

Contents

[1. Introduction 3](#_Toc51008001)

[1.1 Objectives 3](#_Toc51008002)

[2. Data Gathering 3](#_Toc51008003)

[3. Cleaning and Data exploration 3](#_Toc51008004)

[3.1 Checked for Missing Rows 3](#_Toc51008005)

[3.2 Genre to Artist Restructuring 3](#_Toc51008006)

[3.3 Genre Mapping 4](#_Toc51008007)

[3.4 Encoding Genres 4](#_Toc51008008)

[4. Exploratory Data Analysis (EDA) 5](#_Toc51008009)

[4.1 Correlations 5](#_Toc51008010)

[4.2 Feature distribution 6](#_Toc51008011)

[4.3 Genre Popularity 6](#_Toc51008012)

[5. Predictive Models 7](#_Toc51008013)

[5.1 Running the Models 7](#_Toc51008014)

[5.1.1 Logistic Regression Model 7](#_Toc51008015)

[5.1.2 Random Forest Classifier 7](#_Toc51008016)

[5.1.3XGBoosting Classifier 8](#_Toc51008017)

[6.Genre Predictions 9](#_Toc51008018)

[7. App – Desktop & Mobile 9](#_Toc51008019)

[8. Technology Resources 9](#_Toc51008020)

[10. Limitations and Future Analysis 9](#_Toc51008021)

[11. Conclusion 10](#_Toc51008022)

[References 10](#_Toc51008023)

[Appendices 11](#_Toc51008024)

# 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 predictive 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. Next, we checked for missing rows, performed genre to artist restructuring, mapped sub-genre to target genres and encoded our genre column.

## 3.1 Checked for Missing Rows

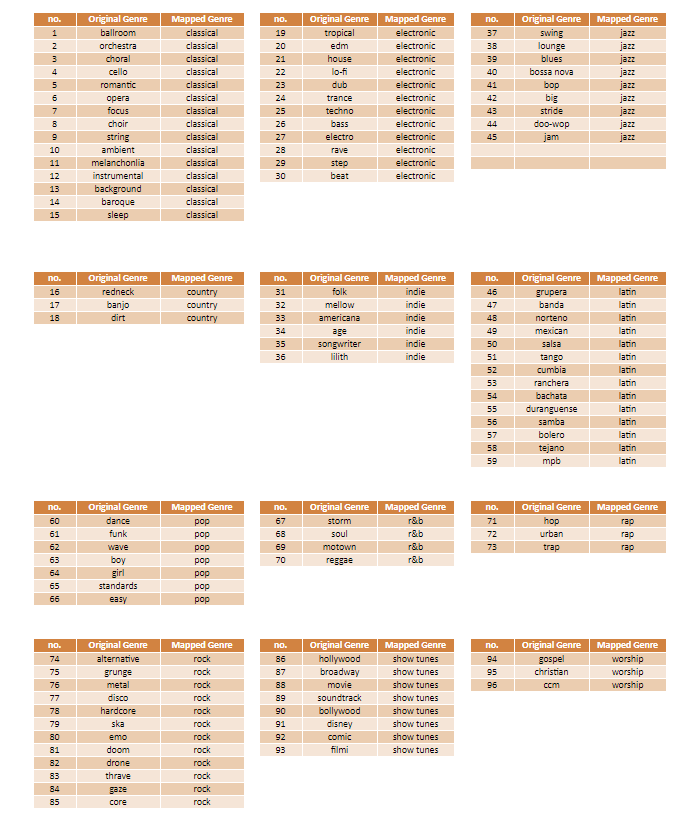
We inspected the dataframe with the info command and found out that all rows were complete.

## 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 overcomplicated the model. Therefore, we simplified our dataframe by mapping all 96 genre groups to 12 genres. Fig. 1 displays the music categories we came up with and their sub-categories.



**Fig 1**: 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.

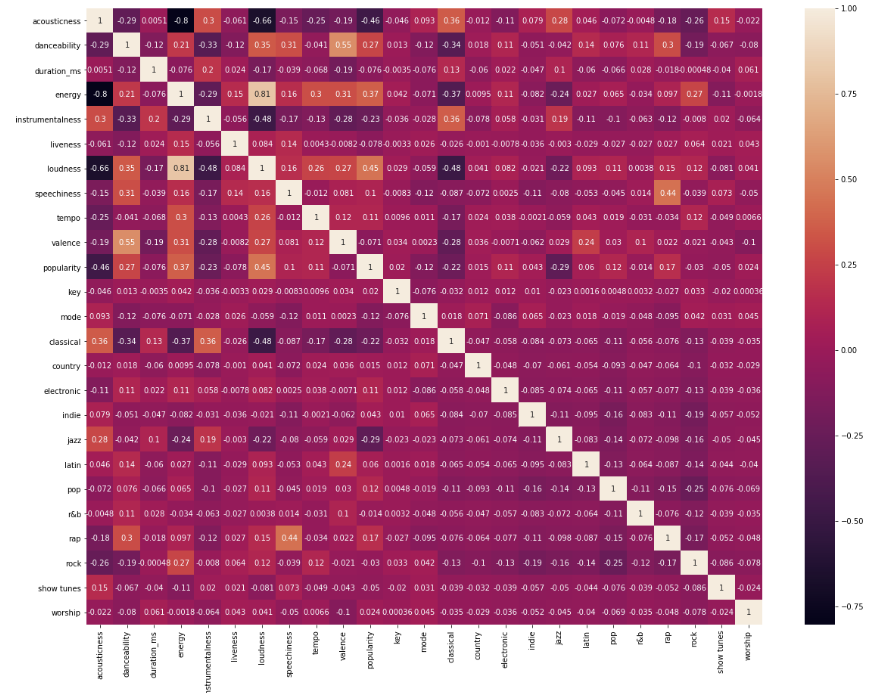


**Fig 2:** screenshot revealing dataframe with one-hot encoded columns.

# 4. Exploratory Data Analysis (EDA)

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

## 4.1 Correlations

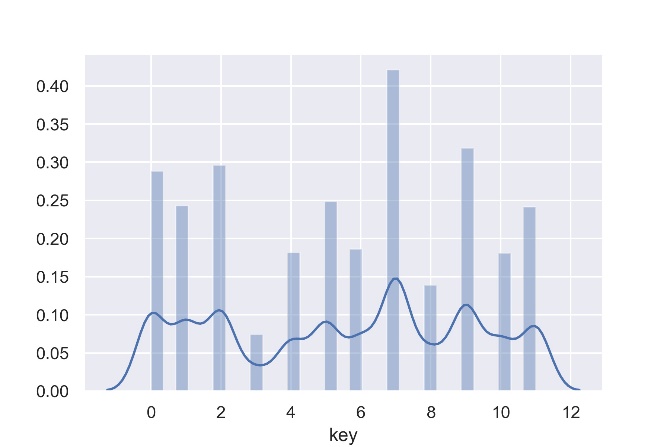
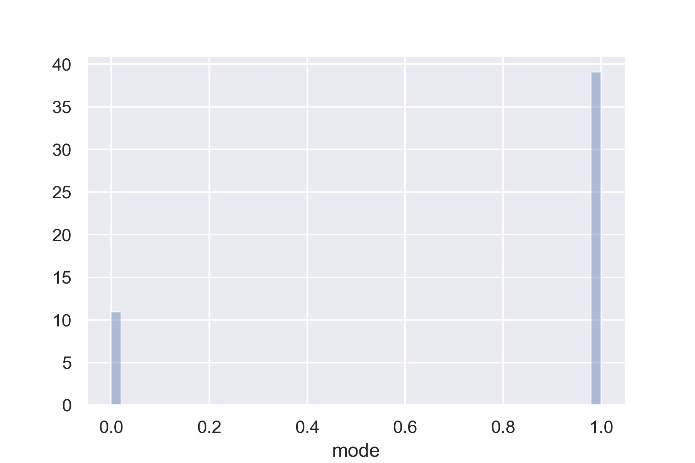


**Fig. 3:** Heatmap showing correlations between features and Genre categories

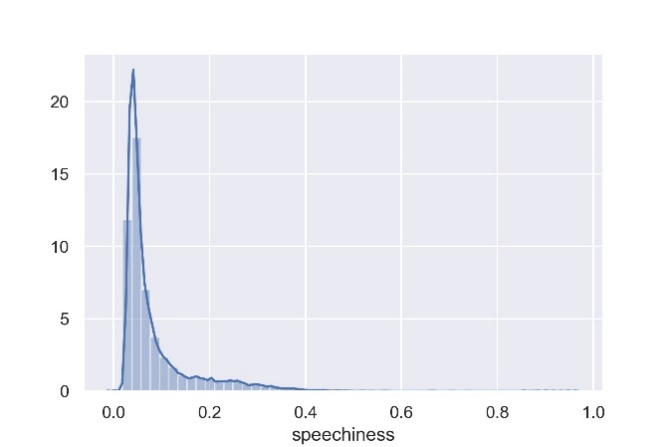
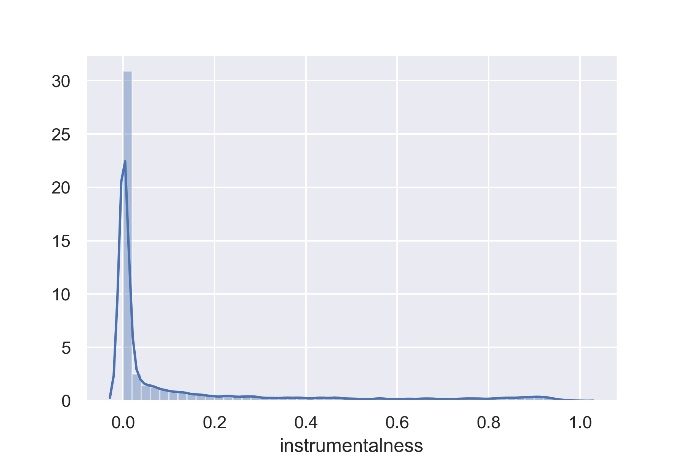
The heatmap in fig.3 visualised the correlations regarding the features within the dataset. The features that had relatively strong positive correlations with each other were valence and danceability (0.55), popularity and loudness (0.40 ), Loudness and energy (0.81), and Speechiness and rap (0.44). The reasoning behind their positive correlations lies in the meanings behind the features. Valence is the level of happiness of a song and danceable music usually tend to be happy; the loudness of a song is the overall average of the amplitude of a song in decibels and it correlated with energy because energy is intensity; Speechiness represents the presence of spoken word and rap tends to have music with a high words per minute count.

Alternatively, the features with negative correlation were acoustic & Energy (-0.8), acoustic & loudness (-0.66), acoustic & popularity (-0.46), instrumental and loudness (-0.48) and Loudness and classical (-0.48). Acoustic sounds tended to not have low energy and tended to be of low loudness; acoustic sounds were inversely correlated with popularity; Instrumental and classical music were inversely correlated to loudness.

## 4.2 Feature distribution



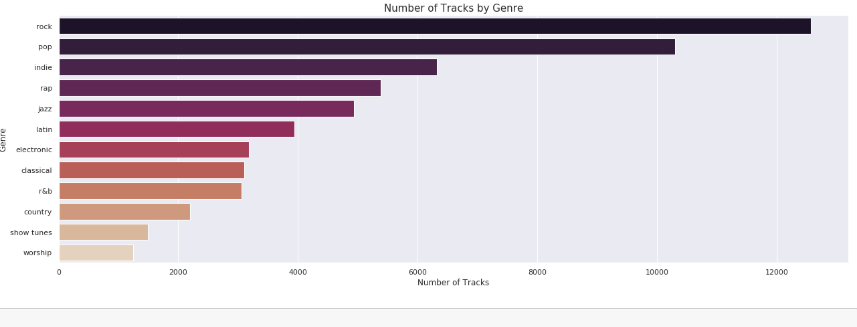
**Fig 4:** mode distribution **Fig 5:** key distribution



**Fig 6:** Instrumentals distribution  **Fig 7:** Speechiness distribution

**Fig**.4-6 and appendix E reveals the distribution of the dataset’s features. Instrumentals, duration Speechiness, liveness and acoustic were all right skewed while Loudness, energy and danceability were left skewed. The modal values were either 0 or 1—most of the data had modes of 1—and key values were relatively evenly distributed. The key with the max occurrence rate was 7 and the key with the lowest occurrence rate was 3.

## 4.3 Genre Distribution



**Fig 8:** Distribution of genre categories based on number of tracks in Dataset

Rock, pop, and indie music had the highest number of tracks in the data whereas country, show tunes, and worship had the lowest.

## 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. Moreover, we attempted cluster algorithms in attempts to improve validation results.

## 5.1 Running the Models

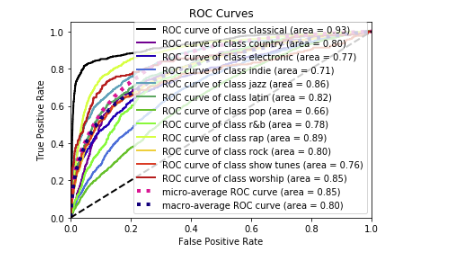
After balancing the data and splitting it into training and testing sets (The test train split for our data set was 20% vs 80% respectively) we scaled the X variables (the features we obtained from Spotify API). The features had to be scaled because there was a huge variance in magnitude between some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



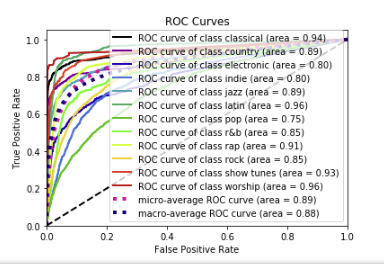
**Fig.9:** ROC curves for Logistic Regression Model.

## 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.10 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.



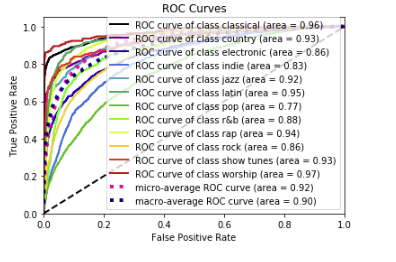
**Fig. 10:** ROC curve for Random Forest Classifier

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

## 5.1.3 XGBoosting Classifier

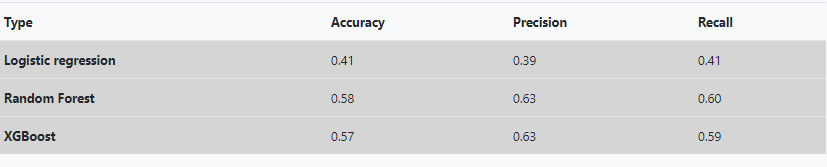
The final model that we tested was the extreme gradient boosting classifier. It is an implementation of gradient boosted decision trees designed for speed and performance [3]. Like the previous models, we fit the model to the trained data and then used it for predicting the test data. Based on the metric 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. 11:** ROC curve of XGBoost classifier

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



**Fig 12**: Metric test results for classifier models

## 5.1.4 Cluster Analysis(K-means)

The purpose of implementing a clustering models is to assist with exploratory data

analysis and to give clues about the structure of the data. For this project, the team

used clustering to give an idea of how many genre should be used in our analysis.

The original Spotify data set has over a hundred different genres/sub-genres, which

was much too broad for this project. We had to find a way to narrow our focus to a

few, select genres. How might the genres be selected and how many should be

used? We looked to K-Means Clustering Analysis for some insights.

**Methodology**

The dataset was pared down to just the 10 musical elements provided by Spotify to

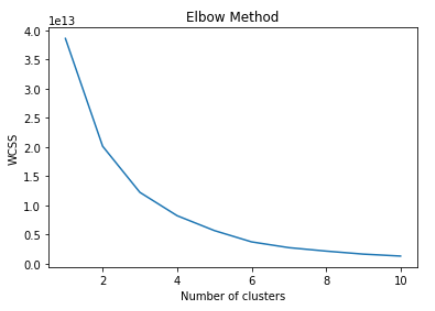
accommodate the K-Means algorithm. The K-Means algorithm only accepts

numerical information. Next the dataset was fed into the K-Means analysis

algorithm and then the following elbow plot was generated. The elbow plot is used

to determine the optimal number of clusters in k-means clustering. The bend in the

“elbow” is the most optimal cluster.



According to the K-Means analysis/elbow chart, 4 to 6 genres should have been selected. After review, the team felt that 6 genres were too small and that perhaps additional analysis was in order. Next a Silhouette Analysis was done. A Silhouette analysis is method that used to determine the separation distance between K-Means clusters. Silhouette analysis can be used to choose an optimal number of clusters. A set of Silhouette score and Silhouette plots were generated. A Silhouette score is between (-1 and 1). The closer to 1, the better the score. Negative values indicate that values are in the wrong cluster. For the plots, similar sized plots with clear separation, and no negative values are most desirable. Of the plots with the higher numbers of genres, 12 clusters were the most desirable set.

Our K-Means analysis could use to make predictions in tandem with other Machine Learning methods as a verification other method finding. One short coming of this implementation is that while it tells that we have 12 clusters, it does not tell which songs go into which category. A future implementation would allow predictions to be made to which genres a song might fit in.

# 6.Genre Predictions

# 7. App – Desktop & Mobile

To aid the average individual and a less technical management with the use of our final model, we developed a website app to provide information on the project as well as make predictions on song genres from Spotify.

The website has a home page, a page for audio features, data exploration information, Model selection methodology, results, summary, and a predictor. The predictor page allows the user to input song features and obtain a genre prediction.

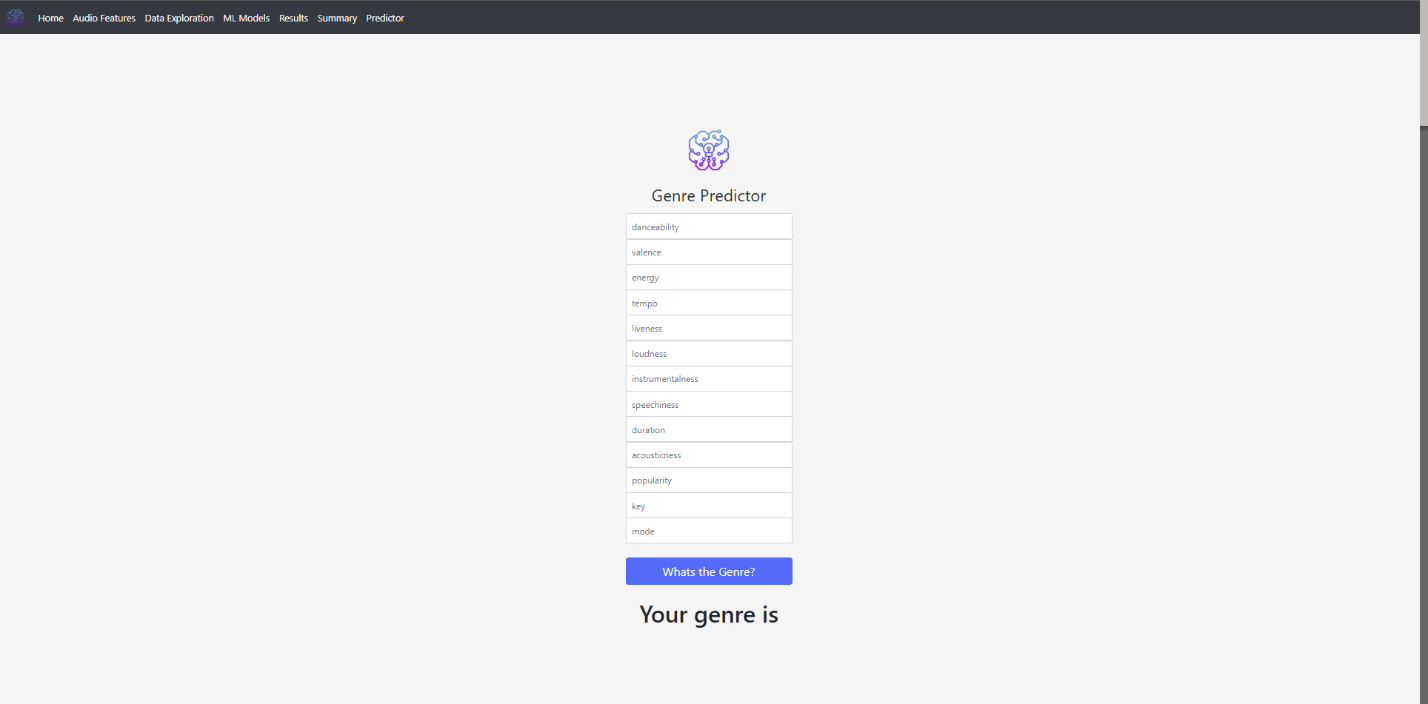


Fig 13: Predictor page from website app

# 8. Technology Resources

Table 1. below show the technology resources used in this project. All the software used for analysis were open source, therefore had zero costs and team members were able to keep track of everyone’s progress via github.

|  |  |  |
| --- | --- | --- |
| Activity | Tools | Libraries |
| Data wrangling and EDA | Python, Tableau | Pandas, NumPy, matplotlib, seaborn, |
| Model Development | Python (Anaconda, Jupyter) | Scikit-learn, pickle, XGBClassifer, matplotlib |
| Front-end Web development | HTML, CSS, JavaScript, Tableau JS API, Bootstrap | jQuery, Bootstrap |
| Back-end web development | Python | Flask |

**Table 1:** Technological resources used in creating the model and website

# 10. Limitations and Future Analysis

Once our analysis was finished and our metric tests were complete, we came to a decision that the accuracy scores could be better. These were the ideas we came up with to improve our scores:

1. Re-mapping Genre groups: The 12 genres we created were arbitrarily selected. A better option would have been to do more research on the main genres that are widely accepted worldwide and then match every other genre to their parent Genre. Based on websites like freeDB, Discogs, Wikipedia and AllMusic, we realised that the fundamental music genres that they could all agree upon are Blues, Classical, Country, Electronic, folk, Hip-hop, jazz, New age, Reggae and Rock.
2. Handling Outliers: Some songs in the data set had feature values that were outliers. We did not clean out the songs that were outliers in the dataset, and this would have had an impact in the final accuracy of our model.
3. Prediction App useability: The Final app product was a website with a prediction webpage that required the user to input values for each of the 13 features in order for the app to make a prediction. This is not user friendly because most users do not know the features of a song unless they know how to traverse the Spotify API. Useability could be improved by using a dropdown list of songs to predict rather than a long form.
4. Use Spotify API for the validation phase:

# 11. Conclusion

By the end of the project, we had used python to clean data from Spotify, performed exploratory data analysis on the dataset, created machine learning algorithms that could predict the genre of a track and finally hosted a dynamic website on Heroku. The results of our project have verified the capabilities of machine learning to perform complex tasks like music prediction and have further highlighted the benefits of tools such as python and scikit-learn library to simplify the role of a Data scientist in making these models.

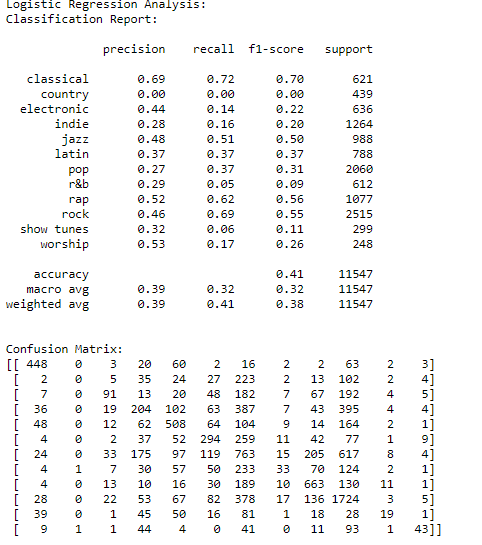
# References

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

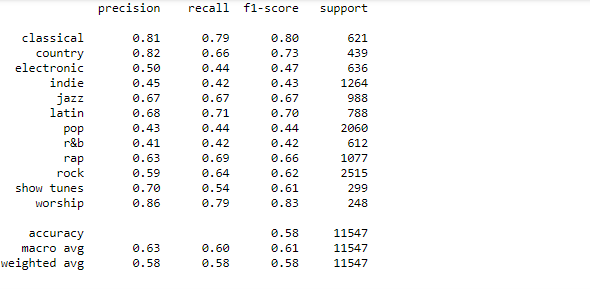
[2] <https://towardsdatascience.com/random-forest-classification-and-its-implementation-d5d840dbead0>

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

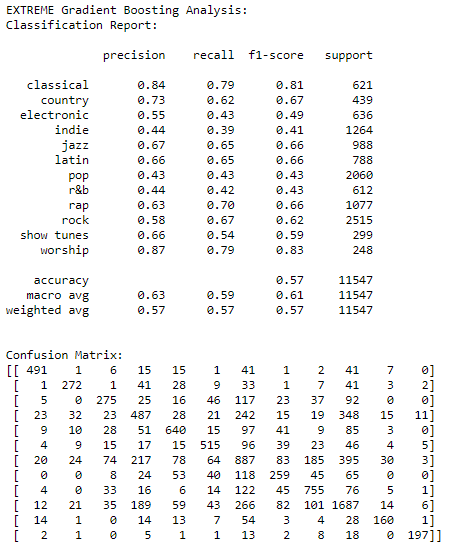
# Appendices



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

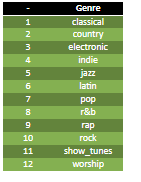


**Appendix B**: Matric test results for Random forest classifier.

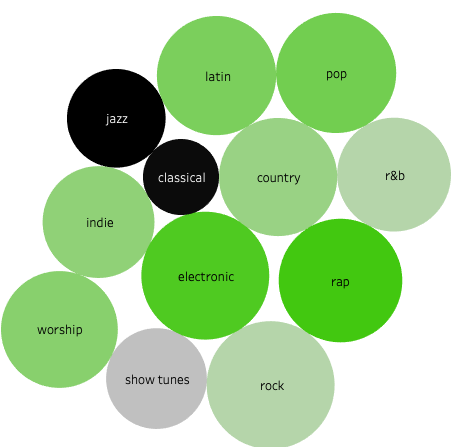


**Appendix C**: Metric test results for XGBoost classifier





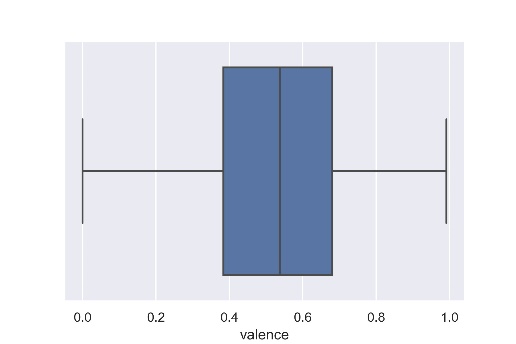
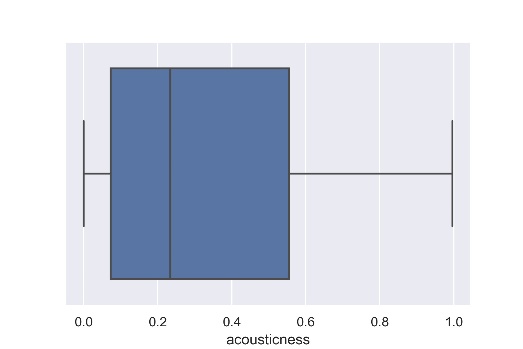
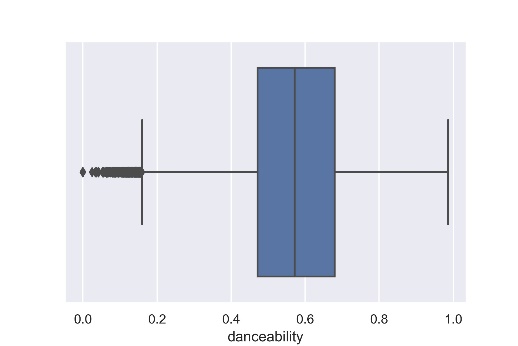
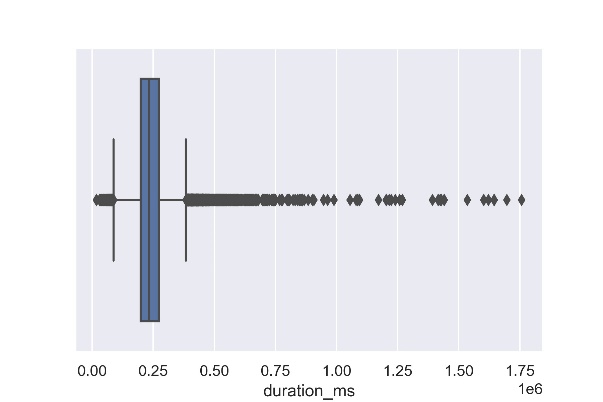
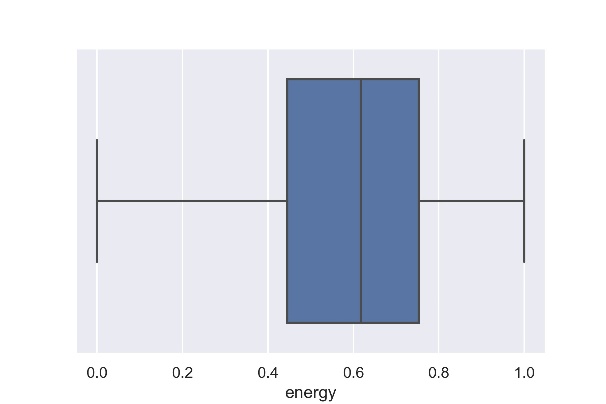
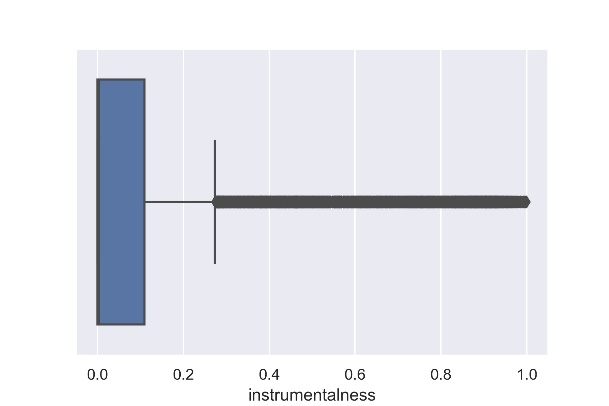
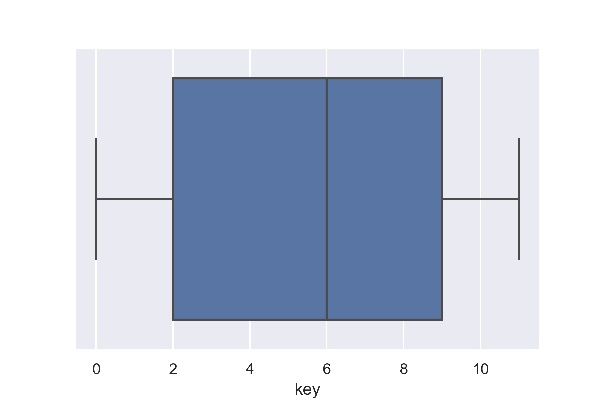
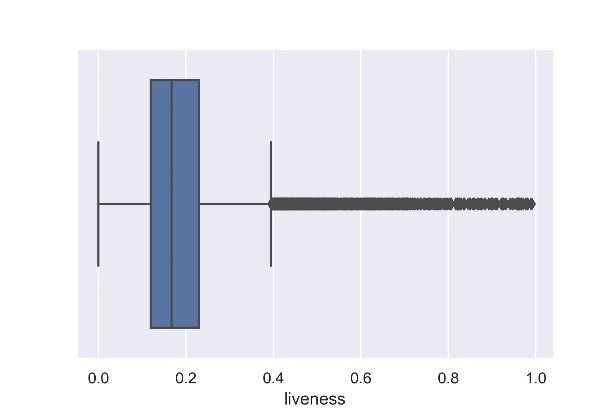
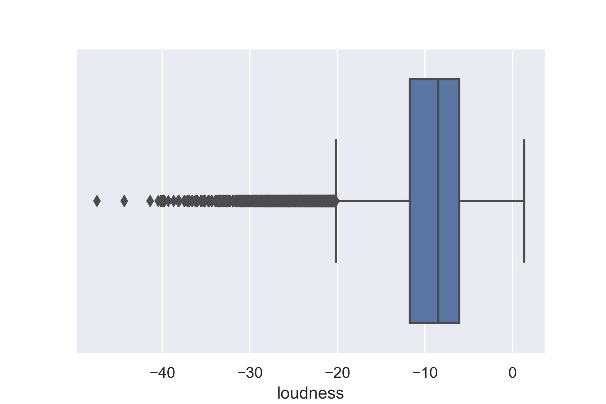
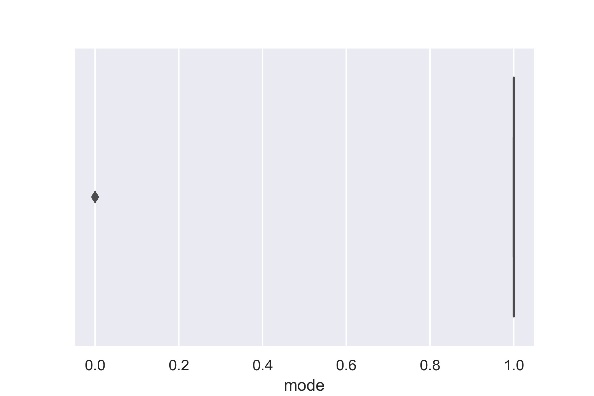
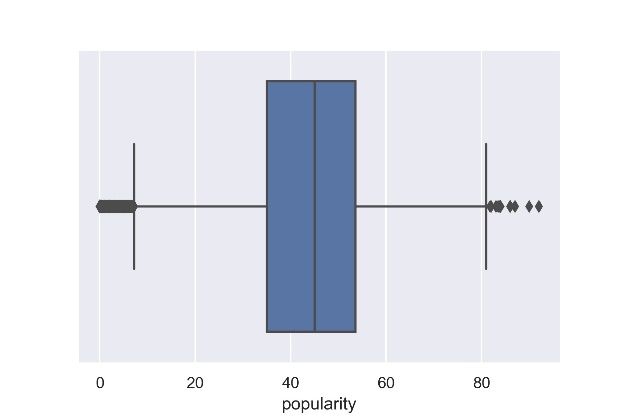
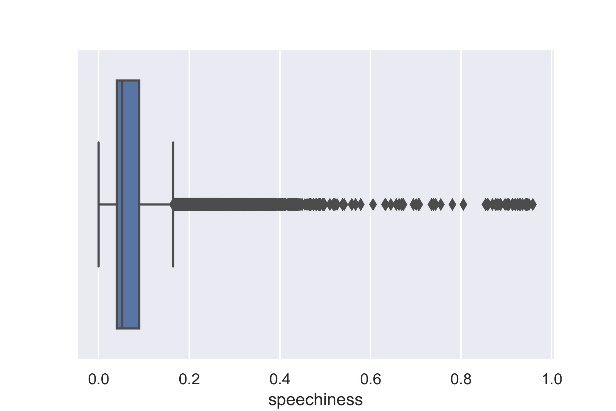
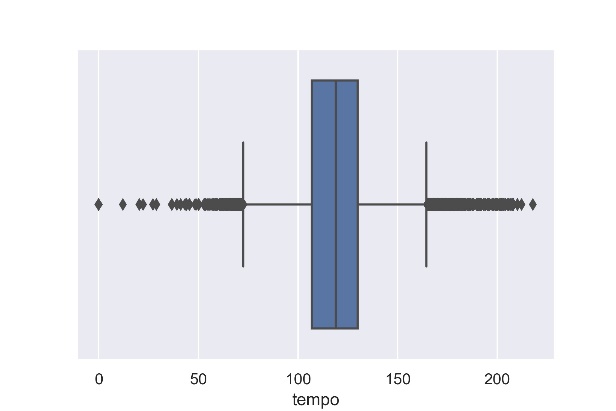
**Appendix D**: Table of feature definitions and final table of genre categories.



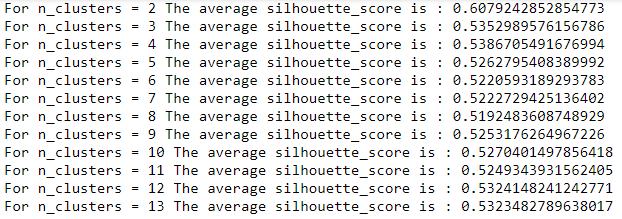
**Appendix E:** Bubble chart showing the genre distributions represented by circle size.



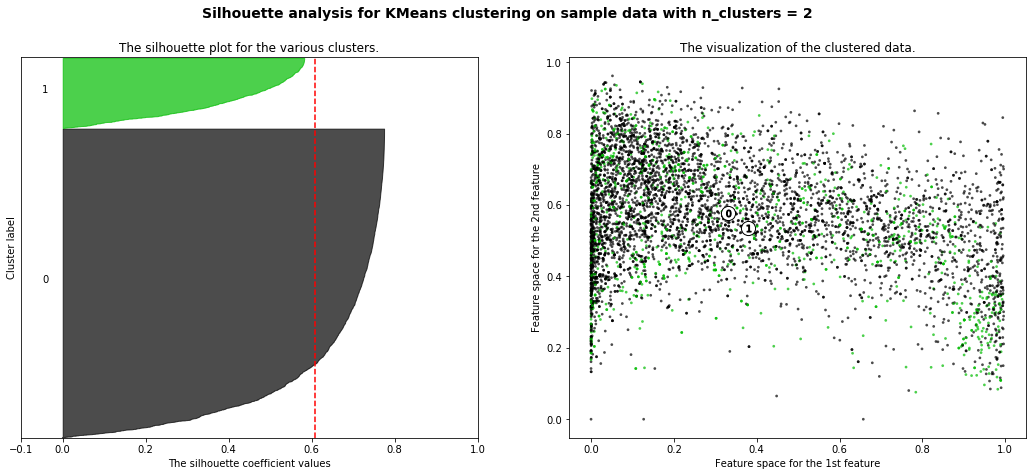
**Appendix F**: Line chart from Tableau visualising feature distribution.



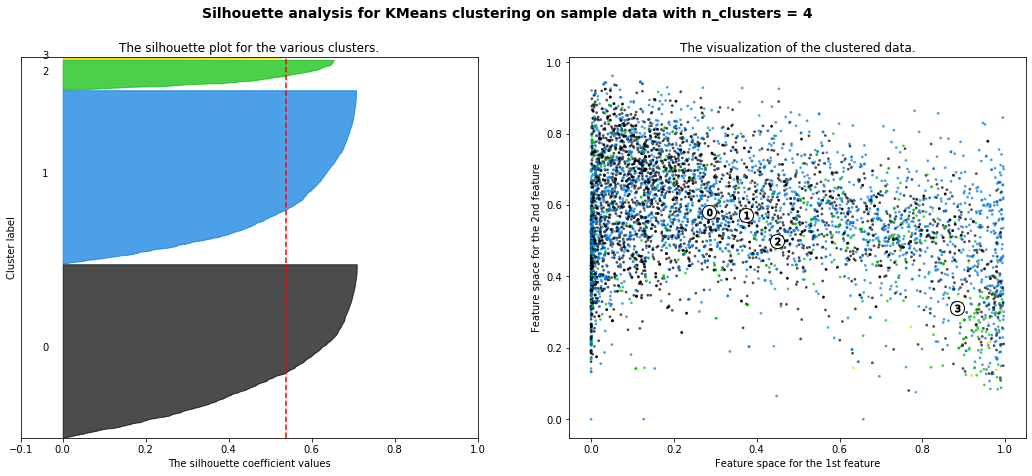
**Appendix G**: boxplot visualisations for outliers in each feature.



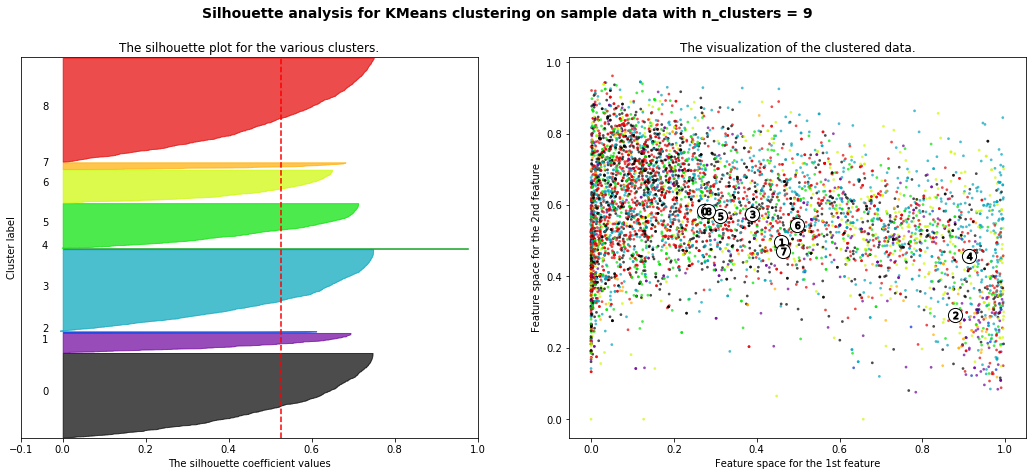
**Appendix H:** silhouette score showing the accuracy for each cluster choice



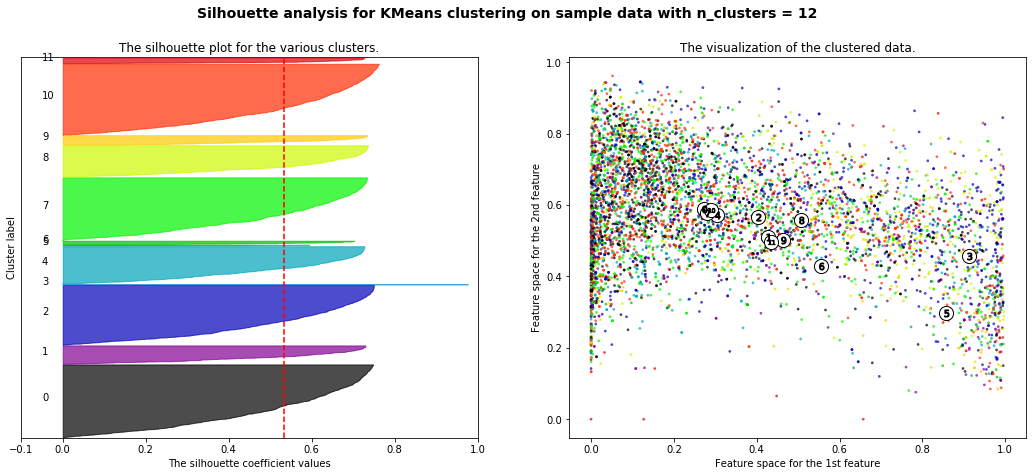
**Appendix I:** Silhouette coefficient values for 2 clusters



**Appendix J:** Silhouette values for 4 clusters



**Appendix K:** Silhouette values for 9 clusters



**Appendix L:** Silhouette values for 12 clusters