# MUSIC-GENRE CLASSIFIER

ANJALI
20010101

# INTRODUCTION

- The Music-genre classifier categorizes music tracks into predefined genres based on their audio characteristics.

- It utilizes machine learning and deep learning algorithms to automate the process traditionally performed by human experts.

- This project has two models, namely the Multi-Layer Perceptron (MLP) and Convolutional Neural Network (CNN) architectures.

# OBJECTIVE

- The primary goal of our project is to automatically categorize music into distinct genres based on the inherent patterns and features within the audio signal like MFCCs lever

- Explore the effectiveness of Mel-frequency cepstral coefficients (MFCCs) and other audio features in improving the accuracy of music genre classification models.

# LIBRARIES USED

- **Librosa:** Used for loading audio data and extracting relevant features.
- **Tensorflow:** Fundamental for creating and training the model.
- **Sci-kit learn:**Required for various data preprocessing tasks.
- **Numpy:** Essential for matrix calculations and numerical operations.
- **Matplotlib:**Necessary for visualizing and plotting the model’s results.

# DATASET

- Our model is based on the GTZAN Dataset - Music Genre Classification dataset, which is a collection of audio clips spanning various genres, making it valuable for evaluating and training machine learning models in the field of music genre classification.
- It has 10 data labels:  blues, classical, country, disco, hiphop, jazz, metal, pop, raggae and rock.
- Each class consists of 100 audio files of 30 seconds length.
- The data also has the MFCC plots of all the audio files saved for image-based processing, like CNN.

# PROCESS FLOW

1. Load the preprocessed data
2. Build the model architecture
3. Address the overfitting issues
4. Compile the model
5. Split the data into train and test sets
6. Train the model
7. Evaluate the model
8. Make prediction for an audio file

# DATA PREPROCESSING

- A dictionary named data is made to store the 'mapping', 'labels; and 'MFCCs' of the data.
- Each audio file is divided to 10 independent segments, each of length 3 seconds to increase the input data.
- We loop through the dataset to get the mapping, i.e, the name of the subfolder, labels as subfloder-count - 1.
- The mfcc is calculated using the feature of the librosa library.
- All the mappings, labels and mfccs are then stored in the JSON file for further use.

# MULTILAYER PERCEPTRON

01

- A Multilayer Perceptron (MLP) is a type of artificial neural network designed for supervised learning. It consists of multiple layers of nodes, including an input layer, one or more hidden layers, and an output layer.
- MLP's architecture allows it to learn and differentiate between diverse genres by capturing nuanced patterns present in the audio features, enhancing its classification capabilities.
- MLP, when applied to audio data, can effectively extract higher-level features from representations like Mel-frequency cepstral coefficients (MFCCs), contributing to the model's ability to discern audio characteristics.

# IMPLEMENTATION OF MLP

- The MLP model consists of an input layer, three hidden layers and one output layer.
- It is a Sequential model which is created with the help of Keras of the TensorFlow library.
- All the hidden layers use the Dense layer with 512, 256 and 64 neurons and ReLU as the activation function.
- For the output layer, it uses the softmax function.

# CONVOLUTION NEURAL NETWORK

- Convolutional Neural Network (CNN) is a deep learning architecture designed for image and spatial data, featuring convolutional layers that extract hierarchical representations, pooling layers for dimensionality reduction, and fully connected layers for classification.
- It can perrform better than MLP even with less parameters.

# IMPLEMENTATION OF CNN

- The model comprises three convolutional layers, each employing a 3x3 grid for convolution, followed by 3x3 max-pooling, and subsequent batch normalization to enhance feature extraction.
- The resultant is then flattened and fed to a Dense layer.
- For the output layer, the model utilizes a softmax function to generate a probability distribution across multiple classes.

# OVERFITTING ISSUE

- Overfitting occurs when a machine learning model learns the training data too well, capturing noise or random fluctuations and performing poorly on new, unseen data

- To prevent overfitting in both the models, we have used Dropout and L2 Regularization methods.

CUSTOMERS

- Dropout introduces randomness during training by randomly deactivating a fraction of neurons, preventing overfitting, while L2 regularization adds a penalty term to the loss function to limit the magnitudes of weights in a neural network.

# MODEL COMPILATION

- Model compilation is the process in deep learning where you configure the necessary settings for a neural network before training.
- This includes specifying the optimizer, loss function, and metrics that the model will use during training and evaluation.
- Our model have Adam optimizer with a learning rate of 0.0001 and is compiled with sparse categorical crossentropy as the loss function and accuracy as the evaluation metric.

# PERFORMANCE

- Our models are trained on the training set using the fit method with a batch size of 32 and 30 epochs for CNN and 50 epochs for MLP.
- The model's performance is assessed on the separate test set using the evaluate method, calculating the test loss and accuracy.The printed test accuracy provides a quantitative measure of the model's generalization to new, unseen data.
- The test accuracy for MLP is 0.594125509262085.
- The test accuracy for CNN is 0.7140568494796753.

# MODEL PREDICTION

- The model can predict the label for an audio file using the custom predict function.
- It takes the model, audio file input and desired label as the input parameters.
- It first adds another layer to the input and pass it through the model.predict method.
- It gives a probability distribution of the possibility of the audio data’s label.
- The label with the highest probability is taken as the predicted answer.