CS564 Machine Learning

Project 3: Audio Genre Classification

Rahul Kalaiselvan (101882764)

Sairamya Madhavi Devi Praturi (101875020)

INTRODUCTION

In this project we are classifying music/audio based on their genres. Our data set consists of 2400 audio samples of 30 secs. Our test data set consists of 1200 audio samples whose classes remains unknown. The audio is loaded using Librosa and saved as NPY file faster computation. In this document the labels 0 to 5 are representing these following genres:

- **Rock**: 0
- **Pop**: 1
- **Folk**: 2
- **Instrumental**: 3
- **Electronic**: 4
- **Hip-Hop**: 5

DATA REPRESENTATION:

The three ways we have represented our data is by extracting audio features (frequency characteristics), extracting features of a time series of audio data, Image classification by converting the audio into spectrogram.

Extracted Audio Features (Domain specific coefficients):

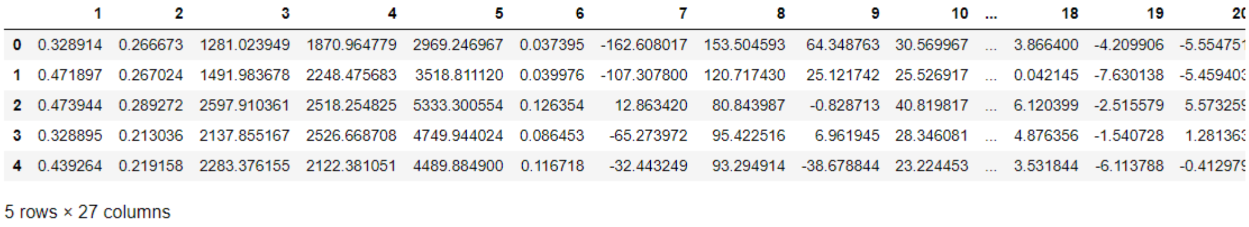
Audio features such as MFCC, spectral centroid, spectral bandwidth, spectral roll-off, zero crossing rate, chromagram are extracted from the audio data.

MFCC is one of most important features when trying to perform a genre classification. We have used different combinations of MFCC's mean, variance, standard deviation and median.

We trained the data on both SVM and ANN (Artificial Neural Network) classifiers and compare.

Using audio features and 20 MFCCs:

This dataset consists of all the mean of extracted audio features and 20 MFCC data. A MFCC value has a shape (20,1920) approx. We calculate the mean of all 20 values and append it to the dataset.

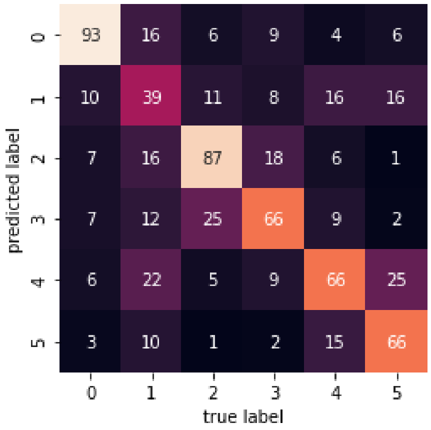


	1	2	3	4	5	6	7	8	9	10	...	18	19	20
0	0.328914	0.266673	1281.023949	1870.964779	2969.246967	0.037395	-162.608017	153.504593	64.348763	30.569967	...	3.866400	-4.209906	-5.554751
1	0.471897	0.267024	1491.983678	2248.475683	3518.811120	0.039976	-107.307800	120.717430	25.121742	25.526917	...	0.042145	-7.630138	-5.459403
2	0.473944	0.289272	2597.910361	2518.254825	5333.300554	0.126354	12.863420	80.843987	-0.828713	40.819817	...	6.120399	-2.515579	5.573259
3	0.328895	0.213036	2137.855167	2526.668708	4749.944024	0.086453	-65.273972	95.422516	6.961945	28.346081	...	4.876356	-1.540728	1.281363
4	0.439264	0.219158	2283.376155	2122.381051	4489.884900	0.116718	-32.443249	93.294914	-38.678844	23.224453	...	3.531844	-6.113788	-0.412979

5 rows × 27 columns

SVM:

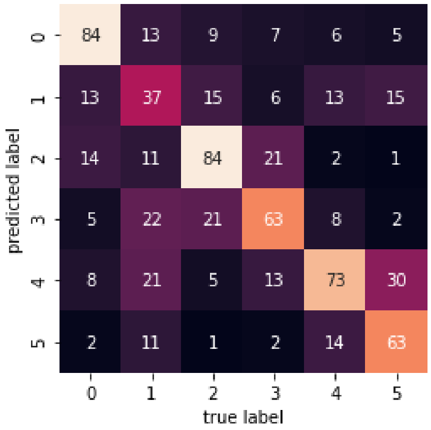
Accuracy: 0.53 (+/- 0.13)



	0	1	2	3	4	5
0	93	16	6	9	4	6
1	10	39	11	8	16	16
2	7	16	87	18	6	1
3	7	12	25	66	9	2
4	6	22	5	9	66	25
5	3	10	1	2	15	66

ANN:

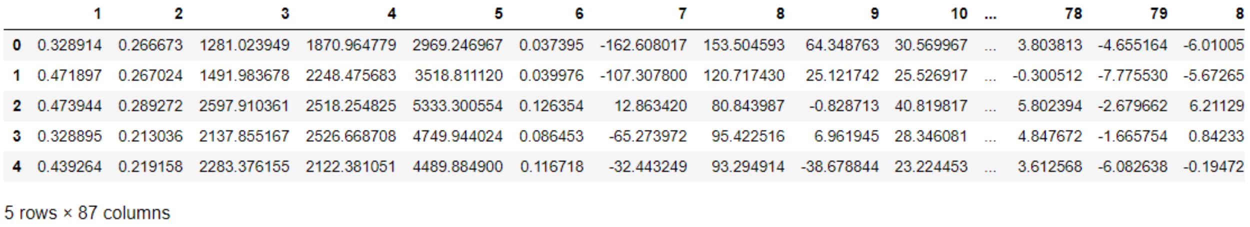
Accuracy: 0.45 (+/- 0.06)



	0	1	2	3	4	5
0	84	13	9	7	6	5
1	13	37	15	6	13	15
2	14	11	84	21	2	1
3	5	22	21	63	8	2
4	8	21	5	13	73	30
5	2	11	1	2	14	63

Using audio features and 20X4 MFCCs:

This dataset consists of all the extracted audio features and 20X4 MFCC data. The mean, median, standard deviation and variance of all 20 MFCCs are included as features.

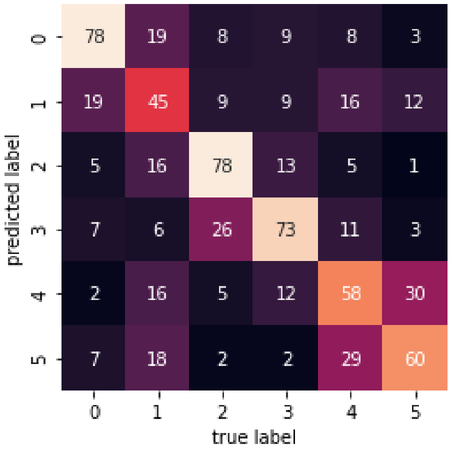


	1	2	3	4	5	6	7	8	9	10	...	78	79	8
0	0.328914	0.266673	1281.023949	1870.964779	2969.246967	0.037395	-162.608017	153.504593	64.348763	30.569967	...	3.803813	-4.655164	-6.01005
1	0.471897	0.267024	1491.983678	2248.475683	3518.811120	0.039976	-107.307800	120.717430	25.121742	25.526917	...	-0.300512	-7.775530	-5.67265
2	0.473944	0.289272	2597.910361	2518.254825	5333.300554	0.126354	12.863420	80.843987	-0.828713	40.819817	...	5.802394	-2.679662	6.21129
3	0.328895	0.213036	2137.855167	2526.668708	4749.944024	0.086453	-65.273972	95.422516	6.961945	28.346081	...	4.847672	-1.665754	0.84233
4	0.439264	0.219158	2283.376155	2122.381051	4489.884900	0.116718	-32.443249	93.294914	-38.678844	23.224453	...	3.612568	-6.082638	-0.19472

5 rows × 87 columns

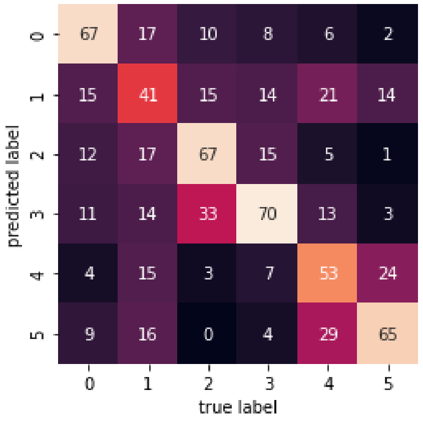
SVM:

Accuracy: 0.53 (+/- 0.11)



	0	1	2	3	4	5
0	78	19	8	9	8	3
1	19	45	9	9	16	12
2	5	16	78	13	5	1
3	7	6	26	73	11	3
4	2	16	5	12	58	30
5	7	18	2	2	29	60

ANN:
Accuracy: 0.50 (+/- 0.06)



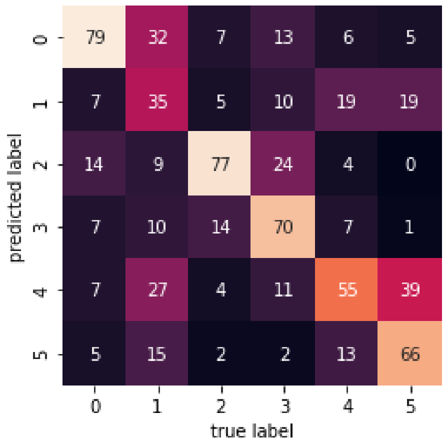
	0	1	2	3	4	5
0	67	17	10	8	6	2
1	15	41	15	14	21	14
2	12	17	67	15	5	1
3	11	14	33	70	13	3
4	4	15	3	7	53	24
5	9	16	0	4	29	65

Using audio features and Crucial points of MFCCs

Along with the audio features, we are extracting interquartile ranges, four means and two medians.

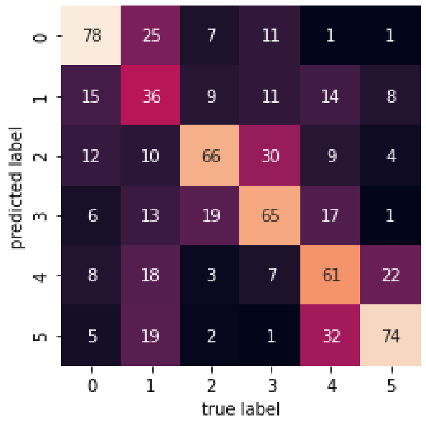
SVM:

Accuracy: 0.52 (+/- 0.12)



	0	1	2	3	4	5
0	79	32	7	13	6	5
1	7	35	5	10	19	19
2	14	9	77	24	4	0
3	7	10	14	70	7	1
4	7	27	4	11	55	39
5	5	15	2	2	13	66

ANN:
Accuracy: 0.51 (+/- 0.10)



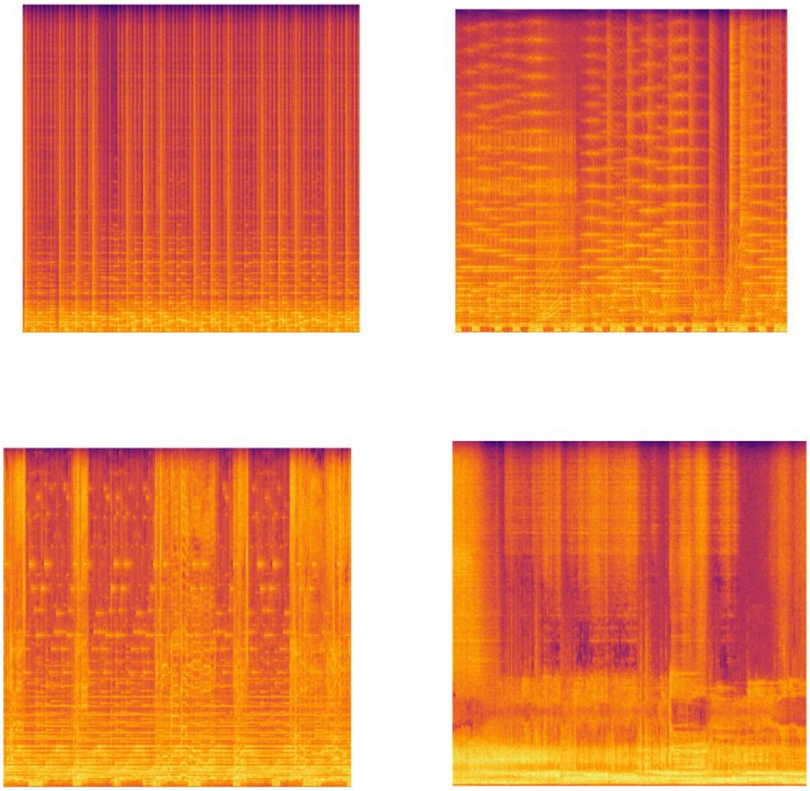
	0	1	2	3	4	5
0	78	25	7	11	1	1
1	15	36	9	11	14	8
2	12	10	66	30	9	4
3	6	13	19	65	17	1
4	8	18	3	7	61	22
5	5	19	2	1	32	74

Result:
MFCC worked the best for my problem. When adding more detailed features of MFCC the classifier worked b etter. SVM gave the best results for classification. I have documented the result of SVM and ANN.

Mel Spectrogram (Frequency domain)

These are images of spectrogram converted into Mel scale. The spectrogram is a map of spectral frequencies of the audio with time.

An image classification using CNN is applied on the audio data which are converted into Mel spectrograms.



CNN:

Accuracy: 45.78%

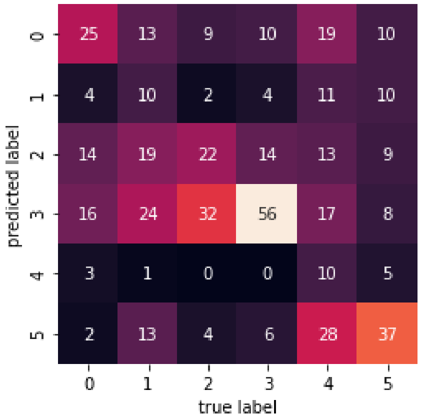
Observation

The image classification was not as successful as the classifier with the extracted MFCC and other frequency features. A bigger training data might have given a better result with CNN. We trained the model for 1000 epochs and the results were documented.

Floating Point Time Series

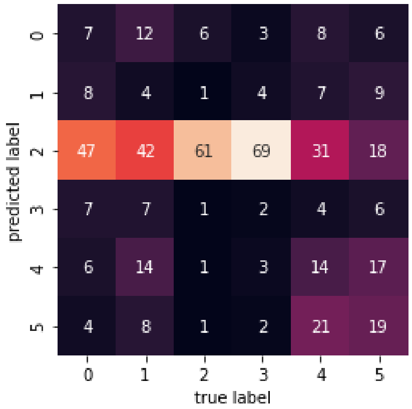
Features of time series data such as mean, standard deviation, variance, mean, interquartile range and quartile deviation were extracted. We did not get any good classifier from this data model.

Accuracy: 0.29 (+/- 0.06)



PCA of Time Series and MFCC

We tried do PCA, ICA and SVD for the audio’s time series data and 20 MFCC data. We did not get a good classifier even after changing them to different configurations and trying them out. The work can be viewed in *PCA_mfcc Jupyter notebook.* The best accuracy we could achieve was 28%.



IMPLEMENTATION:

The classification is implemented in Python 3. The important libraries used are NumPy and Pandas for data preprocessing and storing. SciKit Learn, Keras and TensorFlow for machine learning algorithms, data preprocessing. Matplot Lib and Seaborn for data visualization.

All the data preprocessing like feature extraction from audio data, converting mp3 to WAV, creating mel spectrogram images are in the classes (Python) in *project3.py* module called from *main.py.* All the machine learning algorithms and matrix visualization are implemented in Jupyter notebook.

Audio_Features Notebook (Jupyter):

This notebook implements several different classifiers on the extracted audio features (MFCCs) and plots confusion matrix.

CNN Notebook (Jupyter):

This notebook implements image classification on the mel spectrograms created using Convolution Neural Network. Library used for NN is Keras.

CLASSIFIER USED:

SVM works good with multivalued classification problems. They can be used for both ideal and non-ideal classification problems. SVM suffers with increasing features, SVM worked well when our dataset contained less extracted features and only the most essential ones.

ANN can detect complex nonlinear relationships in the dataset with ease. It has the ability to detect all possible relations within the dataset. Since in our problem we were unsure of the importance of each feature/parameter, ANN gave good results.

When it comes to image classification, CNN works out the best result and thus CNN was used when we tried to classify the audio genres using spectrograms of the respective audios and tried to solve a image classification problem.

BIAS:

While using an ANN model, the results of genre label 4 and 5 (Which are Hip-Hop and Electronic) are confused for most of the dataset. The label 1 (Pop) is the most confused genre among all, this confusion was more while using SVM. The reason for confusion of 4 and 5 from what we observed were because of the similarity in the genre 4 and 5. They have almost similar frequency modulations and pith variations.

CONCLUSION:

SVM when used on the MFCC and other frequency related extracted data gave the best results. ANN on the same data gave the next best result. Other that the features extracted from frequency domain, spectrogram of audio samples solved as an image classification problem gave the best results. We believe a more detailed/extensive training data can significantly improve the performance of the image classification problem, giving better results when using the convolution neural network (CNN).

REFERENCES

- **Librosa** - https://librosa.github.io/librosa/
- **Keras Documentation** - https://keras.io/guides/
- **Scikit-Learn** - https://scikit-learn.org/stable/index.html#
- **NumPy** - https://numpy.org/
- https://medium.com/@mikesmales/sound-classification-using-deep-learning-8bc2aa1990b7
- https://www.tensorflow.org/tutorials/images/cnn
- https://www.tensorflow.org/tutorials/images/classification
- Hershey, Shawn, et al. "CNN architectures for large-scale audio classification." *2017 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, 2017.
- Tzanetakis, George, and Perry Cook. "Musical genre classification of audio signals." *IEEE Transactions on speech and audio processing* 10.5 (2002): 293-302.