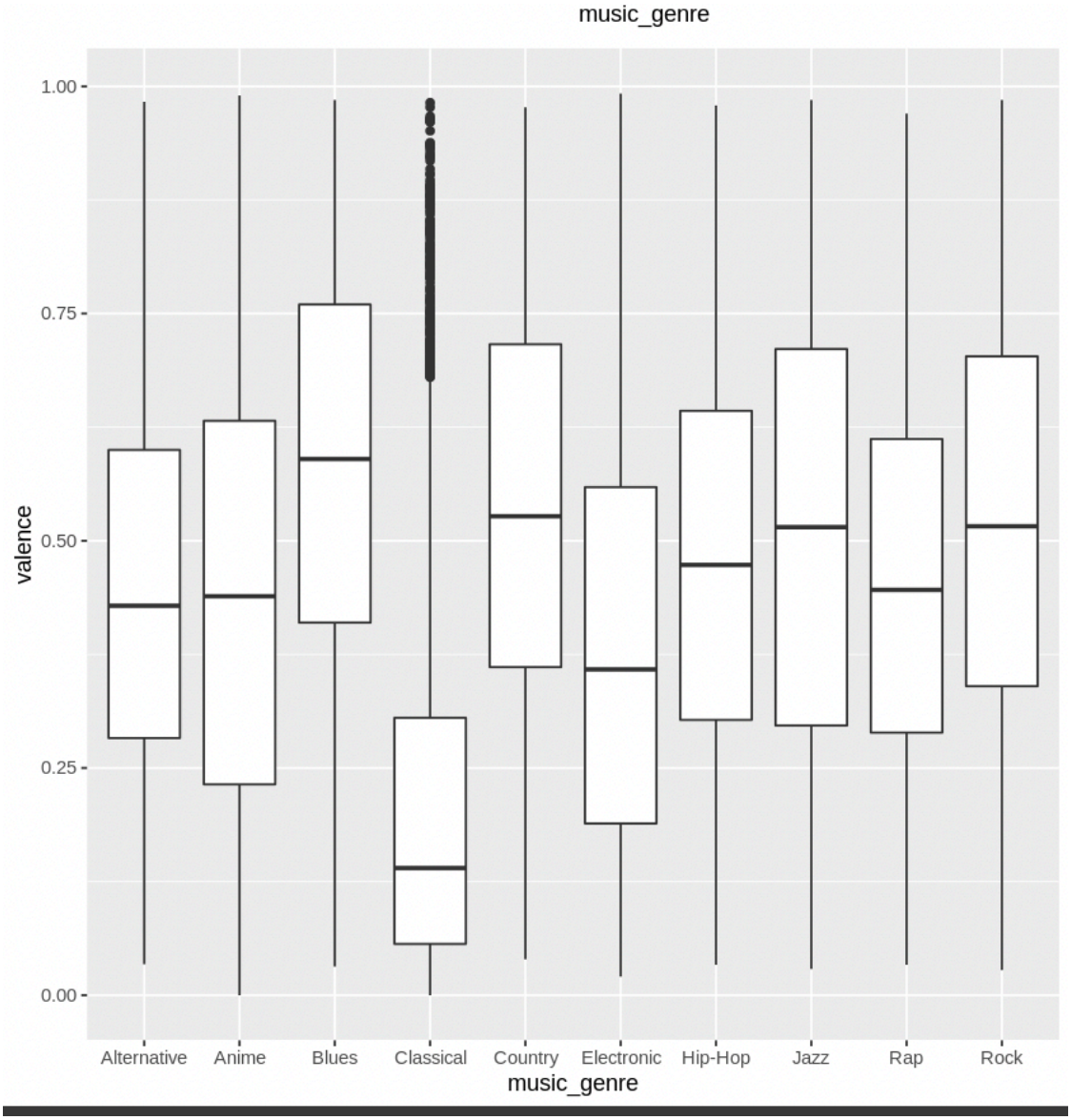

Figure 25: Valence vs music genres

Handling missing values in the Dataset:

From the above dataset visualization, it can be seen that tempo contains some missing values and leads to in correct representations. Hence, it is important to remove the missing values in this column. The rows containing ? in their tempo column are removed. This results in deletion of 5000 rows. The new dimensions of the data are displayed below.

```
> music_data=music_data[music_data$tempo !='?', ]
> dim(music_data)
[1] 45020    14
>
```

Figure 26: Removing rows having missing values

Identifying Categorical and Numerical features:

The dataset visualization for certain columns like key and mode were not proper. This is because, the values of these columns are categorical not numerical. As said before, it is important to identify the numerical and categorical features in a dataset. The is.character() command returns the columns that have categorical data in the dataset. These categorical columns are later converted to numerical using one hot encoding method.

```
[ ] numeric_cols=select_if(music_data, is.numeric)
    categorical_cols=select_if(music_data, is.character)
```

```
[ ] print(names(numeric_cols))
    print(names(categorical_cols))
```

```
 [1] "popularity"      "acousticness"    "danceability"    "duration_ms"
 [5] "energy"          "instrumentalness" "liveness"       "loudness"
 [9] "speechiness"     "valence"
[1] "key"         "mode"        "tempo"       "music_genre"
```

Figure 27: List of categorical and numerical features

Histogram of each numerical feature:

To understand more about the numerical features, we plot a histogram of each feature. The frequency of each of the feature can be useful in classification.

Popularity feature is found to have a bi-modal distribution.

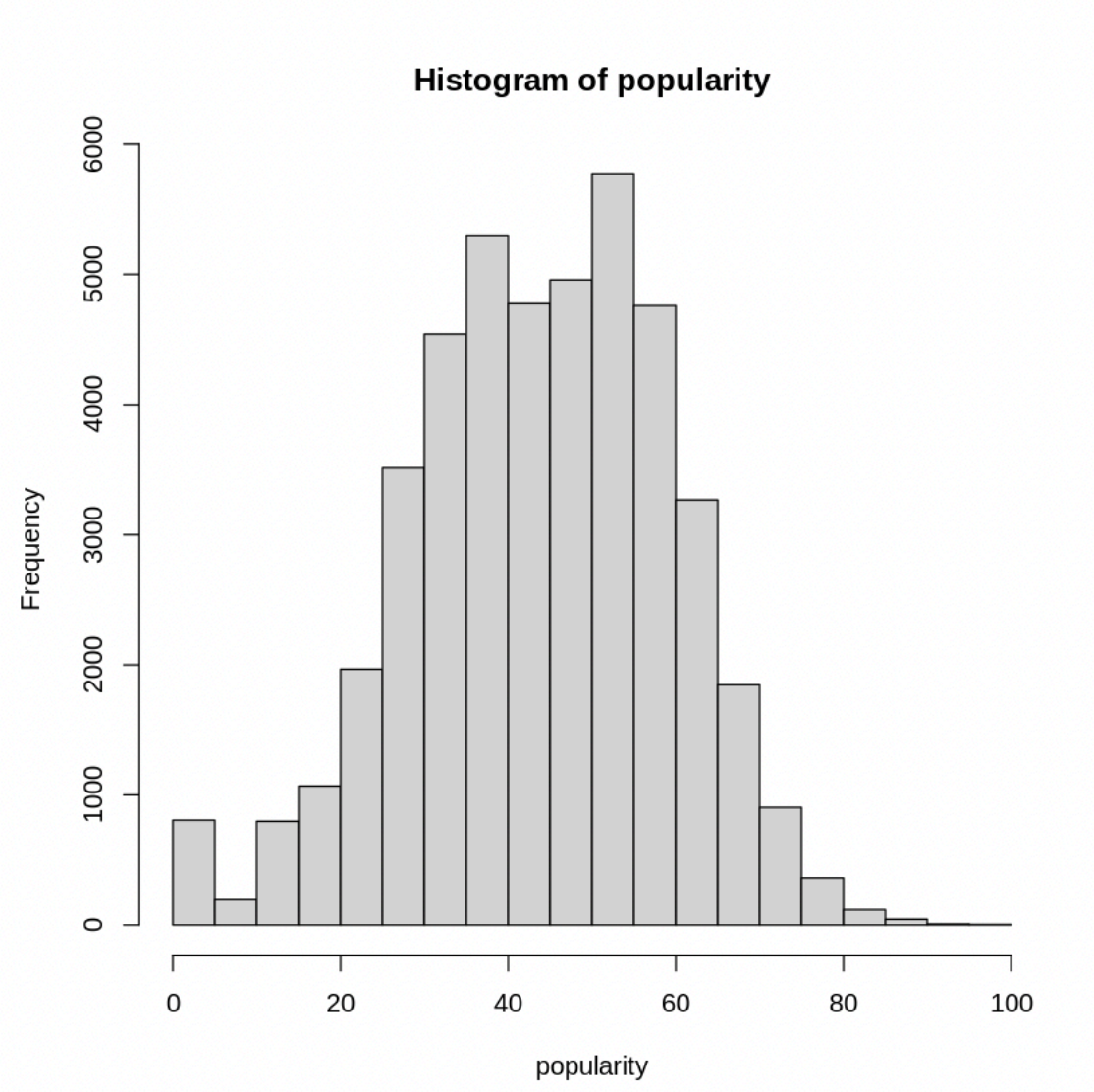

Figure 28: Histogram of popularity feature

Songs with higher acousticness are more likely to use acoustic and non-electronic instruments. This plot shows that most songs are not acoustic in this dataset.

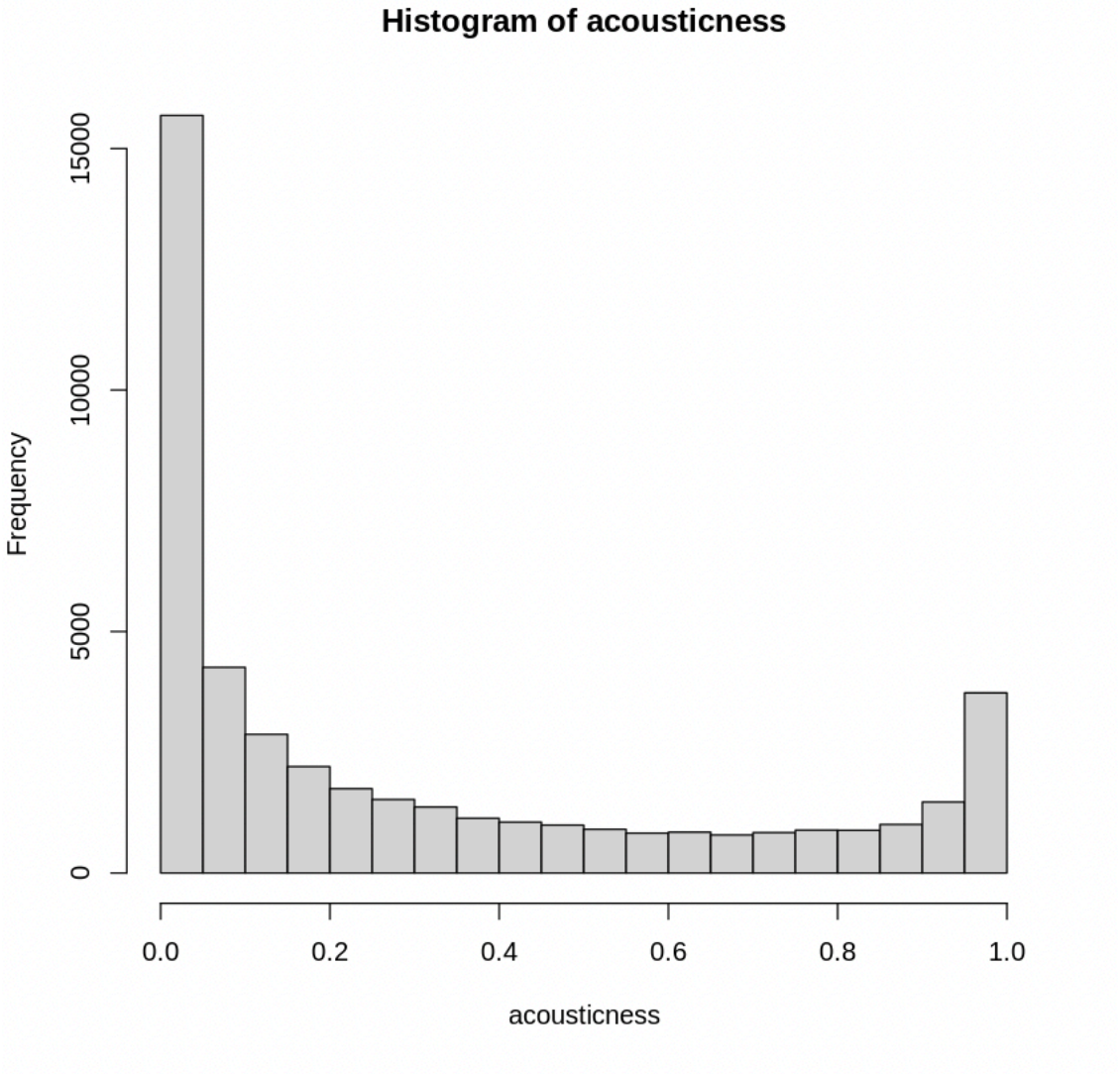

Figure 29: Histogram of acousticness feature

Danceability is measured on a scale of 0.0 (low danceability) to 1.0 (high danceability). In terms of danceability, most of the songs have a normal distribution.

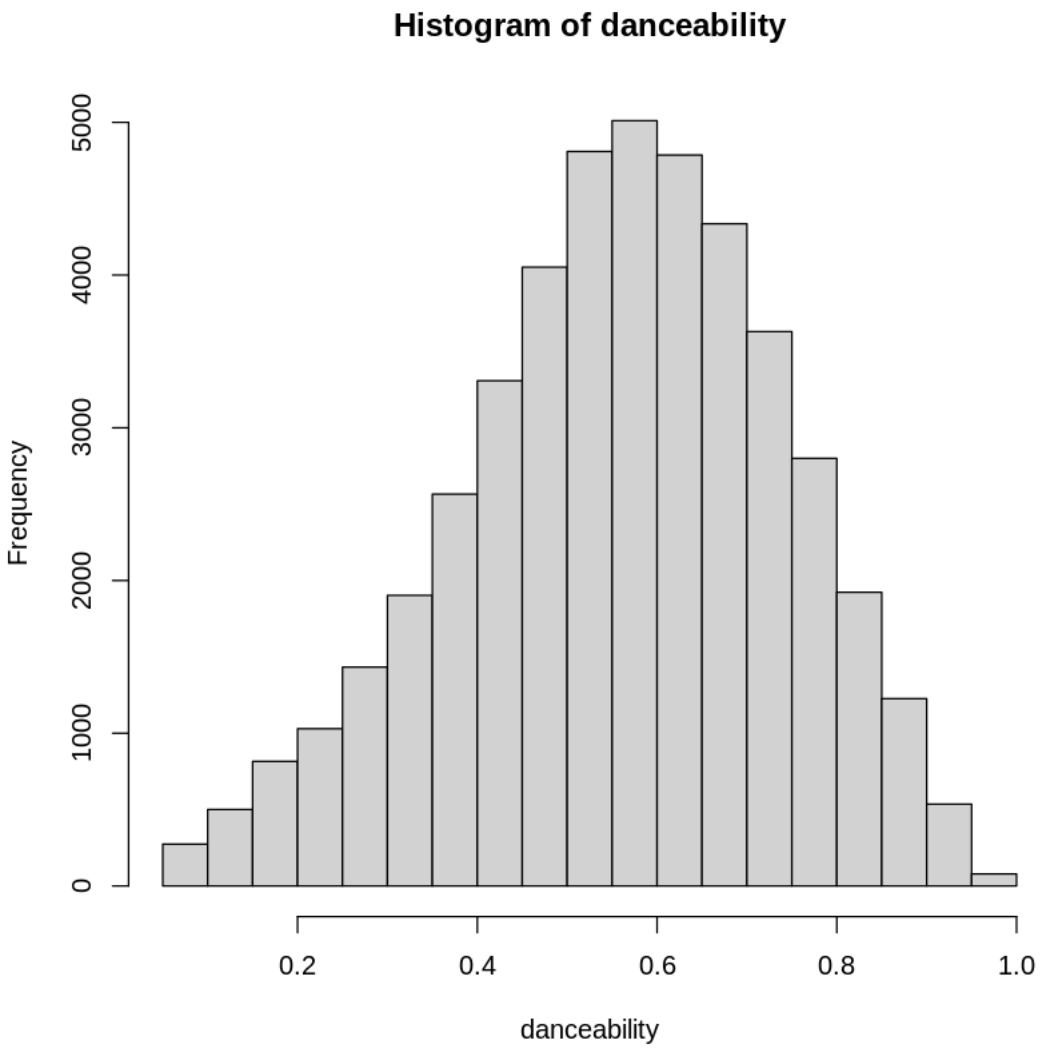

Figure 30: Histogram of danceability feature

Duration of music is given in milliseconds. Almost all the songs are in the same range with a few exception.

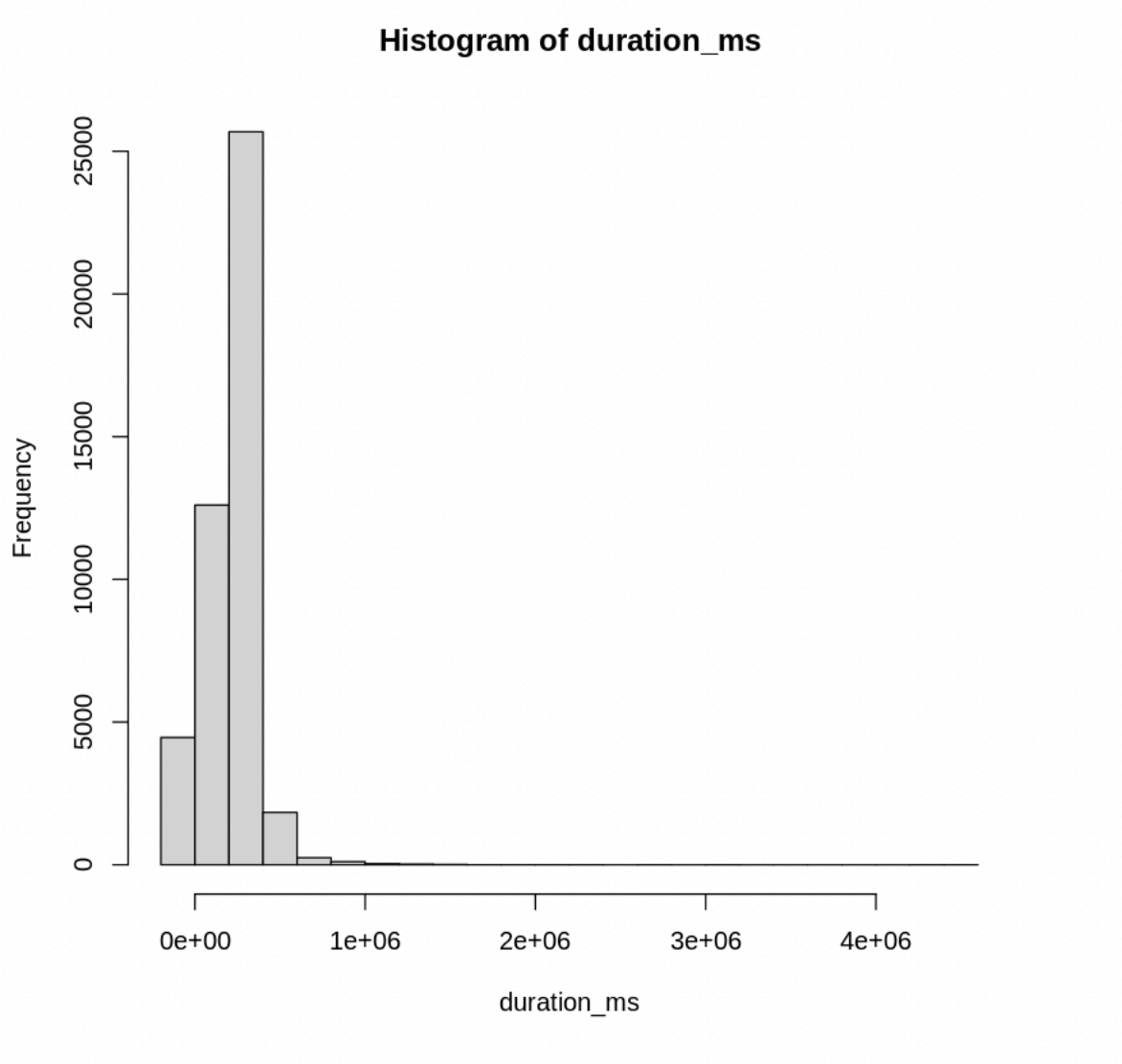

Figure 31: Histogram of duration feature

Songs with higher energy are found to be more intense and loud. The distribution of the histogram shows that most of the songs in the dataset are described with high energy.

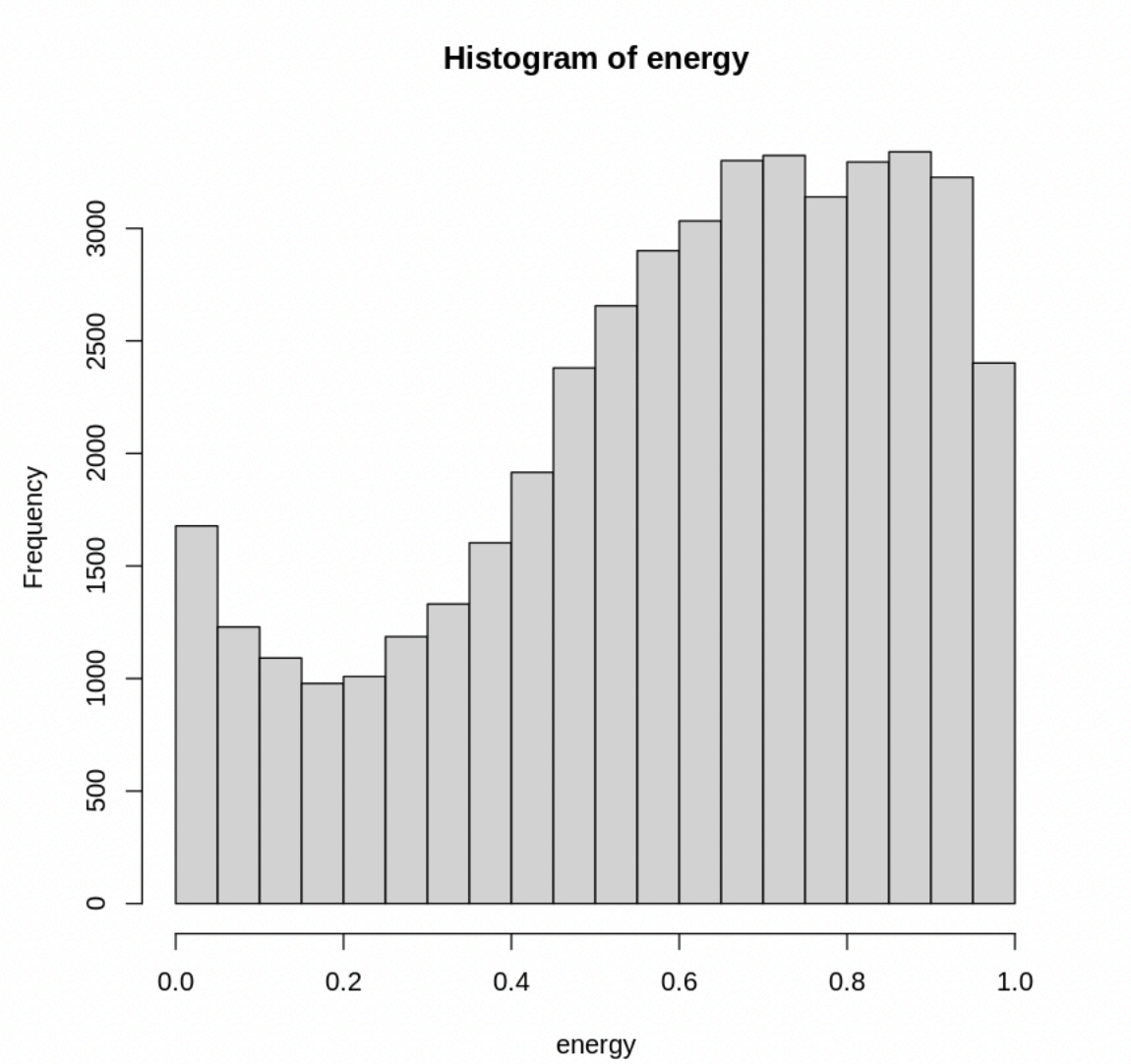

Figure 32: Histogram of energy feature

Instrumentalness is measured on a scale of 0.0 (likely contains vocal content) to 1.0 (likely contains no vocal content. Distribution of data in terms of instrumentalness is skewed here because almost all the songs have zero instrumentalness.

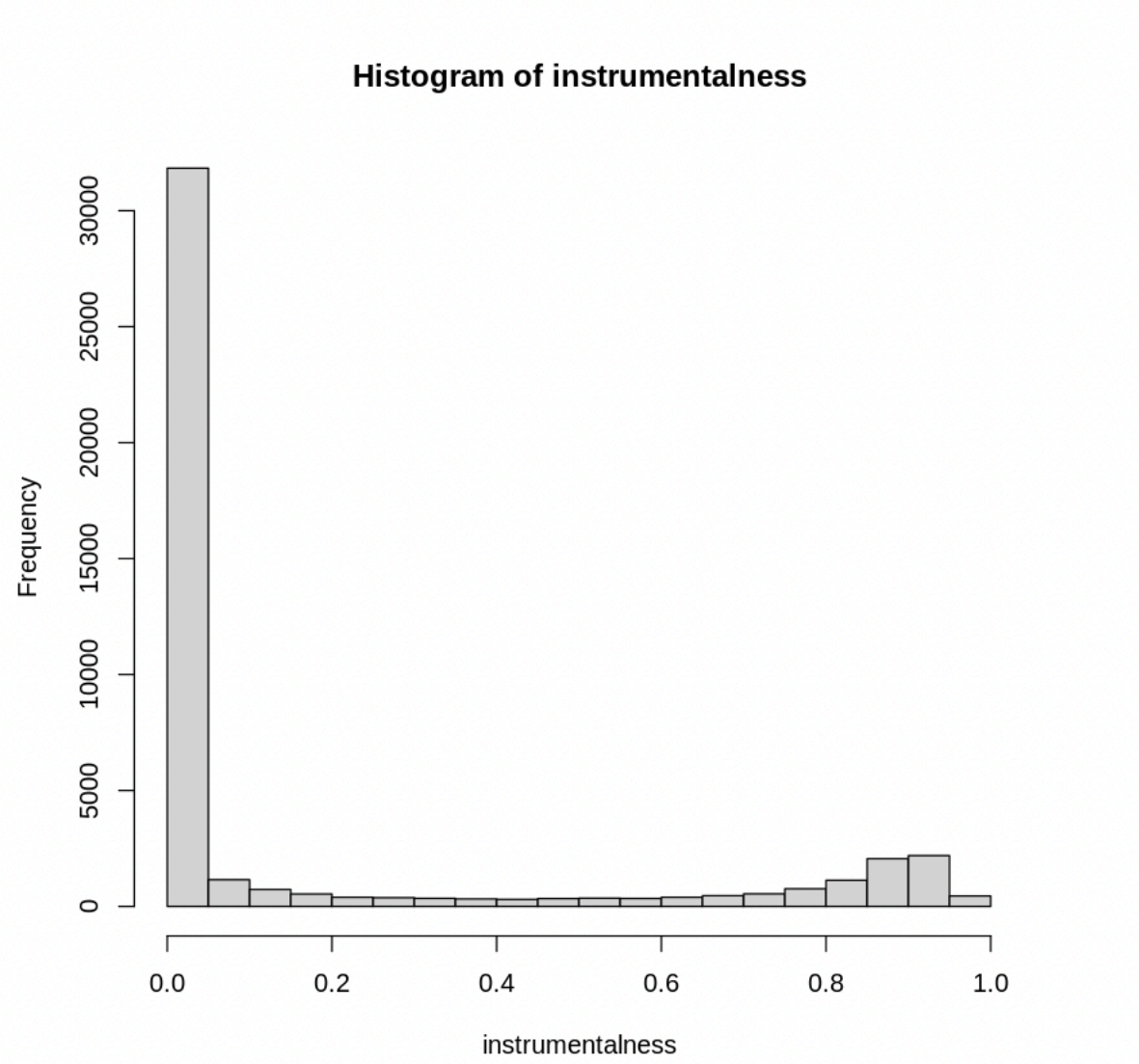

Figure 33: Histogram of instrumentalness feature

Liveness is measured on a scale of 0.0 (no audience) to 1.0 (audible audience). Songs with higher liveness are more likely to have been performed live. Most of the songs in our dataset are found to be sung without an audience.

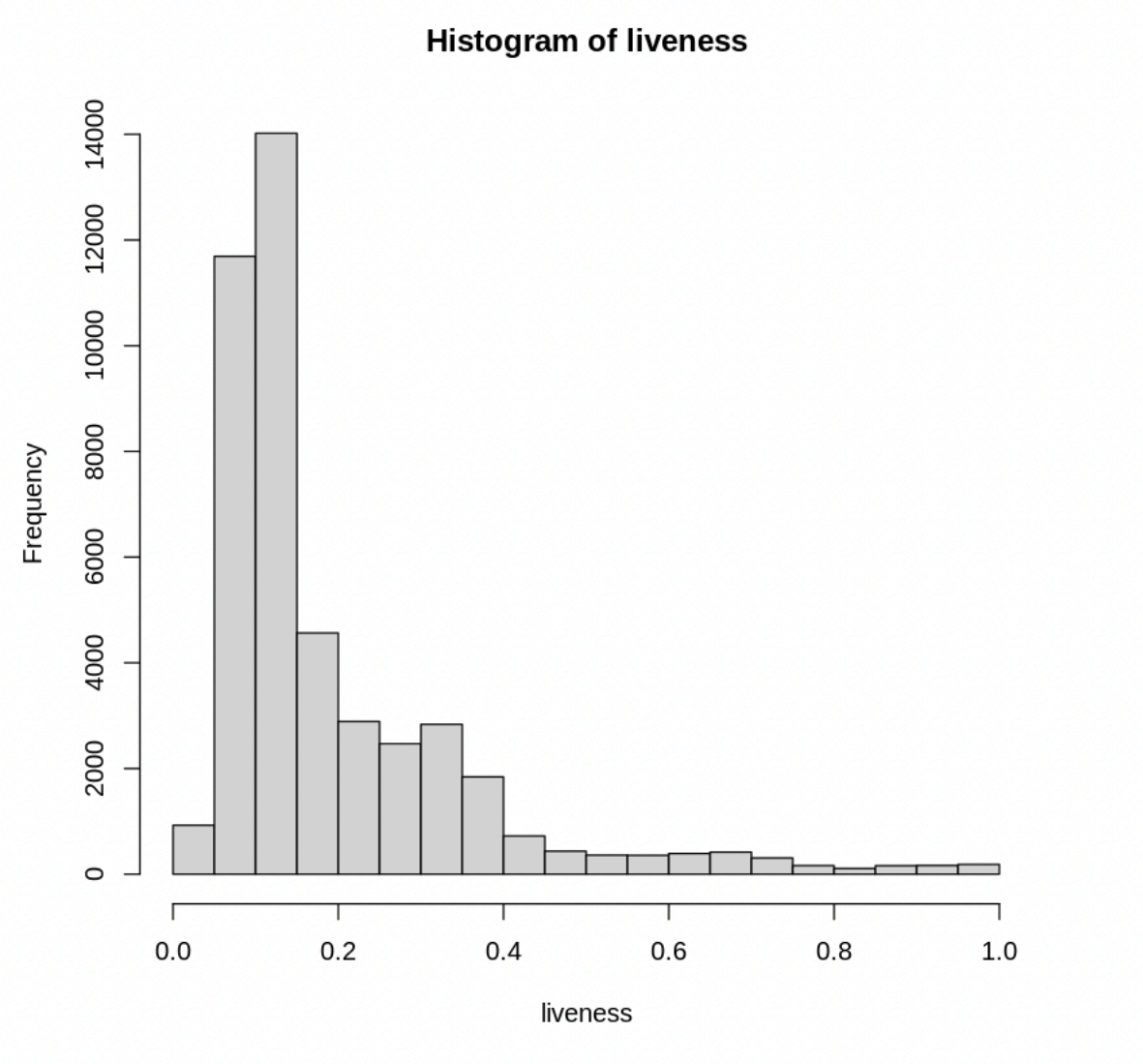

Figure 34: Histogram of liveness feature

Songs with lower loudness values are quieter relative to the reference value of 0. In terms of loudness, the plot shows that most songs are quieter (their loudness is closer to 0).

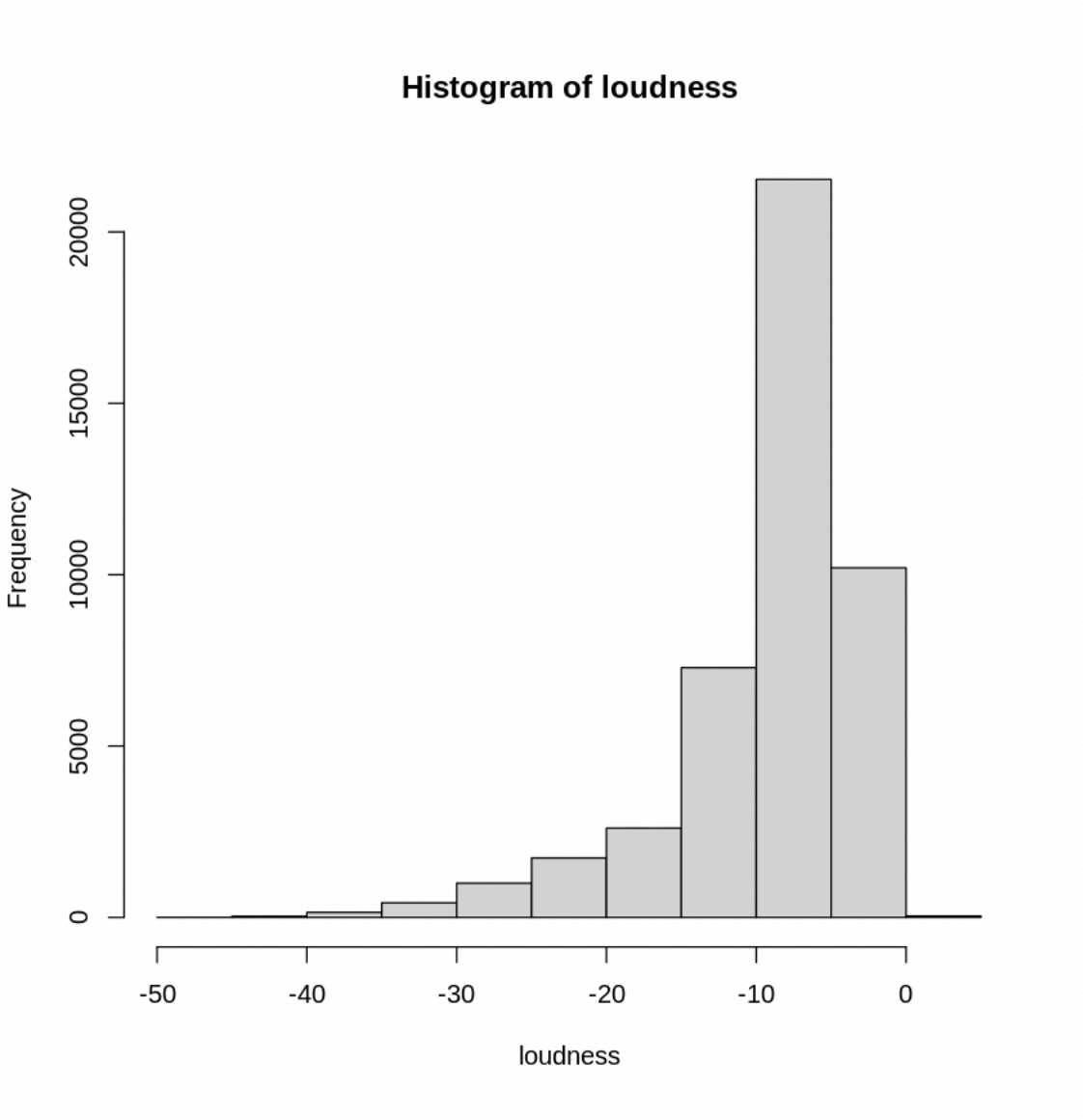

Figure 35: Histogram of loudness feature

Songs with higher speechiness are mostly composed of spoken words. Distribution of speechiness is also skewed in this dataset. Almost most of songs here have no or very small number of spoken words.

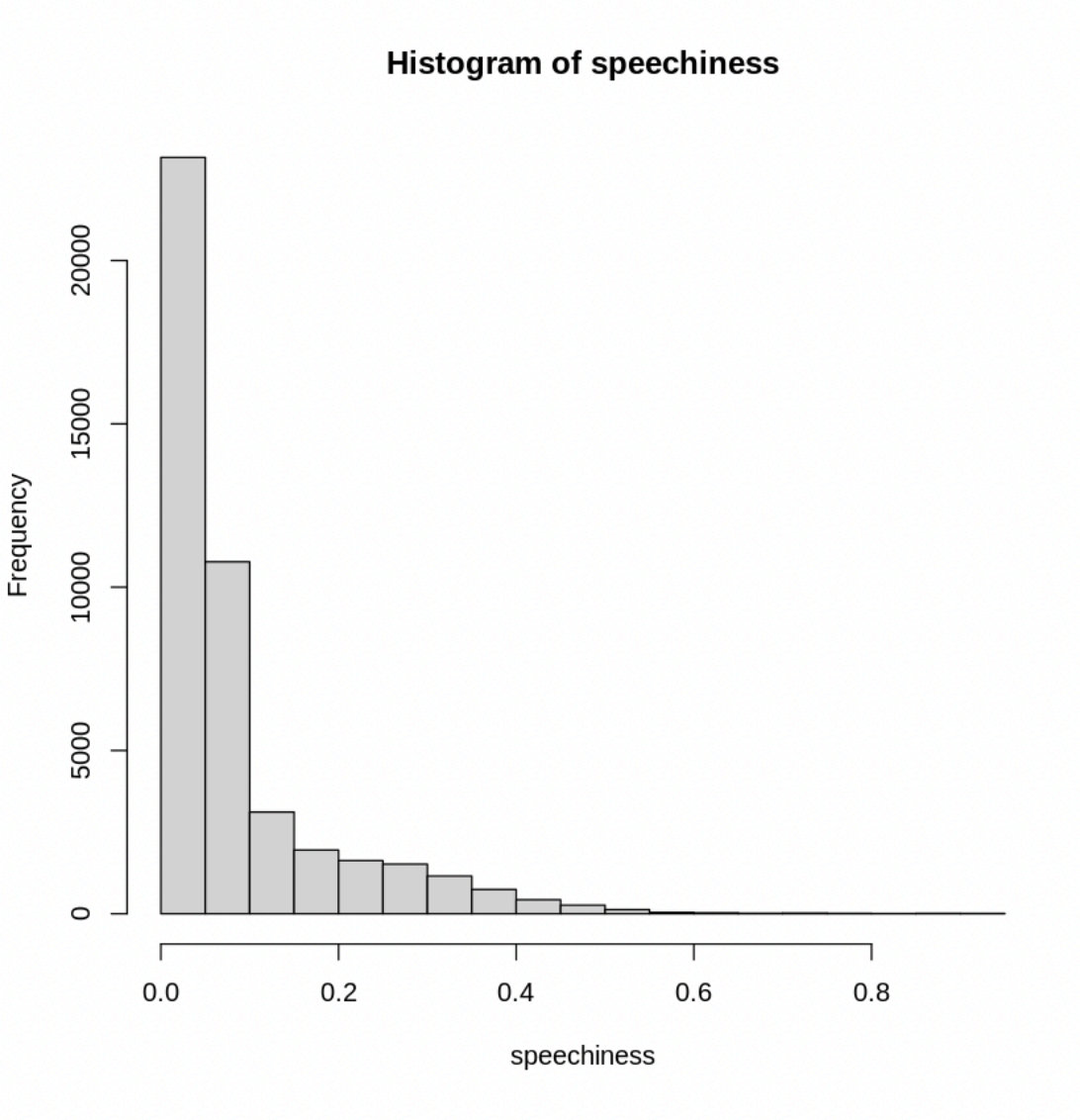

Figure 36: Histogram of speechiness feature

Valence is measured on a scale from 0.0 (low valence) to 1.0 (high valence). Distribution of valence in this dataset is found to be not in normal distribution. Only a small number of songs have high valence. A large part have s similar valence level.

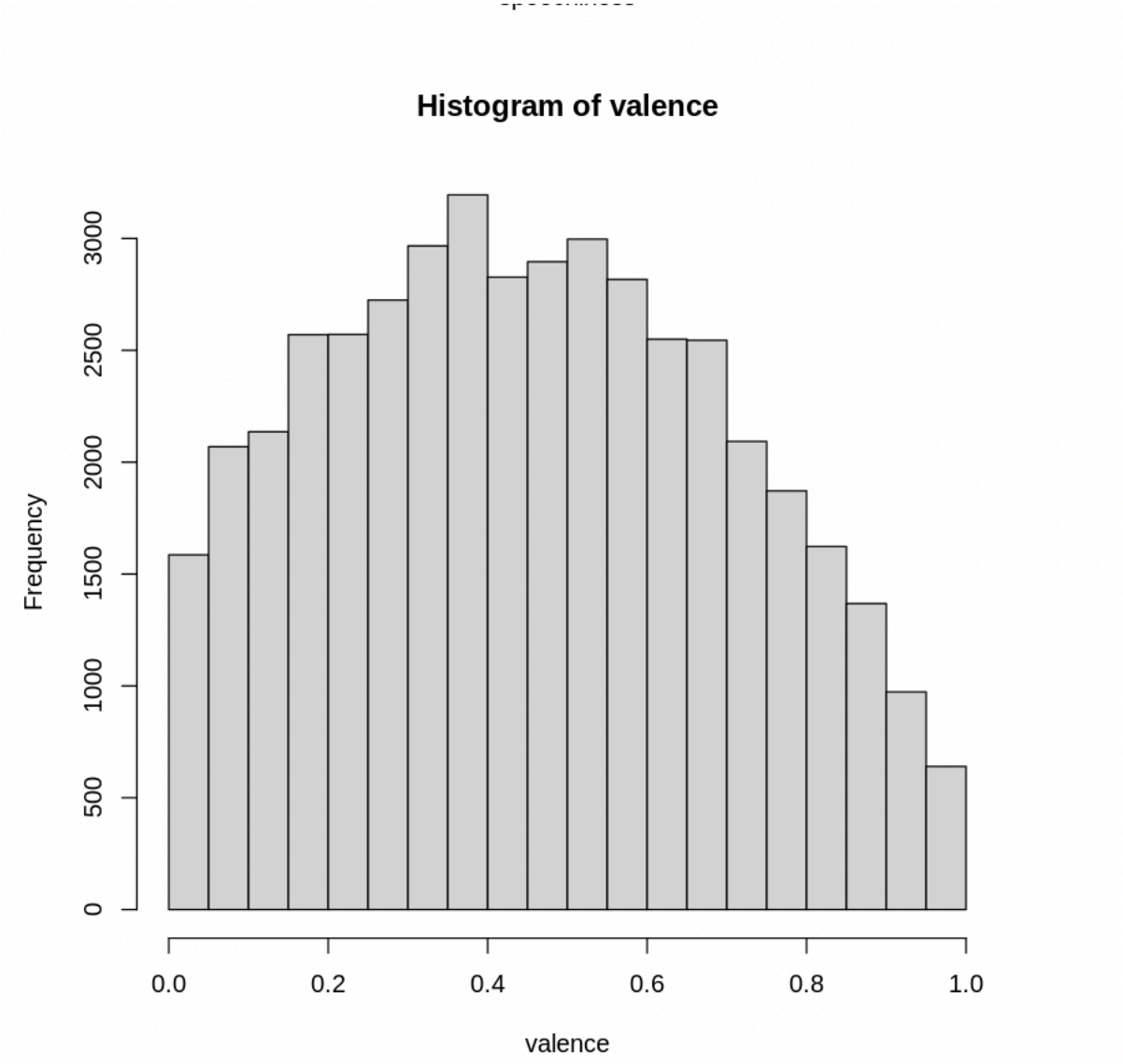

Figure 37: Histogram of valence feature

Checking for duplicate rows in the Dataset:

Duplicate observations arise when two or more rows contain the same or nearly identical values. Duplicates can represent an inaccuracy in the data, influencing subsequent studies of the data. As a result, it is best practice to check a data collection for duplicate and near duplicate observations.

The duplicate command is used to identify duplicate rows in the dataset. The output shows the dataset contains no duplicate rows.

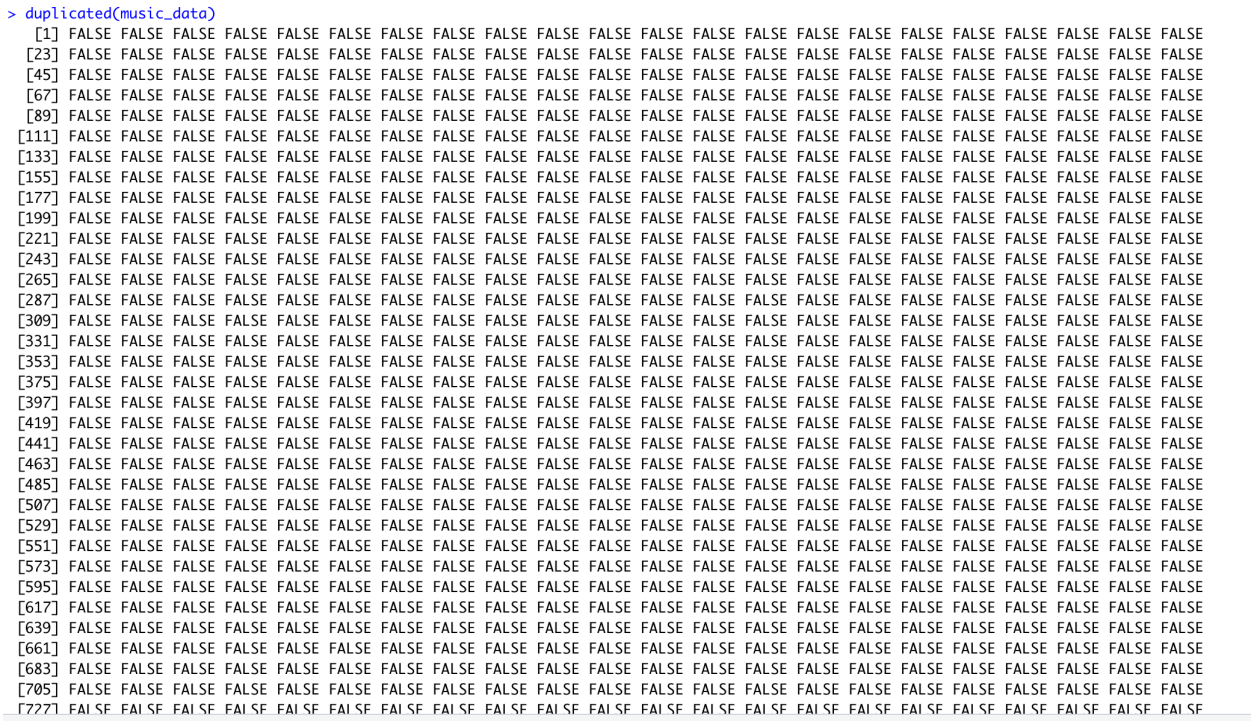

Figure 38: Output of duplicated command

One hot encoding:

One hot encoding is used to convert categorical variables into numeric data for applying to machine learning models. In this dataset, we have three categorical features – key, mode and tempo. From the data visualization, it can be seen that tempo doesn't hold much significance. Hence it can be eliminated completely.

```
[65] names(music_data)
```

'popularity' · 'acousticness' · 'danceability' · 'duration_ms' · 'energy' · 'instrumentalness' · 'key' · 'liveness' · 'loudness' · 'mode' · 'speechiness' · 'tempo' · 'valence' · 'music_genre'

```
[66] music_data=select(music_data,-c(12))
```

```
[67] names(music_data)
```

'popularity' · 'acousticness' · 'danceability' · 'duration_ms' · 'energy' · 'instrumentalness' · 'key' · 'liveness' · 'loudness' · 'mode' · 'speechiness' · 'valence' · 'music_genre'

```
[52] music_data$key <- as.factor(music_data$key)
     music_data <- one_hot(as.data.table(music_data))
```

Figure 39: Removing less significant features

The key feature is converted to numeric feature. A total of 12 new features are now added to the dataset.

```
[68] music_data$key <- as.factor(music_data$key)
     music_data <- one_hot(as.data.table(music_data))
```

```
[69] dim(music_data)
```

45020 · 24

```
[70] names(music_data)
```

'popularity' · 'acousticness' · 'danceability' · 'duration_ms' · 'energy' · 'instrumentalness' · 'key_A' · 'key_A#' · 'key_B' · 'key_C' · 'key_C#' · 'key_D' · 'key_D#' · 'key_E' · 'key_F' · 'key_F#' · 'key_G' · 'key_G#' · 'liveness' · 'loudness' · 'mode' · 'speechiness' · 'valence' · 'music_genre'

Figure 40: One hot encoding of key feature

The mode feature is converted to numeric feature. A total of 2 new features mode-major and mode_minor are now added to the dataset.

```
[71] music_data$mode <- as.factor(music_data$mode)
     music_data <- one_hot(as.data.table(music_data))
```

```
[73] dim(music_data)
```

```
45020 · 25
```

```
[72] names(music_data)
```

```
'popularity' · 'acousticness' · 'danceability' · 'duration_ms' · 'energy' · 'instrumentalness' · 'key_A' · 'key_A#' · 'key_B' · 'key_C' · 'key_C#' · 'key_D' · 'key_D#' · 'key_E' ·
'key_F' · 'key_F#' · 'key_G' · 'key_G#' · 'liveness' · 'loudness' · 'mode_Major' · 'mode_Minor' · 'speechiness' · 'valence' · 'music_genre'
```

Figure 41: One hot encoding of mode feature

Co-relation matrix:

A corelation matrix was plotted using ggplot function for numeric features alone. From the matrix, it could be seen that loudness and energy have a co relation of 1.

```
library(ggcorrplot)
cor_data=cor(numeric_cols)
ggcorrplot(cor_data,hc.order = TRUE,
           type = "lower",
           outline.color = "white")
```

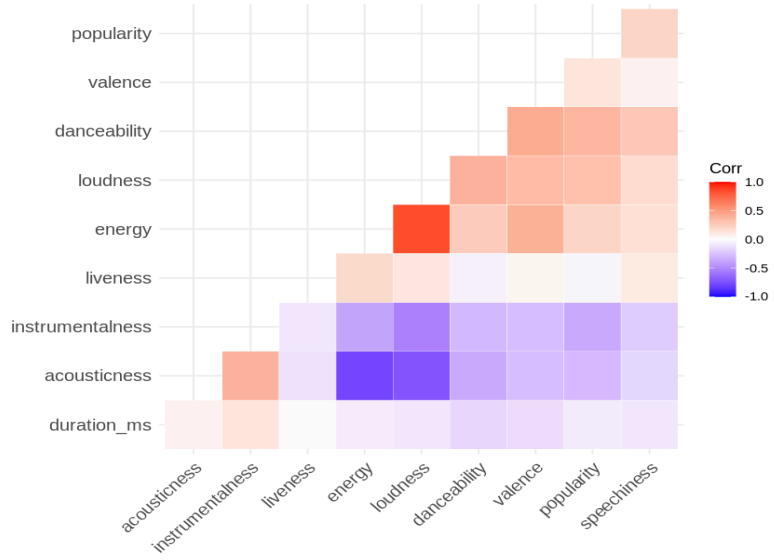

Figure 42: Co relation matrix

Conclusion:

This report performs a comprehensive Exploratory Data Analysis on the Music genre prediction dataset. This analysis will further help in identifying the important features required in the prediction of music genre. The next help involves identifying important features and building the machine learning model to predict the outcome.