Music Genre Classification using Ensemble Learning

Course Code: CSE 472
Course Name: Machine Learning Sessional

Project Group Members:

1905102 - Md. Shafiul Haque
1905103 - Mayesha Rashid

Subsection: B2
Group No: 9

Project Supervisor: Dr. Atif Hasan Rahman

Problem Description:

Our task is to automatically classify music tracks into genres using audio features. Accurate genre classification can aid in music recommendation, music indexing, and personalized music services.

Dataset:

GTZAN Dataset

•**Source** - https://www.tensorflow.org/datasets/catalog/gtzan

•**Dataset Details:**

- Total Tracks: 1,000 audio files, 30 seconds each
- Genres: 10 (e.g., Blues, Classical, Jazz, Pop, Rock)
- Format: .wav files, organized into 10 folders for each genre

•**Basic Statistics:**

- Sample Rate: 22,050 Hz
- File Count per Genre: 100

Experimented Architectures:

- VGG 16 Transfer Learning CNN
- VGG 16 Fine Tuning CNN

- Logistic Regression
- Random Forest
- Gradient Boosting
- SVM (Support Vector Machine)
- Ensemble model of VGG 16 Fine Tuning and Gradient Boosting

Architecture Description:

Spectrogram generation:

From the given .wav files, we generated linear amplitude, log amplitude, log power, CQT transform MEL spectrograms using the librosa library of python. As we saw significant differences in log power spectrograms, we used MEL spectrogram images of this method for the CNN models.

Our VGG 16 architecture with a dropout layer was as following:

Model: "vgg16"

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1,792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36,928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73,856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147,584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295,168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590,080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590,080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1,180,160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2,359,808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2,359,808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0

Total params: 14,714,688 (56.13 MB)
Trainable params: 14,714,688 (56.13 MB)
Non-trainable params: 0 (0.00 B)

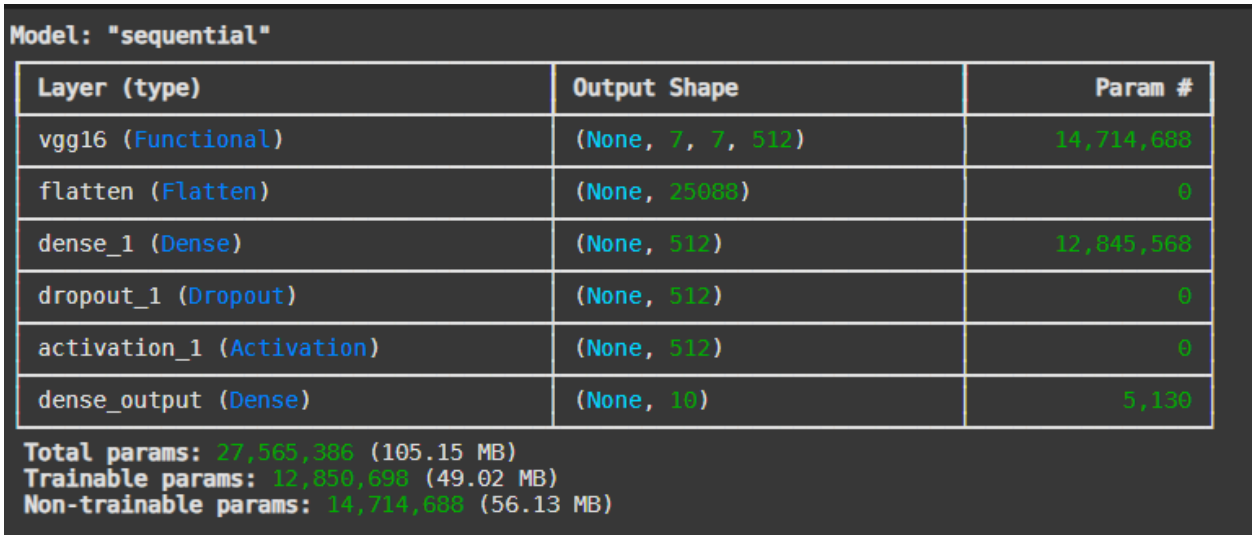
Model: "sequential"

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 7, 7, 512)	14,714,688
flatten (Flatten)	(None, 25088)	0
dense_1 (Dense)	(None, 512)	12,845,568
dropout_1 (Dropout)	(None, 512)	0
activation_1 (Activation)	(None, 512)	0
dense_output (Dense)	(None, 10)	5,130

Total params: 27,565,386 (105.15 MB)
Trainable params: 27,565,386 (105.15 MB)
Non-trainable params: 0 (0.00 B)

- **VGG 16 Transfer Learning:**

The weights in the convolution base are kept fixed but the weights in the feed-forward network are allowed to be tuned to predict the correct genre label.



Model: "sequential"

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 7, 7, 512)	14,714,688
flatten (Flatten)	(None, 25088)	0
dense_1 (Dense)	(None, 512)	12,845,568
dropout_1 (Dropout)	(None, 512)	0
activation_1 (Activation)	(None, 512)	0
dense_output (Dense)	(None, 10)	5,130

Total params: 27,565,386 (105.15 MB)
Trainable params: 12,850,698 (49.02 MB)
Non-trainable params: 14,714,688 (56.13 MB)

- **VGG 16 Fine Tuning:**

In this setting, we start with the pre-trained weights of VGG-16, but allow all the model weights to be tuned during the training process.

Model: "sequential"

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 7, 7, 512)	14,714,688
flatten (Flatten)	(None, 25088)	0
dense_1 (Dense)	(None, 512)	12,845,568
dropout_1 (Dropout)	(None, 512)	0
activation_1 (Activation)	(None, 512)	0
dense_output (Dense)	(None, 10)	5,130

Total params: 27,565,386 (105.15 MB)
Trainable params: 27,565,386 (105.15 MB)
Non-trainable params: 0 (0.00 B)

The spectrogram images have a dimension of 216 x 216. For the feed-forward network connected to the convolution base, a 512-unit hidden layer is implemented. Over-fitting is a common issue in neural networks. In order to prevent this, two strategies are adopted: L2-Regularization and Dropout. The dataset is randomly split into train (90%), validation (5%) and test (5%) sets. The same split is used for all experiments to ensure a fair comparison of the proposed models.

All models were trained for 10 epochs with a batch size of 32 with the ADAM optimizer.

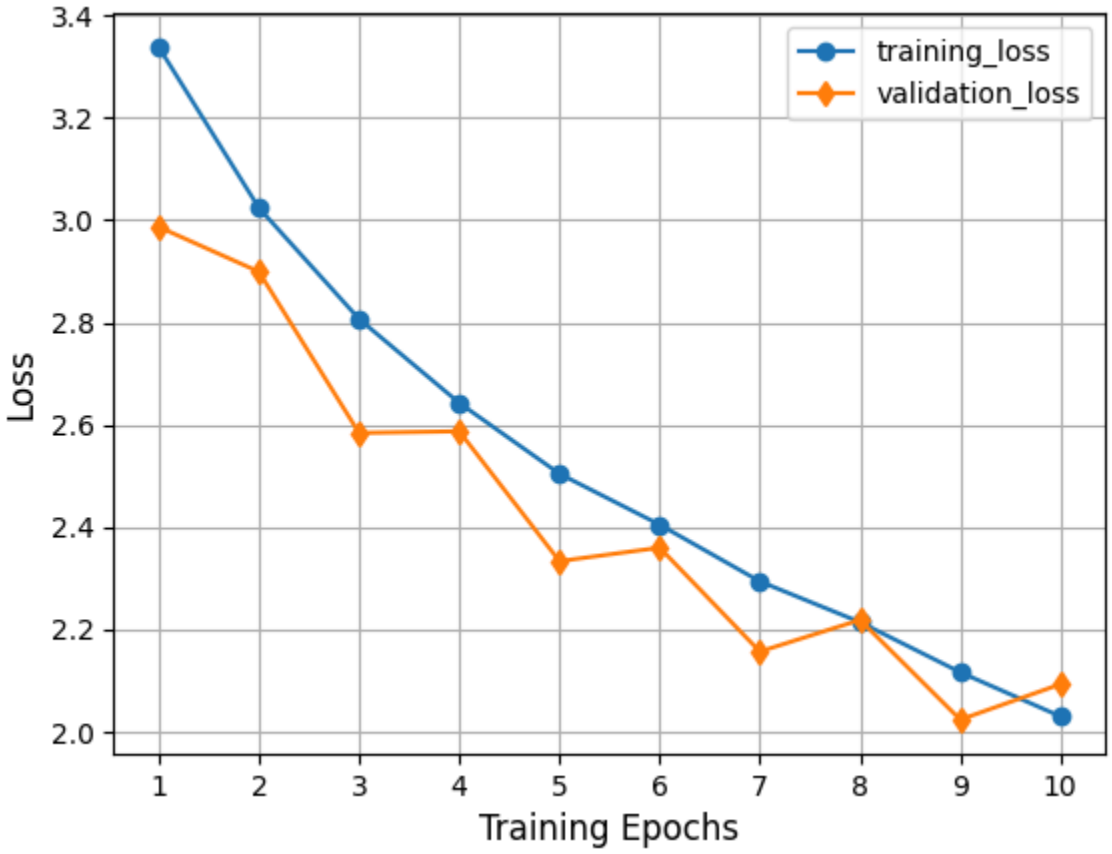
Audio Feature Extraction:

Time domain features like central moments, zero crossing rate, root mean square energy, tempo and frequency domain features like MFCC, Chroma features, spectral centroid, spectral bandwidth, spectral contrast, spectral rolloff were extracted from wav files using the Librosa library of python.

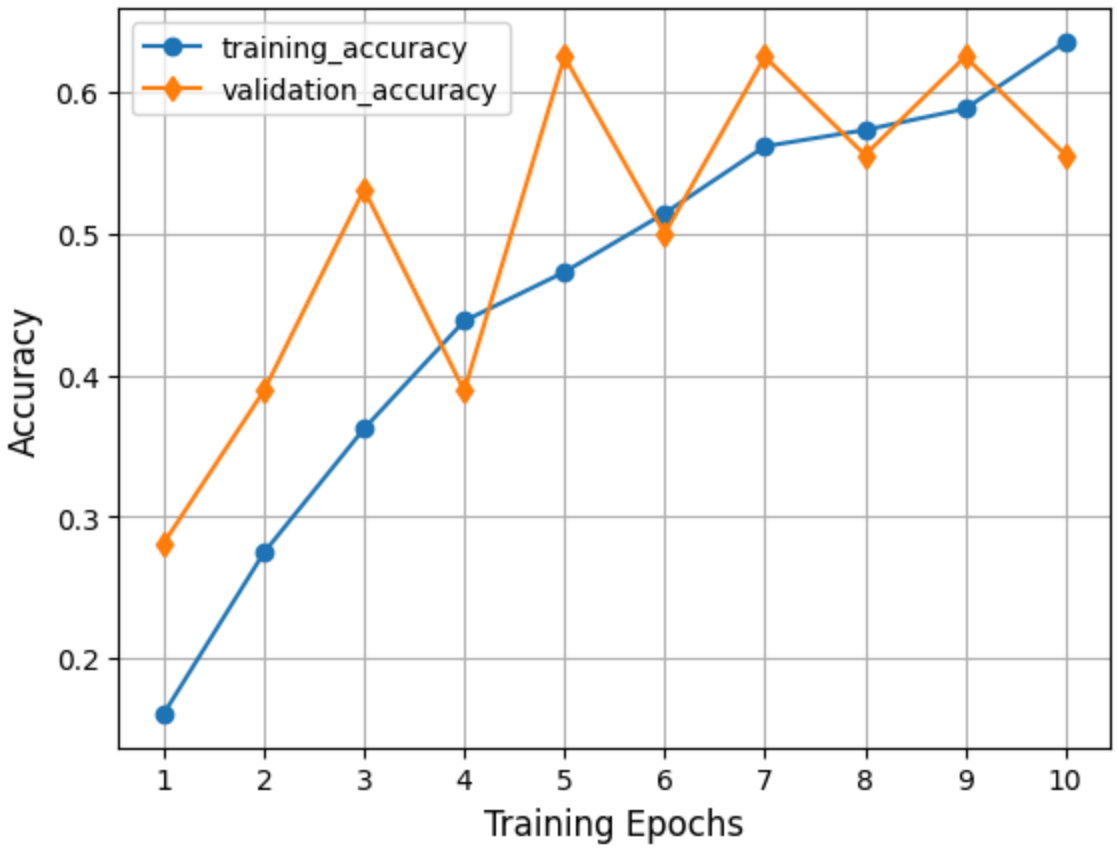
Logistic regression (LR), Random forest, gradient boosting, support vector machine classifiers were trained using these features and used for testing.

Final Result Analysis:

VGG Transfer Learning:

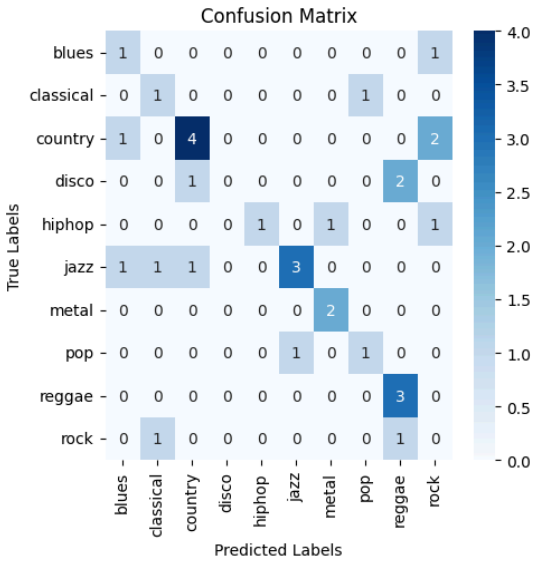


Losses during each epoch



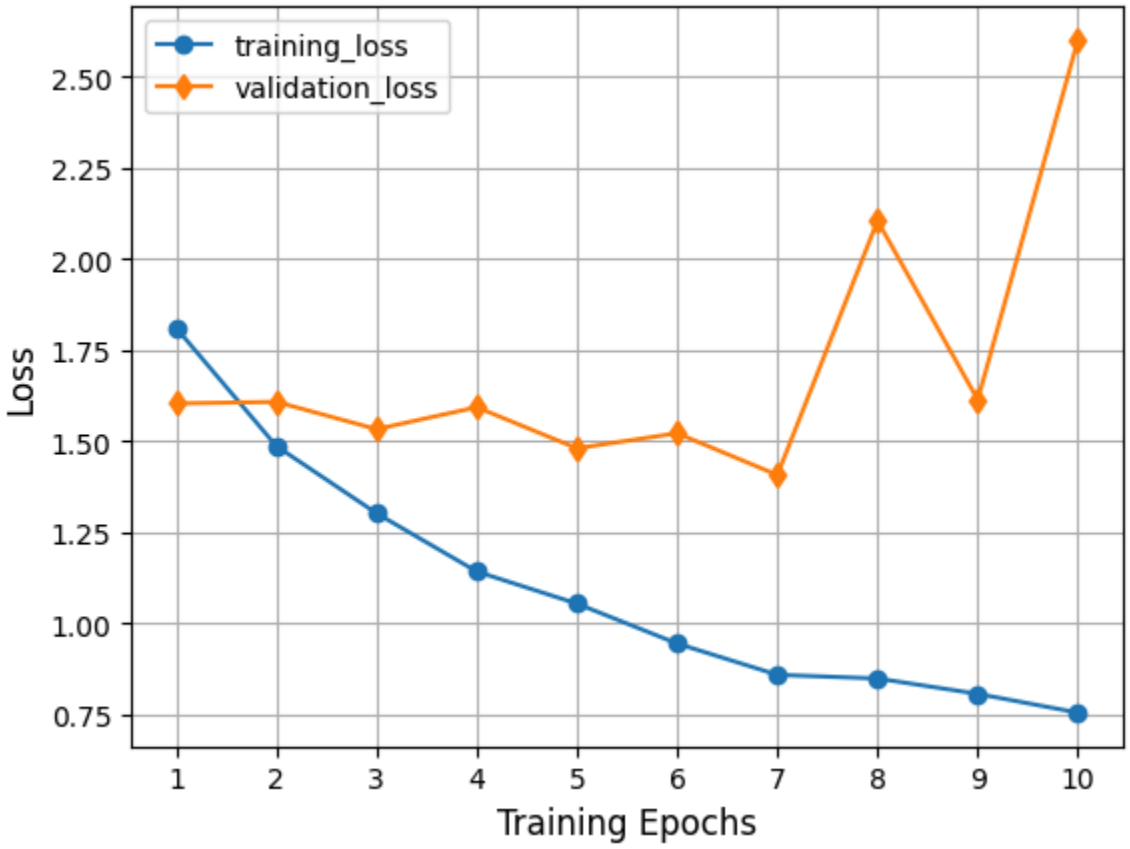
Accuracy during each epoch

Training the model with the 9th epoch weights, we got the confusion matrix

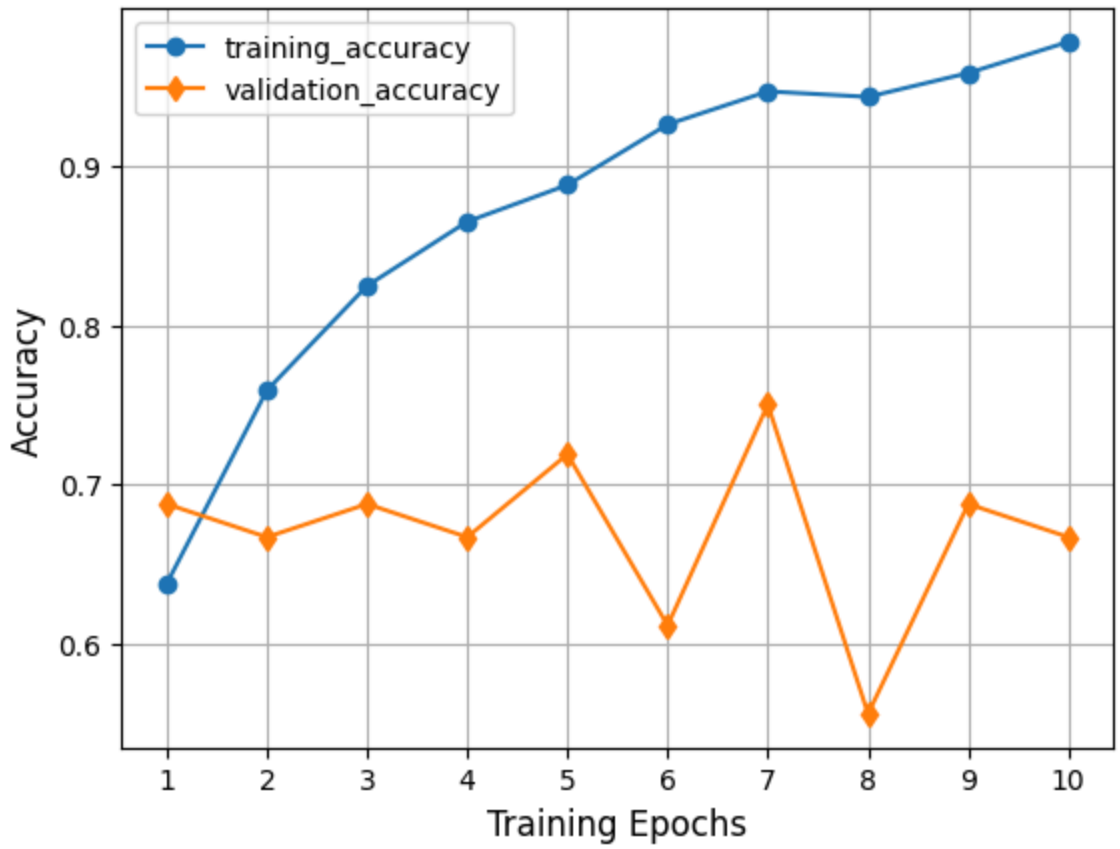


```
Test Set Accuracy =  0.50
Test Set F-score =  0.45
```

VGG 16 with Fine Tuning:

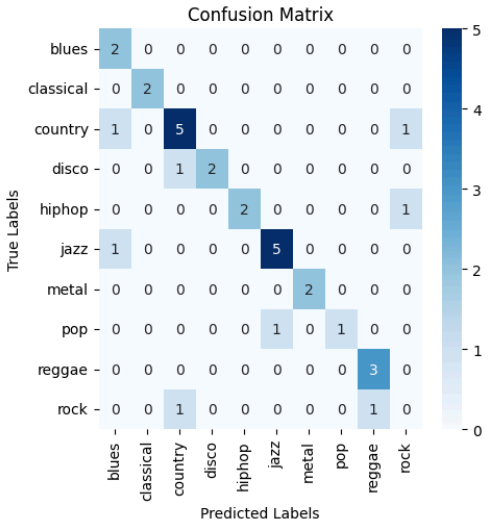


Losses during each epoch

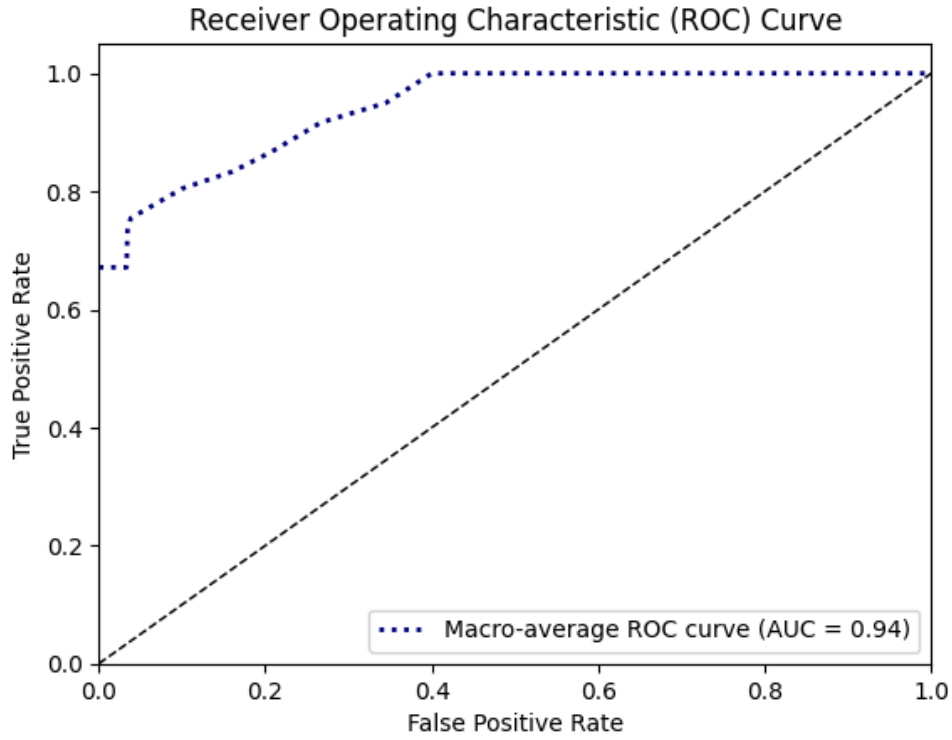


Accuracy during each epoch

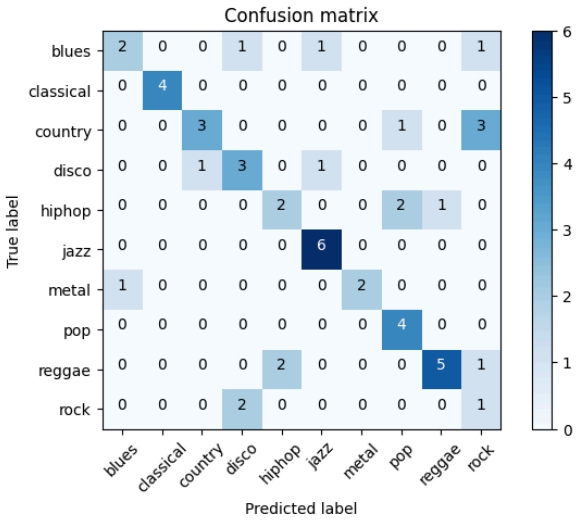
Training our model with 7th epoch, we get the following results:



```
Test Set Accuracy =  0.75
Test Set F-score =  0.73
ROC AUC = 0.936
```



Logistic Regression:

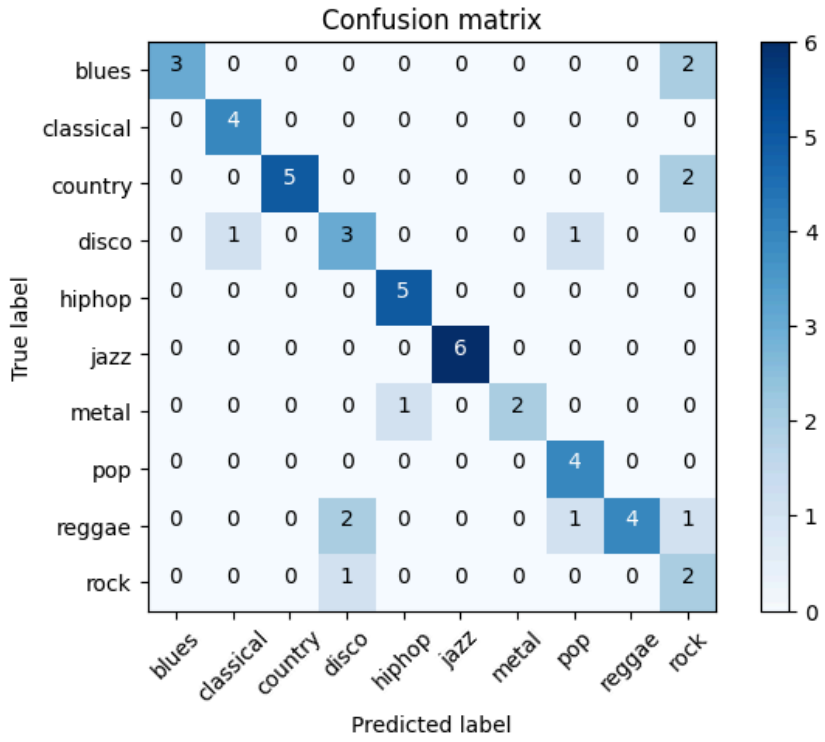



```
Test Set Accuracy =  0.64
```

Test Set F-score = 0.64

ROC AUC = 0.956

Random Forest:

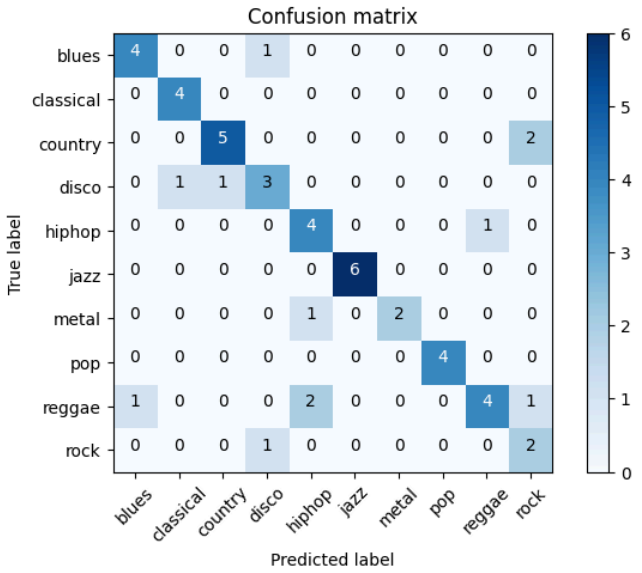


Test Set Accuracy = 0.76

Test Set F-score = 0.76

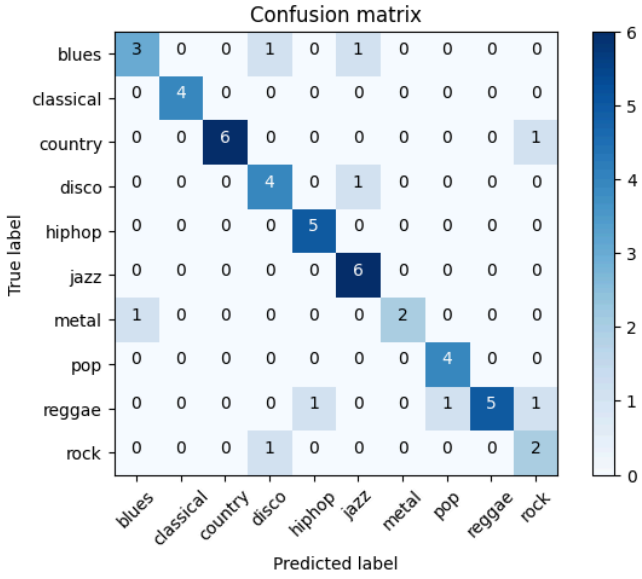
ROC AUC = 0.967

Gradient Boosting:



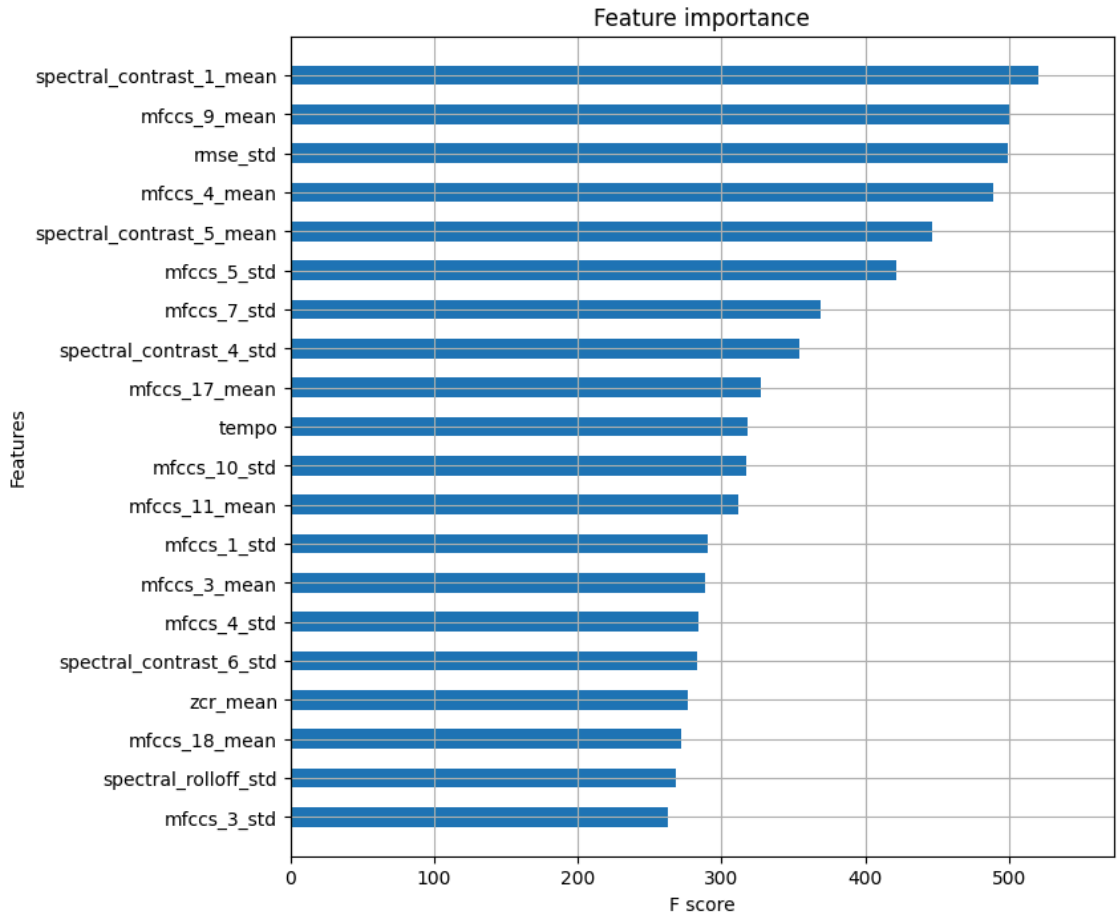

```
Test Set Accuracy =  0.76
Test Set F-score =  0.76
ROC AUC = 0.962
```

Support Vector Machine:



```
Test Set Accuracy =  0.82
Test Set F-score =  0.81
ROC AUC = 0.967
```

Feature importance in gradient boosting:



Stats for top 10 features:

```
Test Set Accuracy =  0.52
Test Set F-score =  0.54
ROC AUC = 0.882
```

Stats for top 20 features:

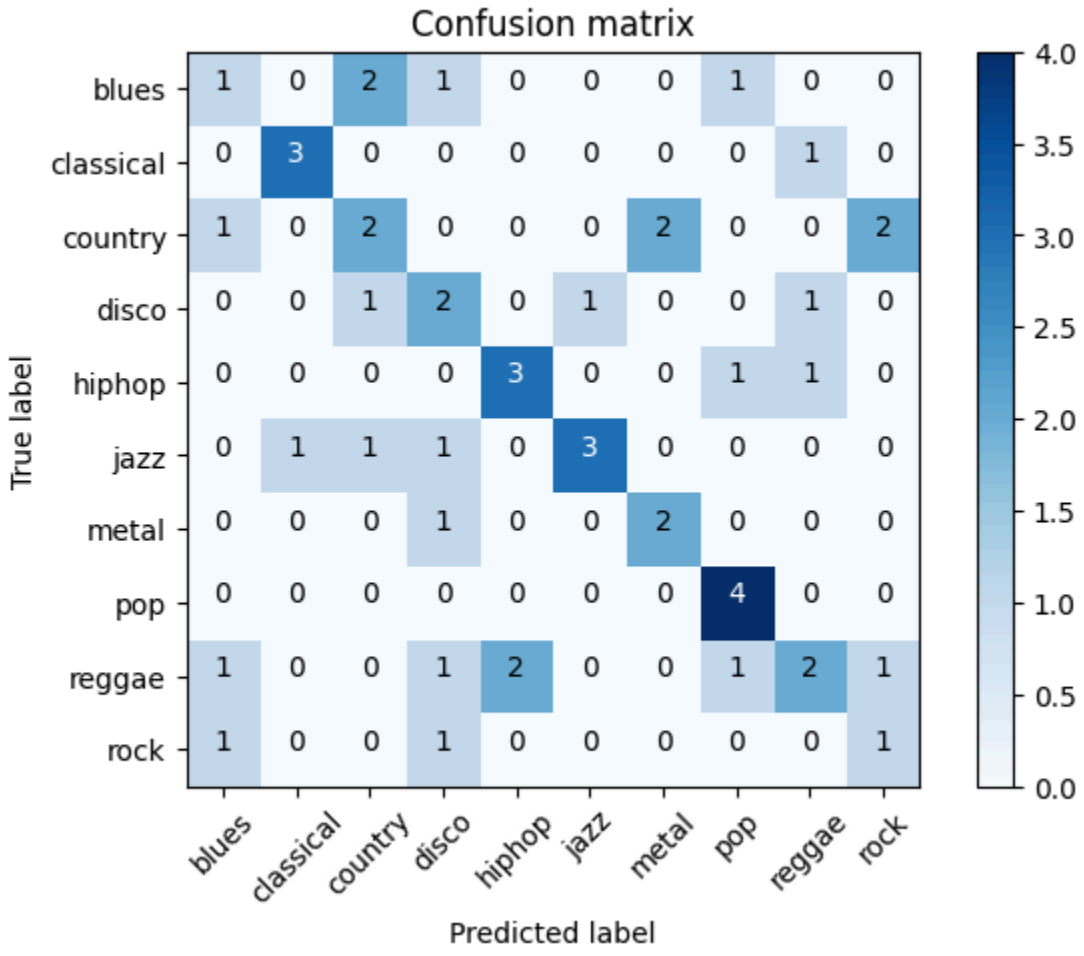
```
Test Set Accuracy =  0.66
Test Set F-score =  0.66
```

```
ROC AUC = 0.947
```

Stats for top 30 features:

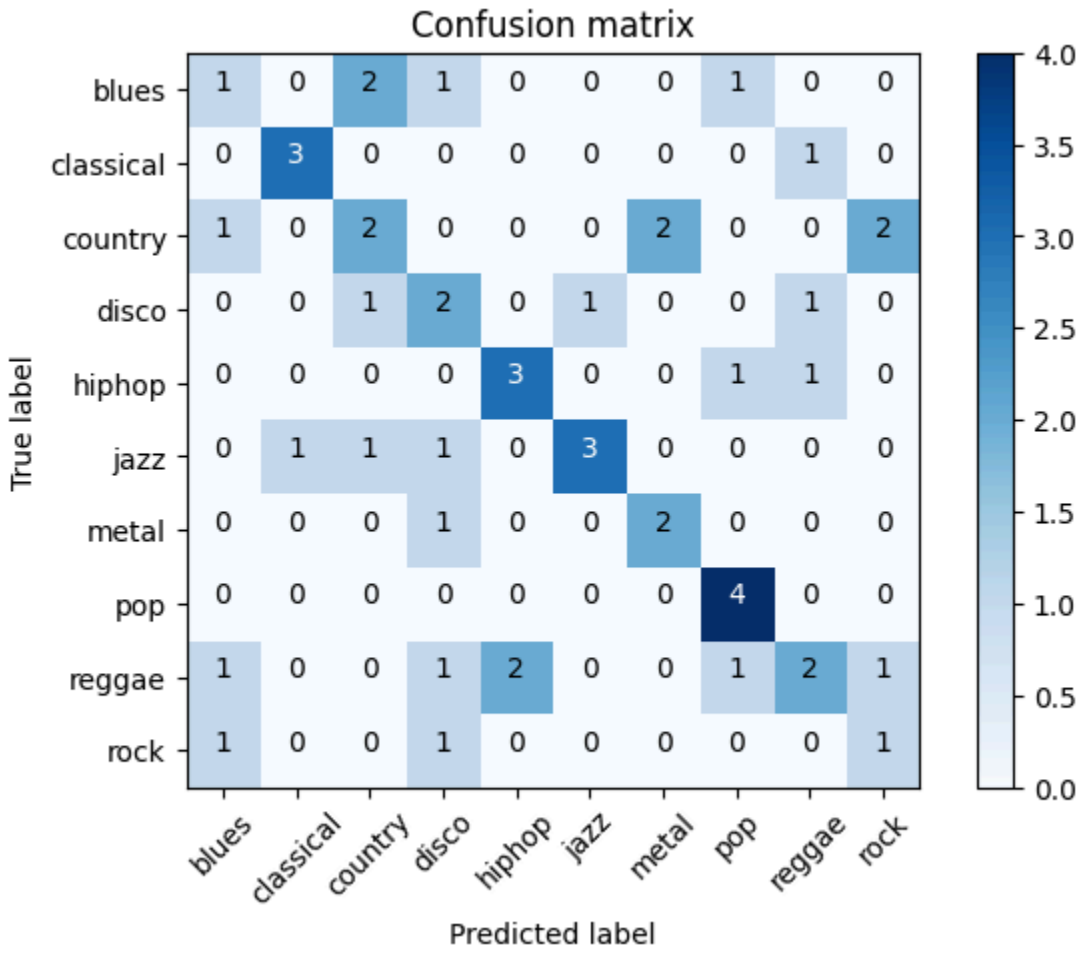
```
Test Set Accuracy =  0.74
Test Set F-score =  0.74
ROC AUC = 0.947
```

Time Domain Features Only:



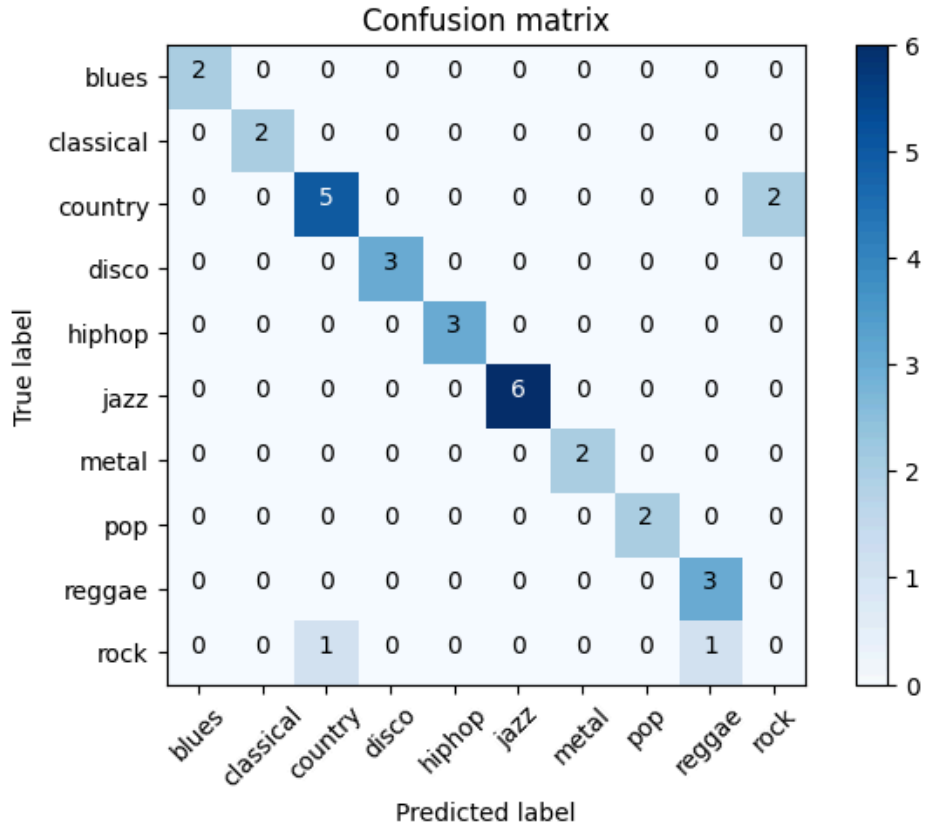

```
Test Set Accuracy =  0.46
Test Set F-score =  0.47
ROC AUC = 0.900
```

Frequency domain features only:

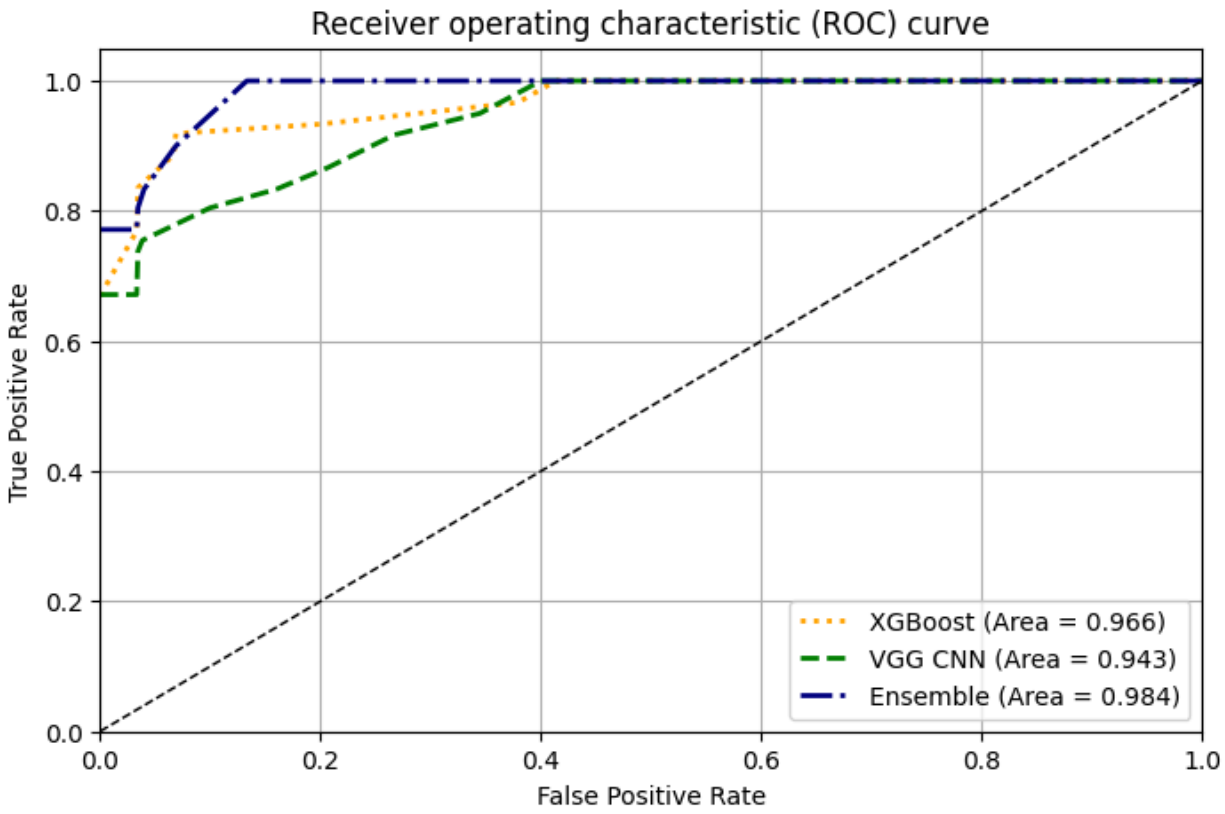



```
Test Set Accuracy =  0.78
Test Set F-score =  0.77
ROC AUC = 0.966
```

Ensemble of VGG 16 with Fine Tuning and Gradient Boosting:




```
Test Set Accuracy =  0.88
Test Set F-score =  0.86
ROC AUC = 0.980
```



Benchmark Metrics for GTZAN Dataset

1. Accuracy

- **Traditional Machine Learning Approaches:**
  - Using Mel-Frequency Cepstral Coefficients (MFCCs) + Support Vector Machines (SVMs):
    - Accuracy: ~65%-75%
  - Using Chroma Features + Random Forests:
    - Accuracy: ~70%-80%

- **Deep Learning Approaches:**
    - Convolutional Neural Networks (CNNs) applied to spectrograms:
        - Accuracy: ~85%-90%
    - Transfer learning with pre-trained models (e.g., VGGish, ResNet):
        - Accuracy: ~90%-93%

2. F1-Score

- For traditional ML methods: ~0.7
- For CNN-based methods: ~0.85-0.9 (depending on the architecture and preprocessing).

3. ROC-AUC

- ROC-AUC is less commonly reported for multi-class classification tasks (since it requires pairwise class comparisons or averaging).
- Approximate values:
    - Using CNN-based models: Macro-averaged ROC-AUC values of **0.90-0.94** have been reported for GTZAN genres.

Conclusion:

We can conclude that our ensemble model has fulfilled all benchmarks for the GTZAN dataset. In future we may try to work on other test datasets and measure our model's efficiency.

References:

1. **Hershey S., Chaudhuri S., Ellis D. P. W., Gemmeke J. F., Jansen A., Moore R. C., Plakal M., Platt D., Saurous R. A., Seybold B., Slaney M., Weiss R. J., & Wilson K.** (2017). CNN Architectures for Large-Scale Audio Classification. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 131–135. Link: https://arxiv.org/pdf/1609.09430

2. **Bahuleyan H.** (2018). *Music Genre Classification using Machine Learning Techniques*. arXiv preprint arXiv:1804.01149.

3. **Choi K., Fazekas G., Sandler M., & Cho K.** (2017). Convolutional Recurrent Neural Networks for Music Classification. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2392–2396. Link: https://arxiv.org/pdf/1609.04243

Our Code:

https://github.com/mrashid5919/Music-Genre-Classification-Using-Ensemble-Learning