Classification Capstone Project

Janya Mirpuri (jm8808)

College of Arts and Sciences, New York University

Fundamentals of Machine Learning

Professor Pascal Wallisch

May 10, 2023

1. Data Pre-Processing

It was first necessary to handle missing values: scattered throughout the original data, there are question marks, exclamation points, and NaN values indicating data that needs to be treated. In general, if there was a missing value in any row, that row was dropped. The only exception to this was if the missing value was in the "duration ms" column (dealt with later in the cleaning process). For the music genre, I treat it by just mapping them to integers from 1-10. The mode column is already binary, so I just replaced the "Major" values with 1, and "Minor" values with 0. I split the key column into two: one indicating what the letter to the key was, and the other indicating if the song was in a sharp key or not (binary valued). For the former column, I one-hot encoded it into columns A-G.

At this point, I split the data into training and testing sets. To split the data, I want to ensure that the validation set has a sufficient number of instances *per class*. I iterate over the classes: for each class, I retrieve 10% of the data points randomly, and store it in the validation[1] set, and the rest are a part of the training set[2]. Now, I handle missing values in the "duration ms" column. From the data in the training set, I took an average of the "duration ms" values, and used that to impute the missing values in **both** sets. The difference between doing this, and just taking an average of the entire column before a train-test split is that this allows the training set to be "pure" from the test set, avoiding any possibility of data leakage - in theory, when the model is trained on the training set, we have no indication of "future" (aka validation) data, so the training data should not be affected by the testing set whatsoever. In the same way, when I z-score the continuous data columns, I do it separately for the training set and the testing set. Finally, since

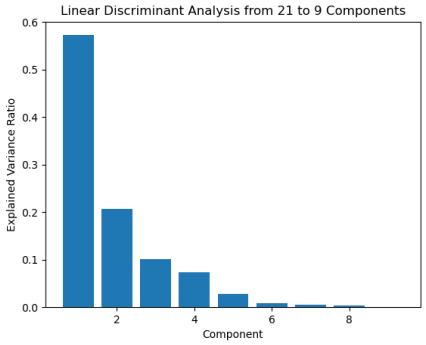
[1] Testing and Validation sets are used interchangeably

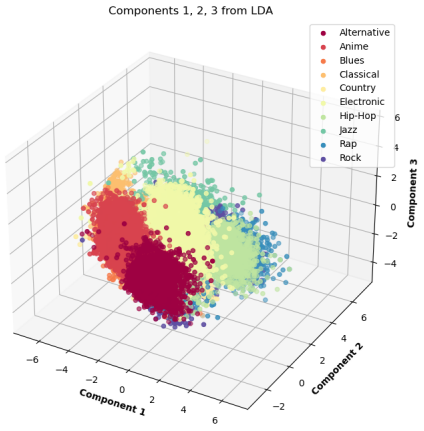
[2] The classification assignment said to do 500 randomly picked songs for the test set and the other 4500 songs from that genre for the training set, however after data cleaning, there are not exactly that many songs per genre. Allowing a 10%-90% split maintains the same theory.

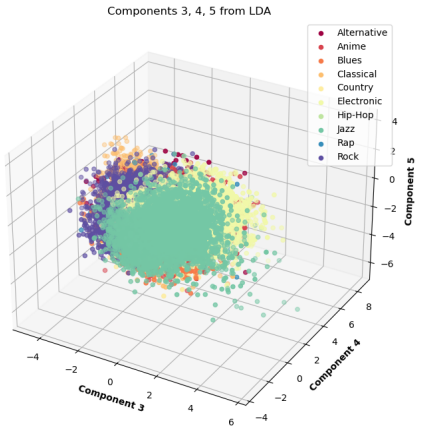
most artists have an associated genre of music that they perform, I didn't want to just drop the artist name completely. Instead, using the training data alone, I found the genre most associated with the artist (out of all the songs by the artist in the training dataset, what the most common genre was), and used that to create a map that for a given artist name, an associated genre could be found. That map was used to create a new column of data.

2. Dimensional Reduction and Clustering

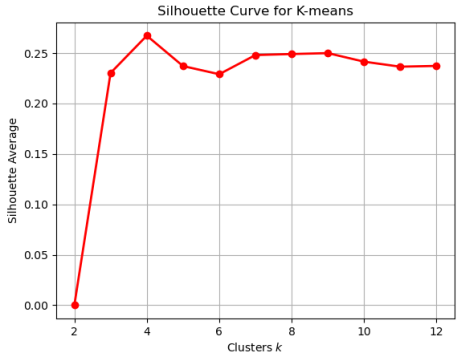
Since this is a classification problem, I used a Linear Discriminant Analysis. From the resulting explained variance ratios for each associated component, it seems reasonable to consider only the first 5 components, as the others had little to no contribution to the variance. From the LDA then, I effectively reduced the data from 21 dimensions to 5.

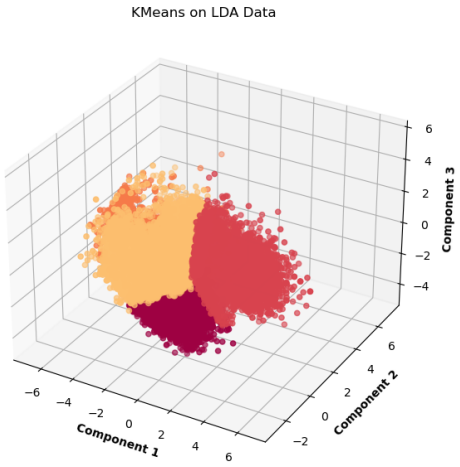


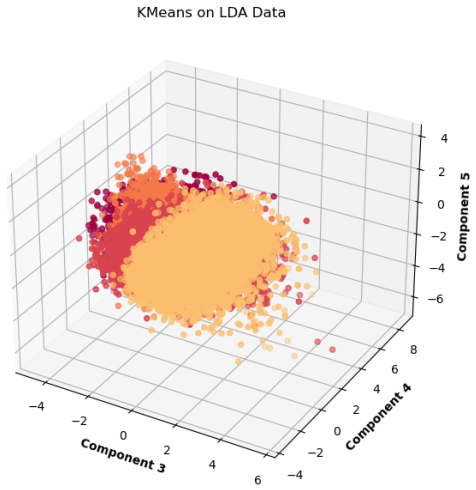




With the first 5 components from LDA, I wanted to use K-Means to cluster the data. I used silhouette scores to identify what the best number of clusters were. Ultimately, I chose 4. I felt that this clustering provides some indication of what genre a song belonged to as there is some visual overlap between the clustering and the true data such that we might be able to identify 5-6 genres that a song could possibly be in. However, that is not a strong upper bound of genres.







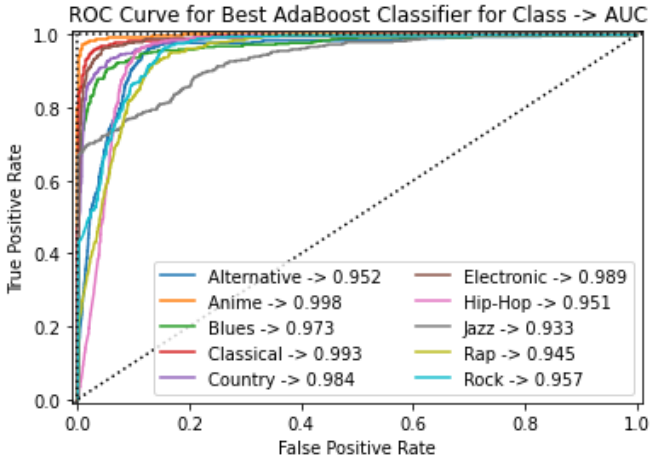
3. Model Development and Analysis of Results

Since the model being created is adaBoost based on decision trees, there are 3 hyperparameters that can be tuned: max depth, number of estimators, and the learning rate. I also wanted to run my models on the original data and the 5-dimensional data from LDA. I tested 144

different models overall. 72 for each of the data sets. To compare them, I recorded a few different metrics. For each class, I calculated the corresponding F1 scores and AUC scores, and took the mean over all the classes to get the Average F1 score, and Average AUC as metrics to represent the model's performance. I also kept track of the Accuracy of each model. Generally speaking, if we compare models using the average AUC score, the models on the original dataset did better than the models trained on the LDA data. The following table summarizes the best models from my hyperparameter tuning[3] based on Average AUC.

Dataset Type	Max Depth	Number of Estimators	Learning Rate	Average F1	Average AUC	Accuracy
Original	4	47	0.3	0.76202	0.96745	0.76343
LDA	4	61	0.3	0.63703	0.93422	0.63626

Given that the goal of this project was to maximize[4] the AUC score of the model created, the objective model of choice would be using the original data to develop the adaBoost, using hyperparameters max_depth = 4, number of estimators = 47, learning rate = 0.3.



[3] The full list of results can be found under the variable "res" within my code.
[4] It should be noted that, in practice, it is not always smart to choose a model based on a single scoring metric (such as Average AUC). This is because the model may be doing well purely due to overfitting, or that one score alone may not be a good representation of the model's performance on a whole.

4. Extra Credit

For this, I created a correlation matrix. I saw two patterns that really stood out. Energy and loudness were very negatively correlated with acousticness, instrumentalness and loudness were negatively correlated, and energy and loudness were positively correlated. I find this interesting because on one hand, it reflects a lot of patterns current music has, from my understanding, but I am also shocked to see that there aren't stronger correlations between a few pairs of columns, like energy and tempo. Furthermore, I think that the correlations shown below are not accurate, on a higher level outside of our dataset, considering the fact that in theory, you could probably make a song that has any combination of values for these columns, songs in general are not technically bound to the correlations seen before.