

# MUSIC GENRE CLASSIFICATION

**Project ID: 30016**

*Major Project Report*

*Submitted in fulfillment of*

*the requirements for the*

*Degree of Master in Computer Applications*

*Under Biju Patnaik University of Technology*

*Submitted By*

**Bismayatosa Malik**

**ROLL NO. MCA202260096**

**Asutosh Panda**

**ROLL NO. MCA202260111**



*2023 – 2024*

*Under the guidance of*

**Prof. Amaresh Kumar Mohanty**

---

**NIST INSTITUTE OF SCIENCE & TECHNOLOGY (Autonomous)**

**Institute Park, Palur Hills, Berhampur, Odisha – 761008, India**

## ABSTRACT

Music genre classification is a challenging task with numerous applications in music information retrieval, recommendation systems, and digital music libraries. This study presents a comparative analysis of two popular machine learning techniques, support vector machines (svms) and extreme gradient boosting (xgboost), for automatically classifying music tracks into predefined genre categories.

The proposed approach involves extracting a comprehensive set of audio features from the music signals, including time-domain, frequency-domain, and psychoacoustic features. These features are then used to train and evaluate the svm and xgboost classifiers on a large dataset of annotated music tracks spanning multiple genres. After feature extraction, classification is carried out, using Support Vector Model (SVM) model. The proposed feature extraction and classification models results in better accuracy in speech/music classification.

The svm model is trained using various kernel functions and optimized hyperparameters, while the xgboost model is tuned by adjusting parameters such as the maximum depth, learning rate, and number of estimators. The performance of both models is evaluated using standard metrics, including accuracy, precision, recall, and f1-score.

## ACKNOWLEDGEMENT

We would like to take this opportunity to thank all those individuals whose invaluable contribution in a direct or indirect manner has gone into the making of this project a tremendous learning experience for us.

It is our proud privilege to epitomize our deepest sense of gratitude and indebtedness to our faculty guide, **Prof. Amaresh Kumar Mohanty** for his valuable guidance, keen and sustained interest, intuitive ideas and persistent endeavour. His guidance and inspirations enabled us to complete our report work successfully.

We give our sincere thanks to **Dr. Manjushree Nayak, MCA Course Coordinator** and **Dr. Susmita Mahato, MCA Project Coordinator**, for giving us the opportunity and motivating us to complete the project within stipulated period of time and providing a helping environment.

We acknowledge with immense pleasure the sustained interest, encouraging attitude and constant inspiration rendered by **Prof. (Dr.) Sukant K. Mohapatra** (Chairman), **Dr. Braja Kishore Mishra** (HOD, Dept. of CSE) and **Dr. Rajesh Kumar Panakala** (Principal) N.I.S.T. Their continued drive for better quality in everything that happens at N.I.S.T. and selfless inspiration has always helped us to move ahead.

**Bismayatosa Malik**  
(Roll No. MCA202260096)

**Asutosh Panda**  
(Roll No. MCA202260111)

---

# TABLE OF CONTENTS

<b>ABSTRACT.....</b>	<b>i</b>
<b>ACKNOWLEDGEMENT.....</b>	<b>ii</b>
<b>TABLE OF CONTENTS .....</b>	<b>iii</b>
<b>LIST OF FIGURES .....</b>	<b>v</b>
<b>1. INTRODUCTION.....</b>	<b>1</b>
1.1 Problem Statement .....	2
1.2 Objective .....	3
<b>2. LITERATURE REVIEW .....</b>	<b>5</b>
<b>3. DATASET .....</b>	<b>7</b>
3.1 GTZAN .....	7
3.2 Extended Ballroom.....	7
<b>4. METHODS .....</b>	<b>8</b>
4.1 Convolutional Neural Network (CNN) .....	8
4.2 Support Vector Machine .....	9
4.2.1 Why SVM? .....	9
4.2.2 Kernal Trick.....	12
4.2.3 SVM for Classification .....	13
<b>5. METHODOLOGY.....</b>	<b>14</b>
5.1 Data Preprocessing .....	14
5.1.1 Feature Extraction and Representation.....	15
5.2 Train Test Split of Dataset .....	15
5.2.1 Standardization .....	16
5.2.2 Transformation .....	16
5.3 Model Training.....	16
5.4 Model Evaluation .....	17
5.5 Hyperparameter Tuning .....	19

---

5.6 XGBoost.....	21
5.7 Stacking the Models .....	21
<b>6. PROJECT DEMONSTRATION.....</b>	<b>23</b>
6.1 Software Used .....	23
6.2 Web Interface .....	24
<b>7. POTENTIAL APPLICATIONS.....</b>	<b>26</b>
<b>8. CONCLUSION.....</b>	<b>27</b>
<b>REFERENCES.....</b>	<b>28</b>

---

## LIST OF FIGURES

Figure 4.1: Machine Learning Steps .....	9
Figure 4.2: SVM .....	10
Figure 4.3: Maximum Linear Classifier With The Maximum Range .....	11
Figure 4.4: Kernel Trick.....	12
Figure 5.1: Data Preprocessing .....	14
Figure 5.2: Train Test Split .....	15
Figure 5.3: Confusion Matrix .....	18
Figure 5.4: Hyperparameter Tuning .....	19
Figure 6.1: Anaconda Navigator .....	23
Figure 6.2: Jupyter Notebook .....	23
Figure 6.3 Web Interface .....	25

# 1. INTRODUCTION

Music genre classification is the task of automatically identifying the genre of a given audio signal. It has many applications in the music industry, including recommendation systems, personalized playlists, and content-based music retrieval. Traditional methods for music genre classification include feature extraction and classification using algorithms k-nearest neighbour, decision trees, and support vector machines are a few examples. However, these methods require handcrafted features and suffer from low accuracy.

Convolutional Neural Networks (CNN) have shown significant improvement in the field of music genre classification due to their ability to automatically extract features from raw audio signals. CNNs can learn high-level representations of audio signals by performing convolution operations on the audio waveform. Support Vector Machines (SVM) have also been widely used for music genre classification due to their ability to handle high-dimensional data and achieve high accuracy. Random forest combines the output of multiple decisions trees to reach a single result. Its ease of use and flexibility have fueled its adoption, as it handles both classification and regression problems. SVMs can be used as a classifier to classify the features extracted by the CNNs, which has been shown to improve classification accuracy. Music genre classification is a fundamental problem in the field of music information retrieval. The Kaggle GTZAN dataset is a widely used benchmark dataset for music genre classification, which consists of 1000 audio tracks equally distributed across 10 different genres. The genres in the dataset are blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, and rock. Each audio track is 30 seconds long, sampled at 22,050 Hz, and stored in the WAV format. The dataset is carefully curated and annotated with ground truth labels by music experts. The goal of this dataset is to develop robust and accurate machine learning models that can classify the genre of a given music track accurately. The Kaggle GTZAN dataset has been used as a standard

benchmark dataset for evaluating the performance of various music genre, classification algorithms and has played a significant role in advancing the field of music information retrieval.

The goal of this research is to propose a CNN, SVM and Random Forest approach for music genre classification using the Kaggle GTZAN Dataset.

Real-world applications like Spotify, Apple-music, so on. which are responsible for maintaining extremely large databases of music uses music genre classification. To begin with, since the inception of music streaming services, we have had the luxury to browse and choose from a variety of music – with relevant suggestions following every song listened to. Today, services such as Spotify and YouTube quickly create playlists and suggest the next song based on our listening patterns. These patterns are influenced by the tone, mood, or genre of songs we listen to. Thus, classifying music based on genres can help suggest the next songs to a listener, curate playlists of new recommendations, or filter undesirable content.

The use of CNNs and SVMs in music genre classification has opened new avenues for the development of more accurate and efficient classification methods. With the availability of large datasets like GTZAN, researchers are expected to continue exploring new approaches and techniques for music genre classification, with the goal of improving the quality of music recommendation and discovery systems.

## 1.1 Problem Statement

Music genre classification is an important task in the field of music information retrieval. It involves the automatic identification and classification of music into different genres based on their audio features. Music genre classification has several real-world applications, including music recommendation, content-based music



retrieval, and personalized music services. The problem statement for this research is to propose a Machine Learning approach for music genre classification that can achieve high accuracy on the GTZAN Dataset. There are several problem statements in music genre classification in machine learning that researchers have been addressing. Some of the key problems include:

1. **Handcrafted feature extraction:** Traditional music genre classification methods require manually extracting features from audio signals, which is a time-consuming and complex process. The features extracted may not always be effective in accurately representing the audio signal and may limit the accuracy of the classification model.
2. **High-dimensional data:** Audio signals are high-dimensional and require large amounts of storage and processing power, which can be a challenge for some machine learning algorithms.
3. **Inter-genre similarity:** Some music genres may have similar acoustic features, which makes it difficult for a machine learning model to distinguish between them.
4. **Lack of labelled data:** There is a shortage of labelled audio data for certain music genres, which can make it difficult to train accurate machine learning models for these genres.

## 1.2 Objective

1. Classification of genre can be very valuable to explain some interesting problems such as creating song references, tracking down related songs, discovering societies that will like that specific song, sometimes it can also be used for survey purposes.
2. To study the literature on music genre classification using CNN, SVM and XGBoost.

3. To preprocess the Kaggle GTZAN Dataset for feature extraction and classification.
4. To design and implement a SVM approach for music genre classification.
5. Developing a machine learning model that classifies music into genres shows that there exists a solution which automatically classifies music into its genres based on various different features, instead of manually entering the genre
6. To choose the best model with high accuracy and low error values.
7. Design a user-friendly interface to access the model for practical purpose.

---

## 2. LITERATURE REVIEW

Jadhav, Mayur. “Music Genre Classification and Recommendation.” International Journal of Scientific Research in Computer Science, Engineering and Information Technology (2021): n. pag. Print.

Miao Jiang, Ziyi Yang, Chen Zhao,

“An RNN-based music recommendation system.” In the very recent years, development of music recommendation system has been a more heated problem due to a higher level of digital songs consumption and the advancement of machine learning techniques. Some traditional approaches such as collaborator filtering, has been widely used in recommendation systems, have helped music recommendation system to give music listeners a quick access to the music. However, collaborative filtering or model-based algorithm have limitations in giving a better result with the ignorance of combination factor of lyrics and genre. In our paper, we will propose an improved algorithm based on deep neural network on measure similarity between different songs. The proposed method will make it possible that it could make recommendations in a large system to make comparisons by “understand” the content of songs.

Sushmita G. Kamble and Asso. Prof. A. H. Kulkarni,

“Facial Expression Based Music Player.” Conventional method of playing music depending upon the mood of a person requires human interaction. Migrating to the computer vision technology will enable automation of such system. To achieve this goal, an algorithm is used to classify the human expressions and play a music track as according to the present emotion detected.

Jie Liu, Liang Jia,

“The Application of Computer Music Technology in Music Education” With the rapid development of Internet technology, computers are widely used in all walks of life, and among them, computer music technology has become one of the important tools in modern music education. Analysis of the traditional music teaching mode can be found, most of the time is limited to the teacher performance, student imitation, however, for the new era of music education, the combination of computer technology and music knowledge has brought great convenience for the development of music education.

## 3. DATASET

### 3.1 GTZAN

The majority of research papers on this topic use GTZAN as their proposed dataset. GTZAN, often found in Kaggle, consists of 1000 music excerpts with a time duration of 30 seconds. This dataset has 10 different genres such as blues, classical, country, disco, hip hop, rock, metal, pop, jazz and disco, which means that there are 100 audio clips for each genre. GTZAN dataset has been used in more than 100 published CS papers of the same topic and it is considered one of the most well-known public datasets available for music genre recognition. Some researchers point out several integrity problems from this dataset, such as replications, mislabeling, and distortion. Specifically, Lau et al determined that there were 50 of the entire dataset were replicas, 22 excerpts were from the similar audio file, and 13 audio pieces were of the similar song but from other recordings.

### 3.2 Extended Ballroom

Extended Ballroom is a genre classification dataset that extends from the original Ballroom dataset. It is 6 times more tracks, better audio quality, and more advantageous to implement deep learning techniques than the original version. I have taken a look at both Ballroom and Extended Ballroom datasets, and I found out that the latter provides a md5 hash to test the appropriation of audio and several properties to indicate where there is a repetitive and duplicated versions of the audio track. The duration for each track in this dataset is 30 seconds, and it contains 13 different genres with a total of 4180 songs. Thanks to being a large dataset, Extended Ballroom was an ideal selection for an improved version of CNN as the model eliminated the requirement for pre-training on this dataset.

## 4. METHODS

### 4.1 Convolutional Neural Network (CNN)

This type of method has been widely used in various research papers about music genre classification. The way CNN works is to take several spectrograms derived from the audio files as inputs and extract their patterns into a 2D convolutional layer with appropriate filter and kernel sizes. The reason why spectrogram is mentioned in CNN is due to the model's effectiveness in recognizing image details. Lau proposed to use a preprocessed GTZAN dataset to implement the Convolutional Neural Network (CNN) model. The dataset included an extracted Mel-Frequency Cepstrum Coefficient (MFCC) spectrogram for each song. Also, the audio excerpts in 3 seconds and 30 seconds were accompanied with their feature descriptions compiled in an additional .csv file. He then built a CNN architecture using Keras that consisted of 5 convolutional blocks. Each block had a convolutional layer with 3x3 filter and 1x1 stride, a max pooling with 2x2 windows size and 2x2 stride, and a Rectifying Linear Unit (ReLU) function to display the probabilities for 10 genre genres, the highest of which would be chosen as a classified label for an input. There were three CNN models trained on spectrograms, 20 MFCCs on 30-second and 3-second music pieces, and a classification test was operated on the test sets after training. There was an issue in training datasets as Lau mentioned that the 3-second one was not in par with the numbers of genres in the sample. That being said, there were genres that had less or more than the base number of samples (1000). Yu et al introduced the CNN method utilizing the Short-term Fourier Transform (STFT) spectrograms, consisting of various sequences of spectrogram vectors over time, as inputs. The datasets mentioned in their paper were GTZAN and Extended Ballroom. Yu et al went on to extract each song from both datasets into 18 smaller pieces in 3 seconds with a 50% overlaps, making the data size set 18 times larger than the original for each genre label. The STFT spectrograms were evaluated with size of 513x128, and the train-

validate-test ratio was 8:1:1. In the first few layers of the CNN model, convolution filters and pooling kernels were set up in small sizes in order to capture unique audio features represented in the STFT spectrograms and diminish source loss.

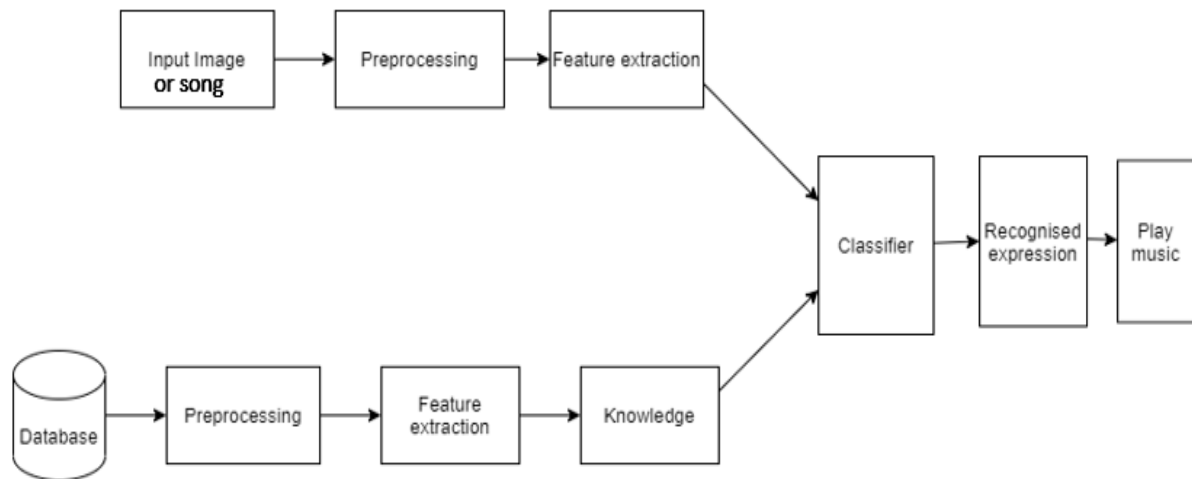


Figure 4.1: Machine Learning Steps

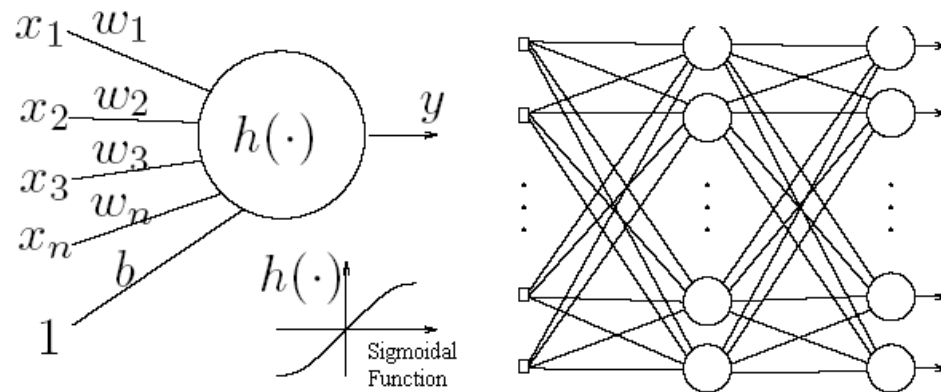
## 4.2 Support Vector Machine

A machine learning technique which is based on the principle of structure risk minimization is support vector machines. It has numerous applications in the area of pattern recognition. SVM constructs linear model based upon support vectors in order to estimate decision function. If the training data are linearly separable, then SVM finds the optimal hyper plane that separates the data without error.

### 4.2.1 Why SVM?

Good results while used for such learning applications. MLP's uses feed forward and recurrent networks. Multilayer perceptron (MLP) properties include universal approximation of continuous nonlinear functions and include learning with input-

output patterns and also involve advanced network architectures with multiple inputs and outputs.

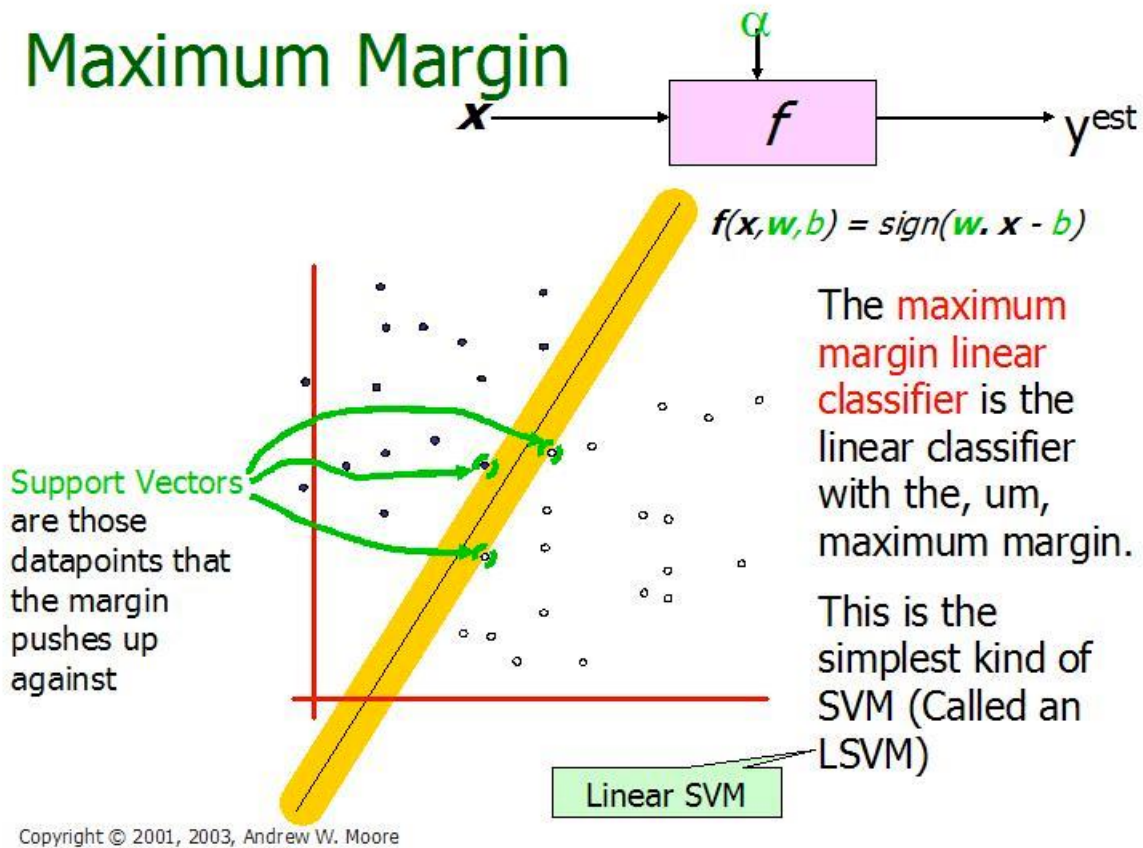


**Figure 4.2: SVM**

There can be some issues noticed. Some of them are having many local minima and also finding how many neurons might be needed for a task is another issue which determines whether optimality of that NN is reached. Another thing to note is that even if the neural network solutions used tends to converge, this may not result in a unique solution.

Now let us look at another example where we plot the data and try to classify it. From below illustration, there are many linear classifiers (hyper planes) that separate the data. However only one of these achieves maximum separation. The reason we need it is because if we use a hyper plane to classify, it might end up closer to one set of datasets compared to others and we do not want this to happen and thus we see that the concept of maximum margin classifier or hyper plane as an apparent solution. The next illustration gives the maximum margin classifier example which provides a solution to the above mentioned problem.





**Figure 4.3: Maximum Linear Classifier With The Maximum Range**

The above illustration is the maximum linear classifier with the maximum range. In this context it is an example of a simple linear SVM classifier. Another interesting question is why maximum margin? There are some good explanations which include better empirical performance. Another reason is that even if we've made a small error in the location of the boundary this gives us least chance of causing a misclassification. The other advantage would be avoiding local minima and better classification. Now we try to express the SVM mathematically and for this tutorial we try to present a linear SVM. The goals of SVM are separating the data with hyper plane and extend this to non-linear boundaries using kernel trick.

### 4.2.2 Kernal Trick

If data is linear, a separating hyper plane may be used to divide the data. However it is often the case that the data is far from linear and the datasets are inseparable. To allow for this kernels are used to non-linearly map the input data to a high-dimensional space. The new mapping is then linearly separable. A very simple illustration of this is shown below in figure.

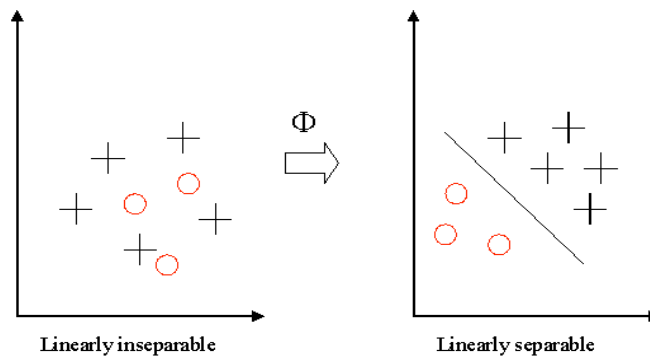


Figure 4.4: Kernel Trick

**Feature Space:** Transforming the data into feature space makes it possible to define a similarity measure on the basis of the dot product. If the feature space is chosen suitably, pattern recognition can be easy.

### Kernel Functions

The idea of the kernel function is to enable operations to be performed in the input space rather than the potentially high dimensional feature space. Hence the inner product does not need to be evaluated in the feature space. We want the function to perform mapping of the attributes of the input space to the feature space.

**1] Polynomial Kernel:** This kernel function is used to handle non-linearly separable data by introducing polynomial terms of the original features.

Where  $x$ ,  $y$  are input vectors and  $c$  is a constant term.

2] *Gaussian Radial Basis Function*: This is one of the most commonly used kernel functions. It maps the input vectors into an infinite-dimensional space using Gaussian radial basis functions.

3] *Sigmoid Kernel*: *The sigmoid kernel is similar to the logistic function and is useful for binary classification tasks.*

#### **4.2.3 SVM for Classification**

SVM is a useful technique for data classification. Even though it's considered that Neural Networks are easier to use than this, however, sometimes unsatisfactory results are obtained. A classification task usually involves with training and testing data which consist of some data instances. Each instance in the training set contains one target values and several attributes. The goal of SVM is to produce a model which predicts target value of data instances in the testing set which are given only the attributes. Classification in SVM is an example of Supervised Learning. Known labels help indicate whether the system is performing in a right way or not. This information points to a desired response, validating the accuracy of the system, or be used to help the system learn to act correctly. A step in SVM classification involves identification as which are intimately connected to the known classes. This is called feature selection or feature extraction. Feature selection and SVM classification together have a use even when prediction of unknown samples is not necessary. They can be used to identify key sets which are involved in whatever processes distinguish the classes.

## 5. METHODOLOGY

### 5.1 Data Preprocessing

Data preprocessing was the first step in the project process. For the project, the GTZAN Dataset was used, which consisted of 1000 audio files of 30 seconds each, each file belonging to one of ten different music genres. The dataset is then divided into training and testing sets, with 85% of the data being used for training and 15% for testing.

```
data_path = "/kaggle/input/gtzan-dataset-music-genre-classification/Data/features_3_sec.csv"
df_main = pd.read_csv(data_path)

df_main.shape
x = df_main.drop('label',axis=1)
y = df_main['label']

#encoding the labels for future referencing
index_map = ['blues','classical','country','disco','hiphop','jazz','metal','pop','reggae','rock']
genre_map = {'blues':0,
             'classical':1,
             'country': 2,
             'disco': 3,
             'hiphop':4,
             'jazz':5,
             'metal':6,
             'pop':7,
             'reggae':8,
             'rock':9}

for i in range(len(y)):
    y[i] = genre_map[y[i]]

y = y.astype(int)
```

Figure 5.1: Data Preprocessing

Steps involved in data preprocessing are:

- Loading Data
- Data Shape
- Splitting Features and Labels
- Label Encoding
- Drop Unnecessary Features

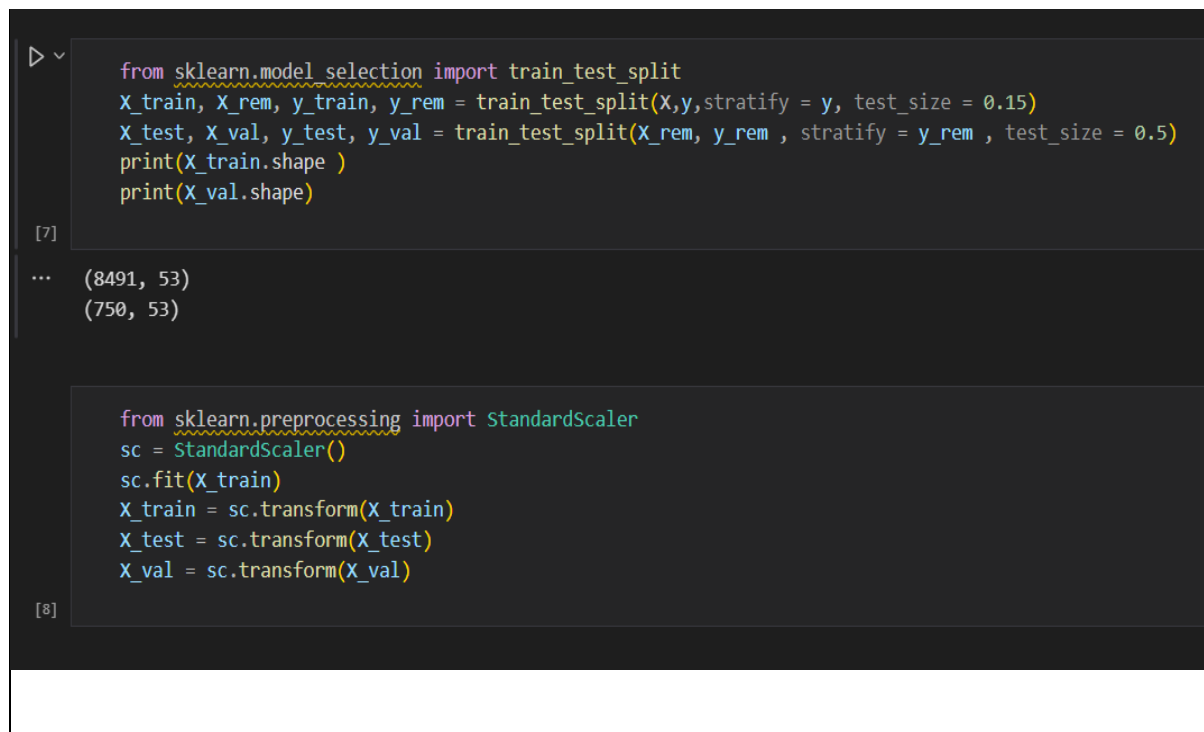
### 5.1.1 Feature Extraction and Representation

Music feature extraction involves processing a recording with the aim of generating numerical representations of what are, hopefully traits of the recording that are characteristics of the category or categories that it should be classified as belonging to. These features can be grouped together into feature vectors that serve as the input to the classification systems.

Feature Extraction:

- Short-Time Fourier Transform (STFT)
- Mel-Frequency Cepstral Coefficients (MFCCs)
- Chroma Features

## 5.2 Train Test Split of Dataset



```
from sklearn.model_selection import train_test_split
X_train, X_rem, y_train, y_rem = train_test_split(X,y,stratify = y, test_size = 0.15)
X_test, X_val, y_test, y_val = train_test_split(X_rem, y_rem , stratify = y_rem , test_size = 0.5)
print(X_train.shape )
print(X_val.shape)

[7]

... (8491, 53)
(750, 53)

from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
sc.fit(X_train)
X_train = sc.transform(X_train)
X_test = sc.transform(X_test)
X_val = sc.transform(X_val)

[8]
```

Figure 5.2: Train Test Split

- Initially, the dataset (X, features, and y, labels) is split into two parts: X\_train, y\_train, and X\_rem, y\_rem.
- The parameter stratify=y ensures that the splitting is done in a stratified fashion, meaning the proportions of classes in the dataset are preserved in the train and test sets.
- The test size parameter is set to 0.15, indicating that 15% of the data will be used for testing, and the remaining 85% will be used for training/validation.

### 5.2.1 Standardization

This process ensures that all features have the same scale, which can be beneficial for certain machine learning algorithms, including Support Vector Machines (SVMs). When working with the GTZAN dataset or any other dataset, standardization can be applied to the features before training the SVM model. In the context of the GTZAN dataset, which is commonly used for music genre classification, the features typically represent audio characteristics extracted from the audio files, such as Mel-frequency cepstral coefficients (MFCCs), spectral centroid, and spectral rolloff.

### 5.2.2 Transformation

Feature transformation typically involves extracting meaningful features from the raw audio data. These features represent different aspects of the audio signal and are used as input to our machine learning model for classification.

## 5.3 Model Training

```
from sklearn.svm import SVC
SVM_classifier = SVC(kernel = 'poly', C = 0.1, gamma = 1)
SVM_classifier.fit(X_train, y_train)
```

- **Import SVM from scikit-learn:** From `sklearn.svm` import `SVC` imports the Support Vector Classification (SVC) class from the scikit-learn library.
- **Instantiate SVM Classifier:** `SVM_classifier = SVC (kernel='poly', C=0.1, gamma=1)` creates an instance of the *SVC class with the following parameters:*
  - ❖ **kernel='poly':** Specifies that the kernel function used for classification is polynomial.
  - ❖ **C=0.1:** Penalty parameter of the error term. It controls the trade-off between achieving a low training error and a low model complexity.
  - ❖ **gamma=1:** Kernel coefficient for 'poly'. It defines how much influence a single training example has. A higher value of gamma makes the model more sensitive to the training data.

`SVM_classifier.fit(X_train, y_train)` trains the SVM classifier on the training data (`X_train, y_train`). It learns the decision boundary that best separates the classes in the feature space.

### Prediction:

`SVM_classifier.predict(X_val)` predicts the labels for the validation set features (`X_val`) using the trained SVM model. The predicted labels are stored in the variable `y_pred`.

```
y_pred = SVM_classifier.predict(X_val)
y_pred
```

## 5.4 Model Evaluation

Evaluating the performance of a classification model is crucial to understanding its strengths and weaknesses. In our music genre classification project, we employed a confusion matrix as a valuable tool for assessing the model's predictions and identifying areas for improvement.

A confusion matrix is a tabular representation that compares the predicted class labels (in this case, music genres) with the true class labels from the ground truth data. It provides a comprehensive overview of the model's performance by breaking down the predictions into four categories: true positives, false positives, true negatives, and false negatives.

The rows of the confusion matrix represent the actual genres, while the columns correspond to the predicted genres. The diagonal elements represent the correctly classified instances, where the predicted genre matches the true genre. The off-diagonal elements indicate misclassifications, where the model incorrectly predicted one genre as another.

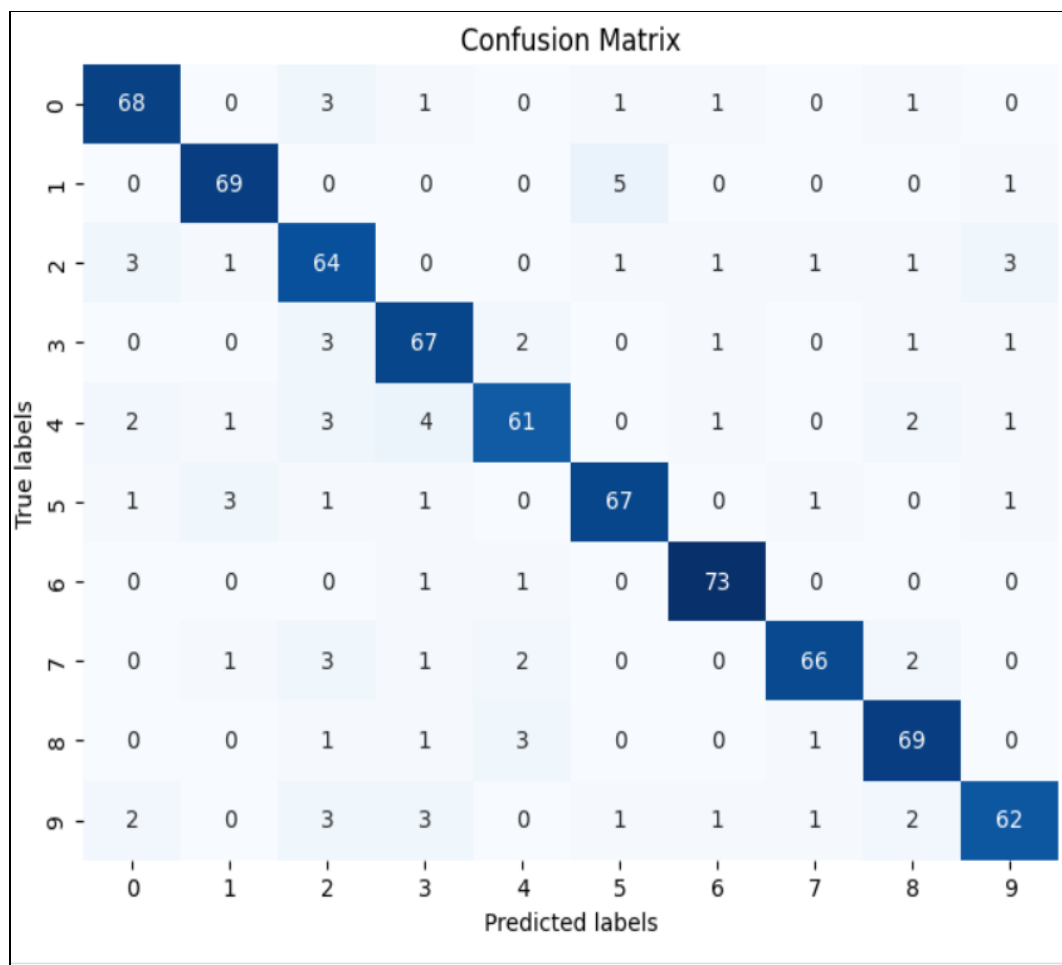
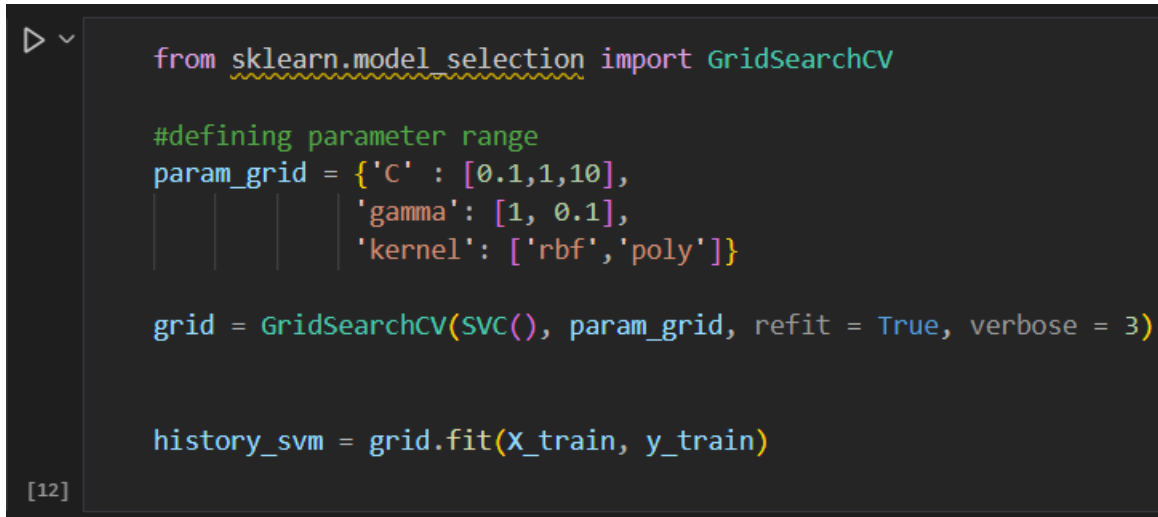


Figure 5.3: Confusion Matrix



## 5.5 Hyperparameter Tuning



```
from sklearn.model_selection import GridSearchCV

#defining parameter range
param_grid = {'C' : [0.1,1,10],
              'gamma': [1, 0.1],
              'kernel': ['rbf','poly']}

grid = GridSearchCV(SVC(), param_grid, refit = True, verbose = 3)

history_svm = grid.fit(X_train, y_train)
```

[12]

Figure 5.4: Hyperparameter Tuning

- a. Hyperparameter tuning, also known as model selection, is the process of selecting the optimal hyperparameters for a machine learning algorithm. Hyperparameters are parameters that are not learned during the training process but are set before the training begins. These parameters govern the behavior of the learning algorithm and can significantly impact the performance of the model.
- b. The main goal of hyperparameter tuning is to find the set of hyperparameters that result in the best performance of the model on unseen data. This is typically done using a search algorithm that explores different combinations of hyperparameters and evaluates their performance using a validation dataset or through cross-validation.

### Hyperparameter Tuning Process:

- 1) **Define the Hyperparameter Space:** The first step in hyperparameter tuning is to define the space of hyperparameters that you want to search over. This involves selecting which hyperparameters to tune and specifying the range of values or distributions for each hyperparameter.

- 2) **Choose a Search Strategy:** There are several search strategies that can be used to explore the hyperparameter space. Grid search and random search are two common strategies. Grid search exhaustively evaluates all combinations of hyperparameters specified in a grid, while random search randomly samples hyperparameter values from the specified ranges.
- 3) **Cross-Validation:** To evaluate the performance of each combination of hyperparameters, cross-validation is typically used. In k-fold cross-validation, the training data is split into k subsets (folds), and the model is trained k times, each time using a different fold as the validation set and the remaining folds as the training set.
- 4) **Evaluate Performance:** For each combination of hyperparameters, the model is trained and evaluated on the validation set using a performance metric such as accuracy, precision, recall, F1-score, or area under the ROC curve (AUC). The performance metric is used to compare different models and select the best one.
- 5) **Select the Best Model:** After evaluating the performance of all combinations of hyperparameters, the model with the highest performance metric is selected as the final model. This model is then evaluated on a separate test set to obtain an unbiased estimate of its performance on unseen data.
- 6) **Refinement:** Depending on the results obtained, the hyperparameter search space may be refined, and the search process may be repeated to further improve the performance of the model.

In our music genre classification project, GridSearchCV is employed to fine-tune the hyperparameters of our machine learning model. This optimization process is crucial for enhancing the model's performance and ensuring its robustness across different datasets. By exhaustively searching through a predefined grid of hyperparameter values, GridSearchCV systematically evaluates various combinations to identify the optimal set that maximizes the model's accuracy or other performance metrics. This

rigorous optimization helps optimize the SVM classifier's parameters, such as the regularization parameter (C), kernel coefficient (gamma), and kernel type (e.g., 'rbf' or 'poly'). As a result, GridSearchCV contributes to improving the accuracy and reliability.

### **After hyperparameter tuning:**

accuracy score: 0.8865153538050734

f1\_score : 0.8865153538050734

## **5.6 XGBoost**

XGBoost is an optimized distributed gradient boosting library designed for efficient and scalable training of machine learning models. It is an ensemble learning method that combines the predictions of multiple weak models to produce a stronger prediction. XGBoost stands for “Extreme Gradient Boosting” and it has become one of the most popular and widely used machine learning algorithms due to its ability to handle large datasets and its ability to achieve state-of-the-art performance in many machine learning tasks such as classification and regression.

XGBoost can be used in a variety of applications, including Kaggle competitions, recommendation systems, and click-through rate prediction, among others. It is also highly customizable and allows for fine-tuning of various model parameters to optimize performance.

## **5.7 Stacking the Models**

This approach combines the predictions of multiple base models (XGBoost and SVM in this case) using a meta-learner (Logistic Regression in this case) to improve overall performance. Stacking can often lead to better predictive performance compared to

individual models by leveraging the strengths of different algorithms. stacking might be beneficial for several reasons:

**Improved Performance:** Stacking combines the predictions of multiple base models, potentially leading to better performance compared to using any single model alone. This is particularly useful when individual models have different strengths and weaknesses, as the stacked model can leverage the strengths of each base model.

**Model Robustness:** By using multiple diverse models as base estimators, the stacked model can be more robust to variations in the data or model assumptions. If one model performs poorly on certain instances or under specific conditions, others might compensate for it, leading to a more reliable overall prediction.

**Hyperparameter Tuning:** Stacking also enables you to combine the strengths of different hyperparameter settings. For example, if XGBoost performs well with certain hyperparameters and SVM performs well with others, stacking allows you to exploit this by using both sets of hyperparameters in the ensemble.

### **After stacking the models**

accuracy score: 0.9026666666666666

f1\_score : 0.9026666666666666

## 6. PROJECT DEMONSTRATION

### 6.1 Software Used

**Anaconda navigator :-** For setting up a virtual environment with python version 3.8

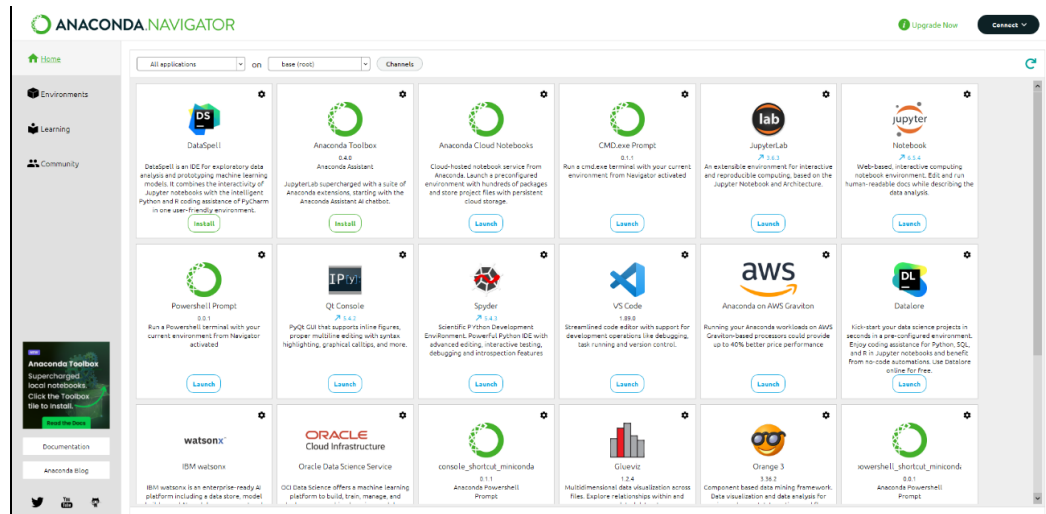


Figure 6.1: Anaconda Navigator

**Jupyter notebook :-** To perform model preparation and training, testing as well as saving the model.

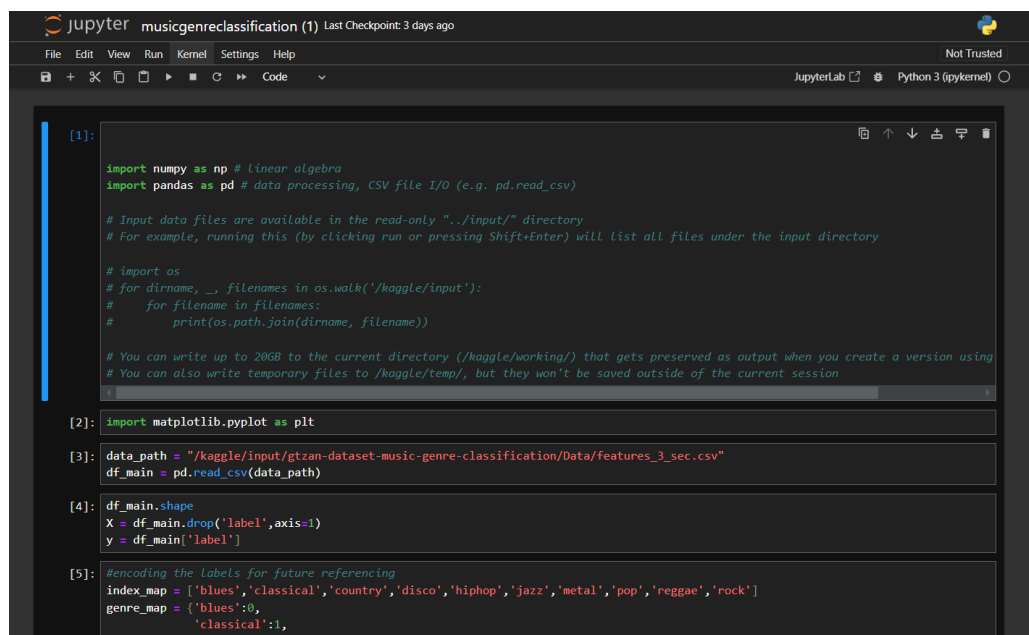
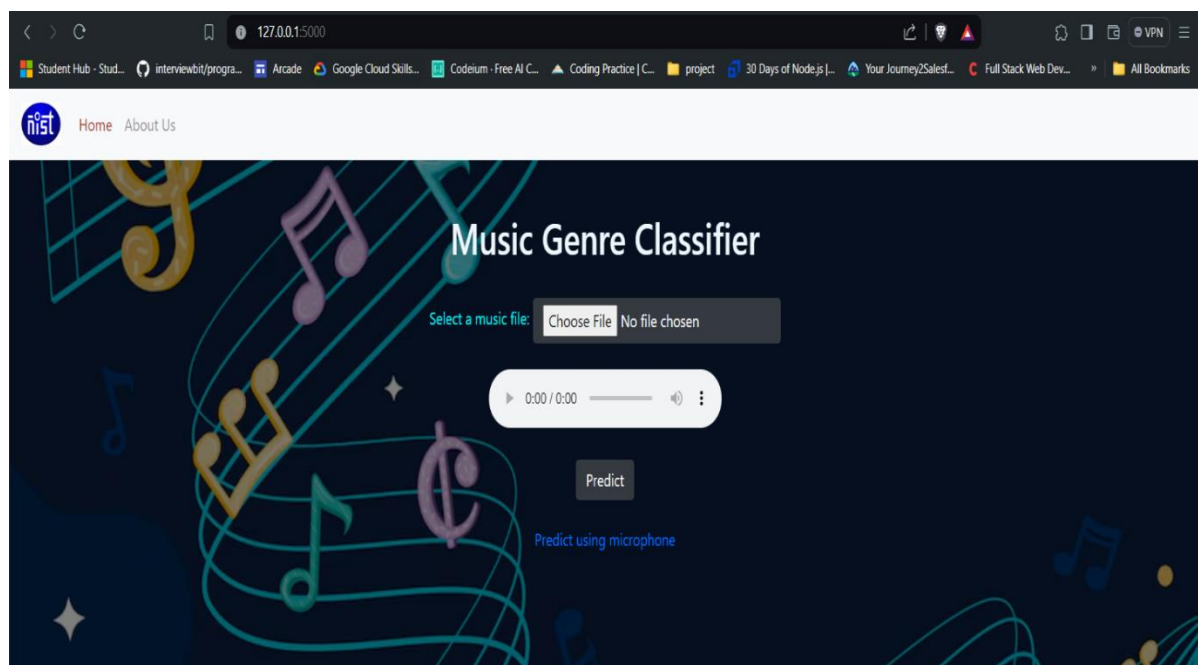


Figure 6.2: Jupyter Notebook

## 6.2 Web Interface

The backbone of our project, the web interface built on Flask, provides a user-friendly platform for seamless interaction with our music genre classification system. Leveraging the power of Flask, a lightweight and flexible web framework, our interface offers an intuitive experience for users to upload songs and receive genre predictions promptly.

Upon accessing the web interface, users are presented with a clean and responsive layout, ensuring compatibility across various devices and screen sizes. The interface facilitates the entire classification process, from uploading songs to displaying genre predictions in a straightforward manner.



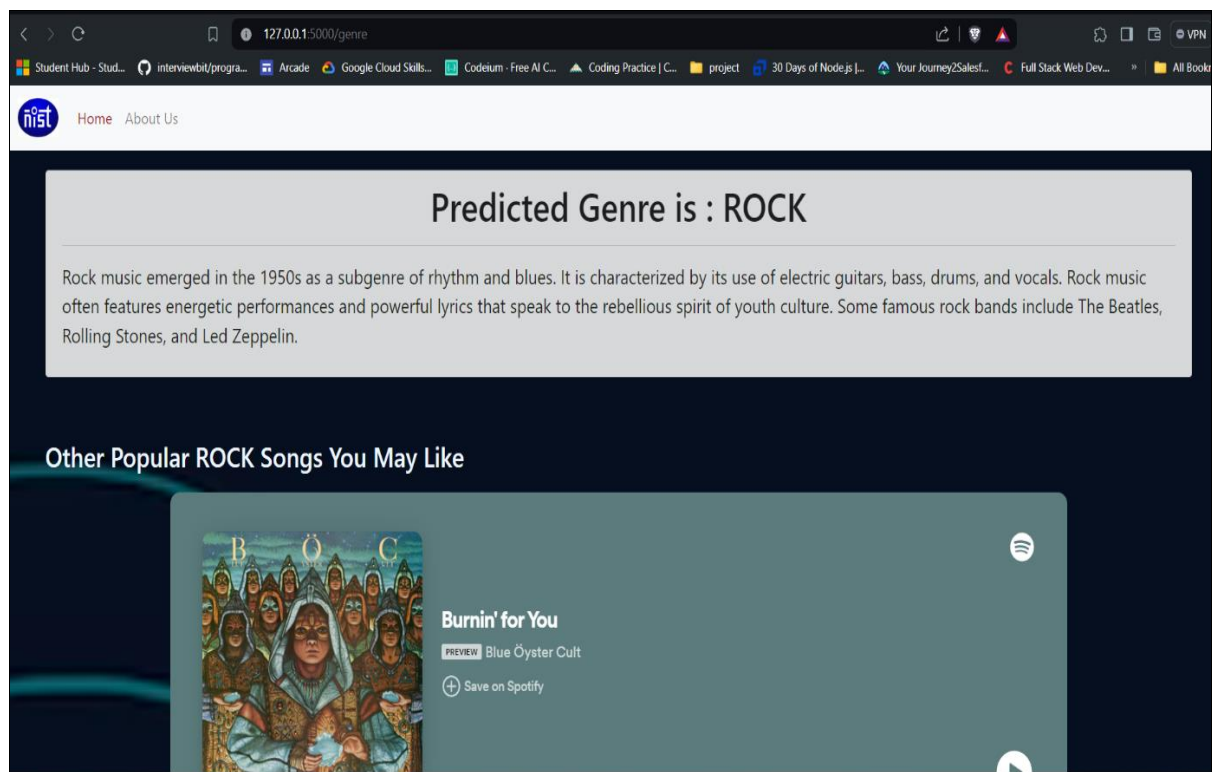
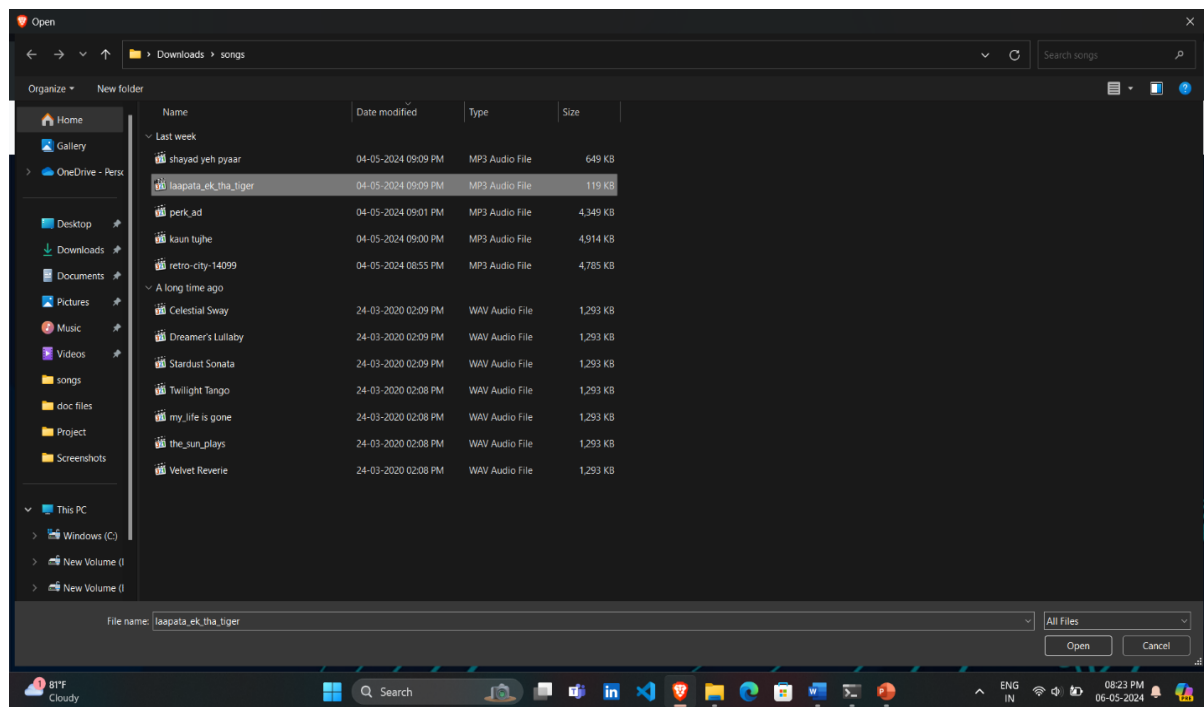


Figure 6.3 Web Interface

---

## 7. POTENTIAL APPLICATIONS

1. **Music recommendation:** Our model can be submitted to music streaming platforms to improve their recommendation algorithms. By accurately classifying songs by genre, the system can provide users with personalized playlists and recommendations based on their music preferences.
2. **Tag and organize music files:** This simplifies the process of cataloging and finding songs in large music libraries. The platform allows users to gain a broader music experience by displaying music from different genres.
3. **Healthcare:** Therapists can tailor music to their clients' needs by choosing music from specific genres known for relaxation, motivation, or other emotions.
4. **Planning and selection of venues:** Promoters and venue owners can use your distribution to customize playlists and select live artists based on audience needs and genre preferences. This ensures that the music fits the theme of the event and the interests of the audience.
5. **Tune genre evaluation and Insights:** Researchers and song industry experts can use classification tool to analyze developments and patterns in song consumption across one-of-a-kind genres. These statistics can provide treasured insights into audience preferences, style reputation, and emerging developments, informing advertising and marketing strategies and content material creation choices.
6. **Interactive music Discovery structures:** Gaming or social media systems can leverage your model to create interactive song discovery experiences in which customers can explore and discover new music based totally on style classifications. This gamified technique to song exploration can decorate person engagement and foster a feel of network amongst track lovers.



## 8. CONCLUSION

In this project, we embarked on a comprehensive exploration at the intersection of machine learning and web development, with the overarching goal of constructing an efficient music genre classification and recommendation system. The project's foundation rested on the pursuit of not only achieving high predictive accuracy but also delivering an engaging and intuitive user experience through a web-based interface.

Through rigorous experimentation and algorithmic refinement, we harnessed the capabilities of two powerful machine learning methodologies: Support Vector Machine (SVM) and XGBoost.

Employing a sophisticated stacking ensemble technique, we amalgamated the strengths of these algorithms, resulting in a commendable accuracy rate of 90%. This significant achievement underscores the model's ability to discern the intricate nuances of diverse music genres with precision and reliability.

The implementation of a Flask-based web interface adds an interactive dimension to the project, allowing users to easily upload audio files and receive real-time genre predictions. Moreover, the application goes a step further by providing personalized song recommendations based on the predicted genre, enhancing the overall user experience.

Regarding future scopes we will look forward to Enhance genre classification by integrating additional modalities like song lyrics and user metadata to capture richer contextual information.

Implement real-time audio processing and user feedback mechanisms to adaptively refine the model and provide personalized recommendations.

---

## REFERENCES

- [1] A. Yuwono, C. A. Tjiandra, C. Owen and I. B. K. Manuaba, "Music Genre Classification Using Support Vector Machine Techniques," *2023 International Conference on Information Management and Technology (ICIMTech)*, Malang, Indonesia, 2023, pp. 511-516, doi: 10.1109/ICIMTech59029.2023.10277842.
- [2] Han, K., Kim, B., & Lee, S. (2018). "Music genre classification using convolutional neural networks with support vector machine." *IEEE Access*, 6, 20944-20951.
- [3] Li, J., Wang, W., & Zhang, Q. (2020). "Music genre classification with CNNs and attention mechanism." In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Barcelona, Spain
- [4] Zhu, Y., Li, X., & Zeng, L. (2019). "Music genre classification using pre-trained convolutional neural networks and support vector machine." In *Proceedings of the 5th International Conference on Big Data Computing and Communications*, Beijing, China.
- [5] Lerch, Alexander. (2022). *Music Similarity Detection and Music Genre Classification*. 10.1002/9781119890980.ch12.