Music Genre Classification

Vinayak Verma, Kartik Choudhary

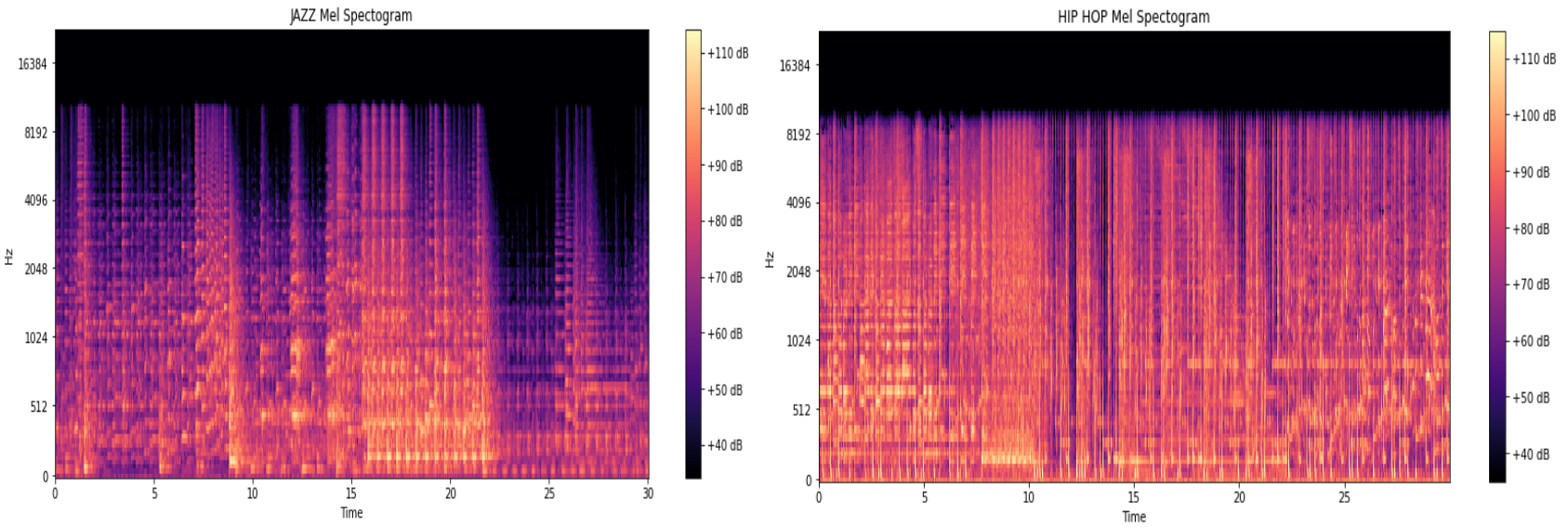
B20EE079, B20CS025

Abstract - This paper reports our experience with building a Music Genre Classifier. We have a dataset containing genres, images original, and two CSV files containing features of the audio files. Genres original - A collection of 10 genres with 100 audio files each, all having a length of 30 seconds. Images Original- A visual representation of each audio file. We use various classification algorithms and compare their results in this report.

I. Introduction

Music Experts have been trying for a long time to understand sound and what differentiates one song from another. How to visualize sound? What makes a tone different from another? The goal of this project is to classify an audio file's genre from 10 different genres 'rock', 'classical', 'metal', 'disco', 'blues', 'reggae', 'country', 'hiphop', 'jazz', 'pop'.

The below images show the spectrograms of two genres.



Datasets

CSV: features3seconds.csv file is used as the training dataset.

The training dataset contains 9990 rows where each row contains 59 features of the audio file and one column containing the label of the audio file.

- Labels: Class Label 0-9 for 'rock', 'classical', 'metal', 'disco', 'blues', 'reggae', 'country', 'hiphop', 'jazz', 'pop' respectively.
- For the classification, the dataset has been split into train and test with a test size of 0.3.

II. Methodology

Overview

There are various classification algorithms present out of which we shall implement the following

- *Gaussian Naive Bayes*
- *Stochastic Gradient Classifier*
- *KNN*
- *Decision Trees*
- *Random Forest Classifier*
- *Multinomial Logistic Regression*
- *SVM Classifier*
- *XGBoost Classifier*
- *MLP*
- *ANN*

We also make use of PCA for dimensionality reduction and to visualize a possible group of genres.

Exploring the Dataset and Pre-Processing

On counting the number of NULL values in the training dataset, it was found that there are no NULL values present. We have used librosa library to visualize all the genres of audio files. Various other crucial analyzing parameters such as Spectrogram, MEL Spectrogram, Chroma feature, Zero Crossing rate, spectral roll of, and spectral centroid. The data is normalized before feeding into the classifiers.

Implementation of Classification algorithms

- *Gaussian Naive Bayes:* Gaussian Naive Bayes is an algorithm having a Probabilistic Approach. It involves prior and posterior probability calculation of the classes in the dataset and the test data given a class respectively.
    - Default GaussianNB () is used for the classification.
- *Stochastic Gradient Classifier:* The class SGDClassifier implements a plain stochastic gradient descent learning routine that supports different loss functions and penalties for classification.
    - SGDClassifier (max_iter=5000, random_state=0) is used for classification.
- *KNN*: KNN is a supervised algorithm that classify on the basis of distance from similar points. Here k is the number of nearest neighbors to be considered in the majority voting process.
    - KNeighborsClassifier(n_neighbors=10) is used.
- *Decision Trees*
    - Default DTC is used.
- *Random Forest Classifier:* Random Forest Classifiers use boosting ensemble methods to train upon various decision trees and produce aggregated results.
    - RandomForestClassifier (n_estimators=1000, max_depth=10, random_state=0) is used.
- *Multinomial Logistic Regression:* Since the data is of multilabel type we have used multinomial logistic regression. Multinomial logistic regression is an extension of logistic regression that adds native support for multi-class classification problems.
    - LogisticRegression (random_state=0, solver='lbfgs', multi_class='multinomial') is used.
- *Support Vector Machine Classifier:* In SVM, data points are plotted into n-dimensional graphs which are then classified by drawing hyperplanes.
    - SVC linear kernel
    - SVC polynomial kernels degrees range from 2-to 7.
    - SVC RBF kernels with varying parameters
    - SVC sigmoid kernel
- *MLP:* MLP is a feedforward Neural Network which uses backpropagation to update weights and improve the results.
    - MLPClassifier (solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(5000, 10), random_state=1) is used.
- *XGBoost Classifier:* a supervised machine learning task in which one should predict if an instance is in some category by studying the instance's features.
    - XGBClassifier(n_estimators=1000, learning_rate=0.05)
- *ANNs*
    - Keras Sequential Models are used.

Dimensionality reduction techniques used: PCA.

IV. Evaluation of Models

The models implemented was evaluated using techniques like - Classification report: precision, recall, f1 score, and support , Confusion matrix, accuracy score.

Model	Accuracy Score
Gaussian Naive Bayes	0.52
Stochastic Gradient Descent	0.65
KNN	0.87
Decision Tree Classifier	0.65
Random Forest Classifier	0.81
Multinomial Logistic Regression	0.69
MLP	0.68
XBG Classifier	0.91
SVM Classifier	0.75

Rigorous analysis is done on SVM Classifier since it is expected to perform better on the multilabel high dimensional data.

SVM Model and Parameters	Accuracy Score
C=1.0,kernel='linear'	0.72
C=1.0,kernel='poly',degree=2	0.76
C=40-1000,kernel='rbf'	0.89-0.90
C=1.0,kernel='sigmoid'	0.15
kernel = poly d=6 c = 0.1	0.86
kernel = poly d=6 c = 0.2	0.87
kernel= poly d=6 c = 0.3	0.88
kernel = poly d=6 c = 0.4-1	0.89
C=160,kernel='sigmoid'	0.74
C=200,kernel='rbf'	0.91
rbf kernel c=200 gamma=0.01	0.75
rbf kernel c=200 gamma=0.02	0.78
rbf kernel c=200 gamma=0.03	0.79
rbf kernel c=200 gamma=0.04	0.8
rbf kernel c=200 gamma=0.05	0.81
rbf kernel c=200 gamma=0.06	0.82
rbf kernel c=200 gamma=0.07	0.83
rbf kernel c=200 gamma=0.08	0.84
rbf kernel c=200 gamma=0.09	0.85
rbf kernel c=200 gamma=0.1	0.86
rbf kernel c=200 gamma=0.2	0.89
rbf kernel c=200 gamma=0.3	0.9
rbf kernel c=200 gamma=0.4-0.7, 10	0.91
rbf kernel c=200 gamma=0.8-1	0.92
rbf kernel c=200 gamma=2	0.93
rbf kernel c=200 gamma=3-5	0.94
rbf kernel c=200 gamma=6-7	0.93
rbf kernel c=200 gamma=8-9	0.92

ANNs

- Keras Sequential Model Type1 (Optimizer = ‘adam’)

Number of Epochs	Loss at the end	Accuracy at the end	Loss on validation set at the end	Accuracy on validation set at the end	Test loss	Best test accuracy
600	0.1638	0.9454	0.4321	0.8849	0.432104378938 6749	88.48848938941 956

- Keras Sequential Type2 (Optimizer = ‘adam’, ‘adam’, ‘rmsprop’, ‘sgd’)
    - Model 1

Number of Epochs	Loss at the end	Accuracy at the end	Loss on validation set at the end	Accuracy on validation set at the end	Test loss	Best test accuracy
70	0.3840	0.8686	0.5826	0.8089	0.574860155582428	81.55053853988647

    - Model 2

Number of Epochs	Loss at the end	Accuracy at the end	Loss on validation set at the end	Accuracy on validation set at the end	Test loss	Best test accuracy
100	0.1785	0.9401	0.3472	0.8969	0.39008885622024536	89.10696506500244

    - Model 3

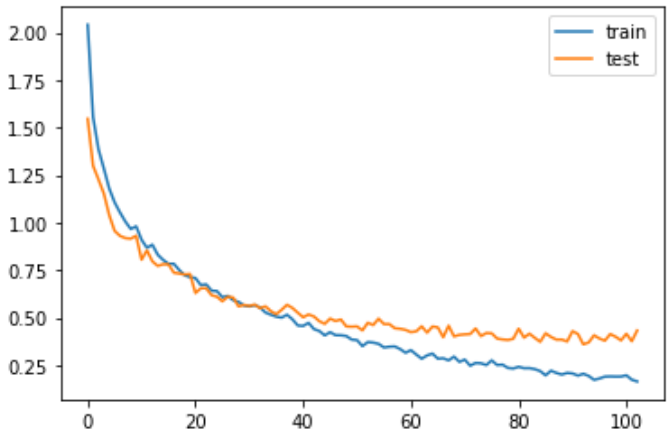
Number of Epochs	Loss at the end	Accuracy at the end	Loss on validation set at the end	Accuracy on validation set at the end	Test loss	Best test accuracy
700	0.3655	0.8777	0.4474	0.8625	0.446345716714859	86.45731210708618

    - Model 4

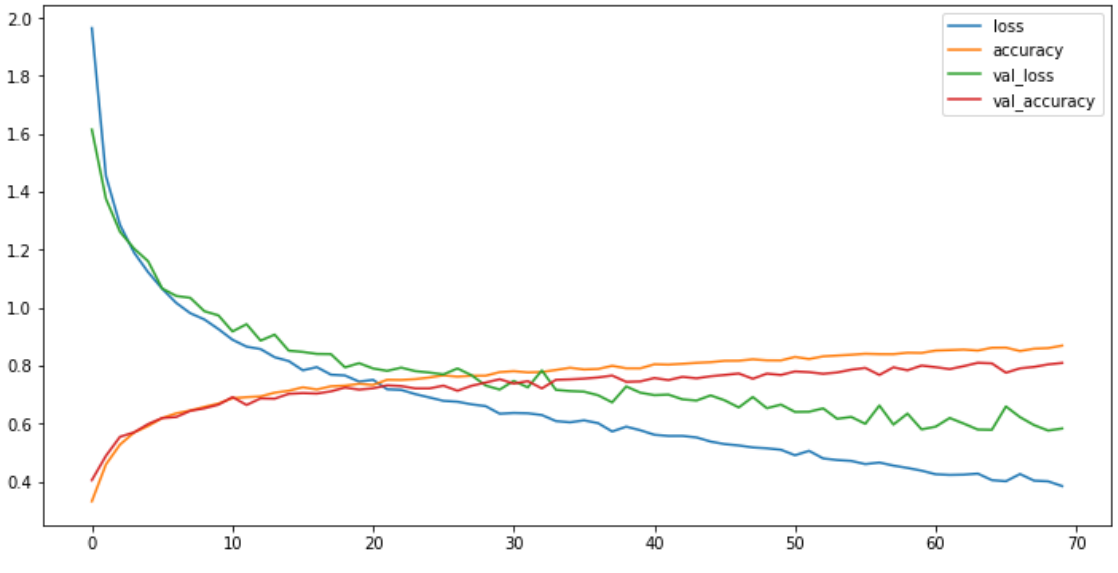
Number of Epochs	Loss at the end	Accuracy at the end	Loss on validation set at the end	Accuracy on validation set at the end	Test loss	Best test accuracy
500	0.1302	0.9658	0.5945	0.9095	0.5687476992607117	92.44357347488403

LOSS PLOTS

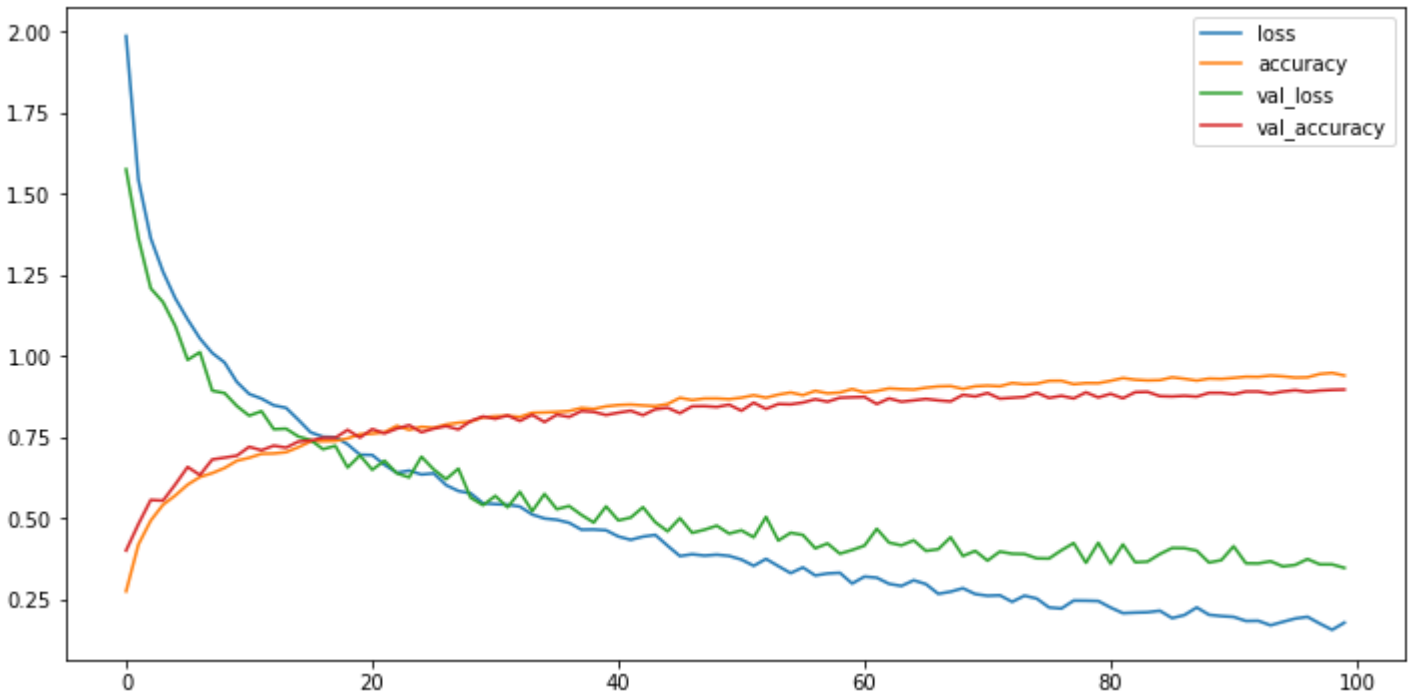
- **Keras Sequential Model Type1**



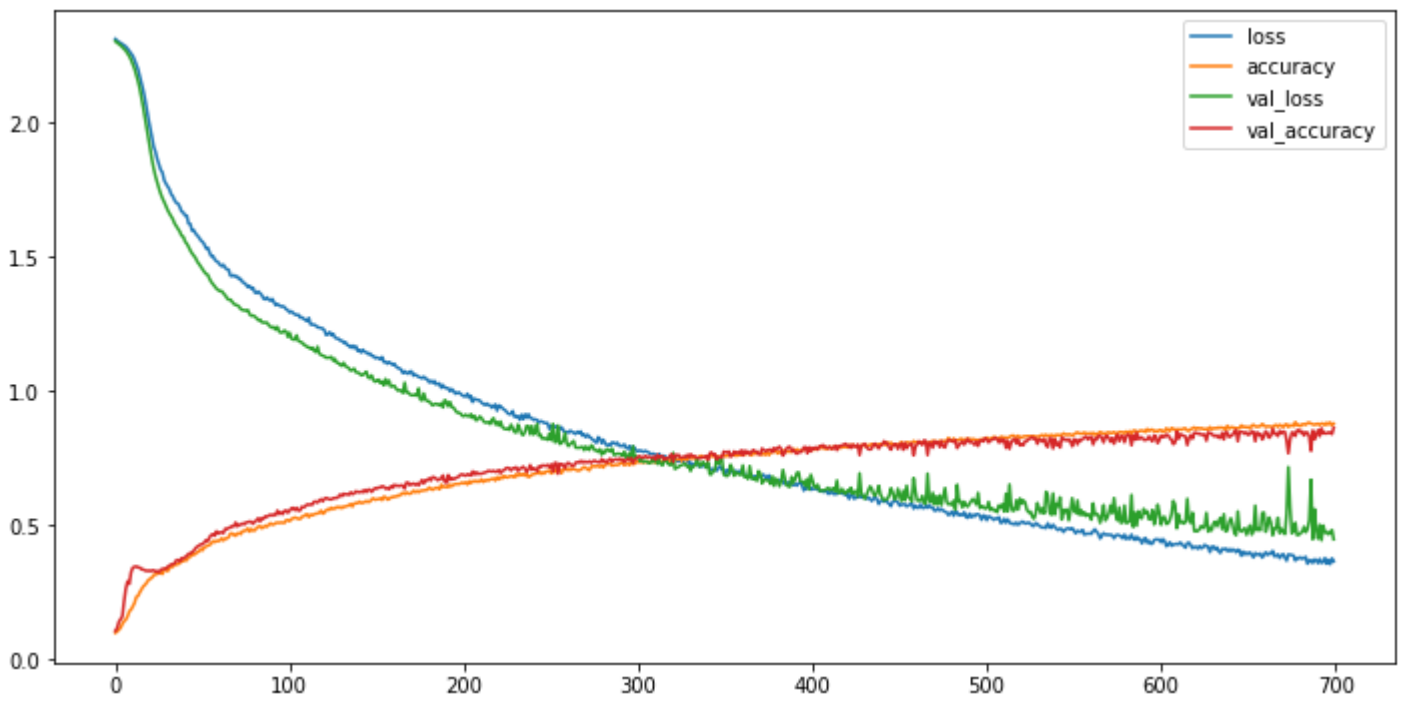
- **Keras Sequential Model Type 2**
    - *model1*



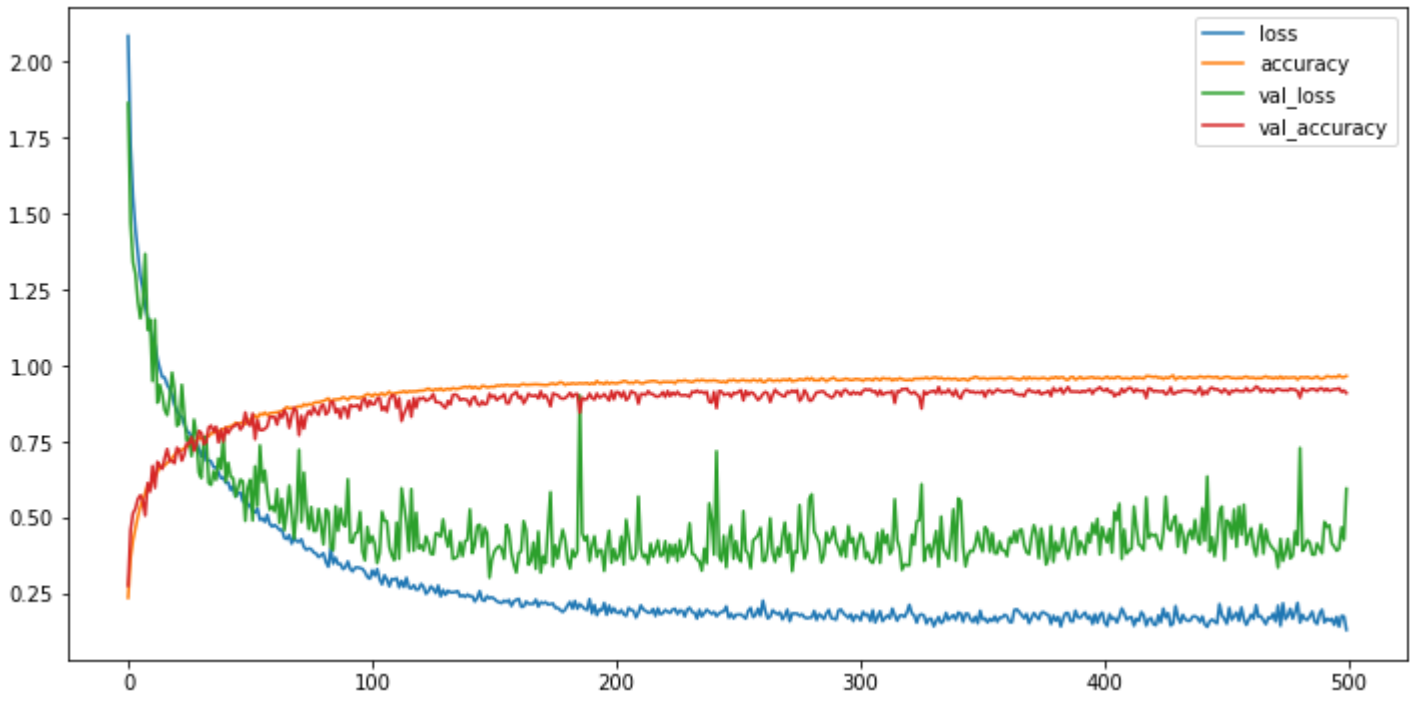
- *model 2*



- *model 3*



- *model 4*



V. RESULTS AND ANALYSIS

The table shows that among all the classifiers the XGBoost, SVM and the ANNs perform the best. The PCA is used to visualize possible groups of genres. The SVM performs exceptionally well for the classification. For the Keras sequential ANN type1, we used the Adam optimizer for training the model. The epoch that was chosen for the training model is 600. All of the hidden layers are using the RELU activation function and the output layer uses the softmax function. The loss is calculated using the sparse_categorical_crossentropy function. Dropout is used to prevent overfitting. We chose the Adam optimizer because it gave us the best results after evaluating other optimizers. The model accuracy can be increased by further increasing the epochs but after a certain period, we may achieve a threshold, so the value should be determined accordingly. For The Keras sequential ANN type2 we twitched the optimizers functions from Adam, SGD to rmsprop and the highest accuracy we achieved for the test set is 92.44 % using the 'rmsprop' optimizer and the accuracy is very decent. So we come to the conclusion that Neural Networks are very effective in machine learning models. However, among all the models, the SVM model outperforms all other models. Taking hyperparameters as kernel = 'RBF', c=200 and gamma {3, 4, 5} the accuracy comes out to be 94% which is the highest among all the models.

The reason for this is due to the high dimensionality and non-linearity of the data the radial basis function kernel of SVM can choose a non-linear decision boundary and effectively perform the classification for the high dimensional data. Thus, twitching the corresponding parameters (C and gamma) for the RBF kernel gives the highest accuracy on the test data.

VI. Additional

Additionally, Recomender system has been incorporated along with the Machine learning pipeline. "Recomender" Systems enable us for any given vector to find the best similarity, ranked in descending order, from the bast match to the least best match.

For Audio files, this will be done through cosine_similarity library.

VII. Contributions

The learning and planning were done as a team. The individual contributions are as given

- Kartik Choudhary[B20CS025]: ANNs, SVM, Report, preprocessing, and analysis of the models and Recommender system.
- Vinayak Verma [B20EE079]: GaussianNB, XGboost, Random Forest, Decision Tree, Multinomial Logistic Regression on GTZAN.

VIII. References

[1] https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification

[2] https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html

[3] https://scikit-learn.org/stable/modules/sgd.html

[4] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

[5] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[6] https://ieeexplore.ieee.org/document/9362364

[7]http://openclassroom.stanford.edu/MainFolder/DocumentPage.php?course=MachineLearning&doc=exercises/ex8/ex8.html