CS 529 Project 3 – Audio Categorization

Sai Aditya Thalluri
Sharath Chandra Tangella
Manideep Potluru

Introduction:

In this project, we are given MP3 files for training and testing. There are 2400 training data and 1200 test data. We must represent the data in three different ways and implement two different classifiers and compare their accuracy with respect to the three data representations we did earlier.

The three different representation that we choose for our project are: Mel frequency cepstral coefficients (MFCCs), Short-time Fourier transform (STFT),Principal Component Analysis (PCA) and Mel-Spectrograms.

Data Representations:

1. <u>Mel Frequency Cepstral Coefficients (MFCC):</u>

MFCCs are the coefficients that combinedly form an MFC, which is a representation of the power spectrum of sound which is short term on a non-linear Mel scale frequency based on a linear cosine transform of a log power spectrum. With the help of cepstral representation of the sound clip the MFCCs can be derived. Cepstrum is the rate of change in spectral band information. On the Mel scale, the frequency bands are equally spaced for MFC which helps in close approximation of the response of human auditory system than the normal cepstrum that uses linearly spaced frequency bands. A Mel scale is a scale that compares the interpreted frequency of a tone to the frequency that is actually measured.

The creation process of an MFCC feature follows this procedure:

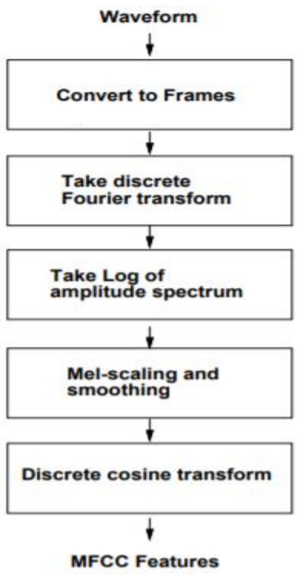

Waveform
Convert to Frames
Take discrete Fourier transform
Take Log of amplitude spectrum
Mel-scaling and smoothing
Discrete cosine transform
MFCC Features

Figure 1: MFCC feature creation process

In the first step, the speech signals are divided into frames generally by using a windowing functions at fixed intervals. The window function removes the edge function. Cepstral feature vector for each frame is generated. Then the Discrete Fourier Transform of every frame is taken and then from the amplitude spectrum we retain only the logarithm of it as the signals perceived loudness is approximately logarithmic. The next step is to smooth the spectrum and Mel scaling. In the last step o MFCC feature construction, we apply a transform to the Mel spectral vectors to reduce the number of parameters in the system.

MFCCs can be used in speech recognition as features and for genre classification, audio similarity measures in music information retrieval.

Rationale behind the selection for MFCC:

MFCC are known to encode information based on the perceived sound quality of a musical note which is known as timbral encoding. They extract the featured data which has audible characteristics from the music content, due to which they are able to identify consistent features even if two musical contents use different digitizing specification.

2. <u>Short-Time Fourier Transform (STFT):</u>

The Short-time Fourier transform is used to find out the phase content and sinusoidal frequency of the local sections of the signal as it varies over time. In order to find the STFTs we need to divide the time signal that is longer into shorter chunks of equal length and then separately calculate the Fourier transform on each shorted chunk. From this the Fourier spectrum on each shorter chunk is revealed. We can then plot a spectrogram as a function of time which is the change in spectra.

In Continuous STFT, a short term non-zero window function is multiplied to the function to be transformed. The resulting signal’s Fourier Transform is taken as the window and this window is slithered along the time axis. This result in 2-D representation of the signal. This can be written mathematically as:

$$STFT\{x(t)\}(\tau,\omega) = \int_{-\infty}^{\infty} x(t)w(t-\tau)e^{-i\omega t}dt$$

Here, $x(t)$ is the signal that needs to be transformed, the window function is $w(\tau)$. In order to subdue the jump discontinuity of the STFT phase result, phase unwrapping is used in either one or both of frequency and time axis.

In Discrete STFT, the Fourier transform is applied to the broken frames or chunks of the data that needs to be transformed. This gives us a complex result which is then added to the matrix. For every point on the frequency and time, phase and magnitude is recorded by this matrix.

$$STFT\{x[n]\}(m,\omega) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\omega n}$$

Inverse STFT is useful to acquire the original signal from the transform because STFT is invertible. One of the issues with STFT is that it has a fixed resolution and this is one the reasons for the designing multi-resolution analysis and wavelet transform.

Rationale behind the selection for STFT:

In all of the time frequency analysis methods, the Short Time Fourier Transform(STFT) is said to be one of the most useful as it provides us with simultaneous insight in the time and frequency of the behaviour of the signal. Applying STFT to our musical samples, gave us the time-frequency distributions. In the time frequency analysis, we have used the default window types, sizes and overlap ratios and have used them for further classification.

3. <u>Principal Component Analysis:</u>

Principal Component Analysis is useful for dimensionality reduction of the dataset that comprises of many variables that are correlated with one another. It is a non-parametric and unsupervised statistical technique. There can be a problem of overfitting with high dimensionality. Hence, with PCA the dimensionality is reduced. A principle component (PC) can be defined as linear combination of optimally weighted observed variables. These Principal components are the outputs of PCAs. The weight vectors in the linear combination are the eigen vectors. PCs are orthogonal and the dissimilarity that is present in the PCs decreases as we move from the first PC to the last one.

The initial Principal Component (PC1) is a linear combination variable that is built to find the direction and magnitude of the highest variance in the dataset. This component has the most information because it has the highest variability compared to other components. To maximize the variance, the initial weight vector $w_{(1)}$ must satisfy

$$\mathbf{w}_{(1)} = \underset{\|\mathbf{w}\|=1}{\arg\max} \left\{ \sum_i (t_1)^2_{(i)} \right\} = \underset{\|\mathbf{w}\|=1}{\arg\max} \left\{ \sum_i \left(\mathbf{x}_{(i)} \cdot \mathbf{w}\right)^2 \right\}$$

Where w is vectors of weights, $x_{(i)}$ is the row vector of X and $t_{(i)}$ is the vector of principal component scores. Since $w_{(1)}$ is defined to be a unit vector, it also satisfies

$$\mathbf{w}_{(1)} = \arg\max \left\{ \frac{\mathbf{w}^T \mathbf{X}^\mathbf{T} \mathbf{X} \mathbf{w}}{\mathbf{w}^T \mathbf{w}} \right\}$$

Similarly, the next principal components calculate the remaining variance without being correlated to the previous components.

Datasets that have low feature correlation can see a reduction in model performance when PCA is applied. PCAs are affected by outliers and normalization of the data should be an important component of a workflow. The importance of individual features are not recognized because the principal component does not allow that.

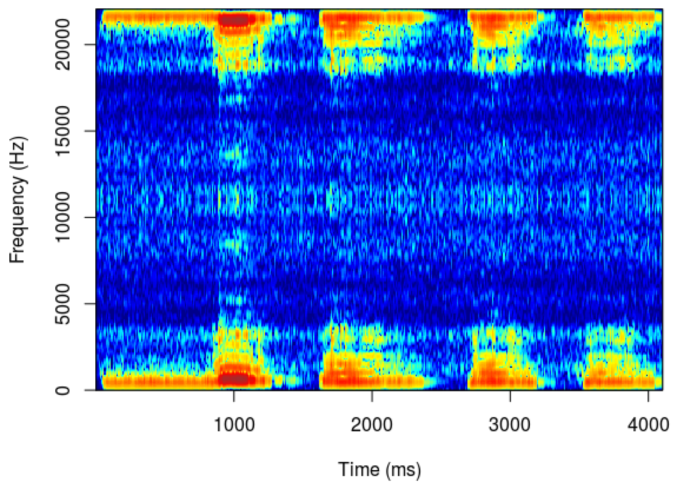
We have implemented Logistic Regression classification and Random Forest for MFCC and STFT, SVM and XGBoost for PCA.

Rationale behind the selection for PCA:

In general, reducing the number of features is always useful as it results in a simplified model which is what we are looking for. Since our musical dataset is not likely to have any strong correlations between the features, we used a common approach to reduce the number of features and retaining as much information as possible i.e.., Principle Component Analysis(PCA) in which we extract the most important features that show the variance of the dataset.

4. **Mel-Spectrograms:**

Spectrogram is the visible depiction of the frequential spectrum of the signal varying with time. They are also called as voicegrams or sonograms when they are applied to audio signal. It is called as waterfall when the data is depicted in 3D model. A spectrogram is generally represented as a heat map. Generally, spectrograms are a 2D graph, with the third dimension as colours. The horizontal axis represents the time and the vertical axis represents the frequency where the lesser frequencies are at the bottom and the higher frequencies are at the top. Weaker amplitude is represented in blue and higher amplitude in red.



Time domain signals can be used to create spectrograms in two different ways: Using the Fourier Transforms to calculate the time signal and filter bank resulting from a series of band pass filters. They can also be created digitally by using FFT. The data that is sampled digitally in the time domain is split into chunks and magnitude of frequency of these chunks is calculated by Fourier Transforming them. Each vertical line in the image is correspondent to each of the chunk. These spectrums are placed side to each other to form a 3D image. We cannot generate the original signal from a spectrogram by reversing the process because the spectrogram does not contain information about the phase of the signal that it depicts. But in case where the starting phase not important, we can generate an estimation of the original signal. Spectrograms give us a lot of information about the acoustic elements of the sound.

The output spectrogram images an be used combinedly with Machine learning classifier. Through spectrogram analysis, application of several learning algorithms and feature extraction, basic classification can be performed, which provides more insight into the genre of the music and also the spectrograms of most audio clips have many distinguished features. These features of spectrogram help in the classification of music genres efficiently and thus we use the spectrograms in classifying.

Classifiers:

1. <u>Logistic Regression:</u>

Logistic regression is a supervised classification algorithm. Logistic regression is a regression model and it uses sigmoid function to model the data. The sigmoid function is:

$$g(z) = \frac{1}{1 + e^{-z}}$$

Logistic regression uses sigmoid function to transform its output and returns a probability value which can be mapped to two or more classes. There are three types of logistic regression models:

1. Binary
2. Multinomial
3. Ordinal

The goal of logistic regression is to train classifier that can make decision about the new input observation. This is done by learning from a set of training data, a vector of weights and bias term. Each weight is associated with one of the input features and represents how essential the input feature is to the classification decision.

The classifier first multiplies each feature x_i by its weight w_i, sums the weighted features and at last adds the bias term b. The resultant weighted sum is:

$$z = \left(\sum_{i=1}^{n} w_i x_i\right) + b$$

The value of z is passed to the sigmoid function to create a probability.

We will be needing a loss function that will tell us how close the output given by the classifier is to the actual output. We choose the values of w, b that maximize the log probability of the actual y labels in the training data, given the observations x and this is called as conditional maximum likelihood estimation. The equation of the loss function is:

$$J(\theta) = -\sum_{i=1}^{n}\sum_{k=1}^{K} 1\{y^i = k\}\, logP(y^i = k|x^i;\theta)$$

Where K is the number of total classes and θ is vector that stores coefficients of the class for all words.

To minimize the loss function, we use Gradient Descent. Gradient Descent finds the minimum of a function by finding out the direction in which the slope of the function is rising more steeply in the opposite direction. The loss function in convex for logistic regression. Since a convex function has no local minima as there is only one minimum, the gradient descent is guaranteed to find the minimum starting from any point.

The learning rate η determines the magnitude of amount to move in gradient descent. The change we make is the learning rate times the gradient. It is:

$$w^{t+1} = w^{t} - \eta \frac{d}{dw} f(x;w)$$

Rationale behind the selection for Logistic Regression:

For this music genre classification Logistic regression is implemented as a one vs rest method. That is, we trained 6 binary classifiers and then during the time of testing, the class with the highest probability among the 6 classifiers is chosen as the predicted class. We applied Logistic Regression on MFCC and STFT feature statistics as it finds good relation among the variables.

2. **Random Forest:**

Random Forest Model is considered as one of the best algorithms when it comes to classification problems. It is an ensemble algorithm. When compared to a single Decision Tree, a Random Forest of many decision trees gives better results. Each tree in the Random Forest is constructed from a random piece of training data, so that we will be able to handle any random test inputs. While performing predictions, it takes outputs from each of its tree for the given test input and considers the majority or the average from whole as final prediction based on our data and application.

They run efficiently on large datasets and can handle many input variables without the deletion of variables. Random forest can be used for classification as well as regression problems. Measuring the relative importance of every feature on the prediction becomes easy with Random Forest and the default parameters it uses produces a good result. Importance of variables in a classification or regression can be ranked in a natural way. As the construction of the forest progresses, an unbiased estimate of the generalization error is generated by it. Estimation of missing data can be done effectively, and the accuracy will be high even when there is large amount of missing data. This algorithm is a great choice for developing a model quickly.

Rationale behind the selection for Random Forest(RF):

Random forest is an ensemble learning classifier, which combines the predictions from a pre-specified number of decision trees. All the trees are trained with the subsets of the samples and each of them make a prediction, from which the final class is predicted as the majority of the individual class predictions. We applied random forest on MFCC and STFT feature statistics, as it works very well with ensemble predictions which reduces overfitting.

3. **Support Vector Machines:**

Support Vector Machines are set of supervised learning methods used for regression and classification. SVM is a linear machine whose norm is to minimize:

$$L_p(w, \xi_n) = \frac{1}{2} b\|w\|^2 + C \sum_{n=1}^{N} \xi_n$$

With the constraints,

$$y_n(w^T x_n + b) > 1 - \xi_n, \xi_n \geq 0$$

They use machine learning theory to increase the accuracy of prediction and at the same time avoid over-fit to the data. SVM construct hyper planes for classification, regression and outlier's detection. When we plot the data and try to classify it, there can be many linear classifiers that separate the data and only one of these linear classifiers achieve maximum separation. The concept of maximum margin classifier is important because if we use a hyper plane to classify, it might end up being close to one set of datasets compared to others and we do not want this to happen. We choose the classifier that has maximum separation as the larger the margin, the lower the error of classifier. SVM are used in various applications like text categorization, tone recognition, image classification, micro-array gene expression analysis and many more. SVM has drawbacks such as they require full labelling of input data, uncalibrated class membership probabilities and parameters of a solved models are hard to interpret. SVM's have flexibility in choosing a similarity function and they have ability to handle large feature spaces. In SVM's, overfitting can be controlled by soft margin approach.

Rationale behind the selection for Support Vector Machines(SVM):

Support Vector Machines are known to use kernel tricks and transform the original input into a higher dimensional space. The transformed data can be linearly separated using a hyperplane, which should me maximized to find the optimal hyperplane. We have used one vs rest classification task. Radical basis function(RBF) kernel trick is used in order to train the svm as it is the best method to be used for non-linear problem such as ours.

4. <u>XGBoost:</u>

XGBoost stands for Extreme Gradient Boost. It is an ensemble decision tree-based Machine Learning algorithm that utilizes a gradient boosting framework. The concept of Boosting and Gradient Descent are carried by the Gradient Boosting to supervised learning and the Gradient Boosted Models (GBM's) are trees built consecutively, in series. In GBM's, multiple model's weighted sum is taken. XGBoost primarily focuses on mode performance enhancement and speed. In order to update or make any corrections to the weights, every new model uses Gradient Descent optimization to reach the local minimum of the cost function. A new function is added by the Gradient Boosting to the function that is existing in every step to predict the output. A different function from the beginning is obtained from the Gradient Boosting since the output is the summation of multiple functions. The three main forms of gradient boosting namely Gradient boosting, Regularized Gradient boosting and Stochastic Gradient boosting are supported. Some of the feature that make XGBoost different from other gradient boosting algorithms are Newton boosting, Extra randomization parameter, leaf nodes shrinking proportionally and penalization of trees smartly.

Rationale behind the selection for XGBoost:

Boosting is obtained by combining a number of weak learners in the same way as Random Forests, however, unlike random forests, boosting algorithms are trained in sequential manner using forward state addictive modelling. The XGBoost focuses more on the instances where learning errors are made previously. The prediction is a weighted linear combination of the output from the individual learners. XGB is a fast and parallelized boosting which will help in training the model in reduced time which helps us build the model faster and very efficiently. XGBoost works well on dimensionality reduced features as they possess high variance information which helps in building efficient trees for classification.

5. <u>2D-Convolutional neural network:</u>

The 2-dimensional neural network is the standard neural network that was introduced. It is generally used on data which has pictorial representation. It is called 2-Dimensional data because the kernel is such a way that it slides along two dimensions on the data. The whole advantage of using CNN is that unlike other network, the 2-D NN can extract the spatial features from the data using it's kernel. It can detect characteristics like edges, and the distribution of colours in the image which makes it very robust in image classifications and other data that has spatial properties.

6. <u>1D-Convolutional neural network:</u>

The one-dimensional neural network is a network which is used on the data which is in the form of time series. The major difference between the 2-Dimensional and the 1-Dimensional Neural Network is that for the data in 1D, the kernel slides only along one dimension and still have spatial properties. The data will usually have two dimensions, the first dimension in the time steps and other is the values of the data in different axes. 1D is widely applied to sensory data such as accelerometer data.

Rationale behind the selection for 1D and 2D CNN:

In-order to classify the spectrograms of the audio, we need a classifier that will find out the frequency patterns in the spectrograms. As we know Convolutional Neural Networks are the best at matching patterns among data, we are using them on this data representation. "better" is relative here. 1D convolutions give results that are not contaminated by random patterns. 2D Convolutions may get better accuracy, but then we have locality patterns that do not mean anything and are thereby random luck. We used both 1-d and 2-d convolutions to compare and check their performance on spectrograms.

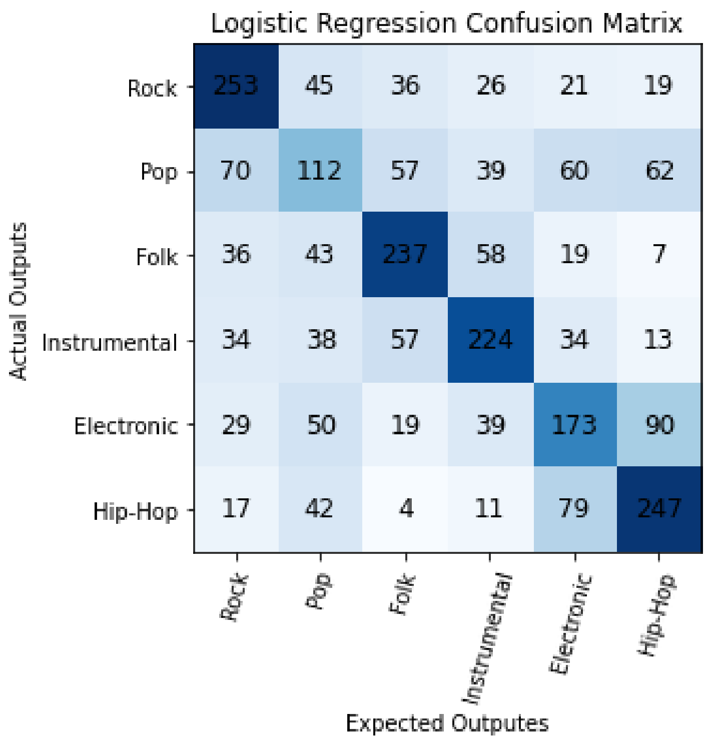
Comparing the Classifiers for each data representation:

1. <u>MFCC: Logistic Regression vs Random Forests</u>

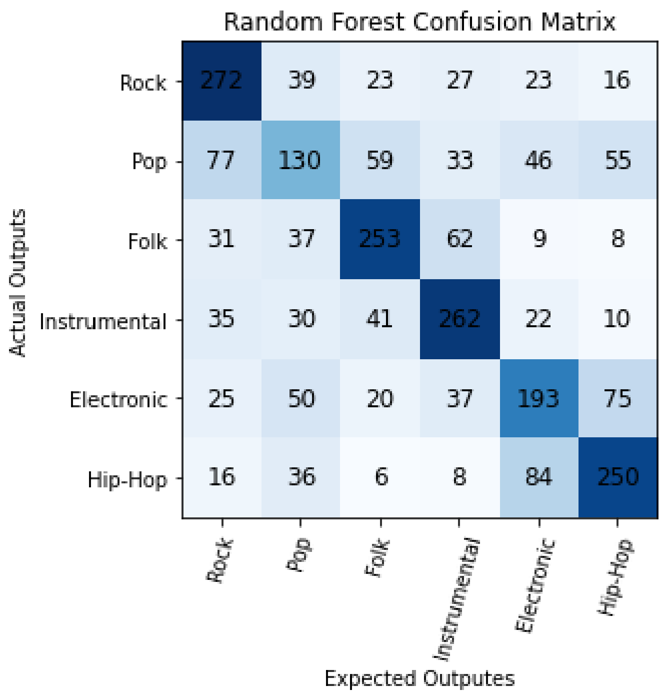
We have first modified our given music data into our first data representation format which was Mel frequency cepstral coefficients (MFCCs). We have used the MFCCs and have tried to classify them with two different classifiers, which are Logistic regression and Random Forests respectively.

Accuracies obtained: When the trained models were used to predict the genre of the music, we have obtained an accuracy around 54% with the Logistic regression classifier vs an accuracy of around 58% on the testing data using the random forest classifier over the coefficients.

A confusion matrix was drawn for both the classifiers after the classification was done on the train set/validation set in order to analyse the results of the classification. The confusion matrices for both of these are given below:


Logistic Regression Confusion Matrix

Actual Outputs \ Expected Outputes	Rock	Pop	Folk	Instrumental	Electronic	Hip-Hop
Rock	253	45	36	26	21	19
Pop	70	112	57	39	60	62
Folk	36	43	237	58	19	7
Instrumental	34	38	57	224	34	13
Electronic	29	50	19	39	173	90
Hip-Hop	17	42	4	11	79	247

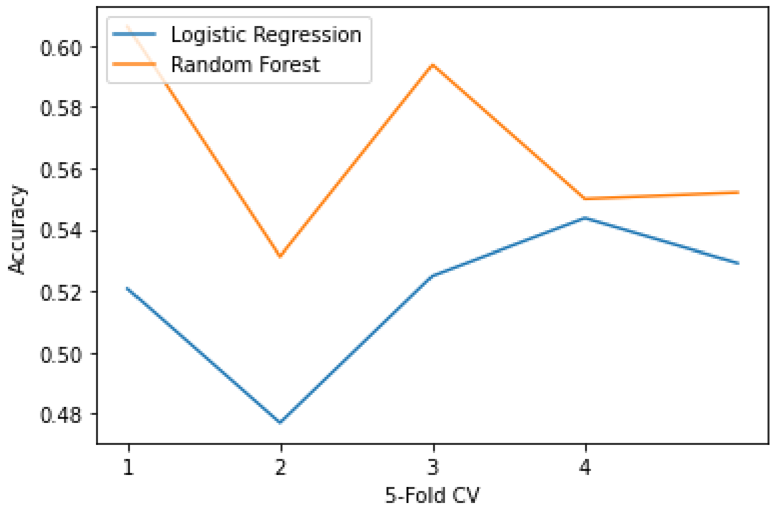

Random Forest Confusion Matrix

Actual Outputs \ Expected Outputes	Rock	Pop	Folk	Instrumental	Electronic	Hip-Hop
Rock	272	39	23	27	23	16
Pop	77	130	59	33	46	55
Folk	31	37	253	62	9	8
Instrumental	35	30	41	262	22	10
Electronic	25	50	20	37	193	75
Hip-Hop	16	36	6	8	84	250

Results and discussion: We can infer from the obtained confusion matrices that both the classifiers have been very good at classifying the Rock, Hip-Hop, folk and instrumental genres. We can also notice that either of them struggles with classifying the Electronic and pop genres correctly and is worse when using logistic regression. We obtained more accuracy with random forest than compared to Logistic. Random forest combines prediction from many different individual trees and have low bias and high variance. On the other hand, Logistic regression is completely flat and that's why it has larger bias and low variance. Error can be reduced with standard regularization but works well only on linear data. These might be the reasons why random forest performs well with our data.

Bias: Based on the misclassifications in the confusion matrices, we can see both the classifiers are having more bias than expected. Both classifiers are not able to differentiate samples from classes pop and electronic in most of the cases and are more biased towards other classes.

We have done a 5-fold cross validation on either of the classifiers and plotted the graph for accuracies at each of the folds in the 5-fold CV. The plotted graph is shown below:



We can observe that at each on every fold of the 5-fold CV, Random forests have given us fairly better accuracies over the Logistic Regression.

Confidence Intervals:

At 99% Confidence Interval:

- True Classification Accuracy of Logistic Regression is likely between 49.41066135787388 and 54.67267197545945

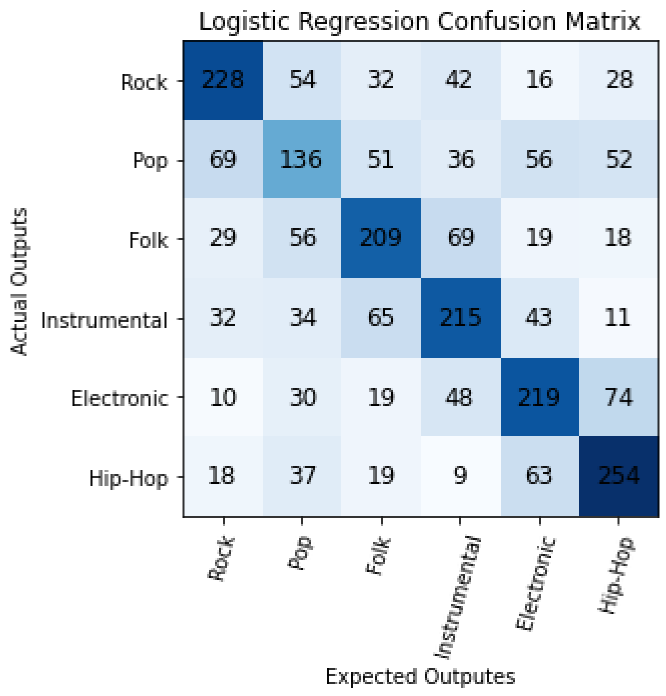
- True Classification Accuracy of Random Forest is likely between 53.67939546905034 and 58.90393786428299.

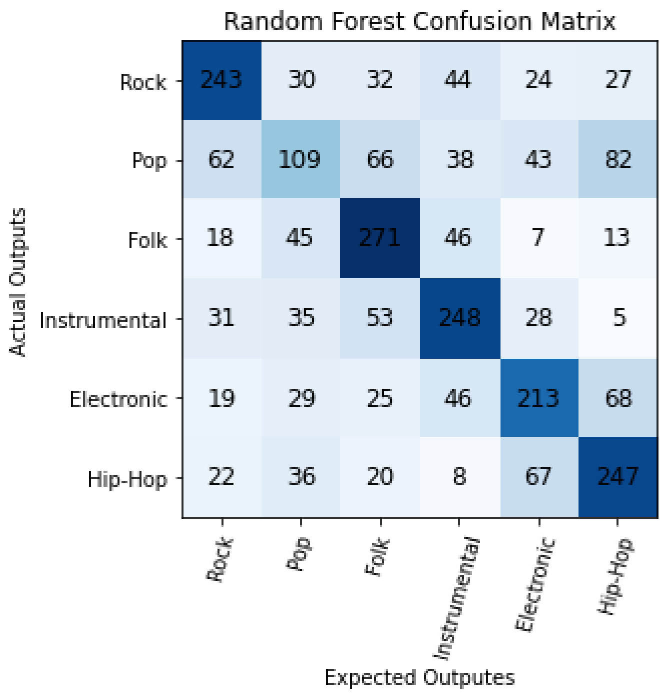
2. <u>STFT: Logistic Regression vs Random Forests</u>

We have first modified our given music data into our first data representation format which was Short Term Fourier Transforms. We have used the STFTs and have tried to classify them with two different classifiers, which are Logistic regression and Random Forests respectively.

Accuracies obtained: When the trained models were used to predict the genre of the music, we have obtained an accuracy around 53% with the Logistic regression classifier vs an accuracy of around 57% on the testing data using the random forest classifier over the Fourier Transforms.

A confusion matrix was drawn for both the classifiers after the classification was done on the train set in order to analyse the results of the classification. The confusion matrices for both are given below:

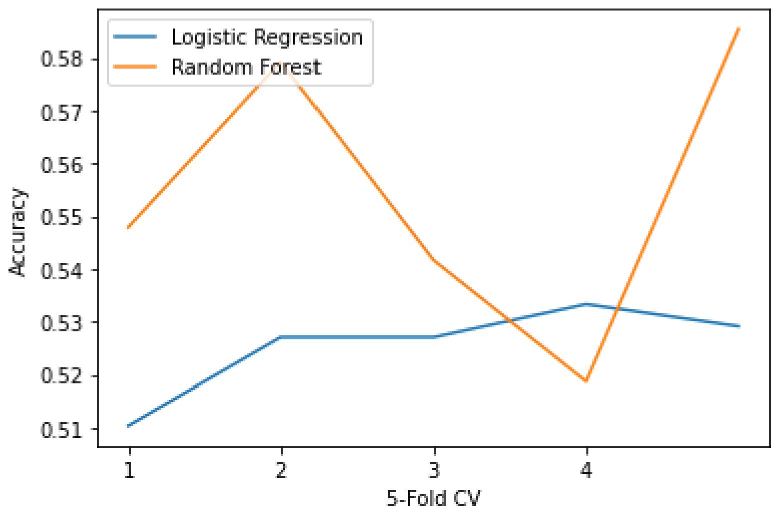




Results and Discussion: We can infer from the obtained confusion matrices that both the classifiers have been very good at classifying the Rock, Hip-Hop, folk and instrumental genres. We can also notice that either of them has classified electronic decently than in when using MFCCs but still struggle with classifying the pop genres correctly. Peculiar to MFCC, random forests have performed worse when classifying pop. Here too we see that random forest is performing well compared to logistic. It might be for the same reasons as mention above for the MFCC classification. Also, the data taken plays an important role in the performance of different classifiers. There might be a possibility that logistic works better than RF for another dataset.

Bias: Based on the misclassifications in the confusion matrices, we can see both the classifiers are having more bias than expected. Both classifiers are not able to differentiate most of sample from class pop and some from electronic. Classifiers are biased towards other classes when they encounter samples from those classes.

We have done a 5-fold cross validation on either of the classifiers and plotted the graph for accuracies at each of the folds in the 5-fold CV. The plotted graph is shown below:



We can observe that at four of the 5-fold CV, Random forests have given us fairly better accuracies over the Logistic Regression, only in the fourth fold, we can observe an case where random forest performs worse than logistic regression.

Confidence Intervals:

At 99% Confidence Interval:

- True Classification Accuracy of Logistic Regression is likely between 49.57770139929242 and 54.83896526737425

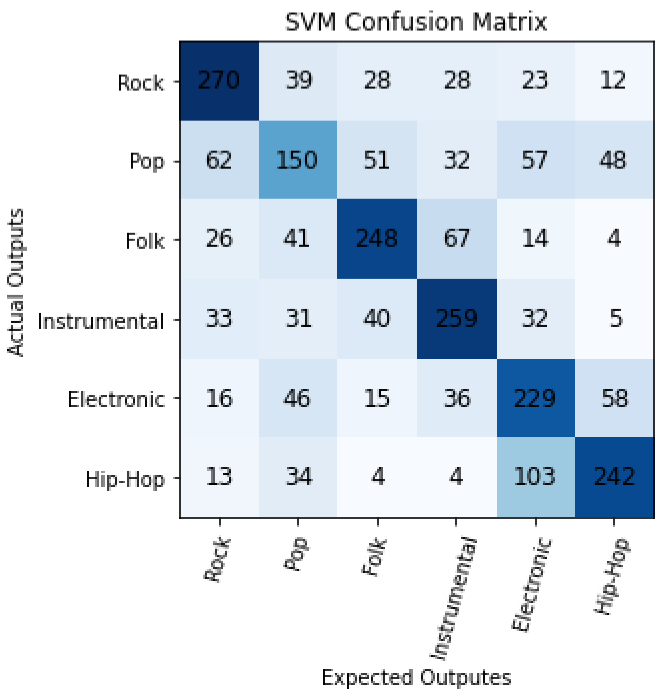
- True Classification Accuracy of Random Forest is likely between 53.09234875579932 and 58.32431791086735.

3. <u>PCA: SVM vs XGBoost</u>

We have first modified our given music data into our first data representation format which was PCA the MFCCs. We have used the PCA and have tried to classify them with two different classifiers, which are Support Vector Machines and XGBoost respectively.

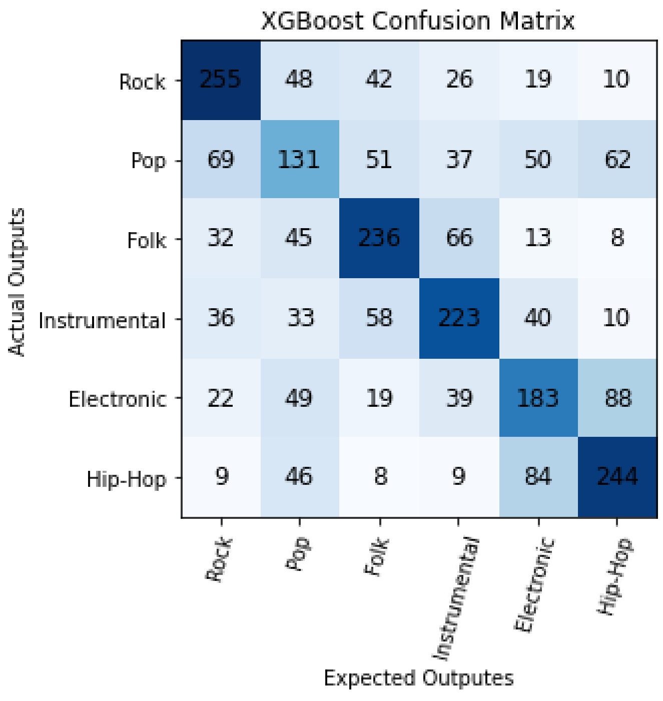
Accuracies obtained: When the trained models were used to predict the genre of the music, we have obtained an accuracy around 62% with the SVM classifier vs an accuracy of around 56% on the testing data using the XGBoost classifier over the PCA.

A confusion matrix was drawn for both the classifiers after the classification was done on the train set in order to analyse the results of the classification. The confusion matrices for both are given below:



SVM Confusion Matrix

Actual Outputs \ Expected Outputes	Rock	Pop	Folk	Instrumental	Electronic	Hip-Hop
Rock	270	39	28	28	23	12
Pop	62	150	51	32	57	48
Folk	26	41	248	67	14	4
Instrumental	33	31	40	259	32	5
Electronic	16	46	15	36	229	58
Hip-Hop	13	34	4	4	103	242



XGBoost Confusion Matrix

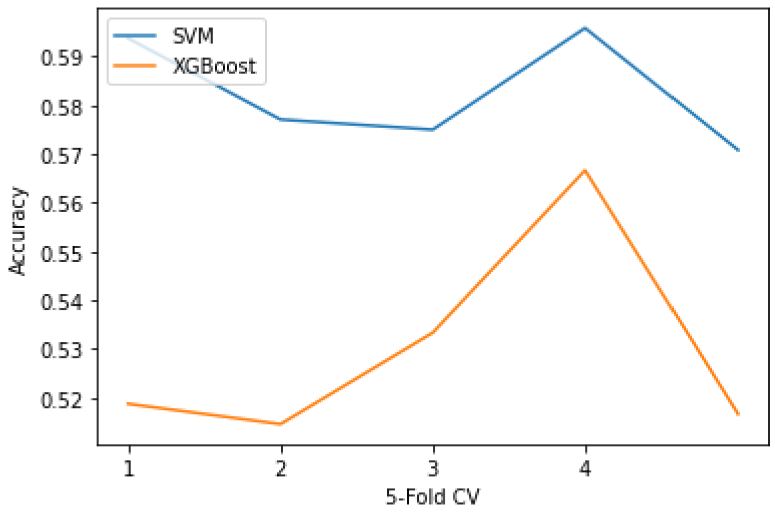
Actual Outputs \ Expected Outputes	Rock	Pop	Folk	Instrumental	Electronic	Hip-Hop
Rock	255	48	42	26	19	10
Pop	69	131	51	37	50	62
Folk	32	45	236	66	13	8
Instrumental	36	33	58	223	40	10
Electronic	22	49	19	39	183	88
Hip-Hop	9	46	8	9	84	244

Results and discussion: We can infer from the obtained confusion matrices that both the classifiers have been very good at classifying the Rock, Hip-Hop, folk and instrumental genres.

SVM particularly has been relatively better than all the data representation and the classifiers including XGBoost. From all the above observations, we can observe that, the classifier still struggled to classify the pop genre correctly. We have got more accuracy with SVM than XGBoost. Usually XGBoost has better performance accuracy than SVM but choosing kernel like RBF will work well for some problems but not every problem.

Bias: Both classifiers SVM and XGBoost are not able to differentiate most of sample from class pop and some from electronic. Classifiers are biased towards other classes when they encounter samples from those classes.

We have done a 5-fold cross validation on either of the classifiers and plotted the graph for accuracies at each of the folds in the 5-fold CV. The plotted graph is shown below:



We can observe that at each and every fold in the 5-fold CV, SVM have given us fairly better accuracies over the XGBoost.

Confidence Intervals:

At 99% Confidence Interval:

- True Classification Accuracy of SVM is likely between 55.779000197118165 and 60.970999802881835

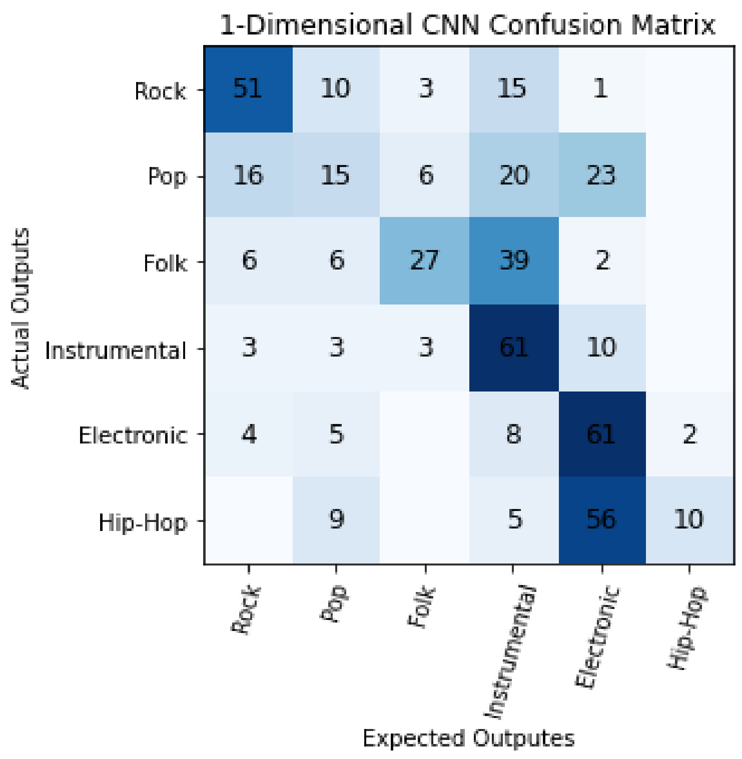
- True Classification Accuracy of XGBoost is likely between 50.0789976406353 and 55.337669026031364.

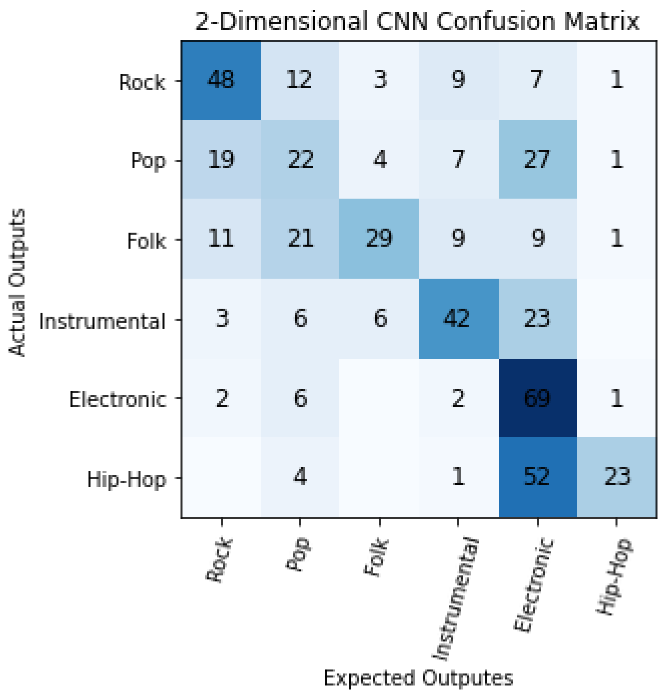
4. <u>Spectrograms: 1D CNN vs 2D CNN</u>

We have first modified our given music data into our first data representation format which was Spectrograms. We have used the Spectrograms and have tried to classify them with two different classifiers, which are 1 Dimensional CNN and 2 Dimensional CNN respectively.

Accuracies obtained: When the trained models were used to predict the genre of the music, we have obtained an accuracy around 49% with the 1D CNN classifier vs an accuracy of around 52% on the testing data using the 2D CNN classifier over the spectrograms.

A confusion matrix was drawn for both the classifiers after the classification was done on the train set in order to analyse the results of the classification. The confusion matrices for both of these are given below:





Results and discussion: For validation we have taken only 480 samples of data. From the confusion matrix above we can see that Electronic is more accurately classified in both 1D and 2D CNN. Also, Instrumental is classified with similar accuracy as Electronic in 1D CNN. Pop and Hip-Hop are least accurately classified in both. Electronic is often highly confused with Hip-Hop. In Validation phase, 1D CNN gives more accuracy but when testing 2D CNN gives more accuracy. It is observed that overfitting of the data is happening.

Confidence Intervals:

At 99% Confidence Interval:

- True Classification Accuracy of 1-D CNN is likely between 40.99849379347622 and 52.75150620652378

- True Classification Accuracy of 2-D CNN is likely between 42.65615415822875 and 54.42717917510458

Future Enhancements:

The training data that we were given were only 2400 instances for 6 different genres of music. We believe that the accuracy could have been improved if we had more data to train. We also think that by considering extra features like the artist, album release year will also help in accurate classification. Building CNN-RNN models also might give a better performance as RNNs are good in understanding of sequential data by making the hidden state at t-1 time dependent on the hidden state at time t-2 and they can do a good job in recognizing the long term and short term temporal features in the song. We believe that more systematic hyper-parameter tuning could potentially improve the overall performance of our models.

References:

[1] Bahuleyan, Hareesh. "Music genre classification using machine learning techniques." arXiv preprint arXiv:1804.01149 (2018).

[2] S. Vishnupriya and K. Meenakshi, "Automatic Music Genre Classification using Convolution Neural Network," 2018 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, 2018, pp. 1-4, doi: 10.1109/ICCCI.2018.8441340.

[3] A. Elbir, H. O. İlhan, G. Serbes and N. Aydın, "Short Time Fourier Transform based music genre classification," 2018 Electric Electronics, Computer Science, Biomedical Engineerings' Meeting (EBBT), Istanbul, 2018, pp. 1-4, doi: 10.1109/EBBT.2018.8391437.

Links:

https://librosa.github.io/librosa/