Georgia State University

MUSIC GENRE CLASSIFICATION

Presented By :

Rujula Malineni

Department of Computer Science

AGENDA

- INTRODUCTION
- PROBLEM STATEMENT
- OBJECTIVE
- SYSTEM DESIGN AND ANALYSIS
- METHODOLOGY
- RESULTS
- CONCLUSION

INTRODUCTION

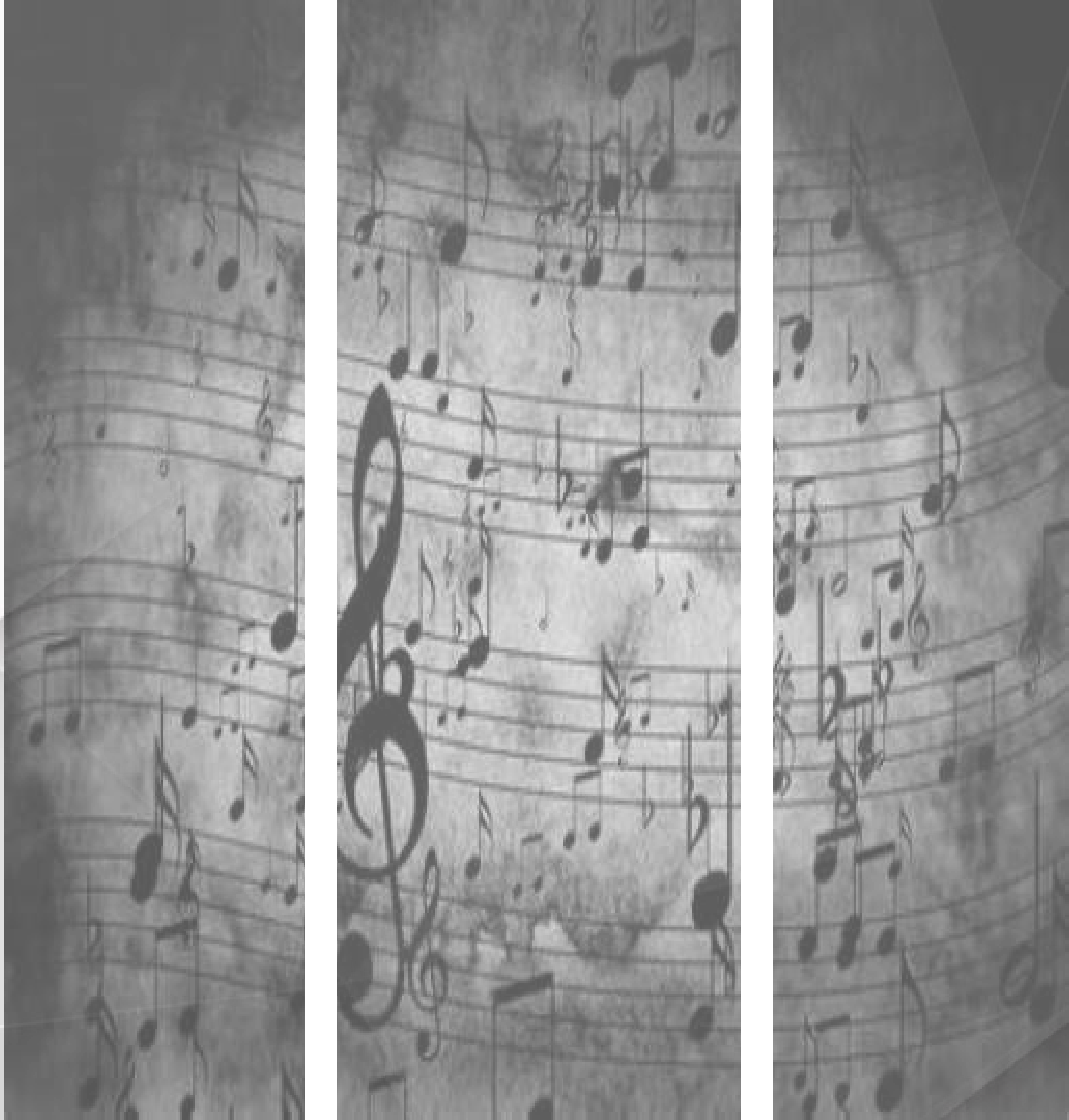
- A classification model is a machine learning algorithm that is used to categorize data into different classes or categories based on specific features or characteristics.

- There are various approaches to building a music genre classification model, including rule-based systems, statistical methods, and machine learning techniques.

- Deep learning models, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have shown promising results in music genre classification due to their ability to automatically extract relevant features from audio signals.

PROBLEM STATEMENT

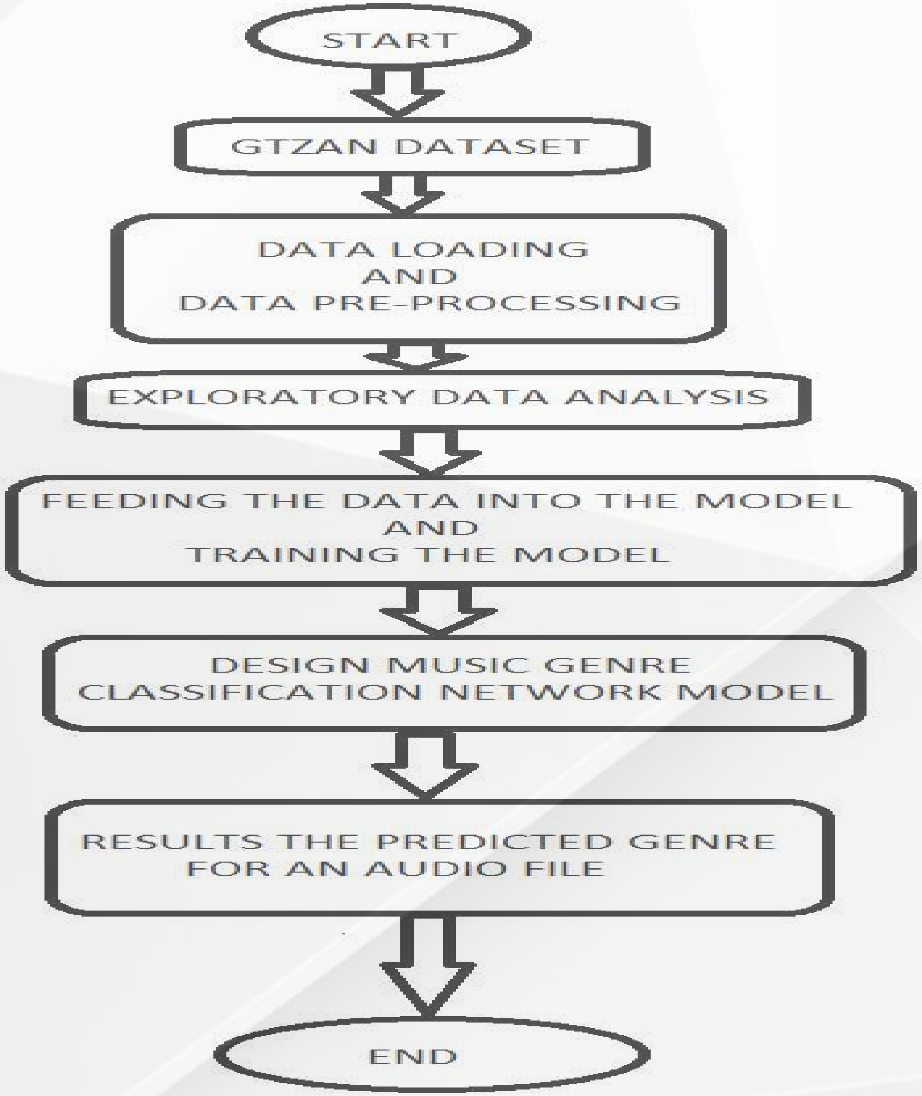
The project's purpose is to classify the music and predict the correct genre to the user. Providing relevant genre classification to users of music from a collection of specific set of data according to the genres.



OBJECTIVES

- To accurately classify music samples into their respective genres.
- To improve the efficiency and speed of music genre classification.
- To identify new and emerging music genres.
- To enable music streaming services to offer personalized music recommendations to their users based on their preferred genres.
- To aid music production companies in identifying which genres are most popular and profitable.
- To develop an automated music tagging system for efficient music management.
- To analyze music data to uncover patterns and trends in music consumption and preferences.
- To create a comprehensive database of music genres for research and analysis purposes.
- To improve the accuracy of music genre classification for music copyright infringement detection.
- To provide a better understanding of the relationship between music genres and demographics.

SYSTEM DESIGN AND ANALYSIS



- The first step in achieving this goal is to collect a dataset from GTZAN.
- The MFCC feature extraction technique is then applied.
- Preprocessing over the data.
- The CNN model is then built.
- To evaluate the model's performance, accuracy is used.
- Finally, the model is deployed for real-world use by integrating it with a web or mobile application, allowing users to input music samples and receive predictions on their genre.

The project's main goal is to develop a system that can accurately classify music into different genres using advanced deep learning techniques.

METHODOLOGY

Gathering of Data

Python has some great libraries for audio processing like Librosa and PyAudio.There are also builtin modules for some basic audio functionality.

We will mainly use two libraries for audio acquisition and playback:

- **Librosa**
- **IPython.display.Audio**

GTZAN genreclassification data set is the most recommended data set for the music genre classification project and it was collected for this task only.

It includes audio tracks from the 10 genres listed below:

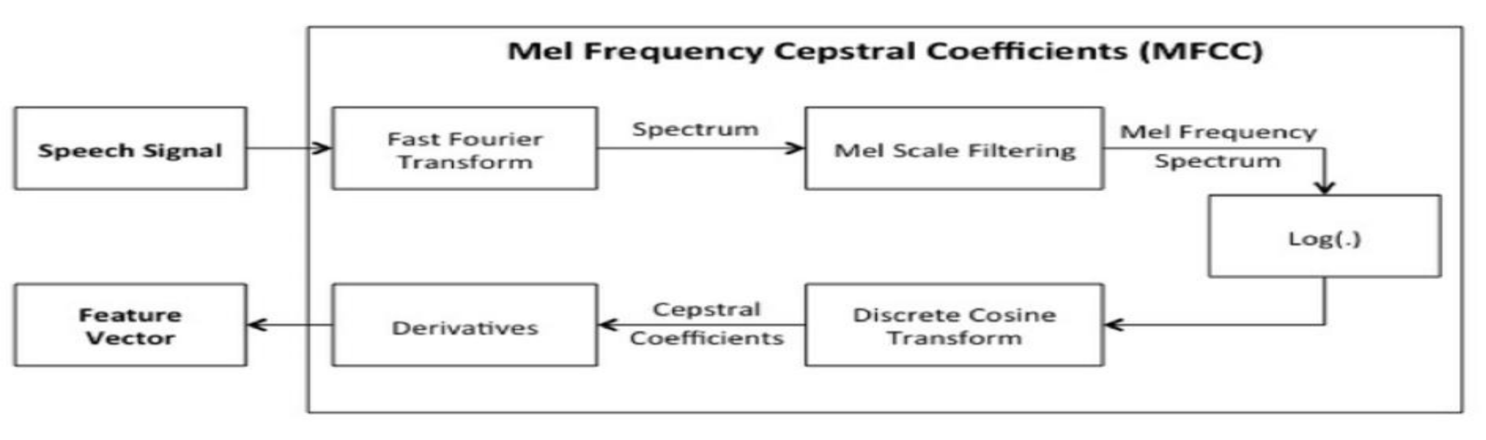
Blues	Jazz
Classical	Metal
Country	Pop
Disco	Reggae
Hip Hop	Rock

METHODOLOGY

Extracting Features with MFCC

Feature extraction is a common technique used in audio signal processing and analysis to extract useful information from audio data. The technique for audio feature extraction used here is :

Mel-frequency cepstral coefficients (MFCC) : In Jupyter notebook using Python, MFCC and spectrograms can be extracted using libraries such as librosa, which is a popular Python package for audio signal processing.

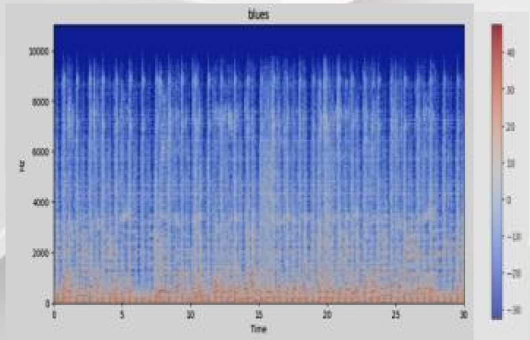


Flowchart of implementation of MFCC

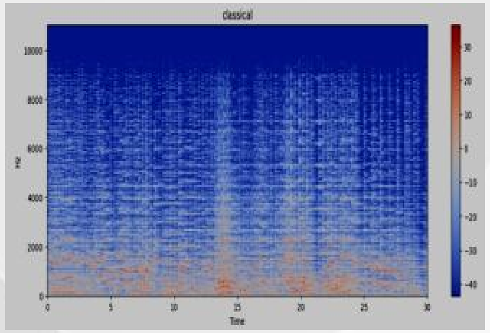
METHODOLOGY

Analysis on Data

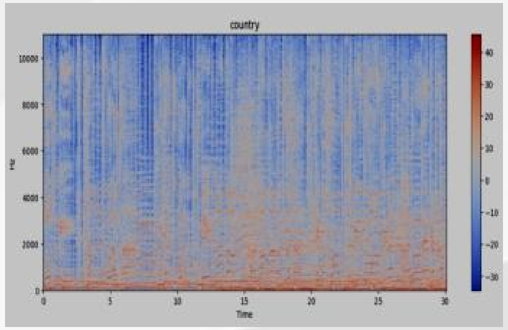
- To perform this analysis, we can use Python libraries such as librosa and matplotlib to generate and display spectrograms for each audio file in the dataset. By examining these spectrograms, we can identify common patterns in the frequency distribution of different music genres.
- For example, we might find that certain genres tend to have higher or lower frequency components, or that certain instruments are more prevalent in certain genres.



Blues



Classical

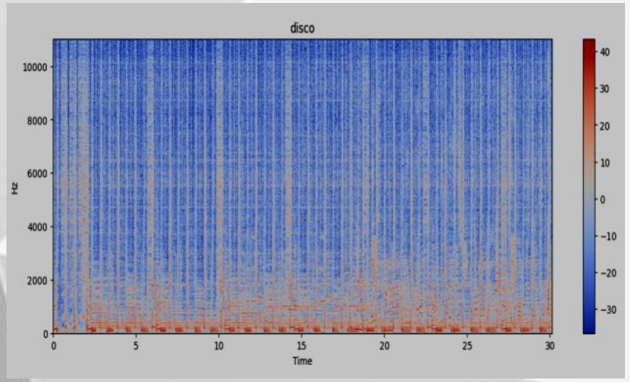


Country

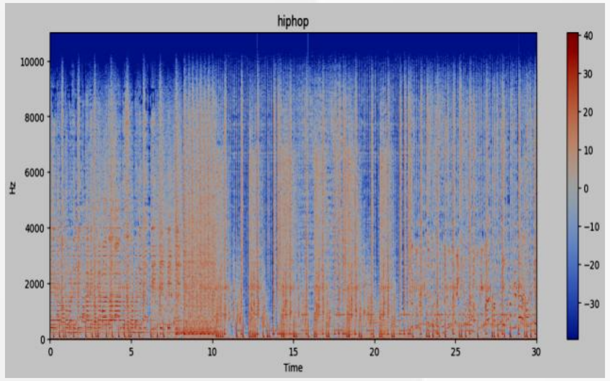
Spectrogram visual frequency representation for each genre.

MUSIC GENRE CLASSIFICATION

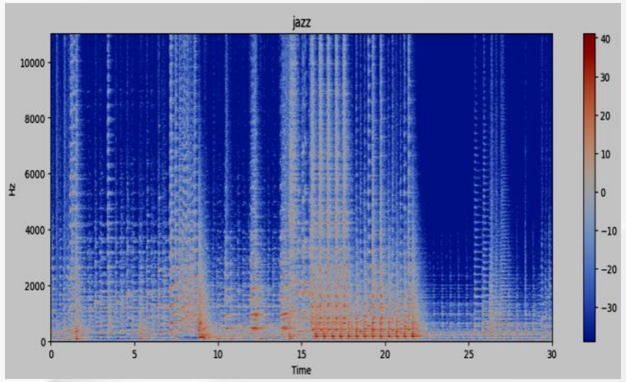
METHODOLOGY



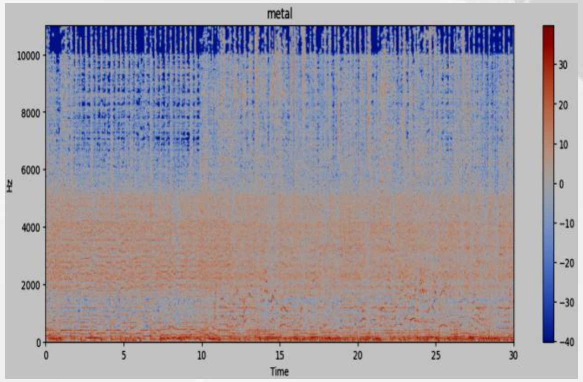
Disco



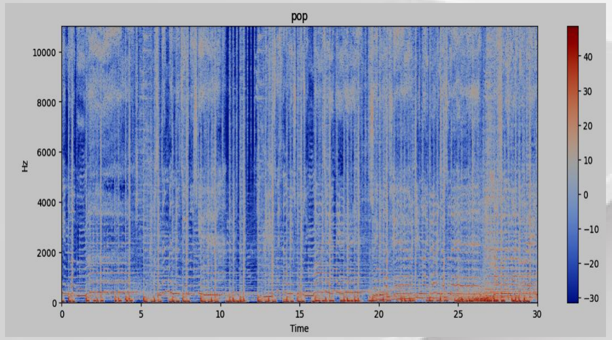
Hip Hop



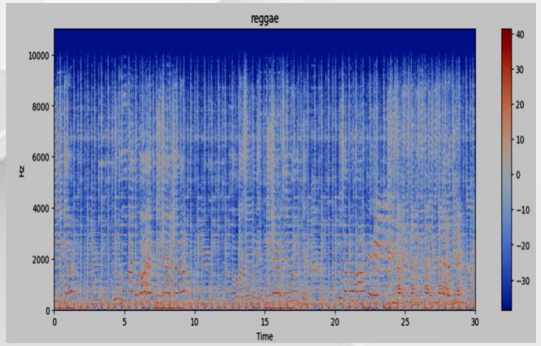
Jazz



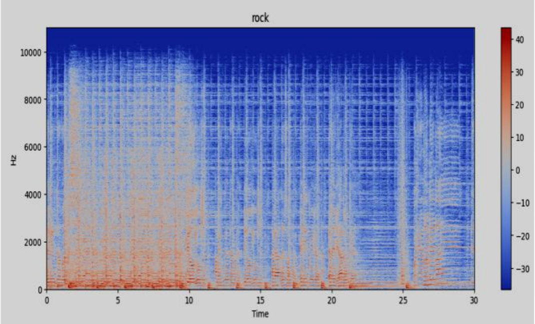
Metal



Pop



Reggae



Rock

The resulting images are a 2D representation of the time-varying frequency content of theaudio signal, with time on the x-axis, frequency on the y-axis, and color or intensity representing the magnitude or amplitude of the frequency content.

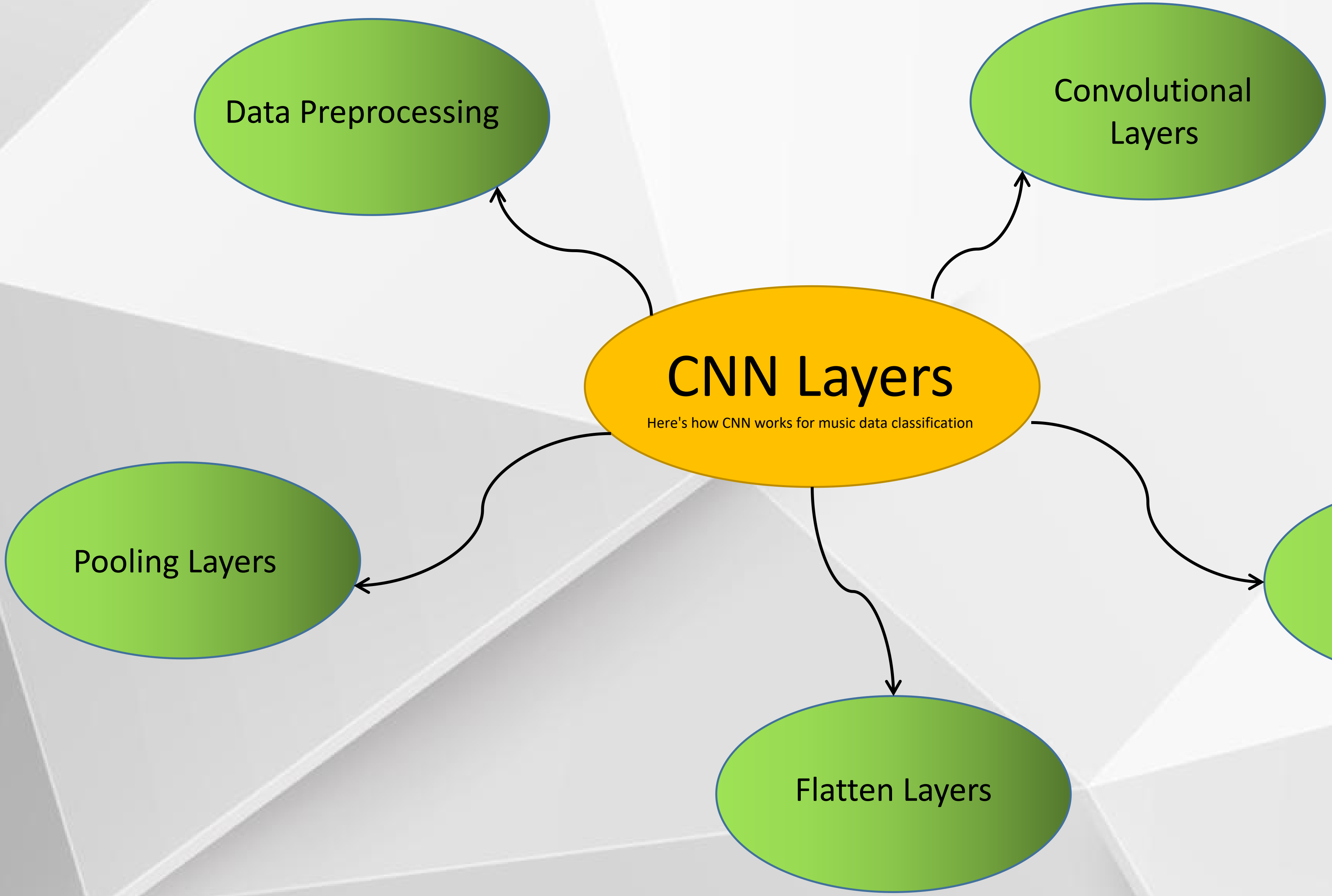
METHODOLOGY

Building the model using Convolutional Neural Networks

•CNNs are particularly useful for music classification tasks that require analysis of spectrogram images.

•The 2D convolutional layers of a CNN can effectively extract features from the spectrogram images, while the pooling layers can help reduce the dimensionality of the features.

METHODOLOGY



METHODOLOGY

This code defines a convolutional neural network (CNN) model for music genre classification.

Here's a breakdown of the layers:

```
model = Sequential()
model.add(Conv2D(64, (3, 3), activation = "relu", input_shape = input_shape))
model.add(MaxPool2D((3, 3), strides=(2, 2), padding="same"))
model.add(BatchNormalization())

model.add(Conv2D(32, (3, 3), activation = "relu"))
model.add(MaxPool2D((3, 3), strides=(2, 2), padding="same"))
model.add(BatchNormalization())

model.add(Conv2D(32, (2, 2), activation = "relu"))
model.add(MaxPool2D((2, 2), strides=(2, 2), padding="same"))
model.add(BatchNormalization())

model.add(Conv2D(16, (1, 1), activation = "relu"))
model.add(MaxPool2D((1, 1), strides=(2, 2), padding="same"))
model.add(BatchNormalization())

model.add(Flatten())
model.add(Dense(64, activation="relu"))
model.add(Dropout(0.3))
model.add(Dense(10, activation="softmax"))

model.summary()
```

METHODOLOGY

Generalizing all the layers to calculate the parameters that are trainable

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 128, 11, 64)	640
max_pooling2d (MaxPooling2D)	(None, 64, 6, 64)	0
batch_normalization (BatchNormalization)	(None, 64, 6, 64)	256
conv2d_1 (Conv2D)	(None, 62, 4, 32)	18464
max_pooling2d_1 (MaxPooling2D)	(None, 31, 2, 32)	0
batch_normalization_1 (BatchNormalization)	(None, 31, 2, 32)	128
conv2d_2 (Conv2D)	(None, 30, 1, 32)	4128
max_pooling2d_2 (MaxPooling2D)	(None, 15, 1, 32)	0
batch_normalization_2 (BatchNormalization)	(None, 15, 1, 32)	128
conv2d_3 (Conv2D)	(None, 15, 1, 16)	528
max_pooling2d_3 (MaxPooling2D)	(None, 8, 1, 16)	0
batch_normalization_3 (BatchNormalization)	(None, 8, 1, 16)	64
flatten (Flatten)	(None, 128)	0
dense (Dense)	(None, 64)	8256
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 10)	650

Total params: 33,242
Trainable params: 32,954
Non-trainable params: 288

Conv2D layer:

Number of parameters = (filter_height * filter_width * input_channels * output_channels) + output_channels
For the first Conv2D layer, the number of parameters = (3 * 3 * 1 * 64) + 64 = 640
For the second Conv2D layer, the number of parameters = (3 * 3 * 64 * 32) + 32 = 18,464
For the third Conv2D layer, the number of parameters = (3 * 3 * 32 * 32) + 32 = 4,128
For the fourth Conv2D layer, the number of parameters = (3 * 3 * 32 * 16) + 16 = 528

MaxPooling2D layer:

This layer does not have any trainable parameters.

BatchNormalization layer:

Number of parameters = 4 * output_channels
For each BatchNormalization layer in the model, the number of parameters = 4 *output_channels.

Flatten layer:

This layer does not have any trainable parameters.

Dense layer:

Number of parameters = (input_neurons * output_neurons) + output_neurons
For the first Dense layer, the number of parameters = (128 * 64) + 64 = 8256
For the second Dense layer, the number of parameters = (64 * 10) + 10 = 650

Dropout layer:

This layer does not have any trainable parameters.

Therefore, the total number of trainable parameters in the model is: 640 + 18,464 + 4,128 + 528 + (4 * 64) + 8256 + 650 = 33,242.

DEMO

RESULTS

Benchmark ML Model Analysis Results

The results show that the Random Forest model achieved the highest accuracy score of 0.65035, followed by the Cross Gradient Booster model with an accuracy score of 0.62937.

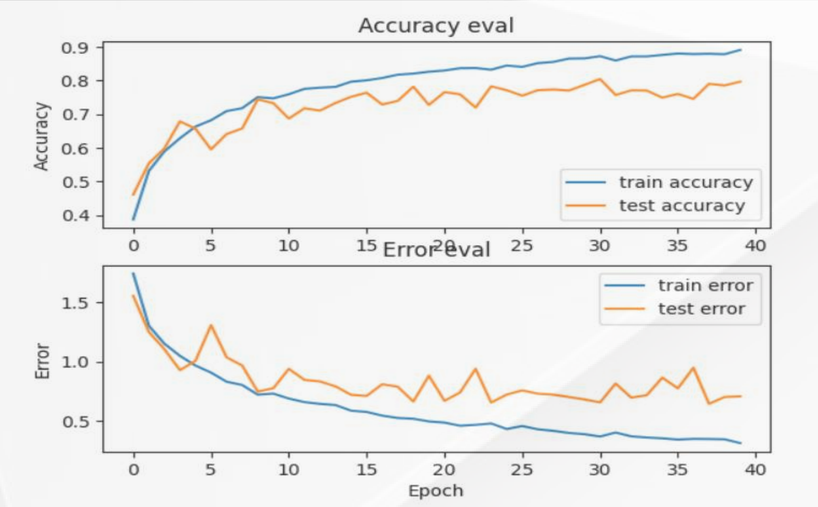
```
Accuracy Naive Bayes : 0.48252

Accuracy Stochastic Gradient Descent : 0.44056

Accuracy KNN : 0.53147

Accuracy Decission trees : 0.42657

Accuracy Random Forest : 0.65035

Accuracy Support Vector Machine : 0.41958

Accuracy Logistic Regression : 0.58741

Accuracy Neural Nets : 0.30769

Accuracy Cross Gradient Booster : 0.62937

Accuracy Cross Gradient Booster (Random Forest) : 0.6014
```

Benchmark Analysis with 10 Machine Learning Models

RESULTS

CNN Results

- The training process lasted for 40 epochs. During the training, the model was able to achieve a maximum accuracy of 78.97% on the training set while the validation accuracy peaked at 80.45%. On the other hand, the loss on the training set kept decreasing until the end of the training while the validation loss was also decreasing but fluctuated after some epochs.



Accuracy and Error Evaluations

```
model.compile(optimizer=adam,
              loss="sparse_categorical_crossentropy",
              metrics=["accuracy"])

hist = model.fit(X_train, y_train,
                 validation_data = (X_val, y_val),
                 epochs = 40,
                 batch_size = 32)
```

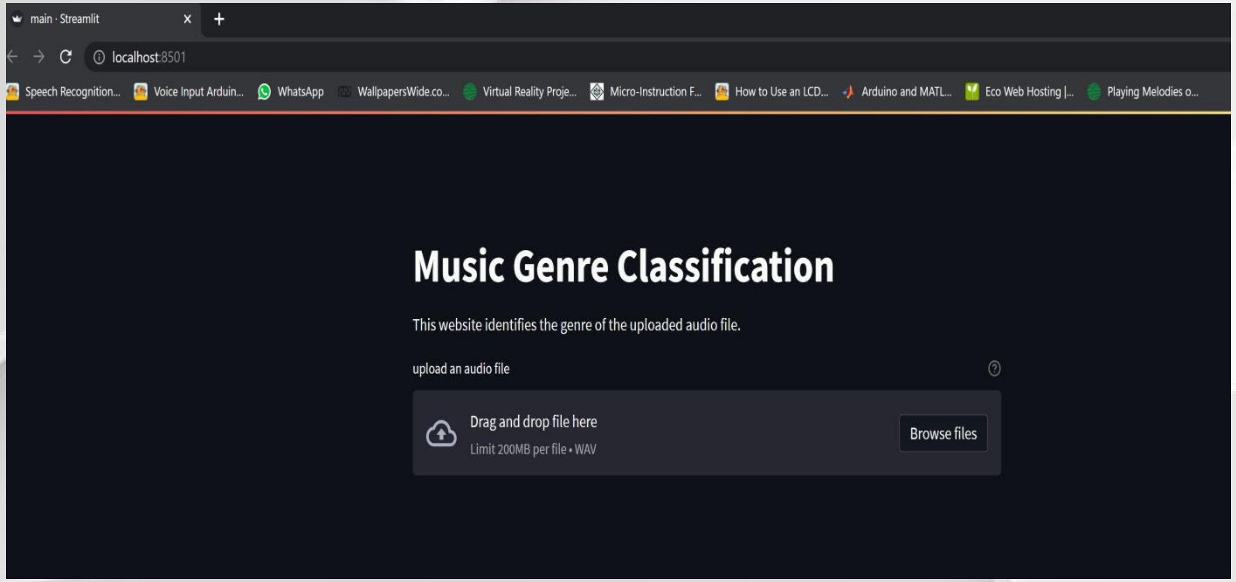
Code snippet used for training the model

RESULTS

Web Application Analysis for Classification

- The PyCharm IDE and Streamlit Python library were used to build a music genre classification system. The system employs a Convolutional Neural Network (CNN) and Mel-Frequency Cepstral Coefficients (MFCC) with the GTZAN dataset. The purpose of the system is to demonstrate the accuracy of the CNN model in predicting the genre of a user-selected song. The user can either browse or drag-and-drop an .mp3 file for genre classification.

Below is the figure representing the web page that is used for displaying the results of the music genre classification system built using PyCharm IDE and Streamlit python library.



RESULTS

- In machine learning, the predictions made by a model are based on probabilities calculated by the model.

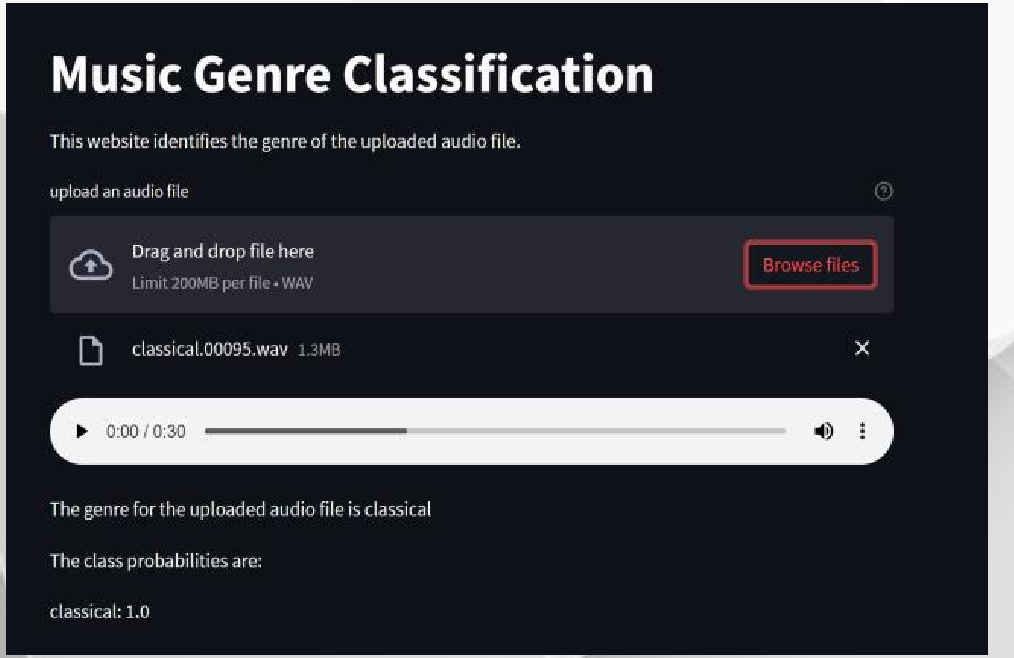
- The model assigns probabilities to each possible outcome, and the highest probability outcome is chosen as the prediction.

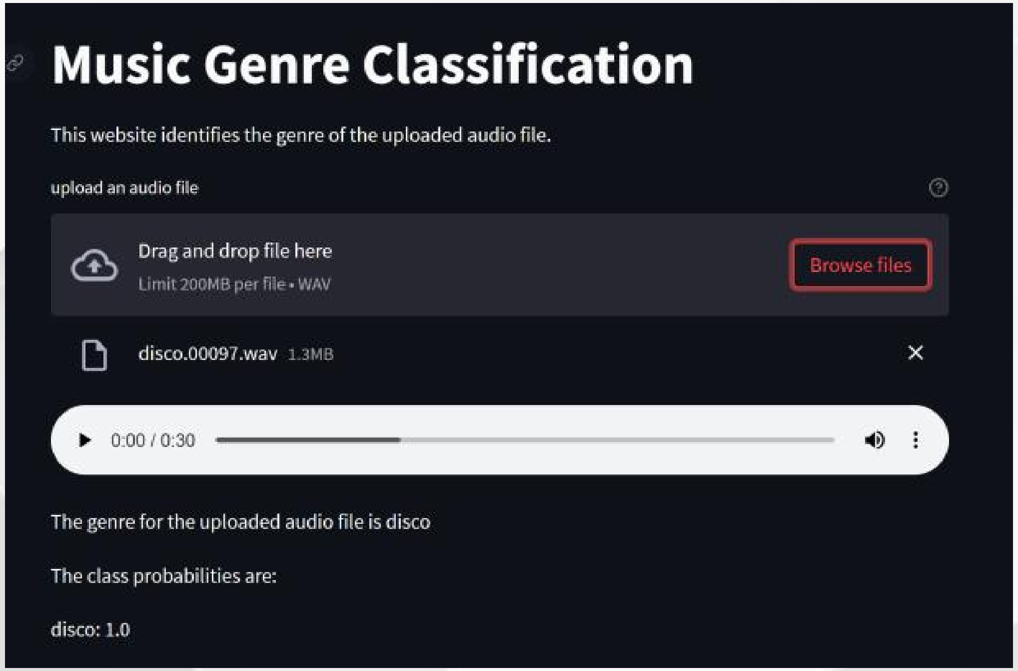
- As a result, the model assigns relatively high probabilities to both genres.

- The training data might not fully represent all possible variations in the music genre, which can lead to the model being less accurate when it comes to predicting less common or less well-defined genres.

RESULTS

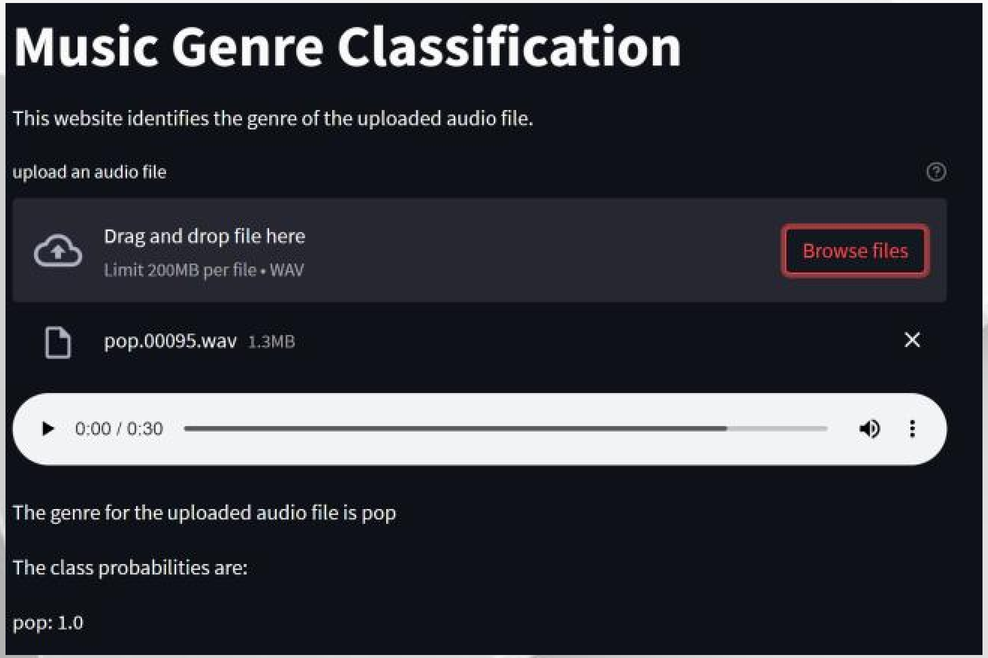
Case 1 : Accurate Predictions of the Music Genre with respective to the music file browsed.

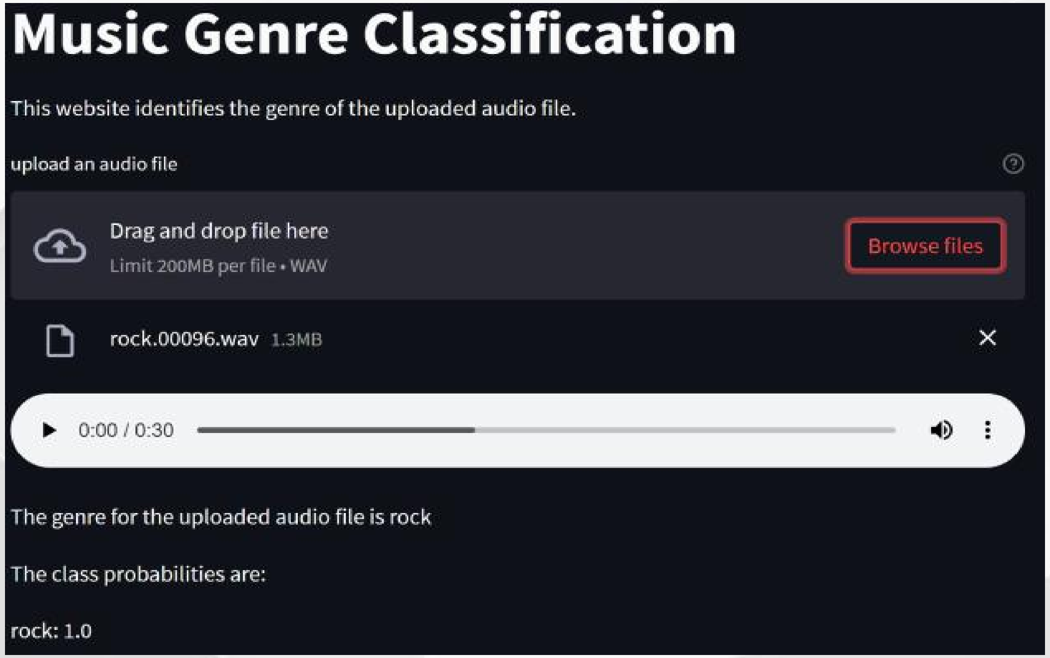




RESULTS

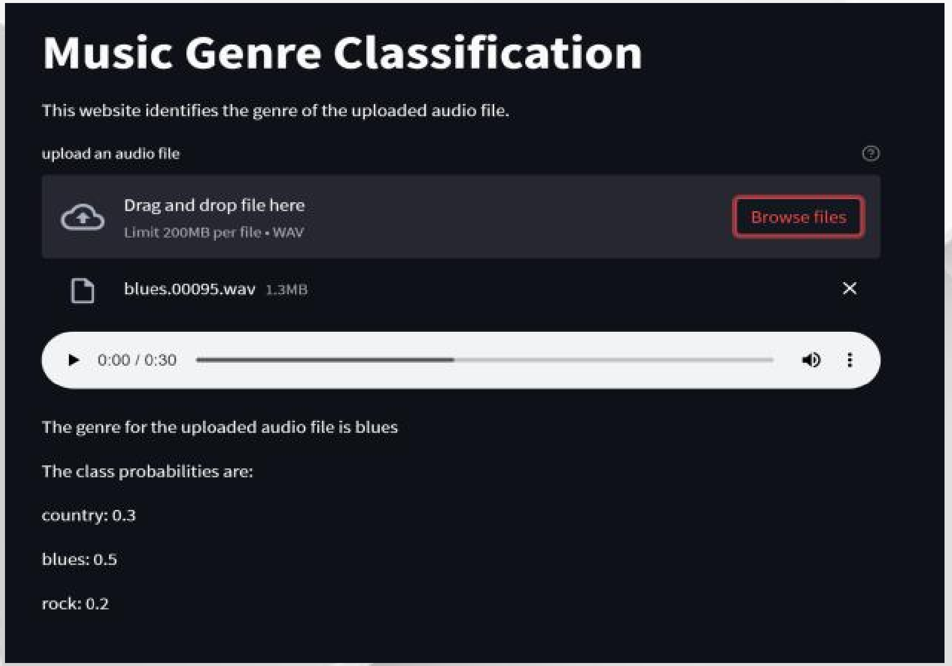
Case 1 : Accurate Predictions of the Music Genre with respective to the music file browsed.

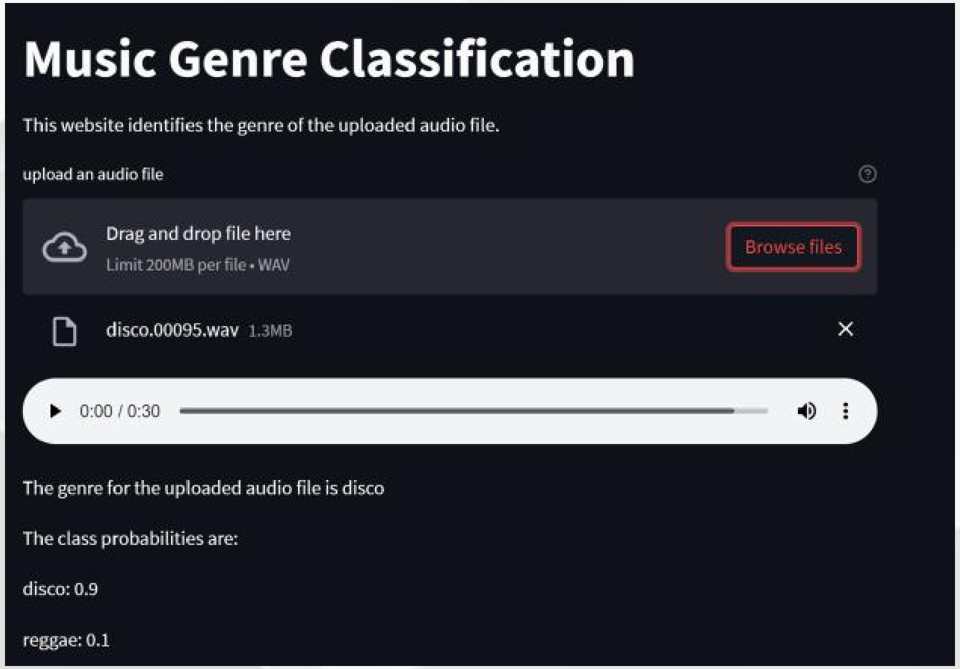




RESULTS

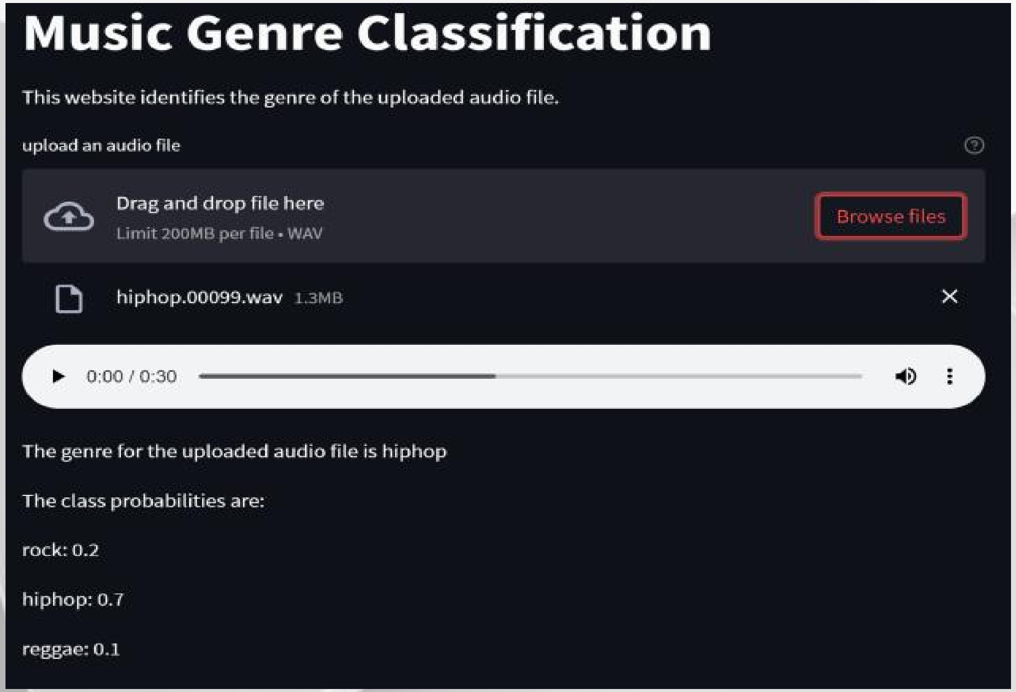
Case 2 : Predictions of the Music Genre with respective to the music file browsed along with a suitable probability dictionary of other genres to the audio.

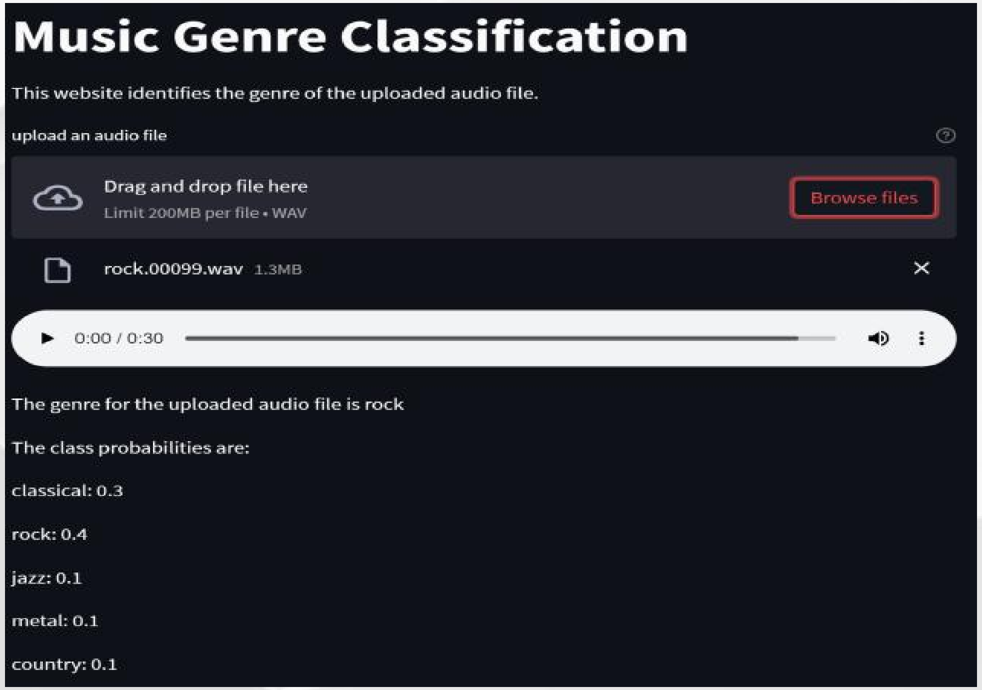




RESULTS

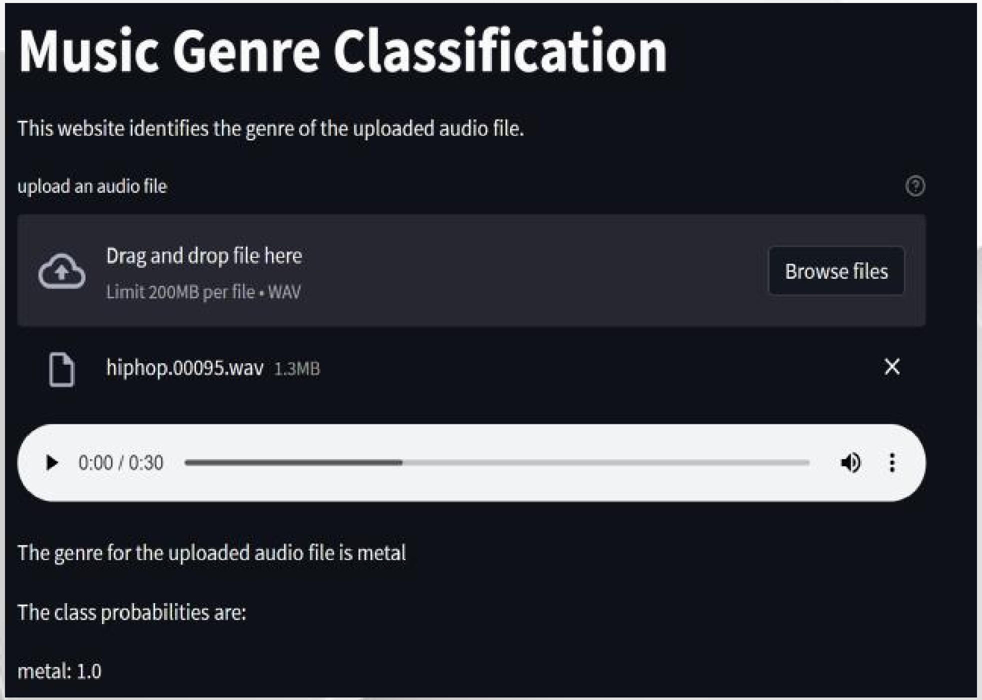
Case 2 : Predictions of the Music Genre with respective to the music file browsed along with a suitable probability dictionary of other genres to the audio.

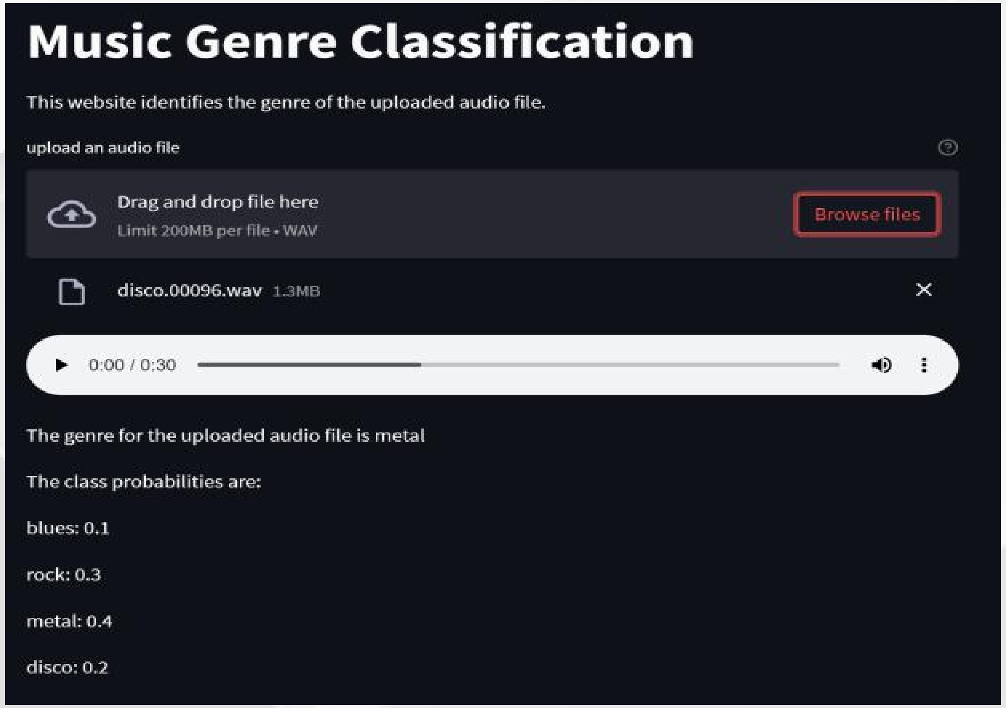




RESULTS

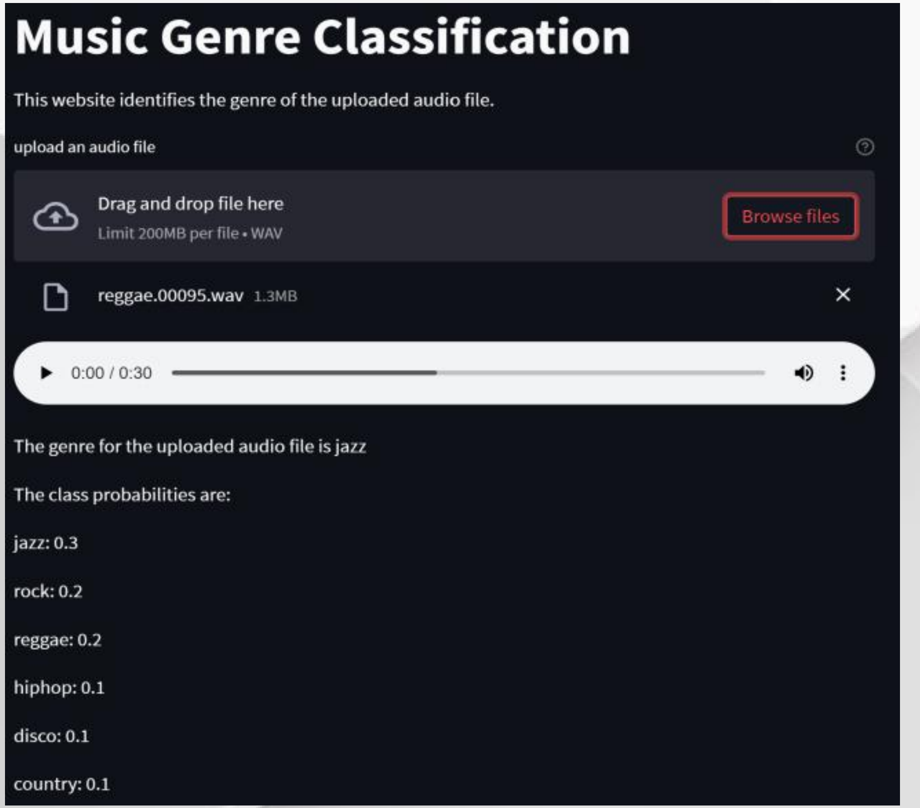
Case 3 : Incorrect Predictions of the Music Genre with respective to the music file browsed

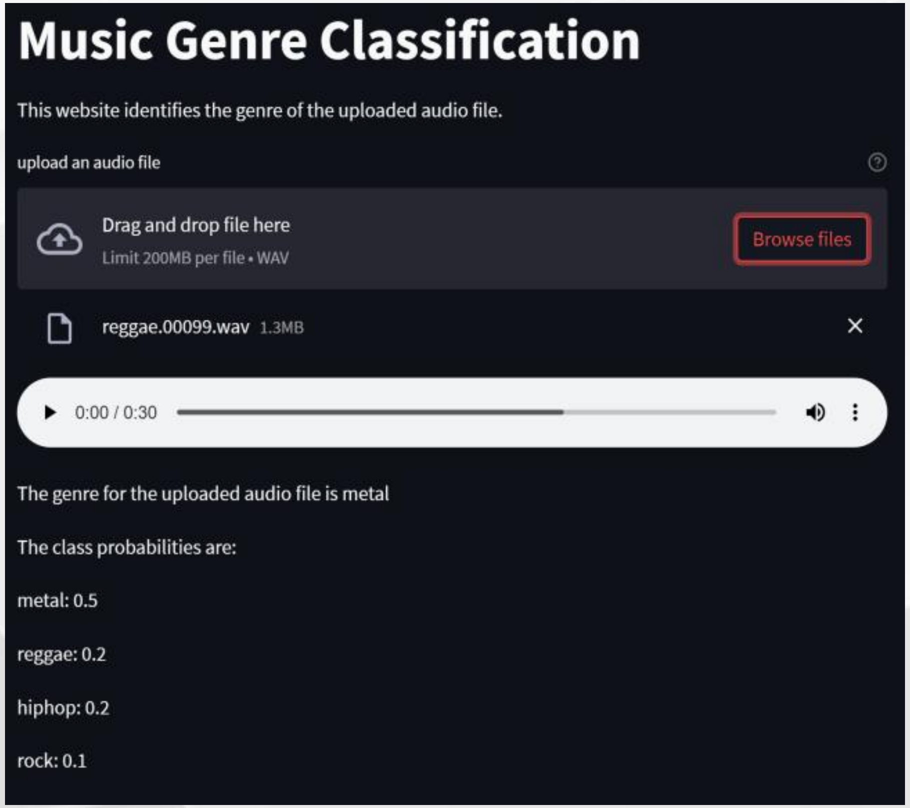




RESULTS

Case 3 : Incorrect Predictions of the Music Genre with respective to the music file browsed





CONCLUSION

- In conclusion, this project successfully demonstrated the effectiveness of using Convolutional Neural Networks for music genre classification.

- I achieved a high level of accuracy using both traditional machine learning methods and deep learning techniques, with the CNN method outperforming the ML method.

- This web application, which was developed using Streamlit, provides a simple and intuitive interface for users to upload their own audio files and classify them by genre.

THANKS