Capstone Project Introduction to Machine Learning
Classification

In this classification project, I try to use the features of the music to predict the genres to which they correspond. The available features I make use of are the numeric features, including popularity, acousticness, danceability, etc. There are 10 genres into which we want to classify the music.

What I have done and why I did these:

I. Loading and pre-handling the data set.

1. Since some of the features in the dataset are not numeric, and from intuition, I suppose they are important for predicting the genres, for example, key and mode, I convert them into numeric values from strings. There are 12 keys in total for each mode, and modes that are major and minor. The first thing I do was to convert them into numeric columns.
2. The second thing I do was to drop NaN from the dataset for later training or clustering.

II. Clustering

1. I then extracted the numeric data columns and form X. We first do a dimensionality reduction on X and a general clustering for us to preview the data. The method I use for dimensionality reduction is t-SNE since this method can adapt to non-linear conditions, and generally has better performance than PCA. Note that standardization is accomplished before dimensionality reduction.
2. I set the component number after dimensionality reduction to 2, which is clearer for visualization on a 2D graph.
3. After dimensionality reduction, based on Silhouette score, I tried to decide on the number clusters which is optimal for KMeans clustering.

III. Classification

1. One challenge is to convert the music genres into several dummy variables which we can use as labels for later classification. Here get_dummies() function is used in this process. Thus, Y is a matrix containing 10 columns, and each stand for a genre.
2. Train_test_split(): as it is suggested in the spec sheet, “for each genre, use 500 randomly picked songs for the test set and the other 4500 songs from that genre for the training set”.
3. Then I begin to set up models for classification. In this part, what I did was
    a. using the ‘original’ data without dimensionality reduction
        i. I set up a neural network model
        ii. I set up a random forest model
        iii. I set up an SVC model, but this does not converge at last, so I abandoned it.

b. Using data after dimensionality reduction for classification

   i. Without cluster labels added to X

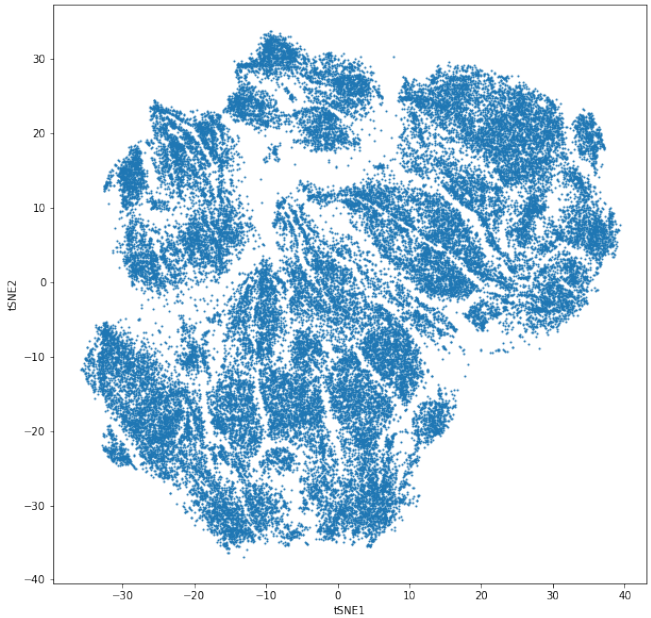
   ii. With cluster labels added to X

   iii. Try other clustering methods to implement the model better

4. To visualize the classification performance, I plot the ROC curves and compute the AUC scores for each class in Y.

The findings and conclusions:

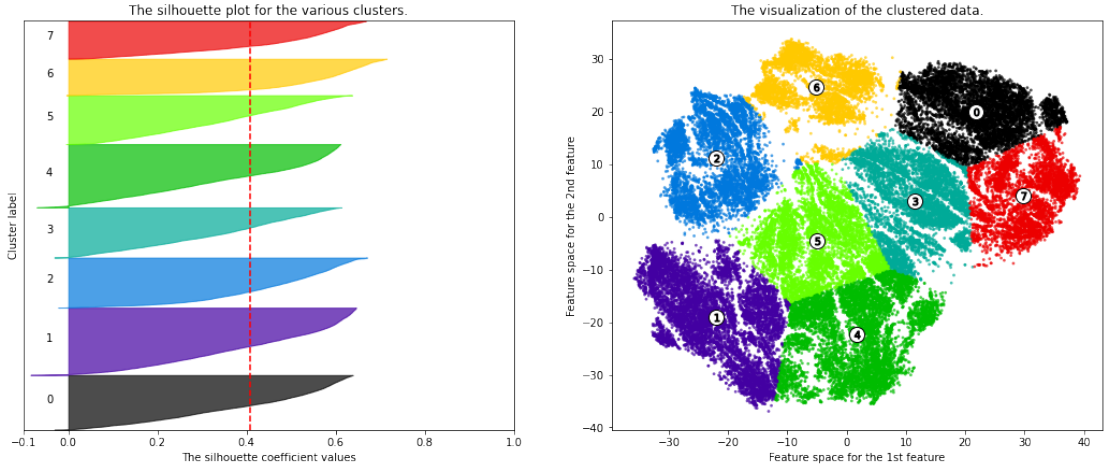
1. Below is the 2D graph generated by plotting the result of t-SNE,



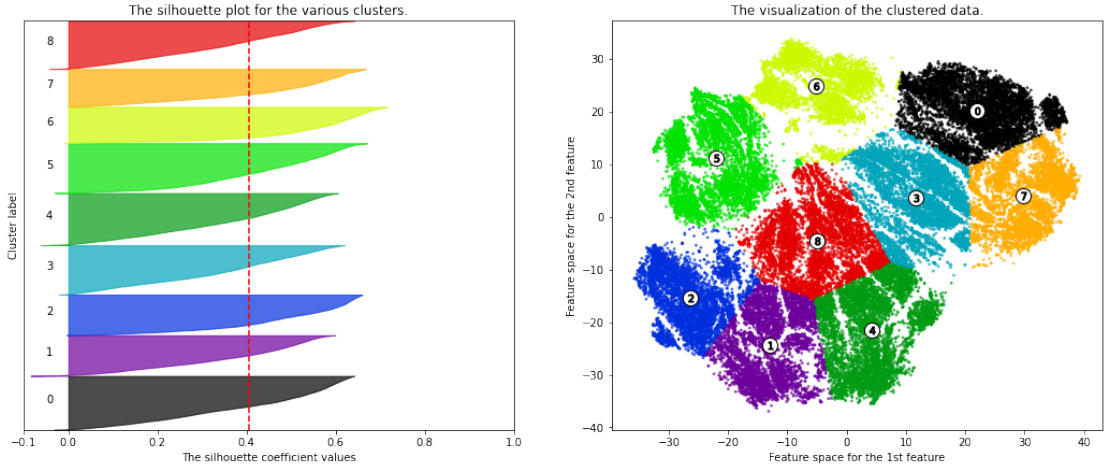
and the result of Silhouette score

For n_clusters = 8 The average silhouette_score is: 0.40809482
For n_clusters = 9 The average silhouette_score is: 0.40595207
For n_clusters = 10 The average silhouette_score is: 0.41391116
For n_clusters = 11 The average silhouette_score is: 0.4220457
For n_clusters = 12 The average silhouette_score is: 0.4127515

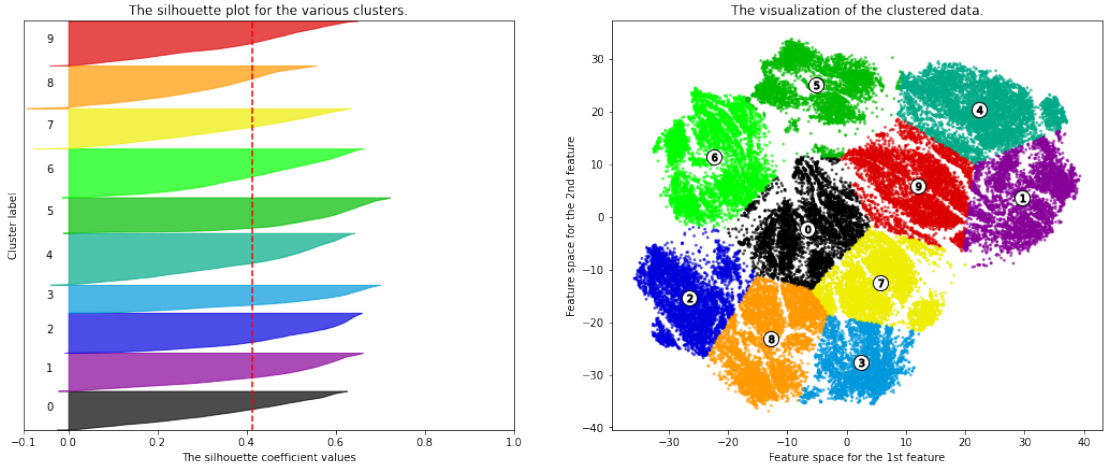
Silhouette analysis for KMeans clustering on sample data with n_clusters = 8

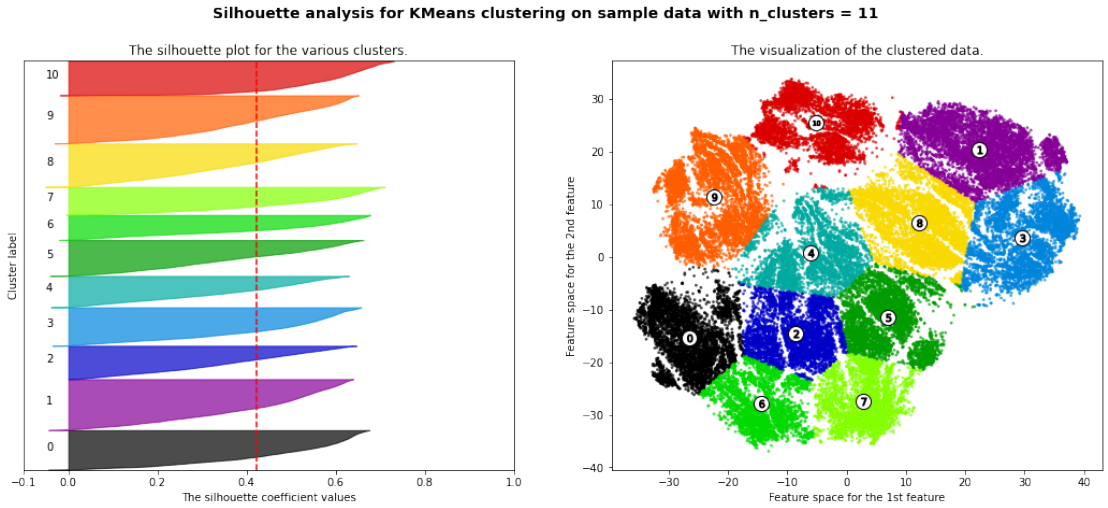


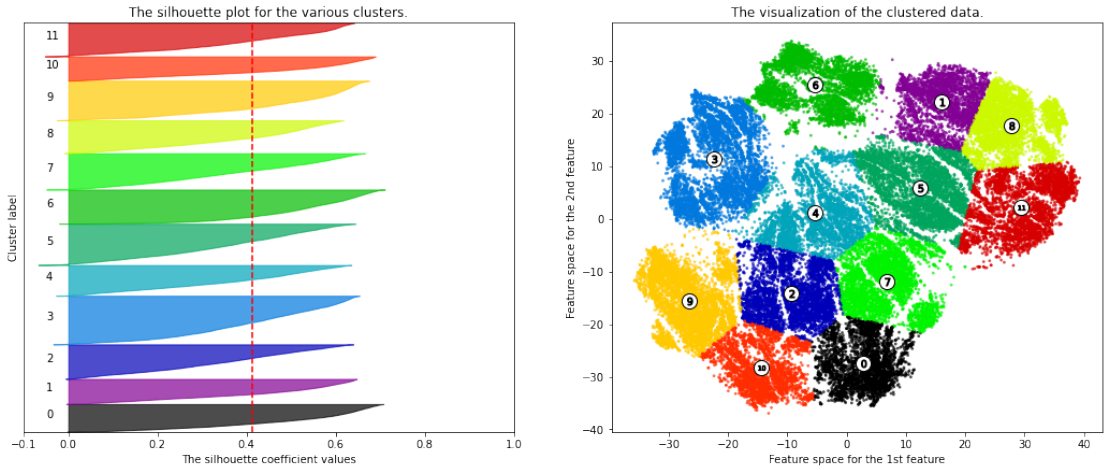
Silhouette analysis for KMeans clustering on sample data with n_clusters = 9



Silhouette analysis for KMeans clustering on sample data with n_clusters = 10

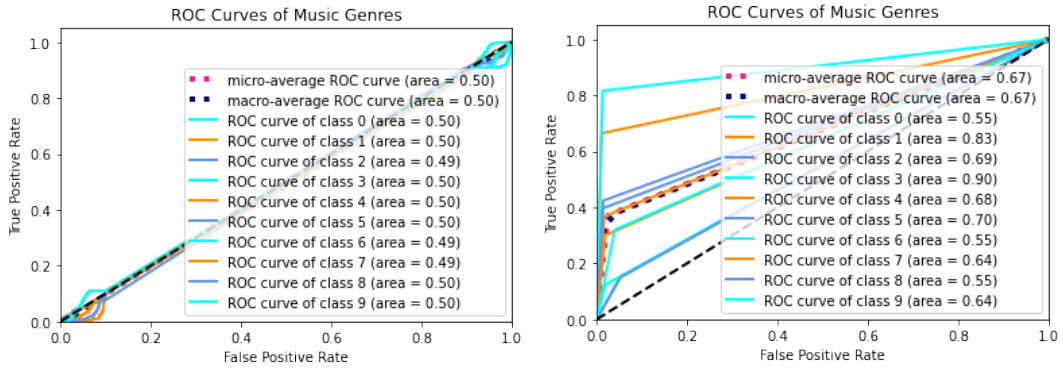






After dimensionality reduction, we can see from the result above that, n_cluster around 10 is an ideal number of categories for classification.

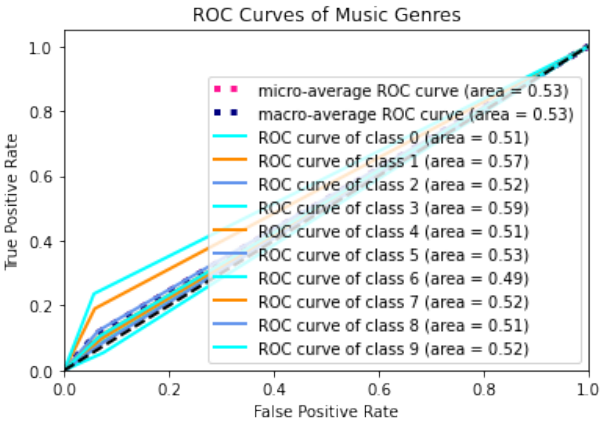
2. Below are the ROC curves from neural network model and random forest



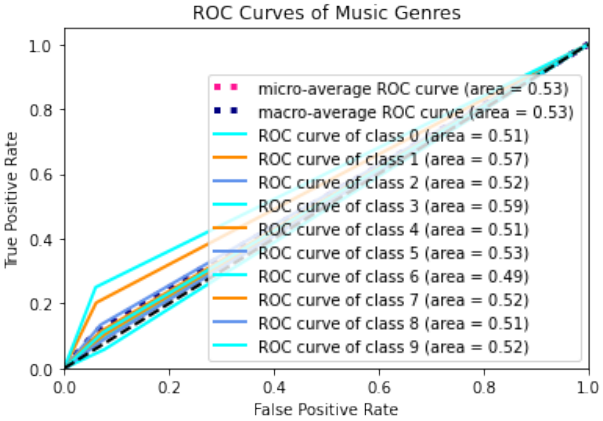
In the neural network, I used 2 hidden layers with ReLU and Sigmoid activation function, since from the plot after dimensionality reduction that the boundary between the categories might not be linear.

Here we can see that the random forest has better performance than the neural network.

Additionally, if we use the results after dimensionality reduction as the base of classification, here is the ROC curves of a random forest model.

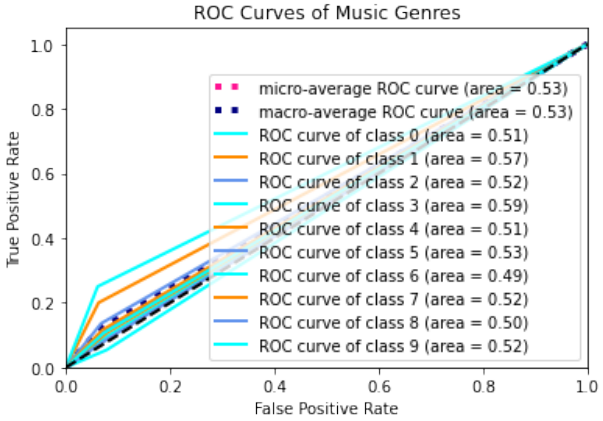


We can see that the result is not very ideal, not even as good as the random forest using the original data. Thus, we may conclude that the drawback of dimensionality reduction into 3D (considering 2D is not very ideal visually) outweigh the benefits. We did not get rid of collinearity or other bad sides of the original model but only ignored the deciding features. After clustering we get the labels for each cluster, another way is to add the cluster labels into X to see if the classification works better. The results are as follows.



This indicates that the clustering isn't effective.

I changed the clustering method into DBSCAN since from the graph of KMeans clustering doesn't show the boundary as our intuition. Below is the result.



To sum up, the most effective model is the random forest model using data before dimensionality reduction and without adding clustering results.