# Music Genre Detection

Janam Patel
Drexel University
Philadelphia, PA, USA
jp3777@drexel.edu

Natasha Lalwani
Drexel University
Philadelphia, PA, USA
nl498@drexel.edu

Nishchala Barde
Drexel University
Philadelphia, PA, USA
nb932@drexel.edu

## ABSTRACT

Music genre recognition (MGR) is a part of music information retrieval (MIR) and audio signal processing research. Humans established music genres as a way to categorize different types of music. The said categorization system serves as a foundation for developing a powerful recommendation system. However, due to their inherent subjective characteristics, music genres are difficult to categorize and explain in a systematic and consistent manner. In this paper we present a comparative study of genre classification models; Machine Learning models (Logistic Regression, Naive Bayes, K-Nearest Neighbors, Decision Tree, Support Vector Machines) and Deep Learning models (Artificial Neural Network and Convolutional Neural Network). Our aim is to train these models using *mfcc* features (for ML models, ANN and MLP) and *melspectrograms* (CNN model) and test them on GTZAN dataset. We also compared the performance of thirty-seconds duration features with three-seconds duration features, dataset. Our dataset is classified into 10 genres; blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, rock. *ANN* provided the best accuracy at *92.87*% on three-seconds features dataset.

## Keywords

Music Genre Recognition, Machine Learning Models, Deep Learning Models, MFCCs, Melspectrograms

## 1. INTRODUCTION

Music genre recognition is a particularly fascinating topic of research in the field of music information retrieval and audio signal processing [2]. Classifying genre for music allows us to create new content like playlist creation. The goal of automating music classification is to make song selection more efficient and less time-consuming. If one has to manually classify songs or music, one has to first listen to a large number of tracks before deciding on a genre. Not only is this time-consuming, but it's also complicated. Music classification software can make it easier to identify essential information like trends, popular genres, and performers. The first step in this strategy is to identify music genres.

A lot of work has been done in the field of music genre detection incorporating the use of various machine learning and deep learning models. [6] made use of different features such as Spectral Centroid (SC), Spectral Roll-off (SR), Spectral Flux (SF), Zero Crossing Rate (ZCR) and Mel Frequency Cepstral Coefficients (MFCC) to train Gaussian Mixture Model (GMM) and K-Nearest Neighbors (KNN) Model. This work was further explored by [1]. Particularly they made use of spectrograms and mfcc features in order to train three classification models; K-NN, Modeling each category as a gaussian distribution and Support Vector Machines. They classified three genres; rock, classical and jazz. They obtained highest accuracy from KNN Model at 75%. Ndou et al. provided a comparative study between deep learning and classical machine learning models [4]. Among ML models like Logistic Regression, Random Forest, Support Vector Machines, k-Nearest Neighbors and Naive Bayes, k-Nearest Neighbors gave the highest accuracy; 92.69%. Between the two DL models implemented; Multilayer Perceptron and CNN, an accuracy of 81.73% was obtained from MLP. [5] worked mainly with Convolutional Neural Network. More specifically they trained two CNN models; a user-defined CNN model and a pre-trained convnet. Their aim was to classify three genres from the GTZAN dataset; blues, classical and rock. The pre-trained convnet gave better accuracy. Another paper that explored the used of CNN for music genre classification is [7]. They trained their CNN network using melspectrograms and mfcc features. Training the model using melspectrograms gave a better accuracy of 76%. Haggblade et al. also provided a comparative study between ML Models and Neural Networks [3]. They purely relied on MFCCs to train k-NN, k-means, multi-class SVM and neural network (CNN). They classified four genres; classical, jazz, metal and pop.

Our study looks into automatic music genre classification with the goal of demonstrating that machine-learning and deep learning approaches can be used to classify music based solely on the audio signal, reducing the time it takes to find music pieces in the massive music databases that have sprung up as a result of digital music platforms. We compare the performance of machine learning classifiers implemented with three-second duration features to those implemented with thirty-second duration features. The same is done with deep learning classifiers. We also compare the

performance of deep-learning classifiers to classical machine-learning models.

# 2. BACKGROUND

## 2.1 Artificial Neural Network

The structure of an Artificial Neural Network (ANN), simply known as Neural Network, is inspired by the human brain as it resembles the way biological neurons communicate with one another. An ANN comprises of an input layer (also know as input node to which input/ information is provided to in order to learn and derive conclusions from out model. It passes the data to the hidden layer), one or more hidden layers (consists of a set of neurons where all the computation is performed on the input data) and an output layer (contains the model's output/conclusions produced from all calculations/computation. It is binary classification problem when there is 1 output node, while in a multi-class classification problem, the output nodes might be more than 1). Each node, or artificial neuron, is connected to the next and has a weight and threshold associated with it. If a node's output exceeds a certain threshold, the node is activated, and data is sent to the next layer of the network. If this is not the case, no data is sent on to the network's next layer. Among various variations in ANN, Multi-Layer Perceptron is a common one. It basically consists of multiple hidden layers. Each node, with the exception of the input nodes, is a neuron with a nonlinear activation function. MLP is distinguished from a linear perceptron by its multiple layers and non-linear activation. It can tell the difference between data that isn't linearly separable. More specifically, we have implemented a Multi-Layer Perceptron.

## 2.2 CNN

Convolutional Neural Networks (ConvNet/CNN) are a type of Deep Learning algorithm that are distinguished from other neural networks by their superior performance with detection and classification of images. There are three main layers: Convolutional Layer, Pooling Layer and Fully Connected (FC) Layer. This is the initial stage in obtaining useful information from an image. The convolution action is performed by many filters in a convolution layer. Every image is seen as a pixel value matrix. Say we have a $5*5$ image and a filter matrix with dimension $3*3$, we slide the filter matrix over the image in order to compute the dot product so as to obtain the convolved feature matrix. This is then passed on to the ReLU layer which performs element wise operation and sets all the negative pixels to 0. We obtained a rectified feature map which is next fed to the pooling layer in order to obtain a pooled feature map.. Various filters are used by this layer to identify different parts of the image. The resultant 2 dimensional array is flattened into a single long continuous linear vector. This is then fed to the FC layer which leverages the softmax function in order to perform the task of classification. With each layer, the CNN increases in its complexity, identifying greater portions of the image. Earlier layers focus on simple features, such as colors and edges. As the image data progresses through the layers of the CNN, it starts to recognize larger elements or shapes of the object until it finally identifies the intended object. Our model is also based on the same architecture.

## 2.3 ML Models

### *2.3.1 Logistic Regression*

It's a predictive technique similar to Linear Regression but with a difference that it uses independent factors to predict the dependent variable, but the dependent variable must be categorical.It is a statistical model that uses logistic function in order to model the conditional probability. It can be used for both, binary classification as well as multi-class classification. For binary classification issues, which are problems with two class values, such as predictions like "this or that," "yes or no,", "A or B," logistic regression is one of the most widely used machine learning techniques. The goal of logistic regression is to estimate event probabilities, which includes establishing a relationship between variables and the likelihood of specific outcomes.

### *2.3.2 Naive Bayes*

A naive Bayes classifier is a simple probabilistic classifier with strong assumptions of independence. An advantage of the naive bayes classification is that it requires only a small amount of training data to estimate the parameters required for classification. The Bayesian classification assumes that the data belongs to a particular class. We then calculate the probability that the assumption is true. In our experimentation, we have used Gaussian Naive Bayes model. When dealing with continuous data, a typical assumption is that the continuous values associated with each class are distributed according to a normal (or Gaussian) distribution. Here the probability density of some observation value v given a class $C_k, p(x = v|C_k)$, can be computed by plugging v into the equation for a normal distribution parameterized by $\mu_k$ and $\sigma_k^2$.

### *2.3.3 K-Nearest Neighbors*

It is a simple, easy-to-implement technique that may be used to address both classification and regression issues. However, in the industry, it is mostly utilized to solve classification and prediction problems. KNN is a lazy learning algorithm because it does not have a dedicated training phase and instead uses all of the data for training and classification. Because it makes no assumptions about the underlying data, KNN is a non-parametric learning algorithm [23]. This algorithm predicts the values of new data points based on 'feature similarity', which implies that the new data point will be assigned a value depending on how closely it matches the points in the training set. The 'k' in the KNN algorithm is based on feature similarity. The process of selecting the appropriate value for K is known as parameter tuning, and it is critical for improved accuracy.

### *2.3.4 Random Forest Classifier*

Random forest is a supervised learning technique that can be used to classify and predict data. However, it is mostly employed to solve classification problems. A forest, as we all know, is made up of trees, and more trees equals a more healthy forest. Similarly, the random forest method constructs decision trees from data samples, extracts predictions from each, and then votes on the best option. It's an ensemble method that's superior than a single decision tree because it averages the results to reduce over-fitting. Random forests perform better than a single decision tree for a wide range of data items. The variance of a random

forest is lower than that of a single decision tree. Random forests are incredibly adaptable and have a high level of accuracy. The random forest algorithm does not require data scaling. Even after supplying data without scaling, it maintains a high level of accuracy. Even when a major amount of the data is missing, the Random Forest algorithms maintain high accuracy.

#### 2.3.5 *Decision Tree Classifier*

ID3 Decision tree classifier is a simple decision tree learning algorithm developed by Ross Quinlan (1983). The basic idea of ID3 algorithm is to construct the decision tree by employing a top-down, greedy search through the given sets to test each attribute at every tree node. In order to select the attribute that is most useful for classifying a given sets, we use a metric-information gain. To find an optimal way to classify a learning set we need some function which provides the most balanced splitting. The information gain metric is such a function. Given a data table that contains attributes and class of the attributes, we can measure homogeneity of the table based on the classes. The index used to measure degree of impurity is called Entropy.

#### 2.3.6 *Support Vector Machines*

Support Vector Machine, or SVM, is a linear model that can be used to solve classification and regression issues. It can solve both linear and nonlinear problems and is useful for a wide range of applications. SVM is a basic concept: The method divides the data into classes by creating a line or hyperplane. SVM technique is used to find the points from both classes that are closest to the line. These points are knows as support vectors. The distance between the line and the support vectors is now computed. The distance calculated is known as margin. Our goal is to maximize the margin. The ideal hyperplane is the one for which the margin is the maximum. As a result, SVM seeks to create a decision boundary with as much separation between the two classes as possible. Important parameters of SVM are $C$ and $Gamma$. C is in charge of the trade-off between a smooth decision border and correctly classifying training points. A large C number indicates that more correct training points will be received. Gamma specifies the extent to which a single training example has an impact. If it has a low value, it signifies that each point has a long reach, whereas if it has a high value, it suggests that each point has a short reach. We trained the model using optimal values. Since ours is a multi-class classification problem, we made use of Support Vector Classifier.

## 3. METHODOLOGY

Throughout this section, we will explain the methods used for every part of our problem, going from the extraction and pre-processing of features (MFCC and Melspectograms), to the implementation of our classification models. We will go through every model in details, including a discussion on the choice of parameters and methods.

### 3.1 Data

We used the GTZAN dataset on Kaggle which is the most-used public dataset for evaluation in machine learning research for music genre recognition (MGR). The dataset files were collected in 2000-2001 from a variety of sources including personal CDs, radio, microphone recordings, in order to represent a variety of recording conditions. The data folder on Kaggle includes three subfolders: 1. genres original - A collection of 10 genres with 100 audio files each, all having a length of 30 seconds. 2. images original - The audio files were converted to Mel Spectrograms to classify data through neural networks. 3. Two CSV files - Containing features of 30sec and 3secs audio files. The audio files are split into 10 genres - blues, classical, country, disco, hiphop, jazz, metal, pop, regae, rock.

### 3.2 Feature Extraction

#### 3.2.1 *Mel Frequency Cepstral Coefficients (MFCC)*

When working with audio files, especially those including human recordings and speeches, Mel Frequency Cepstral Coefficients (MFCC) is a critical component to be considered. The shape of the vocal tract determines any sound produced by humans (including tongue, teeth, etc.). If this shape is correctly determined, any sound produced can be accurately represented. The envelope of the time power spectrum of a speech signal represents the vocal tract, and MFCC accurately represents this envelope. The Mel scale relates a pure tone's perceived frequency, or pitch, to its actual measured frequency. At low frequencies, humans are much better at detecting small changes in pitch than at high frequencies. By incorporating this scale, we can better match our features to what humans hear. In order to obtain MFCCs from the typical frequency values, the signals are converted from Hertz Scale to Mel Scale. A frequency measured in Hertz (f) can be converted to the Mel scale using the following formula:

$$Mel(f) = 2595 \log(1 + \frac{f}{700}) \tag{1}$$

Other features like chromagram, melspectrogram, spectral contrast, tonnetz are extracted as well from the audio so that the emotion can be determined efficiently. In order to be able to do this, there one such package in python known as Librosa. Not only does it serve as a necessary building elements for the creation of music information retrieval systems, but also aids in the visualization of audio signals as well as feature extraction utilizing various signal processing techniques. With the Librosa module we loaded every single .wav file, and extracted the MFCC features for each of them. We then computed the mean of these features, obtaining 20 MFCCs for each audio file, and inserted these features in a dataframe along with the *Speaker Label*.

| | mfcc1_mean | mfcc2_mean | mfcc3_mean | mfcc4_mean | mfcc5_mean | mfcc6_mean |
|---|---|---|---|---|---|---|
| 0 | -113.570648 | 121.571793 | -19.168142 | 42.366421 | -6.364664 | 18.623499 |
| 1 | -207.501694 | 123.991264 | 8.955127 | 35.877647 | 2.907320 | 21.510466 |
| 2 | -90.722595 | 140.446304 | -29.093889 | 31.684334 | -13.984504 | 25.764742 |
| 3 | -199.544205 | 150.090897 | 5.662678 | 26.859079 | 1.771399 | 14.234031 |
| 4 | -160.337708 | 126.219635 | -35.587811 | 22.148071 | -32.478600 | 10.852294 |

5 rows × 21 columns

**Figure 1: Sample of the dataframe containing 20 MFCCs for each genre**

### 3.2.2 *Melspectrograms*

Models that usually take images as input, such as CNN, can also be used to work with audio data. In order to do so, we need to compute the melspectogram of each .wav file, and use this image as input into our model. A spectrogram is a visual depiction of a signal's frequency composition over time. The Mel scale provides a linear scale for the human auditory system. The mel spectrogram remaps the values in hertz to the mel scale. The Mel spectrogram is used to provide our models with sound information similar to what a human would perceive. The raw audio waveforms are passed through filter banks to obtain the Mel spectrogram. The melspectrogram images initially had size of $288 * 432$, but we reduced them to $64 * 64$. Below is the representation of mel spectrograms obtained from one single audio file per music genre in our dataset.

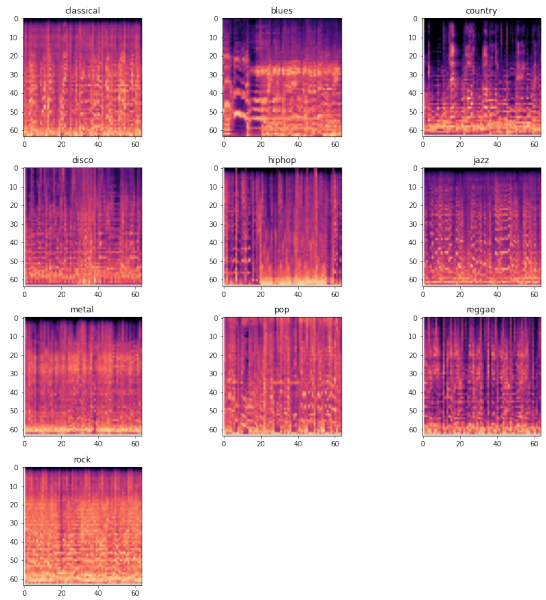


**Figure 2: Melspectrogram sample of each genre**

## 3.3 ANN

The first model we implemented for the classification part is an Artificial Neural Network. We tried various MLP architectures on both the datasets; thirty-seconds duration features and three-seconds duration features. We varied the number of hidden layers and activation functions in order to compare the performance. We set the parameters of final output layer of this model to 10 as the output size, referring to the ten different genres of this classification problem, and $Softmax$ as the activation fucntion. The $Softmax$ function turns numbers into probabilities that sum to one. It outputs an array that represents the probability distributions of a set of potential outcomes. Because it is a multi-class classification task, we decide to use $Crossentropy$ as the loss function, and the $Adam$ optimizer. The $Adam$ optimizer is a replacement optimization algorithm for the stochastic gradient descent in deep learning models. $Adam$ combines the best features of the AdaGrad and RMSProp algorithms to create an optimization algorithm that can deal with sparse gradients on noisy problems.

In Fig. 3 a block diagram of one the architectures implemented by us. For the others, the number of hidden layers, activation function and other parameters were changed which is show in the *results* section.

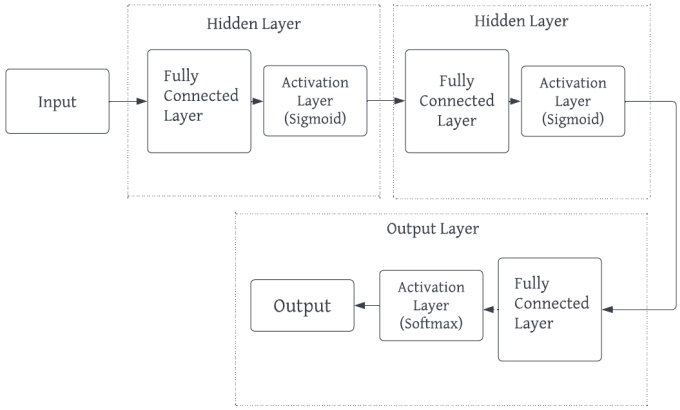


**Figure 3: Block diagram of ANN architecture**

## 3.4 CNN

As previously mentioned, it is possible to use Deep Learning algorithms, which typically work with images, for audio recognition problems. In this alternative approach for the classification part of our problem, we transformed all the audio files into melspectrograms, and used these images as input for a Convolutional Neural Network Model. For our model, we decided to include two convolutional layers, two pooling layers, one dense layer and one final fully connected layer before obtaining the output. For both convolutional layers we used a kernel of spacial size $3*3$, with strides equal to $(2, 4)$, and size as 16 in the first one and 32 in the second one. The activation function used in these two layers was $Relu$, a linear function that outputs the input directly if it is positive, otherwise, it outputs zero. For both pooling layers, we used Max Pooling operation with pool size of $2*2$. After each convolutional and pooling layer, we also added a Batch Normalization layer, which standardizes the inputs for each mini-batch. Before the output layer we also included a flattening layer and another dense layer of size 64, incorporating a $Relu$ activation function. The flattening layer converts the input into a 1-dimensional array before outputting it to the next layer. The final output layer has, as the output size parameter, the number of classes we are trying to predict, and as activation function a $Softmax$, which outputs an array that represents the probability distributions of a set of potential outcomes. The loss function used for this model was the $Sparse\ Categorical\ Crossentropy$, and $ADAM$, Adaptive Moments as optimizer. When fitting the model to our training dataset, all the melspectrograms obtained from the audio samples, we set the learning rate to 0.01, and run the model for 100 iterations. $Figure$ 4 shows the block diagram of the CNN model implemented by us.

## 3.5 Machine Learning Models

As a part of our comparative study, we implemented various classification models; Logistic Regression, Gaussian Naive Bayes, k-Nearest Neighbors, Random Forest, Decision Tree

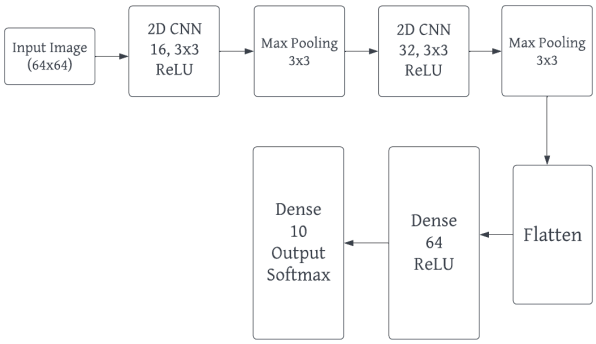


**Figure 4: Block diagram of our CNN model**

and Support Vector Machine, to classify the 10 genres. We performed grid search for hyperparamter tuning in order to obtain the best model along with the optimal hyperparamters that works best on our dataset so as to get a good accuracy.

## 4. RESULTS

The results obtained from the deep learning models were more promising than the ones obtained form the machine learning models. Not only did we compare the models, but also compared the performance between two datasets; thirty-seconds duration features and three-seconds duration features.

We implemented various Multi-Layer Perceptron models by changing different parameters. The tables below summarize the performance of the three models implemented by us.

| 30 seconds | | | |
|---|---|---|---|
| Epochs | ETA | Training Accuracy (%) | Testing Accuracy (%) |
| 1000 | 0.001 | 96.43 | 91.13 |
| 1000 | 0.001 | 92.29 | 87.67 |
| 100 | 0.03 | 100 | 89.33 |

**Table 1: Three best model's performance for 30 seconds feature**

| 3 seconds | | | |
|---|---|---|---|
| Epochs | ETA | Training Accuracy (%) | Testing Accuracy (%) |
| 1000 | 0.001 | 93.59 | 92.61 |
| 1000 | 0.001 | 94.3 | 92.87 |
| 100 | 0.03 | 82.04 | 81.92 |

**Table 2: Same hyperparameter ran onto 3 seconds feature**

*Figure* 5 shows the graph of *Epochs vs Loss* (on the left) and variation in training and validation accuracy (on the right), for the thirty-seconds features dataset.

*Figure* 6 shows the graph of *Epochs vs Loss* (on the left)

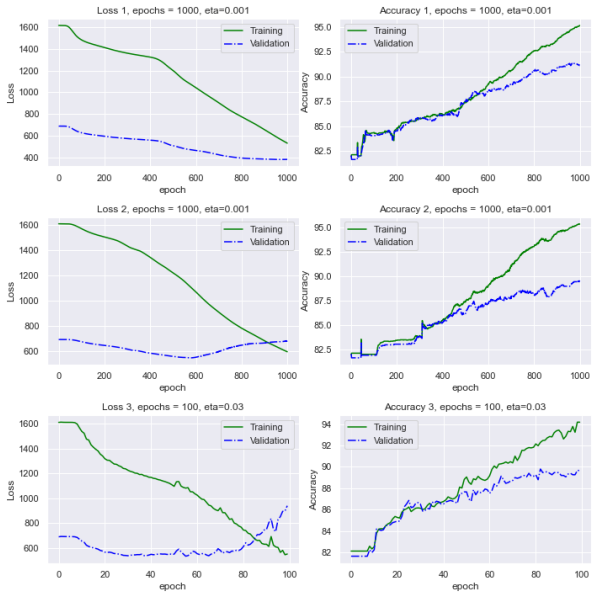


**Figure 5: Epoch vs Loss Variation in Accuracy**

and variation in training and validation accuracy (on the right), for the three-seconds features dataset.

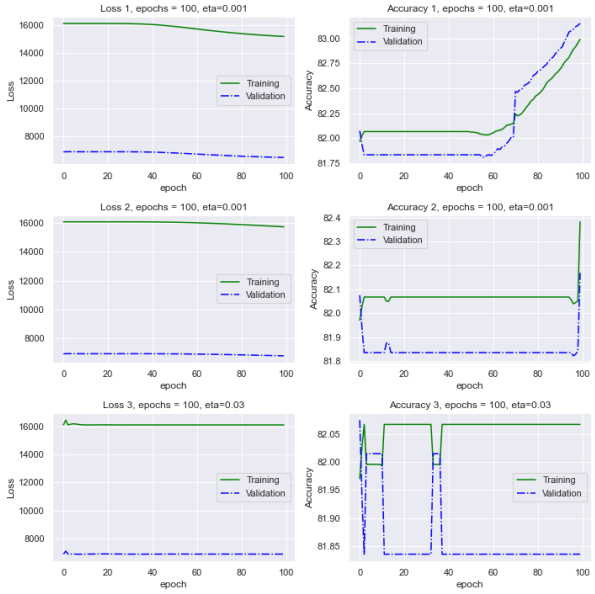


**Figure 6: Epoch vs Loss Variation in Accuracy**

The Convolutional Neural Network gave us decent results in terms of accuracy but, by looking at the plot below, it is clear that while the training accuracy increases at a consistent rate until it reaches 100%, the validation accuracy does not increase after a certain point and is constant. The overall behaviour of the curve is decent giving us an accuracy of 70.56% after 100 iterations.

On the other hand, from all the Machine Learning models

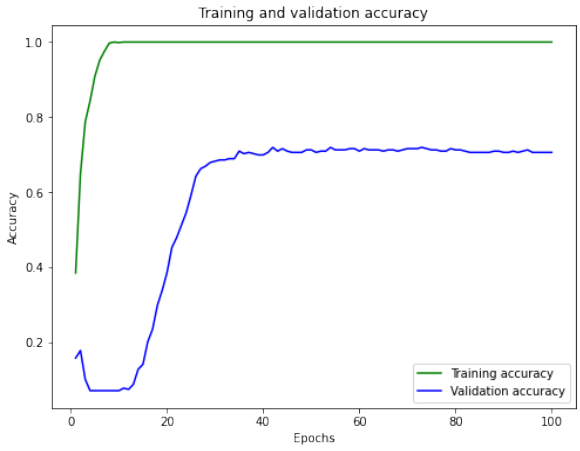


**Figure 7: Curve of Training and Validation Accuracy**

implemented, *Support Vector Classifier* gave us the best results on the thirty-seconds features dataset and *k-Nearest Neighbors* gave us best results on the three-seconds features dataset. We got an accuracy of 63% from SVC and 87% from k-NN.

## 5. CONCLUSION

In this paper, music genre classification is studied using the GTZAN dataset. We proposed a simple approach to solving the classification problem and we drew comparisons with multiple other complex, robust models. We also compared the models based on the kind of input it was receiving. Two kinds of inputs were given to the models: Melspectrogram images for CNN models and audio features stored in a csv for ML, ANN and MLP model. **Multi-Layer Perceptron** was determined to be the best feature based classifier amongst ML and ANN models with a test accuracy of **92.87**%, on three-seconds duration features data file. Since CNN model works best with images and mel spectrograms are images, we implemented a CNN model as well, which gave us an accuracy of **70.56**%. Our CNN model is expected to perform well if the dataset size is increased. Overall, feature based classification is seen to be performing better than image based classification.

## 6. LIMITATIONS AND FUTURE SCOPE

Across all models, using $mfcc$ based ANN produced higher accuracy results. The frequency based mel-spectrograms are visual, and CNNs work better with pictures. We expected the CNN to perform the best however,the one that we implemented from scratch took the longest time to train. Due to limited time to implement, hence limited time for training the data model, we had to reshape the images to reduce its dimensions which may also have affected its performance. Thus, the accuracy of CNN was lower than the mfcc based ANN model. With a larger dataset, we hope our CNN model will perform better. In the future, we hope to experiment with other types of deep learning methods. Given that this is time series data, some sort of RNN model may provide better results as well. We hope to experiment with models like LSTM as well. We are also curious about generative aspects of this project, including some sort of genre conversion. Additionally, we suspect that we may have opportunities for transfer learning, for example in classifying music by artist or by decade.

## 7. REFERENCES

[1] H. Deshpande, R. Singh, and U. Nam. Classification of music signals in the visual domain. In *Proceedings of the COST-G6 conference on digital audio effects*, volume 1, pages 1–4. Citeseer, 2001.

[2] D. Ghosal and M. H. Kolekar. Music genre recognition using deep neural networks and transfer learning. In *Interspeech*, pages 2087–2091, 2018.

[3] M. Haggblade, Y. Hong, and K. Kao. Music genre classification. *Department of Computer Science, Stanford University*, 2011.

[4] N. Ndou, R. Ajoodha, and A. Jadhav. Music genre classification: A review of deep-learning and traditional machine-learning approaches. In *2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, pages 1–6, 2021.

[5] M. Nirmal and S. Mohan. Music genre classification using spectrograms. In *2020 International Conference on Power, Instrumentation, Control and Computing (PICC)*, pages 1–5. IEEE, 2020.

[6] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.

[7] S. Vishnupriya and K. Meenakshi. Automatic music genre classification using convolution neural network. In *2018 international conference on computer communication and informatics (ICCCI)*, pages 1–4. IEEE, 2018.