

spotify-genre-report

Jo Leung

06/01/2021

Introduction

We are tasked with an individual project to demonstrate our ability to apply machine learning techniques on publicly available datasets.

I've chosen to use the "Dataset of songs in Spotify" available from Kaggle which consists of audio features of songs provided by Spotify and the task is to use these features to predict the genre of the song.

Here's a link to the dataset:

<https://www.kaggle.com/mrmorj/dataset-of-songs-in-spotify>

We will be attempting different machine learning algorithms to achieve this.

Analysis

We first examine the dataset and perform data wrangling and data visualization on the dataset to gain more insight to the features available to us and prepare the data for machine learning.

Data Wrangling

Import the dataset into an object "spotify" from the excel sheet to RStudio, the file is saved in the same working directory as the script so relative path is used.

```
spotify <- read.csv("genres_v2.csv")
```

We then examine the dataset with the head and summary functions:

```
##  danceability energy key loudness mode speechiness acousticness
## 1      0.831  0.814   2   -7.364   1      0.4200      0.0598
## 2      0.719  0.493   8   -7.230   1      0.0794      0.4010
## 3      0.850  0.893   5   -4.783   1      0.0623      0.0138
## 4      0.476  0.781   0   -4.710   1      0.1030      0.0237
## 5      0.798  0.624   2   -7.668   1      0.2930      0.2170
## 6      0.721  0.568   0  -11.295   1      0.4140      0.0452
##  instrumentalness liveness valence tempo type
## 1      1.34e-02  0.0556  0.3890 156.985 audio_features
## 2      0.00e+00  0.1180  0.1240 115.080 audio_features
## 3      4.14e-06  0.3720  0.0391 218.050 audio_features
## 4      0.00e+00  0.1140  0.1750 186.948 audio_features
## 5      0.00e+00  0.1660  0.5910 147.988 audio_features
```

```

## 6          2.12e-01    0.1280  0.1090 144.915 audio_features
##              id                      uri
## 1 2Vc6NJ9PW9gD9q343XFRKx  spotify:track:2Vc6NJ9PW9gD9q343XFRKx
## 2 7pgJBLVz5VmnL7uGHmRj6p  spotify:track:7pgJBLVz5VmnL7uGHmRj6p
## 3 0vSWgAlfpye0WCGeNmuNhy  spotify:track:0vSWgAlfpye0WCGeNmuNhy
## 4 OVSXnJqQkwuH2ei1n0Q1nu  spotify:track:OVSXnJqQkwuH2ei1n0Q1nu
## 5 4jCeguq9rMTlbMmPHu07S3  spotify:track:4jCeguq9rMTlbMmPHu07S3
## 6 6fsypiJHyWmeINsOLC1cos  spotify:track:6fsypiJHyWmeINsOLC1cos
##              track_href
## 1 https://api.spotify.com/v1/tracks/2Vc6NJ9PW9gD9q343XFRKx
## 2 https://api.spotify.com/v1/tracks/7pgJBLVz5VmnL7uGHmRj6p
## 3 https://api.spotify.com/v1/tracks/0vSWgAlfpye0WCGeNmuNhy
## 4 https://api.spotify.com/v1/tracks/OVSXnJqQkwuH2ei1n0Q1nu
## 5 https://api.spotify.com/v1/tracks/4jCeguq9rMTlbMmPHu07S3
## 6 https://api.spotify.com/v1/tracks/6fsypiJHyWmeINsOLC1cos
##              analysis_url  duration_ms
## 1 https://api.spotify.com/v1/audio-analysis/2Vc6NJ9PW9gD9q343XFRKx      124539
## 2 https://api.spotify.com/v1/audio-analysis/7pgJBLVz5VmnL7uGHmRj6p      224427
## 3 https://api.spotify.com/v1/audio-analysis/0vSWgAlfpye0WCGeNmuNhy       98821
## 4 https://api.spotify.com/v1/audio-analysis/OVSXnJqQkwuH2ei1n0Q1nu      123661
## 5 https://api.spotify.com/v1/audio-analysis/4jCeguq9rMTlbMmPHu07S3      123298
## 6 https://api.spotify.com/v1/audio-analysis/6fsypiJHyWmeINsOLC1cos      112511
##   time_signature  genre                      song_name
## 1              4 Dark Trap                Mercury: Retrograde
## 2              4 Dark Trap                Pathology
## 3              4 Dark Trap                Symbiote
## 4              3 Dark Trap ProductOfDrugs (Prod. The Virus and Antidote)
## 5              4 Dark Trap                Venom
## 6              4 Dark Trap                Gatteka
##   Unnamed..0 title
## 1              NA
## 2              NA
## 3              NA
## 4              NA
## 5              NA
## 6              NA

##   danceability      energy      key      loudness
## Min.   :0.0651  Min.   :0.000243  Min.   : 0.00  Min.   : -33.357
## 1st Qu.:0.5240  1st Qu.:0.632000  1st Qu.: 1.00  1st Qu.: -8.161
## Median :0.6460  Median :0.803000  Median : 6.00  Median : -6.234
## Mean   :0.6394  Mean   :0.762516  Mean   : 5.37  Mean   : -6.465
## 3rd Qu.:0.7660  3rd Qu.:0.923000  3rd Qu.: 9.00  3rd Qu.: -4.513
## Max.   :0.9880  Max.   :1.000000  Max.   :11.00  Max.   :  3.148
##
##   mode      speechiness      acousticness      instrumentalness
## Min.   :0.0000  Min.   :0.0227  Min.   :0.0000011  Min.   :0.00000
## 1st Qu.:0.0000  1st Qu.:0.0491  1st Qu.:0.0017300  1st Qu.:0.00000
## Median :1.0000  Median :0.0755  Median :0.0164000  Median :0.00594
## Mean   :0.5495  Mean   :0.1366  Mean   :0.0961605  Mean   :0.28305
## 3rd Qu.:1.0000  3rd Qu.:0.1930  3rd Qu.:0.1070000  3rd Qu.:0.72200
## Max.   :1.0000  Max.   :0.9460  Max.   :0.9880000  Max.   :0.98900
##
##   liveness      valence      tempo      type

```

```
## Min. :0.0107 Min. :0.0187 Min. : 57.97 Length:42305
## 1st Qu.:0.0996 1st Qu.:0.1610 1st Qu.:129.93 Class :character
## Median :0.1350 Median :0.3220 Median :144.97 Mode :character
## Mean :0.2141 Mean :0.3571 Mean :147.47
## 3rd Qu.:0.2940 3rd Qu.:0.5220 3rd Qu.:161.46
## Max. :0.9880 Max. :0.9880 Max. :220.29
##
## id uri track_href analysis_url
## Length:42305 Length:42305 Length:42305 Length:42305
## Class :character Class :character Class :character Class :character
## Mode :character Mode :character Mode :character Mode :character
##
##
##
## duration_ms time_signature genre song_name
## Min. : 25600 Min. :1.000 Length:42305 Length:42305
## 1st Qu.:179840 1st Qu.:4.000 Class :character Class :character
## Median :224760 Median :4.000 Mode :character Mode :character
## Mean :250866 Mean :3.973
## 3rd Qu.:301133 3rd Qu.:4.000
## Max. :913052 Max. :5.000
##
## Unnamed..0 title
## Min. : 0 Length:42305
## 1st Qu.: 5256 Class :character
## Median :10480 Mode :character
## Mean :10484
## 3rd Qu.:15709
## Max. :20999
## NA's :21525
```

From these tables, we can see that the type, id, uri, track_href, analysis_url, title, unnamed title columns are not very useful for us, hence we will be removing those columns and use the remaining features to predict the genre.

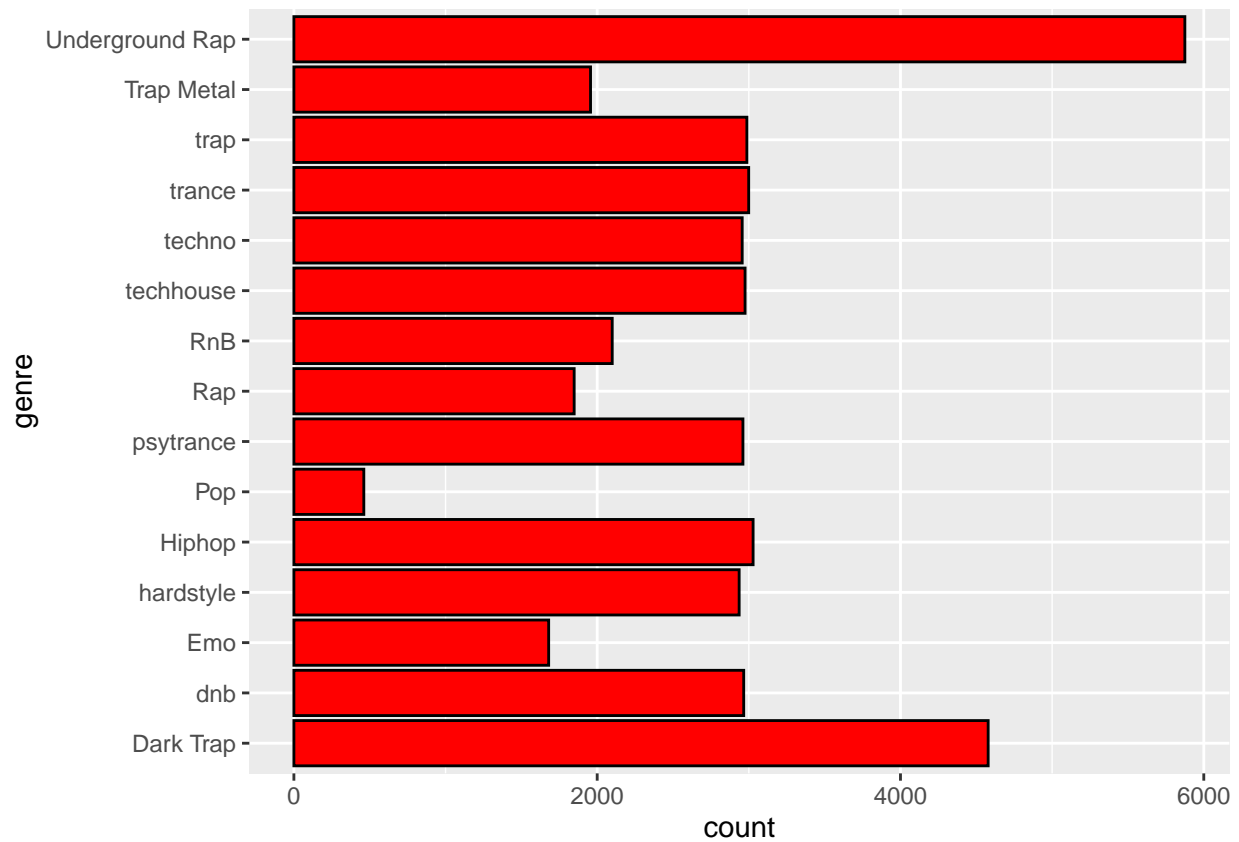
```
spotify <- spotify %>%
  select(danceability, energy, key, loudness, speechiness, acousticness, instrumentalness, liveness, valence)
```

```
## danceability energy key loudness speechiness acousticness instrumentalness
## 1 0.831 0.814 2 -7.364 0.4200 0.0598 1.34e-02
## 2 0.719 0.493 8 -7.230 0.0794 0.4010 0.00e+00
## 3 0.850 0.893 5 -4.783 0.0623 0.0138 4.14e-06
## 4 0.476 0.781 0 -4.710 0.1030 0.0237 0.00e+00
## 5 0.798 0.624 2 -7.668 0.2930 0.2170 0.00e+00
## 6 0.721 0.568 0 -11.295 0.4140 0.0452 2.12e-01
## liveness valence tempo duration_ms time_signature genre
## 1 0.0556 0.3890 156.985 124539 4 Dark Trap
## 2 0.1180 0.1240 115.080 224427 4 Dark Trap
## 3 0.3720 0.0391 218.050 98821 4 Dark Trap
## 4 0.1140 0.1750 186.948 123661 3 Dark Trap
## 5 0.1660 0.5910 147.988 123298 4 Dark Trap
## 6 0.1280 0.1090 144.915 112511 4 Dark Trap
```

Data visualization

We now use data visualization techniques to explore the dataset visually. First we look at the genre distribution,

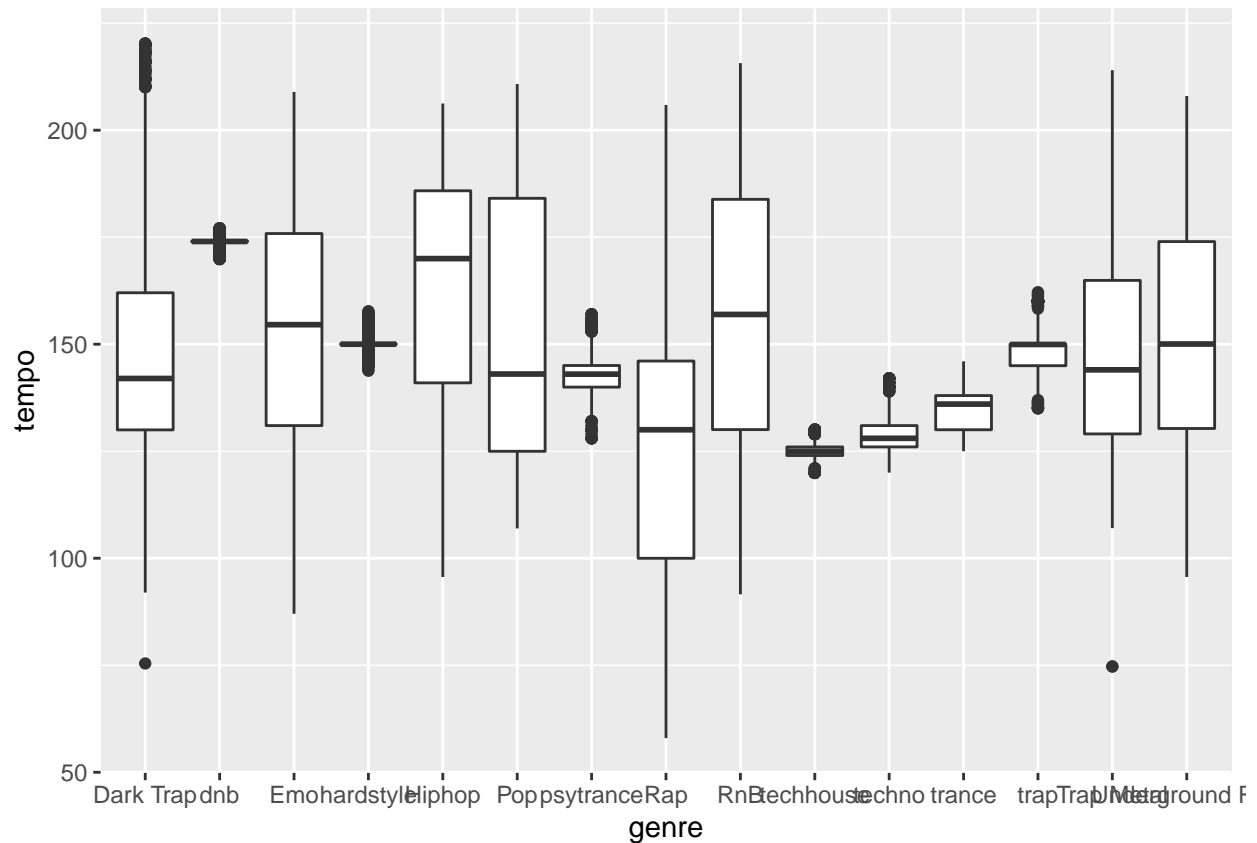
```
#investigate genre distribution
spotify %>%
  group_by(genre) %>%
  ggplot(aes(genre)) +
  geom_bar(fill = "red", color = "black") +
  coord_flip()
```



We can see a list of the genres that are present in the dataset, it is worth noting that the genres in the dataset are not evenly distributed with a lot of the songs belonging to the Underground Rap genre and very few belong to the Pop genre.

We then examine the tempo distribution and average tempo by genre

```
#investigate tempo distribution by genre
spotify %>%
  group_by(genre) %>%
  ggplot(aes(genre, tempo)) +
  geom_boxplot()
```



```
#investigate the average tempo by genre
spotify %>%
  group_by(genre) %>%
  summarise(mean_tempo = mean(tempo)) %>%
  arrange(desc(mean_tempo))
```

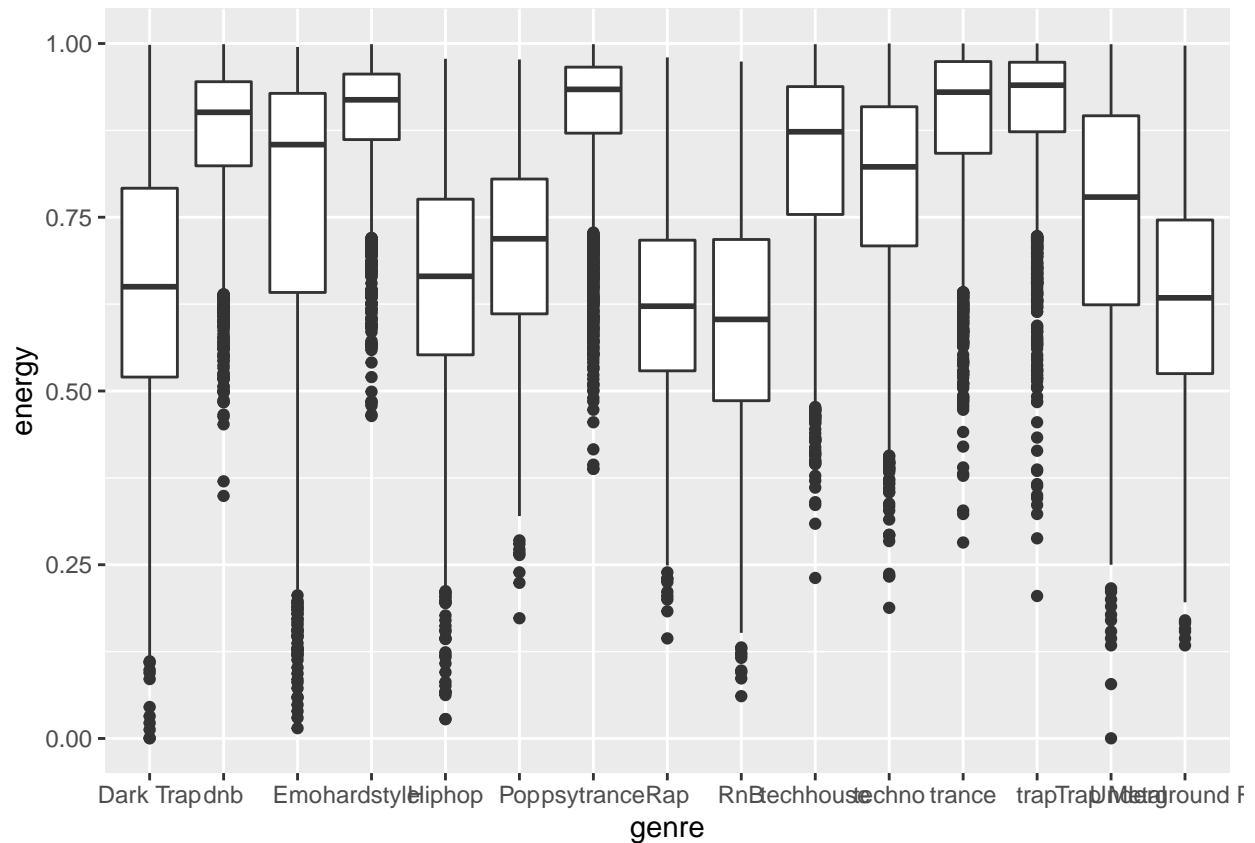
```
## # A tibble: 15 x 2
##   genre                mean_tempo
##   <chr>                <dbl>
## 1 dnb                  174.
## 2 Hiphop               163.
## 3 RnB                  158.
## 4 Emo                 154.
## 5 Underground Rap     153.
## 6 Pop                  152.
## 7 hardstyle           151.
## 8 Dark Trap           150.
## 9 Trap Metal          149.
## 10 trap                148.
## 11 psytrance           143.
## 12 trance              135.
## 13 techno              129.
## 14 Rap                 126.
## 15 techhouse           125.
```

The tempo/mean tempo distribution provide an interesting insight as some genres (techhouse) have very

small inter quartile ranges whilst genres like pop has large inter quartile ranges. Furthermore, there is quite a range when exploring the mean tempo, varying from 125.techhouse to 174.dnb which can be a good predictor for genres.

We then examine the energy distribution and average energy by genre

```
#investigate energy distribution by genre
spotify %>%
  group_by(genre) %>%
  ggplot(aes(genre, energy)) +
  geom_boxplot()
```



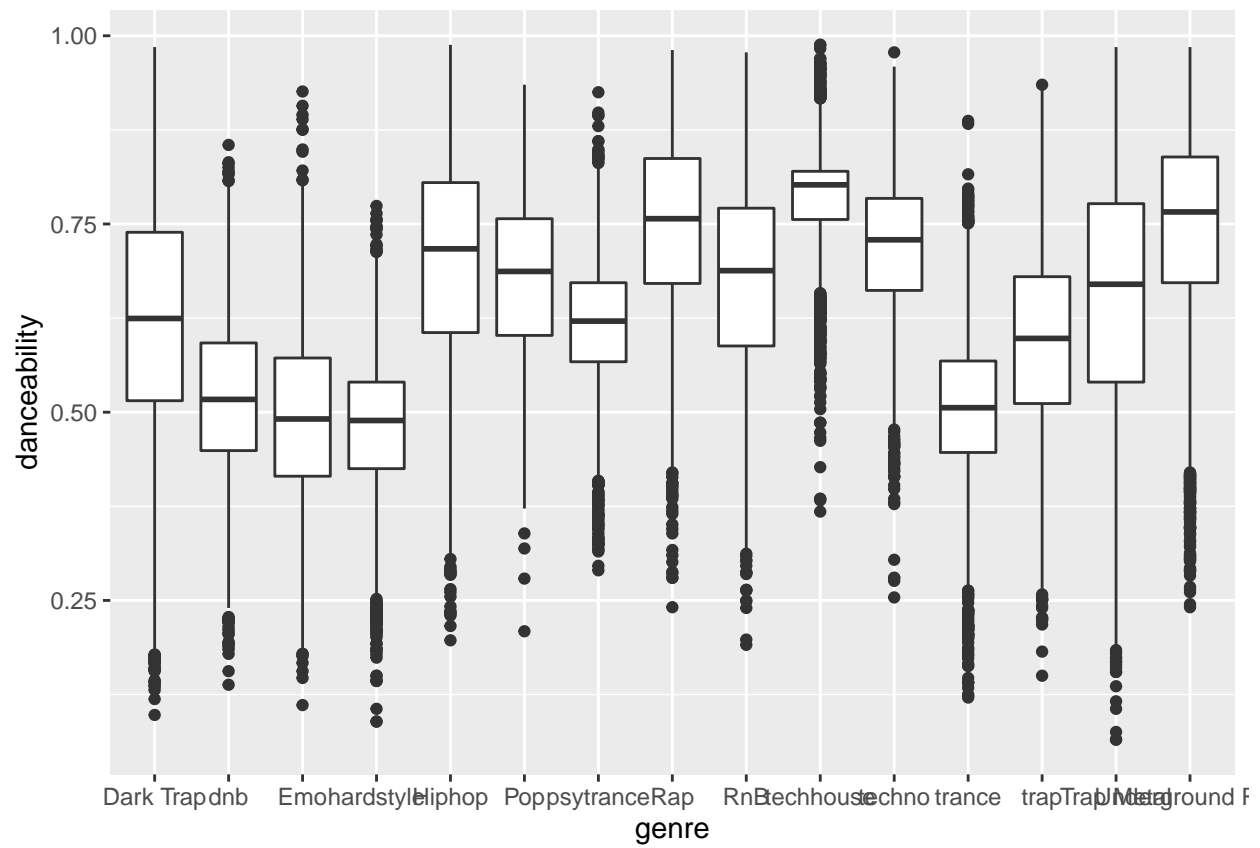
```
#investigate the average energy level by genre
spotify %>%
  group_by(genre) %>%
  summarise(mean_energy = mean(energy)) %>%
  arrange(desc(mean_energy))
```

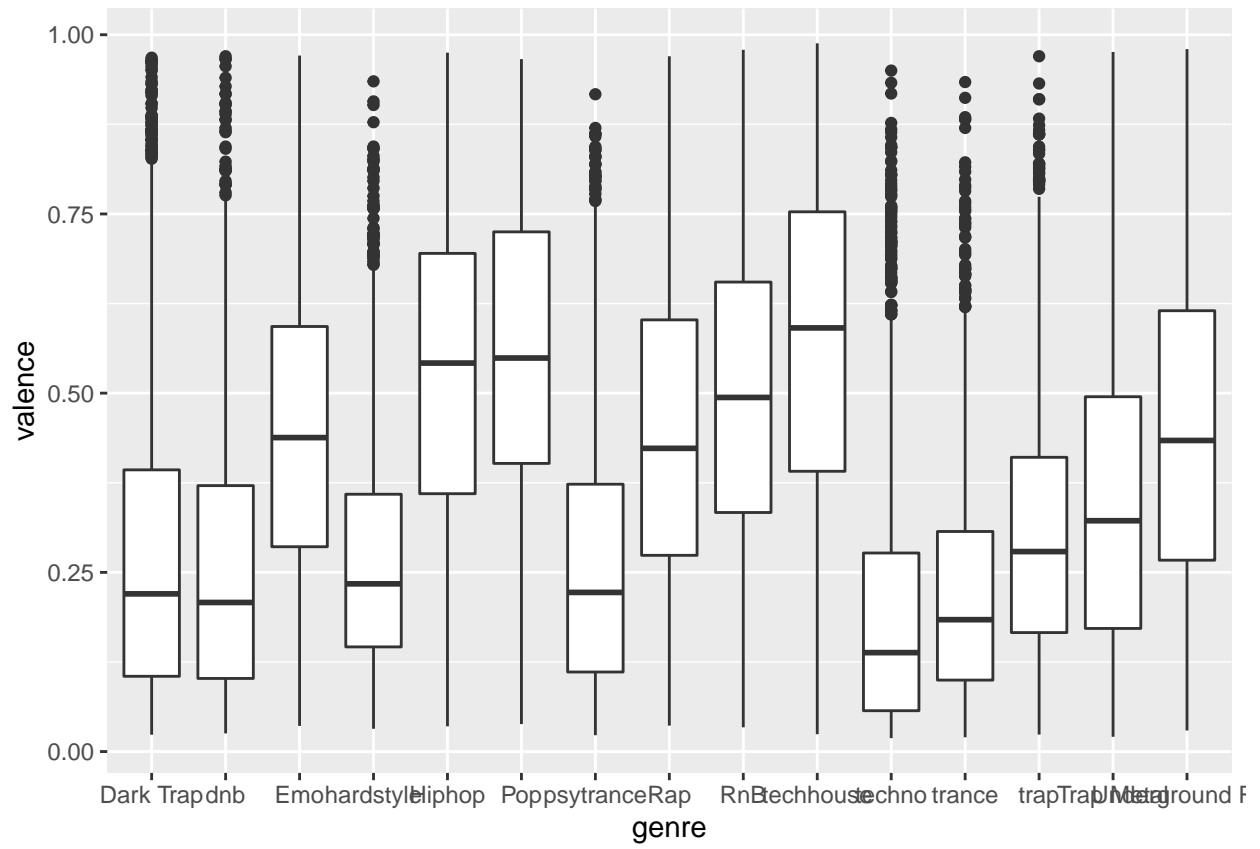
```
## # A tibble: 15 x 2
##   genre      mean_energy
##   <chr>      <dbl>
## 1 trap      0.906
## 2 psytrance 0.902
## 3 hardstyle 0.896
## 4 trance    0.892
## 5 dnb       0.873
```

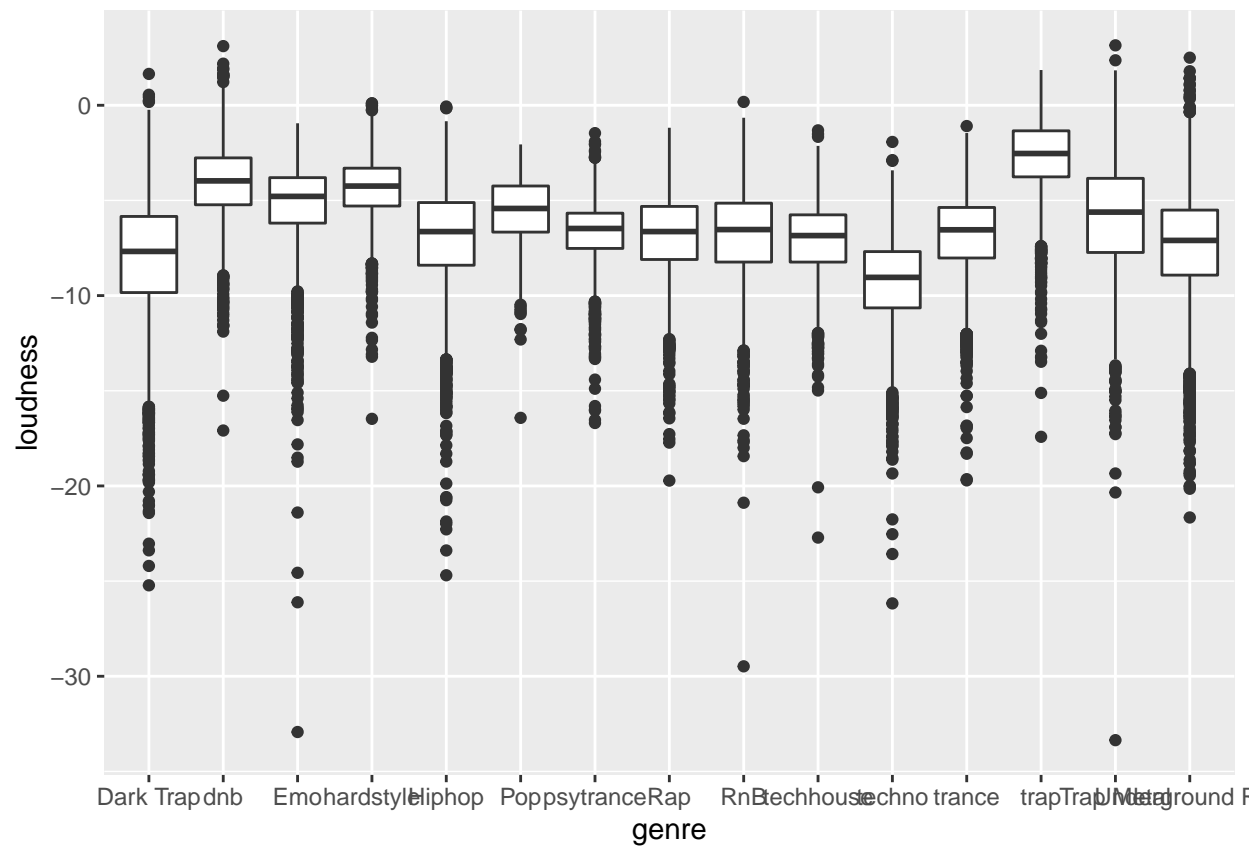
##	6	techhouse	0.834
##	7	techno	0.796
##	8	Emo	0.761
##	9	Trap Metal	0.749
##	10	Pop	0.698
##	11	Hiphop	0.654
##	12	Dark Trap	0.647
##	13	Underground Rap	0.636
##	14	Rap	0.620
##	15	RnB	0.599

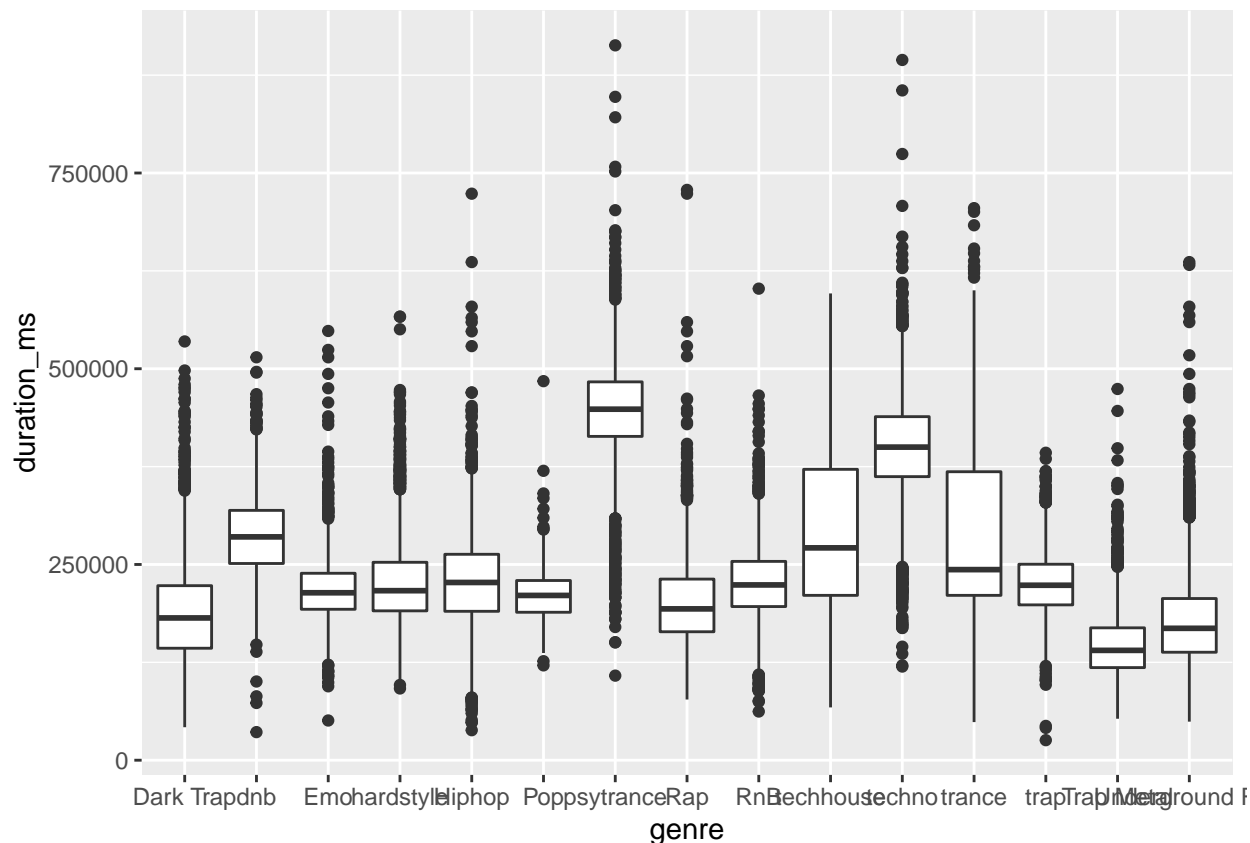
The energy level distribution is more condense as all genres have an average above 0.5.

We also look at danceability, valence, loudness and duration distributions by genre:









From the plots, we can see that danceability, valence are more distributed between genres than loudness and duration, suggesting they are stronger predictors for analysis later.

Separating training, test, and validation dataset

We then move on to creating training, test and validation dataset from the spotify dataset. The validation dataset is strictly used for evaluation only and was not used for training in any way. When working on different algorithms, training dataset is used to train our algorithm and tested on the test dataset.

The validation dataset consists of 20% of the spotify dataset, the remaining 80% is distributed as follows: training 80%, testing 20%.

As the original spotify dataset consists of 42305 observations (songs) it allows for 20% going to the validation dataset as we have enough observations for training. I avoided splitting it into 90%:10% as that will increase computational cost and time.

During the training process, the algorithms are evaluated on the test set.

Later in the report, the final evaluation will be performed on the validation dataset, which was not used in the training process in any way.

```
set.seed(7, sample.kind = "Rounding")
```

```
## Warning in set.seed(7, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```

verify_index <- createDataPartition(spotify$genre, times = 1, p = 0.2, list = FALSE)
analysis <- spotify[-verify_index,]
temp <- spotify[verify_index,]

validation <- temp %>%
  semi_join(analysis, by = "genre")

removed <- anti_join(temp, validation)

```

```
## Joining, by = c("danceability", "energy", "key", "loudness", "speechiness", "acousticness", "instrumentalness")
```

```

analysis <- rbind(analysis, removed)

set.seed(7, sample.kind = "Rounding")

```

```

## Warning in set.seed(7, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used

```

```

test_index <- createDataPartition(analysis$genre, times = 1, p = 0.2, list = FALSE)
training_set <- analysis[-test_index,]
test_set <- analysis[test_index,]

rm(removed, temp, verify_index, analysis, test_index)

```

Machine learning

We will be using a range of machine learning algorithms to predict the genre of the song using the audio features available to us.

The machine learning algorithms that we will use are:

Linear discriminant analysis (LDA) Quadratic discriminant analysis (QDA) k-nearest neighbour algorithm (kNN) Random forest (rf)

Model 1 | LDA model We first try a LDA model:

```

train_lda <- train(genre ~ ., data = training_set, method = "lda")
s_lda <- predict(train_lda, validation)
mean(validation$genre == s_lda)

```

```
## [1] 0.5174776
```

Model 2 | QDA model We then try a qda model:

```
train_qda <- train(genre ~ ., data = training_set, method = "qda")
```

```
## Warning: model fit failed for Resample01: parameter=none Error in qda.default(x, grouping, ...) : ran
```

```

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.

```

```
s_qda <- predict(train_qda, validation)
mean(validation$genre == s_qda)
```

```
## [1] 0.6150213
```

Model 3 | Random forest model 1 We now attempt a random forest model with 100 tree nodes:

```
train_rf <- train(genre ~ ., data = training_set, method = "rf",
                  tuneGrid = data.frame(mtry = seq(1:7)),
                  ntree = 100)
```

```
train_rf$bestTune
```

```
##      mtry
## 4         4
```

```
s_rf <- predict(train_rf, validation)
mean(s_rf == validation$genre)
```

```
## [1] 0.6634388
```

We can also observe the variable importance which is why “rf” is used over “rborist”.

```
varImp(train_rf)
```

```
## rf variable importance
##
##               Overall
## tempo           100.000
## duration_ms     54.237
## danceability    40.525
## instrumentalness 39.171
## loudness        33.652
## energy          32.641
## valence         28.692
## speechiness     27.340
## acousticness    25.490
## liveness        19.539
## key             9.829
## time_signature  0.000
```

As shown, tempo is the most important variable to predict genre, followed by duration and danceability. It seems time signature has no impact at predicting the genre of the song.

Model 4 | Random forest model 2 We now attempt a random forest model with 500 tree nodes to see if it improves the accuracy:

```
train_rf2 <- train(genre ~ ., data = training_set, method = "rf",
                  tuneGrid = data.frame(mtry = seq(1:7)),
                  ntree = 500)
```

```
train_rf2$bestTune
```

```
## mtry
## 4 4
```

```
s_rf2 <- predict(train_rf2, validation)
```

```
mean(s_rf2 == validation$genre)
```

```
## [1] 0.6682806
```

again we observe the variable importance.

```
varImp(train_rf2)
```

```
## rf variable importance
##
## Overall
## tempo 100.000
## duration_ms 54.212
## danceability 41.160
## instrumentalness 38.669
## loudness 33.920
## energy 32.648
## valence 28.609
## speechiness 27.345
## acousticness 25.516
## liveness 19.300
## key 9.898
## time_signature 0.000
```

The variable importance is the same as our previous random forest model.

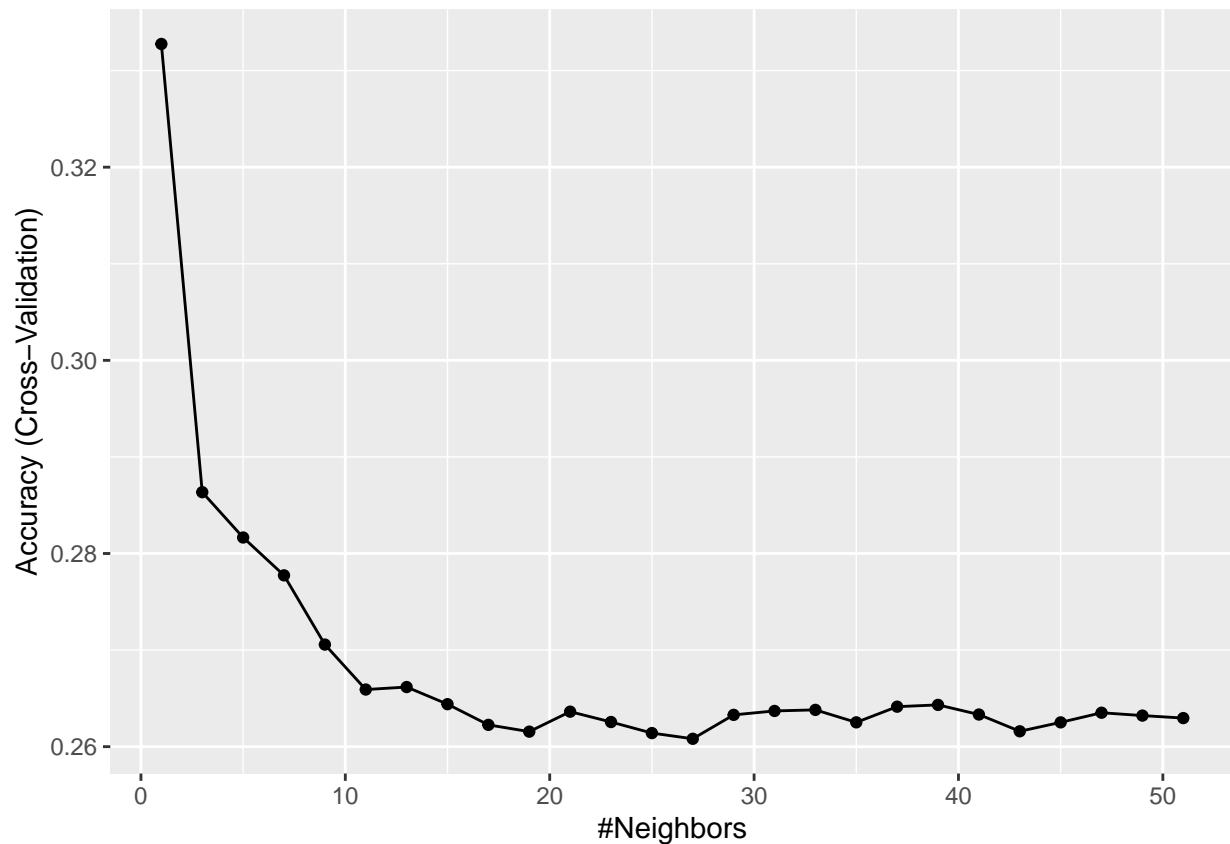
Model 5 | KNN model - All predictors We then apply the KNN model using all the predictors available to us and optimise for number of neighbours:

```
train_knn <- train(genre ~ ., data = training_set, method = "knn",
                  tuneGrid = data.frame(k = seq(1, 51, 2)),
                  trControl = trainControl(method = "cv", number=10, p=0.9))
```

```
train_knn$bestTune
```

```
## k
## 1 1
```

```
ggplot(train_knn)
```



```
s_knn <- predict(train_knn, validation)
```

```
mean(s_knn == validation$genre)
```

```
## [1] 0.3436467
```

Interestingly, this variation of kNN model performed very poorly, worst than both LDA and QDA, a likely cause can be due to the nature of the predictors, we will attempt to use only a subset of the predictors, in particular the “important” variables we obtained from the random forest model earlier which are:

tempo + duration + danceability + instrumentalness + loudness

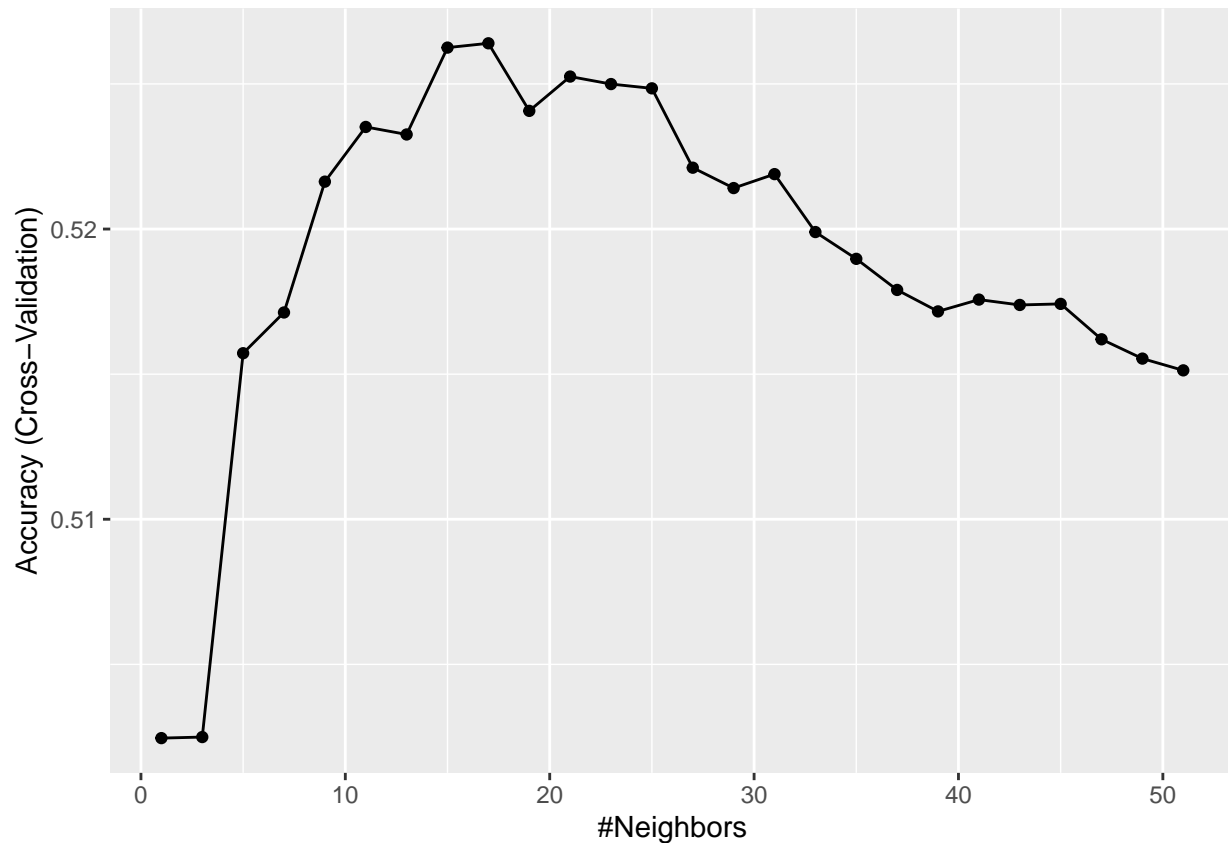
Model 6 | kNN model - important variable An interesting observation, when duration is used as a predictor, the accuracy is still as low as 0.3391908, but when duration is removed as a predictor, we achieved a better accuracy:

```
train_knn2 <- train(genre ~ tempo + danceability + instrumentalness + loudness, data = training_set, method = "knn",
  tuneGrid = data.frame(k = seq(1, 51, 2)),
  trControl = trainControl(method = "cv", number=10, p=0.9))
```

```
train_knn2$bestTune
```

```
## k
## 9 17
```

```
ggplot(train_knn2)
```



```
s_knn2 <- predict(train_knn2, validation)
```

```
mean(s_knn2 == validation$genre)
```

```
## [1] 0.53094
```

Results

Here are our final results applying the trained algorithms on the validation set:

```
## # A tibble: 6 x 2
##   Method                      Accuracy
##   <chr>                      <dbl>
## 1 Random forest model (500 nodes) 0.668
## 2 Random forest model (100 nodes) 0.663
## 3 QDA model                      0.615
## 4 kNN model - important variable 0.531
## 5 LDA model                      0.517
## 6 kNN model - all predictors      0.344
```

The best performing algorithm is the random forest models, with 0.668 and 0.663 accuracy respectively for 100 and 500 nodes.

The kNN model have relatively low accuracy which is unexpected as the QDA model outperforms it by a margin.

Conclusion

We have attempted to use different machine learning algorithms to predict the genre of songs from a Spotify dataset, using audio features available to us as predictors.

There are some interesting observations, such as the negative impact of “duration” as a predictor on a kNN model, and tempo being the best predictor for genre according to the variable importance from random forest model.

One of the reason why the accuracy across different models are lower than expected is because the genre distribution are not evenly distributed and a lot of the genres in the dataset have overlapping similarities, as demonstrated from the data visualization earlier with a lot of features overlapping in ranges, with very few distinct features to separate the songs.