Predicting the genre of songs

2022-08-15

Executive Summary

An audio streaming and media service provider Spotify, having over 365 million monthly active users, including 165 million paying subscribers. As Spotify is the world's largest music streaming service provider, the founders are intrersted in trying to predict which genre a song belongs to.

Forecasting a genre of a song helps them to suggest better recommendation, advertise songs to customers and more effectively update compilation playlists. The founders have asked to cleaned the first and reduced to 1000 songs per genre due to computing power.

After completing a full analysis on different statistical models, it was determind that the year when song was released have the highest impact on genre, followed by the tempo, and speechiness of a song. Whilst mode of a song have an negligible effect on genre, followed by key and liveness of a song.

Methods

(Bivariate summaries, model selection, model diagnostics)

They are interested in how the following factors can predict a song's genre:

- the year the song was released,
- how "speechy" the song is,
- how danceable the song is, and
- the tempo of the song

The following analysis which was completed in R version 4.01.1 using the tidyverse and dplyr packages. The following analysis is done as per the guidelines of 1000 songs per genre. Note: You might see slight change in below mentioned values due to random split for the observation of 1000 songs per genre.

The first step in reviewing the data was to consider the relationship between genre and each of the above variables/predictors. The following relationship were identified:

- All genre have the same median of around 2018-2019 as publication-year except genre rock. Rock type genre have the median of around 2001 as publication-year. Rock type genre have the highest IQR of 30 years whilst edm type genre have the lowest IQR of 4 years. Genre categories of edm, latin, and pop have several outliers lie between year 1980 to 2000. Genre r&b and rap have the several outliers between year 1960 to 1980. Whilst rock type genre have no outliers.

- All genre have the same median of around 0.05 speechiness except genre rap. Rap type genre have the median of 0.175 speechiness. Rap type genre have the highest IQR of 0.211 whilst rock type genre have the lowest IQR of 0.033. Genre categories of edm, latin, pop, and rock have several outliers lie between 0.125 to 0.5 speechiness. Genre r&b have the several outliers between 0.35 to 1 speechiness. Whilst rap type genre have few outliers between 0.62 to 0.875 speechiness.

- Rap type genre have the highest median of 0.74 danceability. Whilst rock type genre have the lowest median of 0.525 danceability. Rock type genre have the highest IQR of 0.1965 whilst latin type genre have the lowest IQR of 0.1442. All genre categories have several outliers lie between 0 to 0.32 danceability. Whilst rock type genre have no outliers.

- Edm type genre have the highest median of 127 tempo. Whilst r&b type genre have the lowest median of 109 tempo. Rap type genre have the highest IQR of 50 whilst edm type genre have the lowest IQR of 6. Edm type genre have the highest number of outliers ranging from 70 to 200 tempo. Whilst rap type genre have no outliers.

Overall there appears to be weak relationship between each features and genre. However, the relationship between publication year and song's genre is the strongest among these four variables.

The founders have asked specific questions:

- Does the popularity of songs differ between genres?

  Yes, it seems like pop type genre have the highest median popularity of 51, whilst edm type genre have the lowest median popularity of 35. The order of song's genres with popularity(descending order):

  pop, latin, rap, rock, r&b, and edm

- Is there a difference in speechiness for each genre?

  Yes, it seems like rap type genre have the highest median of 0.175 speechiness. Whilst all other categories of genre have the same median of around 0.05 speechiness. Hence, rap type genre have signicant difference among others.

- How does track popularity change over time?

  Track popularity have not changed drastically over the time. Track popularity is approximate over few decades (1963 - 2017). In recent years (2018-2020) popularity increased upto 98. Hence, there is no signoficant change in track popularity over time.

Results

(Exploratory data analysis, Model interpretation, Model Evaluation)

Founders demanded to compare the following three models:

- 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.

After a thorough analysis, predicting song's genre should be done using Random Forest model with following features which have passed the threshold of 0.04 importance-significance.

- the year the song was released,
- the tempo of the song
- how "speechy" the song is,

Model select is done by the varity of evaluation tests such as Sensitivity, Specificity, AUC-score, and Roc-auc curve. Where, Random forest performed well in the all the evaluation metrics.

In evaluation of Linear discriminant analysis, model performed well to predict/identify only few of the song's genre such as rock and edm type.

In evaluation of K-nearest neighbours, model performed similar to Linear discriminant analysis, identifying only rock type and edm type genre.

In evaluation of Random forest, model performed to identify rock type, edm type, and rap type genre. Furthermore, Random forest model gave the best performance in AUC-score, Sensitivity, and Specificity among all statistical models.

Following analysis is based on the Random Forest Model (Selected model).

Model displayed that publication year had the maximum impact on song's genre with feature importance around 0.085. Tempo had an importance of 0.045, and Speechiness had an importance of 0.04. Furthermore danceability and energy with importance-significance level near to 0.04. Acousticness and instrumentalness of a song had a moderate impact on genre with importance-significance level near to 0.02.

Model is well tuned with 25 different combinations due to 5 levels and 2 tuned hyperparameters. Furthermore, model is generalised with the 5 fold cross-validation.

Hence, model suggests that the best performance of forecasting of song's genre is achienved by closly observing song's publication year, tempo and speechiness.

Discussion

(Prediction)

Prediction on the test set was apllied with the 95% confidence interval. Therefore, following statements were derived from a specific (95% confidence) criteria.

By observing the predctions on the test set, Model have sensitivity around 60%,

> A model with high sensitivity means that model is not correctly predicting most of the positive results. In this case model is not performing well with identifying the positive observations.

By observing the predctions on the test set, Model have specificity around 90%,

> A model with high specificity means that model is correctly predicting most of the negative results. In this case model is performing well with identifying the negative observations.

By observing the predctions on the test set, Model have around 85% of AUC score,

> A model with high AUC score suggests that model is forecasting all the categories of song's genre correct and accurate. In this case model is performing well with identifying the correct category of genre.

Model (random forest) gave better AUC score for specific categories of genre.(Mentioned in the Results section).

Hence, we can cay with 95% confidence that the model performes well to predict/identify the rock, edm, and rap type genre. Whilest model is unable to perform well to predict other categories of genre.

Conclusions

After considering the recent situation of Spotify, the world's largest music streaming service provider with over 365 million monthly active users and 165 million paying subscribers, founders wants to know the effect of specific factors (4 variables), including 3 questions related the effectiveness between song's popularity and song's genre, including the condition of 1000 songs per genre.

After reviewing the data (1000 songs per genre), the most affecting factor is the publication year, followed by tempo, danceability and, speechiness of a song.

After completing a full analysis, it was determined that popularity of songs differred between genres. Pop genre have the highest popularity followed by latin, rap, rock, r&b, and edm. Furthermore, difference in speechiness for each genre was found, rap genre have the highest speechiness compare to other categories of genre. Other categories have approximate the same speechiness. Lastly, track popularity have not changed significantly over the past few decades. Hence, there is no proper trend/relationship of popularity over time.

Any future work in this area should consider to identify the latin, r&b, and pop categories of song's genre more efficiently.

Appendix

Data Cleaning

```r
# load packages
pacman::p_load(tidyverse,dplyr,lubridate,inspectdf,tidyr,stringr,tidymodels,skimr,glmx,discrim,themis,da

# load the data
spotify_songs <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/
```

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

```r
# Converted to data type.
spotify_songs$track_album_release_date <- as.Date(spotify_songs$track_album_release_date)

# Removed missing values and created new dataset with 1000 observations per genre
spotify_songs <- na.omit(spotify_songs)
pop <- spotify_songs %>% filter(playlist_genre == "pop")
pop_data <- pop[sample(nrow(pop),size=1000),]
rap <- spotify_songs %>% filter(playlist_genre == "rap")
rap_data <- rap[sample(nrow(rap),size=1000),]
rock <- spotify_songs %>% filter(playlist_genre == "rock")
rock_data <- rock[sample(nrow(rock),size=1000),]
latin <- spotify_songs %>% filter(playlist_genre == "latin")
latin_data <- latin[sample(nrow(latin),size=1000),]
rb <- spotify_songs %>% filter(playlist_genre == "r&b")
rb_data <- rb[sample(nrow(rb),size=1000),]
edm <- spotify_songs %>% filter(playlist_genre == "edm")
edm_data <- edm[sample(nrow(edm),size=1000),]
data <- bind_rows(pop_data,rap_data,rock_data,latin_data,rb_data,edm_data)
View(data)

# Removed variables containing id and name, due to not having any relationship to target variable.
data <- data[,c("playlist_genre","track_popularity","track_album_release_date","speechiness","danceabil

# Renamed variables for readiness.
data <- data %>% rename(genre = playlist_genre)
data <- data %>% rename(year = track_album_release_date)
data <- data %>% rename(popularity = track_popularity)

# Converted target variable to factors.
data$genre <- factor(data$genre)
```

Exploratory Data Analysis

```r
# checked the variable types and proportions of labels in genre.
skim_without_charts(data)
```

Table 1: Data summary

Name	data
Number of rows	6000
Number of columns	15
Column type frequency:	
Date	1
factor	1
numeric	13
Group variables	None

Variable type: Date

skim_variable	n_missing	complete_rate	min	max	median	n_unique
year	0	1	1964-10-02	2020-01-17	2016-10-21	2133

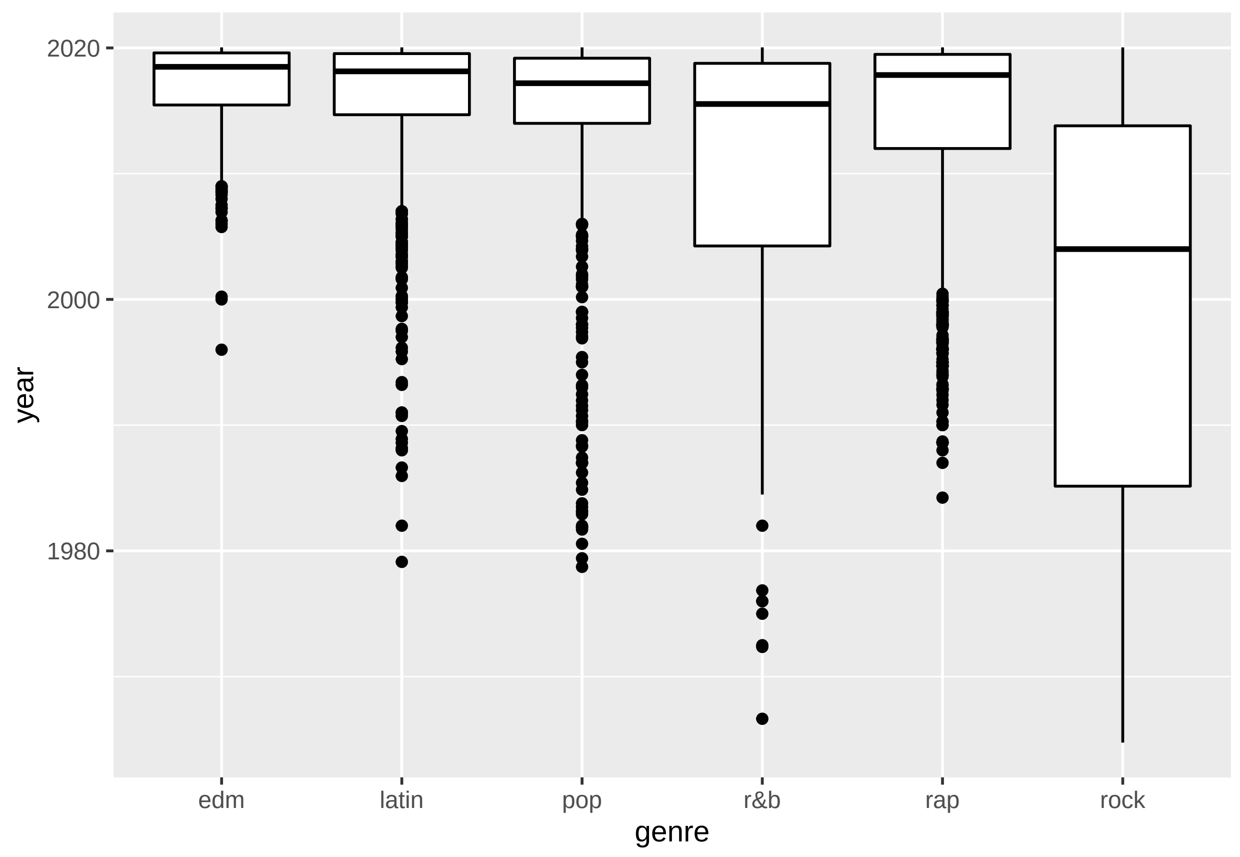
Variable type: factor

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

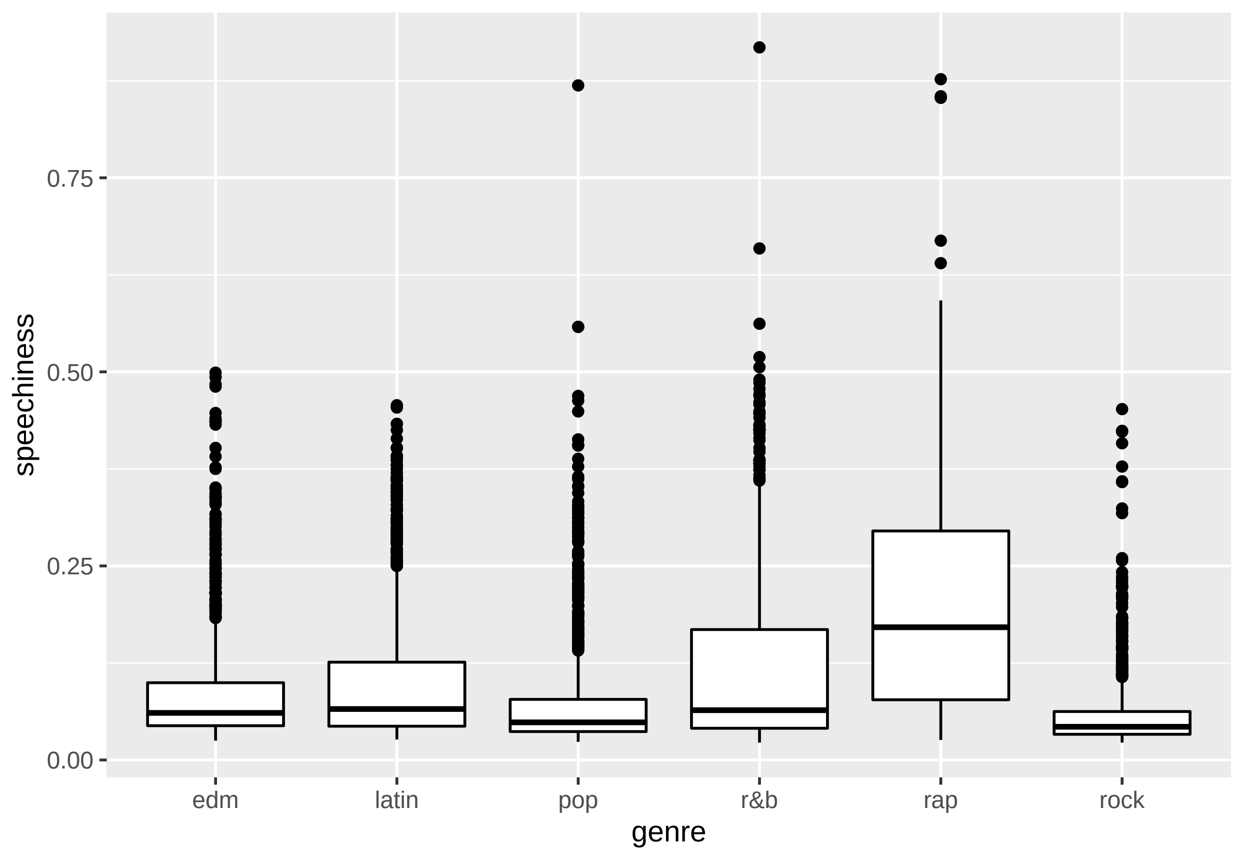
Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
popularity	0	1	42.85	24.98	0.00	25.00	46.00	62.00	98.00
speechiness	0	1	0.11	0.10	0.02	0.04	0.06	0.13	0.92
danceability	0	1	0.65	0.15	0.10	0.56	0.67	0.76	0.98
tempo	0	1	120.92	26.62	37.11	99.95	121.92	133.04	220.25
energy	0	1	0.70	0.18	0.02	0.58	0.72	0.84	1.00
key	0	1	5.36	3.61	0.00	2.00	6.00	9.00	11.00
loudness	0	1	-6.72	2.97	-29.56	-8.15	-6.17	-4.65	-0.38
mode	0	1	0.57	0.50	0.00	0.00	1.00	1.00	1.00
acousticness	0	1	0.18	0.22	0.00	0.02	0.08	0.26	0.99
instrumentalness	0	1	0.08	0.22	0.00	0.00	0.00	0.00	0.99
liveness	0	1	0.19	0.15	0.01	0.09	0.13	0.25	0.99
valence	0	1	0.51	0.23	0.00	0.33	0.51	0.70	0.98
duration_ms	0	1	223650.00	58398.22	29493.00	187143.50	214480.00	250363.25	515867.00

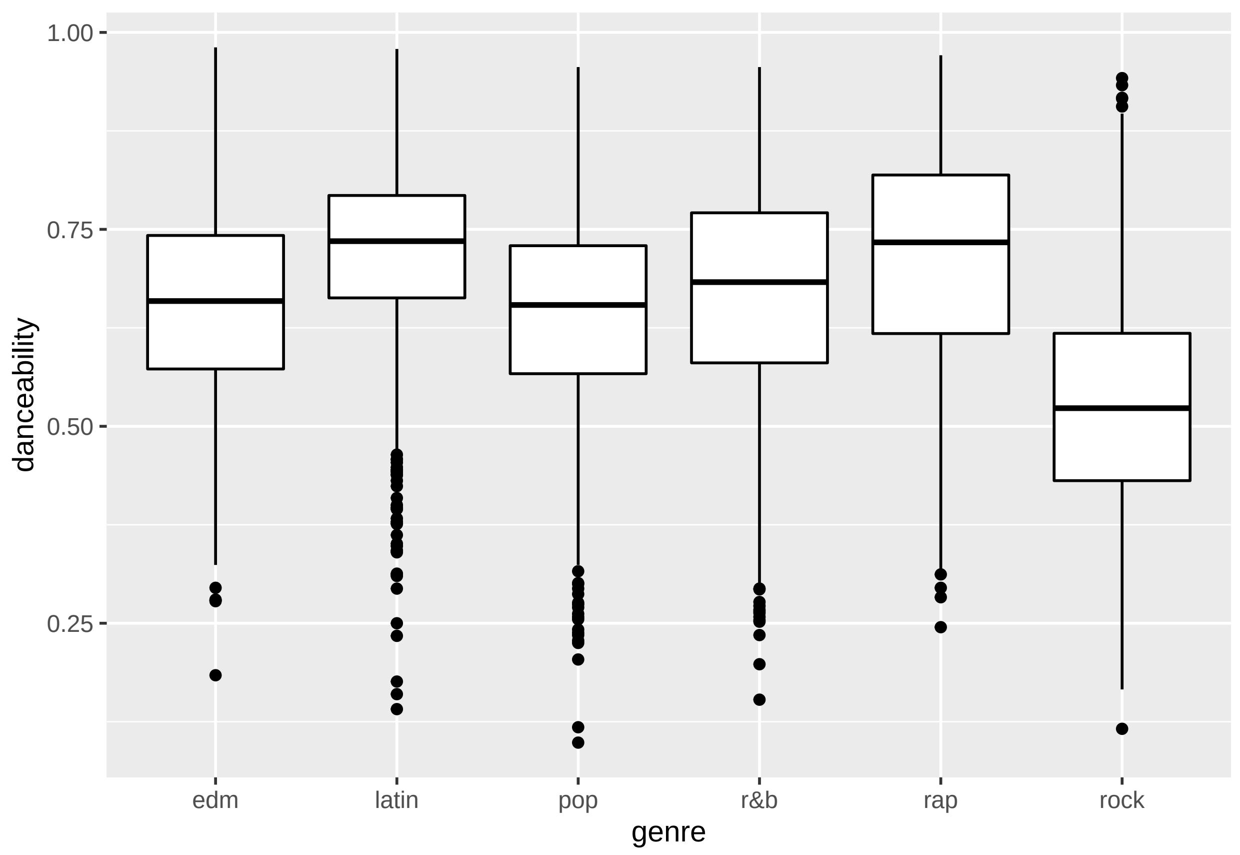
```r
# Box-plot of genre and year.
ggplot(data,aes(x = genre, y = year)) + geom_boxplot(col="black")
```



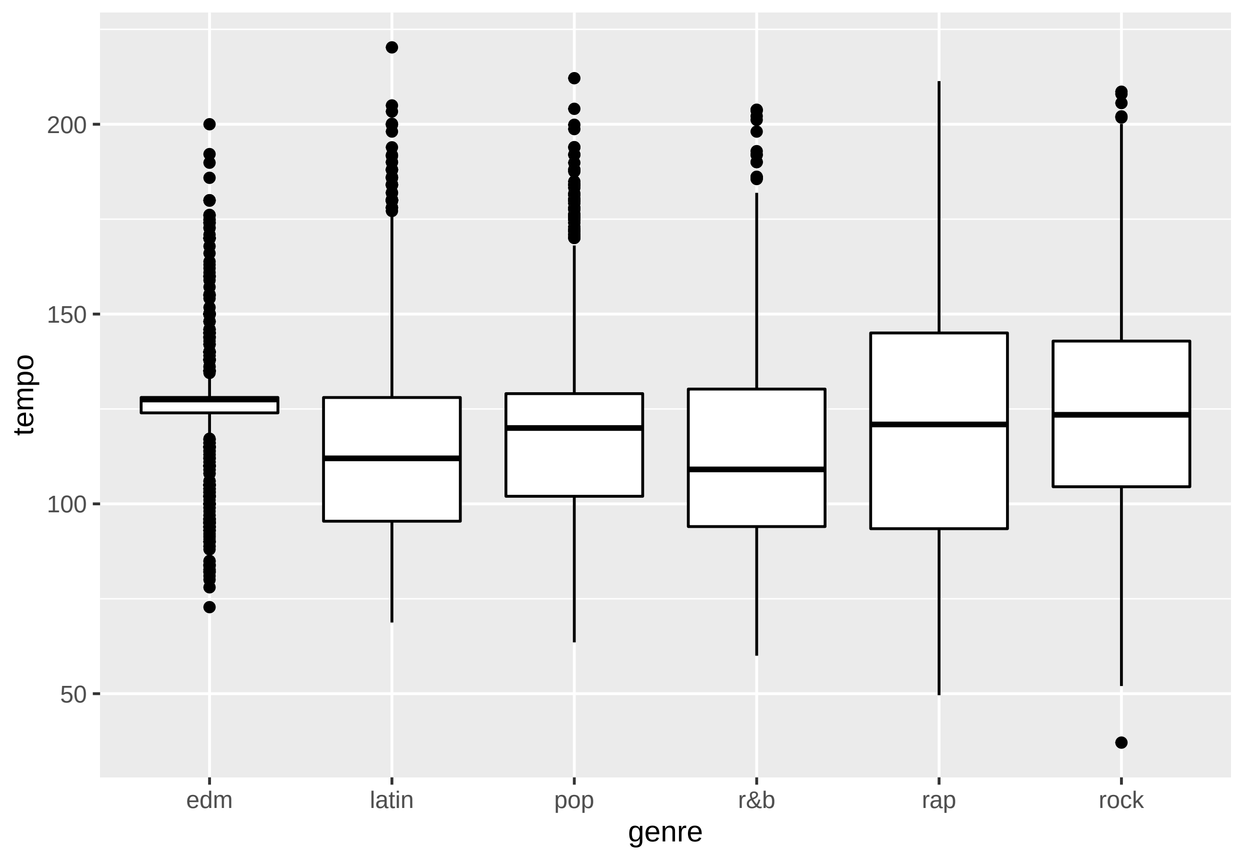
```r
# Box-plot of genre and speechiness.
ggplot(data,aes(x = genre, y = speechiness)) + geom_boxplot(col="black")
```



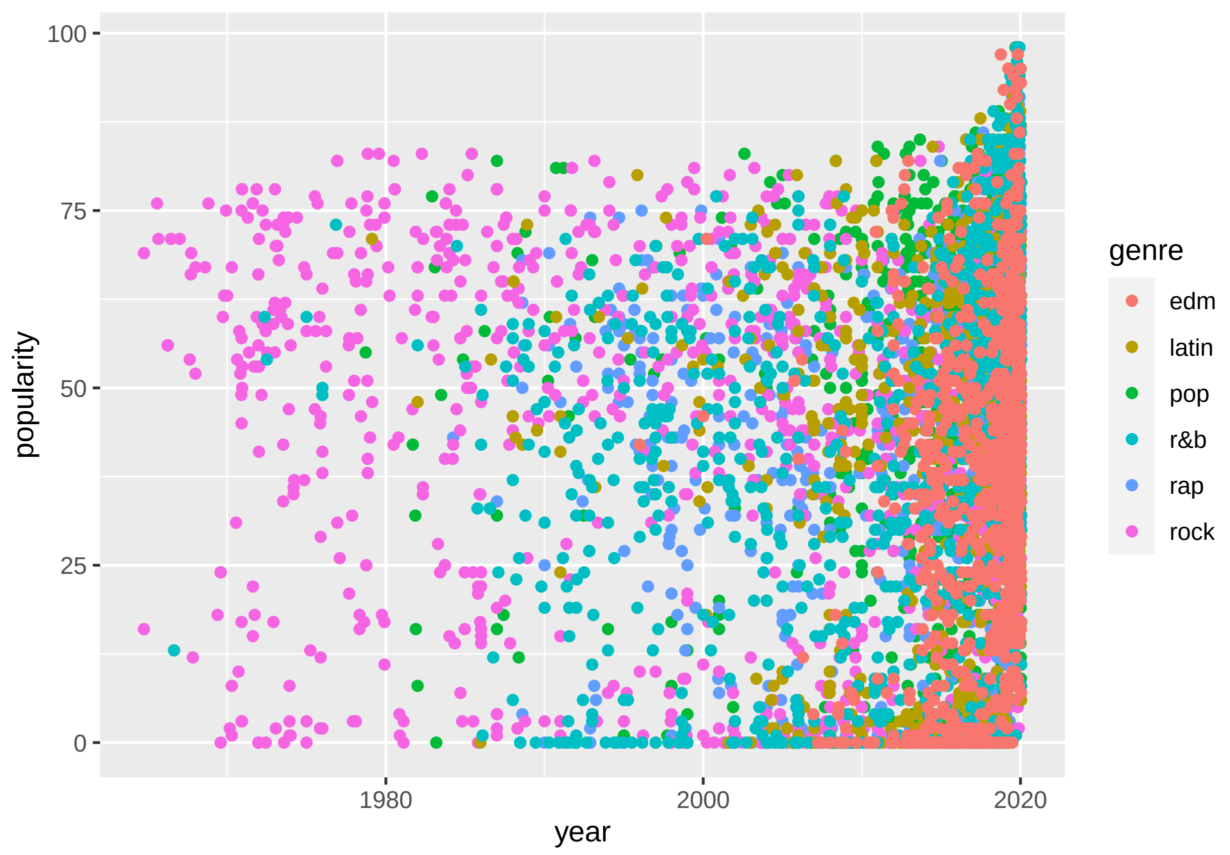
```
# Box-plot of genre and danceability.
ggplot(data,aes(x = genre, y = danceability)) + geom_boxplot(col="black")
```



```
# Box-plot of genre and tempo.
ggplot(data,aes(x = genre, y = tempo)) + geom_boxplot(col="black")
```



```
# Scatter plot of different genre's popularity over the time.
ggplot(data,aes(x=year,y=popularity,colour=genre)) +
  geom_point()
```



```
# Defined change in popularity over time.
period <- as.integer(data$year)
cor(data$popularity,period)
```

```
## [1] 0.05476633
```

Split the data

```
# split the data with 75% training set and 25% testing set.
set.seed( 123 )
split <- initial_split(data, strata = genre )
train <- training(split)
test <- testing(split)
split
```

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

Pre-process the data

```r
# Created a recipe for data preprocessing.
spotify_recipe <- recipe( genre ~ ., data = train ) %>%
  step_date(year, features = c("year")) %>%
  step_rm(year) %>%
  step_zv( all_predictors() ) %>%
  step_normalize( all_predictors() ) %>%
  step_corr( all_predictors() ) %>%
  prep()
spotify_recipe
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor         14
##
## Training data contained 4500 data points and no missing data.
##
## Operations:
##
## Date features from year [trained]
## Variables removed year [trained]
## Zero variance filter removed <none> [trained]
## Centering and scaling for popularity, speechiness, danceability, tempo, e... [trained]
## Correlation filter on <none> [trained]
```

```r
# Data should be scaled in KNN. Whilst, Random forest and LDA do not need scaled data to be fitted.
pre_train <- juice( spotify_recipe )
pre_test <- bake( spotify_recipe, test )
head(pre_test)
```

```
## # A tibble: 6 x 15
##   popularity speechiness danceability  tempo  energy     key loudness   mode
##        <dbl>       <dbl>        <dbl>  <dbl>   <dbl>   <dbl>    <dbl>  <dbl>
## 1     -0.242      0.406        0.0214  2.23    0.539   0.454     1.18   -1.16
## 2     -0.482     -0.754       -1.46    2.35   -1.56    0.454    -0.275  0.863
## 3     -1.52      -0.385        0.587   0.422   1.32    1.01     -0.371 -1.16
## 4      1.47      -0.714        1.20    0.120 -0.0440   0.454    -0.401  0.863
## 5      1.43       0.0499       1.65   -0.590   0.517  -0.930     0.678  0.863
## 6     -0.402      0.377       -0.326   2.23    0.149  -0.0997    0.735  0.863
## # ... with 7 more variables: acousticness <dbl>, instrumentalness <dbl>,
## #   liveness <dbl>, valence <dbl>, duration_ms <dbl>, genre <fct>,
## #   year_year <dbl>
```

Define different model specifications

```r
# LDA
spoty_lda <- discrim_linear( mode = "classification" ) %>%
  set_engine( "MASS" ) %>%
  fit( genre ~ ., data = train )
spoty_lda
```

```
## parsnip model object
##
## Call:
## lda(genre ~ ., data = data)
##
## Prior probabilities of groups:
##       edm     latin       pop       r&b       rap      rock
## 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667
##
## Group means:
##       popularity     year speechiness danceability    tempo    energy      key
## edm     36.12800 17270.37  0.08981147    0.6522747 125.8334 0.8001080 5.281333
## latin   47.07867 16610.65  0.10151880    0.7191920 117.6243 0.7116427 5.460000
## pop     48.23600 16496.51  0.07433893    0.6422833 119.7724 0.7026317 5.418667
## r&b     41.74800 14860.27  0.11702040    0.6659453 114.6195 0.5825831 5.413333
## rap     43.65333 16034.99  0.20043787    0.7128133 121.2132 0.6517140 5.398667
## rock    41.62400 10667.91  0.05778067    0.5186747 125.4044 0.7332879 5.189333
##        loudness      mode acousticness instrumentalness  liveness   valence
## edm   -5.400023 0.5280000   0.08281573       0.19552409 0.2070195 0.3956773
## latin -6.134357 0.5533333   0.21301624       0.05390295 0.1802863 0.5975050
## pop   -6.212443 0.6133333   0.16789800       0.05187755 0.1695867 0.5036800
## r&b   -7.940619 0.5306667   0.26909749       0.02831010 0.1733052 0.5294712
## rap   -6.960176 0.5480000   0.19882537       0.09113007 0.1942030 0.5050320
## rock  -7.529665 0.6666667   0.14041815       0.06749114 0.2197640 0.5327128
##       duration_ms
## edm      219101.8
## latin    217426.8
## pop      216673.8
## r&b      235615.0
## rap      208396.2
## rock     245819.0
##
## Coefficients of linear discriminants:
##                            LD1           LD2           LD3           LD4
## popularity        2.506810e-03 -7.014494e-04 -7.018482e-03  1.744581e-02
## year             -1.868400e-04  6.346297e-05 -1.331490e-04 -5.824235e-05
## speechiness      -3.488524e+00 -4.778846e+00  7.574217e+00  1.851319e+00
## danceability     -4.800470e+00  2.479730e-01 -3.213733e-02  1.184084e+00
## tempo             5.829977e-04  5.257024e-03  5.731557e-03  4.428990e-03
## energy            6.134033e-01  4.138300e+00  2.292468e-01  3.636067e+00
## key              -1.001381e-03  2.004475e-03 -1.149061e-02  7.291967e-03
## loudness         -6.821664e-02  2.589471e-03 -8.913540e-04 -2.778205e-02
## mode              1.240396e-01  2.546107e-02 -3.284954e-02  3.345443e-01
## acousticness     -5.323496e-01 -5.838993e-01 -1.104495e+00  7.943854e-01
## instrumentalness -4.432549e-01  1.421856e+00  1.773915e+00  1.353976e-01
## liveness         -3.062628e-02  5.023241e-02  5.225269e-01  1.461595e-01
## valence           6.075376e-01 -1.616293e+00 -1.961064e+00  1.857562e+00
## duration_ms       4.791793e-09 -1.813226e-06 -4.148510e-06 -5.212601e-06
##                            LD5
## popularity       -1.596640e-02
## year             -5.505201e-05
## speechiness      -1.383067e+00
## danceability      3.622275e+00
## tempo             1.260875e-03
## energy            2.358891e+00
## key              -2.418915e-02
## loudness         -2.877228e-02
## mode             -6.027720e-01
## acousticness      2.323632e+00
## instrumentalness  7.967329e-01
## liveness          1.387549e+00
## valence           3.379045e-01
## duration_ms       6.745835e-06
##
## Proportion of trace:
##    LD1    LD2    LD3    LD4    LD5
## 0.5498 0.2646 0.1339 0.0340 0.0178
```

```
# KNN
knn_spec <- nearest_neighbor( mode = "classification", neighbors = tune() ) %>%
  set_engine( "kknn" )
knn_spec
```

```
## K-Nearest Neighbor Model Specification (classification)
##
## Main Arguments:
##   neighbors = tune()
##
## Computational engine: kknn
```

```
# Random forest
set.seed( 1223 )
rf_cv <- vfold_cv( train, v = 5 )
rf_spec_tune <- rand_forest( mode = "classification",
                             trees = 100,
                             mtry = tune(),
                             min_n = tune() ) %>%
  set_engine( "ranger", importance = "permutation" )
rf_spec_tune
```

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

Tuning of different models

```r
# KNN
set.seed( 1223 )
cv <- vfold_cv( pre_train, v = 5 )
k_grid <- grid_regular( neighbors( range = c( 1, 100 ) ),
                        levels = 20 )
model_control <- control_grid(save_pred = TRUE)
model_metrics <- metric_set(accuracy,sens,spec,roc_auc)
knn_tune <- tune_grid(object = knn_spec,
                      preprocessor = recipe(genre ~ .,
                                            data = pre_train),
                      resamples = cv,
                      grid = k_grid,
                      control = model_control,
                      metrics = model_metrics)
```

```
## Warning: package 'kknn' was built under R version 4.2.1
```

```r
best <- select_best(knn_tune,"accuracy")
best
```

```
## # A tibble: 1 x 2
##   neighbors .config
##       <int> <chr>
## 1        79 Preprocessor1_Model16
```

```r
final <- finalize_model(knn_spec,best)
final
```

```
## K-Nearest Neighbor Model Specification (classification)
##
## Main Arguments:
##   neighbors = 79
##
## Computational engine: kknn
```

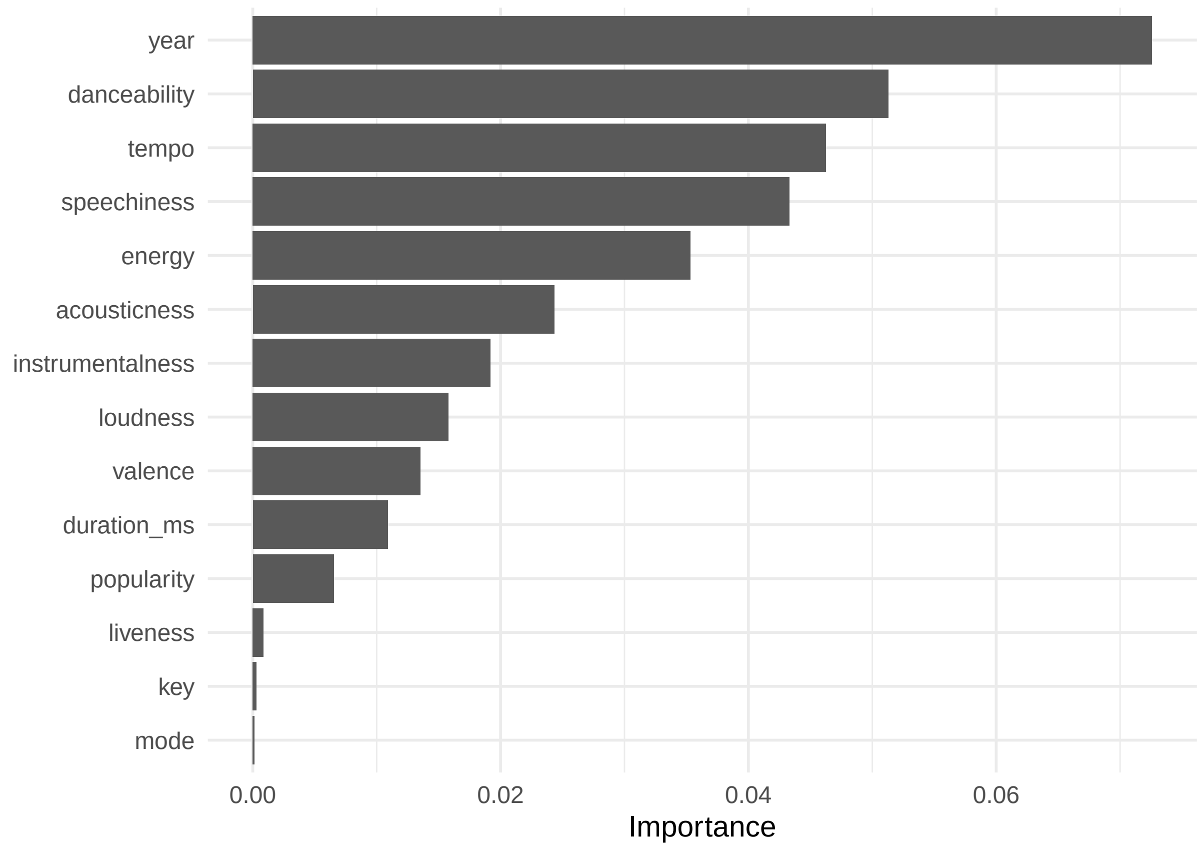
```r
knn <- final %>% fit(genre ~ .,data=pre_train)

# Random forest
set.seed( 1223 )
rf_cv <- vfold_cv( train, v = 5 )
rf_spec_tune <- rand_forest( mode = "classification",
                             trees = 100,
                             mtry = tune(),
                             min_n = tune() ) %>%
  set_engine( "ranger", importance = "permutation" )
rf_spec_tune
```

```
## Random Forest Model Specification (classification)
```

```
##
## Main Arguments:
##   mtry = tune()
##   trees = 100
##   min_n = tune()
##
## Engine-Specific Arguments:
##   importance = permutation
##
## Computational engine: ranger
```

```
params_grid <- grid_regular( finalize( mtry(), train %>% dplyr::select( -genre ) ),
                             min_n(),
                             levels = 5)
rf_tuned <- tune_grid( object = rf_spec_tune,
                       preprocessor = recipe(genre ~ . , data = train),
                       resamples = rf_cv,
                       grid = params_grid )
best_auc <- select_best( rf_tuned, "roc_auc" )
final_rf <- finalize_model( rf_spec_tune, best_auc )
rf_model <- final_rf %>%
  fit( genre ~ . , data = train )
rf_model %>%
  vip(num_features = 14) +
  theme_minimal()
```



Evaluate on the test data

```
# LDA
cc_preds <- predict( spoty_lda, new_data = test ) %>%
  bind_cols( test %>% dplyr::select( genre ) )
cc_preds %>%
  head()
```

```
## # A tibble: 6 x 2
##   .pred_class genre
##   <fct>       <fct>
## 1 edm         pop
## 2 r&b         pop
## 3 edm         pop
## 4 latin       pop
## 5 latin       pop
## 6 edm         pop
```

```
cc_preds %>% conf_mat(truth=genre,estimate=.pred_class)
```

```
##            Truth
## Prediction edm latin pop r&b rap rock
##      edm   160    31  35  10  33   26
##      latin  28   108  55  45  39   13
##      pop    34    36  88  29  22   33
##      r&b     6    32  30  82  34   25
##      rap    22    32  21  56 119    6
##      rock    0    11  21  28   3  147
```

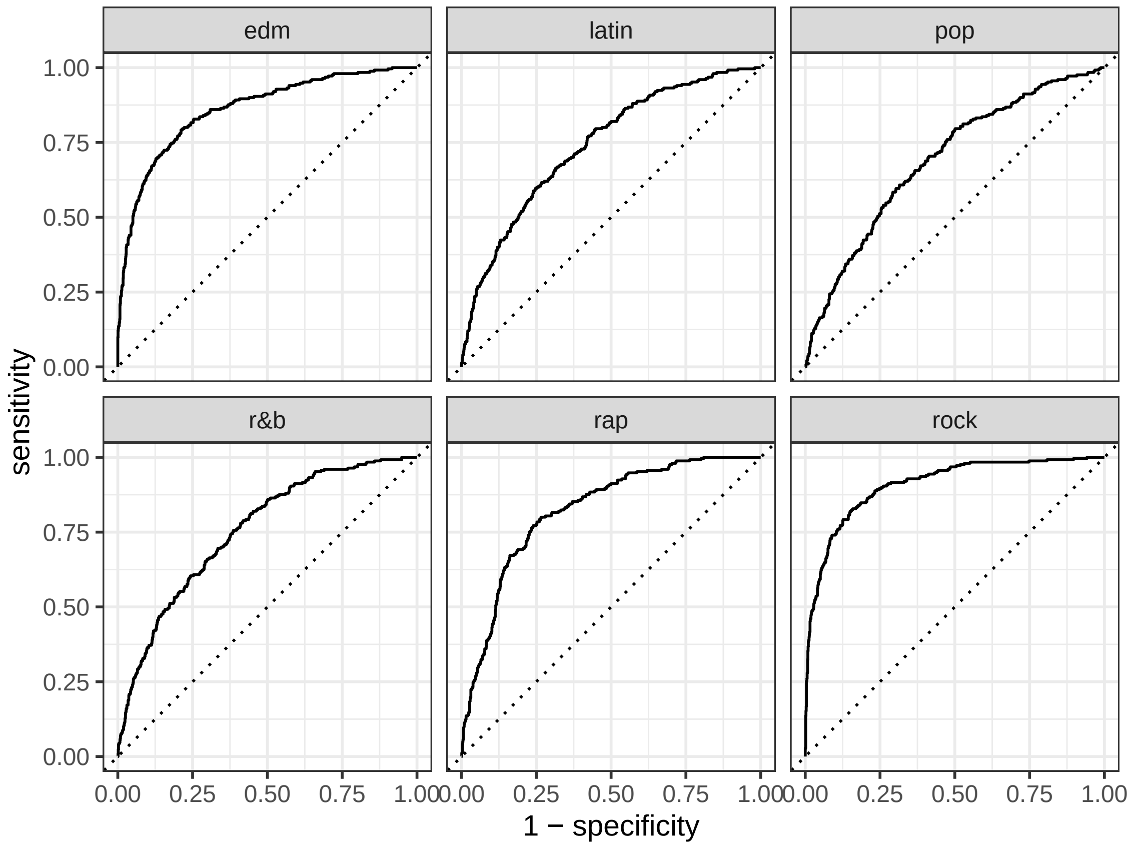
```
lda_sens_spec <- cc_preds %>%
  sens( truth = genre, estimate = .pred_class ) %>%
  bind_rows( cc_preds %>%
             spec( truth = genre, estimate = .pred_class ) )
lda_sens_spec
```

```
## # A tibble: 2 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 sens    macro          0.469
## 2 spec    macro          0.894
```

```
pp_preds <- predict(spoty_lda,test,type="prob") %>% bind_cols(test %>% dplyr::select(genre))
lda_auc <- pp_preds %>% roc_auc(genre, .pred_edm:.pred_rock ,estimator ="macro_weighted")
lda_auc
```

```
## # A tibble: 1 x 3
##   .metric .estimator     .estimate
##   <chr>   <chr>              <dbl>
## 1 roc_auc macro_weighted     0.796
```

```r
pp_preds %>% roc_curve(truth = genre, estimate = .pred_edm:.pred_rock)%>%
  autoplot()
```



```r
# KNN
class_preds <- predict(knn,pre_test,type="class") %>% bind_cols(pre_test %>% dplyr::select(genre))
class_preds %>%
  head()
```

```
## # A tibble: 6 x 2
##   .pred_class genre
##   <fct>       <fct>
## 1 latin       pop
## 2 r&b         pop
## 3 latin       pop
## 4 latin       pop
## 5 latin       pop
## 6 pop         pop
```

```r
class_preds %>% conf_mat(truth=genre,estimate=.pred_class)
```

```
##           Truth
## Prediction edm latin pop r&b rap rock
##      edm   173    31  45  13  24   28
##      latin  24   104  50  47  42    8
##      pop    34    56  91  48  27   27
##      r&b     7    29  23  85  22   31
##      rap    12    22  24  48 132    8
##      rock    0     8  17   9   3  148
```

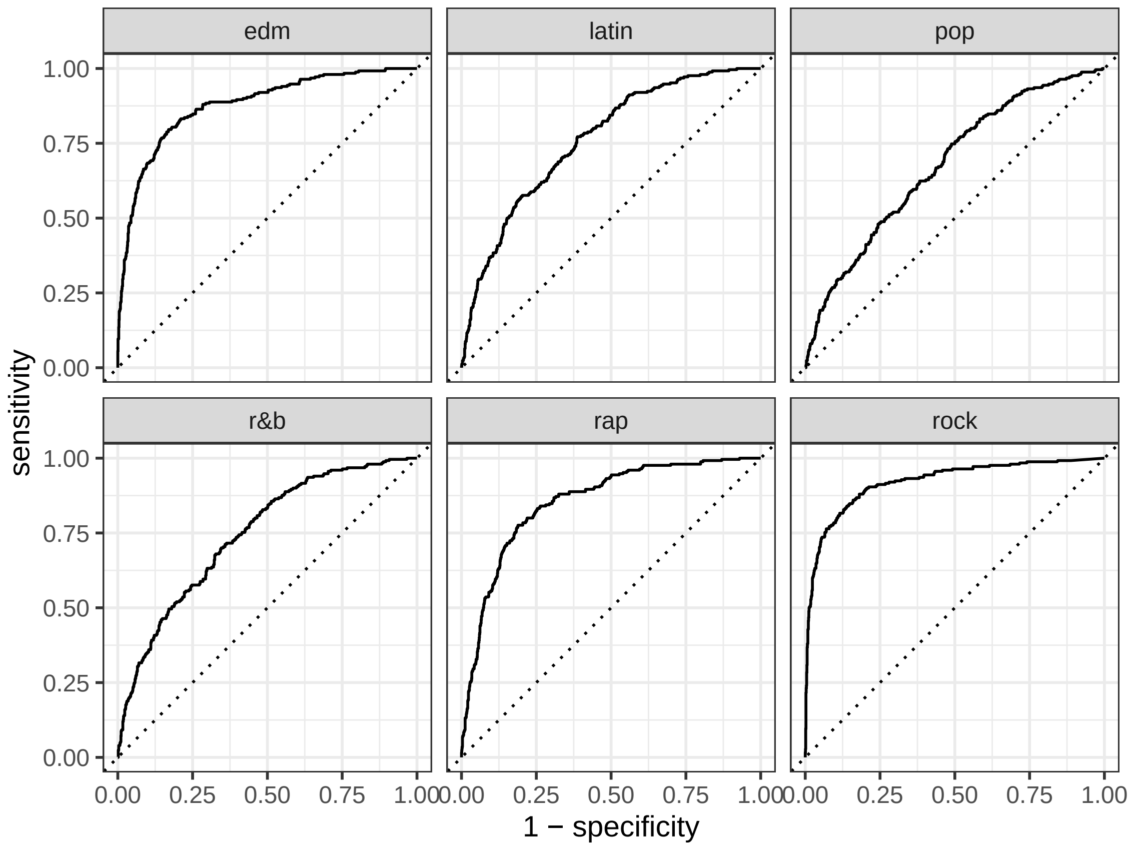
```
knn_sens_spec <- class_preds %>%
  sens( truth = genre, estimate = .pred_class ) %>%
  bind_rows( class_preds %>%
               spec( truth = genre, estimate = .pred_class ) )
knn_sens_spec
```

```
## # A tibble: 2 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 sens    macro          0.489
## 2 spec    macro          0.898
```

```
prob_preds <- predict(knn,pre_test,type="prob") %>% bind_cols(pre_test %>% dplyr::select(genre))
knn_auc <- prob_preds %>% roc_auc(genre, .pred_edm:.pred_rock ,estimator ="macro_weighted")
knn_auc
```

```
## # A tibble: 1 x 3
##   .metric .estimator     .estimate
##   <chr>   <chr>              <dbl>
## 1 roc_auc macro_weighted     0.806
```

```
prob_preds %>% roc_curve(genre, .pred_edm:.pred_rock) %>%
  autoplot()
```



```r
# Random forest
c_preds <- predict(rf_model,test,type="class") %>% bind_cols(test %>% dplyr::select(genre))
c_preds %>%
  head()
```

```
## # A tibble: 6 x 2
##   .pred_class genre
##   <fct>       <fct>
## 1 latin       pop
## 2 r&b         pop
## 3 edm         pop
## 4 latin       pop
## 5 rap         pop
## 6 latin       pop
```

```r
c_preds %>% conf_mat(truth=genre,estimate=.pred_class)
```

```
##           Truth
## Prediction edm latin pop r&b rap rock
##      edm   169    23  37   7  12   11
##      latin  18   104  37  40  18    3
##      pop    33    48  85  34  19   10
##      r&b     5    31  36 104  27   26
##      rap    18    38  29  54 166    4
##      rock    7     6  26  11   8  196
```

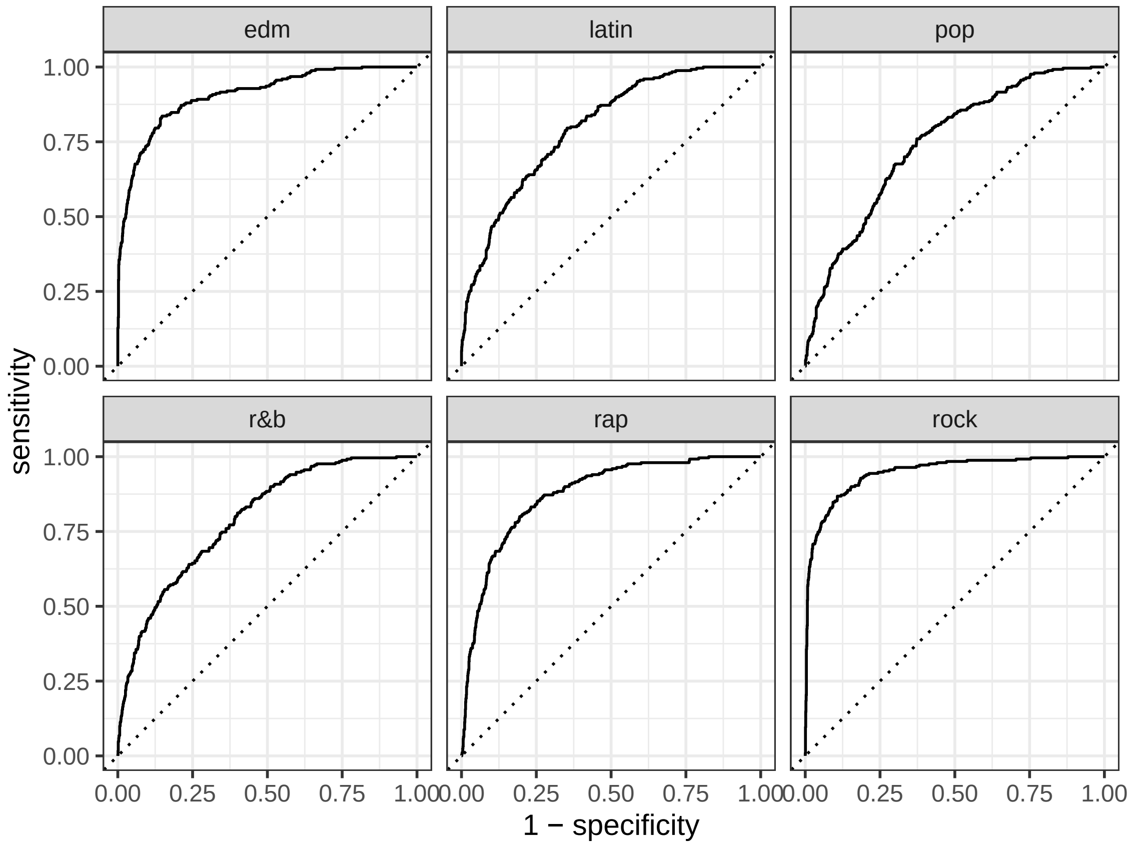
```r
rf_sens_spec <- c_preds %>%
  sens( truth = genre, estimate = .pred_class ) %>%
  bind_rows( c_preds %>%
               spec( truth = genre, estimate = .pred_class ) )
p_preds <- predict(rf_model,test,type="prob") %>% bind_cols(test %>% dplyr::select(genre))
rf_sens_spec
```

```
## # A tibble: 2 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 sens    macro          0.549
## 2 spec    macro          0.910
```

```r
rf_auc <- p_preds %>% roc_auc(genre, .pred_edm:.pred_rock)
rf_auc
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc hand_till      0.844
```

```r
p_preds %>%
  roc_curve(truth = genre, estimate = .pred_edm:.pred_rock) %>%
  autoplot()
```



Model Comparison

```r
# Created a table for model comparison.
matrix_data <- as.numeric(c(lda_sens_spec[1,3],lda_sens_spec[2,3],lda_auc[1,3],knn_sens_spec[1,3],knn_s
matrix_data <- round(matrix_data,3)
comparison_table <- matrix(matrix_data,ncol=3,byrow=TRUE)
colnames(comparison_table) <- c("Sensitivity","Specificity","AUC Score")
rownames(comparison_table) <- c("Linear Discriminant Analysis","K-nearest","Random Forest")
comparison_table <- as.table(comparison_table)
comparison_table
```

```
##                              Sensitivity Specificity AUC Score
## Linear Discriminant Analysis       0.469       0.894     0.796
## K-nearest                          0.489       0.898     0.806
## Random Forest                      0.549       0.910     0.844
```