MATHS 7107 Data Taming Assigmment Final Report

Possakorn Kittipipatthanapong (a1873765)

Executive Summary

Spotify, the world's largest music streaming service provider with over 365 million monthly active users, aims to enhance customer satisfaction and maintain its position as the best music streaming platform through genre prediction. This approach can improve the recommendation system and personalize compilation playlists more efficiently.
In this study, the relationship between the outcome and relevant variables, such as the released year, popularity, speechiness, tempo, and other potential features, was identified through exploratory data analysis. The best model was then selected and evaluated for accuracy. Despite the fact that the random forest model performed better than the other models, such as linear discriminant analysis and K-nearest neighbors, the results were still unsatisfactory, indicating that accurately classifying a song by its genre was challenging.

Methods

(Data cleaning, Model Selection)
Spotify, largest music streaming service provider, have provided music performance data released year between 1957 to 2020 to model the genre classification to enhance customer satisfaction and ultimately preserve the music streaming service's status as the leading provider in the industry. The interesting factors will be considered to predict the genre including:

- the year the song was released: **release_year**
- how "speechy" the song is: **speechiness**
- how danceable the song is: **danceability**
- the tempo of the song: **tempo**

Following other variables in the record, we considered only additional 5 potential factors as the predictors to build the following analysis which was completed in R version 4.2.3 using tidyverse and dplyr packages:

- perceptual measure of intensity and activity: **energy**
- the overall loudness of the track -60 to 0 decibels: **loudness**
- how acoustic the track is: **acousticness**
- how instrumental the song is: **instrumentalness**
- indicating popularity: **track_popularity**

The first step in cleaning data to make it properly following the substep below and all of manipulated code referred in appendix section:

- all track which consisted of many genres were filtered out due to duplicate songs in difference playlist following the **Table1** and **Table2**.
- the released year the songs were extracted from the release date using substr.
- all character variables change to factors.

The second step in optimizing the computing power of our model by reducing the size of songs in each genre to 1000 using slice_sample and performing the balance classification model before exploring the relationship from the Spotify's founders question and performing exploratory data analysis.

After data manipulation was completed, there are few steps before putting into the model including data splitting, preprocessing, building model specification , Tuning model, and Model Selection. These procedures are compulsory to perform the standard methodology to select the most appropriate model. The following steps were identified:

- To begin with, when splitting data into the training and testing set was to consider the ratio as 0.75:0.25, while ensuring that the genre was uniformly distributed within the sample.
- Next, in order to enhance the model's performance and training stability, preprocessing was necessary to normalize all numeric predictors in order to scale the features uniformly and potentially remove columns having large absolute correlations with others.
- In the third step, we focused on constructing the model specifications, performing V-fold cross-validation, and fine-tuning it, which involved testing various models such as linear discriminant analysis, A K-nearest neighbours model, and random forest, and comparing their performance to identify the best possible outcome. All of the results from tuning hyper parameter were located in appendix section.

- Finally, we selected the model based on the most appropriate metric, and evaluated it using the testing data to predict the genre of the song, based on the variables. We then compared the predicted genre with the actual genre of the song to assess the model's accuracy.

Results

(Exploratory Data Analysis, Model Evaluation)

After performing of data manipulation to clean and minimize the processing resources. The dataset should be further explored to answer the following questions based on Spotify founders and determined the relationship between song genre and related predictors.

First, the high level management of Spotify focused on how genre impacted following these variables:

- Based on **Figure 1**, the **trend of popularity over time** shows a weak negative linear association, and this relationship continues until the year 2016 where the opposite trend is observed. In response to the original question requested by the founder, there is no discernible pattern or clear trend in the popularity of a song over time.
- Related to the relationship between **genre** and **popularity** on **Figure2**, Shape: all seems to be symmetric. Location: EDM represent the lowest median as 33. on the other hands, RAP equal to 44 which the highest median was performed. Spread: R&B and rock interpret the highest spread related to IQR. Outlier: there is no outlier presented in this plot. Following the founder question, there is no much difference popularity in each genre, with there appears to be no strong association with slightly lower median on EDM.
- Related to the relationship between **genre** and **speechiness** on **Figure3**, Shape: RAP seems to be symmetric but others present the right-skewed. Location: RAP has the highest median as 0.166. Spread: RAP also has highest variability according to IQR. Outlier: there is highest outlier on R&B. Related to the request of founder, there is some difference represented between genre and speechiness especially on RAP Genre.

Second, we will investigate how potential factors that are related to each other and to genre are connected, in order to obtain an overall understanding of the associations. However, the detail relationships will be visualized in the chart located in the appendix section. The associations that will be illustrated include:

- From Relationship between **genre** and **released year** on **Figure4**, ROCK and POP present the systematic shape. Conversely, others has left-skewed shape. Location: All genre except ROCK, which has a median of released year around 2000, have the newer median year close to 2020. Spread: There is highest spread on ROCK, while R&B and RAP perform the moderate spread. However, narrow distribution presented on EDM, Latin and POP genre. Outlier: there are outliers across dataset except ROCK. Finally, There are relationships between genre and released with extremely cases on ROCK.
- From Relationship between **genre** and **danceability** on **Figure5**, location: ROCK show the less danceability compared to the median from other genre. While All genre seems to be similar systematic shape, spread. And, outlier show in all genre. The association of genre and danceability is shown especially on ROCK.
- From Relationship between **genre** and **tempo** on **Figure6**, all genre perform the systematic shape. While EDM present the extremely small variability compared to others. Location: all median beats per minute located roughly between 105 and 130. Outlier spots present on every genre. Finally, there looks to be a some association between genre and tempo.

- From Relationship between **genre** and **energy** on **Figure7**, all genre seems to be systematic and similar spread pattern. For location, EDM genre, with the highest energy, has high median value around 0.838. However, The lowest energetic and intensity is on R&B type. Outliers still perform on all genre. Following the overview plot, there looks to be a strong association between genre and energy.
- From Relationship between **genre** and **loudness** on **Figure8**, all seems to systematic shape, similar variability with some outliers. There appears to be no strong association related to the median range between 0 and -10 decibels.
- From Relationship between **genre** and **acousticness** on **Figure9**, right-skewed pattern show across all genre. The highest median acousticness represent in R&B with highest spread. On the other hands, EDM and rock present the median value close to zero. Ouliters could be noticed along data points. Finally, There is association between genre and acousticness.
- From Relationship between **genre** and **instrumentalness** on **Figure10**, EDM has the highest instrumentalness which fewer vocals present on track. Moreover, EDM also introduce the extremely highest variability. there are outliers across dataset. Related to this detail, there are some associations with extremely case related on EDM genre.

Most of interesting and potential factors represent some associations to genre except **popularity** and **loudness** with poor relationships following the shown plot.

After a thorough analysis following the tuning models including:

- a linear discriminant analysis
- a K-nearest neighbours model with a range of 1 to 100 and 20 levels, and
- a random forest with 100 trees and 5 levels.

The results evaluated based on **accuracy** which metric represent the most proper benchmark based on **balanced multiclass classification**:

- Referred to the results from **Table3**, Estimating the performance of model to predict which genre a song belongs to using indicator as accuracy. The highest accuracy present which found in random forest model equal to 0.547, with mtry and min_n equal to 3 and 30 respectively. On the other hands, K-nearest neighbors with 63 neighbors has 0.524. Finally, the results from linear discriminant analysis, with lowest accuracy, is 0.460.
- Related to results above, we could interpret that following example. For every 100 possibilities to predict the genre which song belong, there are 54.7 times to get the correct answer using random forest, 52.4 times using K-nearest neighbors, and 46 times using linear discriminant analysis. Therefore, **Random forest** was presented the best performance compared to other models

Discussion

(Model Evaluation, Model Prediction)
After evaluating the performance of the three models proposed, we determined that Random Forest was the most appropriate model for our case. The most importance related to the outcome following **Figure11** is **released year** which determined in the interesting factors. On the other hands, **popularity** and **loudness** represent the least importance to classify the genre.
Refer to the outcome from random forest with testing dataset, we found that our model did not perform as accurately as desired. The accuracy, which is the proportion of corrected predicted by our model, was **0.563** shown in **Table4**, indicating a low probability of correctly classifying a song by its genre. Moreover, if we focus on each genre performance following **Table5**, ROCK will represent the possibility to classify correctly as 0.752. On the Other hands, R&B, LATIN, and POP has a very low likelihood of being classified correctly as 0.480, 0.440, and 0.404 respectively.

Conclusion

With more than 365 million monthly active users, Spotify is the leading music streaming service provider, persistently pursuing customer satisfaction to remain the top music streaming platform. One of the ways they are doing this is through genre prediction, which can improve the recommendation system and personalize compilation playlists more efficiently.
Through exploratory data analysis, we identified the relationship between the outcome and related variables after performing data manipulation and preprocessing with normalization and related procedure. The best model which tunes the specification was then selected and evaluated with the testing dataset to measure its accuracy.
Although the random forest model performed better than the other two models, the results were still poor, indicating a difficulty in accurately classifying a song by its genre. In light of the poor results obtained in this study which has the limitation on single genre track, future work in this area should focus on utilizing better computing resources and considering a higher number of data points to achieve better accuracy. Additionally, applying the model with imbalanced classification could provide actual use cases and contribute to a more comprehensive understanding of the problem.

Appendix

Methods: Import library

```
pacman::p_load(
  tidyverse,skimr, # data manipulation
  tidymodels,themis,discrim,vip, # model selection
  tinytex, knitr # formatting
)
```

Methods: Import Dataset

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

```
spotify_songs.DuplicatedGenre <- spotify_songs %>%
  group_by(track_id) %>%
  summarise(n = n(),
            list = paste(playlist_genre, collapse = ",")
            ) %>%
  arrange(desc(n)) %>%
  filter(n>1)

# display the spotify example preprocessed data
kable(head(spotify_songs.DuplicatedGenre,10), caption = "First 10 example rows for duplicated g
```

Table 1: First 10 example rows for duplicated genre tracks

track_id	n	list
7BKLCZ1jbUBVqRi2FVlTVw	10	pop,pop,pop,pop,rap,latin,latin,latin,r&b,edm
14sOS5L36385FJ3OL8hew4	9	pop,pop,pop,pop,latin,latin,latin,r&b,edm
3eekarcy7kvN4yt5ZFzltW	9	pop,rap,rap,rap,latin,latin,r&b,r&b,edm
0nbXyq5TXYPCO7pr3N8S4I	8	rap,rap,rap,latin,latin,r&b,r&b,edm
0qaWEvPkts34WF68r8Dzx9	8	pop,latin,latin,r&b,r&b,edm,edm,edm
0rIAC4PXANcKmitJfoqmVm	8	pop,pop,latin,latin,r&b,r&b,r&b,edm
0sf12qNH5qcw8qpgymFOqD	8	pop,pop,pop,latin,latin,r&b,r&b,edm
2Fxmhks0bxGSBdJ92vM42m	8	pop,pop,pop,rock,latin,r&b,r&b,edm
2b8fOow8UzyDFAE27YhOZM	8	pop,pop,latin,latin,latin,r&b,r&b,edm
2tnVG71enUj33Ic2nFN6kZ	8	pop,pop,latin,latin,r&b,r&b,edm,edm

```
spotify_songs.DuplicatedGenre_example <- spotify_songs %>%
  filter(track_id == "7BKLCZ1jbUBVqRi2FVlTVw") %>%
  dplyr::select(
    track_name,
    track_artist,
    playlist_name,
    playlist_genre
  )
```

```
# display the spotify example preprocessed data
kable(spotify_songs.DuplicatedGenre_example, caption = "Example duplicated data on different pl
```

Table 2: Example duplicated data on different playlists, track id: 7BKLCZ1jbUBVqRi2FVlTVw

track_name	track_artist	playlist_name	playlist_genre
Closer (feat. Halsey)	The Chainsmokers	Dance Pop	pop
Closer (feat. Halsey)	The Chainsmokers	Post pop teen	pop
Closer (feat. Halsey)	The Chainsmokers	Electropop Hits 2017-2020	pop
Closer (feat. Halsey)	The Chainsmokers	A Loose Definition of Indie Poptimism	pop
Closer (feat. Halsey)	The Chainsmokers	Hip Hop Dance Music – Urban – Trap – Breaking Locking Popping Bopping – WOD – World of Dance	rap
Closer (feat. Halsey)	The Chainsmokers	Tropical House Run 190 BPM	latin
Closer (feat. Halsey)	The Chainsmokers	2020 Hits & 2019 Hits – Top Global Tracks	latin
Closer (feat. Halsey)	The Chainsmokers	2020 Hits & 2019 Hits – Top Global Tracks	latin
Closer (feat. Halsey)	The Chainsmokers	2020 Hits & 2019 Hits – Top Global Tracks	r&b
Closer (feat. Halsey)	The Chainsmokers	2015 songs	edm

Methods: Data Cleaning

```
set.seed(1873765)
spotify_songs.clean <- spotify_songs %>%
  filter(!track_id %in% spotify_songs.DuplicatedGenre$track_id ) %>%
  mutate(release_year = as.integer(substr(track_album_release_date,1,4))) %>%
  dplyr::select(playlist_genre,
                release_year, speechiness, danceability, tempo,
                energy, loudness, acousticness, instrumentalness, track_popularity
  ) %>%
  mutate_if(is.character, as.factor) %>%
  group_by(playlist_genre) %>%
  slice_sample(n=1000) %>%
  ungroup()
spotify_songs.clean
## # A tibble: 6,000 x 10
##    playlist_genre release_year speechiness danceability tempo energy loudness
##    <fct>                 <int>       <dbl>        <dbl> <dbl>  <dbl>    <dbl>
##  1 edm                    2018      0.0651        0.679  124.  0.995    -4.34
##  2 edm                    2018      0.04          0.814  124.  0.398    -8.92
##  3 edm                    2015      0.0351        0.49   126.  0.72     -3.80
##  4 edm                    2019      0.0403        0.619  124.  0.71     -3.06
##  5 edm                    2016      0.0617        0.373  130.  0.902    -3.47
##  6 edm                    2018      0.0517        0.773  126.  0.578   -11.8
##  7 edm                    2016      0.0315        0.545  144.  0.65     -5.85
##  8 edm                    2013      0.0298        0.552  130.  0.831    -5.22
##  9 edm                    2019      0.194         0.739  124.  0.882    -7.45
## 10 edm                    2013      0.0428        0.661  130.  0.721    -5.70
## # i 5,990 more rows
## # i 3 more variables: acousticness <dbl>, instrumentalness <dbl>,
## #   track_popularity <dbl>
```

Results: Exploratory Data Analysis

relationship between released year and popularity of the song

```
spotify_songs.clean %>%
  group_by(release_year) %>%
  summarise(track_popularity = mean(track_popularity)) %>%
  ggplot(aes(x = release_year, y = track_popularity)) +
  geom_line() +
  geom_point() +
  geom_smooth(method = lm,
              color = "red",
              se = FALSE
              ) +
  labs(x = "Released Year", y = "Popularity")
```

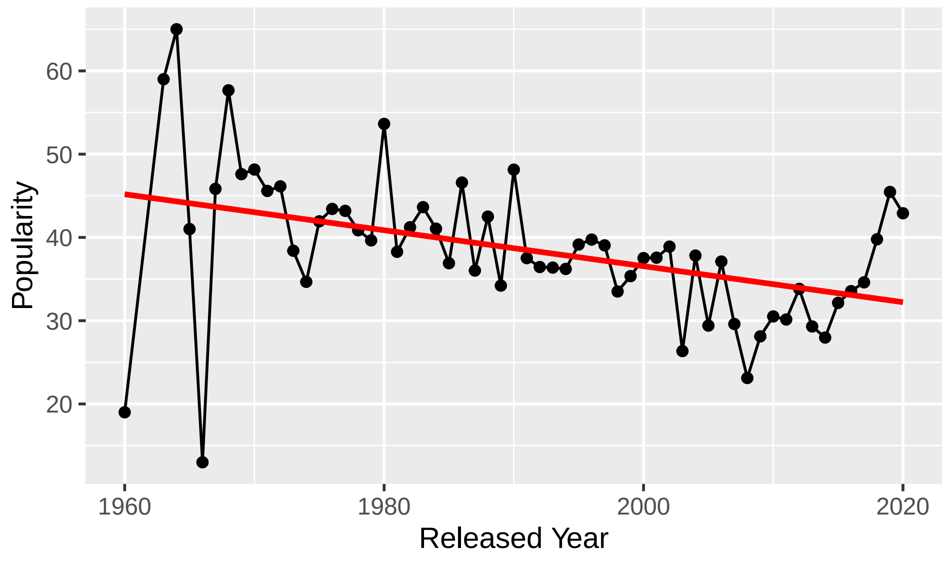

Figure 1: Line chart represent the popularity of the songs change over time

Relationship between genre and popularity of the song

```r
spotify_songs.clean %>%
  ggplot(aes(x=playlist_genre, y=track_popularity, fill=playlist_genre))+
  geom_boxplot()+
  labs(x = "Music Genre", y = "Popularity", fill = "Genre")
```

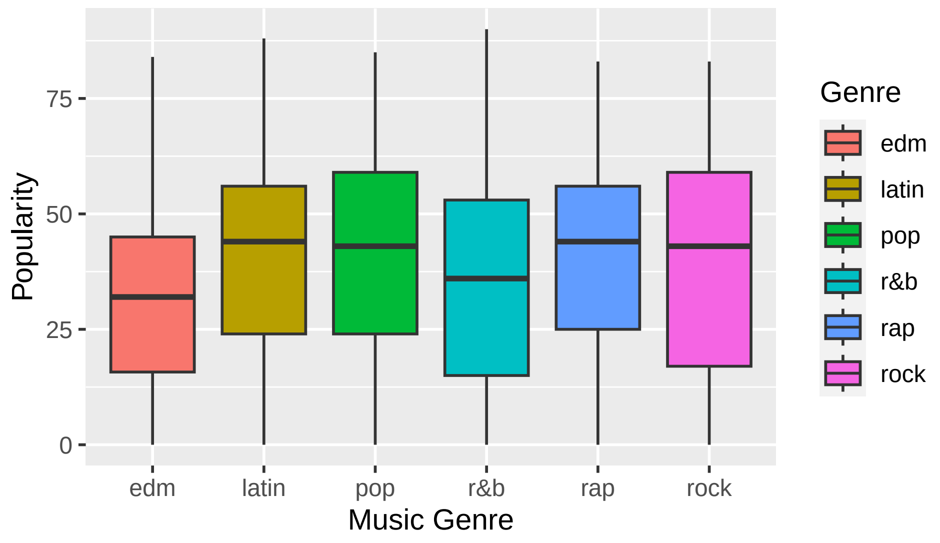

Figure 2: Box plot represent the relationship between genre and popularity of the songs

relationship between genre and speechiness of the song

```
spotify_songs.clean %>%
  ggplot(aes(x=playlist_genre, y=speechiness, fill=playlist_genre))+
  geom_boxplot()+
  labs(x = "Music Genre", y = "Speechiness", fill = "Genre")
```

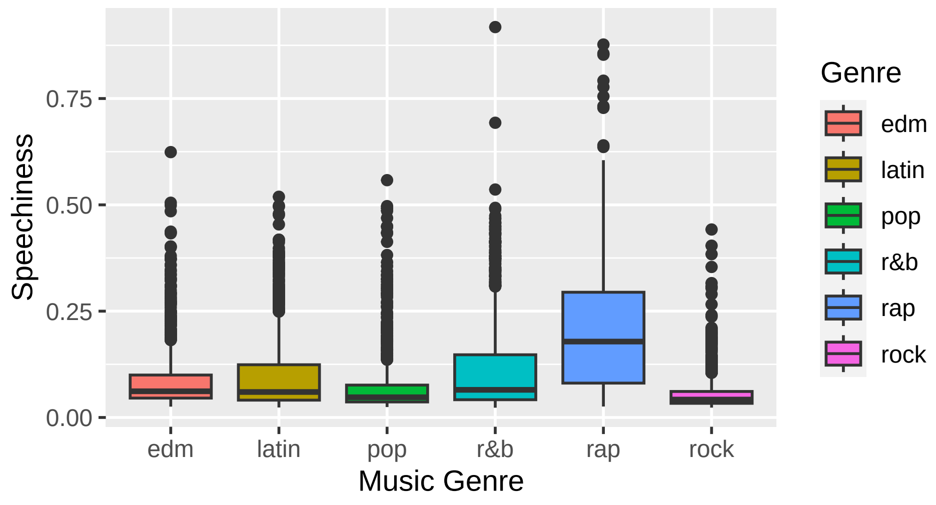

Figure 3: Box plot represent the relationship between genre and speechiness of the songs

relationship between genre and released year of the song

```
spotify_songs.clean %>%
  ggplot(aes(x=playlist_genre, y=release_year, fill=playlist_genre))+
  geom_boxplot()+
  labs(x = "Music Genre", y = "Released year", fill = "Genre")
```

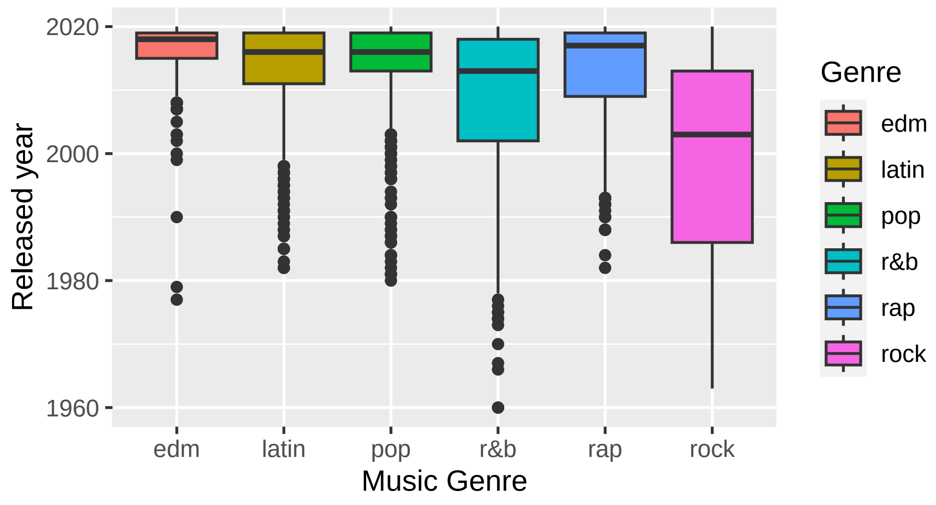

Figure 4: Box plot represent the relationship between genre and released year of the songs

relationship between genre and danceability of the song

```
spotify_songs.clean %>%
  ggplot(aes(x=playlist_genre, y=danceability, fill=playlist_genre))+
  geom_boxplot()+
  labs(x = "Music Genre", y = "Danceability", fill = "Genre")
```

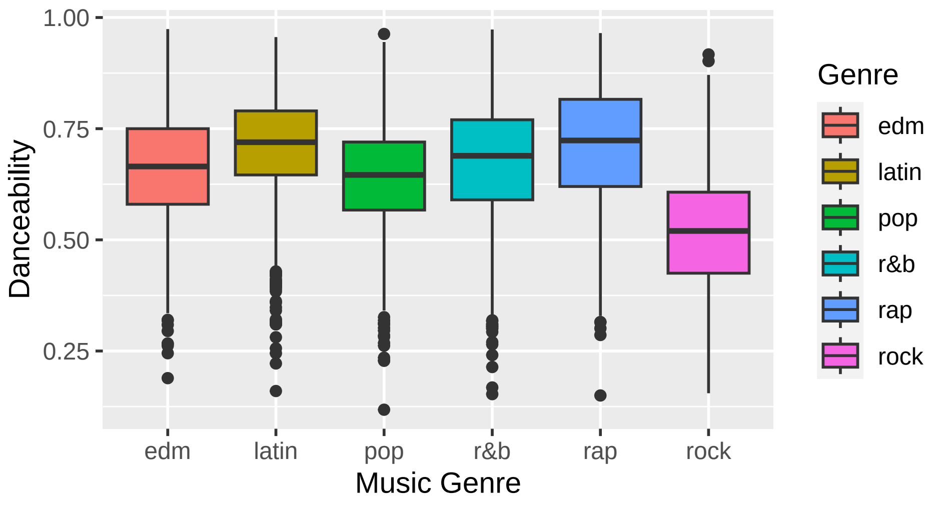

Figure 5: Box plot represent the relationship between genre and danceability of the songs

relationship between genre and tempo of the song

```
spotify_songs.clean %>%
  ggplot(aes(x=playlist_genre, y=tempo, fill=playlist_genre))+
  geom_boxplot()+
  labs(x = "Music Genre", y = "Tempo", fill = "Genre")
```

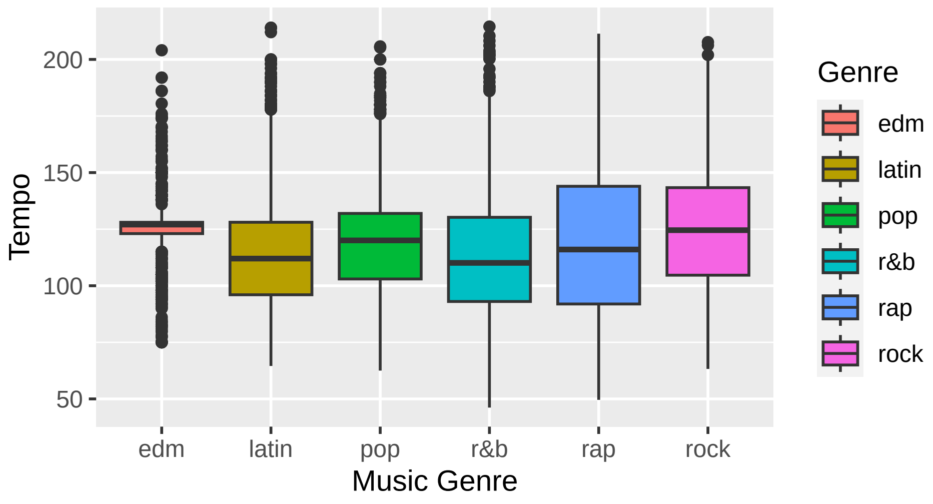

Figure 6: Box plot represent the relationship between genre and tempo of the songs

relationship between genre and energy of the song

```
spotify_songs.clean %>%
  ggplot(aes(x=playlist_genre, y=energy, fill=playlist_genre))+
  geom_boxplot()+
  labs(x = "Music Genre", y = "Energy", fill = "Genre")
```

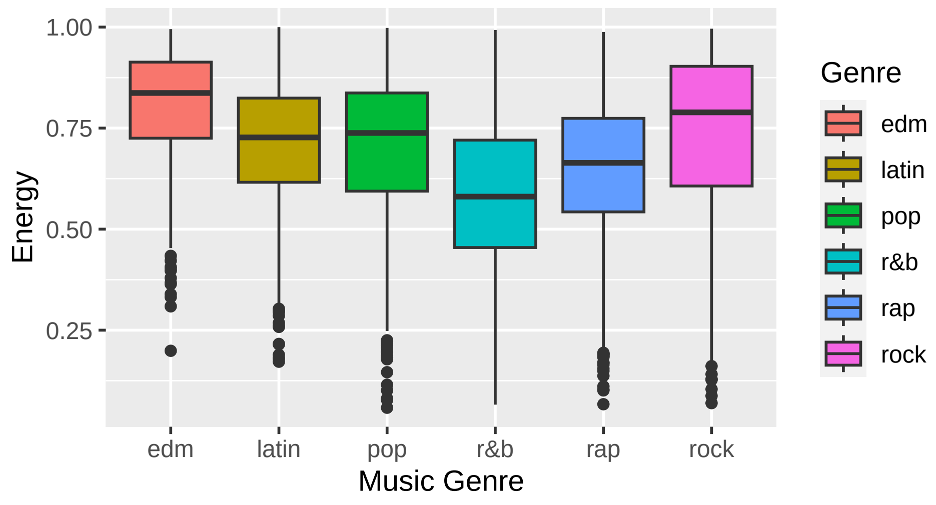

Figure 7: Box plot represent the relationship between genre and energy of the songs

relationship between genre and loudness of the song

```
spotify_songs.clean %>%
  ggplot(aes(x=playlist_genre, y=loudness, fill=playlist_genre))+
  geom_boxplot()+
  labs(x = "Music Genre", y = "Loudness", fill = "Genre")
```

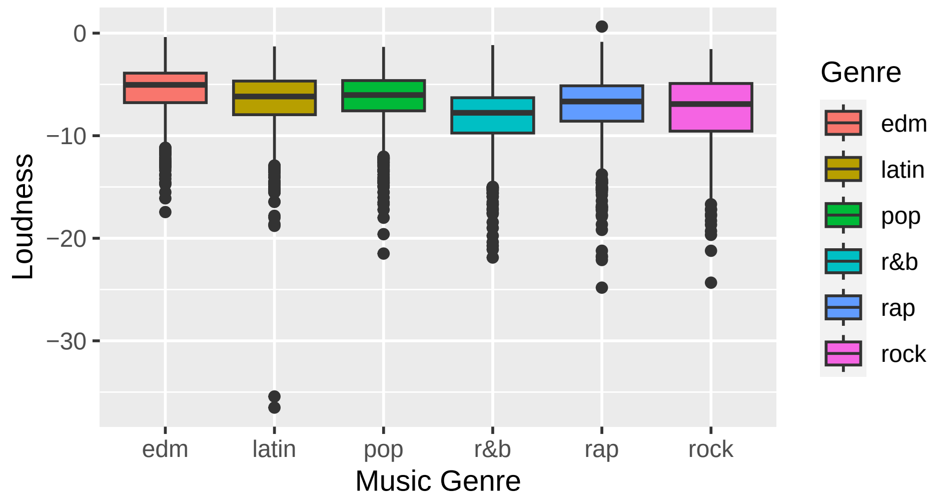

Figure 8: Box plot represent the relationship between genre and loudness of the songs

relationship between genre and acousticness of the song

```
spotify_songs.clean %>%
  ggplot(aes(x=playlist_genre, y=acousticness, fill=playlist_genre))+
  geom_boxplot()+
  labs(x = "Music Genre", y = "Acousticness", fill = "Genre")
```

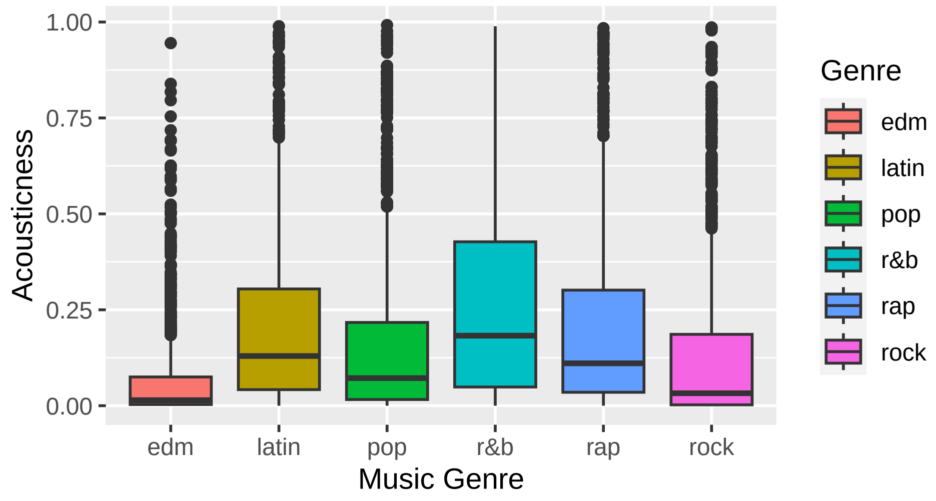

Figure 9: Box plot represent the relationship between genre and acousticness of the songs

relationship between genre and instrumentalness of the song

```
spotify_songs.clean %>%
  ggplot(aes(x=playlist_genre, y=instrumentalness, fill=playlist_genre))+
  geom_boxplot()+
  labs(x = "Music Genre", y = "Instrumentalness", fill = "Genre")
```

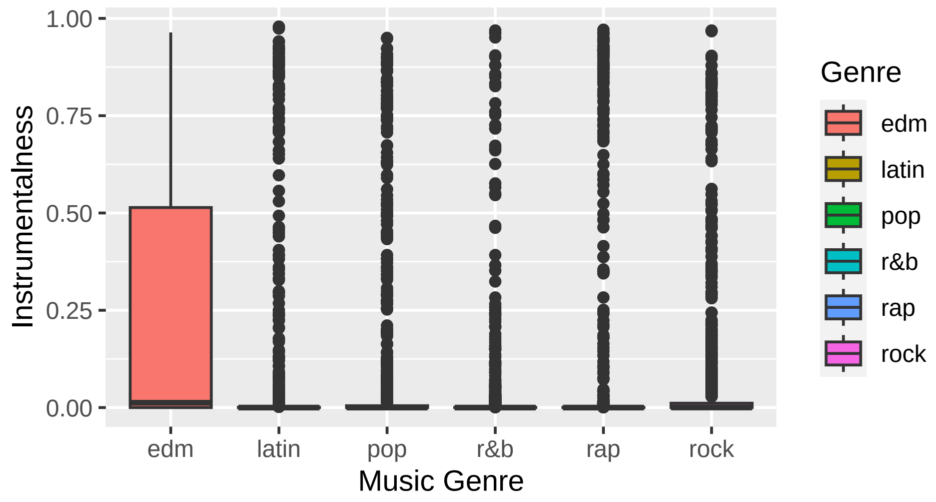

Figure 10: Box plot represent the relationship between genre and instrumentalness of the songs

Methods: Data Preprocessing

01 Data splitting

```r
set.seed(1873765)
spotify_split <- initial_split(spotify_songs.clean, strata = "playlist_genre")
spotify_songs.train <- training(spotify_split)
spotify_songs.test <- testing(spotify_split)
spotify_split
## <Training/Testing/Total>
## <4500/1500/6000>
```

02 Preprocesing

```r
spotify_recipe <- recipes::recipe(playlist_genre ~ ., data = spotify_songs.train) %>%
  step_normalize( all_predictors() ) %>%
  step_corr( all_predictors() ) %>%
  prep()
spotify_recipe
# apply preprocess on training dataset
spotify_train_preprocess <- juice(spotify_recipe)
# apply preprocess on testing datase
spotify_test_preprocess <- bake(spotify_recipe, spotify_songs.test)
```

03 Build model specification

```r
# model specification - IDA
spotify_lda_spec <- discrim_linear(
  mode = "classification"
) %>%
  set_engine("MASS")
spotify_lda_spec
## Linear Discriminant Model Specification (classification)
##
## Computational engine: MASS

# model specification - KNN
spotify_knn_spec <- nearest_neighbor(
  mode = "classification",
  neighbors = tune()
) %>%
  set_engine("kknn")
spotify_knn_spec
## K-Nearest Neighbor Model Specification (classification)
##
## Main Arguments:
##   neighbors = tune()
```

```
##
## Computational engine: kknn

# model specification - random forest
spotify_rf_spec <- rand_forest(
  mode = "classification",
  mtry = tune(),
  trees = 100,
  min_n = tune()
) %>%
  set_engine( "ranger", importance = "permutation"  )
spotify_rf_spec
## Random Forest Model Specification (classification)
##
## Main Arguments:
##   mtry = tune()
##   trees = 100
##   min_n = tune()
##
## Engine-Specific Arguments:
##   importance = permutation
##
## Computational engine: ranger
```

04 Tuning model

```
set.seed(1873765)
spotify_cv = vfold_cv(spotify_train_preprocess, v = 5)
spotify_cv
## #  5-fold cross-validation
## # A tibble: 5 x 2
##   splits             id
##   <list>             <chr>
## 1 <split [3600/900]> Fold1
## 2 <split [3600/900]> Fold2
## 3 <split [3600/900]> Fold3
## 4 <split [3600/900]> Fold4
## 5 <split [3600/900]> Fold5
```

cross validation set to training data

```
# create grid for tuning
spotify_knn_grid <- grid_regular(parameters(neighbors(range = c(1,100))),
                                 levels = 20)
set.seed(1873765)
spotify_knn_tune <- tune_grid(object = spotify_knn_spec,
                              preprocessor = recipe(playlist_genre ~ ., data = spotify_train_pr
                              resamples = spotify_cv,
                              grid = spotify_knn_grid
                              )

best_accuracy_knn <- select_best( spotify_knn_tune, "accuracy")
spotify_knn_spec_final <- finalize_model( spotify_knn_spec, best_accuracy_knn )
spotify_knn_spec_final
## K-Nearest Neighbor Model Specification (classification)
##
## Main Arguments:
##   neighbors = 63
##
## Computational engine: kknn
```

Tuning KNN model

```

spotify_rf_grid <- grid_regular(
  finalize(
    mtry(),
    spotify_train_preprocess %>%
      dplyr::select( -playlist_genre )
    ),
  min_n(),
  levels = 5
)
set.seed(1873765)
spotify_rf_tune <- tune_grid(object = spotify_rf_spec,
                             preprocessor = recipe(playlist_genre ~ ., data = spotify_train_pre
                             resamples = spotify_cv,
                             grid = spotify_rf_grid
                             )

best_accuracy_rf <- select_best( spotify_rf_tune, "accuracy")
spotify_rf_spec_final <- finalize_model( spotify_rf_spec, best_accuracy_rf )
spotify_rf_spec_final
## Random Forest Model Specification (classification)
##
## Main Arguments:
##   mtry = 3
##   trees = 100
##   min_n = 30
##
## Engine-Specific Arguments:
##   importance = permutation
##
```

```
## Computational engine: ranger
```

Tuning random Forest

Results: 05 Model Selection

```
# linear discriminant analysis
lda_cv <- fit_resamples( object = spotify_lda_spec,
                        preprocessor = recipe(playlist_genre ~ ., data = spotify_train_preproce
                        metrics = metric_set(accuracy),
                        resamples = spotify_cv )

# KNN
knn_cv <- fit_resamples( object = spotify_knn_spec_final,
                        preprocessor = recipe(playlist_genre ~ ., data = spotify_train_preproce
                        metrics = metric_set(accuracy),
                        resamples = spotify_cv )

# Random forest
rf_cv <- fit_resamples( object = spotify_rf_spec_final,
                        preprocessor = recipe(playlist_genre ~ ., data = spotify_train_preproce
                        metrics = metric_set(accuracy),
                        resamples = spotify_cv )

# model comparison
spotify_model_compare <- lda_cv %>%
  collect_metrics() %>%
  mutate(model = 'Linear discriminant analysis') %>%
  bind_rows(
    knn_cv %>%
      collect_metrics() %>%
      mutate(model = 'K-nearest neighbours')
    ,
    rf_cv %>%
      collect_metrics() %>%
      mutate(model = 'random forest')
  ) %>%
  dplyr::select(-".config")
spotify_model_compare
## # A tibble: 3 x 6
##   .metric  .estimator  mean     n std_err model
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy multiclass 0.460     5 0.0114  Linear discriminant analysis
## 2 accuracy multiclass 0.524     5 0.00971 K-nearest neighbours
## 3 accuracy multiclass 0.547     5 0.00994 random forest

# display the spotify example preprocessed data
```

```
kable(spotify_model_compare, caption = "Comparison the performance of related model using Accu
```

Table 3: Comparison the performance of related model using Accuracy

.metric	.estimator	mean	n	std_err	model
accuracy	multiclass	0.4595556	5	0.0114266	Linear discriminant analysis
accuracy	multiclass	0.5242222	5	0.0097081	K-nearest neighbours
accuracy	multiclass	0.5471111	5	0.0099418	random forest

Discussion: 06 Model evaluation

Fitting with testing dataset

```
spotify_rf <- spotify_rf_spec_final %>%
  fit( playlist_genre ~ ., data = spotify_train_preprocess)

# prediction
spotify_predict <- predict(spotify_rf,
                           new_data = spotify_test_preprocess
                           ) %>%
  bind_cols(
    spotify_test_preprocess %>%
      dplyr::select(playlist_genre)
  )
```

feature importance

```
# feature importance
spotify_rf %>%
  vip()
```

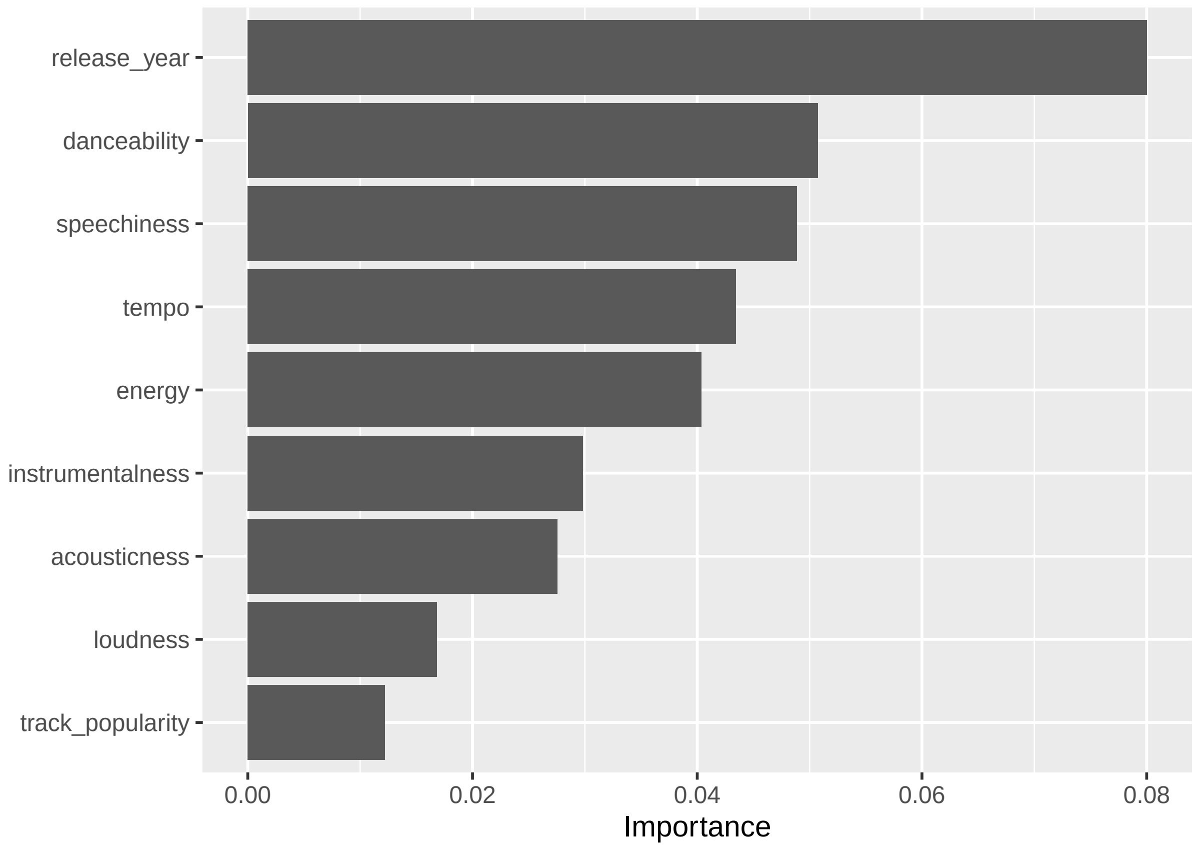

Figure 11: Feature importance following tuned random forest model

```r
## metric evaluate
spotify_metric_test <- spotify_predict %>%
  accuracy(
    truth = playlist_genre,
    estimate = .pred_class
    )
# display the spotify example preprocessed data
kable(spotify_metric_test, caption = "Evaluate the performance of random forest on testing data
```

Table 4: Evaluate the performance of random forest on testing dataset

.metric	.estimator	.estimate
accuracy	multiclass	0.5626667

```r
## metric evaluate
spotify_metric_test_each <- spotify_predict %>%
  group_by(playlist_genre) %>%
  accuracy(
    truth = playlist_genre,
    estimate = .pred_class
    ) %>%
  dplyr::arrange(desc(.estimate))
# display the spotify example preprocessed data
kable(spotify_metric_test_each, caption = "Evaluate the performance of random forest on testin
```

Table 5: Evaluate the performance of random forest on testing dataset related to each genre

playlist_genre	.metric	.estimator	.estimate
rock	accuracy	multiclass	0.752
edm	accuracy	multiclass	0.680
rap	accuracy	multiclass	0.620
r&b	accuracy	multiclass	0.480
latin	accuracy	multiclass	0.440
pop	accuracy	multiclass	0.404