# Machine Learning - Capstone Project

Abhilasha Tandon

Wednesday, May 10th, 2023

## 1 Abstract

In this report, I will explain the model I created and the decisions I made when making it to classify genre of songs given artist name, track name, and a variety of musical characteristics such as volume, length, key, mode, and other variables. I did this using a gradient boosting method and I achieved an accuracy of 69.1% and a ROC AUC score (using the "OVR" approach) of 0.962.

## 2 Methods

### 2.1 Description of Data

The dataset in its raw form contains 18 columns: 'instance_id', 'artist_name', 'track_name', 'popularity', 'acousticness', 'danceability', 'duration_ms', 'energy', 'instrumentalness', 'key', 'liveness', 'loudness', 'mode', 'speechiness', 'tempo', 'obtained_date', 'valence', 'music_genre'. The goal of this task is to predict 'music_genre' as accurately as possible given the other variables. It is clear that 2 columns provide no use to us initially in this task: 'obtained_date' and 'instance_id'.

### 2.2 Preprocessing Numerical Data

Firstly there are several null values I dealt with immediately before processing. There are 5 fully null rows that we remove, and several songs for which tempo and duration are unknown. We remove these since they take up a relatively small portion of the dataset.

We can use many of these columns directly since they contain numerical values, but several require more forethought. Categorical variables like 'key' and 'mode' are converted into binary variables using "one hot encoding". For example the variable 'C#' is 1 if and only if the key is C#.

### 2.3 Preprocessing Non-Numerical Data

For numerical variables, we then normalize each predictor independently using z-scoring. For non-numerical variables, the only two of which are 'artist_name' and 'track_name', we use an approach taken from the field of natural language processing to convert these into meaningful vectors. We incorporate textual data such as the artist and track names by first taking each column and converting it to a tf-idf matrix, a word occurrence matrix where the value depends on frequency of a word in a title and occurrence overall in all titles (using sklearn.feature_extraction.text.TfidfVectorizer()). After removing "stopwords" (a set of commonly occurring words with little to no semantic information in English made up of pronouns, articles, etc.) we then convert these to denser but still meaningful word vectors using PCA, keeping the 20 largest components.

After this we split the data into training and test sets, with a test size of 0.1

### 2.4 Running the Model

We then run all of this data through a gradient boosting model, specifically scikit-learn's "HistGradientBoostingClassifier", with hyperparameters of a learning rate of 0.1, 31 max_leaf_nodes, and a tolerance of 1e-7. This is a gradient boosting method, which is an ensemble method that combines a number of weak decision trees together to a larger whole.

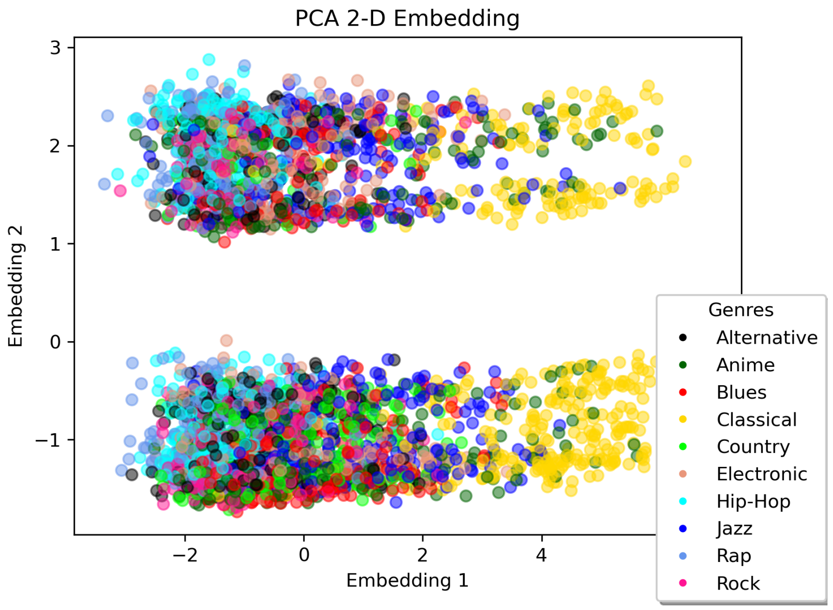
## 3 Dimension Reduction

To properly visualize the data it is useful to use some sort of dimension reduction method, to visualize patterns in our data more easily. It may also be advantageous to apply a clustering algorithm to a lower dimensional embedding as a means of classification. Here we will explore several dimension reduction algorithms to better understand our data and see if dimension reduction is a reasonable method towards classification.

All of the following embeddings were done using the form of the data used right before the gradient boosting model was applied, including all preprocessing and the NLP techniques.

### 3.1 Principal Component Analysis

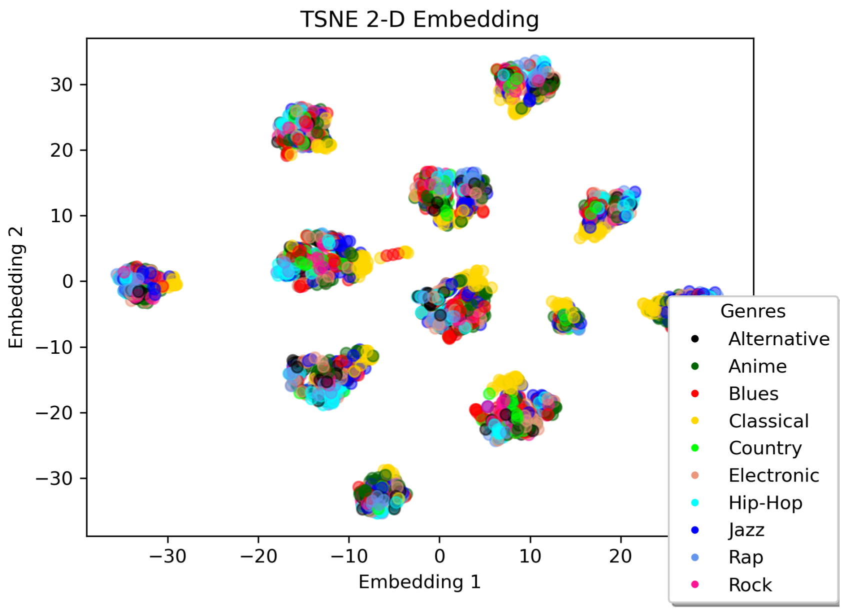
Below is a chart showing a 2-D embedding using PCA:



As we can see there are 2 clusters here, but they seem to be unrelated to the music genre. The horizontal principal component seems to be correlated somewhat with genre, but it is still not a clean fit and applying a clustering method to this embedding is likely to lead to poor results.

### 3.2 T-Distributed Stochastic Neighbor Embedding

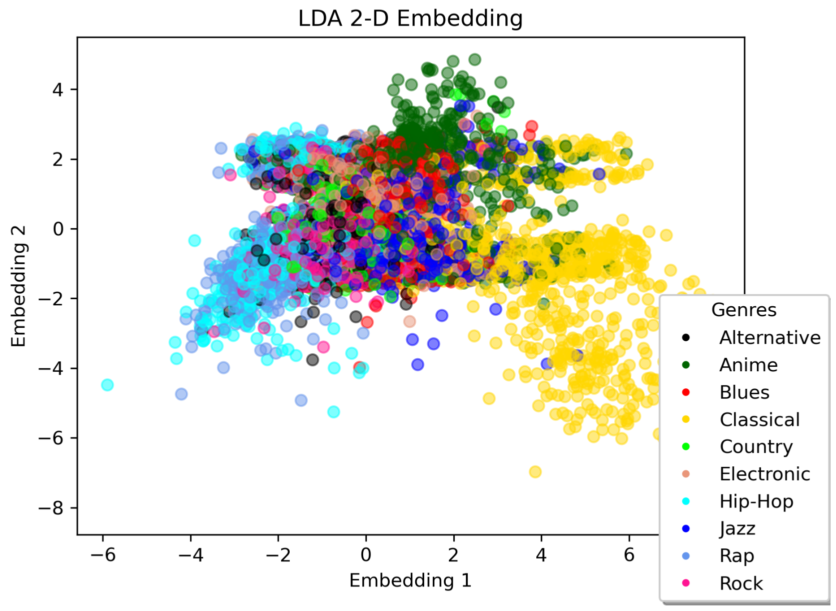
Below is a chart showing a 2-D embedding using T-SNE, using a relatively high perplexity of 200:



One notable feature of this chart is that there appear to be easily identifiable clusters, yet these seem to be made up of data points from all music genres. This is interesting and deserves investigation, but is outside the goals of this project. As we can see there is little correlation between cluster and music genre, so this method is unsuitable for classification after applying clustering.

## 3.3 Linear Discriminant Analysis

While the previous two methods were unsupervised, there is a dimension reduction technique known as LDA that is supervised, which is useful for dimensionally reducing data to maximize class separation. Below is a chart showing a 2D embedding with LDA:

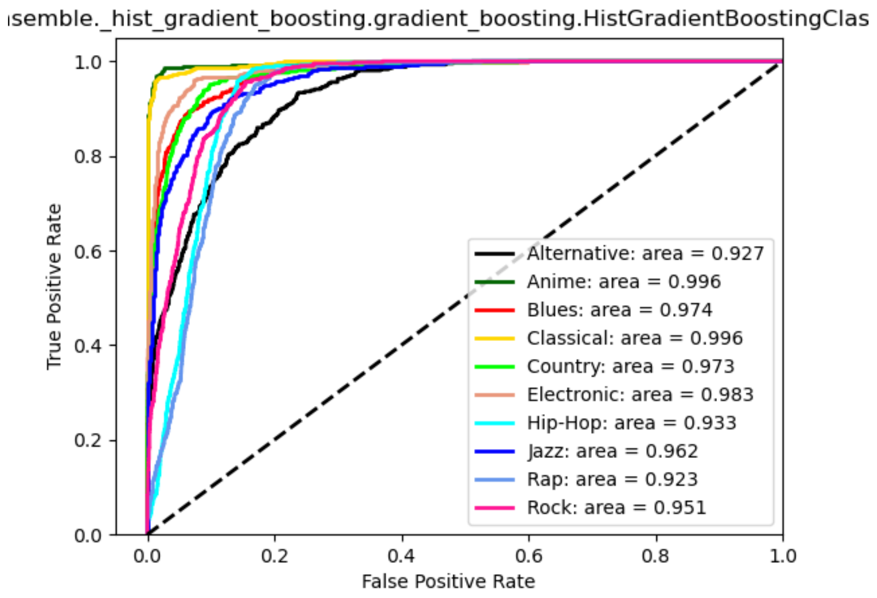


As might be expected, this embedding has the best visual class separation of all 3 dimension reduction techniques, but most classes still are mixed together. Only 'Anime' and 'Classical' seem to be partially separable, the rest are mostly mixed together.

Thus dimension reduction followed by clustering is likely to not lead to promising results with classification.

## 4 Results of Model

In making this model, I tried a number of different algorithms, an SVM method, an AdaBoost method, a Random Forest method, but a Gradient Boosting method performed the best. This method achieved an accuracy of 69.1%, and an ROC AUC score (using the "one versus rest" approach and averaging over all classes) of 0.962. Below is a graph showing the ROC curves for each class.



Below is a chart showing the precision, recall, and f1-scores for each music genre:

As you can see, the model is much better at classifying certain genres than others. It is best at correctly classifying 'Classical' and 'Anime' genres, and worst at 'Rap' and 'Hip-Hop'.

| Genre | precision | recall | f1-score |
|---|---|---|---|
| Alternative | 0.57 | 0.57 | 0.57 |
| Anime | 0.95 | 0.92 | 0.93 |
| Blues | 0.77 | 0.77 | 0.77 |
| Classical | 0.95 | 0.92 | 0.93 |
| Country | 0.76 | 0.72 | 0.74 |
| Electronic | 0.81 | 0.82 | 0.82 |
| Hip-Hop | 0.43 | 0.44 | 0.44 |
| Jazz | 0.74 | 0.70 | 0.72 |
| Rap | 0.41 | 0.40 | 0.40 |
| Rock | 0.56 | 0.66 | 0.61 |

## 5 Conclusion

A very significant aspect to the performance of the model was the incorporation of NLP features. Adding these led to a large improvement in the accuracy of the model. A basic linear SVM model had about 50% accuracy, while the gradient boosting model without the NLP attributes had 59% accuracy. We can then see that adding attributes for title and artist improved the accuracy by about 10%.