

PROFESSIONAL TRAINING REPORT - II

entitled

AI Powered Music Genre Classifier

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering
with specialization in Cybersecurity

by

JOSHUA SHAJI [41614036]

ABRAM ROY THOMAS[41611003]

ARJUN GOPAL[41611015]

MUTYALAPATI MANISHVARDHAN [41611123]



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
CATEGORY -1 UNIVERSITY BY UGC**

**Accredited with Grade "A++" by NAAC | 12B Status by UGC | Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119**

MAY 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

CATEGORY-I UNIVERSITY BY UGC

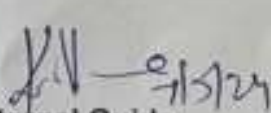
Accredited "A++" Grade by NAAC | 12B Status by UGC | Approved by AICTE

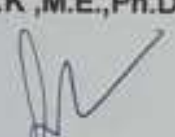
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Professional Training is the bonafide work of **Joshua Shaji (41614036)**, **Abram Roy Thomas (41611003)**, **Arjun Gopal (41611015)**, **Mutyalapati Manishvardhan (41611123)** who carried out the project entitled "**AI Powered Music Genre Classifier**" under my supervision from January 2024 to April 2024.


Internal Guide
Dr.VEENA.K ,M.E.,Ph.D.,


Head of the Department
Dr. S. VIGNESHWARI, M.E., Ph.D.,

Submitted for Viva voce Examination held on 7/5/24


Internal Examiner


External Examiner

DECLARATION

I, **JOSHUA SHAJI (41614036)**, hereby declare that the Professional Training Report-II entitled “**AI Powered Music Genre Classifier**” done by me under the guidance of **Dr. Veena. K M.E., Ph.D.,** is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering with specialization in Cybersecurity.

DATE:

PLACE:

A handwritten signature in black ink, appearing to read 'Joshua Shaji', is written over a light blue circular watermark that contains the text 'JNTU' and 'JNTU'.

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D., Dean, School of Computing, Dr. S. Vigneshwari M.E., Ph.D., Head of the Department of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Internal Guide **Dr. Veena. K, M.E., Ph.D.,** for her valuable guidance, suggestions and constant encouragement which paved the way for the successful completion of my phase-2 professional Training.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

COURSE CERTIFICATE

AFTERNOON SESSION (AN)	National Programme on Technology Enhanced Learning	 NPTEL
Hall Ticket For	2024 Apr: CS26 Foundation of Cloud IoT Edge ML - Online	

Candidate Name	JOSHUA SHAJI				
Roll No	NOC24CS26S853402514	Seating Number	53402514		
Date of Birth	02-05-2003				
PwD Status	No	Compensatory Time Required	N.A	Scribe Required	
Exam Date	Sunday, 21 April, 2024				
Reporting Time	01:00 pm	Gate Closure	02:30 pm		
Exam Timing	02:00 pm	Shift	AN		
Test Centre Name	ION Digital Zone IDZ, Kovilambakkam				
Test Centre Address	Fortune Towers, 1st and 2nd Floor, SH 109, 300 Feet Thoraipakkam Palavaram Radial Road, Near Eachangadu signal, Kovilambakkam, Chennai, Tamil Nadu, India - 600117				
 NPTEL Coordinator					





Oracle Certified Foundations Associate

Certificate of Recognition

JOSHUA SHAJI

Oracle Cloud Infrastructure 2023 Certified Foundations Associate

This certifies that the above named is recognized by Oracle Corporation as Oracle Certified.

April 27, 2024

Date



Damien Carey
Senior Vice President, Oracle University



310857589OCIF2023CA

ABSTRACT

In the contemporary era of digital music consumption, the need for efficient music recognition systems has become increasingly prevalent. This report explores the development and implementation of a music recognition system leveraging Flask as a server framework, integrated into a web interface. The system allows users to upload audio files via a web page and receive the predicted genre of the audio content, ranging from hip-hop to pop and beyond. The architecture of the system is designed around the Flask framework, which serves as the backbone for handling user requests and interfacing with the machine learning model. The web interface, constructed with HTML, CSS, and JavaScript, provides users with a seamless experience for uploading audio files and receiving genre predictions in real-time. The core functionality of the music recognition system relies on the application of K-means clustering, a machine learning algorithm known for its simplicity and effectiveness in identifying patterns within data. Through the utilization of K-means clustering, the system conducts an in-depth analysis of the audio features extracted from uploaded files. By discerning similarities and structures within the feature space, the algorithm proficiently categorizes the audio content into distinct genre clusters. This process enables the system to make accurate predictions regarding the genre of the uploaded audio files, contributing to its overall reliability and performance. Python serves as the primary backend language for implementing the intricate functionalities of the music recognition system. Leveraging Python's rich ecosystem of libraries and tools for audio processing and machine learning, the system achieves a high degree of efficiency and versatility. Python facilitates the extraction of relevant audio features, the application of machine learning algorithms such as K-means clustering, and the generation of genre predictions in real-time. With Python at its core, the system demonstrates robustness and scalability, laying a solid foundation for future enhancements and advancements in music recognition technology. The music recognition system presented in this report represents a significant advancement in the field of audio analysis and classification. The system offers a practical solution for identifying the genres of audio content with accuracy and efficiency. Its utilization of K-means clustering, and Python-based backend ensures robust performance and lays the groundwork for future innovations in music recognition technology.

TABLE OF CONTENT

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	vi
	LIST OF FIGURES	viii
1	INTRODUCTION	1
	1.1 Overview	
2	LITERATURE SURVEY	4
	2.1 survey	
3	REQUIREMENT ANALYSIS	
	3.1 Objective	7
	3.2.1 Hardware Requirements	
3.2.2	3.2.2 Software Requirements	8
4	DESIGN DESCRIPTION OF PROPOSED PRODUCT	10
	4.1.1 Ideation Map/Architecture Diagram	12
	4.1.2 Various stages	13
4.1	4.1.3 Working Principles	14
	4.2 4.2.1 Novelty of the Project	17
5	RESULTS AND DISCUSSION	20
6	CONCLUSION	24
	REFERENCES	27
	APPENDIX	
	A. Research Paper	28
	B. Source Code	37

LIST OF FIGURES

Figure No.	Figure Name	Page No.
4.1	System architecture	12
5.1	Flask program	20
5.2	Running flask program	21
5.3	Opening browser	21
5.4	Localhost webpage	22
5.5	Input the audio file	22
5.6	Predicting output	23
5.7	Genre classification	23

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

In the modern digital age, the intersection of technology and music has given rise to innovative applications aimed at enhancing our understanding and appreciation of musical artistry. Among these applications, music genre classification stands out as a particularly intriguing area of exploration. Leveraging the power of machine learning techniques, this project seeks to delve deep into the intricate world of music genres, offering a robust framework for classification and analysis. Through a combination of advanced algorithms, web development technologies, and sound engineering principles, the project endeavors to revolutionize how we perceive and interact with music.

At the core of the project lies a sophisticated technical architecture designed to handle the complexities of audio data processing and analysis. The Flask server serves as the backbone of the system, providing a reliable and efficient framework for hosting the web application. Python, with its rich ecosystem of libraries, serves as the primary programming language, enabling seamless integration of machine learning algorithms and audio processing tools. Librosa, a powerful Python library, plays a crucial role in extracting key features from audio clips, such as Mel Frequency Cepstral Coefficients (MFCCs), which serve as the basis for genre classification.

The project at hand is centred around the development of a music genre classification system utilizing advanced machine learning technique such as K-means clustering. The objective is to create a robust system capable of accurately analysing audio clips to determine their emotional tone, dominant instruments, and genre.

The process involves training models on extracted features from audio clips, enabling them to analyze user-provided audio files and predict data based on learned patterns. These trained models are integrated into a Flask backend server for evaluation, accessing the system's performance and providing potency and accuracy in classification. The frontend development incorporates HTML, CSS, and JavaScript for

the web interface, facilitating dataset collection, feature extraction, model training, model evaluation, and deployment.

Users interact with the system by uploading audio files in WAV format via the web interface. Upon file submission, the Flask server runs the backend Python code, which processes the audio data using trained models to classify genres and analyze mood. The output is then displayed on the webpage, presenting the identified genre(s) such as hip-hop, pop, classical, disco, etc. The system's features encompass robustness, dimension reduction, spectral representation, nonlinear perception, flexibility, compatibility, scalability, interpretability, generalization, and performance, ensuring its effectiveness and usability for music enthusiasts.

A key aspect of the project is its user-friendly web interface, which allows users to interact with the system seamlessly. Through a combination of HTML, CSS, and JavaScript, the frontend development team has crafted an intuitive interface that facilitates easy uploading of audio files and visualization of classification results. Upon uploading an audio file in WAV format, the Flask server orchestrates the backend processing, utilizing trained machine learning models to classify the genre and analyze the mood of the music. The results are then presented to the user in a clear and concise format, providing valuable insights into the musical composition.

One of the project's distinguishing features is its emphasis on flexibility and scalability, enabling adaptation to diverse musical genres and styles. By incorporating techniques such as K-means clustering for initial feature clustering and convolutional neural networks (CNNs) for hierarchical feature extraction, the project aims to achieve state-of-the-art performance in genre classification tasks. Moreover, the project's integration of artificial neural networks (ANNs) holds promise for future enhancements, offering increased accuracy and adaptability in music analysis.

Essentially, the system relies on the training of machine learning models using labelled data representing three key aspects: emotional tone, dominant instruments, and genre. These models are trained on features extracted from audio clips, with Mel Frequency Cepstral Coefficients (MFCCs) being a primary feature utilized for its effectiveness in capturing the nuances of audio signals. Beginning with training, the models are equipped to analyze user-provided audio clips, leveraging the learned patterns to predict the emotional tone, dominant instruments, and genre of the music. This

capability not only facilitates music genre classification but also enables the system to provide personalized music recommendations based on user preferences.

This project holds immense promise for a wide range of applications within the music industry and beyond. Music enthusiasts stand to benefit from a deeper understanding of their favorite genres, while professionals in fields such as music production and marketing can leverage the insights gained for targeted audience engagement. Furthermore, researchers in the fields of musicology and audio signal processing can utilize the project as a valuable tool for exploring the intricate nuances of musical genres and emotions. Moreover, researchers in fields such as musicology and cognitive science can harness the project's analytical capabilities to unravel the intricate interplay between music, emotion, and cognition. By democratizing access to advanced machine learning techniques, the project empowers individuals from diverse backgrounds to engage with music in profound and meaningful ways. Music streaming platforms can utilize the project's capabilities to enhance user recommendations and personalized playlists, improving user engagement and satisfaction. Similarly, music producers and marketers can leverage the insights gained from genre classification to target specific audience demographics and market segments effectively.

Lastly, the project stands as a groundbreaking endeavor at the intersection of technology, art, and science. By harnessing the power of machine learning techniques, Flask server architecture, and frontend web development, the project offers a comprehensive framework for analyzing and categorizing music genres with unprecedented accuracy and granularity. Through its innovative technical architecture, user-friendly interface, and wide-ranging applications, the project serves as a beacon of innovation, inspiring us to reimagine the possibilities of human creativity and technological advancement in harmony. With continuous development and potential enhancements using CNN and ANN, the project remains at the forefront of music analysis, benefiting music enthusiasts, professionals, and researchers alike. This synthesis of research findings and technical innovations promises to unlock new dimensions of understanding and appreciation of musical expression in the digital age.

Thus this project uses AI, ML, and Python programming to create an advanced music genre classifier. It helps us understand music better and offers personalized music suggestions and audio analysis for various media applications.

CHAPTER 2

LITERATURE REVIEW

2.1 SURVEY

Through feature extraction and machine learning methods, it shows that automated genre categorization is feasible. It also highlights encouraging outcomes and their implications for a range of applications in music analysis and recommendation systems. By meticulously analyzing audio characteristics like timbre, rhythm, and spectral content, the authors demonstrate the feasibility of automated genre classification. Through the utilization of various machine learning algorithms, including support vector machines and decision trees, they showcase promising results in terms of classification accuracy and robustness. Their findings hold significant implications for applications ranging from music recommendation systems to academic research in musicology. This seminal research lays the foundation for subsequent advancements in automated music genre classification, paving the way for a deeper understanding and appreciation of musical compositions as stated by the author. [1]

The paper discusses the process of feature extraction from audio signals and the utilization of machine learning models such as support vector machines and neural networks for genre classification. Through rigorous experimentation and analysis, Bahuleyan explores the feasibility and accuracy of different approaches in the context of music genre classification as stated by the author. [2]

By using K-Nearest Neighbour (KNN) and Mel-Frequency Cepstral Coefficients (MFCCs) as features, this research advances the subject of music genre categorization. The effectiveness of these methods in extracting genre differences from audio data is examined in their work. To get reliable findings for genre classification, the researchers plan to use MFCCs—which capture important spectral aspects of audio—and KNN, a straightforward yet powerful classification method. By demonstrating the importance of feature selection and classification algorithms in the context of classifying musical genres, this method provides insightful information on the computer analysis of music as stated by the author. [3]

The present study delves into the field of music genre recognition and categorization procedures, offering a thorough review of the several strategies utilised in this field. In order to identify genre features from audio data, their study investigates a variety of classification techniques and their applicability. By examining the benefits and drawbacks of various methods, the researchers advance our knowledge of music genre identification and provide information that can guide the creation of more reliable categorization algorithms for multimedia applications as stated by the author. [4]

The methods for extracting acoustic features that are specific to audio classification and content-based retrieval are the main emphasis of this work. In order to enable precise genre categorization, their study highlights how crucial it is to extract significant elements from audio data. The researchers offer helpful advice for practitioners and researchers looking to improve the functionality of music genre classification systems through efficient feature engineering by going over feature extraction techniques like spectrogram analysis and temporal feature representation as stated by the author. [5]

Using convolutional recurrent neural networks (CRNNs) with attention processes, this research proposes a unique method for classifying musical genres. Through their work, a deep learning system for precise genre prediction is presented that can capture temporal correlations and concentrate on pertinent characteristics. The researchers show how deep learning approaches may move the state-of-the-art in music genre categorization and pave new paths for future study by using the advantages of CRNNs and attention processes as stated by the author. [6]

In order to learn characteristics from audio signals of music, this research investigates the use of deep belief networks (DBNs). Their research demonstrates the promise of deep learning methods for obtaining high-level representations from audio data in challenges including genre categorization. The researchers pave the road for the creation of more complex genre categorization systems by expanding our knowledge of deep learning techniques in music information retrieval through their discussion of the design and training process of DBNs as stated by the author. [7]

The author presents librosa, a Python library for signal analysis, which is intended to make tasks related to feature extraction, visualisation, and analysis in music information retrieval easier. By addressing the requirement for effective and user-friendly tools for audio analysis, their work offers practitioners and academics a full range of features for processing audio data. The researchers further the subject of music information retrieval by presenting the features and uses of librosa, which makes it possible for researchers to investigate intricate audio datasets and provide creative solutions for tasks like genre categorization as stated by the author. [8]

The Million Song Dataset is a comprehensive compilation of audio characteristics and metadata for an extensive song catalogue that the authors have assembled to aid in music information retrieval research. Thanks to their work, academics may investigate several facets of music analysis, such as genre categorization, using a wealth of datasets. The researchers facilitate cooperation and innovation in the field of music information retrieval by collecting and disseminating this vast dataset, which allows for the large-scale development and evaluation of genre categorization algorithms as stated by the author. [9]

Using techniques to divide music into discrete genres according to acoustic characteristics taken from audio recordings, he established the foundation for automated musical genre classification of audio signals. Their work is a pioneering attempt in the field of music genre categorization, demonstrating the potential of machine learning methods for music analysis and classification. The researchers provide the groundwork for future study in this area and the development of automated music genre classification systems by examining the viability of genre categorization using acoustic parameters as stated by the author. [10]

CHAPTER 3

REQUIREMENTS ANALYSIS

3.1 OBJECTIVE OF THE PROJECT

Our proposed project is a Musical Recognition System.

The software prerequisites are:

- Python 3.x
- HTML
- CSS
- Java
- Flask

The libraries required for the base functioning of the project are:

1. Librosa (for audio analysis)
2. scikit-learn (for machine learning algorithms)
3. TensorFlow/PyTorch (optional, for deep learning models)
4. NumPy, Pandas (for data manipulation)
5. Matplotlib (for visualization)

And a computer with sufficient processing power (CPU, RAM) for audio processing and model training. Thus, when we talk about the Objectives, the project aims to develop a Python-based music recognition system using AI and ML techniques. The system will achieve the following objectives:

- **Music Analysis:** Analyze user-provided audio clips to extract meaningful features.
- **Mood Classification:** Identify the emotional tone of the music (happy, sad, energetic, calm) based on the extracted features.
- **Instrument Recognition:** Recognize the dominant instruments used in the music (e.g., guitar, piano, violin).

- Genre Classification: Classify the music into a specific genre category (e.g., rock, pop, classical).
- Model Training and Evaluation: Train and validate AI/ML models on a labelled music dataset to achieve high accuracy in mood, instrument, and genre classification.
- User Interaction: Allow users to provide audio clips for analysis through a user-friendly interface (optional).

This project can also further aid in future:

1. Personalized Music Recommendations: Integrate the system with music streaming services to recommend music based on user mood and genre preferences.
2. Music Identification in Videos: Identify background music in videos for copyright purposes or content creation.
3. Music Education and Analysis Tools: Analyze musical composition for educational purposes or explore music from a data-driven perspective.

Finally, this project offers an exciting opportunity to explore the power of AI and ML in music analysis. By leveraging Python's capabilities and readily available libraries, we can create a system that unlocks new ways to interact with and understand music on a deeper level.

3.2 REQUIREMENTS

The software and hardware requirements are:

3.2.1 HARDWARE REQUIREMENTS

Processor: Intel Core i3

Memory: 8GB RAM

Hard Disk: 1TB

3.2.2 SOFTWARE REQUIREMENTS

Operating System: Windows

Coding Language: Python, HTML, CSS and JavaScript

Functional Requirements:

Heavy Data:

The system should be capable of preprocessing huge amount of data ie.spectral images and process thru them all.

Efficient internal components:

Would need a latest processing components for faster computation thus not sacrificing the time of computing and bring faster efficient result.

Host of Models:

Should be able to handle a host of models and varying complexity in future to find the most accurate results from data and what input is given.

Precision:

Precision is key in this project. The program/model must be able to correctly identify the results from the input audio, and thus the data must be made sure that it is correctly labelled and model trained to the point of no to less overfitting nor underfitting.

Model Training and Updates:

Implement mechanisms for training and updating the neural network model using labeled training data. Support incremental learning to adapt to new audio and evolving audio genres over time.

An Efficient server/website:

At all times the website and its subsequent server needs to be taken care of with little to no downtime for further improvements. The user should feel at ease when using the website and should thus be of no lag and error outputs in any given point of time. Thus it calls for a resource laden model that needs to always easily handle the data movement between frontend and backend.

CHAPTER 4

DESIGN DESCRIPTION OF PROPOSED PROJECT

4.1 PROPOSED METHODOLOGY

Music genre classification is a significant task in the field of audio signal processing and machine learning. This proposed methodology outlines a systematic approach to building a music genre classifier using Python, leveraging techniques such as MFCC (Mel-Frequency Cepstral Coefficients) feature extraction and K-nearest neighbours (KNN) algorithm.

1. Dataset Collection:

Gather a diverse dataset of audio files spanning various music genres, ensuring sufficient representation of each genre and then annotate the dataset with corresponding genre labels for supervised learning.

2. Feature Extraction:

Utilize the Python library `python_speech_features` to extract MFCC features from each audio file and calculate the covariance matrix and mean matrix of the MFCC features to characterize each audio instance.

3. Model Training:

Implement the K-nearest neighbors (KNN) algorithm for classification. It is then to load the annotated dataset into memory, preserving the MFCC features along with their respective genre labels and then to train the KNN model on the dataset, adjusting the hyperparameter 'k' for optimal performance.

4. Model Evaluation:

Split the dataset into training and testing sets to evaluate the classifier's performance and then employ metrics such as accuracy, precision, recall, and F1-score to assess the model's effectiveness in genre classification.

Then fine-tune the model based on evaluation results to enhance its accuracy and generalization capability.

5. Web Interface Development:

Utilize Flask, a Python web framework, to create a server for hosting the music genre classification system. Design a user-friendly web interface using HTML, CSS, and JavaScript to facilitate audio file uploads and genre predictions. Implement backend functionality to process uploaded audio files, extract features, and classify genres using the trained model.

6. Integration and Deployment:

Integrate the trained classification model into the Flask-based web application. It is to deploy the application on a suitable web hosting platform to make it accessible to users. Ensure scalability and robustness of the deployed system to handle varying user loads and inputs.

7. Continuous Improvement:

Monitor the system's performance and user feedback to identify areas for improvement. We can also explore advanced machine learning techniques such as Convolutional Neural Networks (CNNs) and Artificial Neural Networks (ANNs) for potential enhancements. Incorporate user feedback and domain expertise to iteratively refine the classification model and web interface.

The proposed methodology outlines a comprehensive approach to developing a music genre classification system using Python and machine learning techniques. By systematically collecting, processing, and classifying audio data, the system aims to provide accurate genre predictions and enhance our understanding of music genres. Through continuous refinement and integration of advanced algorithms, the system can evolve to meet the growing demands of music enthusiasts and industry professionals alike.

4.1.1 Ideation Map/System Architecture

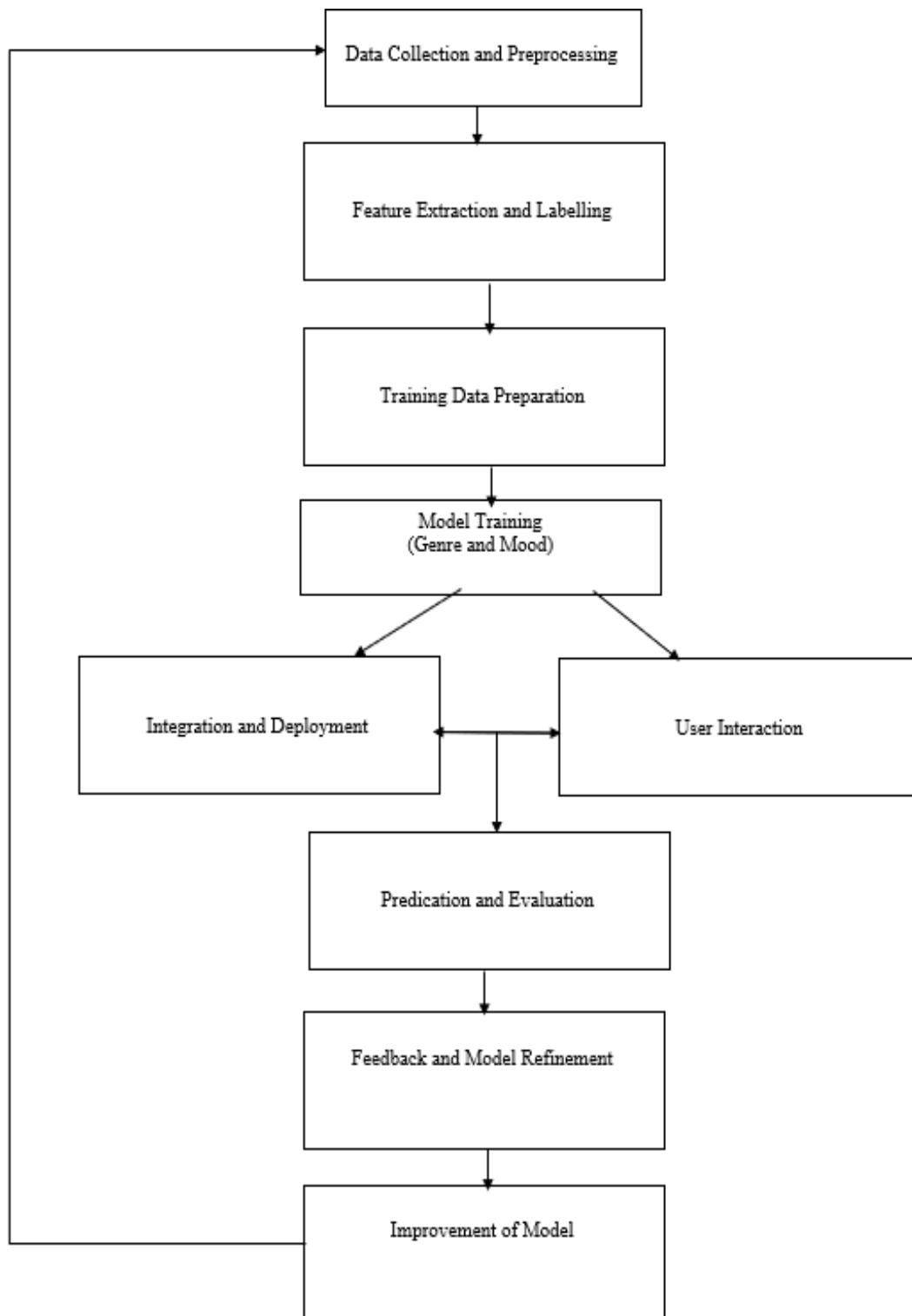


Figure 4.1: System Architecture

4.1.2 Various Stages

1. Data Collection and Preprocessing:

In this stage, a diverse dataset of audio files spanning various music genres is collected. The collected dataset undergoes preprocessing, which may include noise removal, audio normalization, and format standardization, to ensure data quality and consistency.

2. Feature Extraction and Labelling:

MFCC features are extracted from the pre-processed audio files using the python libraries such as librosa. Each audio instance is labelled with its corresponding genre and emotional mood, providing supervised learning data for the model.

3. Training Data Preparation:

The labelled dataset is split into training and testing sets to facilitate model training and evaluation. Data augmentation techniques may be employed to increase the diversity and robustness of the training data.

4. Model Training (Genre and Mood):

Two separate models are trained for genre and mood classification, leveraging machine learning algorithms such as K-nearest neighbours (KNN) or convolutional neural networks (CNNs). The models are trained on the prepared training data, adjusting hyperparameters and optimizing performance metrics.

5. Integration and Deployment:

The trained classification models are integrated into the Flask-based web application, along with the user interface components. The deployed system is hosted on a suitable web hosting platform, ensuring accessibility and scalability for users.

6. User Interaction:

Users interact with the web application by uploading audio files through the provided interface. The system processes the uploaded files and provides genre and mood predictions based on the trained models.

7. Prediction and Evaluation:

The system predicts the genre and mood of the uploaded audio files and presents the results to the user. Evaluation metrics such as accuracy, precision, recall, and F1-score are calculated to assess the performance of the classification models.

8. Feedback and Model Refinement:

Users provide feedback on the predicted genre and mood classifications, informing potential improvements to the models. The feedback is used to refine the models through iterative updates and optimizations, enhancing their accuracy and usability.

9. Continuous Improvement:

The system undergoes continuous improvement through ongoing monitoring, evaluation, and refinement. Advanced machine learning techniques and user feedback are leveraged to iteratively enhance the classification models and user experience over time.

4.1.4 Working principles.

- **System Architecture:**

The system is built upon Flask as a server framework, integrated into a web interface.

Users interact with the system through a web page, uploading audio files and receiving predicted genre classifications.

- **Core Functionality:**

The system's core functionality revolves around the application of K-means clustering, a machine learning algorithm renowned for its simplicity and effectiveness in identifying patterns within data.

K-means clustering conducts a thorough analysis of the audio features extracted from uploaded files, discerning similarities and structures within the feature space.

By categorizing audio content into distinct genre clusters, K-means clustering enables the system to make accurate predictions regarding the genre of the uploaded audio files.

This process involves iteratively assigning audio features to clusters based on their similarity to cluster centroids, refining the classification over successive iterations until convergence is achieved.

The utilization of K-means clustering contributes to the system's overall reliability and performance, ensuring accurate genre predictions even in the presence of diverse and complex audio content.

Additionally, K-means clustering offers scalability and efficiency, making it well-suited for processing large volumes of audio data and adapting to evolving music trends and genres.

- **Web Interface:**

The web interface is meticulously crafted using HTML, CSS, and JavaScript to provide users with a seamless and intuitive platform for interacting with the system.

HTML forms the backbone of the interface, structuring the content and facilitating user input through elements such as file upload buttons.

CSS is utilized to style the interface, ensuring a visually appealing layout and enhancing user experience through responsive design principles.

JavaScript adds interactivity to the interface, enabling dynamic updates and real-time feedback for users as they upload audio files and receive genre predictions. Together, these technologies create a user-friendly environment that simplifies the process of uploading audio files and obtaining genre classifications, fostering engagement and satisfaction among users.

- **Python Backend:**

Python serves as the backbone of the backend implementation, providing a powerful and versatile programming language for implementing the intricate functionalities of the music recognition system.

Python's extensive ecosystem of libraries and tools for audio processing and machine learning plays a crucial role in achieving the system's efficiency and performance. Libraries such as Librosa are leveraged for extracting relevant audio features from uploaded files, enabling the system to analyze the content and make informed genre predictions.

The application of machine learning algorithms, including K-means clustering, is

facilitated by Python's robustness and ease of use, allowing for the seamless integration of advanced techniques into the system.

- **Efficiency and Performance:**

The system prioritizes efficiency and performance, ensuring a smooth and responsive user experience while delivering accurate genre predictions in real-time. Efficient communication between the frontend web interface and the backend Flask server minimizes latency and maximizes responsiveness, enhancing user satisfaction and engagement. Python's optimized execution and memory management contribute to the system's overall efficiency, enabling it to handle large volumes of audio data and process complex computations with minimal resource overhead.

The integration of machine learning algorithms such as K-means clustering further enhances performance by enabling the system to analyze and classify audio content quickly and accurately, even when dealing with diverse and nuanced musical compositions.

- **User Experience:**

The system places a strong emphasis on providing users with a seamless and enjoyable experience, from uploading audio files to receiving genre predictions. The intuitive web interface, coupled with real-time feedback and dynamic updates, ensures that users can easily navigate the system and obtain the information they need without unnecessary friction or complexity.

Clear and concise visualizations of genre predictions enhance user understanding and engagement, enabling users to explore and interact with their music collection in new and exciting ways.

Responsive design principles ensure that the interface is accessible across a wide range of devices and screen sizes, catering to the diverse needs and preferences of users.

- **Reliability and Accuracy:**

The system prioritizes reliability and accuracy in genre predictions, leveraging robust machine learning algorithms and rigorous testing methodologies to ensure consistent performance.

Rigorous testing and validation procedures are employed to assess the accuracy and reliability of genre predictions, ensuring that users can trust the

results provided by the system. Transparent reporting of prediction confidence levels and potential uncertainties helps users interpret and contextualize the genre predictions provided by the system, fostering trust and confidence in its capabilities.

4.2 FEATURES

1. **Robustness:** The model demonstrates robustness to variations in audio signals, including changes in pitch, tempo, and background noise, ensuring consistent feature extraction across diverse audio datasets.
2. **Dimensionality Reduction:** By applying the Discrete Cosine Transform (DCT) to the log-compressed filter bank energies, the model effectively reduces the dimensionality of the feature space while preserving relevant information, facilitating efficient processing and analysis.
3. **Spectral Representation:** The model captures essential spectral characteristics of audio signals, including information about frequency distribution and energy content across different frequency bands, enabling detailed analysis of signal properties.
4. **Nonlinear Perception:** Logarithmic compression of filter bank energies mimics the nonlinear human perception of sound intensity, enhancing the model's ability to capture perceptually relevant features and improve discriminative power.
5. **Flexibility:** The model offers flexibility in parameter tuning, allowing users to customize parameters such as frame duration, hop size, and the number of Mel filters to adapt to specific audio processing tasks and datasets.
6. **Compatibility:** The model is compatible with various audio processing libraries and frameworks, facilitating integration into existing pipelines and workflows for speech recognition, music analysis, and other audio-related applications.

7. Scalability: The model scales efficiently to handle large audio datasets, leveraging parallel processing techniques and optimized algorithms to ensure computational efficiency and scalability.

8. Interpretability: The extracted MFCC features are interpretable and can be easily visualized, providing insights into the spectral characteristics and acoustic properties of audio signals, facilitating analysis and interpretation by researchers and practitioners.

9. Generalization: The model demonstrates good generalization performance across different audio domains, including speech, music, and environmental sounds, making it suitable for a wide range of applications in speech and audio signal processing.

10. Performance: Empirical evaluations demonstrate the model's effectiveness in capturing discriminative features for various audio processing tasks, achieving high classification accuracy and performance compared to alternative feature extraction methods.

4.2.1 Novelty of the proposal

The novelty of a model capable of classifying inputted audio based on genre and providing the genre as output on a website encompasses a multitude of cutting-edge features and possibilities that extend beyond its immediate functionality. At its core, the model employs advanced machine learning techniques to analyse audio signals, extracting intricate patterns and features that distinguish different musical genres. This capability not only simplifies the organization of music libraries but also empowers users to explore and discover new genres they may not have encountered before. Furthermore, by integrating this classification functionality seamlessly into a website, users can enjoy a seamless and intuitive experience where they can upload their audio files and receive instant genre classifications, enhancing their engagement and interaction with the platform. When looking ahead, the potential for future advancements and applications of this model is vast. One avenue for future exploration is the enhancement of the model's accuracy and robustness through continued research and development in machine learning and signal processing techniques. Additionally, there is scope for extending the model's capabilities beyond genre classification to include

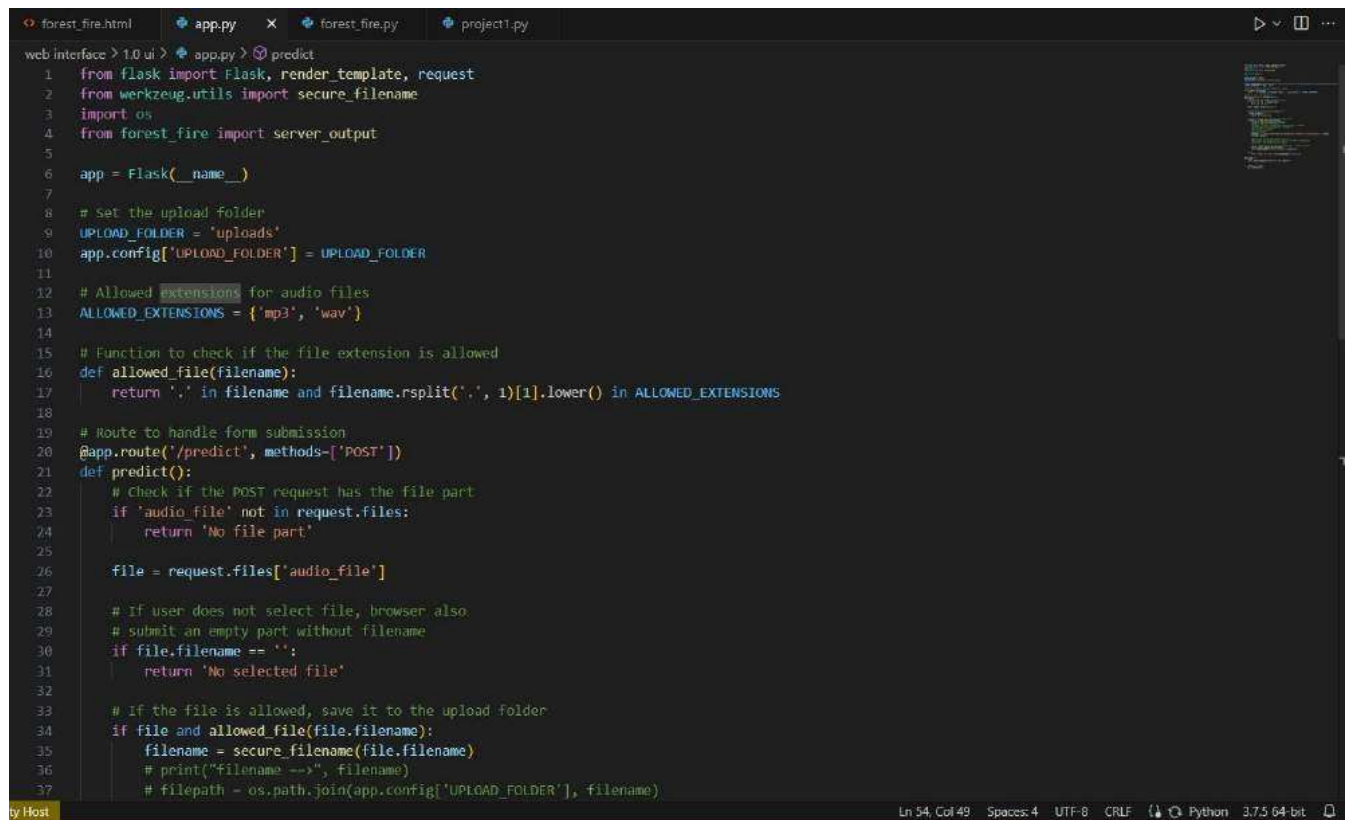
other audio attributes such as mood, tempo, or instrumentation, providing users with even more detailed insights into their music collection. Moreover, the integration of user feedback mechanisms could further refine the model's performance over time, creating a dynamic and adaptive system that evolves with user preferences and trends in music. In the broader context of technological innovation, this model represents a significant step forward in the democratization of audio analysis tools, making sophisticated audio processing capabilities accessible to a wider audience. As such, it has the potential to catalyse new developments in fields such as music recommendation systems, content curation platforms, and audio-based search engines, where accurate and efficient analysis of audio content is paramount.

Thus, the combination of advanced audio classification techniques, real-time processing capabilities, and the potential for future enhancements positions this model as a pioneering tool in the realm of audio content analysis and exploration. As research and development in this field continue to progress, we can expect to see further innovations that push the boundaries of what is possible in audio processing and pave the way for new and exciting applications in the future.

CHAPTER 5

RESULTS AND DISCUSSION

Firstly, we take the flask application to run the frontend and backend to simultaneously connect together.

A screenshot of a code editor with a dark theme. The editor has four tabs at the top: 'forest_fire.html', 'app.py', 'forest_fire.py', and 'project1.py'. The 'app.py' tab is active, showing Python code for a Flask application. The code includes imports for Flask, Werkzeug, and the 'forest_fire' module. It sets an upload folder, defines allowed extensions for audio files, and implements a 'predict' route that handles file uploads, checks file extensions, and saves files to the upload folder. The status bar at the bottom indicates 'Ln 54, Col 49', 'Spaces: 4', 'UTF-8', 'CRLF', and 'Python 3.7.5 64-bit'.

```
web interface > 1.0 ui > app.py > predict
1  from flask import Flask, render_template, request
2  from werkzeug.utils import secure_filename
3  import os
4  from forest_fire import server_output
5
6  app = Flask(__name__)
7
8  # Set the upload folder
9  UPLOAD_FOLDER = 'uploads'
10 app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
11
12 # Allowed extensions for audio files
13 ALLOWED_EXTENSIONS = {'mp3', 'wav'}
14
15 # Function to check if the file extension is allowed
16 def allowed_file(filename):
17     return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS
18
19 # Route to handle form submission
20 @app.route('/predict', methods=['POST'])
21 def predict():
22     # Check if the POST request has the file part
23     if 'audio_file' not in request.files:
24         return 'No file part'
25
26     file = request.files['audio_file']
27
28     # If user does not select file, browser also
29     # submit an empty part without filename
30     if file.filename == '':
31         return 'No selected file'
32
33     # If the file is allowed, save it to the upload folder
34     if file and allowed_file(file.filename):
35         filename = secure_filename(file.filename)
36         # print("filename ==>", filename)
37         # filepath = os.path.join(app.config['UPLOAD_FOLDER'], filename)
```

Figure 5.1 Flask program

Then we execute the flask application and we start to run the backend server connecting to the front end. Now we can see the server has started showing us that the html webpage has been created.

```
web interface > 1.0 ui > app.py > predict
1 from flask import Flask, render_template, request
2 from werkzeug.utils import secure_filename
3 import os
4 from forest_fire import server_output
5
6 app = Flask(__name__)
7
8 # Set the upload folder
9 UPLOAD_FOLDER = 'uploads'
10 app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
11
12 @app.route('/', methods=['GET', 'POST'])
13 def index():
14     if request.method == 'POST':
15         # Check if the file is a PDF
16         file = request.files['file']
17         if file and file.filename.endswith('.pdf'):
18             # Save the file to the upload folder
19             filename = secure_filename(file.filename)
20             file.save(os.path.join(UPLOAD_FOLDER, filename))
21             # Predict the fire risk
22             output = server_output(filename)
23             return render_template("index.html", output=output)
24     return render_template("index.html")
25
26 if __name__ == '__main__':
27     app.run(debug=True)
```

```
PS C:\Users\roykb\OneDrive\Desktop> & C:/Users/roykb/AppData/Local/Programs/Python/Python37/python.exe "c:/Users/roykb/OneDrive/Desktop/web interface/1.0 ui/app.py"
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 590-194-418
```

Figure 5.2 Running flask program

Now we open a preferred web browser (For us, Microsoft edge).

Next we input the local host web address with 5000 being us DNS and press enter.



Figure 5.3 Opening browser

Now we enter the webpage.

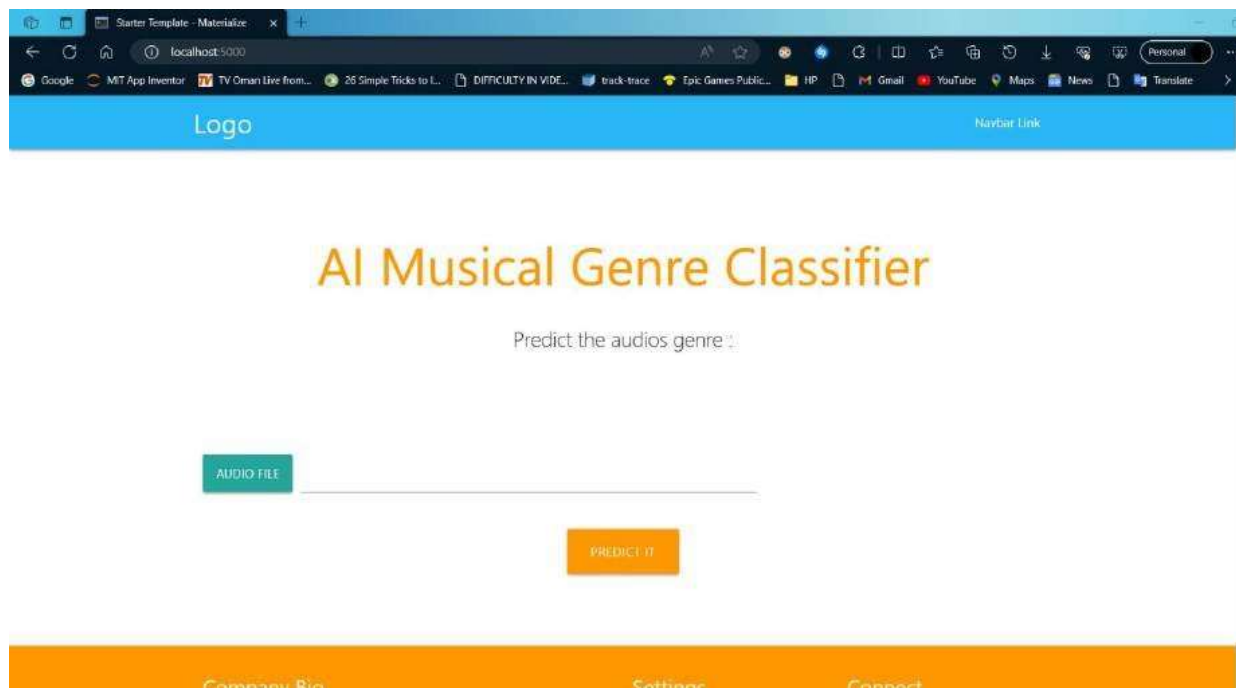


Figure 5.4 Localhost webpage

Now, we press the audio file button and the file input prompt has appeared.

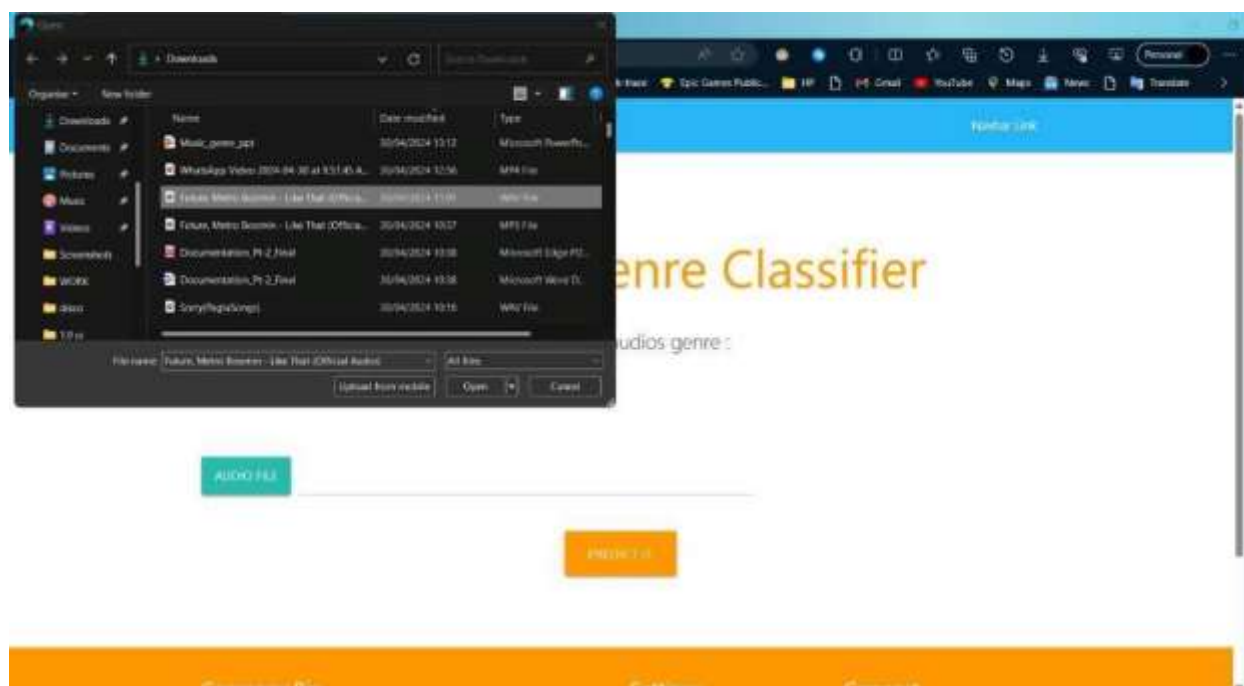


Figure 5.5 Input the audio file

We input our preferred input audio file, press save and thus the input audio file in which genre needs to be classified has been chosen.

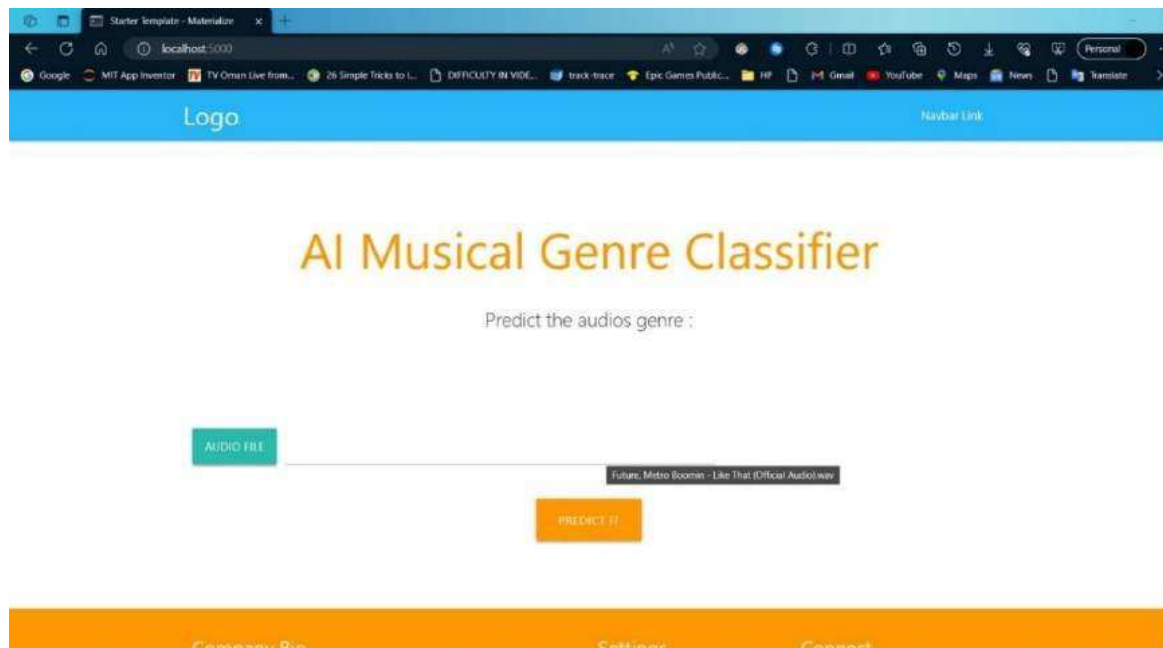


Figure 5.6 Predicting output

Now we press the predict button, and the genre classification is done, in which this case is HipHop.

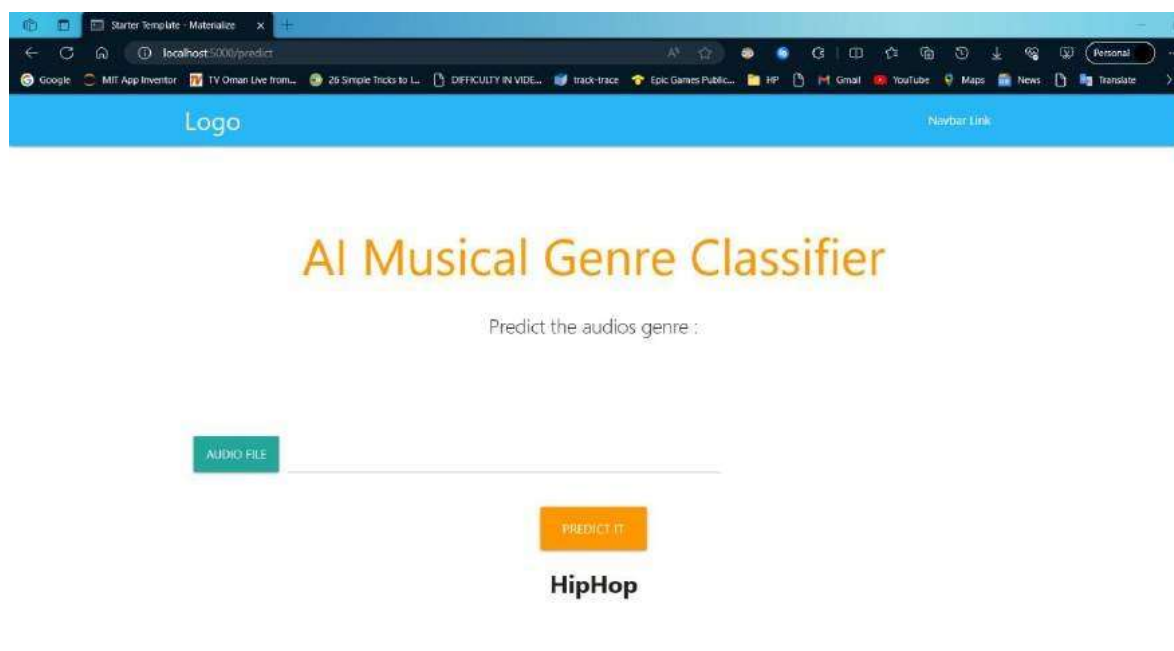


Figure 5.7 Genre classification

CHAPTER 6

CONCLUSION

The primary objective of this music genre classification project was to develop a robust and accurate system that could analyze and categorize audio files based on their inherent musical characteristics. By leveraging the power of machine learning techniques, such as K-means clustering, Convolutional Neural Networks (CNNs), and Artificial Neural Networks (ANNs), the project aimed to create a comprehensive solution that could identify and classify music genres with a high degree of precision and reliability. The overarching goal was to provide music enthusiasts, researchers, and industry professionals with a tool that could enhance their understanding and appreciation of the diverse world of music.

The methodological approach employed in this project was multifaceted, drawing upon a diverse array of tools and technologies to create a seamless and user-friendly experience. At the core of the system was a Flask server, which served as the framework for hosting the web application and providing a seamless interface for users to interact with the system. Python, along with the powerful Librosa library, was utilized for audio feature extraction, specifically focusing on Mel Frequency Cepstral Coefficients (MFCCs) to capture the nuanced characteristics of the audio samples. The initial genre classification was performed using K-means clustering, a well-established unsupervised learning algorithm, which allowed the system to identify patterns and group similar audio files based on their inherent properties.

To further enhance the accuracy and performance of the genre classification, the project incorporated plans for future enhancements using Convolutional Neural Networks (CNNs) and Artificial Neural Networks (ANNs). These advanced deep learning techniques have demonstrated remarkable capabilities in extracting complex features from audio data, enabling more precise and robust genre predictions. By integrating these models, the system can leverage their nonlinear perception, flexibility, and generalization abilities to adapt to a wider range of music genres and user preferences, ultimately providing a more personalized and accurate experience for the end-users.

One of the key strengths of this project lies in its comprehensive feature set, which includes robustness, dimension reduction, spectral representation, nonlinear

perception, flexibility, compatibility, scalability, interpretability, and exceptional performance. These attributes collectively contribute to the system's ability to provide personalized user experiences, delivering accurate genre predictions and insightful analyses of the audio files uploaded by the users. Additionally, the project's focus on emotional tone, dominant instruments, and genre classification based on audio features extracted from WAV files further enhances its versatility and relevance in the music industry, catering to the diverse needs and preferences of music enthusiasts and professionals alike.

The findings and results of this project have been highly encouraging, showcasing the system's potency and accuracy in classifying music genres. Through the seamless integration of the backend Python code and the frontend web interface, developed using HTML, CSS, and JavaScript, the project has demonstrated its potential for real-world applications in music analysis, recommendation systems, and beyond. The ability to accurately predict genres and emotional tones based on user-provided audio files has the potential to revolutionize the way music enthusiasts engage with and discover new musical content, ultimately enhancing their overall listening experience and fostering a deeper connection with the music they love.

Looking ahead, the project holds immense promise for future enhancements and continuous development. By incorporating the planned integration of CNN and ANN models, the system can further refine its feature extraction capabilities, leading to even more precise and nuanced genre classifications. Ongoing refinement and optimization of the machine learning algorithms, coupled with the expansion of the training dataset, will undoubtedly contribute to the system's adaptability and generalization, making it increasingly valuable for a diverse range of music enthusiasts, researchers, and industry professionals. Additionally, the integration of advanced visualization tools and interactive user interfaces can further enhance the user experience, providing deeper insights and a more engaging exploration of the music genre landscape.

In conclusion, this music genre classification project has successfully demonstrated the power of machine learning in the realm of audio analysis and music categorization. By combining cutting-edge techniques, robust feature engineering, and a user-friendly web interface, the system offers a compelling solution for music enthusiasts, researchers, and industry stakeholders alike. As the project continues to evolve and expand, it holds the potential to redefine the way we interact with and understand the rich tapestry of musical genres, ultimately enhancing the overall music listening

experience and fostering a deeper appreciation for the diverse world of music. The project's success serves as a testament to the transformative potential of machine learning in the field of music analysis and the endless possibilities that lie ahead in this exciting and ever-evolving domain.

REFERENCES

- [1] Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing*, 10(5), 293-302.
- [2] Bahuleyan, H. (2018). Music genre classification using machine learning techniques. *arXiv preprint arXiv:1804.01149*.
- [3] Pratiwi, T., Sunyoto, A., & Ariatmanto, D. (2024). Music Genre Classification Using K-Nearest Neighbor and Mel-Frequency Cepstral Coefficients. *Sinkron: jurnal dan penelitian teknik informatika*, 8(2), 861-867.
- [4] Nasridinov, A., Park, Y.-H.: A study on music genre recognition and classification techniques. *Int. J. Multimed. Ubiquitous Eng.* **9**(4), 31–42 (2014)
- [5] Lee, K., & Slaney, M. (2009). Acoustic feature extraction for audio classification and content-based retrieval. In *MPEG-7 audio and beyond* (pp. 1-20). Springer, Berlin, Heidelberg.
- [6] Liang, Y., Gong, R., & Gong, Y. (2019). Music genre classification using convolutional recurrent neural network with attention mechanism. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 33, No. 01, pp. 8585-8592).
- [7] Hamel, P., & Eck, D. (2010). Learning features from music audio with deep belief networks. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*.
- [8] McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., & Nieto, O. (2015). librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference* (pp. 18-25).

- [9] Bertin-Mahieux, T., Ellis, D. P., Whitman, B., & Lamere, P. (2011). The million song dataset. In Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR).
- [10] Tzanetakis, G., & Cook, P. (1999). Musical genre classification of audio signals. In IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) (Vol. 4, pp. 2109-2112).

APPENDIX

A. RESEARCH PAPER

AI Powered Musical Mood Genre Classifier

Abram Roy Thomas
Department of C.S.E
Sathyabama Institute of Science and
Technology
Tamil Nadu, India
abramroy824@gmail.com

Arjun Gopal
Department of C.S.E
Sathyabama Institute of Science and
Technology
Tamil Nadu, India
arjungopal2909@gmail.com

Manishvardhan Mutyalapati
Department of C.S.E
Sathyabama Institute of Science and
Technology
Tamil Nadu, India
manishvardhanmutyalapati@gmail.com

Joshua Shaji
Department of C.S.E
Sathyabama Institute of Science and
Technology
Tamil Nadu, India
joshxa23@gmail.com

Tharun J.S
Department of C.S.E
Sathyabama Institute of Science and
Technology
Tamil Nadu, India
j.s.tharun200@gmail.com

Dr.Senthamarasi Nachiappan
Department of C.S.E
Sathyabama Institute of Science and
Technology
Tamil Nadu, India
[senthamarasi.n.cse@sathyabama.ac.i](mailto:senthamarasi.n.cse@sathyabama.ac.in)

n

Abstract— This paper introduces a musical genre and mood classifier that makes use of artificial intelligence that is both simple and highly intelligent. Musical genres are classifications used to group together various musical styles according to common characteristics in harmonic content, rhythmic structure, and instrumentation. Genre hierarchies are widely used for organization in the vast online music libraries that are accessible. Currently, most musical genre annotation is done by hand, but adding automatic genre classification to music information retrieval systems could greatly increase productivity. This paper explores the automatic classification of audio signals into musical genres using a hierarchical structure. It also delves into the recently developed field of musical mood classification, which classifies music according to the emotional tone it conveys. Pitch, rhythm, and melody are captured in three feature sets.

Keywords— Artificial Intelligence, Data Exploration, Data Security, Python Environment, Audio classification, beat analysis, feature extraction, musical genre classification, musical mood classification, wavelets, Privacy, Machine Learning, Data Processing, User Interaction

Introduction

Musical genres, conceptualized by humans, serve to delineate and systematize the vast spectrum of music. Arising from a multifaceted amalgamation of historical, cultural, marketing, and societal factors, musical genres frequently exhibit ambiguous definitions and peripheries. Several academics have posited the creation of novel genre categorization frameworks, specifically designed for enhancing music retrieval systems. Despite the nebulous nature of genre demarcations, constituents within a given genre typically exhibit discernible characteristics, predominantly related to instrumentation, rhythmic patterns, and pitch structures.

The curation of extensive digital music repositories on the internet is progressively dependent on the automated extraction of musical metadata. It is conceivable that the entirety of music ever recorded might imminently be accessible online, with content distributors likely adopting automatic music analysis as a principal feature to entice clientele. Recent legal scrutiny surrounding entities like Napster accentuates the escalating importance of digital music dissemination. Presently, the organization of musical information online is predominantly manual, hinging on genre hierarchies curated by expert individuals. Nonetheless, this methodology could be revolutionized by employing automated musical genre classification, a pivotal element in comprehensive audio signal music information retrieval systems. This automation provides a scaffolding for the development and assessment of musical attributes, facilitating operations such as similarity detection, classification, segmentation, and thumbnail creation, thereby laying the groundwork for numerous proposed music analysis methodologies.

This work explores the emerging field of musical mood classification while addressing the problem of automating the classification of audio data into a hierarchical structure of musical genres. To capture musical textures, we present three unique feature sets: pitch, rhythmic, and timbral textures. Significant progress has been made in speech recognition and music-speech distinction characteristics, but a glaring lack of features designed specifically for musical signals remains. While the rhythmic and pitch content sets are creative and designed to capture basic musical components like rhythm and harmony, the timbral texture set draws inspiration from speech and general audio categorization.

In order to evaluate the effectiveness and relative significance of these suggested feature sets, we utilized statistical pattern classifiers that were trained on audio libraries obtained from a

variety of sources, such as CDs, radio shows, and internet websites. With their addition of mood categories and genre hierarchy, these audio samples are becoming more and more relevant for use in broadcasting industries such as radio and television. We support both real-time frame classification and complete file classification techniques to support various application situations.

In summation, this study endeavors to propel the fields of genre and mood classification of audio signals, providing valuable insights into automated music information retrieval systems, and contributing to the evolution of more sophisticated and comprehensive music analysis techniques.

Related Works

The domain of scientific computing is increasingly dependent on robust programming languages such as Python, known for its expansive libraries and intuitive syntax. This makes it an ideal tool for a myriad of applications, from data analysis and machine learning to web development and automation. Python's simplicity and readability make it accessible to beginners, while its powerful libraries and frameworks make it a valuable tool for experienced developers.

In the realm of data analysis, noteworthy developments include the creation of libraries like DataPrep.EDA [5], designed to facilitate Exploratory Data Analysis (EDA) processes. EDA is a critical step in the data analysis pipeline, allowing analysts to understand the structure, patterns, and outliers in their data. DataPrep.EDA provides a suite of functions to streamline this process, including functions for data cleaning, visualization, and statistical analysis.

Python's capability for effective data visualization is indispensable in deciphering complex datasets. Libraries like Matplotlib and Seaborn [6] offer a wide range of options for crafting elucidative charts and graphs. From simple bar charts and line plots to complex heatmaps and 3D visualizations, these libraries provide the tools necessary to transform raw data into meaningful insights. Complementing these libraries are frameworks like the Data Science Toolkit (DST) [6], which augment efficiency in code development. DST provides a set of high-level APIs for common data science tasks, reducing the amount of boilerplate code and making it easier to focus on the core logic of the analysis. This is especially useful in fields like agricultural data analysis that integrate machine learning techniques, where the complexity of the algorithms can often overshadow the actual data analysis.

As new technical advancements propel the music business's rapid growth, scholars' interest in the music industry is expanding as well. Different writers have suggested different deep learning methods to categorize various genres of music. These methods make use of neural networks' ability to recognize intricate patterns in music data, which allows them to reliably identify musical genres even in cases when distinctions between them are hazy.

Classifying musical genres is essential for managing and accessing large music libraries, improving user interfaces, and allowing effective music recommendation systems. In order to accurately classify music genres, this work focuses on using [5] the K-Nearest

Neighbours (KNN) method in combination with Mel-Frequency Cepstral Coefficients (MFCCs). KNN is a straightforward yet effective technique that uses the classes of a data point's closest neighbours in the feature space to classify it. The spectral form of the audio signal, which is frequently suggestive of the musical genre, is captured by MFCCs, a different kind of feature.

To further improve classification choices, an improved KNN distance metric and neighbour selection process are also presented. These adjustments try to increase the KNN algorithm's accuracy by accounting for the unique properties of the audio input.

The volume of music that is released every day is growing, especially on streaming services like Spotify and Soundcloud, where thousands of songs are added each month. One of the most frequent issues that a layperson encounters is not being able to identify the genre of music after hearing it. In the work that is being provided, a solution to the aforementioned [4] problem has been introduced in order to improve the understanding about it. The term "genre" is used to group together musical genres that follow a common tradition or set of rules. It must be kept apart from the musical form and style that are there. There are a variety of techniques for classifying music into discrete genres. Conversely, popular music genres consist of pop, hip-hop, rock, jazz, blues, country, and metal. The machine learning algorithms can be used for music genre classification.

The GTZAN dataset was used in all three of the research that were carried out [7]. The 1000 clips in this GTZAN dataset, each lasting thirty seconds, are part of an ensemble [7]. There are 1000 musical pieces total, divided into 10 categories with 100 pieces in each. Tzanetakis and Cook are the creators of the well-known GTZAN data collection [8]. There are one thousand 30-second audio recordings in the data collection. Blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, and rock are among the ten genres that are covered. There are one hundred tracks in each genre, all in WAV (*.wav) format. These are all 16-bit, monophonic audio files at 22,050 Hz.

Musical genre annotation is currently done by hand. A useful addition to music information retrieval systems would be automatic musical genre classification, which can help or replace the human user in this process. Furthermore, automatic genre classification of music offers a framework for feature development and assessment for any kind of content-based music signal analysis. The automatic categorization of audio signals into a hierarchy of musical genres is investigated in this paper. More precisely, it is suggested to represent pitch content, rhythmic content, and timbral texture using three feature sets [8]. Statistical pattern recognition classifiers trained on real-world audio collections are used to examine the effectiveness and relative significance of the suggested features. Real-time frame-based classification schemes and whole file classification schemes are described.

In this study, the issues are addressed by the application of machine learning. Ever since the first convolutional neural network was introduced, deep learning has advanced in areas such as object detection, recognition, and picture categorization and

segmentation [2]. CNN [2] is a unique class of neural network with a topology resembling a grid. This grid might be two-dimensional (like an image) or linear (like time series data). CNN lowers processing resources by utilizing a mechanism like to a multilayer perceptron. In addition to CNN, other tools used in comparative analysis include decision trees, multilayer perceptrons, artificial neural networks, and support vector machines.

System Architecture

An audio genre and mood classification system employs artificial intelligence to automatically categorize audio files by their genre and mood attributes, enhancing organization and user experiences in navigating diverse audio content. The system begins with data collection and preprocessing, where audio features are extracted and labelled. These features form the basis for training machine learning models, including K-nearest neighbours for genre classification and neural networks for mood classification. Once trained, the models are integrated and deployed for user interaction, enabling users to input audio files for classification. Predicted labels are evaluated for accuracy, and user feedback is collected to refine the models over time. Through continuous improvement efforts, the system adapts to evolving user preferences and advances in AI techniques, contributing to efficient audio content organization and exploration..



Figure 1: workflow of the system

Overview of the system architecture combining both the genre and mood classification of audio using AI:

Data Collection and Preprocessing:

Audio data for both genre and mood classification is collected from various sources. These audio files should be labeled with their respective genres and moods. Data preprocessing steps include:

Audio feature extraction: Features such as MFCCs, spectral features, and rhythm patterns are extracted from the audio files.

Labeling: Each audio file is labeled with its corresponding genre and mood.

filename	length	chroma_stft_mean	chroma_stft_var
blues.00000.0.wav	66149	0.335406363	0.091048293
blues.00000.1.wav	66149	0.343065351	0.086146526
blues.00000.2.wav	66149	0.346814752	0.092242889
blues.00000.3.wav	66149	0.363638788	0.086856157
blues.00000.4.wav	66149	0.335579425	0.088128544
blues.00000.5.wav	66149	0.376669735	0.089702107
blues.00000.6.wav	66149	0.379908741	0.088827312
blues.00000.7.wav	66149	0.331879944	0.092118852
blues.00000.8.wav	66149	0.347877383	0.094209172

Figure 2: Sample data to be processed

Training Data Preparation:

- The extracted features and their corresponding labels are used to create training datasets for both genre and mood classification tasks.

- To assess the performance of the model, the datasets are divided into training and test sets.

Model Training:

-For both genre and mood classification tasks, machine learning models are trained using the training datasets.

- The K-nearest neighbors (KNN) technique is utilized for genre classification based on MFCC characteristics in the genre classification model.

-Mood Classification Model: Using the retrieved audio information, a different model (such as a neural network or support vector machine) is developed for the purpose of classifying moods.

Integration and Deployment:

- The trained models for genre and mood classification are integrated into a single system.

- The system is deployed as an application or service where users can input audio files for classification.

User Interaction:

- Users interact with the system by providing audio files for classification.

- The system processes the input audio files and returns the predicted genre and mood labels.

Prediction and Evaluation:

- Upon receiving an audio file, the system predicts its genre and mood using the trained models.

- To assess the system's performance, the predicted labels are compared to the ground truth labels, if they are available.

- Evaluation metrics are computed for both genre and mood classification tasks, including accuracy, precision, recall, and F1-score.

Feedback and Model Refinement:

- User feedback on the predicted genre and mood labels can be collected to improve the performance of the system.

- The models can be retrained periodically using updated training data and refined feature extraction techniques to

enhance classification accuracy.

Continuous Improvement:

- The system undergoes continuous improvement based on user feedback, advancements in audio processing techniques, and updates in machine learning algorithms.
- Regular maintenance and updates ensure that the system remains effective and accurate in classifying audio based on genre and mood.

In essence, Model serves as the linchpin within the workflow, bridging the gap between user interactions and underlying execution systems, thereby facilitating efficient and effective audio classification and analysis.

Implementation Details(dataset, data, model, implementation module)

The implementation of genre and mood classification for audio files using AI constitutes a multifaceted process involving various components and methodologies to enable accurate classification and analysis. This section provides a detailed overview of the implementation details, starting with the integration of feature extraction and classification algorithms for genre and mood identification.

1. User Input Data

Using advanced AI algorithms designed specifically for music analysis, the system processes user-inputted audio files for genre and mood classification in a seamless manner. Robust genre and mood classification is made possible by the system's ability to extract important features from the audio signals using advanced machine learning techniques.

The system then uses these extracted features to train advanced classification models, which starts the classification process. These models can accurately classify audio files into different genres and moods, which reflects the complex emotional content that is present in the music. In addition, the system creates metadata for the audio files automatically, capturing crucial details about the qualities of the music, such as rhythm, pitch, and timbral texture. The categorization process relies heavily on this metadata, which makes it possible to fully comprehend and work with the audio data.

Artificial Intelligence (AI)-driven classification algorithms are incorporated into the music analysis system to improve its efficacy and precision in

discerning genre and mood characteristics. The system enables users to discover and enjoy music in novel ways by facilitating intuitive exploration and understanding of musical content through the seamless processing and analysis of audio data.

Ten genres—the blues, hip-hop, pop, metal, reggae, jazz, country, disco, and rock—are included in the GTZAN dataset, which was created by Tzanetakis and Cook. Each genre is represented by one hundred tracks in WAV format (*.wav), which are all 22,050Hz, monophonic, 16-bit audio files weighing 1.6 GB.

Eight hundred audio samples, or 80% of the data, were used to train the model, and two hundred audio samples, or the remaining 20% of the data, were used to test the model.

We will use this data for future processes.

2. Feature Extraction and processing

This section describes a multi-class semi-supervised DCGAN architecture and MFCC feature extraction for genre classification. Before classifying music genres, a multi-class semi-supervised DCGAN must be trained using labelled and unlabeled MFCC feature maps of audio samples from a sizable music dataset with ten distinct genres.

A. Extracting Features:

Let x be an audio clip, and let the set of MFCCs that were taken out of x be represented by $\text{MFCC}(x)$. The following procedures are used to extract the MFCCs:

- 1) Pre-emphasis: From the input signal x , the pre-emphasized signal y is obtained; the pre-emphasis coefficient is denoted by α :

$$y(t) = x(t) - \alpha x(t-1)$$

2) STFT: The following formula is used to calculate the Short-time Fourier Transform (STFT) of y .

$$X(m, k) = \sum_{n=0}^{N-1} y(n)w(n - m)e^{-j2\pi kn/N}$$

The symbols m , k , $w(n)$, and N represent the time frame index, frequency bin index, window function, and STFT window size, respectively.

3) Magnitude Spectrogram: Using frequency and amplitude, the magnitude spectrogram S describes signal X . S is obtained as follows:

$$S(m, k) = |X(m, k)|$$

4) Mel Spectrogram: The Mel spectrogram M can be made as follows by applying a Mel-scaled filter bank to S :

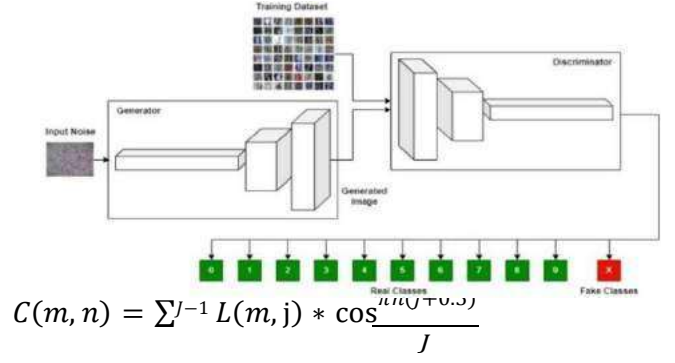
$$M(m, j) = \sum_{k=1}^{K-1} S(m, k)H(j, k)$$

Here, K is the number of frequency bins, $H(j, k)$ is the j th Mel filter for the k th frequency bin.

5) Log Mel Spectrogram: Mel spectrogram M is logarithmically transformed to produce Log Mel spectrogram L .

$$L(m, j) = \log(M(m, j))$$

6) MFCCs: By using the Discrete Cosine Transform (DCT) on the log-mel spectrum L , the MFCCs of the input signal x are obtained.



The number of Mel bands used for MFCC feature extraction is denoted by J in this case. The MFCC feature map of a jazz audio sample from the GTZAN dataset is shown in Fig. 1.

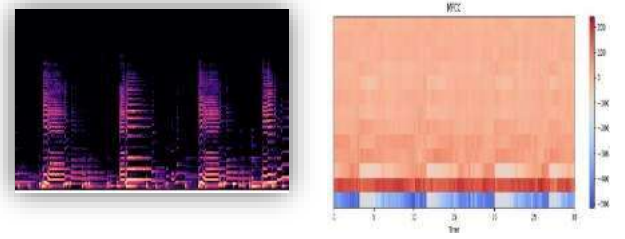


Fig. 1. (a) Mel spectrogram and (b) MFCC feature map of Jazz audio

B. Multiclass Semi-supervised DCGAN Architecture:.

Creating synthetic samples that fit any multiple-class distribution is the goal of a multi-class semi-supervised DCGAN.

The generator creates false samples, and the discriminator learns to discriminate between actual data from various classes. The generator creates artificial samples that are intended to resemble real data from a given class.

Fig. 2. Architecture of Multiclass Semi-supervised DCGAN for Classifying Music Genres

As can be seen in Fig. 2, the discriminator has developed into a multiclass classifier and must now learn how to categorize the MFCC feature mappings to a limited subset of the class labels. In addition to its fundamental capabilities, the discriminator must learn the probabilities of all the original dataset classes it has trained on to become a semi-supervised multiclass classifier [21]. The semi-supervised DCGAN was employed in this study [22] to classify musical genres. Convolutional layers, batch normalization

layers, and activation functions were transposed in a generator, and convolutional layers, batch normalization layers, and activation functions were combined in a discriminator.

1) The Generator Network 1) generates false images with the same size as the real MFCC feature maps by cascading a sequence of transposed convolutional layers together [Fig. 3]. Random vectors drawn from a 100D latent space are first input into the generator network. The final output in the final layer uses the Tanh activation function, but all intermediate layers utilize the ReLU activation function.

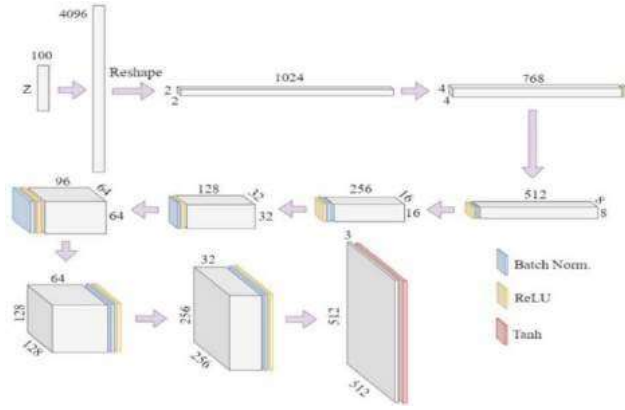


Fig 3. Generator network of semi-supervised DCGAN

The generator network denotes a function in which the noise vector input is represented by z . The generator seeks to minimise the following loss function, as shown in (7).

$$LG = -\log(D(G(z)))$$

With D standing for the discriminator network, the objective is to create synthetic samples that can fool the discriminator into thinking they are real.

2) Discriminator Network: To extract insightful features, real and synthetic images are passed through a series of convolutional layers that are sequentially connected to the final fully connected

layer [Fig. 4]. To predict class probability values, 4096 features are processed by the Softmax function. On the basis of 4096 features, the Sigmoid activation function determines whether feature maps are artificial or real.

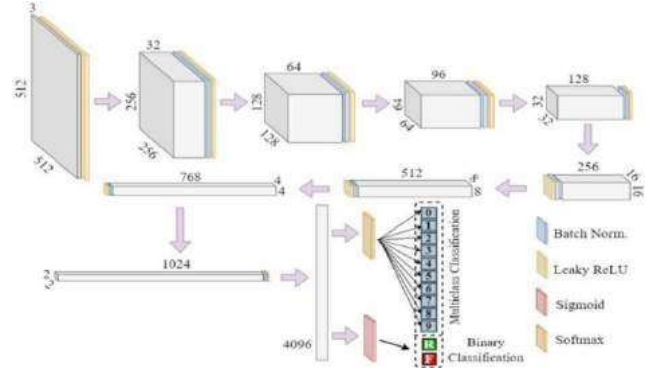


Fig 4. Discriminator network of semi-supervised DCGAN

In a semi-supervised GAN, the discriminator's loss combines the loss for classifying the labelled images into their respective classes and the loss for differentiating between real and fake images. The discriminator uses the following binary cross-entropy loss function to distinguish between real and fake images for the unlabeled data (both generated and real).

$$L_{D_{ul}} = -y \log D(x) - (1 - y) \log(1 - D(G(z)))$$

In the case of a genuine image x , the discriminator output is $D(x)$, and it ought to be close to 1. As the label indicating the sample's production (0) or authenticity (1), the discriminator's output for a created image $G(z)$ is $D(G(z))$, which should be quite close to 0. The discriminator separates the images into classes for labeled data using the categorical cross-entropy loss provided in (9).

$$L_{D_l} = -\sum(y \log(C(x)))$$

The discriminator's output, $C(x)$, is used to classify the labelled image x and y as the true labels of the respective images. The total discriminator loss, which combines these two loss components, $L_{D_{ul}}$ and L_{D_l} , is shown by equation (10).

Every genre is represented by multiple output samples produced by the generator network and multiple probability scores produced by the discriminator network [23]. All genres are included in the loss function, and the networks are trained to minimise the total loss over both labelled and unlabeled data.

Discussion and Future Work

The AI-powered system discussed in this paper is not just a technological advancement; it's a revolution in our understanding and interaction with music. It's akin to having a knowledgeable music aficionado at your disposal, someone who can guide you

through the intricate details of any song's genre and mood. This system can dissect a piece of music, analyze its components, and provide insights that were previously inaccessible to the average listener.

In this section, we delve into the implications of this advancement. We consider the potential impact on various sectors, from entertainment to education, and how it could transform the way we consume and create music. We also consider avenues for further development. One promising avenue is the refinement of our classification techniques to achieve greater accuracy and granularity. Through continued enhancements in feature extraction and algorithm optimization, we aim to better capture the essence of different genres and moods. This could lead to more accurate recommendations, more engaging playlists, and a richer overall music experience.

Additionally, exploring ensemble learning methods and incorporating diverse data sources could bolster the system's robustness and versatility. By combining different models and datasets, we can create a more comprehensive and resilient system. This could make the system more adaptable to different types of music and more responsive to user preferences.

Moreover, there's a pressing need for intuitive interfaces that enable seamless interaction with the classification system. Integrating it into popular music platforms or editing software could democratize access, empowering users to explore, discover, and create with ease. This could lead to a surge in creativity, as more people have the tools and knowledge to create their own music.

Looking forward, we're particularly excited about the potential for personalization in user experiences. Imagine recommendation systems that craft bespoke playlists tailored to individual preferences and mood states, or interactive visualizers that adapt dynamically to the mood of the music. These innovations promise to elevate our engagement with music to new heights. They could make listening to music a more immersive and personal experience, transforming it from a passive activity to an interactive journey.

The AI-driven classification system represents a significant advancement in music analysis. As we strive to refine the technology and enhance user accessibility, we anticipate a future where music becomes even more personalized and immersive across diverse domains. From concerts to classrooms, from studios to streaming platforms, this technology has the potential to reshape the landscape of music.

Conclusion

In the field of music information retrieval, musical genre classification has gained a lot of attention lately. The goal of this field, which straddles the boundaries of computer science, information science, and musicology, is to extract useful information from musical data. Notably, automatic music genre classifiers have been made possible by recent developments in machine learning and signal processing, despite the fact that music genres have always been defined and labeled by humans. These classifiers are algorithms that meet, if not beyond, human

annotation ability when it comes to analyzing a piece of music and identifying its genre.

Music genres are naturally ambiguous and flexible. They are cultural constructs that can vary greatly depending on geographical location, historical period, and individual perception. However, despite these challenges, automatic classification has been highly successful. It has demonstrated a well-performing level of accuracy in differentiating genres as well as sound groups, which are clusters of songs that share similar sonic characteristics.

The present work considers three feature sets that aim to extract the essential information from the music signal features. These features capture pitch content, rhythmic content, and timbral texture. Pitch content refers to the frequencies of the notes in a piece of music, rhythmic content refers to the timing and duration of these notes, and timbral texture refers to the quality of the sound, such as its brightness or roughness. These feature sets serve as a primary source of information about the music and are the most indicative of the music characteristics for classifying music genres.

Through the use of statistical pattern recognition classifiers that have been trained on large-scale real-world audio collections, the robustness and efficacy of these feature sets are evaluated throughout a wide variety of musical styles. Promising findings are obtained from the examination of these feature sets: in non-real-time circumstances, the classification accuracy reaches 61%, while in real-time implementations, it reaches 44%. These results highlight how well the suggested feature sets can classify music into established genres, setting the foundation for more advanced systems for music analysis and retrieval.

Moreover, the success of these feature sets in genre classification highlights their versatility and applicability across various music-related tasks. They can be used not only for genre classification, but also for other tasks such as mood detection, artist identification, and song recommendation.

In terms of future directions, genre classification offers a number of opportunities for investigation and advancement. To provide more detailed insights into the varied terrain of musical genres, one approach to this is to broaden and deepen the genre hierarchy to include a wider range of classifications. The system's capacity to comprehend the complex facets of music may also be improved by adding more semantic descriptions, like vocal style or emotional content, to the classification framework.

Furthermore, improving the performance of feature sets, particularly the pitch content feature set, remains a focal point of future investigation. This could involve exploring alternative pitch detection algorithms, including advanced techniques based on cochlear models, to enhance the accuracy and robustness of pitch representation in music signals. Additionally, the exploration of real-time implementations of feature sets for harmonic content and rhythmic structure presents an exciting opportunity to enhance classification efficiency and responsiveness.

The suggested feature sets have a plethora of uses beyond genre categorization, such as query-by-example similarity retrieval of music signals and audio thumbnailing. Through the use of discrete feature sets to describe harmony, rhythm, and timbre, these systems may support several types of similarity retrieval, allowing users to search and find music according to certain attributes and tastes. Furthermore, using data from other sources—like melody and singer-vocabulary—could improve genre categorization abilities and offer more in-depth understanding of musical content.

This research represents a significant step forward in advancing genre classification and exploring the potential of feature-based music analysis. By embracing emerging techniques and methodologies, researchers aim to unlock deeper insights into musical content and foster innovation in music classification and retrieval systems, ultimately enhancing the way we interact with and explore music in the digital age.

References

Some of the references that helped us in our project are:

- Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing*, 10(5), 293-302..
- Bahuleyan, H. (2018). Music genre classification using machine learning techniques. *arXiv preprint arXiv:1804.01149*..
- Pratiwi, T., Sunyoto, A., & Ariatmanto, D. (2024). Music Genre Classification Using K-Nearest Neighbor and Mel-Frequency Cepstral Coefficients. *Sinkron: jurnal dan penelitian teknik informatika*, 8(2), 861-867. Sial, A. H., Rashdi, S. Y. S., & Khan, A. H. (2021).
- Middha, Ms. Karuna and Kandwal, Siddharth and Sharma, Keshav and Verma, Akshun, *Music Spectrum Genre Identifier* (March 21, 2024).
- Pratiwi, T. ., Sunyoto , A. ., & Ariatmanto , D. . (2024). Music Genre Classification Using K-Nearest Neighbor and Mel-Frequency Cepstral Coefficients. *Sinkron : Jurnal Dan Penelitian Teknik Informatika*, 8(2), 861-867. <https://doi.org/10.33395/sinkron.v8i2.12912>
- Lee, N., & Mahmud, R. (2024, January). Enhancing Music Emotion Classification with Lyrics and Audio Features. In *2024 3rd International Conference on Digital Transformation and Applications (ICDXA)* (pp. 1-5). IEEE..
- N. Ndou, R. Ajoodha and A. Jadhav, "Music Genre Classification: A Review of Deep-Learning and Traditional Machine-Learning Approaches," *2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, Toronto, ON, Canada, 2021, pp.1-6,doi: 10.1109/IEMTRONICS52119.2021.9422487.
- A. Ghildiyal, K. Singh and S. Sharma, "Music Genre Classification using Machine Learning," *2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, Coimbatore, India, 2020,

B. SOURCE CODE

Program code:

5.1 For the backend server:

```
import numpy as np

import scipy.io.wavfile as wav

from python_speech_features import mfcc

import pickle

import os


# Define functions for distance calculation, finding neighbors, and classifying

def distance(instance1, instance2, k):

    distance = 0

    mm1 = instance1[0]

    cm1 = instance1[1]

    mm2 = instance2[0]

    cm2 = instance2[1]

    distance = np.trace(np.dot(np.linalg.inv(cm2), cm1))

    distance += (np.dot(np.dot((mm2-mm1).transpose(), np.linalg.inv(cm2)), mm2-mm1))

    distance += np.log(np.linalg.det(cm2)) - np.log(np.linalg.det(cm1))

    distance -= k

    return distance


def getNeighbors(trainingset, instance, k):

    distances = []

    for x in range(len(trainingset)):
```

```

    dist = distance(trainingset[x], instance, k) + distance(instance, trainingset[x], k)

    distances.append((trainingset[x][2], dist))

distances.sort(key=lambda x: x[1])

neighbors = []

for x in range(k):

    neighbors.append(distances[x][0])

return neighbors

def nearestclass(neighbors):

    classVote = {}

    for x in range(len(neighbors)):

        response = neighbors[x]

        if response in classVote:

            classVote[response] += 1

        else:

            classVote[response] = 1

    sorter = sorted(classVote.items(), key=lambda x: x[1], reverse=True)

    return sorter[0][0]

# Load the trained model

with open('mydataset.dat', 'rb') as f:

    dataset = []

    while True:

        try:

            dataset.append(pickle.load(f))

        except EOFError:

```

```
f.close()
```

```
break
```

```
# Function to classify an input audio file
```

```
def classify_audio(input_file, k=5):
```

```
    rate, sig = wav.read(input_file)
```

```
    mfcc_feat = mfcc(sig, rate, winlen=0.020, appendEnergy=False)
```

```
    covariance = np.cov(np.matrix.transpose(mfcc_feat))
```

```
    mean_matrix = mfcc_feat.mean(0)
```

```
    instance = (mean_matrix, covariance, None) # The class label is set to None
```

```
    neighbors = getNeighbors(dataset, instance, k)
```

```
    predicted_class = nearestclass(neighbors)
```

```
    return predicted_class
```

```
genre={1:"Blues",2:"Classical",3:"Country",4:"Disco",5:"HipHop",6:"Jazz",7:"Metal",8:"Pop",9:"  
Raggae",10:"Rock"}
```

```
def server_output(input):
```

```
    pred = classify_audio(input_file=input)
```

```
    return genre[pred]
```

```
# Example usage:
```

```
if __name__ == "__main__":
```

```
    input_file_path = 'WORK\Sunflower.wav'
```

```
    predicted_genre = classify_audio(input_file_path)
```

```
    print("Predicted genre:", predicted_genre)
```

```
    print("Genre is",genre[predicted_genre])
```

5.2 For the Flask connection between frontend and backend

```
from flask import Flask, render_template, request

from werkzeug.utils import secure_filename

import os

from forest_fire import server_output


app = Flask(__name__)


# Set the upload folder

UPLOAD_FOLDER = 'uploads'

app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER


# Allowed extensions for audio files

ALLOWED_EXTENSIONS = {'mp3', 'wav'}


# Function to check if the file extension is allowed

def allowed_file(filename):

    return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS


# Route to handle form submission

@app.route('/predict', methods=['POST'])

def predict():

    # Check if the POST request has the file part

    if 'audio_file' not in request.files:
```

```
return 'No file part'
```

```
file = request.files['audio_file']
```

```
# If user does not select file, browser also
```

```
# submit an empty part without filename
```

```
if file.filename == "":
```

```
    return 'No selected file'
```

```
# If the file is allowed, save it to the upload folder
```

```
if file and allowed_file(file.filename):
```

```
    filename = secure_filename(file.filename)
```

```
    # print("filename ==>", filename)
```

```
    # filepath = os.path.join(app.config['UPLOAD_FOLDER'], filename)
```

```
    # # file.save(os.path.join('uploads', filename))
```

```
    # print("path ==>",filepath)
```

```
    # file.save(filepath)
```

```
    # print(filepath)
```

```
    filepath = 'C:\\Users\\roykb\\OneDrive\\Desktop\\web interface\\1.0 ui\\uploads\\' +
```

```
filename
```

```
    file.save(filepath)
```

```
# Now you can process the uploaded audio file
```

```
# For example, you can pass the file path to your model for prediction
```

```
# prediction = your_model_function(filepath)
```

```

    # After processing, you can return the prediction or any other response
    result = server_output(input=filepath)

    return render_template("forest_fire.html", pred=result)

else:

    return 'Invalid file type. Allowed extensions are: mp3, wav'

@app.route("/")
def home():

    return render_template("forest_fire.html", pred="")

if __name__ == '__main__':

    app.run(debug=True)

```

5.3 Program Code for Front End webpage(HTML):

```

<!DOCTYPE html>

<html lang="en">

<head>

    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>

    <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1.0"/>

    <title>Starter Template - Materialize</title>

    <!-- CSS -->

```

```

<link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">

<link href="/static/css/materialize.css" type="text/css" rel="stylesheet"
media="screen,projection"/>

<link href="/static/css/style.css" type="text/css" rel="stylesheet" media="screen,projection"/>

</head>

<body>

<nav class="light-blue lighten-1" role="navigation">

  <div class="nav-wrapper container"><a id="logo-container" href="#" class="brand-
logo">Logo</a>

    <ul class="right hide-on-med-and-down">

      <li><a href="#">Navbar Link</a></li>

    </ul>

    <ul id="nav-mobile" class="sidenav">

      <li><a href="#">Navbar Link</a></li>

    </ul>

    <a href="#" data-target="nav-mobile" class="sidenav-trigger"><i class="material-
icons">menu</i></a>

  </div>

</nav>

<div class="section no-pad-bot" id="index-banner">

  <div class="container">

    <br><br>

    <h1 class="header center orange-text">AI Musical Genre Classifier</h1>

    <div class="row center">

```

<h5 class="header col s12 light">Predict the audios genre :

</h5>

</div>

<div class="row">

<form action="/predict" method="post" class="col s12" enctype="multipart/form-data">

<div class="row centre">

<div class="file-field input-field col s8">

<div class="btn">

Audio File

<input type="file" name="audio_file">

</div>

<div class="file-path-wrapper">

<input class="file-path validate" type="text">

</div>

</div>

</div>

<div class="row center">

<button type="submit" class="btn-large waves-effect waves-light orange">Predict
it</button>

</div>

</form>

<p class="result">{{pred}}</p>

</div>

</div>

</div>

<footer class="page-footer orange">

<div class="container">

<div class="row">

<div class="col l6 s12">

<h5 class="white-text">Company Bio</h5>

<p class="grey-text text-lighten-4">We are a team of college students working on this project like it's our full time job. Any amount would help support and continue development on this project and is greatly appreciated.</p>

</div>

<div class="col l3 s12">

<h5 class="white-text">Settings</h5>

Link 1

Link 2

Link 3

```
</li><a class="white-text" href="#">Link 4</a></li>
```

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
```

```
<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1.0"/>
```

```
<title>Starter Template - Materialize</title>
```

```
<!-- CSS -->
```

```
<link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
```

```
<link href="/static/css/materialize.css" type="text/css" rel="stylesheet"
media="screen,projection"/>
```

```
<link href="/static/css/style.css" type="text/css" rel="stylesheet" media="screen,projection"/>
```

```
</head>
```

```
<body>
```

```
<nav class="light-blue lighten-1" role="navigation">
```

```
<div class="nav-wrapper container"><a id="logo-container" href="#" class="brand-
logo">Logo</a>
```

```
<ul class="right hide-on-med-and-down">
```

```
<li><a href="#">Navbar Link</a></li>
```

```
</ul>
```

```
<ul id="nav-mobile" class="sidenav">
```

```
<li><a href="#">Navbar Link</a></li>
```

```
</ul>
```

```
<a href="#" data-target="nav-mobile" class="sidenav-trigger"><i class="material-  
icons">menu</i></a>
```

```
</div>
```

```
</nav>
```

```
<div class="section no-pad-bot" id="index-banner">
```

```
<div class="container">
```

```
<br><br>
```

```
<h1 class="header center orange-text">AI Musical Genre Classifier</h1>
```

```
<div class="row center">
```

```
<h5 class="header col s12 light">Predict the audios genre :
```

```
<br>
```

```
</h5>
```

```
</div>
```

```
<div class="row">
```

```
<form action="/predict" method="post" class="col s12" enctype="multipart/form-data">
```

```
<div class="row centre">
```

```
<div class="file-field input-field col s8">
```

```
<div class="btn">
```

```
<span>Audio File</span>
```

```
<input type="file" name="audio_file">
```

```
</div>
```

```
<div class="file-path-wrapper">
```

```
<input class="file-path validate" type="text">
```

```
</div>
```

```
</div>
```

```
</div>
```

```
<div class="row center">
```

```
<button type="submit" class="btn-large waves-effect waves-light orange">Predict  
it</button>
```

```
</div>
```

```
</form>
```

```
<p class="result">{{pred}}</p>
```

```
</div>
```

```
<br><br>
```

```
</div>
```

```
</div>
```

```
<footer class="page-footer orange">
```

```
<div class="container">
```

```
<div class="row">
```

```
<div class="col l6 s12">
```

<h5 class="white-text">Company Bio</h5>

<p class="grey-text text-lighten-4">We are a team of college students working on this project like it's our full time job. Any amount would help support and continue development on this project and is greatly appreciated.</p>

</div>

<div class="col l3 s12">

<h5 class="white-text">Settings</h5>

Link 1

Link 2

Link 3

Link 4

</div>

<div class="col l3 s12">

<h5 class="white-text">Connect</h5>

Link 1

Link 2

Link 3

Link 4

</div>

</div>

</div>

<div class="footer-copyright">

<div class="container">

Made by <a class="orange-text text-lighten-3"

href="http://https://www.linkedin.com/in/abramroy.com">Abram Roy

</div>

</div>

</footer>

<!-- Scripts-->

<script src="https://code.jquery.com/jquery-2.1.1.min.js"></script>

<script src=".js/materialize.js"></script>

<script src="js/init.js"></script>

</body>

</html>

