Progect
Group Deadwing

Vashist Hegde
Harshit Lohan
Richard Samuelson
Luke Morris

May 2024

1 Introduction

Progressive rock, often abbreviated as prog rock, is a genre of rock music that emerged in the late 1960s and reached its peak popularity in the 1970s. It is characterized by its ambitious and experimental approach to music, often incorporating elements of classical, jazz, and other genres into its compositions. Prog rock is known for its complex musical structures, virtuosic performances, and elaborate concept albums.

Prog rock bands frequently explore unconventional time signatures, extended song lengths, and intricate instrumental arrangements. Lyrically, prog rock often delves into philosophical, political, or fantastical themes, often conveyed through intricate storytelling or allegory.

Key features of prog rock include:

1. **Instrumental proficiency**: Prog rock musicians are often highly skilled instrumentalists, known for their technical prowess and ability to perform complex musical passages.

2. **Conceptual albums**: Many prog rock albums are conceptual in nature, with overarching themes or narratives that tie together the individual songs into a cohesive whole.

3. **Experimentalism**: Prog rock artists are known for pushing the boundaries of conventional rock music, experimenting with new sounds, structures, and instrumentation.

4. **Long-form compositions**: Prog rock songs are often longer than typical rock songs, with some extending beyond the ten-minute mark or even spanning entire album sides.

5. **Eclecticism**: Prog rock draws inspiration from a wide range of musical styles and influences, including classical music, jazz, folk, and world music, resulting in a diverse and eclectic sound.

Given the diverse and intricate nature of prog rock music, classifying it from non-prog rock can be a challenging task. In this machine learning project, we aim to develop a classification model that can accurately distinguish between prog rock and non-prog rock music based on audio features, lyrics, and other relevant metadata. By leveraging machine learning algorithms, we seek to automate the process of genre classification and provide insights into the distinct characteristics of prog rock that differentiate it from other genres within the rock music spectrum.

2 Baseline Model

To establish a benchmark for evaluating the performance of our primary architectures, we utilized a Logistic Regression model as the baseline model. Logistic Regression is particularly suitable as a baseline in classification tasks due to its simplicity, interpretability, and efficiency in training, even with large datasets.

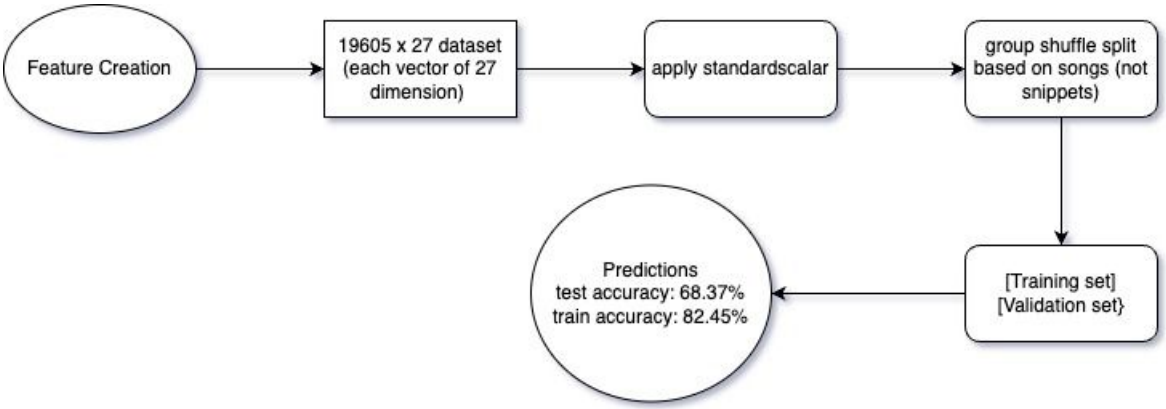
2.1 Rationale for Using Logistic Regression

Logistic Regression is a robust statistical method that predicts the probability of a binary outcome based on one or more predictor variables. The reasons for selecting Logistic Regression as a baseline for our song genre classification task include:

- **Simplicity and Speed:** Logistic Regression is straightforward to implement and requires relatively less computational resources compared to more complex models like deep neural networks. This makes it an ideal choice for quickly setting up a baseline to gauge initial model performances.
- **Probabilistic Interpretation:** It provides a probability score for observations being in a particular class, which is beneficial for tasks where you want to measure certainty in predictions rather than just classifications.
- **Good Performance in Binary Classification:** For a binary classification problem such as distinguishing between two genres of music, Logistic Regression can perform remarkably well, especially when the feature set is appropriately pre-processed and normalized.
- **Benchmarking:** It helps in setting a performance benchmark for other sophisticated models. If a complex model does not perform significantly better than the Logistic Regression model, it may indicate that either the feature set is insufficiently robust or the model complexity is unnecessary.

By comparing the performance of Logistic Regression with our primary models like Deep Neural Networks, CNNs, and LSTMs, we can assess the effectiveness of more complex architectures against a well-understood baseline. This comparative analysis helps in validating the added complexity of advanced models and ensures that they provide a significant improvement over simpler methods.

2.2 Architecture



2.3 Performance:

2.3.1 Performance on training set:

82.45%

2.3.2 Performance on test set:

68.35%

3 Feature Extraction and Model Performance Analysis

3.1 Initial Feature Extraction Using MFCCs

For the purpose of distinguishing between progressive rock (prog rock) and non-progressive rock songs, we cropped the songs into 30 second snippets and for each snippet, we extracted Mel-frequency cepstral coefficients (MFCCs) using the `librosa` library. MFCCs are a widely used feature in audio signal processing, especially valuable in music and speech analysis applications due to their ability to encapsulate the primary characteristics of audio signals in a compact form. Each audio frame generated 20 MFCCs, capturing various aspects of the audio's spectral envelope.

The lower-order coefficients (e.g., 1st to 5th) typically represent the general spectral shape, capturing the broad audio textures that are influential in identifying different instruments and vocal tones. Higher-order coefficients provide finer spectral details which, though less robust in noisy conditions, are essential for capturing the nuanced features of complex music genres like prog rock, known for its elaborate compositions and frequent shifts in tempo, key, and instrumentation.

3.1.1 Model Training and Performance on Initial Feature Extraction

Three different neural network architectures were evaluated for their effectiveness in classifying the genres:

- **Deep Neural Network (DNN):** A fully connected deep neural network was trained, yielding an accuracy of 67.5% on the validation set of snippets. This model likely benefited from its ability to form complex hierarchical patterns from the static input features of MFCCs.

  Architecture:

Model: "sequential_6"

Layer (type)	Output Shape	Param #
flatten_4 (Flatten)	(None, 25840)	0
dense_16 (Dense)	(None, 512)	13,230,592
dropout_8 (Dropout)	(None, 512)	0
dense_17 (Dense)	(None, 128)	65,664
dropout_9 (Dropout)	(None, 128)	0
dense_18 (Dense)	(None, 64)	8,256
dropout_10 (Dropout)	(None, 64)	0
dense_19 (Dense)	(None, 32)	2,080
dropout_11 (Dropout)	(None, 32)	0
dense_20 (Dense)	(None, 16)	528
dropout_12 (Dropout)	(None, 16)	0
dense_21 (Dense)	(None, 8)	136
dropout_13 (Dropout)	(None, 8)	0
dense_22 (Dense)	(None, 1)	9

Total params: 13,307,265 (50.76 MB)

Trainable params: 13,307,265 (50.76 MB)

Accuracy: 67.5%

- **Convolutional Neural Network (CNN):** The CNN, typically strong in capturing spatial patterns in data, achieved an accuracy of 65.5% on the validation set of snippets. Despite CNNs being more common in image processing, their application in processing time-series data like audio can be beneficial, particularly in recognizing patterns across time frames.

Architecture:

Layer Name	Layer Type	Output Shape	Parameters
conv2d_1	Conv2D	(None, height-2, width-2, 32)	896
max_pooling2d_1	MaxPooling2D	(None, ceil((height-2)/2), ceil((width-2)/2), 32)	0
batch_normalization_1	BatchNormalization	(None, ceil((height-2)/2), ceil((width-2)/2), 32)	128
conv2d_2	Conv2D	(None, ceil((height-4)/4), ceil((width-4)/4), 32)	9,248
max_pooling2d_2	MaxPooling2D	(None, ceil((height-4)/4)/2, ceil((width-4)/4)/2, 32)	0
batch_normalization_2	BatchNormalization	(None, ceil((height-4)/4)/2, ceil((width-4)/4)/2, 32)	128
conv2d_3	Conv2D	(None, ceil((height-8)/8), ceil((width-8)/8), 32)	4,128
max_pooling2d_3	MaxPooling2D	(None, ceil((height-8)/8)/2, ceil((width-8)/8)/2, 32)	0
batch_normalization_3	BatchNormalization	(None, ceil((height-8)/8)/2, ceil((width-8)/8)/2, 32)	128
flatten	Flatten	(Variable, based on input size)	0
dense	Dense	(Variable, based on flatten output shape) 64	Variable
dropout	Dropout	(Same as above dense layer)	0
dense_1	Dense	(None, 2)	130

- **Total Parameters**: 14,658 (approximate)
- **Trainable Parameters**: 14,402
- **Non-trainable Parameters**: 256

Accuracy: 65.5%

- **Long Short-Term Memory Network (LSTM):** The LSTM also recorded an accuracy of 64.5% on the validation set of snippets. As LSTMs are designed to handle sequences of data, they are inherently suitable for audio analysis. The performance here suggests that while LSTMs are good at capturing temporal dependencies, the complex nature of prog rock might require more nuanced temporal features or additional context that goes beyond the scope of standard MFCCs.

Architecture:

Model: "sequential_2"

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 1292, 64)	21,760
lstm_1 (LSTM)	(None, 64)	33,024
dense_6 (Dense)	(None, 64)	4,160
dropout_4 (Dropout)	(None, 64)	0
dense_7 (Dense)	(None, 2)	130

Total params: 59,074 (230.76 KB)

Trainable params: 59,074 (230.76 KB)

Accuracy and Error Plots:

50/50 ━━━━━━━━━━━━━━━━━━━━ **7s** 133ms/step - accuracy: 0.6469 - loss: 1.0299
Accuracy on test set is: 0.6453232765197754

Accuracy: 64.5%

3.1.2 Data Size Considerations

The extracted features from 30-second snippets of each song in our dataset resulted in a significantly large dataset. Each 30-second audio snippet, when converted into a series of overlapping frames (typically 20-40 ms per frame), produced a large number of data points per song snippet. Each frame generated 20 MFCCs, leading to extensive data matrices that required substantial computational resources for training and evaluation. This large data volume posed challenges in terms of storage, processing speed (typically taking about 2 hours for every attempt of modifying and retraining our model), and model training efficiency. The Data creation took 2 hours and model evaluation each time took about 2 hours.
Due to the high data volume and low accuracy, we decided to improve our feature extraction with a goal of reduced dimentionality and modified our architectures to maximize accuracy.

3.2 Improving Feature Extraction

Due to data size considerations and low accuracy, we got onto restricting the feature points to more relevant and less bulky in size. One obvious choice was to take the 20 MFCC points generated for each feature and taking their mean over the 10 seconds, effectively giving us a relevant metric but technically summarized, for each peice of the song. We also decided to include other features.

3.2.1 Additional Audio Features Used

In addition to Mel-frequency cepstral coefficients (MFCCs), our analysis incorporated several other audio features extracted using the `librosa` library to enhance the classification of prog rock versus non-prog rock songs. These features provide a comprehensive representation of the audio characteristics, aiding in the accurate classification through diverse aspects of the audio signal.

3.2.2 Spectral Features

- **Spectral Centroid:** This feature represents the center of mass of the spectrum and is a measure of the brightness of a sound. It is calculated as the weighted mean of the frequencies present in the sound, with their magnitudes as the weights. A higher spectral centroid indicates a brighter sound with more high frequencies.
- **Spectral Bandwidth:** This measures the width of the band of light at half the peak maximum and is used to describe the spread of the spectrum around the spectral centroid. It quantifies the spectral width of the sound. A larger bandwidth indicates a more noisy and complex sound.
- **Spectral Rolloff:** This is a measure of the shape of the sound spectrum, which indicates the frequency below which a specified percentage of the total spectral energy, typically 85%, lies. This feature is used to distinguish between harmonic and noisy sounds.

3.2.3 Time-Domain Features

- **Zero Crossing Rate:** This feature measures the rate at which the signal changes sign or crosses zero. It is a simple measure of the noisiness and the frequency content of a sound. Higher zero crossing rates can indicate percussive and noisy sounds.

- **Root Mean Square Energy (RMSE):** This feature measures the average power of the audio signal. It is useful for detecting the loudness of the audio, providing a basis for normalizing sounds before further processing.

3.2.4 Harmonic Features

- **Chroma STFT:** Short for Chroma Short-Time Fourier Transform, this feature represents the twelve different pitch classes. Chroma features are typically used in music information retrieval for analyzing the harmony of the music, identifying chords and matching patterns in audio tracks.

These additional features were leveraged to capture a more detailed and diverse representation of the audio signals in our dataset. By combining these with MFCCs, we aimed to improve the accuracy and robustness of our genre classification models, providing a richer set of inputs for distinguishing complex musical structures in prog rock.

3.2.5 Further Improvements - (important feature extraction)

We further incorporated the use of Random Forests to rank the features on the basis of their significance.

The procedure described involves using a Random Forest Classifier to analyze the importance of different features in a dataset for predicting song genres. Here's a generalized summary of the steps:

1. **Model Initialization**: A Random Forest Classifier model is initialized with a specified number of decision trees and a fixed random seed to ensure reproducibility.
2. **Model Training**: The model is trained using a set of input features and corresponding target values, which helps the model learn how to classify the data effectively.
3. **Feature Importance Extraction**: After training, the model assesses the significance of each input feature in making accurate predictions. This step involves calculating a score for each feature that indicates its relative importance.
4. **Storing Feature Names**: The names of the input features are stored to provide clear labeling and reference for the importance scores.
5. **Creating a Summary DataFrame**: A DataFrame is created to systematically display each feature alongside its importance score. This table organizes the features and their scores in a structured format.
6. **Sorting by Importance**: The DataFrame is sorted in descending order based on the importance scores. This ordering highlights which features have the most significant impact on the model's predictions.
7. **Displaying Results**: The final sorted DataFrame is displayed, providing a clear and prioritized list of features based on their importance in the classification task.

Feature importance in a Random Forest model is a useful way to see which features are contributing most to the prediction. Random Forest, an ensemble learning method, works by building multiple decision trees during training and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Each tree in the forest is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set.

How Random Forest Estimates Feature Importance Random Forest estimates the importance of a feature by looking at how much the tree nodes that use that feature reduce impurity on average across all trees in the forest. The impurity decrease from each feature is averaged and normalized to sum to one. More technically, it measures the decrease in node impurity (Gini impurity for classification, variance reduction for regression) weighted by the probability of reaching that node (which

is approximated by the proportion of samples reaching that node) for all trees in the forest.

3.2.6 Result

	Feature	Importance
5	zero_crossing_rate	0.457632
1	rmse	0.033397
8	mfcc3	0.032213
22	mfcc17	0.026358
11	mfcc6	0.023804
2	spectral_centroid	0.023307
9	mfcc4	0.023232
12	mfcc7	0.023121
6	mfcc1	0.023044
10	mfcc5	0.022981
15	mfcc10	0.022743
3	spectral_bandwidth	0.022293
13	mfcc8	0.022132
4	rolloff	0.021070
7	mfcc2	0.020699
18	mfcc13	0.020142
16	mfcc11	0.019393
20	mfcc15	0.019259
21	mfcc16	0.019153
23	mfcc18	0.019091
0	chroma_stft	0.018419
14	mfcc9	0.018191
25	mfcc20	0.017204
17	mfcc12	0.017136
24	mfcc19	0.017018
19	mfcc14	0.016966

3.3 Model training and performance analysis on Deep Neural Networks (DNN) with improved data extraction

We re-attempted to use the DNN with modified architecture to suit the new set of features.

3.3.1 Rationale behind using DNN

Deep Neural Networks are particularly suited for music genre classification due to their deep architecture, which allows for learning at multiple levels of abstraction. The layers in a DNN can learn to recognize various features of the music from basic to complex, making it possible to capture the intricate characteristics of different genres effectively.

3.3.2 Advantages of Using DNN

- **Feature Learning:** Unlike traditional machine learning models that require manual feature engineering, DNNs have the inherent ability to learn features automatically. This capability is crucial in music where the relevance of features can vary significantly across different genres.
- **Handling Non-linear Relationships:** Music data often contains complex, non-linear relationships that DNNs are particularly good at modeling. This allows DNNs to capture the subtle nuances that distinguish between genres like prog rock and its counterparts.
- **Scalability:** DNNs are highly scalable with respect to the amount of data and the complexity of the model. This is advantageous in genre classification as more layers or neurons can be added to improve the model's learning capacity without the need for re-engineering the solution.
- **Robustness to Overfitting:** With appropriate regularization techniques such as dropout, batch normalization, and data augmentation, DNNs can be made robust to overfitting, thus enhancing their generalization over unseen music tracks.
- **Improved Accuracy:** The deep architecture of DNNs can lead to better performance and higher accuracy in classification tasks compared to simpler models, provided that they are trained with a sufficiently large and representative dataset.

In summary, the use of a Deep Neural Network in our project leverages its strong feature-learning and non-linear modeling capabilities, offering a sophisticated approach to distinguishing complex patterns inherent in different music genres. This methodological choice aligns with the advanced requirements of genre classification, aiming to achieve high accuracy and reliable predictions in our classification system.

3.3.3 Architecture

Layer (type)	Output Shape	Param #
dense_45 (Dense)	(None, 256)	6,912
dropout_37 (Dropout)	(None, 256)	0
dense_46 (Dense)	(None, 128)	32,896
dropout_38 (Dropout)	(None, 128)	0
dense_47 (Dense)	(None, 1)	129

Total params: 119,813 (468.02 KB)

Trainable params: 39,937 (156.00 KB)

3.3.4 Performance

Performance on training set: 82% **Performance on test set** 67.72%

4 Final Model: Long Short-Term Memory Network (LSTM)

4.1 Rationale behind using LSTM Network

LSTMs are a type of recurrent neural network (RNN) suited for sequence prediction tasks because they can maintain information in 'memory' for long periods of time. In the context of music, this means LSTMs are capable of learning from sequences of audio features extracted over time, allowing them to capture not just individual notes or sounds but also the overall temporal dynamics that define a genre.

4.2 Advantages of Using LSTM

- **Temporal Sequence Learning:** LSTMs excel at learning from sequences, making them ideal for music where timing, rhythm, and progression are key elements. They can recognize patterns over time, which is essential for capturing the structure and progression in complex music genres like prog rock.
- **Handling Long-term Dependencies:** Music tracks often contain long-term dependencies, where future sounds depend on a sequence of previous sounds over an extended period. LSTMs are specifically designed to handle these long-term dependencies, unlike traditional RNNs that struggle with vanishing and exploding gradients.
- **Robustness to Sequence Length Variability:** LSTMs handle variability in sequence lengths effectively, which is beneficial when dealing with songs of varying durations and structures. This adaptability ensures consistent performance across different tracks and recording lengths.
- **Improved Generalization:** By learning and remembering sequences, LSTMs generalize better to new music tracks that share similar temporal patterns and structures, enhancing their predictive accuracy in genre classification.
- **Versatility:** LSTMs are not limited to audio features alone; they can be integrated with other types of data (like lyrics or metadata), providing a holistic approach to music genre classification.

By employing an LSTM network, our project takes advantage of its sophisticated temporal processing capabilities, which are critical for identifying and classifying the nuanced and complex patterns characteristic of different music genres. This approach ensures that both the immediate and historical contexts of musical elements are considered, leading to more accurate and meaningful genre classification.

4.3 Architecture

Layer (type)	Output Shape	Param #
lstm_147 (LSTM)	(None, 1, 128)	76,288
lstm_148 (LSTM)	(None, 64)	49,408
dense_129 (Dense)	(None, 16)	1,040
dense_130 (Dense)	(None, 8)	136
dense_131 (Dense)	(None, 2)	18

Total params: 126,890 (495.66 KB)

Trainable params: 126,890 (495.66 KB)

4.4 Layers Description

4.4.1 LSTM Layers

- **lstm_147**: This first LSTM layer contains 128 units. LSTM layers are specialized for sequence processing, allowing the model to maintain information across longer sequences. The output shape *(None, 1, 128)* indicates that the layer outputs a sequence of vectors, each with 128 features.
- **lstm_148**: Following the first LSTM layer, this second LSTM layer with 64 units processes the sequence further. The output shape *(None, 64)* suggests a configuration where only the final sequence output is returned, useful for transitioning to dense layers that expect fixed-length inputs.

4.4.2 Dense Layers

- **dense_129**: A fully connected layer with 16 units that processes features derived from the preceding LSTM layer, aiming to refine and transform the learned representations.
- **dense_130**: Another fully connected layer, this one with 8 units, serves to continue the pattern recognition process from the previous layer.
- **dense_131**: The final output layer with 2 units, likely configured for binary classification, providing the network's predictions.

Overall, the network comprises a total of 126,890 trainable parameters, indicating a substantial model capable of learning detailed and nuanced representations of the input data. This structure is ideally suited for tasks requiring an understanding of complex and time-dependent patterns within the data.

4.5 Performance Analysis on Training Data

4.5.1 Accuracy on snippets:

79.2%

4.5.2 Accuracy on songs:

80.8%

4.5.3 Training set missclassifications

filename	true	predicted
prog08_-_I_Spy.mp3	0	1
nonprog07_-_Signals_(Orchestrion_Sketch).mp3	1	0
nonprog11_Jeru_The_Damaja_-_Come_Clean.mp3	1	0
nonprog12-modest_mouse-spitting_venom.mp3	1	0
nonprog03_-_What_the_Water_Gave_Me.mp3	1	0
nonprogBeethoven_Symphony_5.mp3	1	0
prog1_robert_wyatt_SEA_SONG.mp3	0	1
prog05_-_Stargazers.mp3	0	1
nonprog03_Katy_Song.mp3	1	0
prog101-emerson_lake_and_palmer-jerusalem.mp3	0	1
nonprog02_QUEENS_OF_THE_STONE_AGE-NO_ONE_KNOWS...	1	0
nonprog07_-_Speed_Of_Sound.mp3	1	0
nonprog02_-_Sigur_Rós_-_Svefn-G-Englar.mp3	1	0
prog01_-_Nothing_At_Best.mp3	0	1
prog02_Freewill.mp3	0	1

We randomly picked the song: Signals Orchestrion Sketch by Pat Metheny. As a social experiment, we made some of our associates who are familiar with prog rock music listen to this song and asked them which genre they think it belongs to. Most of them incorrectly classified it as a prog rock song. We concluded that this was the case because of the song's unconventional chord progreession.

4.6 Performance Analysis on Test Data

We discovered that dropout layers among the dense layers were imepeding the perormance of the model.

Explanation of dropout layers:

Dropout works by randomly setting a fraction of input units to 0 at each update during training time, which means that their contribution to the forward pass and any weight updates is temporally removed on the forward pass and any backward pass by applying a dropout mask. This is as if they are "dropped out" of the network. This random dropping is meant to prevent units from co-adapting too much. We believe that since we had a sufficiently large dataset for training, model may not require regularization methods like dropout because the extensive data provides enough variance and examples to learn robust features without overfitting. Dropout is more beneficial in scenarios where the data is limited or highly imbalanced.

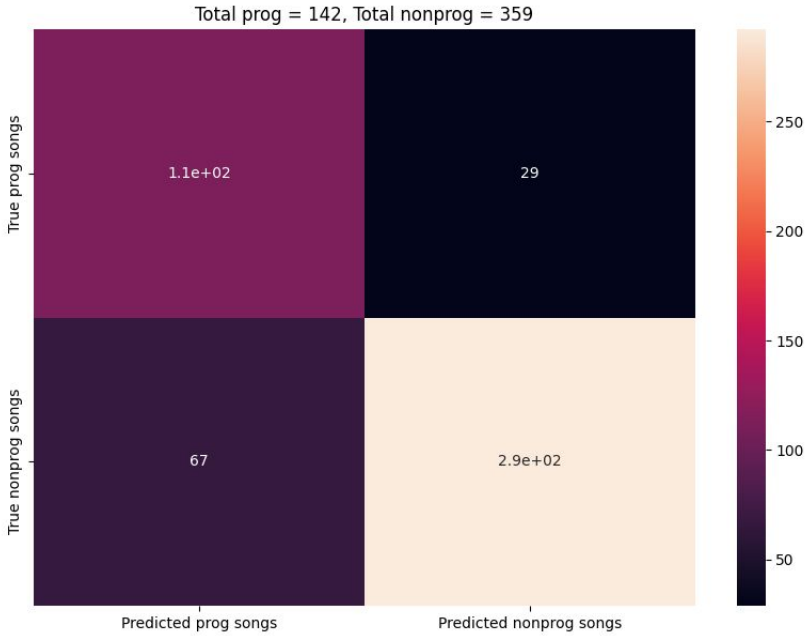
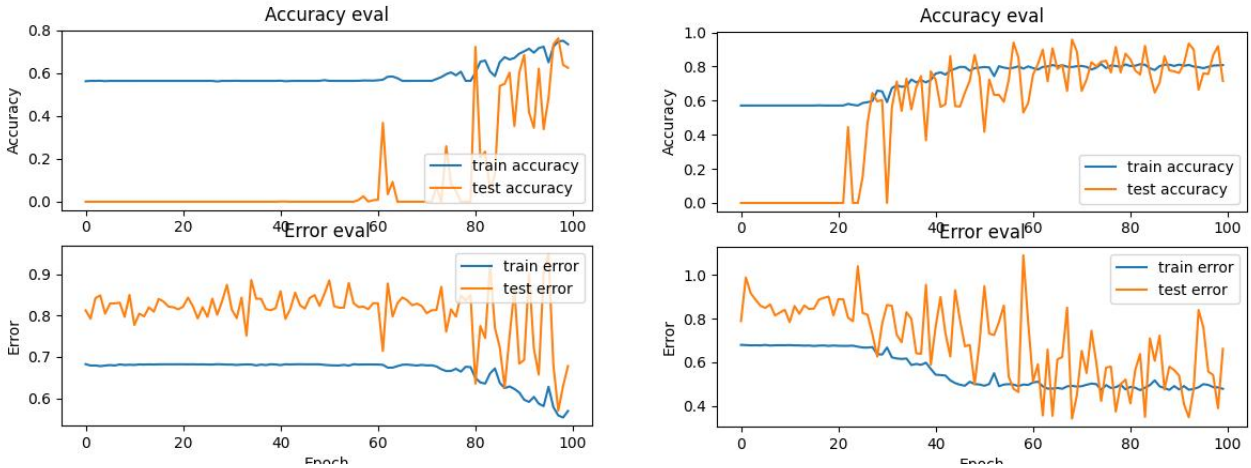


Figure 1: Confusion Matrix for Training set

4.6.1 Accuracy and Loss Evaluation over 2 K-fold split batches of test data



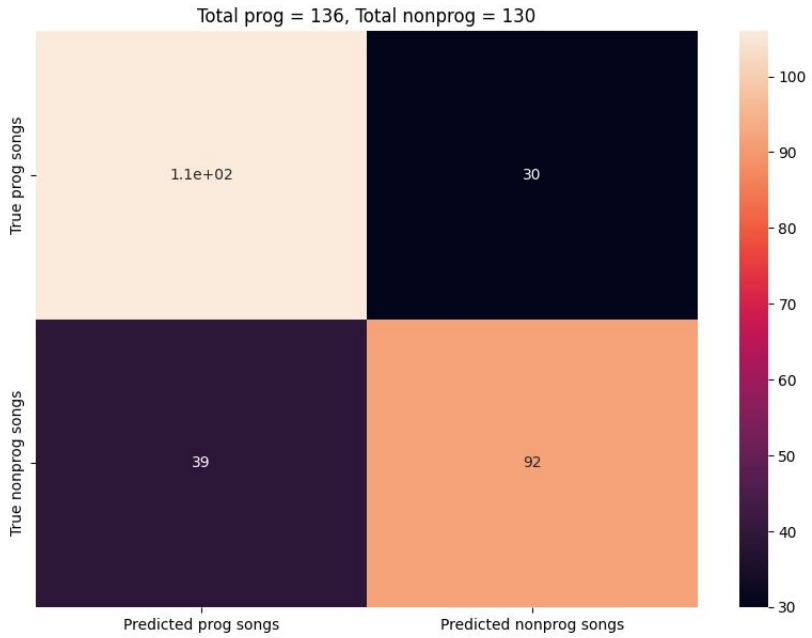


Figure 2: Confusion Matrix for Test set

4.6.2 Test set Miss-classifications

filename	true	predicted
nonprog1993_Snap_Rhythm_Is_A_Dancer.mp3	1	0
nonprog01_-_Under_the_Pressure.mp3	1	0
prog04_Operation-_Mindcrime.mp3	0	1
prog-04-_Knots.mp3	0	1
nonprogAli_Farka_Toure_with_Ry_Cooder_-_Talkin...	1	0
nonprog10-hozier-be-65228571.mp3	1	0
progCaravan_The_Love_In_Your_Eye.mp3	0	1
nonprog04-deacon_blues_320_lame_cbr.mp3	1	0
nonprog06_The_Temple_of_the_King.mp3	1	0
nonprog01_-_Mustt_Mustt_(Lost_in_His_Work).mp3	1	0
prog01._Vert.mp3	0	1
prog01_-_When_the_Heart_Rules_the_Mind.mp3	0	1
prog01_-_Songs_From_The_Wood_(2003_Digital_Rem...	0	1
nonprog16_-_Navras.mp3	1	0
prog07_-_Zartlicher_Abschied.mp3	0	1

Out of this song, we randomly selected the song "Be" by Hozier and analised it. It makes sense that this pop song is being classified as a prog rock song because of the distorted guitar giving a rock touch and change in time signatures.

4.7 True and Predicted Labels on Songs in Training set

filename	true	predicted
prog01._The_Same_River.mp3	0	0
nonprog07_Guy's_Eyes.mp3	1	1
nonprogCranberries_-_04_-_Zombie.mp3	1	1
nonprog04-The_Rip.mp3	1	1
progCaravan_The_Love_In_Your_Eye.mp3	0	1
prog06-In_a_glass_house.mp3	0	0
nonprog23-The-Weeknd---I-Feel-It-Coming-(Feat-...	1	1
prog4_Foxlight.mp3	0	0
prog01.A_passion_play_1.mp3	0	0
nonprog1994_Ace_Of_Base_All_That_She_Wants.mp3	1	1
nonprog2005_Kelly_Clarkson_Since_U_Been_Gone.mp3	1	1
prog03_Time.mp3	0	1
nonprog06_Californication.mp3	1	0
prog01-hymn_of_the_seventh_galaxy_320_lame_cbr...	0	1
progSteve_Hackett_The_Return_of_the_Giant_Hogw...	0	0

4.8 True and Predicted Labels on Songs in Test set

filename	true	predicted
prog07_-_Chimera's_Wreck.mp3	0	0
prog04_Clairvoyant.mp3	0	0
nonprog07_By_Myself.m4a	1	1
prog05_-_Stargazers.mp3	0	1
nonprog04-Fade_Away_(And_Radiate).mp3	1	1
nonprogJohn_Williams-The_Phantom_Menace-0_-_Du...	1	1
prog09_-_Quantum_Factor.mp3	0	0
prog04_-_Running_Hard.mp3	0	0
nonprogBeethoven_Symphony_5.mp3	1	0
nonprog11_variations_on_a_cocktail_dress.mp3	1	1
prog09_-_The_World_We_Used_to_Know.mp3	0	0
nonprog03-Sweet_Blindness.mp3	1	1
nonprogOingo-Boingo---Weird-Science-(Extended-...	1	1
prog03_Sartori_In_Tangier.mp3	0	1
prog01_Prologue_-_Autumn.flac	0	0

NOTE: The complete .csv file for the the prediction labels are attached in the final submission folder.

4.9 Performance on Post-Prog Dataset

```
prog = 0 || nonprog = 1
b1-master_boot_record-dma_4_cascade.mp3 ---------------- predicted 0
Cloudkicker_-_Let_yourself_be_huge.mp3 ---------------- predicted 0
HEART_OF_A_COWARD_-_Hollow.mp3 ---------------- predicted 1
06-1289,_Voyeur_Will_Shine,_Fight_For_Distinction,_Evolution_Is_Mine..mp3 ---------------- predicted 0
PERIPHERY_-_Zyglrox.mp3 ---------------- predicted 1
03_-_Language_II_Conspire.mp3 ---------------- predicted 1
07_The_Race_Is_About_To_Begin.mp3 ---------------- predicted 0
Textures_-_Laments_Of_An_Icarus.mp3 ---------------- predicted 1
SikTh_-_Hold_My_Finger.mp3 ---------------- predicted 0
05._Physical_Education.mp3 ---------------- predicted 1
The_Algorithm_-_Isometry.mp3 ---------------- predicted 1
Meshuggah-_Soul_Burn.mp3 ---------------- predicted 1
Veil_Of_Maya_-_Punisher.mp3 ---------------- predicted 1
BORN_OF_OSIRIS_-_Divergency.mp3 ---------------- predicted 1
01-darko_us-splinter_cell.mp3 ---------------- predicted 1
08-the_haarp_machine-machine_over.mp3 ---------------- predicted 0
MONUMENTS_-_I,_The_Creator.mp3 ---------------- predicted 1
06_A_Light_Will_Shine.mp3 ---------------- predicted 1
01_Arithmophobia.mp3 ---------------- predicted 0
Hacktivist_-_DECEIVE_AND_DEFY.mp3 ---------------- predicted 1
AFTER_THE_BURIAL_-_A_Wolf_Amongst_Ravens.mp3 ---------------- predicted 1
CHIMP_SPANNER_-_Bad_Code.mp3 ---------------- predicted 0
```