Introduction

In this report, I would be elaborating on my experience and process of working on the final project for the Programming for Data Science class.

First, let me introduce what I will be working on. My topic of choice is:

Given music from only two genres, can you predict the genre of music from using Spotify data.

The genres I chose to compare data of is **piano classical** and **rock** music genres (yes you really cannot go further away on the music spectrum than that). My machine learning methods of choice are Principle Component Analysis paired with Random Forest, and Decision Trees.

Data Collection

For Data Collection, I used most of the tools offered from "Lab 10 - Spotify". First I needed to find playlists containing piano classical music, and another containing rock music. Once I found the playlist for piano classical music, I ran the the while loop code that prints all of the user's playlists as well as their URIs and realised that the same user has a playlists for rock music too! So I just used the URIs for two of those playlists.

This was followed by using `get_playlist_URIs()` function to get URIs for all the tracks in the playlists, followed by `get_audio_features()` function to get dictionaries of features of all the tracks. I then concatenated all this data into a data frame and exported all the data into a CSV. My final CSV contained 464 data points with 10 columns, 8 of which are features I am hoping to analyse.

Analysis

Principle Component Analysis with Random Forests

My motivation for pairing PCA with Random Forests because random forests randomly select features to build multiple decisions trees from and then average the results. Thus, lowering the number of features to consider would've made Random Forests more efficient, especially since the components not considered hold little predictive value.

Writing Script

So the features I want to analyse are accousticness, danceability, energy, instrumentality, liveness, loudness, speechiness, and tempo. So the first thing I did was split them into `X` and `Y` data frames, where `X` held all the features of the songs, and `Y` contained the genre of the tracks.

Since PCA is not scale invariant, I standardised the data points using the `StandardScaler()` imported from `sklearn.preprocessing`. This is followed by splitting the data into training and test data using `train_test_split` imported from `sklearn.model_selection`. Originally, when I was writing all of my code for PCA processing, I forgot about splitting the data into test sets and trainning sets. Once I was done with my PCA categorisation, I realised I had no test set to test my predicitions on... So I had to go back, split my data, then go through my code again transforming my test set as well.

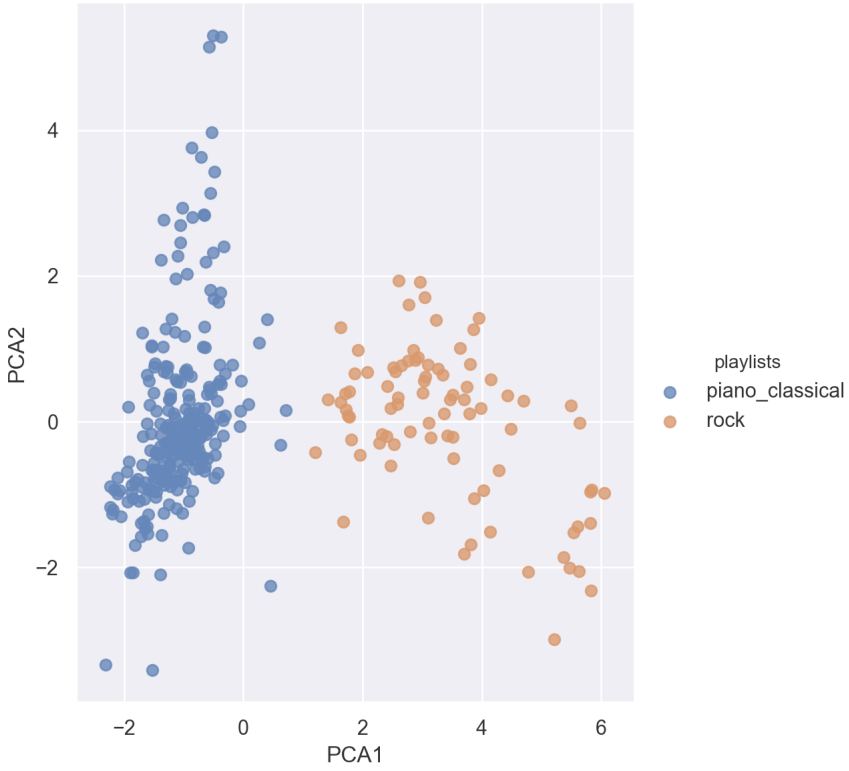
Once I standardised my data points, I transformed them using PCA. I encoded the program to keep 3 PCA components to lower number of dimensions of the data set. Next, I wanted to visually plot them. Note that to plot, I needed a data frame that contains the column of track's genres, as well as PCA1 and PCA2 (I only wanted to plot the first 2). But data frame X only contains the features, and if X contains the playlists categories, I would get errors when trying to standardise the data set. So I had to go back and rewrite some of the code again.

So after going through the script changing variable names couple of times, I finally started working on predictions. For my predictions, I'm using `RandomForestClassifier()`. Because I have changed my variable names so many times, I got confused and fit the `RandomForestClassifier()` using standardised data set that hasn't been transformed using PCA, and predicted for the standardised test data set as well. As my predictions results were fairly accurate between predicted genre of a song and its actual genre, I didn't notice this mistake until a while later. The python script exports my final predicitions to another CSV file.

I was then curious about the program predicting the genre of songs that I know. So I collected data for the Immigrant Song by Led Zeppelin, and Bohemian Rhapsody by Queen. After standardising the data and transforming it using the same PCA parameters from the training data, I predicted the genre of these two song using the same prediction fit parameters as the standardised training data (instead of the PCA transformed training data as metioned above). Once my program predicted that these two songs are Piano Classicals... I double checked my code for any errors. So I managed to catch my error of predicting on standardised data that hasn't been transformed using PCA.

Result Analysis

When plotting the first two PCA results, I would get Graph A. This graph is not deterministic, ie. there are variations in the graph after each run.



Graph A

We can already see a clear distinction between the two genres just along PCA1. From running `model.explained_variance_ratio_`, on average, the first PCA already explains around 52% of the variance, the second one explains around 18%, and the third around 9%.

With these components, predicting using the `RandomForestClassifier()` then predicts with the accuracy of, on average, 0.99 (where 1 is the maximum accuracy score).

	Actual	Predicted
0	piano_classical	piano_classical
1	piano_classical	piano_classical
2	rock	rock
3	piano_classical	piano_classical
4	piano_classical	piano_classical
5	piano_classical	piano_classical
6	piano_classical	piano_classical
7	rock	rock
8	piano_classical	piano_classical
9	piano_classical	piano_classical
10	rock	rock
11	piano_classical	piano_classical
12	piano_classical	piano_classical
13	piano_classical	piano_classical
14	piano_classical	piano_classical
15	piano_classical	piano_classical
16	piano_classical	piano_classical
17	rock	rock
18	piano_classical	piano_classical
19	rock	rock
20	piano_classical	piano_classical
21	piano_classical	piano_classical
22	piano_classical	piano_classical
23	piano_classical	piano_classical

Result of the first 23 tracks in the test set

And finally, the program was able to predict that Immigrant Song and Bohemian Rhapsody are rock tracks.

Decision Trees

Unsupervised learning method PCA allows us to pick components with the highest predictive value. However, these components lack a label and the user is not able to understand which features of these songs are most productive in helping the program predict the classifications. So, I wanted to know out of the 8 features I picked, which ones reveal the most information about which genre the song is. Thus, I decided to use Decision Trees.

Writing Script

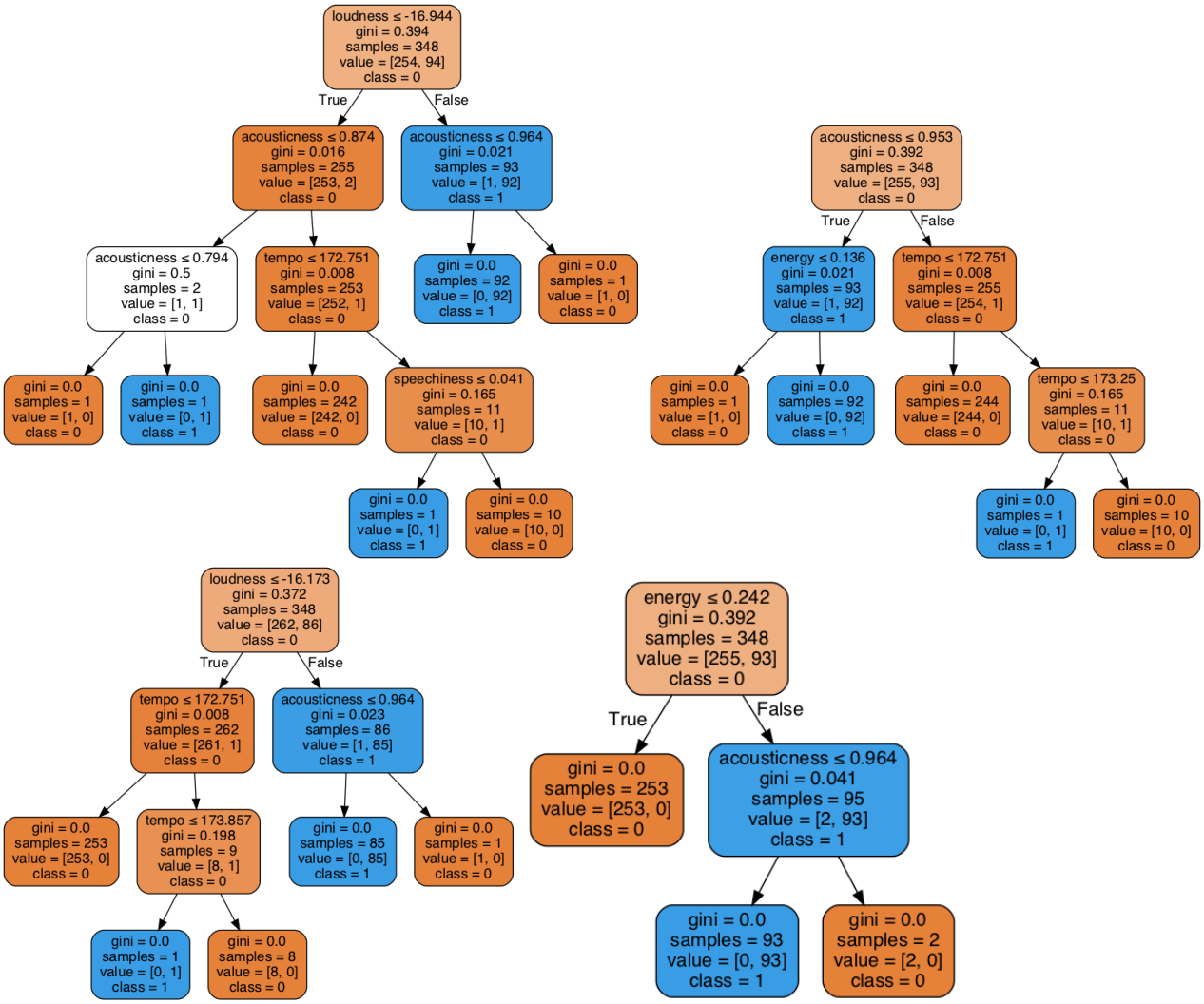
Since I already used Random Forests in the previous script and `RandomForestClassifier()` and `DecisionTreeClassifier()` use similar syntax and functions, I struggled less writing the script for this part. Again, I split the X and Y into the training set and test set using `train_test_split`, where X is the data frame containing the features of the tracks, and Y is the data frame containing which genre it is. Then I just run the `DecisionTreeClassifier()` on the training sets, which is then used to predict the genres of tracks in the test set.

Simiarly, I tested whether the classification method was able to predict that Immigrant Song and Bohemian Rhapsody are rock songs.

Result Analysis

As a reminder, the 8 features chosen for analysis are: accousticness, danceability, energy, instrumentality, liveness, loudness,speechiness, and tempo. As I didn't set any parameters in `DecisionTreeClassifier()`, by default, the classifier would pick attributes based on the Gini index, so the feature with the smaller Gini index is chosen as the split point.

When I plot the decision tree, there are a couple of iterations of the tree that are usually returned.



From looking at them, there are only five features that show up: loudness, acousticness, tempo, energy and speechiness, with only acousticness and tempo showing up in nearly all of them, and loudness usually being at the root of the tree.

Looking at these features, it makes sense as the distinction would lie within how rock music is louder, has higher tempo and energy, more speech and less acousticness (this feature measure the confidence level of the track being accoustic) than piano classical pieces. However, it also implies that danceability, instrumentality, and liveness of these tracks are less significant in distinguishing whether a track belongs to the rock genre or piano classical genre.

The accuracy score for this method of classification is slightly lower than the previous with 0.97 as it's average score. But from all my runs so far, has always been able to correctly identify that Immigrant Song

and Bohemian Rhapsody are rock songs.

Conclusion

Using PCA coupled with Random Forest method can be more efficient as PCA lowers number of components needed to process, thus also removing noise by removing components that add little predictive value (although performing PCA itself is already computationally expensive). Therefore, this method allows for fairly accurate classification.

Using Decision Trees allows us to better understand which features add more predictive value. They also require less processing as we didn't need to standardise our data to transform it. However, there is possibility of it overfitting to all features, especially since we only had 8.

Due to the smaller size of this dataset, as well as only considering 8 features, both methods had high accuracy scores for prediction and terminated in a couple of seconds. Knowing this, I would recommend using Decision Trees more simply because you learn more about the dataset, and you can use to for preidictions.