**1. Introduction**

Genres in any industry are subjective. Definitions can change over time, and the more specific you try to get, the more people will angrily try to correct you. But the basic genres will have a few principle aspects that make it easier to identify them [1].

For the most part, genres are defined by the experiences they evoke and common compositional elements. In the movie industry we have Drama, Action, horror movies, comedy etc. whereas in other artistic fields like the music industry, there are hundreds of genres and subgenres that have been made distinct by the cultures they come from, the languages spoken and the types of instruments played.

In this project, we attempted to algorithmically predict the genre of a track based on the features of the song. This required using a large dataset obtained from Spotify API to create machine learning models.

**1.1 Objectives**

* Clean and Analyse Data from Spotify’s API containing information on the features of several tracks.
* Perform exploratory data analysis (EDA) on the large sample of data, plotting quick and interactive graphs in Tableau and heatmaps that display correlations.
* Identify a machine learning model that can predict the genre of a song with little bias and variance.
* Create a Website that incorporates machine learning algorithms to predict the genre of a song based on the features that are input by the user.

**2. Data Gathering**

To begin building a model, we needed a lot of data. A large dataset was needed so that our model could make accurate class label assignments.

On the Kaggle website, there were multiple pre-compiled csv files with collections of information from Spotify API. Therefore, we simply chose a dataset containing Spotify data from 1921 up until 2020 and containing 160,000 tracks. Doing this made data collection quick and efficient as well as provided our model with large data.

We obtained CSV files for “songs by year”, “songs by genre”, “types of genres” and a “master” CSV. Each of which contained song features from the Spotify API.

**3. Cleaning and Data exploration**

The data cleaning process was done in Jupyter notebook using the pandas and NumPy libraries. We used the data\_w\_genres.csv file and converted it into a dataframe. These were the steps we took to clean our data:

**3.1 Checked for missing rows**

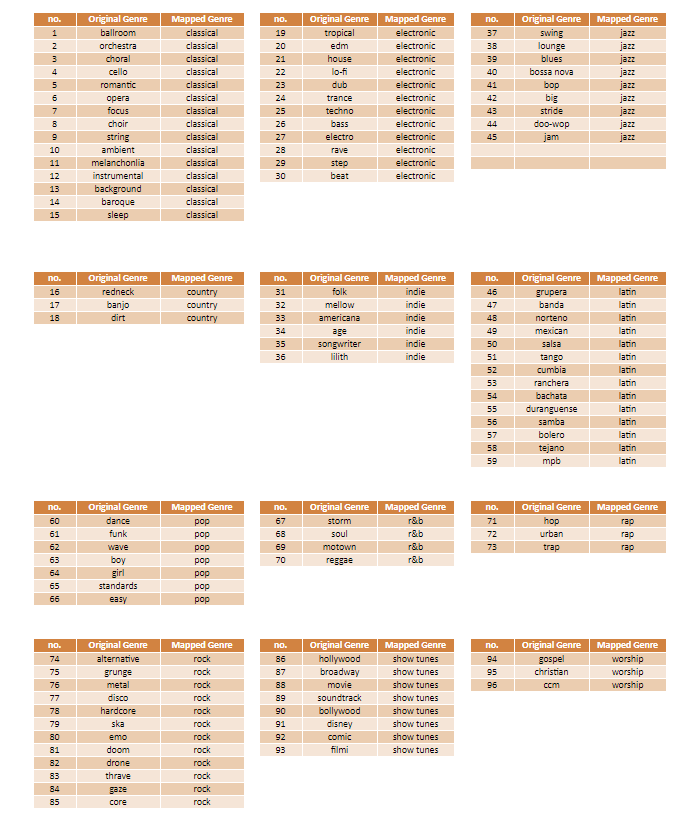
We inspected the dataframe with the info command and found out that all rows were complete.

**3.2 Genre to Artist restructuring**

For each artist on each row, we recognised that their music fell into multiple genres and was represented by lists. We restructured the dataframe by breaking down the list— so that each song could match one genre and if it matched more than one, the song would appear twice on the dataframe for each genre. Finally, we dropped rows that had null values.

**3.3 Genre Mapping**

We figured that there were far too many types of genre in our dataset that would overcomplicate the model. Therefore, we simplified our dataframe by mapping all 96 genre groups to 12 genres. These were the genre categories we came up with:



**Fig X**: Diagram displaying the Original sub-genres grouped into twelve main genres.

**3.4 Encoding Genres**

The final stage of our data cleaning process was to one-hot encode the data. This entailed converting the genre columns into 12 columns representing genre categories, and then using 0’s and 1’s to represent the nonexistence or existence of a category, respectively.



Fig 1:

**4. Exploratory Data Analysis (EDA)**

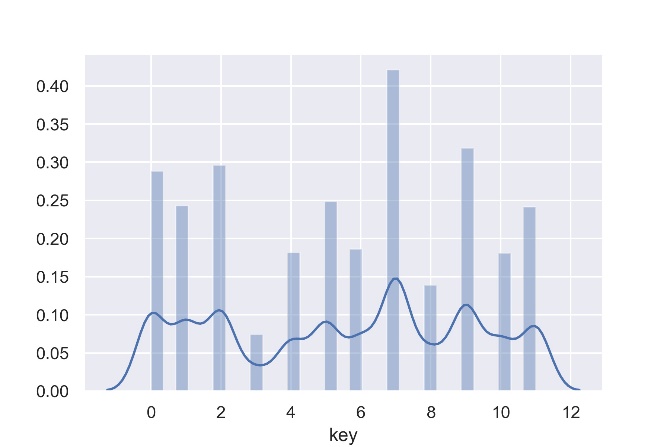
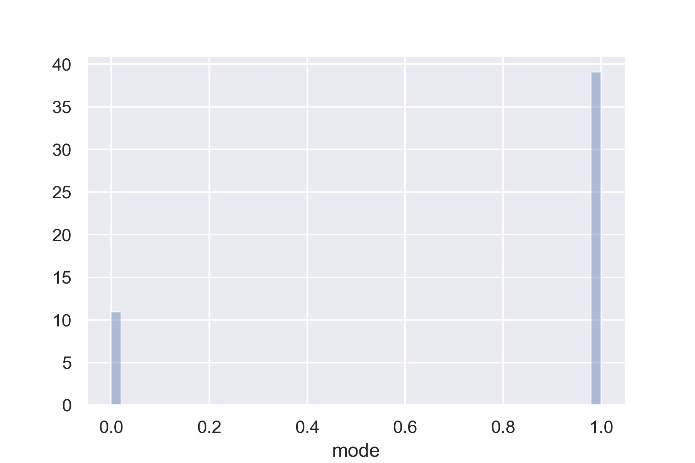
Before fitting machine learning models to the data set, we had to understand the nature of each genres audio attribute and get acquainted with the data. Tools used for this were Python’s libraries—Matplotlib, Pandas and seaborn— as well as Tableau.

4.1 Correlations

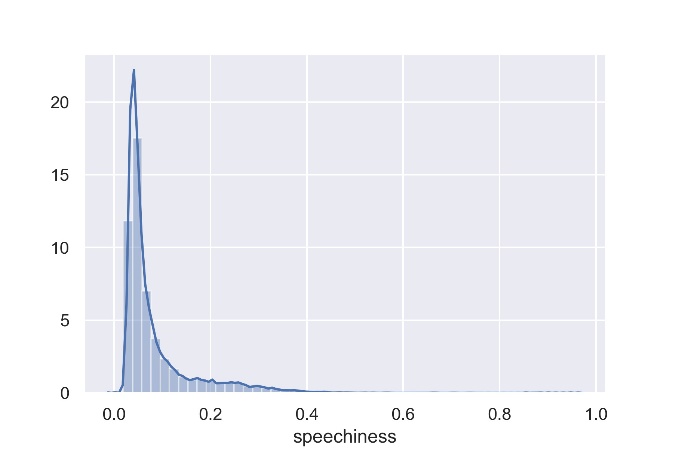
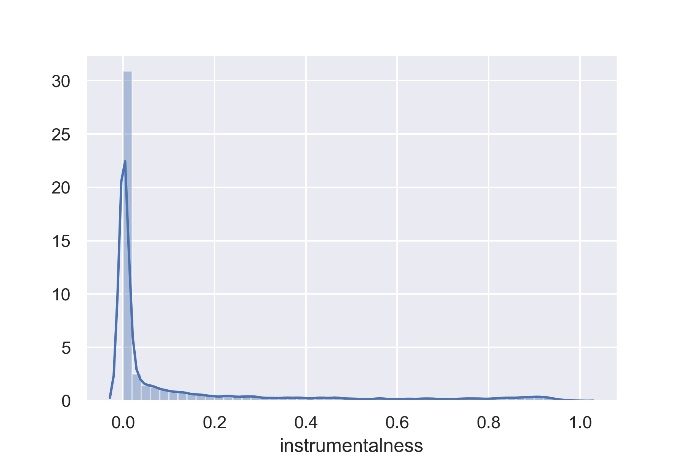


**Fig. X:**

**4.2 Feature distribution**

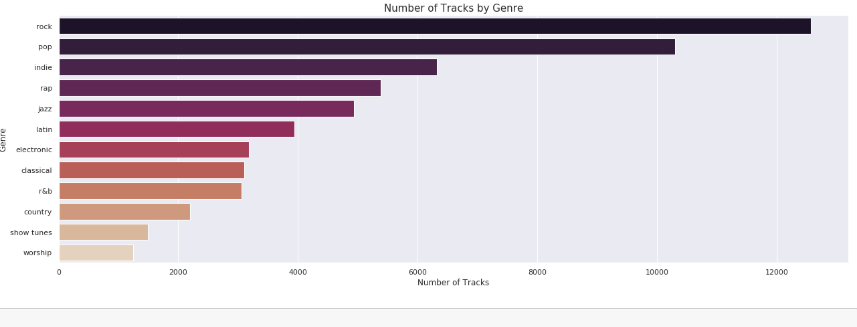


**Fig X: mode distribution Fig X: key distribution**



**Fig X: Instrumentalness distribution Fig X: Speechiness distribution**

**4.3 Genre Popularity**



**Fig X:**

**4.4**

**5. Predictive Models**

Once we determined the main categories of genre, we created our predictive model.

We used supervised learning, classification algorithms to predict which genre a track would belong too. The 3 models We used were: Logistic regression analysis, random forest classifier and Xtreme gradient boosting classifier.

5.1 Running the Models

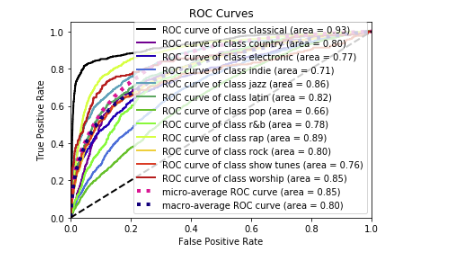
After balancing the data and splitting it into training and testing sets (The test train split for our data set was 20% vs 80% respectively) we scaled the X variables (the features we obtained from Spotify API). The features had to be scaled because there was a huge variance in magnitude between some features and this impacts the model.

Finally, we fit our models to the trained data and used the following metrics to score the quality of each model: ROC AUC, Accuracy, Precision and Recall.

**5.1.1 Logistic Regression Model**

We started our modelling with the simplest classification model—logistic regression. This model was chosen due to the dependent variable(target) in our data being categorical. It works by using the sigmoid function (logistic function) to map and predict the probabilities of a track falling into a genre. If the probability of a track were closer to 1, it will be transformed into a binary value of 1(meaning it falls into that category). Else, it will be transformed into a binary value of 0(meaning it does not fall into that genre).

From the metric tests, the accuracy score was low at 0.41 and precision values were unimpressive (More information on this can be found in **Appendix A**). The ROC test showed that the logistic model was relatively accurate for classical music, rap, jazz and worship. On the other hand, indie music, pop and show tunes were poorly predicted.



**Fig.2:** ROC curves for Logistic Regression Model.

**5.1.2 Random Forest Classifier**

Our second model was slightly more complex. A random forest classifier is an ensemble tree-based learning algorithm. The Random Forest Classifier is a set of decision trees from randomly selected subset of training set. It aggregates the votes from different decision trees to decide the final class of the test object [2].

We used this model because of its ability to run efficiently on large databases and its ability to produce a highly accurate classifier [2]. We expected to face problems with overfitting.

Fig.3 shows our ROC curve metric test results. The ROC curve for the random forest classifier was clearly better than the Logistic regression model at predicting the Genre of tracks for all categories in our dataset and the accuracy score was 0.58. We felt like this could be improved.

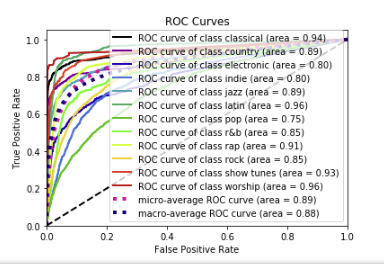


Fig. 3: ROC curve for Random Forest Classifier

The music categories with the best ROC curve ratings were worship (0.96) and Latin (0.96).

**5.1.3XGBoosting Classifier**

The final model that we tested was the extreme gradient boosting classifier. It is an implementation of gradient boosted decision trees designed for speed and performance [3]. Like the previous models, we fit the model to the trained data and then used it for predicting the test data. Based on the metric test, the XGBoost model was better than the logistic model but was ever so slightly worse than the random forest classifier.

The XGBoost model had an accuracy of 0.57 and the ROC curve had similar values and a similar shape to the random forest.

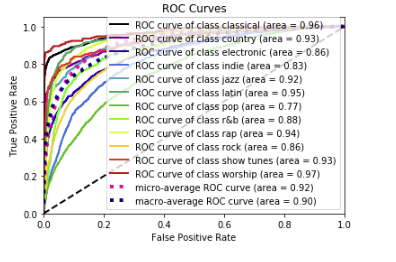


Fig. 4: ROC curve of XGBoost classifier

The categories with the best ROC curve scores were Worship (0.97) and Classical (0.96).

**6. Dashboard – Desktop & Mobile**

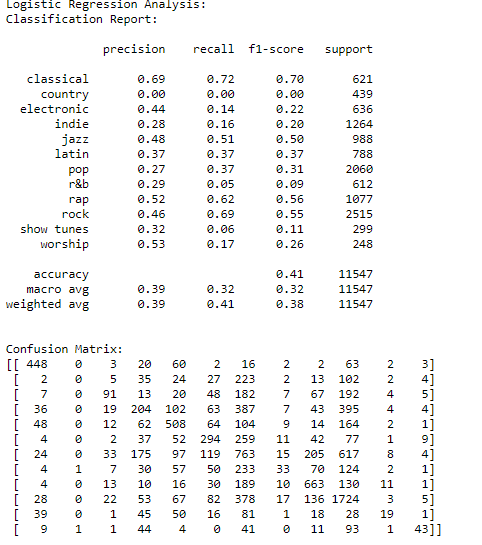
**References**

[1] <https://towardsdatascience.com/music-genre-prediction-with-spotifys-audio-features-8a2c81f1a22e>

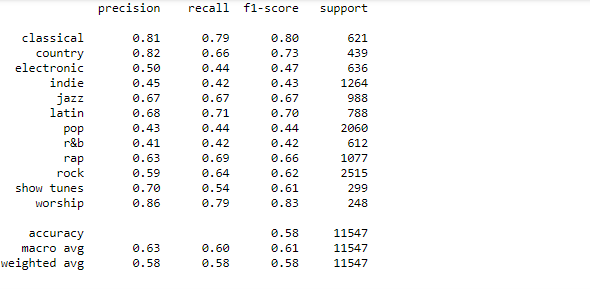
[2] <https://towardsdatascience.com/random-forest-classification-and-its-implementation-d5d840dbead0>

[3] <https://medium.com/datadriveninvestor/using-extreme-gradient-boosted-trees-in-machine-learning-classification-problems-a7bb04be759>

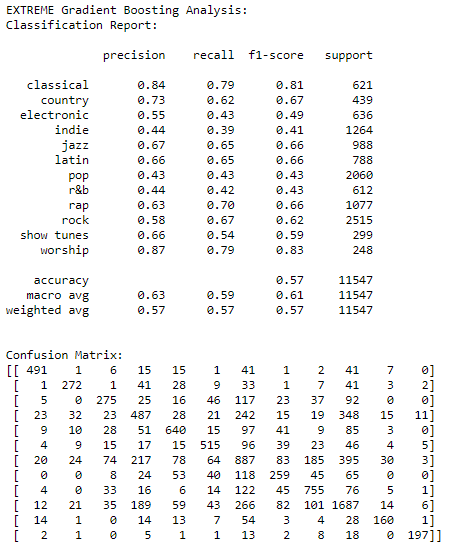
**Appendices**



**Appendix A**: Metric test results for Logistic regressions Model. This image displays the Classification report and the confusion matrix for the data.

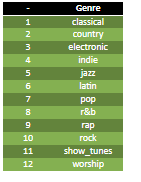


**Appendix B**: Matric test results for Random forest classifier.

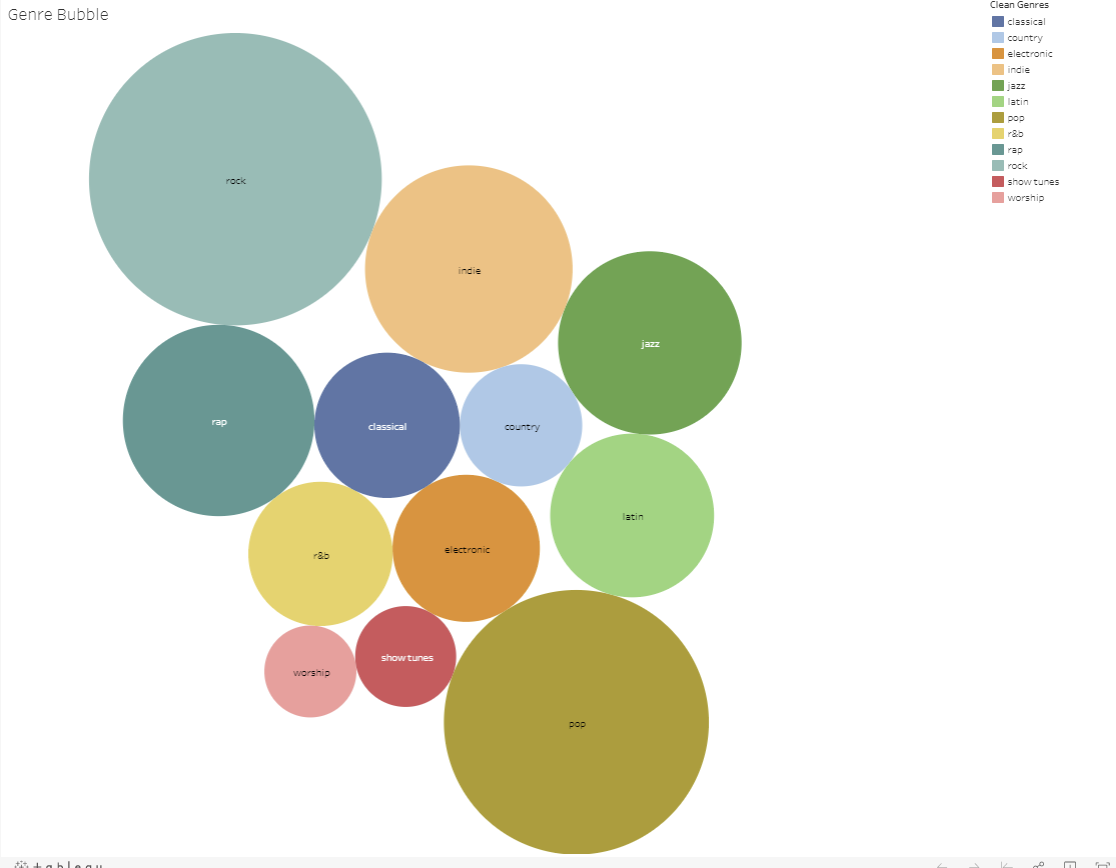


**Appendix C**: Metric test results for XGBoost classifier

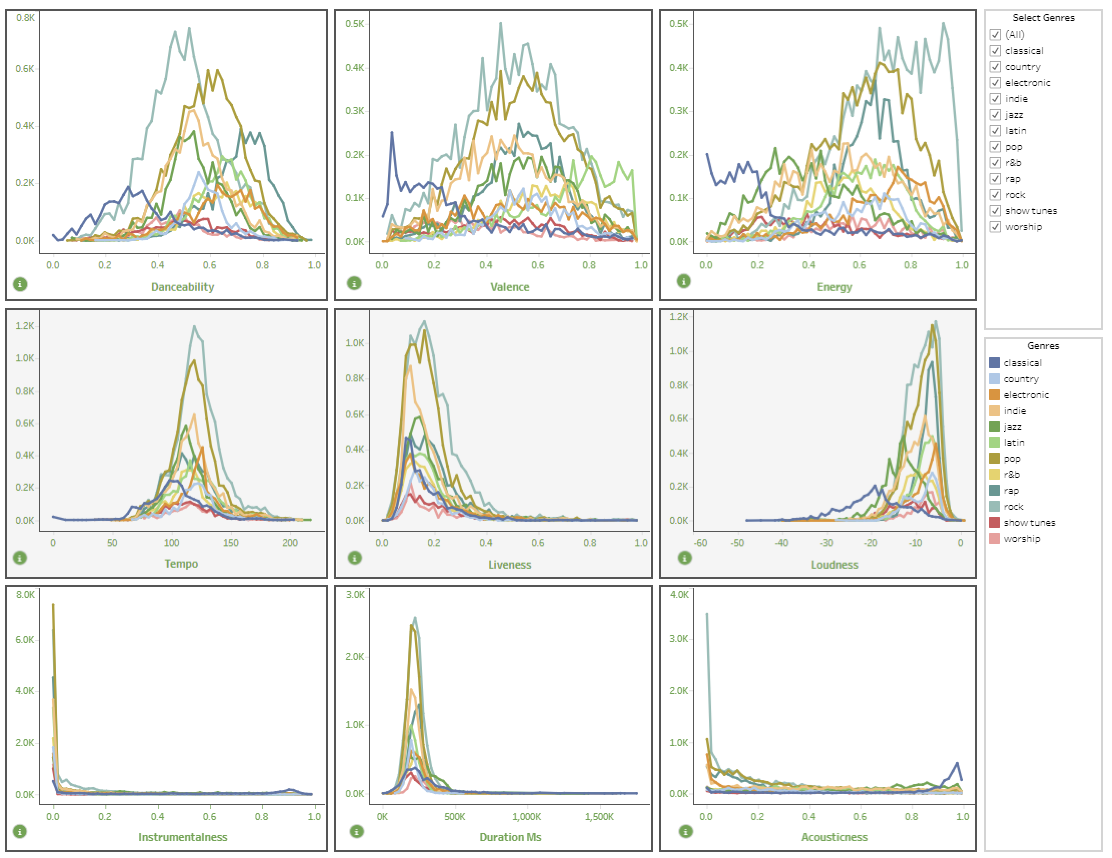




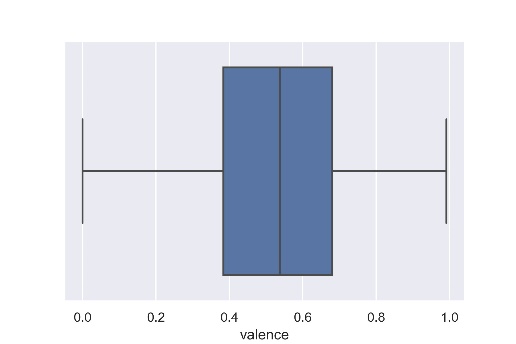
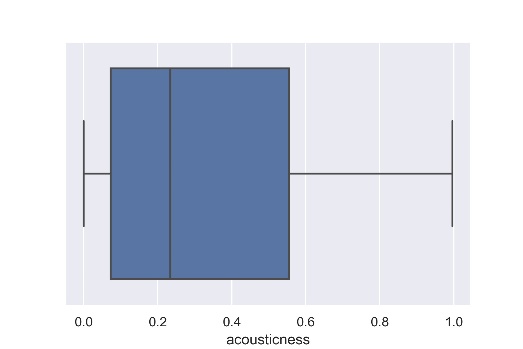
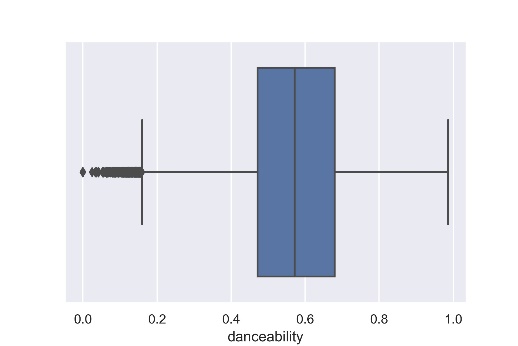
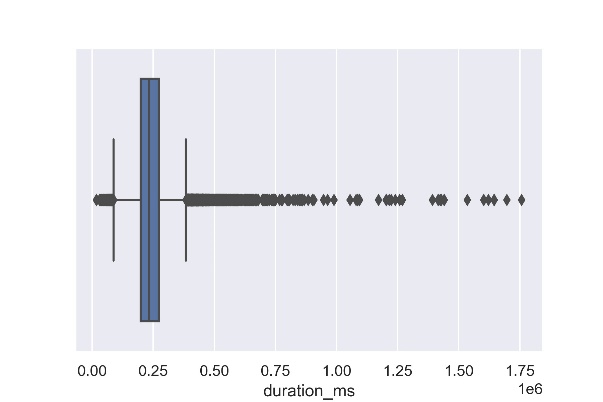
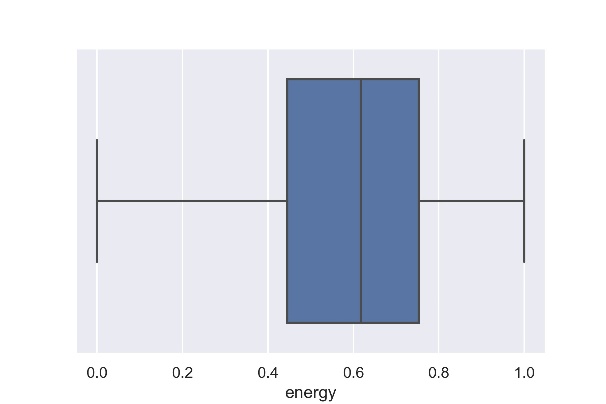
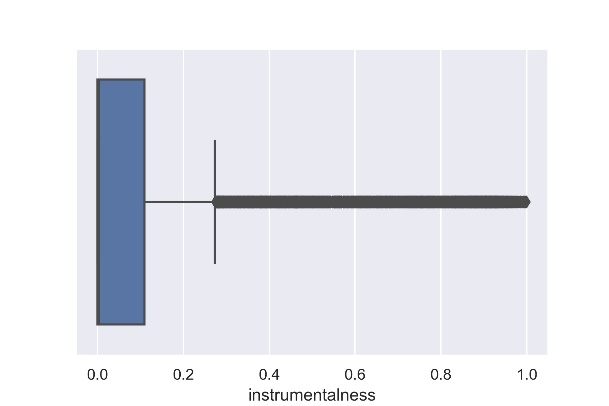
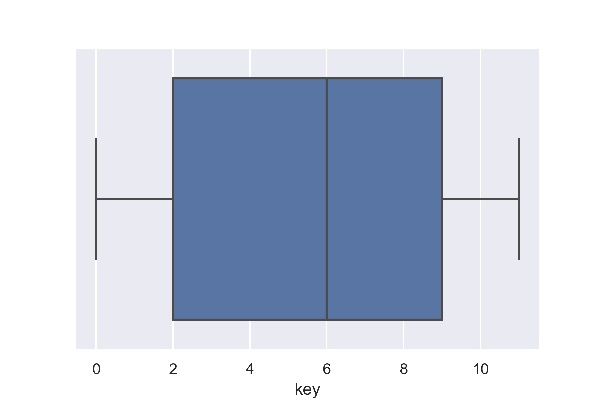
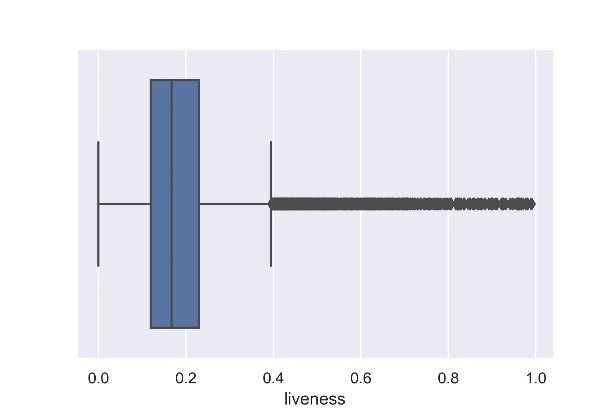
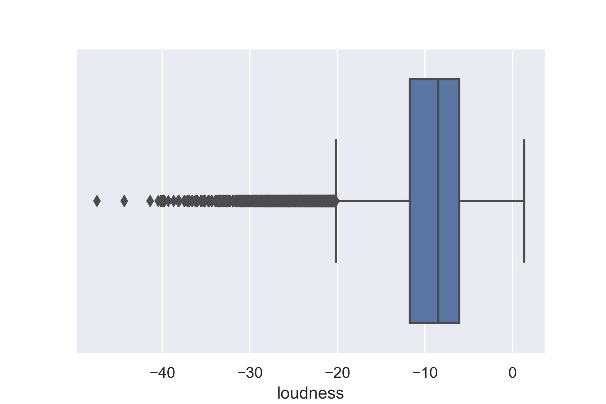
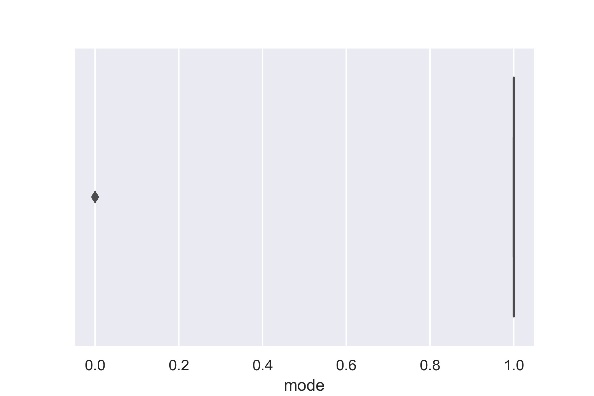
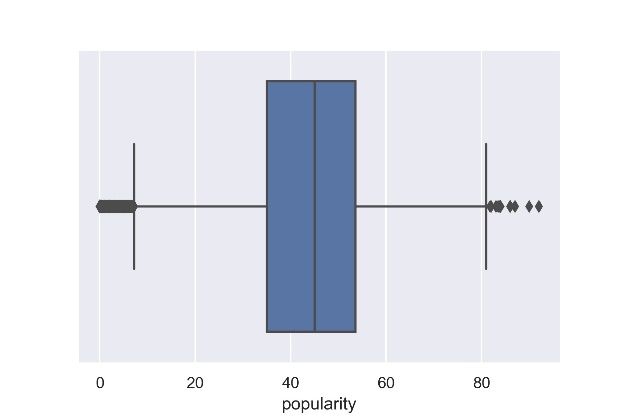
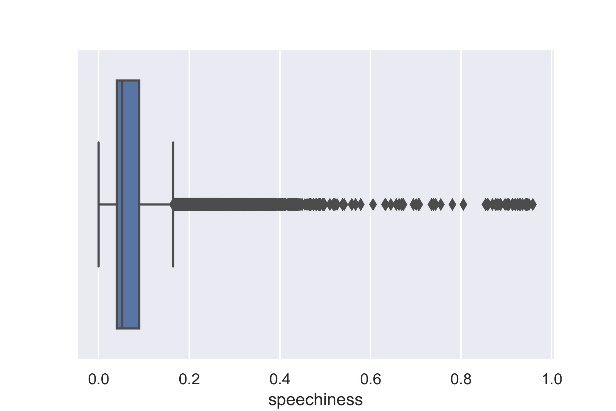
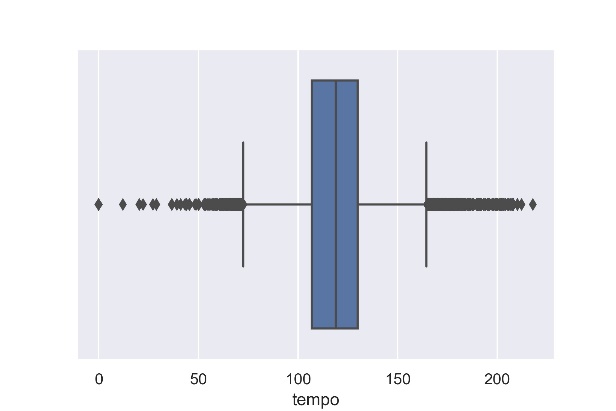
Appendix D:



Appendix E:



Appendix F:



Appendix G: