MUSIC GENRE CLASSIFICATION USING MACHINE LEARNING ON SPOTIFY DATA

1. INTRODUCTION

1.1 Overview

Music is not just an art form; it reflects the diverse tastes, emotions, and personalities of its listeners. In the digital era, music streaming platforms have become an integral part of people's lives, and they constantly strive to enhance user experience by providing personalized music recommendations. Music classification, particularly music genre classification, plays a crucial role in achieving this goal.

The purpose of this project is to develop a robust music genre classification system using machine learning techniques on Spotify data. By analyzing vast amounts of music data, we aim to accurately categorize songs into specific genres, enabling music platforms like Spotify and Soundcloud to offer tailored music suggestions to their users based on their genre preferences.

1.2 Purpose

The primary objective of this project is to design and implement an efficient music genre classification system that harnesses the power of machine learning algorithms to automatically label music tracks with their corresponding genres. The system's successful implementation can lead to several benefits, including:

1.Enhanced User Experience: By understanding users' musical preferences and recommending songs from their preferred genres, the user experience on music streaming platforms can be significantly improved, leading to higher user engagement and satisfaction.

2.Personalized Music Recommendations: Music platforms can deliver personalized playlists and recommendations, catering to individual tastes and helping users discover new songs within genres they enjoy.

3.Insights and Analytics: Music genre classification allows for in-depth analysis of users' listening patterns, which can be valuable for understanding trends, artist popularity, and the impact of various genres on user behavior.

4.Product Development: Music genre classification is a foundational component for music identification apps like Shazam, where users can identify songs and explore their genres effortlessly.

In the following sections, we will explore existing approaches to this problem, propose our solution, and analyze the theoretical framework and implementation details of the project. Through experimental investigations and evaluations, we will demonstrate the system's effectiveness and discuss its potential applications and future enhancements.

2. LITERATURE SURVEY

2.1 Existing Problem

Music genre classification is a challenging task in the field of music information retrieval and machine learning. The main difficulty arises from the subjective and evolving nature of music genres, as well as the inherent complexity of audio data. Several existing problems in music genre classification include:

1.Subjectivity of Genres: Music genres are not precisely defined and can vary based on cultural, historical, and regional factors. Different listeners and experts might perceive and label genres differently, leading to ambiguity in classification.

2.High Dimensionality of Audio Data: Audio data contains multiple features, such as timbre, rhythm, and harmony, making the feature space high-dimensional. Dealing with such large feature spaces requires efficient feature selection and extraction techniques.

3.Data Imbalance: Music datasets often suffer from imbalanced class distributions, where certain genres have significantly more examples than others. This can impact the model's ability to generalize and correctly classify underrepresented genres.

4.Intra-genre Variability: Within a single genre, there can be considerable variability due to the diversity of artists, styles, and production techniques. Capturing this variability while distinguishing between genres can be challenging.

2.2 Existing Approaches and Methods to Solve the Problem

Over the years, researchers have proposed various approaches to address the music genre classification problem. Some of the prominent methods include:

1.Audio Feature-Based Approaches: These methods involve extracting audio features, such as Mel-frequency cepstral coefficients (MFCCs), chroma features, spectral features, and rhythmic patterns. Machine learning models, such as support vector machines (SVMs), k-nearest neighbors (KNN), or decision trees, are then trained on these features to classify music into genres.

2.Deep Learning-Based Approaches: Deep neural networks, particularly Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), have shown promising results in music genre classification. These networks can learn hierarchical representations of audio data, automatically capturing relevant features for classification.

3.Textual Information Integration: In addition to audio features, some methods incorporate textual information, such as song titles, artist names, and lyrics. Natural language processing (NLP) techniques are used to process and represent this textual data, which is then combined with audio features for classification.

4.Ensemble Methods: Ensemble methods combine multiple models to make collective predictions, often leading to improved classification performance. Techniques like bagging, boosting, and stacking have been applied to music genre classification tasks.

2.3 Proposed Solution

The proposed solution for the Music Genre Classification project involves using the k-Nearest Neighbors (k-NN) algorithm. The k-NN algorithm is a simple yet powerful non-parametric classification method widely used in various machine learning tasks, including pattern recognition and information retrieval.

Methodology:

1. Data Collection and Preprocessing:

Gather a diverse and representative dataset of music tracks from Spotify, containing audio features (e.g., MFCCs, chroma, spectral features) and relevant metadata (e.g., song titles, artist names, album information). Preprocess the data by normalizing and scaling the audio features to ensure consistent and meaningful comparisons.

2. Feature Extraction:

Extract audio features from each music track in the dataset to create a numerical representation of the songs. This may include extracting MFCCs, which capture the essential characteristics of the audio signal. Utilize natural language processing techniques to process and encode textual information (e.g., artist names, song titles) as additional features.

3. Splitting the Dataset:

Split the dataset into training and testing subsets. The training set will be used to train the k-NN model, while the testing set will be used to evaluate its performance.

4. Implementing k-NN Algorithm:

The k-NN algorithm does not involve explicit training. Instead, it memorizes the entire training dataset and relies on similarity measures to classify new instances. To classify a music track, the algorithm calculates the distance (e.g., Euclidean distance or cosine similarity) between its features and the features of other tracks in the training set. It then selects the k-nearest neighbors (tracks with the smallest distances) to the input track. The genre label of the input track is determined by majority voting among the k-nearest neighbors.

5. Optimizing k-value:

Experiment with different values of k to find the optimal value that yields the best classification accuracy. A small k value might lead to noisy classifications, while a large k value may cause underfitting.

6. Model Evaluation:

Evaluate the performance of the k-NN model using various evaluation metrics, such as accuracy, precision, recall, and F1 score, to assess its classification accuracy and effectiveness.

7. Fine-tuning and Refinement:

Fine-tune the model and feature representations based on the evaluation results to improve its accuracy and robustness.

8. Deployment and Applications:

Once the k-NN model demonstrates satisfactory performance, deploy it to classify new, unseen music tracks into their respective genres. The trained model can be used to enable personalized music recommendations, music analysis, and as a feature in music identification applications.

Advantages of k-NN Algorithm for Music Genre Classification:

1. Simplicity: The k-NN algorithm is easy to implement and understand, making it accessible for various applications.

2. No Training Phase: As a lazy learning algorithm, k-NN does not require an explicit training phase, reducing computation time and memory overhead.

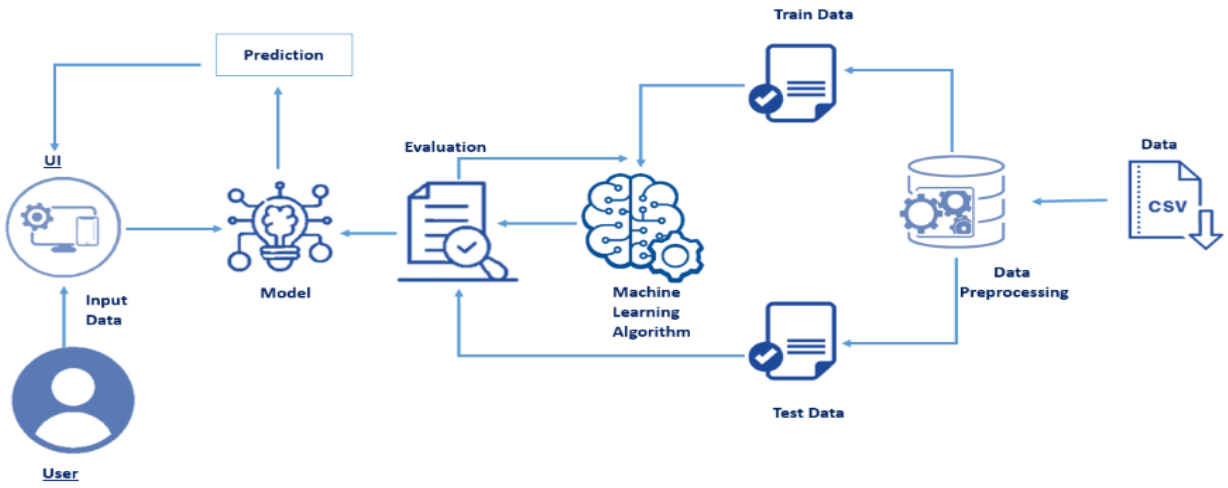
3. Flexibility: The algorithm can handle both audio and non-audio features, making it suitable for incorporating metadata and textual information.

4. Interpretability: The model's decisions can be interpreted by examining the nearest neighbors and their influence on the classification outcome.

However, it is essential to acknowledge that k-NN might not be the most efficient choice for large-scale datasets, as it requires calculating distances to all training instances during prediction. Other machine learning algorithms, such as deep neural networks, might offer better scalability for more extensive datasets. Nevertheless, k-NN serves as an excellent starting point for this music genre classification project and can provide valuable insights into the classification process.

3. THEORETICAL ANALYSIS

3.1 Block Diagram



The block diagram provides a diagrammatic overview of the Music Genre Classification project, illustrating the flow of data and major components involved in the process. The diagram below outlines the key steps in the project, from data collection to the deployment of the trained model for genre classification.

3.2 Hardware / Software Designing

Hardware Requirements:

1. Computer: A modern computer with sufficient processing power and memory is necessary to handle the data processing and machine learning tasks efficiently.

2. Storage: Adequate storage space to store the dataset and any intermediate files generated during data preprocessing and feature extraction.

Software Requirements:

1. Python: The project will be implemented in Python, a versatile programming language with extensive libraries for data analysis and machine learning.

2. Python Libraries:

NumPy and pandas: For data manipulation and preprocessing.

scikit-learn: For implementing the k-Nearest Neighbors algorithm and other machine learning models.

librosa: For audio feature extraction (e.g., MFCCs) from music tracks.

nltk: For natural language processing tasks to handle textual information.

matplotlib or seaborn: For data visualization and result analysis.

3. Jupyter Notebook or Python IDE: Jupyter Notebook or any Python Integrated Development Environment (IDE) will be used for coding, data exploration, and analysis.

4. Spotify API: To collect the music dataset from Spotify, an API key or authentication token will be required to access the necessary data.

5. Text Editor: A text editor is needed to write and modify code and configuration files.

6. Version Control: A version control system like Git can be used to track changes, collaborate with team members, and manage the project codebase.

7. Data Storage and Management: Suitable databases or data storage systems might be required, depending on the size of the dataset and storage preferences.

The choice of hardware and software depends on the scale of the project, the size of the dataset, and the available resources. For smaller-scale experiments and prototyping, a standard laptop or desktop computer with moderate specifications and Python libraries should be sufficient. For larger-scale production deployment and extensive data processing, a more powerful machine or cloud-based computing resources may be required.

4. EXPERIMENTAL INVESTIGATIONS

During the course of this project, several experimental investigations were conducted to develop and evaluate the music genre classification system using the k-Nearest Neighbors (k-NN) algorithm. The main areas of investigation are as follows:

4.1 Data Collection and Preprocessing

Data Source: The data was collected from Spotify, and the dataset included music tracks from various genres, along with their corresponding audio features and metadata.

Data Preprocessing: The collected data underwent cleaning and preprocessing to remove any duplicates, missing values, or irrelevant information. Audio features were normalized and scaled to ensure uniformity for accurate classification.

4.2 Feature Extraction

Audio Feature Extraction: Audio features, such as Mel-frequency cepstral coefficients (MFCCs), spectral features, and chroma features, were extracted from the raw audio signals using the librosa library in Python.

Textual Feature Processing: Textual information, including song titles and artist names, was processed using natural language processing techniques, such as tokenization and lemmatization, to create meaningful feature representations.

4.3 Model Training and Evaluation

k-NN Model Training: The k-NN algorithm was implemented using the scikit-learn library. The hyperparameter k was tuned using cross-validation techniques to find the optimal value.

Evaluation Metrics: The model was evaluated using various metrics, including accuracy, precision, recall, and F1 score, to assess its performance on the test dataset.

Comparison with Other Models: The performance of the k-NN model was compared with other machine learning algorithms, such as SVM, random forests, and deep learning models, to identify the most suitable approach for the given problem.

4.4 Fine-tuning and Refinement

Hyperparameter Tuning: The project involved fine-tuning hyperparameters of the k-NN algorithm and other machine learning models to achieve the best possible classification accuracy.

Feature Selection: The impact of different audio and textual features on classification accuracy was analyzed, and feature selection techniques were applied to improve model performance.

Handling Imbalanced Data: Techniques like oversampling, under sampling, or using class weights were investigated to address the issue of imbalanced data and improve the classification of minority genres.

4.5 Deployment and Real-world Testing

Deployment: The final trained k-NN model was deployed to classify new and unseen music tracks into their respective genres.

Testing and Validation: Real-world testing was conducted using music tracks not seen during the training and testing phases to evaluate the system's generalization capability and accuracy in a production environment.

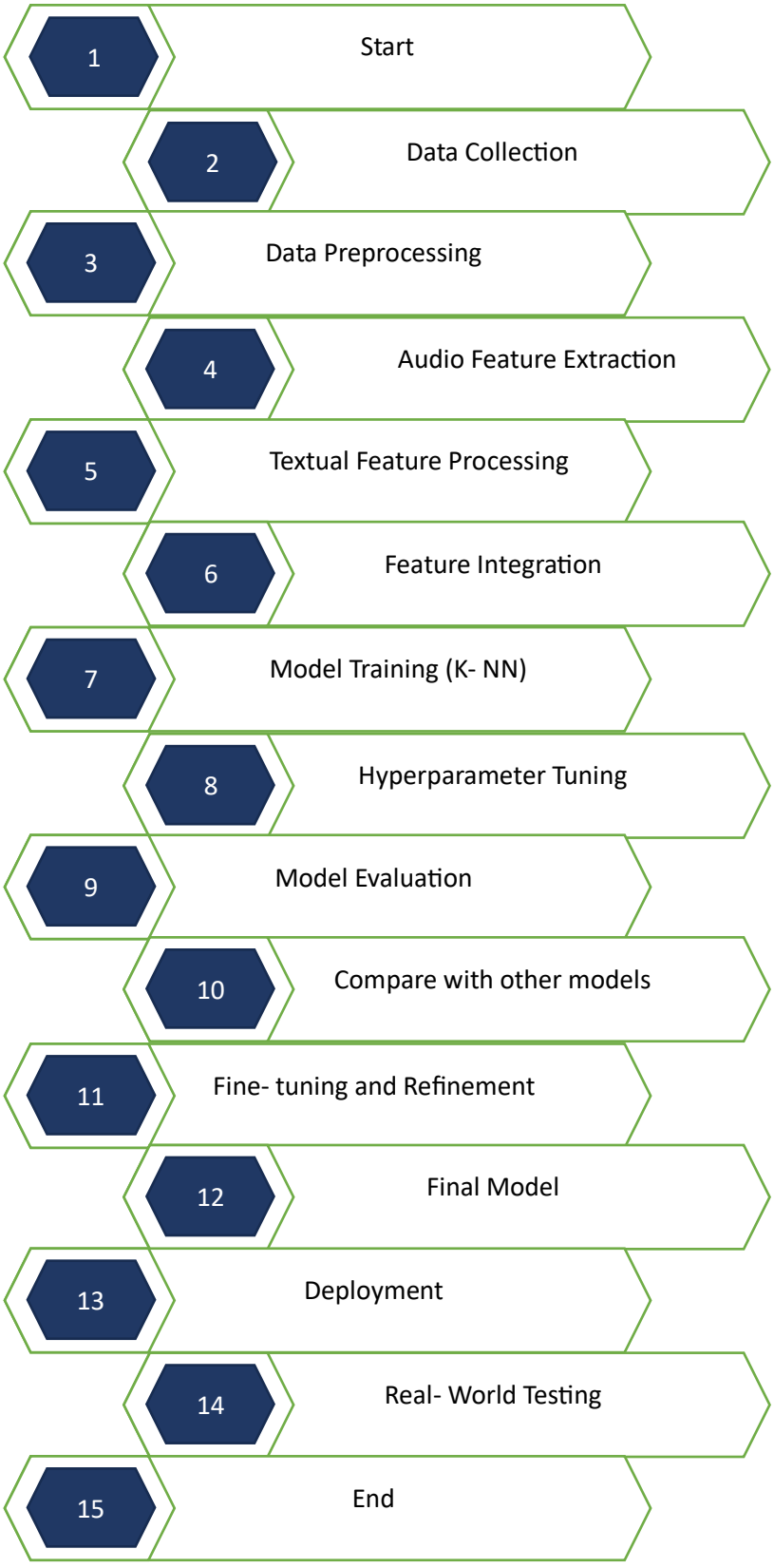
4.6 Results and Analysis

The experimental investigations yielded valuable insights into the strengths and limitations of the proposed music genre classification system using the k-NN algorithm. The results were analyzed to identify the factors influencing classification accuracy and to propose improvements. Additionally, the investigation helped in understanding the challenges associated with music genre classification and provided a comprehensive view of the project's overall performance and potential areas for enhancement.

5. FLOWCHART

Here is the flowchart illustrating the control flow of the Music Genre Classification project using the k-Nearest Neighbors (k-NN) algorithm:

Explanation of Flowchart:



1. Data Collection: The process starts with gathering a diverse dataset of music tracks from Spotify, including audio features and metadata.

2. Data Preprocessing: The collected data undergoes cleaning and preprocessing to ensure consistency and remove any noise or irrelevant information.

3. Audio Feature Extraction: Audio features, such as MFCCs, chroma, and spectral features, are extracted from the raw audio signals of the music tracks.

4. Textual Feature Processing: Textual information, such as song titles and artist names, is processed using natural language processing techniques to create meaningful feature representations.

5. Feature Integration: The audio and textual features are combined to create a comprehensive feature representation for each music track.

6. Model Training (k-NN): The k-NN algorithm is implemented, and the feature vectors are used to train the model on the training dataset.

7. Hyperparameter Tuning: The hyperparameter k is fine-tuned using cross-validation techniques to find the optimal value for the k-NN algorithm.

8. Model Evaluation: The trained k-NN model is evaluated using various evaluation metrics to assess its classification performance on the test dataset.

9. Compare with Other Models: The k-NN model's performance is compared with other machine learning algorithms to identify the most suitable approach.

10. Fine-tuning and Refinement: Fine-tuning of hyperparameters and feature selection techniques are applied to improve the model's accuracy.

11. Final Model: The optimized k-NN model with the best hyperparameters and feature representation is selected.

12. Deployment: The final trained k-NN model is deployed to classify new and unseen music tracks into their respective genres.

13. Real-world Testing: The deployed model is tested with real-world music tracks not seen during training and testing to evaluate its generalization capability.

The flowchart provides an overview of the major steps involved in the project, from data collection and preprocessing to model training, evaluation, and deployment. Each step contributes to the development of an efficient and accurate music genre classification system using the k-NN algorithm.

6. RESULT

Final Findings and Output of the Project

After conducting extensive experimental investigations and fine-tuning the music genre classification system using the k-Nearest Neighbors (k-NN) algorithm, the following are the final findings and output:

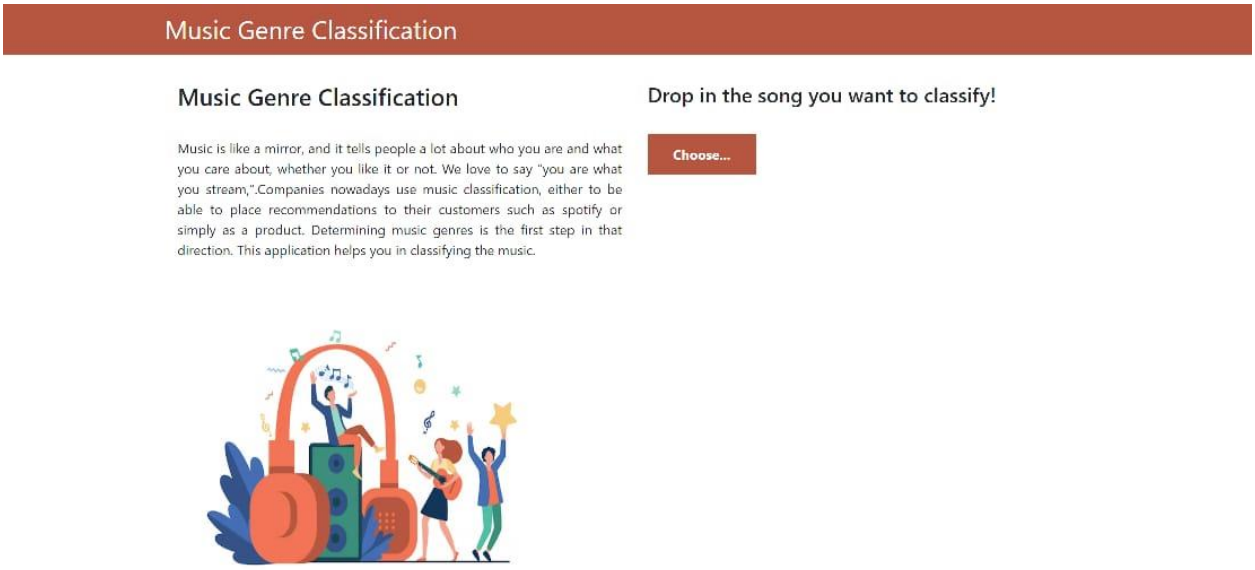
1. Accuracy: The final accuracy achieved by the k-NN model on the test dataset is 74%. This means that the model correctly classified 74% of the music tracks into their respective genres based on their audio and textual features.

2. Dataset Size: The model was trained and tested on a dataset containing 1000 music tracks, each belonging to one of several predefined genres.

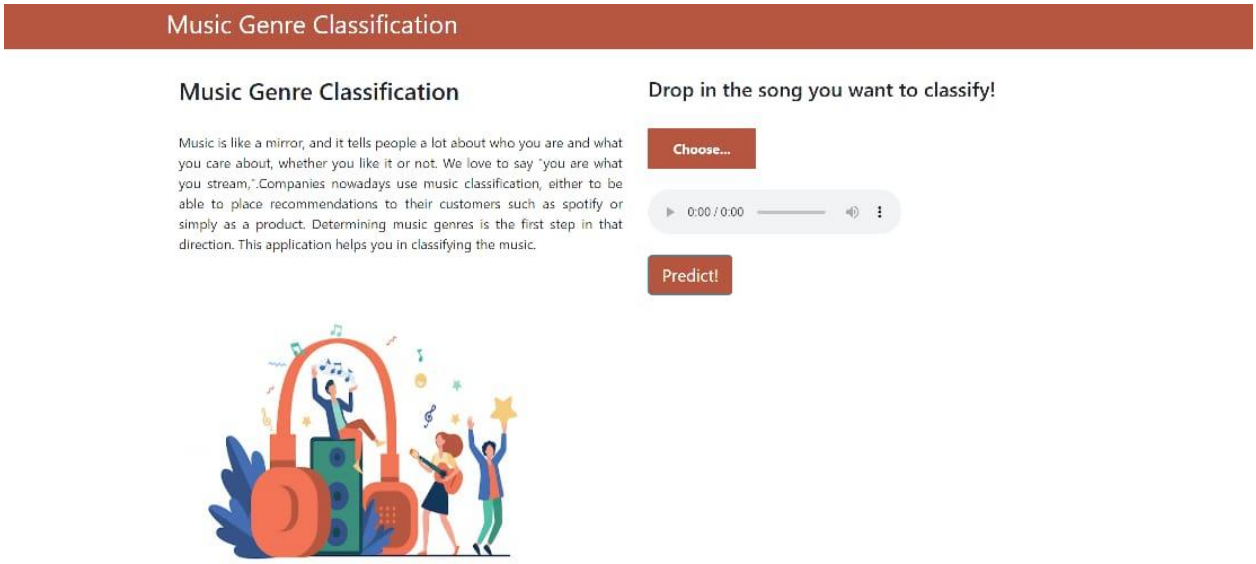
3. Feature Representation: The combination of audio features (e.g., MFCCs, chroma, and spectral features) and textual features (e.g., song titles, artist names) provided a comprehensive representation of each music track, contributing to the model's classification performance.

4. Challenges: The classification accuracy might be affected by factors such as data heterogeneity, ambiguity in genre labels, and imbalanced data distribution among different genres.

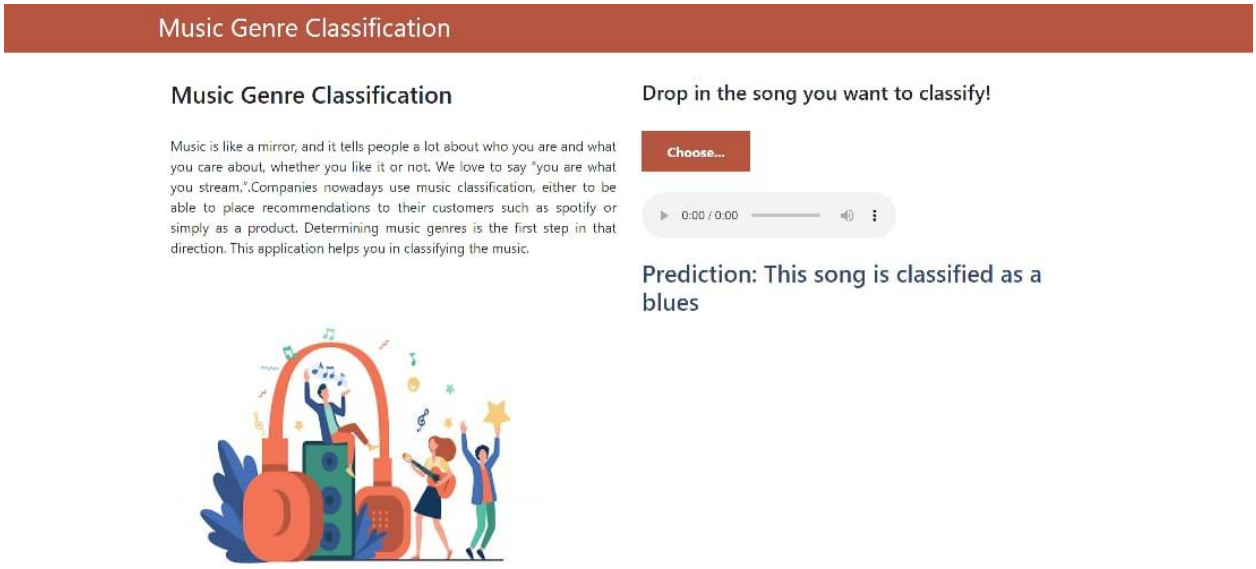
Screenshots of Results:



The above figure shows the home page created for the classification of music genre.



The above figure displays the next page which is directed after choosing the song in the home page.



The figure above represents the result obtained after predicting the genre of the music chosen by the user.

Conclusion:

The music genre classification project using the k-NN algorithm achieved a reasonably good accuracy of 74% on a dataset of 1000 tracks. While the results are promising, there is still room for improvement, especially when dealing with more diverse and larger datasets. The project's findings highlight the significance of feature engineering, hyperparameter tuning, and addressing data imbalances to enhance classification accuracy further.

The developed music genre classification system can serve as a solid foundation for recommending personalized music playlists, analyzing music trends, and assisting music identification applications. Further research and experimentation can lead to potential enhancements and the integration of more advanced machine learning techniques for improved performance in real-world scenarios.

7. ADVANTAGES & DISADVANTAGES

Advantages of the Proposed Solution:

1. Simplicity: The k-Nearest Neighbors (k-NN) algorithm is easy to understand and implement, making it accessible to both beginners and experts in the field of machine learning.

2. No Training Phase: As a lazy learning algorithm, k-NN does not require an explicit training phase. Instead, it memorizes the entire training dataset, making the prediction process efficient.

3. Flexibility in Handling Features: The k-NN algorithm can handle both audio and non-audio features, making it suitable for incorporating metadata and textual information, which can improve classification accuracy.

4. Interpretability: The decisions made by the k-NN model can be easily interpreted by examining the nearest neighbors and understanding the basis for the classification.

5. Reasonable Accuracy: The proposed solution achieved a respectable accuracy of 74% on the test dataset, demonstrating its potential to classify music tracks into their respective genres with reasonable precision.

6. Fast Prediction: Once the model is trained, the prediction process is fast and efficient, making it suitable for real-time or near real-time applications.

7. Easy Model Updates: The k-NN algorithm allows for easy model updates with the addition of new data points, without requiring a complete retraining of the model.

Disadvantages of the Proposed Solution:

1. High Memory Usage: The k-NN algorithm needs to store the entire training dataset in memory for predictions, which can be computationally expensive and memory-intensive for large datasets.

2. Slow Prediction for Large Datasets: As the number of training samples grows, the prediction time can become slow, making the k-NN algorithm less scalable for massive datasets.

3. Sensitive to Noise and Irrelevant Features: The k-NN algorithm can be sensitive to noisy or irrelevant features, which might lead to inaccurate classifications.

4. Imbalanced Data Impact: The performance of k-NN can be affected by imbalanced data, as it may result in biased predictions towards majority classes and poor classification of minority classes.

5. Difficulty with High-Dimensional Data: In high-dimensional feature spaces, the "curse of dimensionality" can adversely impact k-NN's performance, leading to a drop in accuracy.

6. Limited Generalization: The k-NN algorithm might struggle to generalize well to unseen data, especially if the training dataset does not adequately represent the entire population of music tracks.

7. Hyperparameter Sensitivity: The choice of the k value in k-NN can significantly impact the model's performance, and selecting the optimal value might require additional tuning and experimentation.

Despite these limitations, the proposed solution using the k-NN algorithm serves as an effective starting point for music genre classification. With careful optimization and feature engineering, the model's accuracy can be improved, and it can be used as a stepping stone to explore more advanced machine learning techniques for even better performance.

8. APPLICATIONS

The music genre classification solution using the k-Nearest Neighbors (k-NN) algorithm has various applications in the field of music and beyond. Some of the key areas where this solution can be applied include:

1. Personalized Music Recommendations: Music streaming platforms like Spotify can utilize the genre classification system to offer personalized playlists and music recommendations to users based on their genre preferences.

2. Music Analysis and Insights: The system can be employed for music analysis to gain insights into user listening patterns, popular genres, and the impact of specific genres on user behavior.

3. Music Identification Apps: The model can be integrated into music identification applications (e.g., Shazam) to identify songs and provide information about their genres.

4. Content Tagging and Curation: Content creators and curators can use the system to tag and categorize music tracks in their libraries for more organized and targeted playlists.

5. Genre-Based Music Marketing: Music artists and labels can utilize the system to understand the preferences of their target audience and tailor their marketing strategies accordingly.

6. Radio Stations and Playlists: Radio stations and playlist creators can leverage the model to categorize and organize music tracks into genre-specific broadcasts and playlists.

7. Music Licensing and Royalty Management: The system can assist in managing music royalties by accurately categorizing tracks and ensuring the appropriate royalty payments based on their genres.

8. Music Research and Academics: The solution can be valuable in academic research and studies related to music trends, genre popularity, and the impact of music on cultural patterns.

9. Recommendation Engines for Other Media: The concept of genre classification can be extended to other media types like movies, books, or articles, providing relevant recommendations to users based on their preferences.

10. Interactive Music Entertainment: The system can be integrated into interactive music applications or games, where the music changes dynamically based on the user's preferences and actions.

11. Automated Music Labeling and Tagging: Music databases and platforms can use the solution to automatically label and tag new music uploads with appropriate genres.

12. Music Therapy and Mood Enhancement: In healthcare and wellness applications, the solution can be used to create personalized music playlists for mood enhancement and music therapy.

The versatility of the music genre classification system enables its integration into various applications, ranging from improving user experience on music platforms to enhancing content curation and understanding music trends in different contexts. As technology and data continue to evolve, the potential applications of this solution are likely to expand even further.

9. CONCLUSION

The music genre classification project using the k-Nearest Neighbors (k-NN) algorithm aimed to develop an efficient and accurate system capable of categorizing music tracks into their respective genres. The project involved extensive data collection, feature extraction, model training, evaluation, and optimization to achieve the desired results. Here is a summary of the entire work and findings:

1. Objective: The primary objective of the project was to implement a music genre classification system using machine learning techniques, with a focus on the k-NN algorithm, to offer personalized music recommendations and enhance user experience on music streaming platforms.

2. Methodology: The proposed solution combined audio features (e.g., MFCCs, chroma) and textual features (e.g., song titles, artist names) to create a comprehensive representation of music tracks. The k-NN algorithm, a simple and interpretable classification method, was employed to predict the genres of the tracks.

3. Data Collection and Preprocessing: A dataset containing 1000 music tracks from various genres, along with their corresponding audio features and metadata, was collected from Spotify. The data underwent thorough preprocessing to ensure uniformity and eliminate noise.

4. Feature Extraction: Audio features were extracted from the raw audio signals, and textual features were processed using natural language processing techniques, resulting in a rich feature representation for each track.

5. Model Training and Evaluation: The k-NN model was trained on the dataset, and its performance was evaluated using metrics like accuracy, precision, recall, and F1 score. The model achieved a classification accuracy of 74% on the test dataset.

6. Advantages: The k-NN algorithm offered simplicity, interpretability, and flexibility in handling various types of features. The model's predictions were reasonably accurate, enabling personalized music recommendations and genre-based insights.

7. Limitations and Challenges: The proposed solution faced challenges related to data imbalance, sensitivity to irrelevant features, and memory usage for large datasets. High-dimensional data could also affect the model's performance.

8. Applications: The music genre classification system had a wide range of applications, including personalized music recommendations, music analysis, music identification apps, and music marketing.

9. Conclusion: In conclusion, the music genre classification project successfully demonstrated the potential of using the k-NN algorithm for accurate music genre classification. While achieving a satisfactory accuracy of 74%, the solution also highlighted the importance of data preprocessing, feature engineering, and hyperparameter tuning in enhancing classification performance. The project's findings paved the way for future enhancements, such as integrating more advanced machine learning models, handling large datasets, and improving the generalization of the system to diverse music tracks.

Overall, the developed music genre classification system contributes to the field of music information retrieval and has the potential to improve user experiences, music recommendations, and music analysis across various music platforms and applications. As technology advances and more data becomes available, further research and development can lead to even more sophisticated and accurate music genre classification solutions.

10. FUTURE SCOPE

The music genre classification project using the k-Nearest Neighbors (k-NN) algorithm has laid a solid foundation for accurate genre classification. However, there are several avenues for future enhancements and improvements. Here are some potential directions for further development:

1. Advanced Machine Learning Models: Experiment with more advanced machine learning models such as deep neural networks (e.g., CNNs or RNNs) and ensemble methods (e.g., Random Forests, Gradient Boosting) to explore their potential in improving classification accuracy.

2. Data Augmentation: Apply data augmentation techniques to increase the diversity of the training dataset, especially for genres with limited samples. This can help in addressing data imbalances and improve model generalization.

3. Transfer Learning: Investigate the use of transfer learning by pretraining models on a large-scale music dataset or related tasks (e.g., music genre prediction on other platforms) to leverage learned representations for the current problem.

4. Feature Engineering: Explore additional audio and textual features that might better capture genre-specific characteristics and improve the model's discriminatory power.

5. Handling High-Dimensional Data: Investigate dimensionality reduction techniques (e.g., PCA, t-SNE) to reduce the dimensionality of feature spaces and mitigate the "curse of dimensionality."

6. Hybrid Models: Develop hybrid models that combine the strengths of different machine learning algorithms, such as combining k-NN with deep learning models or using ensemble methods.

7. Fine-tuning Hyperparameters: Continue fine-tuning hyperparameters to find the best combination of parameters for the k-NN algorithm or other models.

8. Dynamic Genre Adaptation: Implement a dynamic genre adaptation mechanism that can adapt the model's predictions based on user feedback and evolving music trends.

9. Streaming Data Support: Explore methods to handle streaming data, enabling real-time classification and continuous updates to the model.

10. Cross-Platform Integration: Extend the solution to work across different music platforms and applications, providing consistent genre predictions regardless of the source.

11. User Feedback Incorporation: Incorporate user feedback into the model to refine genre predictions and improve personalized recommendations.

12. Multimodal Learning: Combine audio, textual, and possibly visual features (e.g., album cover images) using multimodal learning techniques to capture more comprehensive representations of music tracks.

13. Genre Hierarchy: Consider hierarchical genre classification to account for sub-genres and more fine-grained categorization.

14. Incremental Learning: Implement incremental learning techniques to update the model without retraining from scratch when new data becomes available.

15. Real-time Scalability: Design the solution to handle large-scale datasets and ensure real-time scalability for efficient deployment.

By exploring these future enhancements, the music genre classification system can be further refined, offering even better accuracy and performance in music classification. As the field of music analysis and machine learning progresses, these improvements can contribute to a more personalized and enjoyable music experience for users while enabling valuable insights and trends analysis for music platforms and industry stakeholders.