Using Audio Characteristics for Song Prediction

ISYE 7406

Group 9:
Kerstin Fontus (GT ID: 100, kfontus3@gatech.edu)
Nirupa Manohar (GT ID: 006, nmanohar7@gatech.edu)
Niral Patel (GT ID: 343, npatel452@gatech.edu)

April 14, 2024

Abstract

This report investigates the possibility of assigning a genre and a popularity prediction to a song using its audio characteristics as explanatory variables. After data cleaning and tuning the parameters of the models, it was found that the classification methods studied were highly inaccurate while the popularity prediction methods were promising. Cross validation and statistical testing showed that a Random Forest classifier performed the best, but not well enough for production level output. A LASSO model was similarly shown to be the best performing in regards to predicting song popularity. Music platforms could possibly benefit from the popularity prediction when deciding what music to promote and recommend to its users.

Introduction

Spotify is a music streaming platform that allows artists to upload their music for the public's enjoyment. Their services include song recommendation algorithms to help users discover new music they may enjoy based on their listening habits and on new additions from popular and rising artists. A large music platform wants to maximize their profitability by staying on top of trends, but by letting artists set their own genre and favoring established popular artists, there isn't as much market fairness. Unknown and niche artists are less likely to be suggested to users despite the fact that their music may be similar to established artists. A song that is misclassified into a more widely listened to or generalized genre, such as pop, may be recommended differently than if it were assigned its true genre. This analysis investigates the possibility of automatically assigning a song's genre and predicting its future popularity based on the audio characteristics of the song to prevent this type of bias when creating recommendations.

This analysis leverages a Spotify dataset composed of tracks and their different audio characteristics in order to investigate two primary focus areas: genre classification and popularity prediction.

We aim to answer the following questions:

- Do audio characteristics allow us to differentiate between music genres?
- What classification model gives the most accurate predictions?
- Can track popularity be predicted with the given audio characteristics?
- What model provides the most accurate prediction?
- Is popularity prediction more accurate within each genre or over the whole data set?

The classification focus area explores whether audio characteristics allow us to assign a song to one of six genres: Pop, Rock, Rap, Latin, R&B, and EDM. The classification models that will be implemented to answer the questions above are K-Nearest-Neighbor, Discriminant Analysis, Naive Bayes, Random Forest, and Multinomial Logistic Regression.

The prediction focus area delves into predicting song popularity on a scale of 0 to 100 based on audio characteristics. The predictive models investigated to answer the question above are LASSO regression, Linear Regression, Stepwise Regression, Principal Component Analysis, and Random Forest.

The data was first investigated graphically to see if it needed to be transformed and to explore the relationships between different variables. The methodology section discusses the models mentioned above and how their performance is measured. This leads into the initial results and cross validated results for each model, followed by conclusions on the viability of these methods in this context.

Exploratory Data Analysis

Data

The data for this analysis was obtained from Kaggle and derived using Spotify API. The original data set contained 30,000 observations, or songs, and 23 variables. The variables can be split into 3 categories: identification, dependent, and independent. The identification variables consisted of identifying qualities of the song tracks such as track name, artist, album ID and playlist name. These variables were not deemed pertinent to the analysis. The independent variables used for analysis are:

1. **Danceability** - how suitable a song is for dancing (0-1), 1 being the most danceable
2. **Energy** - the intensity and activity of a song (0-1), 1 being the highest energy
3. **Key** - the key of the song, factor numeric variable, no key detected = -1
4. **Loudness** - overall loudness of a track in decibels (dB), (-60-0)
5. **Mode** - classifies the song as major (1) or minor (0) modality
6. **Speechiness** - the amount of spoken word in the song (0-1), closer to 1 represents more speech in the track
7. **Acousticness** - how acoustic the song is (0-1), closer to 1 means higher confidence of being acoustic
8. **Instrumentalness** - predicts if a track as no vocals (0-1), closer to 1 means bigger likelihood of no vocals
9. **Liveness** - detects presence of an audience in the track recording (0-1), value above 0.8 means strong likelihood that it was recorded live
10. **Valence** - the music positiveness conveyed by the song (0-1), high valence (1) means a more happy or cheerful song, low valence (0) means a more sad or angry song
11. **Tempo** - overall estimated tempo of a track in beats per minute (BPM)
12. **Duration_ms** - song length in milliseconds

The response variables are:

1. **Playlist Genre**
2. **Track Popularity** - Song popularity (0-100) where higher is better

For the classification task, the focus is to classify each song into one of the 6 unique genres. For the prediction task, the focus is to predict the exact popularity value.

Data Cleaning

Before conducting any analysis, the data set underwent cleaning to maximize the accuracy of the results. First, any data points with N/A values were removed. The data set included points where the track and artist name were identical, but contained different album names and track popularity values - most likely due to the track coming from different albums (e.g. compilation albums, singles, remasters, etc.). The duplicate tracks were deleted, keeping the observation with the highest track popularity in order to retain the data point that appeared to be the most listened. Observations with a track popularity of 0 were removed due to overabundance and inaccuracy. The top 90% of popular tracks were then retained to make the popularity distribution more normal and prevent skewed results. After cleaning, the final data set consisted of 21,569 observations. There was an almost equally proportional amount of tracks labeled for each genre, each one comprising between 14% and 21% of the data (⅙ of the data would be 16.67%) (Appendix A).

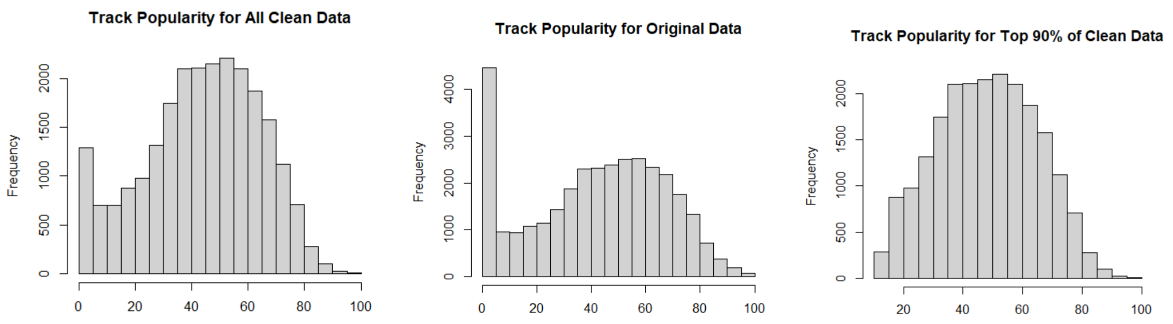


Figure 1: Distribution of popularity during data cleaning

Classification

To investigate the correlation between the independent and response variables, the genres were transformed into one-hot encoded dummy variables and were treated as the response variable.

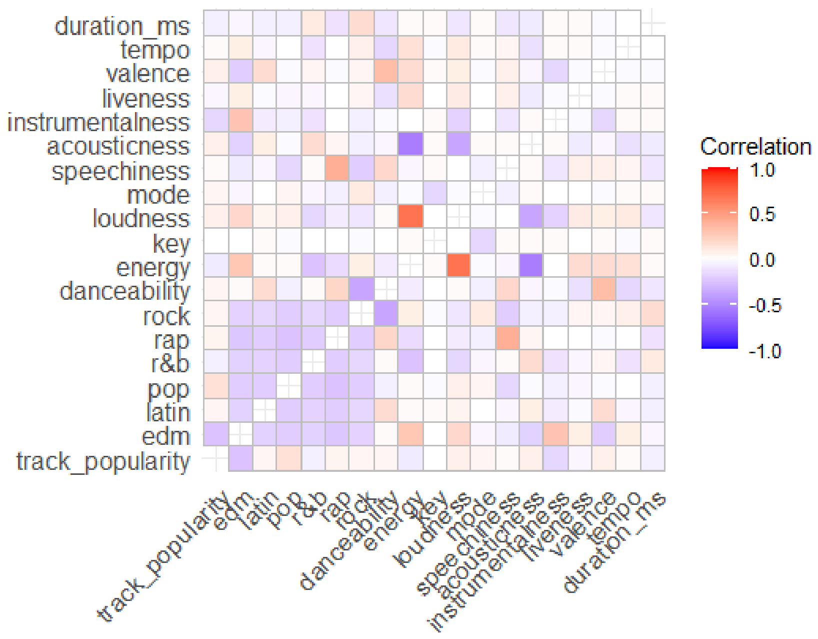


Figure 2: Correlation of predictors to genre

From the plot above, there were few independent variables with strong correlation to a specific genre. Rock songs have lower danceability and longer duration. Rap music mainly consists of spoken words, explaining its higher correlation with speechiness. It also had strong negative relationships with energy and loudness. R&B tracks are less likely to be high energy, loud, or have high tempo, and are more likely to be acoustic. The Pop tracks' strongest correlation was negative in regards to speechiness, with nothing else being of note. Latin tracks also had weaker correlations across the features, with its strongest ones being high in danceability and valence/happiness. EDM tracks had higher energy, loudness, and instrumentalness, which is typical for that genre. As a result, they had a more negative correlation with acousticness and valence/happiness.

Among the independent variables, there are strong associations among loudness, energy, and acousticness, as well as between danceability and valence. Looking at the histograms of some of these stronger variables for each genre shows there is little differentiation to be made (Appendix B). This did not have good implications for the results of the classification attempt. Model building proceeded with this in mind, starting out with all of the independent variables to try to improve final results.

Prediction

The cleaned data set was split into 6 smaller data sets, one for each genre - Pop, Rock, Rap, R&B, Latin, and EDM. The track popularity had a fairly normal distribution for each genre (Appendix C). The correlation between predicting variables and track popularity was calculated and showed a weak correlation across most audio characteristics (Appendix D). Based on this result, it is anticipated that most to all variables will be required in the models. It is also expected that genres with the weakest correlation may present the least accurate predictions. Rock and Latin seemed to show the weakest correlation between audio characteristics and track popularity.

Methodology

Model building and evaluation were done using various R libraries and functions.

Classification

Five methods of multinomial classification were considered in this investigation. The models were evaluated based on the percentage of tracks that were classified incorrectly.

K-Nearest-Neighbors

This method uses a distance based approach, where the response values of the **k** closest data points to the new observation are surveyed, and the mode of these values is assigned as the final response. The algorithms were built with odd values of **k** between 1 and 15 inclusive to prevent the possibility of ties. The *knn()* function used to build the models is found in the *class* library.

Discriminant Analysis

This method involves constructing linear functions of the data points to calculate the strength of evidence that one observation belongs to class *j*. The goal is to maximize this evidence function $d_j(x)$ for each x, or find the strongest evidence that its independent variables point to class *j*. Linear Discriminant Analysis (LDA) uses the Bayes rule under the assumption that these functions are multivariate normal with a common variance across all response classes. Quadratic Discriminant Analysis (QDA) again assumes these functions are normal, but that the variance for each class is the sample covariance of that class. Both models also assume that the data observations are independent. The *lda()* and *qda()* functions used can be found in the *MASS* library.

These R functions do not provide a way to tune the parameters for these methods, so Python's *scikit learn* package was required to make sure the most accurate models possible were being built. Through its GridSearchCSV() function, the best solver for LinearDiscriminantAnalysis() was found to be the Least Squares solver with a shrinkage parameter of 0. For QuadraticDiscriminantAnalysis(), the best regularization parameter was also found to be 0. The error rate of models built with these parameters are identical to that of models with the default parameters, so it was decided to proceed with the default R version of these models for easier recording of results alongside the other methods.

Naive Bayes

This classifier takes the mean and variance for each component for each class, and uses them to build the discriminant functions. The general form of Naive Bayes (NB) ignores any dependence among the explanatory variables. It also assumes normality among the independent variables. The *naiveBayes()* function is found in the *e1071* library.

Random Forest Classification

In a Random Forest Algorithm, a large number of classification trees are built and the mode of the results are chosen as a final prediction. Each tree is built using a random subset of the independent variables. This clears the user of the responsibility of ensuring a tree optimal branching and pruning. Cross validation helped determine the best number of trees to build the forest (700) and the number of variables to use in each tree (2). Due to computational time, the default number of terminal nodes in each tree was used. The *randomForest()* function comes from the aptly named *randomForest* library.

Logistic Regression

Classic Logistic Regression with a Bernoulli distribution assumption is not possible since there are more than two possible classes. The *nnet* library provides a method of multinomial Logistic Regression through Neural Networks using the *multinom()* function. This function calls on the general neural network function *nnet()*, which requires parameter tuning for the most accurate performance. In the end, a neural network with 14 nodes in the hidden layer with a decay value of 0.1 gave the best results.

Variable Selection

Attempting to select the most relevant features only via the exploratory data analysis was not successful as described above. After building a Multinomial Logistic Regression model trained with 70% of randomly selected data points, a stepwise algorithm (using the *step()* function in the *leaps* library) in both directions that result in a model that had a significantly lower AIC value and only missing the key variable in its formula. From this, key was omitted as an independent variable for all future models built.

Prediction

Track popularity was predicted using 6 different methods. All of them operate under the assumption of the data having linearity, constant variance, independence, and normality.

Linear Regression

Linear regression is a supervised method and works by creating a regression model based on a set of data and fitting the new data point to the model. All predicting variables were used in the linear regression model created. The *lm()* function was used to build this model.

LASSO

LASSO regression is a modification of linear regression where the model is penalized to prevent overfitting. It is a regularization technique used to make more accurate predictions via variable selection. From the table, it can be seen that all genres except Rock utilized at least 8 out of the 12 predicting variables in the model, supporting the earlier assumption that many variables would be used in the model due to weak correlation across predictors (Appendix E). The *lars()* function from the *lars* library was used for this regression model.

Stepwise Regression

Stepwise Regression is a form of linear regression which constructs the regression model using a step by step iterative process that chooses to include or omit independent variables based on AIC. The combination of predicting variables which results in the lowest AIC forms the final model. The majority of the variables were used in the models created (Appendix F). The *step()* function was used for this method.

Ridge Regression

Ridge Regression is another regularization technique, however variable selection is not used in this method. Instead, the coefficients in the model are shrunk in proportion to their size using a penalization term in order to equalize the weight of each variable. It is typically used in instances to correct for multicollinearity. As there is no explicit correlation amongst the variables, it is not expected for this model to perform the best. Ridge Regression is not a form of variable selection. *lm.ridge()* from the *MASS* library was used to build this model.

Principal Component Analysis

PCA is a dimensionality reduction technique that transforms the data and uses the transformation to then perform variable selection. The PCA models for all genres selected all variables. The *pcr()* function from the *pls* library was used to perform the PCA analysis.

Random Forest.

Random Forest models in this context are built similarly to the classification context, but the final value is the average of the values predicted by each tree. Cross-validation was also run here to find the best number of trees to build the forest and the optimal number of variables to use in each tree (Appendix G). The *randomForest()* function in the *randomForest* library was used to create these models.

Methods of Evaluation

The classification models were evaluated based on the misclassification error, with the goal of minimizing it. The popularity prediction models were evaluated by calculating the mean squared error compared to the actual values, again with the goal of minimization.

Cross Validation

A 20-Fold Cross Validation algorithm was implemented on each classification model and the classification errors of each fold were compared using paired statistical t-tests. The same procedure was done on the split genre datasets but instead with Monte Carlo Cross Validation using the average results over 50 runs. The error measurements for the cross validated results were compared using paired statistical t-tests, to see if there was a statistically significant difference in how each model performed.

Results

Classification

The classification error rates after one 70-30 train-test split (with proportional amounts of songs from each genre) were as follows for each model:

Testing Classification Error: Single Run													
Model	KNN, k=1	KNN, k=3	KNN, k=5	KNN, k=7	KNN, k=9	KNN, k=11	KNN, k=13	KNN, k=15	LDA	QDA	NB	RF	LR
Error	0.6917	0.6822	0.6732	0.6668	0.6688	0.6701	0.6628	0.6657	0.5667	0.5491	0.5712	0.4675	0.5452

Table 1: Testing Classification Error from Single Run

The error rates range between 45% and 70%, indicating that these models might not be doing more than randomly guessing the genre for each song. The Random Forest classifier performed the best, but not by much. A look at the confusion matrix for its results reveals that each genre was predicted the same amount of times (16.67%, or ⅙), with EDM, Rap, and Rock having the highest probability of being predicted correctly (sensitivity) (Appendix H). Every genre had a high specificity value, which only indicates that true negative predictions occurred often.

The Random Forest importance plots for this train-test split show that speechiness, danceability, instrumentalness, tempo, and valence were the most important variables for creating accurate trees (Appendix I). Speechiness, tempo, danceability, energy, and track duration were the top variables for contributing to the homogeneity of each node in the trees. Referring back to the correlation matrix, these variables all had stronger correlations with one or more of the genres with the top three sensitivity values. It is possible that limiting these methods to training on these attributes for classifying between those three genres would result in models with much greater success.

After performing cross validation, the mean classification error for each model was:

Mean Testing Classification Error: Cross Validation													
Model	KNN, k=1	KNN, k=3	KNN, k=5	KNN, k=7	KNN, k=9	KNN, k=11	KNN, k=13	KNN, k=15	LDA	QDA	NB	RF	LR
Error	0.6775	0.6773	0.6652	0.6623	0.6622	0.6562	0.6545	0.6534	0.5498	0.5389	0.5644	0.4503	0.5481

Table 2: Testing Classification Error from Cross-Validation

The variance within each model's testing errors were small, so the final classification error from each model was very similar compared to the single train-test split (Appendix J). Once again, the Random Forest Classifier performed the best on average. Paired statistical t-tests show that there is a statistically significant difference when comparing the performance between the Random Forest classifier and all the other models (Appendix K). However, having an error level of ~45% means that it should not be trusted as a method of classifying the six genres based only on the audio measurements.

Prediction

The single run testing MSE calculated for each model created were:

Testing Single Run MSE							
Model	Latin	EDM	R&B	Pop	Rap	Rock	Overall
LASSO	0.0702	0.4604	1.5005	0.0058	0.0009	0.1490	0.0390
Linear Regression	238.455	182.313	271.034	310.060	223.270	263.410	-
Stepwise Regression	238.705	183.258	272.434	310.389	224.370	264.320	-
Ridge Regression	306.860	217.343	304.297	456.473	267.050	272.050	-
PCA	238.455	182.313	271.034	310.060	223.270	263.410	-
Random Forest	0.1281	0.8597	1.5648	0.0134	0.0142	0.0329	-

Table 3: Testing MSE from Single Run

It can be seen that the LASSO and Random Forest models returned the lowest MSE. After cross validation, the resulting testing MSEs were:

Testing MSE: Cross Validation							
Model	Latin	EDM	R&B	Pop	Rap	Rock	Overall
LASSO	0.3444	0.2194	0.4695	0.4790	0.3343	0.6069	0.0791
Linear Regression	235.089	194.071	271.114	304.254	221.837	264.932	-
Stepwise Regression	235.467	194.074	271.737	303.884	222.260	265.114	-
Ridge Regression	292.119	239.612	313.063	456.692	265.895	284.074	-
PCA	235.082	194.071	271.114	304.254	221.837	264.942	-
Random Forest	0.3132	0.2259	0.4618	0.4929	0.3259	0.6743	-

Table 4: Testing MSE from Cross-Validation

LASSO and Random Forest again returned the lowest MSE. Paired statistical t-tests comparing the MSEs for all methods showed LASSO and Random Forest were not different from each other on a statistically significant level for all genres except for Rap, Pop, and Rock (Appendix L). Based on results from the t-test, the LASSO model only was run on the overall data set for comparison. From the single run the overall data set was not better or worse than the individual genres. After cross-validation there also did not seem to be a large difference in MSE in the overall data set versus the individual genres.

The LASSO model was also implemented on the whole, undivided data set to see if it had a better performance than within each genre. After cross validation, this model outperformed the others with a MSE of just 0.0791.

Conclusion

The results demonstrate that audio characteristics alone may not be sufficient to effectively differentiate between genres. Among the classification models evaluated, the Random Forest model showed the most promise in terms of classification accuracy after cross validation.

Since the spread of genres was mostly proportional across the dataset, it was decided to proceed with the challenge of a multi-class problem. Another approach would have been to turn this into a binary classification problem, either by choosing a dominant genre and classifying each song as belonging to it or not or by clustering the genres into two groups and predicting between the two of them. As seen by the single-run confusion matrix, the specificity per class potentially could have been increased if there was a focus on fewer genres at a time.

Furthermore, analysis conducted suggests that track popularity can potentially be predicted based on audio characteristics. Among the predictive models examined, LASSO regression demonstrated the best performance in predicting song popularity after cross validation. LASSO Regression is able to perform feature selection, which is beneficial in this context as it can put more weight on the predictors that affect popularity more. It is important to note that prediction within genres did not lead to higher accuracy when compared to prediction made over the entire dataset. This suggests that the predictive power of audio characteristics may not vary significantly across different genres.

To further this analysis, the problem statement could be turned into a binary classification problem. A threshold for what determines high or low popularity would be determined, data points labeled accordingly, and using binary classification to predict labels.

Music platforms are unlikely to adopt automatic genre classification based purely on this analysis. Manual assignment will ensure that songs are recommended to each genre's fans accurately. The ability to predict a song's popularity could have strong implications in how they could promote new releases in order to maximize the amount of artists promoted for maximum profitability for both parties. Exploring additional features beyond audio characteristics could improve the predictive capabilities of the models above. Spotify uses a vast array of features for its recommendation and analysis algorithms. Additional features that could be considered in future analysis could be user interaction features and textual features. User interaction features like play counts, skip counts, and amount of likes could provide insight into track popularity. Textual features such as song content and song lyrics could help to improve genre classification.

Lessons Learned

One of the main takeaways from this project is that multi-class classification is very difficult to implement, especially on a data set with many dimensions. There would have been more "success" in our results if we had chosen to find a way to separate the songs into two classes and then used binary classification techniques. In addition, our initial EDA indicated that there was not a lot of strong separation between the classes, and if we were in a production setting we probably would have wanted to either transform the data or find a new data set. Using PCA in this context also might have worked, but there ended up not being enough time left for us to study it.

One thing that we had initially implemented was a bagging algorithm that made final genre and popularity predictions using our models of choice, but we had very inaccurate results due to what we perceived as a bias-variance tradeoff. We included it in the presentation portion of this project, but got negative feedback on it and decided to omit it for the final report. In hindsight, there was a lack of understanding on how bagging is used in practice and it would have been better to clear it with an instructor first.

As far as the project requirements are concerned, our team did not see the full value of the presentation. In the workforce, good presentation slides are meant to be high level, with your verbal explanation used to support the information. However, the expectations for this project presentation required more detail than would be expected had it been presented to an actual management committee in the workforce.

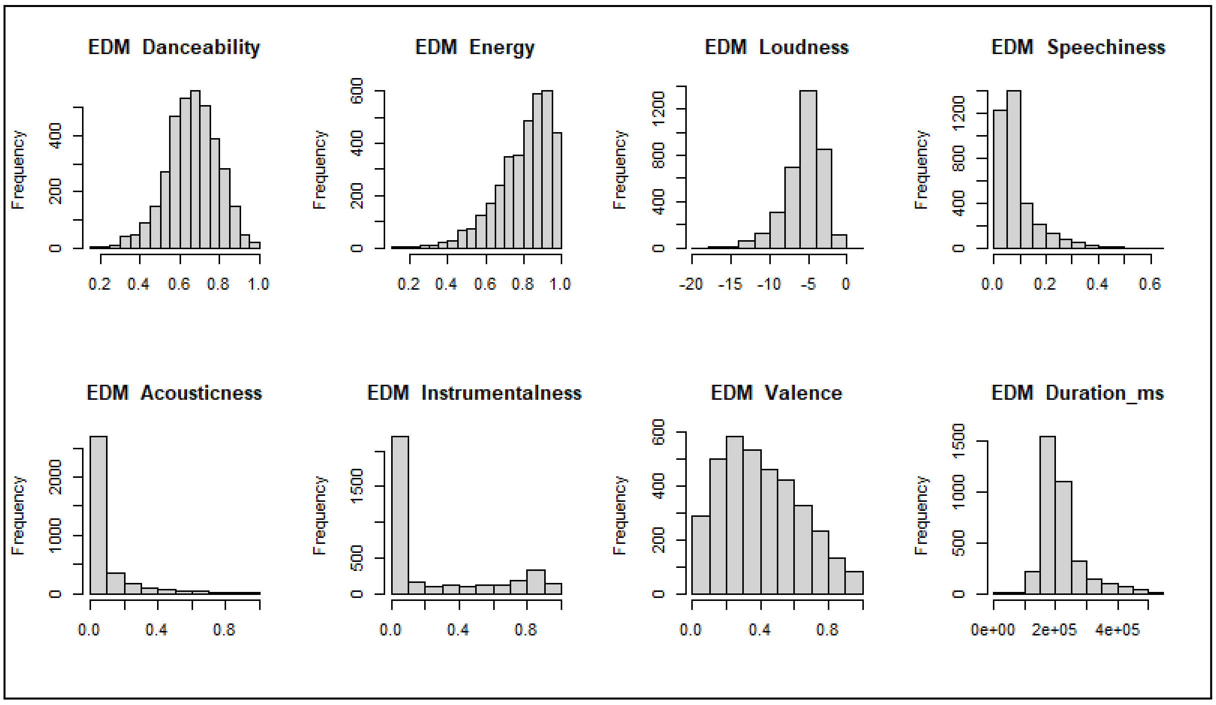
Overall, this class was informative and we learned a lot about the different data mining techniques and at the same time were able to hone report writing skills, which will be applicable in the workforce.

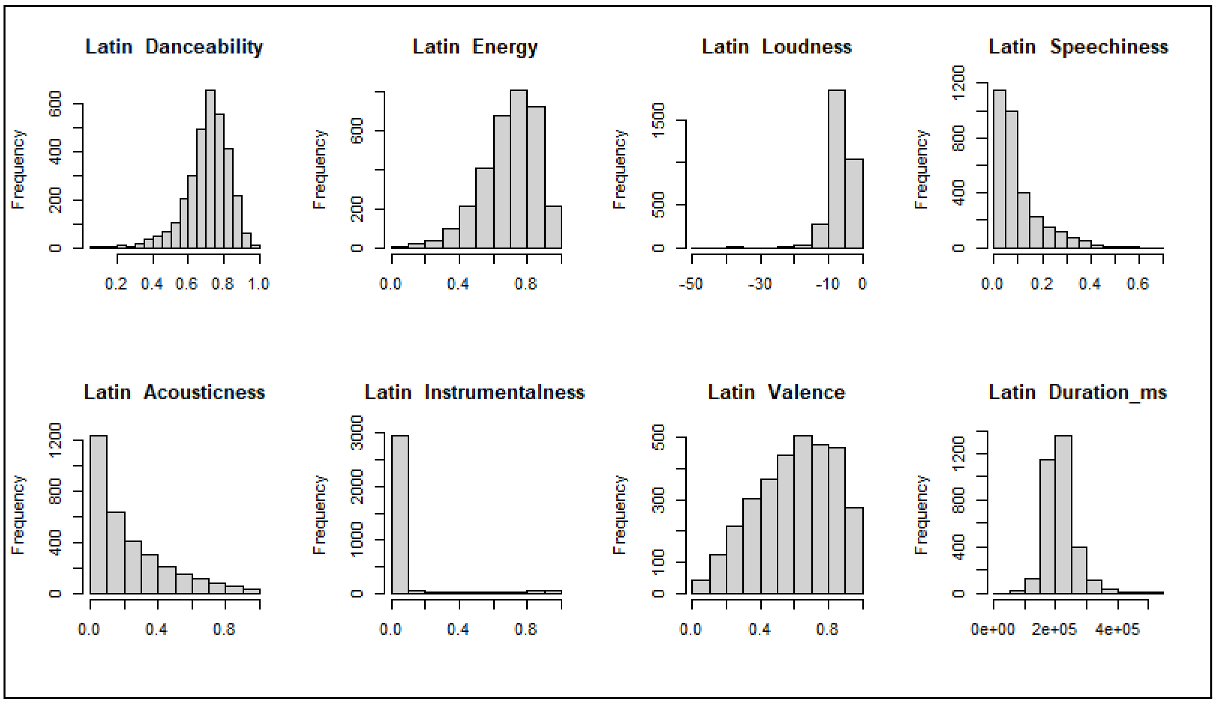
Appendix

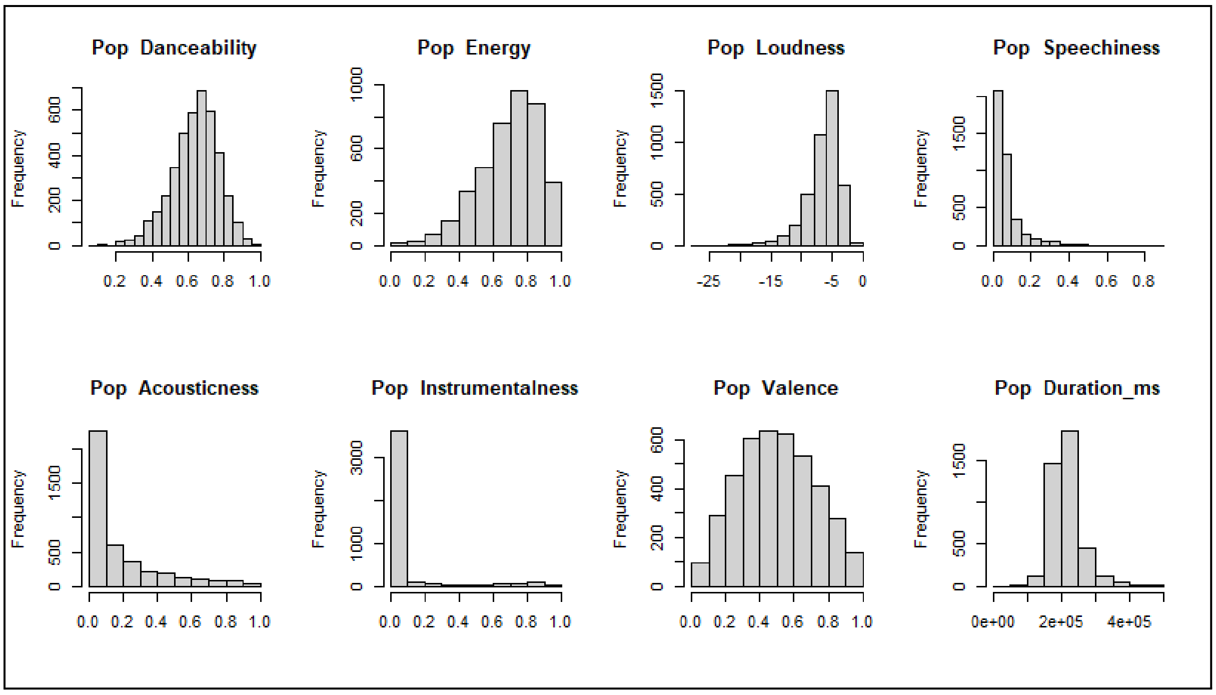
A. Data Observations for Each Genre

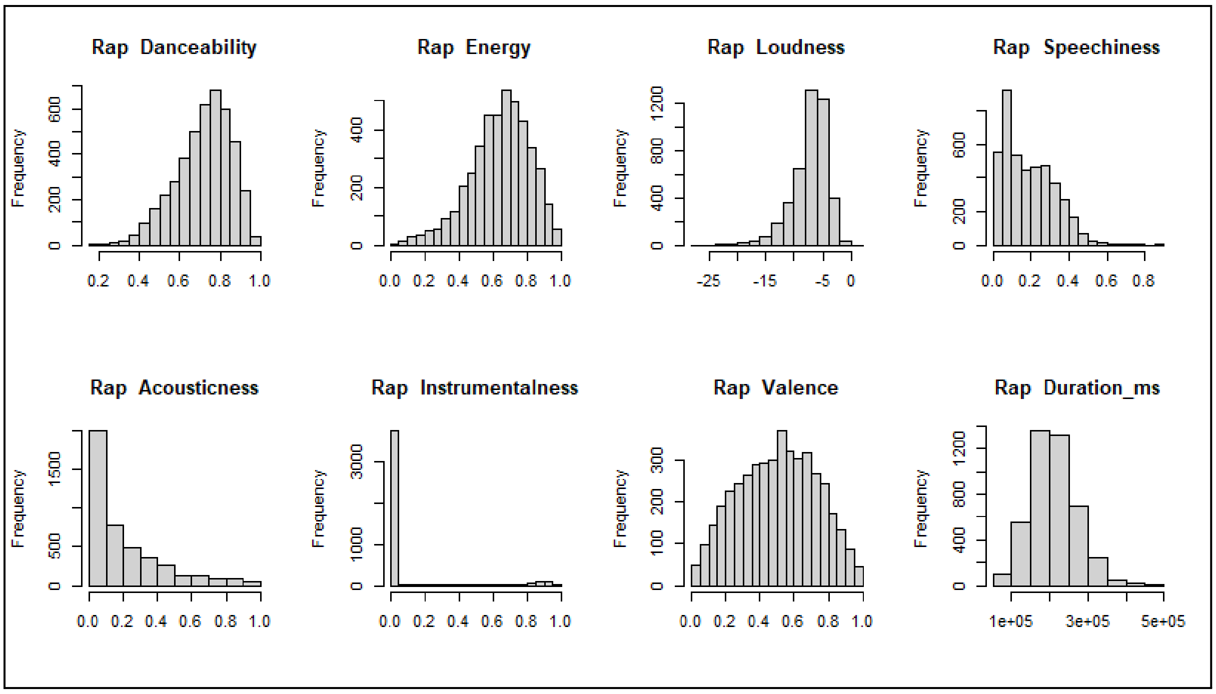
Proportion of Data per Genre							
Genre	EDM	Latin	Pop	R&B	Rap	Rock	All
Count	3561	3215	4065	3273	4353	3102	21569
% of Data	16.5%	14.9%	18.4%	15.2%	20.2%	14.4%	100%

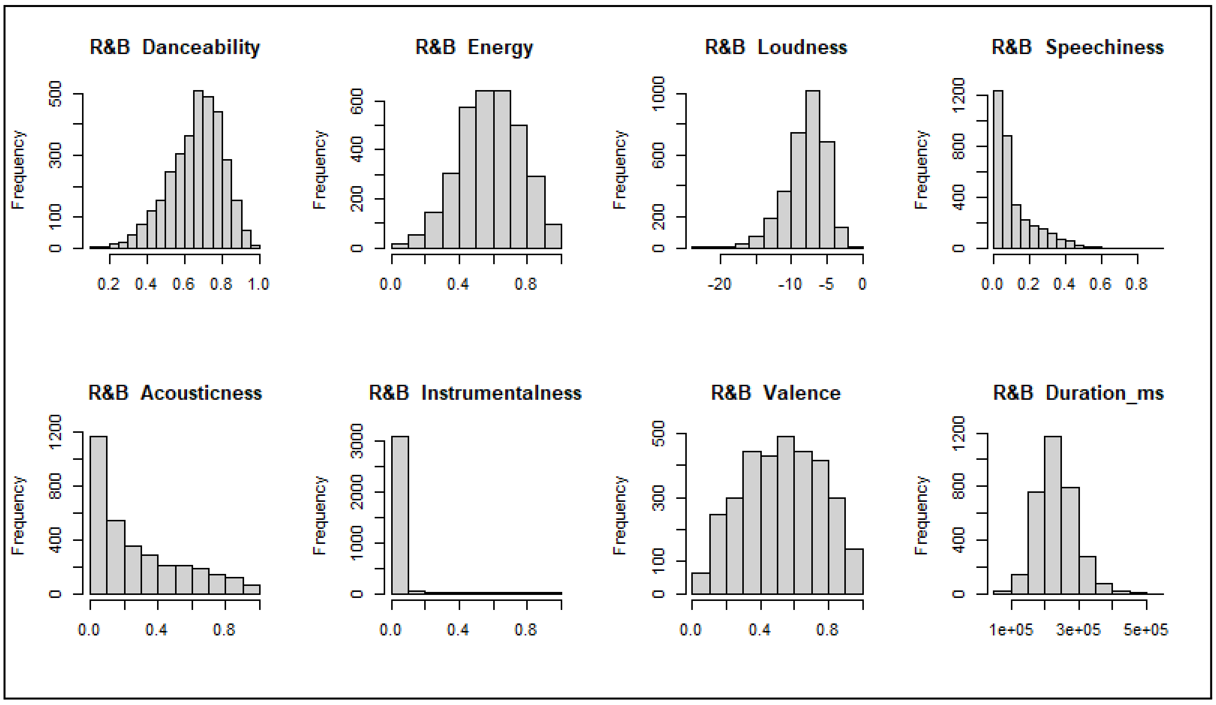
B. Distributions of independent variables with strong correlations to genre classification

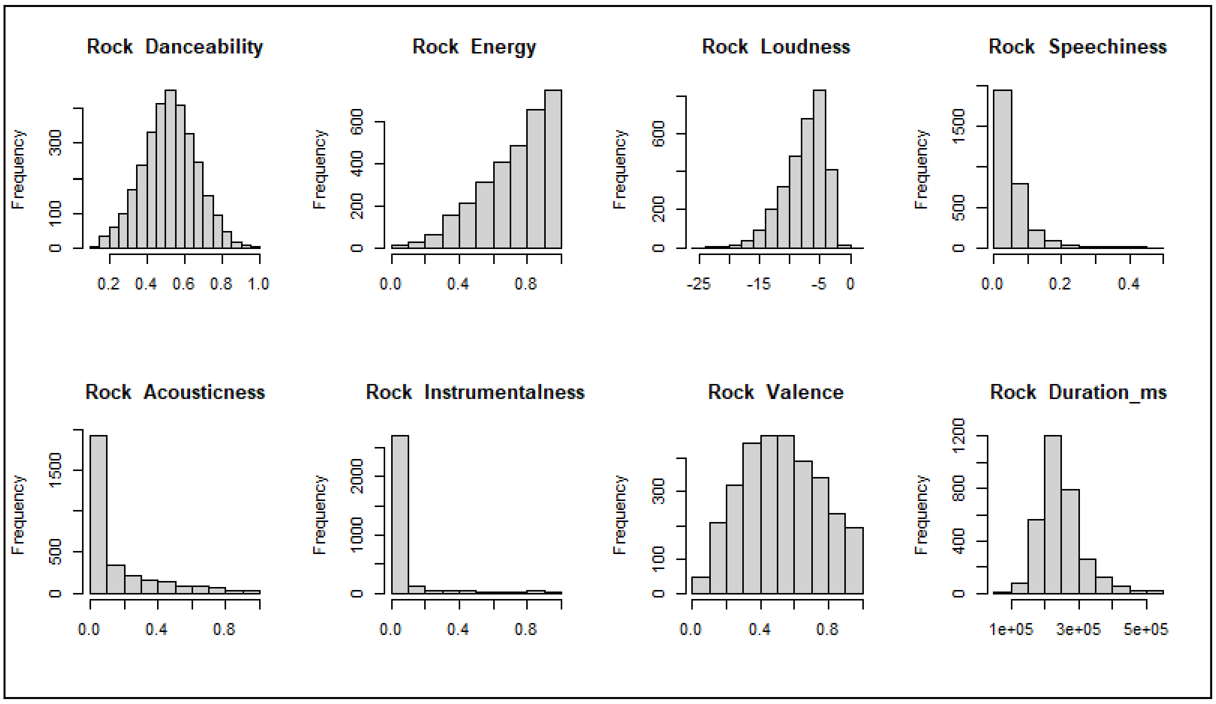

EDM Danceability
EDM Energy
EDM Loudness
EDM Speechiness
EDM Acousticness
EDM Instrumentalness
EDM Valence
EDM Duration_ms
Frequency
0
200
400
600
800
1200
1000
1500
2000
500
0.0
0.2
0.4
0.6
0.8
1.0
-20
-15
-10
-5
0e+00
2e+05
4e+05


Latin Danceability
Latin Energy
Latin Loudness
Latin Speechiness
Latin Acousticness
Latin Instrumentalness
Latin Valence
Latin Duration_ms
Frequency
0
100
200
300
400
500
600
800
1000
1200
1500
2000
3000
0.0
0.2
0.4
0.6
0.8
1.0
-50
-30
-10
0e+00
2e+05
4e+05

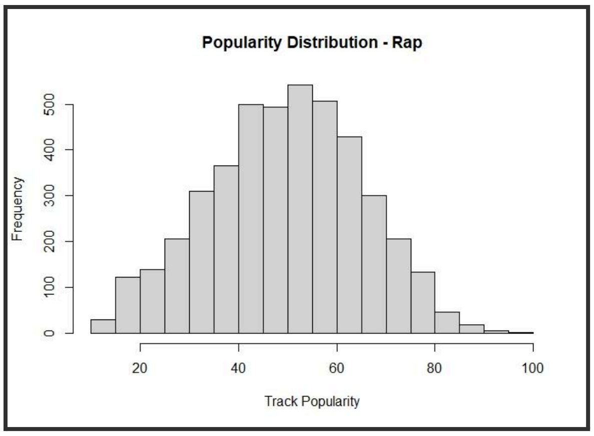

Pop Danceability
Pop Energy
Pop Loudness
Pop Speechiness
Pop Acousticness
Pop Instrumentalness
Pop Valence
Pop Duration_ms
Frequency

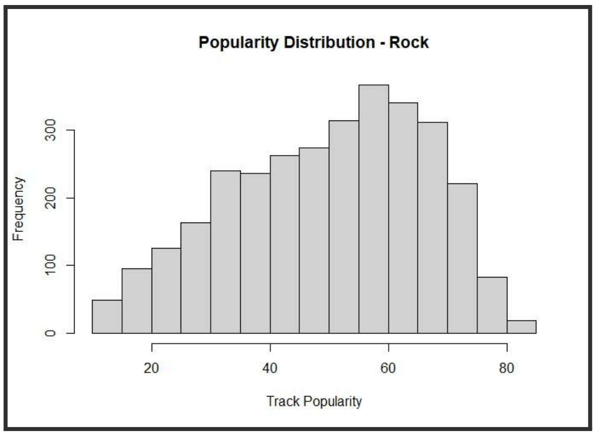

Rap Danceability
Rap Energy
Rap Loudness
Rap Speechiness
Rap Acousticness
Rap Instrumentalness
Rap Valence
Rap Duration_ms
Frequency

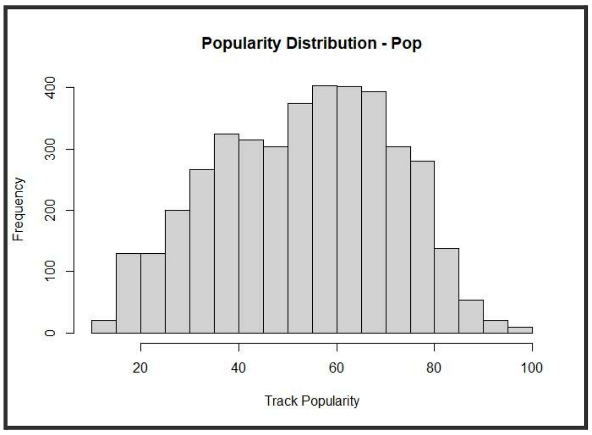


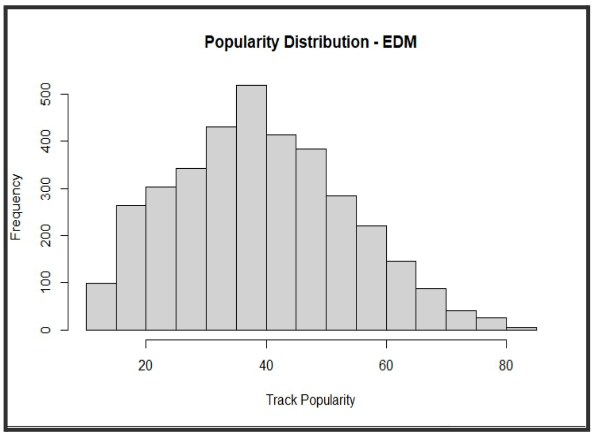


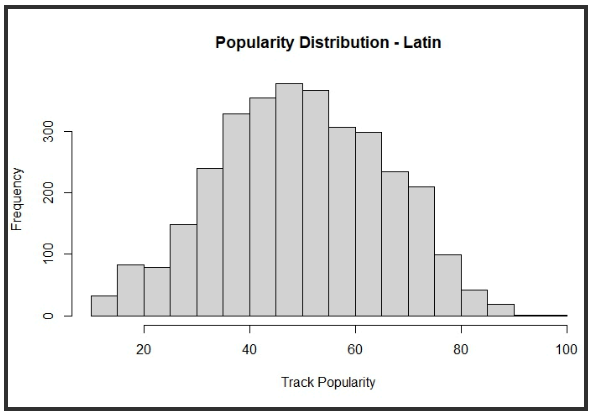
C. Genre Distribution

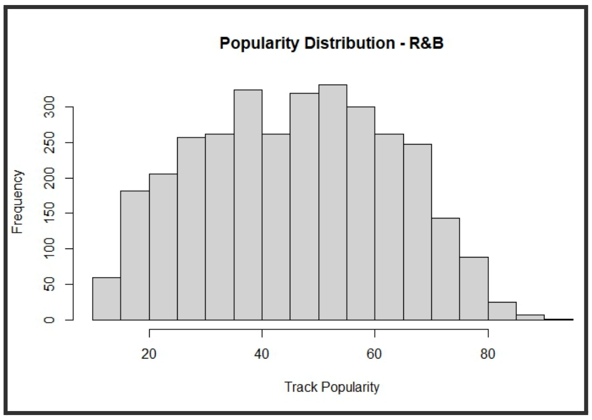






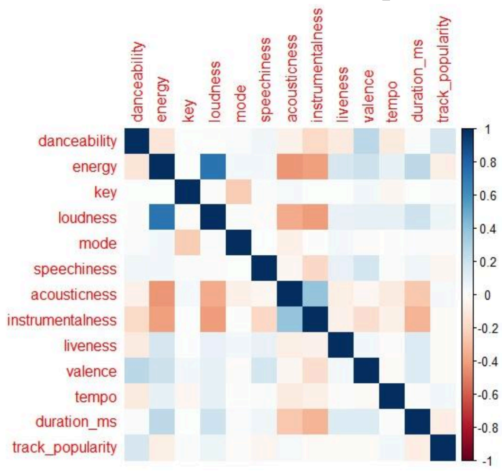




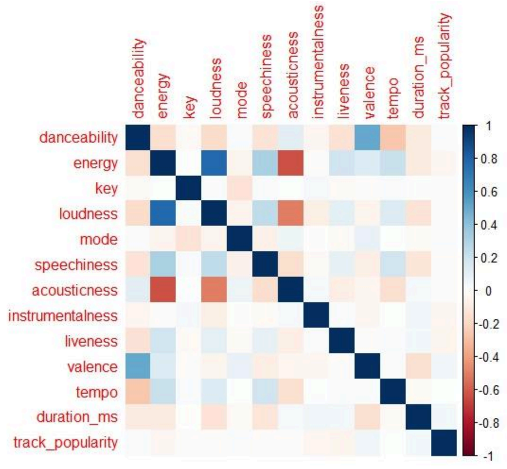


D. Correlation Plots by Genre

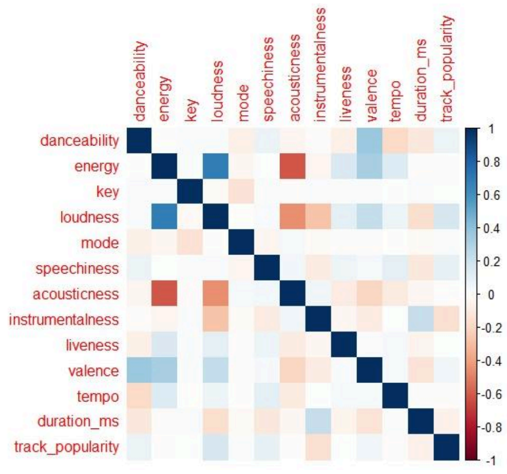
Correlation - Rap



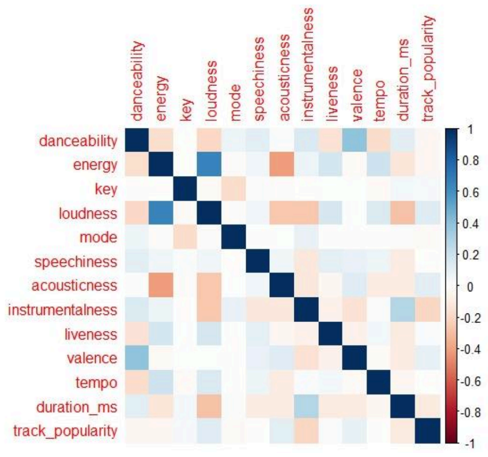
Correlation - Rock



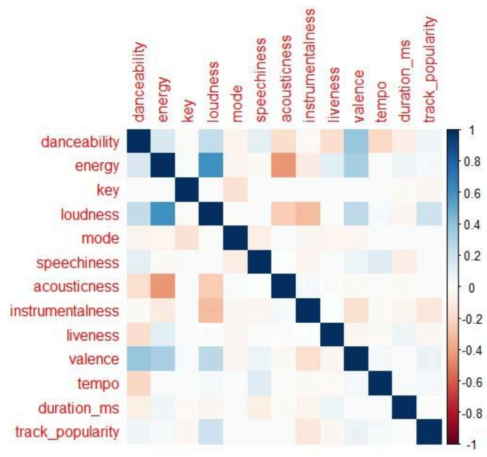
Correlation - Pop



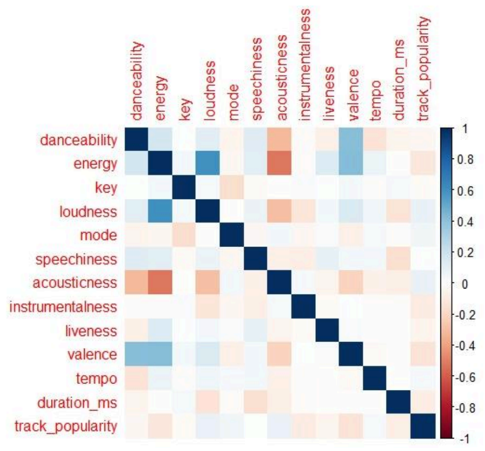
Correlation - EDM



Correlation - Latin



Correlation - R&B



E. Variables Selected in LASSO models for each genre

LASSO	
Genre	**Selected Variables**
Pop	dancebaility, energy, key, loudness, speechiness, acousticness,

	instrumentalness, valence, tempo
Rock	energy, acousticness, instrumentalness, liveness, valence, duration
Rap	danceability, energy ,key, loudness, speechiness, acousticness, instrumentalness, valence, tempo, duration
R&B	Energy, loudness, mode, acousticness, instrumentalness, liveness, valence, temp, duration_ms
Latin	Energy, key, loudness, speechiness, acousictness, instrumentalness, valence, tempo
EDM	Energy, key, loudness, mode, speechiness, acousticness, instrumentalness, valence, duration_ms
Overall	Energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, temp, duration

F. Variables selected in Stepwise model for each genre

Stepwise	
Genre	**Selected Variables**
Pop	dancebaility, energy, loudness, speechiness, instrumentalness, valence
Rock	Liveness, loudness, valence, duration, acousticenss, instrumentalness, energy
Rap	Danceability, energy, loudness, speechiness, acousticenss, instrumentalness, tempo, duration
R&B	Danceability, energy, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, key, duration_ms
Latin	Danceability, energy, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, key, duration_ms
EDM	Danceability, energy, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, key, duration_ms

G. Single Run, Random Forest Test Dataset Confusion Matrix

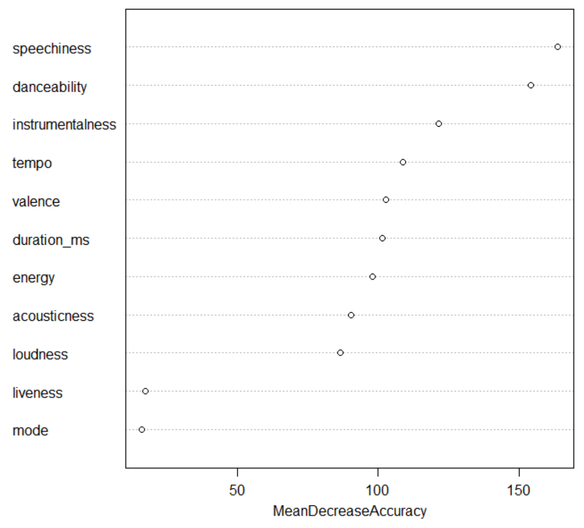
Random Forest Parameters		
Genre	**Number of Trees**	**Number of Variables**
Pop	300	2
Rock	200	1
Rap	700	4
R&B	200	3
Latin	100	4

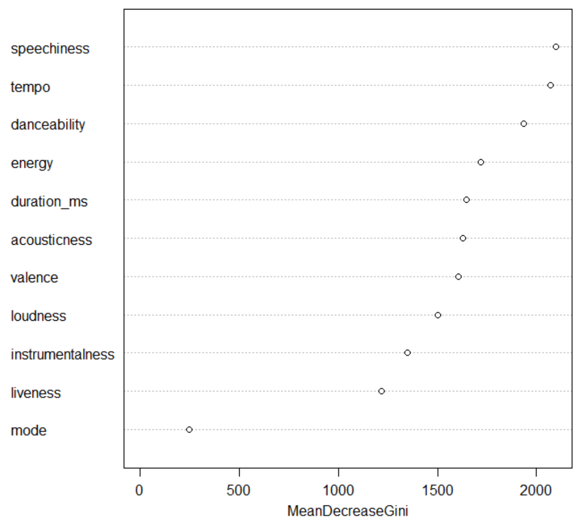
EDM	200	4

H. Single Run, Random Forest Test Dataset Importance Plots

```
Confusion Matrix and Statistics

          Reference
Prediction edm latin pop r&b rap rock
     edm   718    77 130  24  57   49
     latin  35   338  78  80  59   28
     pop   194   256 497 195 100  181
     r&b    18    99  94 401  63   79
     rap    81   240 142 261 770   35
     rock   26    62 131 111  23  700

Overall Statistics

               Accuracy : 0.5323
                 95% CI : (0.5201, 0.5446)
    No Information Rate : 0.1667
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4388

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: edm Class: latin Class: pop Class: r&b Class: rap Class: rock
Sensitivity              0.6698      0.31530    0.46362    0.37407     0.7183      0.6530
Specificity              0.9371      0.94776    0.82724    0.93414     0.8584      0.9341
Pos Pred Value           0.6806      0.54693    0.34926    0.53183     0.5036      0.6648
Neg Pred Value           0.9342      0.87375    0.88521    0.88182     0.9384      0.9308
Prevalence               0.1667      0.16667    0.16667    0.16667     0.1667      0.1667
Detection Rate           0.1116      0.05255    0.07727    0.06234     0.1197      0.1088
Detection Prevalence     0.1640      0.09608    0.22124    0.11723     0.2377      0.1637
Balanced Accuracy        0.8035      0.63153    0.64543    0.65410     0.7883      0.7936
```

I. Single Run, Random Forest Test Dataset Importance Plots





J. Variance of CrossValidated Classification Errors

Mean Testing Classification Error Variance: Cross Validation

Model	KNN, k=1	KNN, k=3	KNN, k=5	KNN, k=7	KNN, k=9	KNN, k=11	KNN, k=13	KNN, k=15	LDA	QDA	NB	RF	LR
Error	1.6E-4	2.E-4	1.6E-4	1.8E-4	1.4E-4	1.3E-4	7.1E-5	1.8E-4	1.9E-4	2.1E-4	2E-4	1.7E-4	3.7E-3

K. P-Values of Paired Statistical T-Tests - Classification

Null Hypothesis: The error rates of the Random Forest Model are not different from the error rates of Model X at a 95% level.

Alternate Hypothesis: The error rates of the Random Forest Model are different from the error rates of Model X at a 95% level.

Classification Statistical T-Test: P-Values												
Model X	KNN, k=1	KNN, k=3	KNN, k=5	KNN, k=7	KNN, k=9	KNN, k=11	KNN, k=13	KNN, k=15	LDA	QDA	NB	LR
Error	6E-23	4E-23	1E-23	2E-23	9E-26	6E-25	1E-25	5E-24	4E-18	4E-17	9E-19	1.6E-6

L. P-Values of Paired Statistical T-Tests - Prediction

Null Hypothesis: The error rates of the LASSO Model are not different from the error rates of Model X at a 95% level.

Alternate Hypothesis: The error rates of the LASSO Model are different from the error rates of Model X at a 95% level.

Prediction Statistical T-Test: P-Values - Rock					
Model X	**Linear Regression**	**Stepwise**	**Ridge Regression**	**PCR**	**Random Forest**
Error	5.358e-71	2.846e-71	4.397e-60	5.162e-71	2.2E-25

Prediction Statistical T-Test: P-Values - Rap					
Model X	**Linear Regression**	**Stepwise**	**Ridge Regression**	**PCR**	**Random Forest**
Error	1.013e-74	6.561e-75	1.854e-64	1.013e-74	3.04E-7

Prediction Statistical T-Test: P-Values - Pop					
Model X	**Linear Regression**	**Stepwise**	**Ridge Regression**	**PCR**	**Random Forest**
Error	1.211e-77	2.018e-77	3.069e-60	1.211e-77	1.3E-12

Prediction Statistical T-Test: P-Values - EDM					
Model X	**Linear Regression**	**Stepwise**	**Ridge Regression**	**PCR**	**Random Forest**
Error	1.002E-73	1.2E-734	1.9E-68	1.002E-73	0.8137

Prediction Statistical T-Test: P-Values - Latin					
Model X	**Linear Regression**	**Stepwise**	**Ridge Regression**	**PCR**	**Random Forest**
Error	5.2E-68	9.6E-68	4.7E-65	5.3E-68	0.2872

Prediction Statistical T-Test: P-Values - R&B					
Model X	**Linear Regression**	**Stepwise**	**Ridge Regression**	**PCR**	**Random Forest**
Error	9.7E-76	1.8E-75	2.4E-71	9.7E-76	0.7202

References

Arvidsson, Joakim. (October 2023). 30000 Spotify Songs, Version 2. Retrieved February 27, 2024 from https://www.kaggle.com/datasets/joebeachcapital/30000-spotify-songs/data.