Final Report

Yukta Chavan

25/04/2023

Executive Summary

As a data scientist at Spotify, I was tasked to predict the genre of a song based on its attributes, such as the year the song was released, how "speechy" and danceable the song is, and the tempo of the song. The founders of Spotify are interested in this prediction to enhance their customer's experience and remain the best music streaming service.

First, I cleaned and reduced the dataset to 1,000 songs per genre due to computing power. Then, I explored whether the popularity of songs differs between genres and found that it does, with pop being the most popular genre. I also analyzed speechiness and found that genres such as rap and r&b have a higher speechiness level compared to others.

To build a model to predict the genre based on the provided dataset, I compared three models: linear discriminant analysis, k-nearest neighbours model, and random forest. After evaluating the models using metrics such as AUC, sensitivity, and specificity, I concluded that the random forest model with 100 trees and 5 levels is the best model.

Finally, I tested the model to predict the genre of the song based on its attributes and compared it to the actual genre of the song. The results showed that the model is effective in predicting the genre, with an accuracy of 50.4%.

Based on these findings, I recommend that Spotify continues to utilize this model to enhance its customer's experience by recommending and advertising songs to customers effectively.

Methods

The dataset used in this analysis is the Spotify Songs dataset containing information about different songs from different playlists on Spotify. The dataset was cleaned and reduced to 1,000 songs per genre due to computing power limitations. I analyzed the dataset using R programming language and various packages such as tidyverse, tidymodels, dplyr, and ggplot2.

To build the model to predict the genre based on the provided dataset, I compared three models: linear discriminant analysis, k-nearest neighbours model, and random forest. I evaluated the models using metrics such as AUC, sensitivity, and specificity.

Results

The popularity of songs differed significantly among genres, with Pop being the most popular genre, followed by latin and rock. The speechiness of songs also differed significantly among genres, with rap being the most speechy genre, followed by r&b and latin. The track popularity generally did not had any significant change over time. The evaluation of models revealed that Random Forest with 100 trees and 5 levels was the best model for predicting the song's genre based on AUC, sensitivity, and specificity metrics. The prediction accuracy of the Random Forest model was 0.504.

Discussion:

The analysis showed that some variables such as danceability, speechiness, and tempo can be used to predict the genre of a song.To the check the popularity of genre, we constructed a side-by-side box plot (figure 1), it resulted that the popularity of a song also differed significantly among genres, indicating that some genres are more popular than others. From figure 2,we see that the speechiness of a song also differed significantly among genres, with rap being the most speechy genre, followed by R&B and latin. The line graph (figure 3) illustrates that in 1990s, r&b genre was the only and most popular genre followed with the rock but however,there was no significant change over the time period.Later, we discussed three models- the linear discriminant analysis,k-nearest neighbours and random forest. The best model for predicting the song's genre was found to be Random Forest with 100 trees and 5 levels, which had an accuracy of 50.4% which was highest then the accuracy of lda with 41.8% which was least and knn with 47.53%.

Conclusion:

In conclusion, this project demonstrated that it is possible to predict the genre of a song based on its attributes such as danceability, speechiness, and tempo. The analysis showed that some genres are more popular than others, and the track popularity didnt really changed over time. The Danceability, Speechiness, and Tempo variables were found to be the most important variables in predicting the song's genre, indicating that these variables play a crucial role in determining the song's genre.The Rf model gave the best prediction on the research as compared to other models providing maximum accuracy. This information can be useful for Spotify to better enhance their customer's experience and remain the best music streaming service available.

Appendix

Load and clean the data

We will load the data from the provided URL and check for missing values, duplicates, and irrelevant columns. We will also select only 1000 songs per genre to reduce the dataset's size.

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

```
## Rows: 32833 Columns: 23
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (10): track_id, track_name, track_artist, track_album_id, track_album_na...
## dbl (13): track_popularity, danceability, energy, key, loudness, mode, speec...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
spotify_songs
```

```
## # A tibble: 32,833 x 23
##    track_id               track_name track_artist track_popularity track_album_id
##    <chr>                  <chr>      <chr>                   <dbl> <chr>
##  1 6f807x0ima9a1j3VPbc7~ I Don't C~ Ed Sheeran                 66 2oCs0DGTsRO98~
##  2 0r7CVbZTWZgbTCYdfa2P~ Memories ~ Maroon 5                   67 63rPSO264uRjW~
##  3 1z1Hg7Vb0AhHDiEmnDE7~ All the T~ Zara Larsson               70 1HoSmj2eLcsrR~
##  4 75FpbthrwQmzHlBJLuGd~ Call You ~ The Chainsm~               60 1nqYsOef1yKKu~
##  5 1e8PAfcKUYoKkxPhrHqw~ Someone Y~ Lewis Capal~               69 7m7vv9wlQ4i0L~
```

```
##   6 7fvUMiyapMsRRxr07cU8~ Beautiful~ Ed Sheeran                 67 2yiy9cd2QktrN~
##   7 2OAylPUDDfwRGfe0lYql~ Never Rea~ Katy Perry                 62 7INHYSeusaFly~
##   8 6b1RNvAcJjQH73eZO4BL~ Post Malo~ Sam Feldt                  69 6703SRPsLkS4b~
##   9 7bF6tCO3gFb8INrEDcjN~ Tough Lov~ Avicii                     68 7CvAfGvq4RlIw~
##  10 1IXGILkPm0tOCNeq00kC~ If I Can'~ Shawn Mendes               67 4QxzbfSsVryEQ~
## # i 32,823 more rows
## # i 18 more variables: track_album_name <chr>, track_album_release_date <chr>,
## #   playlist_name <chr>, playlist_id <chr>, playlist_genre <chr>,
## #   playlist_subgenre <chr>, danceability <dbl>, energy <dbl>, key <dbl>,
## #   loudness <dbl>, mode <dbl>, speechiness <dbl>, acousticness <dbl>,
## #   instrumentalness <dbl>, liveness <dbl>, valence <dbl>, tempo <dbl>,
## #   duration_ms <dbl>
```

Missing values

```
colSums(is.na(spotify_songs))%>%
  kable()
```

	x
track_id	0
track_name	5
track_artist	5
track_popularity	0
track_album_id	0
track_album_name	5
track_album_release_date	0
playlist_name	0
playlist_id	0
playlist_genre	0
playlist_subgenre	0
danceability	0
energy	0
key	0
loudness	0
mode	0
speechiness	0
acousticness	0
instrumentalness	0
liveness	0
valence	0
tempo	0
duration_ms	0

```
spotify_songs <- na.omit(spotify_songs)
```

Removing the irrelevant columns

```r
playlist_col <- dplyr::select(spotify_songs, track_popularity, track_album_release_date,
                          playlist_genre, danceability , speechiness, tempo)
```

Creating a new column for to year the song was released & converting year and genre as factor

```r
playlist_col$track_album_release_date <- as.character(playlist_col$track_album_release_date, "%m/%d/%Y")
playlist_col$year <- substr(playlist_col$track_album_release_date,1,4)
dplyr::glimpse(playlist_col)
```

```
## Rows: 32,828
## Columns: 7
## $ track_popularity         <dbl> 66, 67, 70, 60, 69, 67, 62, 69, 68, 67, 58, 6~
## $ track_album_release_date <chr> "2019-06-14", "2019-12-13", "2019-07-05", "20~
## $ playlist_genre           <chr> "pop", "pop", "pop", "pop", "pop", "pop", "po~
## $ danceability             <dbl> 0.748, 0.726, 0.675, 0.718, 0.650, 0.675, 0.4~
## $ speechiness              <dbl> 0.0583, 0.0373, 0.0742, 0.1020, 0.0359, 0.127~
## $ tempo                    <dbl> 122.036, 99.972, 124.008, 121.956, 123.976, 1~
## $ year                     <chr> "2019", "2019", "2019", "2019", "2019", "2019~
```

```r
playlist_col<- playlist_col %>%
  dplyr::mutate(playlist_genre = forcats::as_factor(playlist_genre), year = as.numeric(year) )
dplyr::glimpse(playlist_col)
```

```
## Rows: 32,828
## Columns: 7
## $ track_popularity         <dbl> 66, 67, 70, 60, 69, 67, 62, 69, 68, 67, 58, 6~
## $ track_album_release_date <chr> "2019-06-14", "2019-12-13", "2019-07-05", "20~
## $ playlist_genre           <fct> pop, pop, pop, pop, pop, pop, pop, pop, pop, ~
## $ danceability             <dbl> 0.748, 0.726, 0.675, 0.718, 0.650, 0.675, 0.4~
## $ speechiness              <dbl> 0.0583, 0.0373, 0.0742, 0.1020, 0.0359, 0.127~
## $ tempo                    <dbl> 122.036, 99.972, 124.008, 121.956, 123.976, 1~
## $ year                     <dbl> 2019, 2019, 2019, 2019, 2019, 2019, 2019, 201~
```

Slicing to 1000 songs

```r
playlist_col <- playlist_col %>% group_by(playlist_genre)
songs_slicing <- playlist_col %>% slice_sample(n = 1000)
table(songs_slicing$playlist_genre)
```

```
##
##   pop   rap  rock latin   r&b   edm
##  1000  1000  1000  1000  1000  1000
```

Exploratory data analysis

We will perform a detailed analysis of the data and answer the founders' questions about the popularity of songs, speechiness of each genre, and how track popularity changes over time.

Does the popularity of songs differ between genres?

```
ggplot(songs_slicing, aes(x = playlist_genre, y = track_popularity, fill = playlist_genre)) +
  geom_boxplot() +
  labs(title = "Popularity across genres", x = "Genre", y = "Popularity",
       caption = "Figure 1: Side-by-side boxplots for how does the popularity of songs differ between g
```

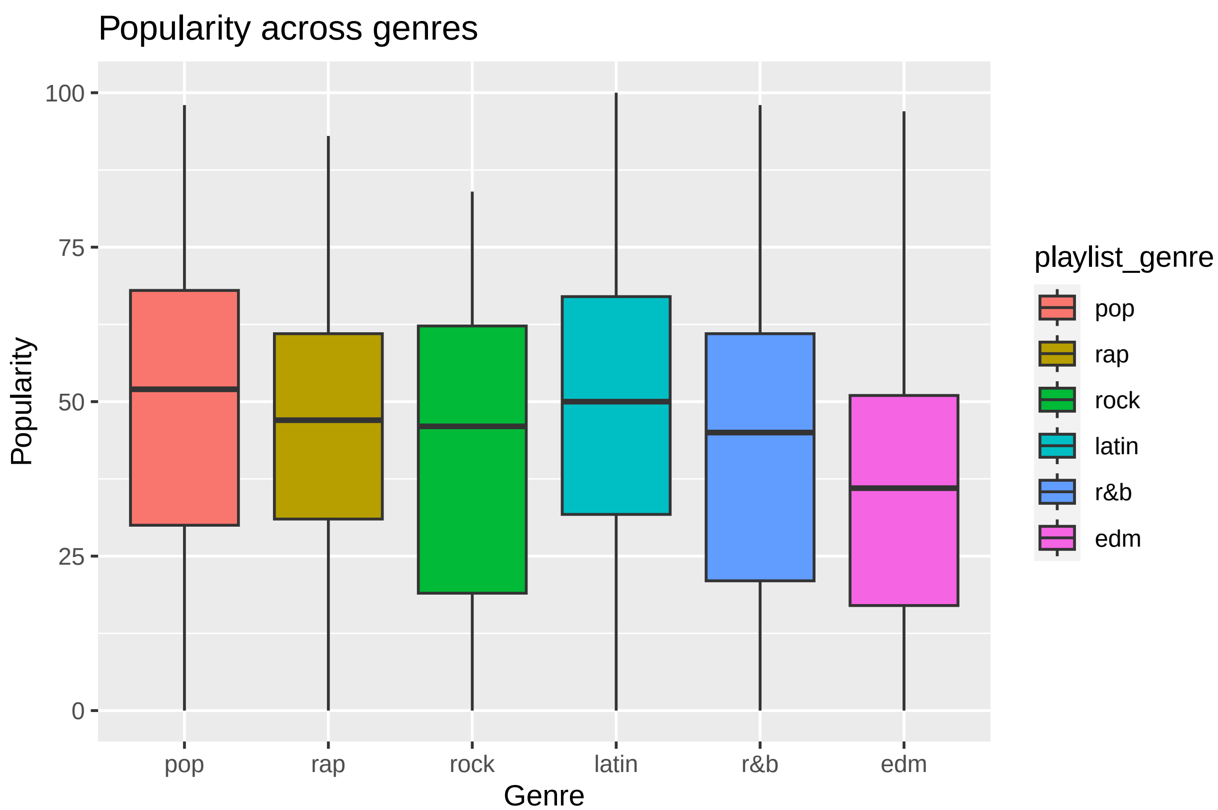

Figure 1: Side−by−side boxplots for how does the popularity of songs differ between genres

From this plot, we can see that some genres, such as pop and latin, have higher median popularity scores than other genres, such as rap and edm.

Is there a difference in speechiness for each genre?

```
speechiness_genre <- songs_slicing %>%
  group_by(playlist_genre) %>%
  summarise(avg = mean(speechiness) )
ggplot(speechiness_genre, aes(x = playlist_genre, y = avg, fill = playlist_genre)) +
  geom_bar(stat = "identity") +
  labs(title = "Speechiness across genres", x = "Genre", y = "Mean Speechiness",
       caption = "Figure 2: Side-by-side boxplots for difference in speechiness for each genre")
```

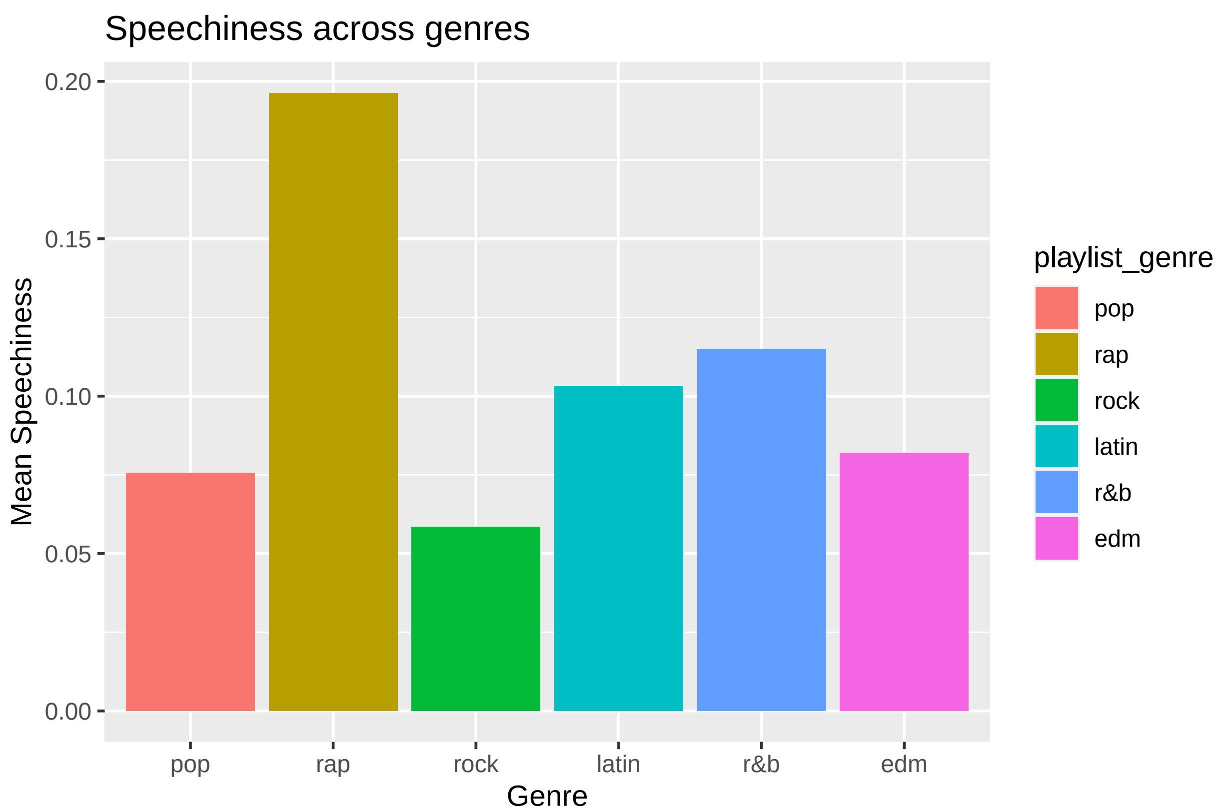

Figure 2: Side−by−side boxplots for difference in speechiness for each genre

From this plot, we can see that some genres, such as rap and r&b, have higher median speechiness scores than other genres, such as rock and pop.

How does track popularity change over time?

Grouping by year and average popularity

```
popularity_years <- songs_slicing %>%
  group_by(playlist_genre, year) %>%
  summarise(avg = mean(track_popularity) )
```

```
## 'summarise()' has grouped output by 'playlist_genre'. You can override using
## the '.groups' argument.
```

```
ggplot(popularity_years, aes(x =  year, y = avg, group=playlist_genre, color = playlist_genre)) +
  geom_line() +
  labs(title = "Track Popularity", x = "Year of release",y = "Mean Popularity",
       caption = "Figure 3: Line Graph for track popularity change over time") +
  theme(legend.position = "none",axis.text.x = element_text(angle = 90, hjust = 1))
```

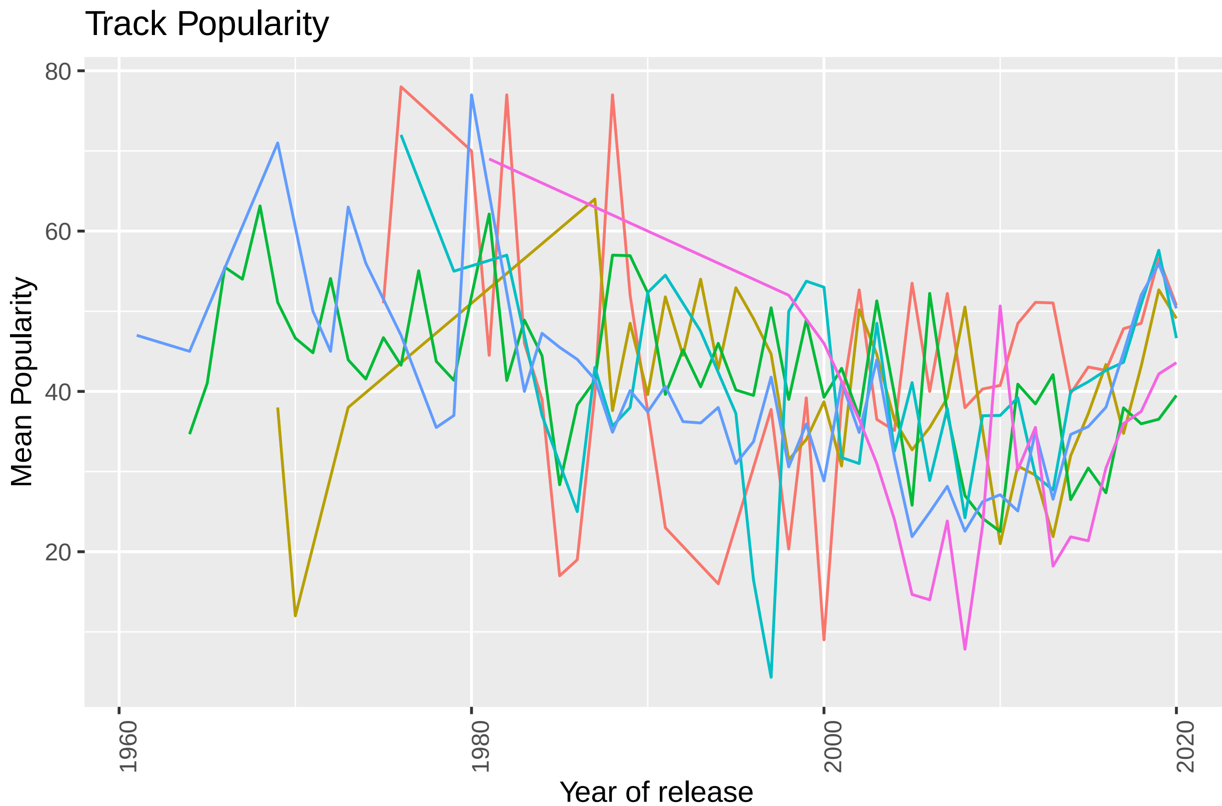

Figure 3: Line Graph for track popularity change over time

There is the no significant change in the popularity of the songs over the time. R&B genre was the most popular in earlier times.

Splitting

```
set.seed(1873167)
songs_splitting <- initial_split(songs_slicing)
songs_splitting
```

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

```
songs_training <- training( songs_splitting )
songs_testing <- testing( songs_splitting )

dplyr::glimpse(songs_training)
```

```
## Rows: 4,500
## Columns: 7
## Groups: playlist_genre [6]
## $ track_popularity         <dbl> 25, 63, 0, 57, 54, 66, 0, 9, 49, 60, 7, 31, 7~
## $ track_album_release_date <chr> "2012-10-30", "1988-05-15", "2007-05-01", "20~
## $ playlist_genre           <fct> rap, rock, rock, latin, edm, rap, r&b, r&b, e~
```

```
## $ danceability             <dbl> 0.675, 0.530, 0.775, 0.836, 0.424, 0.689, 0.3~
## $ speechiness              <dbl> 0.3430, 0.0313, 0.0303, 0.1970, 0.0358, 0.053~
## $ tempo                    <dbl> 90.411, 174.441, 114.040, 100.010, 124.657, 9~
## $ year                     <dbl> 2012, 1988, 2007, 2020, 2015, 2016, 2003, 201~
```

Feature Selection

We will select the most important features that contribute significantly to predicting the genre of a song.

Using recipe to use the data for models and removing the 'track_album_release_date' variable

```
songs_recipe <- recipe( playlist_genre ~ track_popularity+tempo+speechiness+danceability+year,
                        data = songs_training) %>%
                        step_rm(contains("date")) %>%
                        step_dummy( all_nominal(), -all_outcomes() ) %>%
                        step_normalize( all_predictors() ) %>%
                        prep()
songs_recipe
```

```
##

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

##

## -- Inputs

## Number of variables by role

## outcome:   1
## predictor: 5

##

## -- Training information

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

##

## -- Operations

## * Variables removed: <none> | Trained

## * Dummy variables from: <none> | Trained

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

```r
songs_juiced <- juice(songs_recipe)
songs_juiced
```

```
## # A tibble: 4,500 x 6
##    track_popularity  tempo speechiness danceability   year playlist_genre
##               <dbl>  <dbl>       <dbl>        <dbl>  <dbl> <fct>
##  1           -0.707 -1.12        2.40         0.163  0.109 rap
##  2            0.825  1.97       -0.743       -0.833 -1.93  rock
##  3           -1.72  -0.251      -0.753        0.849 -0.315 rock
##  4            0.583 -0.767       0.929        1.27   0.787 latin
##  5            0.462  0.139      -0.697       -1.56   0.363 edm
##  6            0.946 -0.950      -0.524        0.259  0.448 rap
##  7           -1.72  -1.28       -0.746       -2.32  -0.654 r&b
##  8           -1.35  -0.226      -0.385       -0.716  0.363 r&b
##  9            0.260  0.157      -0.427       -0.805  0.702 edm
## 10            0.704  1.32       -0.709       -2.81  -2.60  rock
## # i 4,490 more rows
```

```r
songs_bake <- bake(songs_recipe, new_data = songs_testing)
songs_bake
```

```
## # A tibble: 1,500 x 6
## # Groups:   playlist_genre [6]
##    track_popularity   tempo speechiness danceability   year playlist_genre
##               <dbl>   <dbl>       <dbl>        <dbl>  <dbl> <fct>
##  1           2.24   -0.0307     -0.660         0.300  0.702 pop
##  2           0.0992  1.88       -0.134        -0.675  0.533 pop
##  3           1.03    0.257      -0.657        -0.394  0.617 pop
##  4          -1.72    0.630      -0.367        -0.325 -0.315 pop
##  5           0.543   0.0793      0.0212        0.554  0.533 pop
##  6           0.139   0.189       0.112        -0.703  0.363 pop
##  7           0.462   1.07       -0.791        -1.60   0.363 pop
##  8          -0.869  -0.438      -0.739         0.176  0.363 pop
##  9           0.341   1.07       -0.418        -1.57  -0.400 pop
## 10           0.543  -0.510      -0.790        -0.263  0.533 pop
## # i 1,490 more rows
```

```r
skim_without_charts(songs_slicing)
```

Table 2: Data summary

Name	songs_slicing
Number of rows	6000
Number of columns	7
Column type frequency:	
character	1
numeric	5

Group variables	playlist_genre

Variable type: character

skim_variable	playlist_genre	n_missing	complete_rate	min	max	empty	n_unique	whitespace
track_album_release_date	pop	0	1	4	10	0	573	0
track_album_release_date	rap	0	1	4	10	0	594	0
track_album_release_date	rock	0	1	4	10	0	617	0
track_album_release_date	latin	0	1	4	10	0	536	0
track_album_release_date	r&b	0	1	4	10	0	608	0
track_album_release_date	edm	0	1	4	10	0	529	0

Variable type: numeric

skim_variable	playlist_genre	n_missing	complete_rate	mean	sd	p0	p25	p50	p75
track_popularity	pop	0	1	47.79	25.26	0.00	30.00	52.00	68.00
track_popularity	rap	0	1	43.56	22.54	0.00	31.00	47.00	61.00
track_popularity	rock	0	1	41.09	24.92	0.00	19.00	46.00	62.25
track_popularity	latin	0	1	46.89	25.54	0.00	31.75	50.00	67.00
track_popularity	r&b	0	1	41.43	25.30	0.00	21.00	45.00	61.00
track_popularity	edm	0	1	34.91	22.75	0.00	17.00	36.00	51.00
danceability	pop	0	1	0.64	0.13	0.18	0.56	0.65	0.73
danceability	rap	0	1	0.72	0.14	0.23	0.63	0.73	0.82
danceability	rock	0	1	0.52	0.14	0.12	0.42	0.52	0.61
danceability	latin	0	1	0.71	0.11	0.16	0.65	0.73	0.79
danceability	r&b	0	1	0.67	0.14	0.19	0.58	0.68	0.76
danceability	edm	0	1	0.66	0.12	0.21	0.58	0.67	0.74
speechiness	pop	0	1	0.08	0.07	0.02	0.04	0.05	0.08
speechiness	rap	0	1	0.20	0.14	0.03	0.08	0.17	0.29
speechiness	rock	0	1	0.06	0.05	0.02	0.03	0.04	0.07
speechiness	latin	0	1	0.10	0.09	0.02	0.04	0.07	0.13
speechiness	r&b	0	1	0.12	0.10	0.02	0.04	0.07	0.16
speechiness	edm	0	1	0.08	0.06	0.03	0.04	0.06	0.09
tempo	pop	0	1	121.52	26.05	62.62	102.04	120.01	133.00
tempo	rap	0	1	120.76	31.94	38.98	92.38	117.76	146.26
tempo	rock	0	1	125.21	27.99	37.11	105.02	124.54	143.57
tempo	latin	0	1	119.48	29.49	62.83	96.03	112.93	129.76
tempo	r&b	0	1	114.89	28.51	61.67	93.94	108.52	132.95
tempo	edm	0	1	125.28	14.25	75.02	122.99	126.07	128.02
year	pop	0	1	2014.32	6.91	1975.00	2013.00	2016.00	2019.00
year	rap	0	1	2012.81	8.63	1969.00	2008.00	2017.00	2019.00
year	rock	0	1	1997.10	16.27	1964.00	1983.00	2000.00	2011.00
year	latin	0	1	2014.76	6.51	1976.00	2013.00	2017.00	2019.00
year	r&b	0	1	2009.70	10.61	1961.00	2002.00	2015.00	2018.00
year	edm	0	1	2016.71	3.26	1981.00	2015.00	2018.00	2019.00

Model Selection

We will train three models to predict the genre of a song and compare their performance using different evaluation metrics such as AUC, sensitivity, and specificity.

A linear discriminant analysis

```
dplyr::glimpse(songs_training)
```

```
## Rows: 4,500
## Columns: 7
## Groups: playlist_genre [6]
## $ track_popularity         <dbl> 25, 63, 0, 57, 54, 66, 0, 9, 49, 60, 7, 31, 7~
## $ track_album_release_date <chr> "2012-10-30", "1988-05-15", "2007-05-01", "20~
## $ playlist_genre           <fct> rap, rock, rock, latin, edm, rap, r&b, r&b, e~
## $ danceability             <dbl> 0.675, 0.530, 0.775, 0.836, 0.424, 0.689, 0.3~
## $ speechiness              <dbl> 0.3430, 0.0313, 0.0303, 0.1970, 0.0358, 0.053~
## $ tempo                    <dbl> 90.411, 174.441, 114.040, 100.010, 124.657, 9~
## $ year                     <dbl> 2012, 1988, 2007, 2020, 2015, 2016, 2003, 201~
```

```
songs_lda <- discrim_linear( mode = "classification" ) %>%
  set_engine( "MASS" ) %>%
  fit(playlist_genre ~ track_popularity+danceability+speechiness+tempo+year,
      data = songs_training)
# songs_lda
pred_lda <- predict(songs_lda, songs_testing, type = "class")
pred_lda <- pred_lda %>%
  bind_cols(songs_testing$playlist_genre)
```

```
## New names:
## * `` -> `...2`
```

```
names(pred_lda) <- c("predicted","observed")
tab_lda = table(pred_lda$observed, pred_lda$predicted,
                      dnn = c("obs","pred"))
tab_lda
```

```
##        pred
## obs     pop rap rock latin r&b edm
##   pop   103  19   19    50   6  63
##   rap    30 117    6    38  20  26
##   rock   30   4  158     4   9  28
##   latin  43  58    8    86  21  59
##   r&b    39  55   44    29  35  35
##   edm    70  23    2    48   5 110
```

A K-nearest neighbours model with a range of 1 to 100 and 20 levels

```r
set.seed(1873167)

knn_splits=vfold_cv(songs_training,v=20) # 20 levels
knn_tune = parameters(neighbors(range = c(1,100)))
# range of 1 to 100
knn_mode= nearest_neighbor() %>%
  set_engine("kknn") %>%
  set_mode("classification") %>%
  set_args(neighbors = tune())
knn_tune = tune::tune_grid(knn_mode,preprocessor = songs_recipe,
                           resamples = knn_splits,
                           control = tune::control_resamples(save_pred = TRUE))
knn_tune %>% select_best(metric = "roc_auc", n = 20)
```

```
## # A tibble: 1 x 2
##   neighbors .config
##       <int> <chr>
## 1        14 Preprocessor1_Model10
```

```r
knn_best = nearest_neighbor() %>%
  set_engine("kknn") %>%
  set_mode("classification") %>%
  set_args(neighbors = 15)
wf_knn <- workflow() %>%
  add_model(knn_best) %>%
  add_recipe(songs_recipe)

knn_fit <- fit(wf_knn, songs_juiced)
knn_pred <- predict(knn_fit, new_data = songs_bake)
tab_knn = table(songs_bake$playlist_genre,knn_pred$.pred_class,
                dnn = c("obs","pred"))
tab_knn
```

```
##        pred
## obs     pop rap rock latin r&b edm
##   pop    71  26   27    37  34  65
##   rap    14 122    8    37  34  22
##   rock   20   4  163    10  19  17
##   latin  48  52   13   102  27  33
##   r&b    30  48   23    42  70  24
##   edm    38  16    7    24   6 167
```

A random forest with 100 trees and 5 levels.

```r
songs_spec <- rand_forest(mtry=tune(),trees=100,min_n=tune()) %>%
  set_mode("classification") %>%
  set_engine("ranger")

songs_wf <- workflow() %>%
  add_recipe(songs_recipe) %>%
```

```r
  add_model(songs_spec)

set.seed(1873167)
trees_folds <- vfold_cv(songs_training,v = 5)
rf_grid <- grid_regular(
  mtry(range=c(1, 5)),
  min_n(range=c(2, 8)),
  levels=5)
rr <- tune_grid(songs_wf,resamples=trees_folds,grid=rf_grid)
rr
```

```
## # Tuning results
## # 5-fold cross-validation
## # A tibble: 5 x 4
##   splits             id    .metrics           .notes
##   <list>             <chr> <list>             <list>
## 1 <split [3600/900]> Fold1 <tibble [50 x 6]> <tibble [0 x 3]>
## 2 <split [3600/900]> Fold2 <tibble [50 x 6]> <tibble [0 x 3]>
## 3 <split [3600/900]> Fold3 <tibble [50 x 6]> <tibble [0 x 3]>
## 4 <split [3600/900]> Fold4 <tibble [50 x 6]> <tibble [0 x 3]>
## 5 <split [3600/900]> Fold5 <tibble [50 x 6]> <tibble [0 x 3]>
```

```r
best_auc <- select_best(rr, "roc_auc")
final_rf <- finalize_model( songs_spec, best_auc )
final_rf
```

```
## Random Forest Model Specification (classification)
##
## Main Arguments:
##   mtry = 1
##   trees = 100
##   min_n = 8
##
## Computational engine: ranger
```

```r
Rf <- rand_forest(mtry=1,trees=100,min_n=8) %>%
  set_mode("classification") %>%
  set_engine("ranger")

wf_rf <- workflow() %>%
  add_model(Rf) %>%
  add_recipe(songs_recipe)

Rf_fit <- fit(wf_rf, songs_juiced)
Rf_pred <- predict(Rf_fit, new_data = songs_bake)
tab_rf = table(songs_bake$playlist_genre,Rf_pred$.pred_class,dnn = c("obs","pred"))
tab_rf
```

```
##        pred
## obs      pop rap rock latin r&b edm
##   pop     99  25   34    37  26  39
##   rap     17 142    8    22  31  17
```

```
##   rock   17   5  182     8  13   8
##   latin  49  49   17   106  28  26
##   r&b    35  47   31    39  67  18
##   edm    37  12   10    26  11 162
```

Model Testing

We will test the best model on new data and evaluate its performance using different metrics.

```r
tab_lda;tab_knn;tab_rf
```

```
##        pred
## obs     pop rap rock latin r&b edm
##   pop   103  19   19    50   6  63
##   rap    30 117    6    38  20  26
##   rock   30   4  158     4   9  28
##   latin  43  58    8    86  21  59
##   r&b    39  55   44    29  35  35
##   edm    70  23    2    48   5 110

##        pred
## obs     pop rap rock latin r&b edm
##   pop    71  26   27    37  34  65
##   rap    14 122    8    37  34  22
##   rock   20   4  163    10  19  17
##   latin  48  52   13   102  27  33
##   r&b    30  48   23    42  70  24
##   edm    38  16    7    24   6 167

##        pred
## obs     pop rap rock latin r&b edm
##   pop    99  25   34    37  26  39
##   rap    17 142    8    22  31  17
##   rock   17   5  182     8  13   8
##   latin  49  49   17   106  28  26
##   r&b    35  47   31    39  67  18
##   edm    37  12   10    26  11 162
```

```r
lda_per <- confusionMatrix(tab_lda)
knn_per <- confusionMatrix(tab_knn)
rf_per <- confusionMatrix(tab_rf)

accuracy_df <- data.frame(
              Model=c("Linear Discriminant Analysis", "K-Nearest Neighbors", "Random Forest"),
              Accuracy=c(lda_per$overall["Accuracy"], knn_per$overall["Accuracy"], rf_per$overall["Accu

accuracy_df
```

```
##                          Model  Accuracy
## 1 Linear Discriminant Analysis 0.4060000
## 2          K-Nearest Neighbors 0.4633333
## 3                Random Forest 0.5053333
```

```r
ggplot(accuracy_df, aes(x=Model, y=Accuracy, fill=Model)) +
  geom_bar(stat="identity") +
  labs(title = "Accuracy of Different Models", x = "Model",y = "Accuracy",
       caption = "Figure 4: Comparision of Model Accuracies") +
  theme(plot.title = element_text(hjust = 0.5), legend.position="none")
```

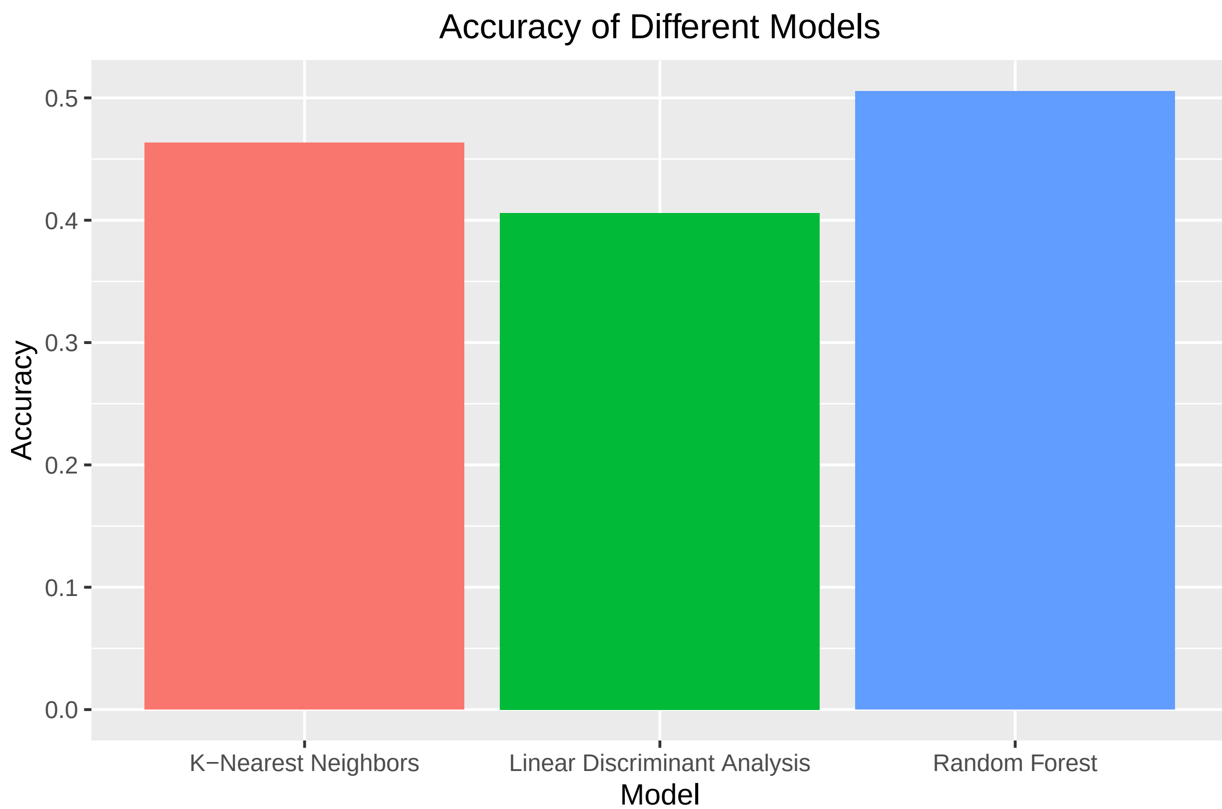

Figure 4: Comparision of Model Accuracies

```r
lda_per
```

```
## Confusion Matrix and Statistics
##
##        pred
## obs      pop rap rock latin r&b edm
##   pop    103  19   19    50   6  63
##   rap     30 117    6    38  20  26
##   rock    30   4  158     4   9  28
##   latin   43  58    8    86  21  59
##   r&b     39  55   44    29  35  35
##   edm     70  23    2    48   5 110
##
## Overall Statistics
##
##                Accuracy : 0.406
##                  95% CI : (0.381, 0.4313)
##     No Information Rate : 0.214
##     P-Value [Acc > NIR] : < 2.2e-16
```

```
##
##                   Kappa : 0.286
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: pop Class: rap Class: rock Class: latin Class: r&b
## Sensitivity             0.32698     0.4239      0.6667      0.33725    0.36458
## Specificity             0.86751     0.9020      0.9406      0.84819    0.85613
## Pos Pred Value          0.39615     0.4937      0.6781      0.31273    0.14768
## Neg Pred Value          0.82903     0.8741      0.9376      0.86204    0.95170
## Prevalence              0.21000     0.1840      0.1580      0.17000    0.06400
## Detection Rate          0.06867     0.0780      0.1053      0.05733    0.02333
## Detection Prevalence    0.17333     0.1580      0.1553      0.18333    0.15800
## Balanced Accuracy       0.59725     0.6629      0.8036      0.59272    0.61035
##                      Class: edm
## Sensitivity             0.34268
## Specificity             0.87447
## Pos Pred Value          0.42636
## Neg Pred Value          0.83011
## Prevalence              0.21400
## Detection Rate          0.07333
## Detection Prevalence    0.17200
## Balanced Accuracy       0.60857
```

```
knn_per
```

```
## Confusion Matrix and Statistics
##
##         pred
## obs      pop rap rock latin r&b edm
##   pop     71  26   27    37  34  65
##   rap     14 122    8    37  34  22
##   rock    20   4  163    10  19  17
##   latin   48  52   13   102  27  33
##   r&b     30  48   23    42  70  24
##   edm     38  16    7    24   6 167
##
## Overall Statistics
##
##                Accuracy : 0.4633
##                  95% CI : (0.4379, 0.489)
##     No Information Rate : 0.2187
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3556
##
##  Mcnemar's Test P-Value : 0.0003183
##
## Statistics by Class:
##
##                      Class: pop Class: rap Class: rock Class: latin Class: r&b
## Sensitivity             0.32127    0.45522      0.6763       0.4048    0.36842
```

```
## Specificity            0.85223   0.90666    0.9444     0.8614   0.87252
## Pos Pred Value         0.27308   0.51477    0.6996     0.3709   0.29536
## Neg Pred Value         0.87903   0.88440    0.9384     0.8776   0.90499
## Prevalence             0.14733   0.17867    0.1607     0.1680   0.12667
## Detection Rate         0.04733   0.08133    0.1087     0.0680   0.04667
## Detection Prevalence   0.17333   0.15800    0.1553     0.1833   0.15800
## Balanced Accuracy      0.58675   0.68094    0.8104     0.6331   0.62047
##                      Class: edm
## Sensitivity              0.5091
## Specificity              0.9224
## Pos Pred Value           0.6473
## Neg Pred Value           0.8704
## Prevalence               0.2187
## Detection Rate           0.1113
## Detection Prevalence     0.1720
## Balanced Accuracy        0.7158
```

```
rf_per
```

```
## Confusion Matrix and Statistics
##
##        pred
## obs      pop rap rock latin r&b edm
##   pop     99  25   34    37  26  39
##   rap     17 142    8    22  31  17
##   rock    17   5  182     8  13   8
##   latin   49  49   17   106  28  26
##   r&b     35  47   31    39  67  18
##   edm     37  12   10    26  11 162
##
## Overall Statistics
##
##                Accuracy : 0.5053
##                  95% CI : (0.4797, 0.5309)
##     No Information Rate : 0.188
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4064
##
##  Mcnemar's Test P-Value : 0.0005084
##
## Statistics by Class:
##
##                      Class: pop Class: rap Class: rock Class: latin Class: r&b
## Sensitivity              0.3898    0.50714      0.6454      0.44538    0.38068
## Specificity              0.8708    0.92213      0.9581      0.86609    0.87160
## Pos Pred Value           0.3808    0.59916      0.7811      0.38545    0.28270
## Neg Pred Value           0.8750    0.89074      0.9211      0.89224    0.91370
## Prevalence               0.1693    0.18667      0.1880      0.15867    0.11733
## Detection Rate           0.0660    0.09467      0.1213      0.07067    0.04467
## Detection Prevalence     0.1733    0.15800      0.1553      0.18333    0.15800
## Balanced Accuracy        0.6303    0.71464      0.8018      0.65573    0.62614
##                      Class: edm
## Sensitivity              0.6000
```

```
## Specificity              0.9220
## Pos Pred Value           0.6279
## Neg Pred Value           0.9130
## Prevalence               0.1800
## Detection Rate           0.1080
## Detection Prevalence     0.1720
## Balanced Accuracy        0.7610
```