

DATA INTENSIVE COMPUTING PROJECT REPORT

Made By

Shubham Soni, UBID - 50593888

Hazel Mahajan, UBID - 50592568

Poojan Kaneriya, UBID- 50604221

ABSTRACT

The project aims to analyze song data to uncover trends in song popularity and genre classification. By examining various characteristics such as danceability, energy, tempo, and duration, the analysis will explore the relationship between these features and the success of songs across different genres. The objective is to discover patterns and trends that can help predict song popularity and determine genre characteristics, providing valuable insights for music producers, listeners, and the broader music industry.

CONTRIBUTION TO THE PROBLEM DOMAIN

This project has the potential to provide actionable insights into the dynamics of the music industry, revealing how certain audio features influence song success across genres. By analyzing trends in song popularity, it can help artists and producers understand what makes a song popular in different contexts, improving their ability to create hits. Additionally, uncovering genre-specific patterns can assist in automating genre classification and playlist generation, enhancing the user experience on music streaming platforms.

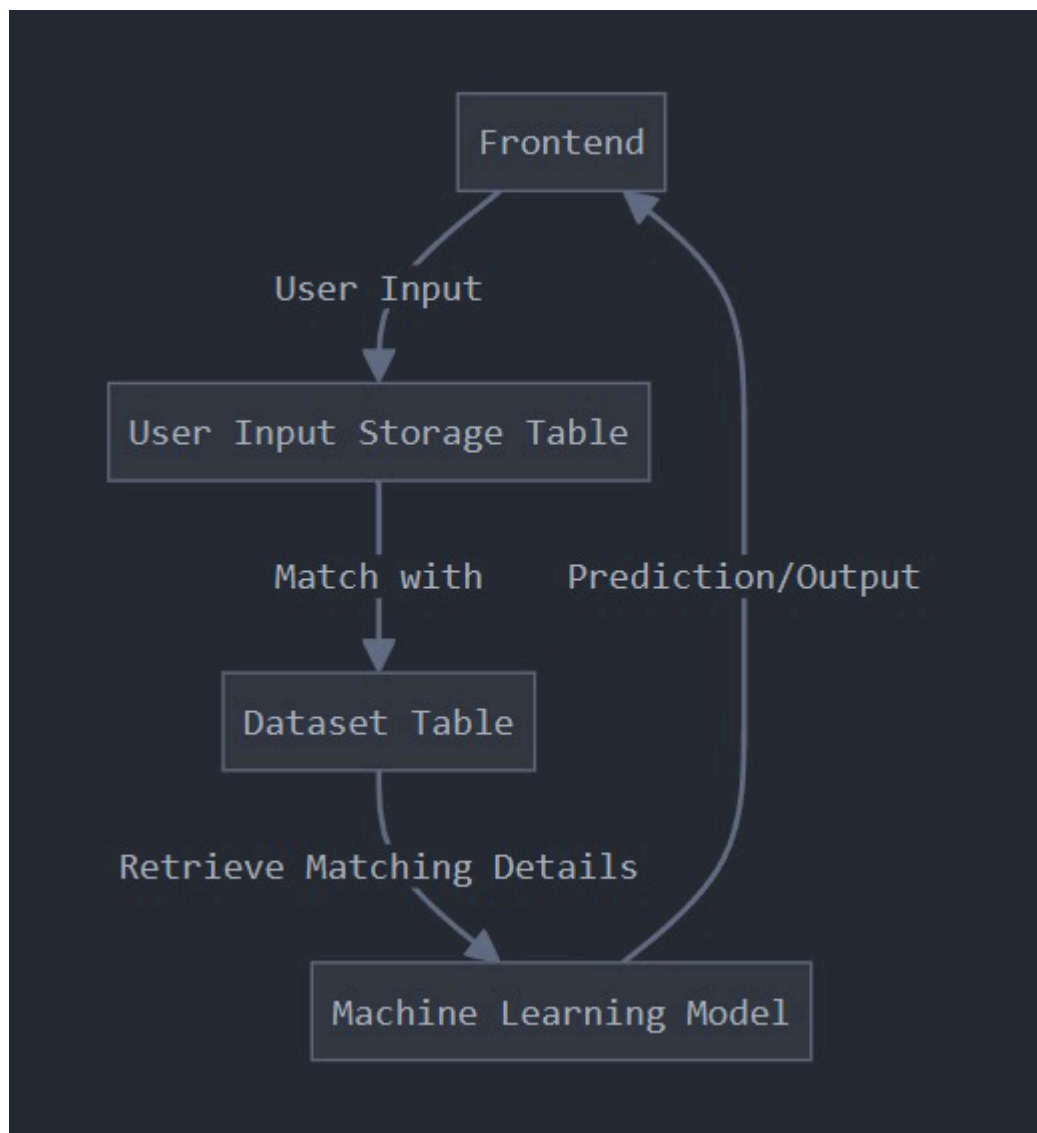
Why this Contribution is Crucial: The contribution of this project is crucial for multiple reasons:

1. **Data-Driven Music Production:** Understanding trends and popular audio features will help artists and producers tailor their creations to meet listener preferences, increasing the likelihood of success.
2. **Genre Evolution Tracking:** The music industry is constantly evolving. Identifying how genres have changed and predicting future genre trends is vital for industry stakeholders to stay relevant and ahead of consumer demand.
3. **Improved Recommendation Systems:** By understanding how specific features affect genre classification and popularity, music streaming platforms can

improve recommendation algorithms, leading to more personalized user experiences.

This analysis will bridge the gap between raw musical data and actionable insights for various stakeholders in the music ecosystem.

APPLICATION FLOW



APPLICATION USE CASES

- Genre Prediction
- Popularity Prediction
- Mood Prediction
- Similar Song Suggestion

EXPLORATORY DATA ANALYSIS STEPS

1. Data Collection and Loading:

- Gather the dataset containing song features (e.g., tempo, key, loudness, duration) and associated genres.
- Load the dataset using libraries like Pandas or Numpy.

2. Initial Data Inspection:

- Explore the dataset structure, check column names, data types, and overall size.
- Use `.head()`, `.info()`, and `.describe()` to understand the data at a glance.

3. Data Cleaning:

- Identify and handle missing values (e.g., filling with mean/median or dropping).
- Remove or replace duplicate rows, if any.

4. Feature Understanding and Univariate Analysis:

- Analyze each feature individually using descriptive statistics and visualizations (e.g., histograms, boxplots).
- Look for patterns or irregularities in song features like tempo, key, and loudness.

5. Target Variable Exploration:

- Investigate the distribution of song genres using value counts and visual tools like bar plots or pie charts.
- Check for class imbalance in the target variable.

6. Correlation Analysis:

- Compute the correlation matrix to identify relationships between features.
- Visualize with a heatmap to detect highly correlated pairs or redundant features.

7. Multivariate Analysis:

- Analyze relationships between features using pair plots, scatter plots, or violin plots.
- Observe how features vary across different genres.

8. Outlier Detection and Treatment:

- Use boxplots, z-scores, or interquartile ranges (IQR) to identify outliers in features like tempo or duration.
- Decide whether to retain, transform, or remove these outliers.

9. Feature Engineering and Scaling:

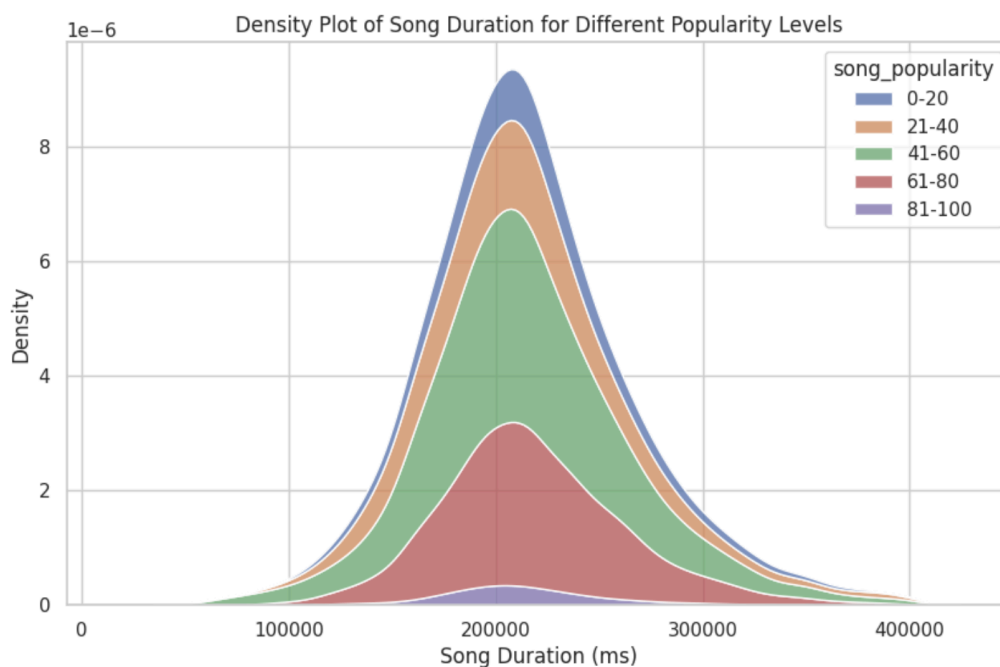
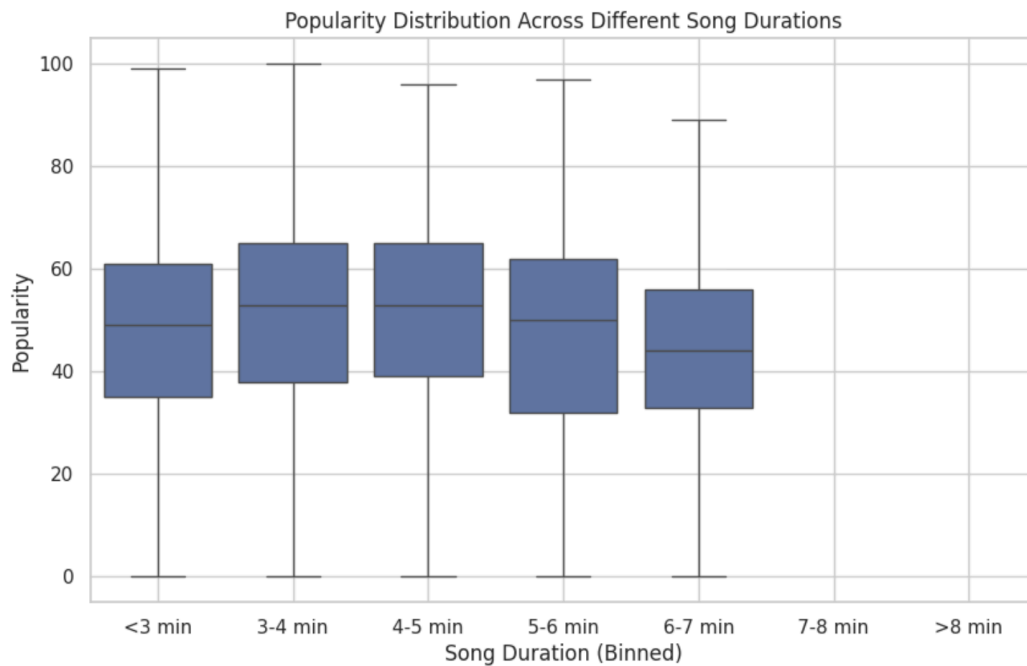
- Create new features if needed (e.g., tempo per minute, loudness scaled by duration).
- Normalize or standardize features for models that are sensitive to scaling.

10. Feature Selection and Insights:

- Identify important features contributing to genre differentiation using statistical methods or domain knowledge.
- Summarize key findings and prepare the dataset for model building.

ANALYSIS OF HYPOTHESIS FORMED

Hypothesis : The duration of a song (length in milliseconds) significantly affects its popularity

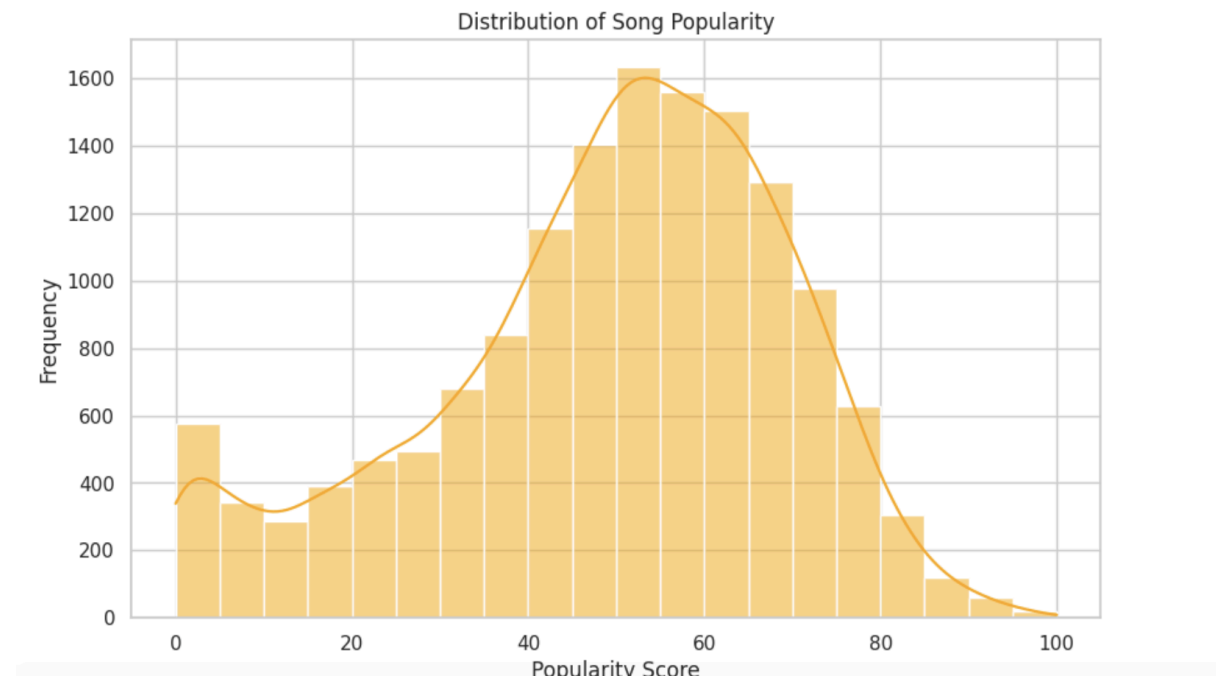


Analysis of results:

The EDA shows a weak correlation between song duration and popularity, with no significant trend observed in the scatter, box, or density plots.

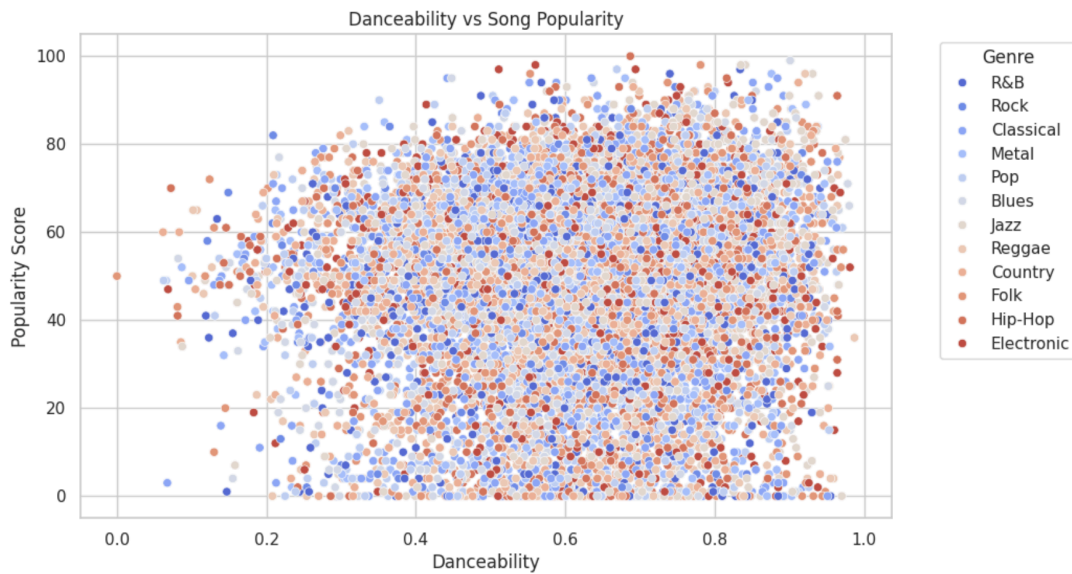
The low R-squared value from the regression analysis indicates that song duration does not significantly impact popularity, suggesting that song length alone is not an Armstrong predictor of its success

Hypothesis: Certain genres are consistently more popular than others.



Analysis of results : The ANOVA test reveals a statistically significant difference in popularity between genres, supporting the hypothesis that "specific genres consistently have higher average popularity scores" based on the data. Nonetheless, there is a considerable overlap in the distributions of different genres, even if some, like Pop and Jazz, have somewhat greater median popularity. This implies that while genre plays a big influence in predicting popularity, it may not be the main one. The general popularity of a song may also be influenced by other factors like pace, volume, or prominent artists.

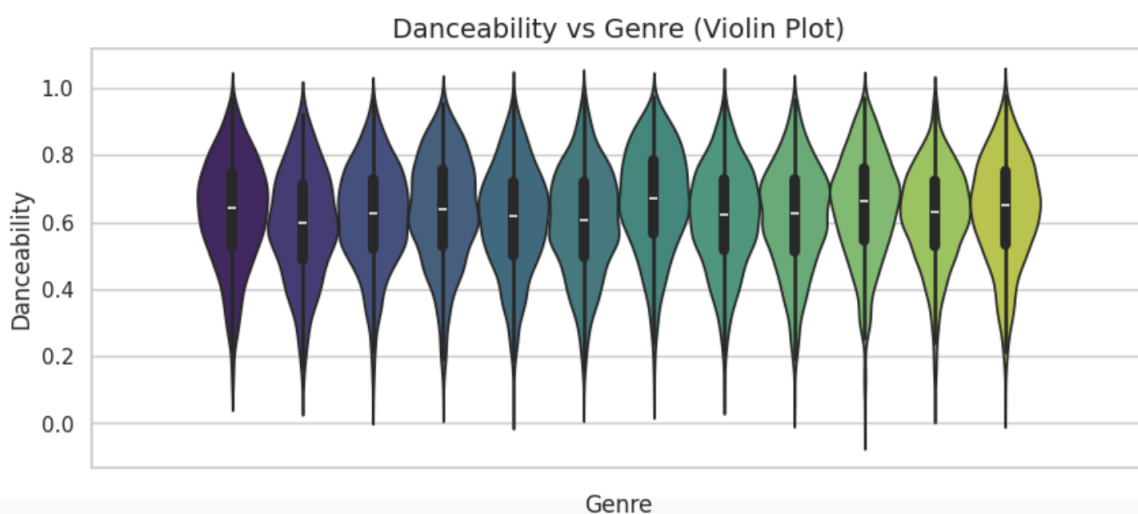
Hypothesis: Songs with higher danceability, energy, or tempo tend to have higher popularity ratings.



Analysis of results :

The data does not provide substantial evidence for the hypothesis that "songs with higher danceability, energy, or tempo tend to have higher popularity ratings". Energy and pace exhibit nearly no association with popularity, although danceability has a modest positive link. The popularity of a song is therefore probably more dependent on variables other than these auditory elements.

Hypothesis: Different music genres tend to exhibit distinct characteristics in terms of valence (musical positivity) and danceability.



Analysis of results:

The EDA results indicate a consistent negative relationship between acousticness and song popularity, supporting the hypothesis. Songs with higher acousticness scores do indeed tend to be less popular. Therefore, the hypothesis is true based on the EDA findings.

MACHINE LEARNING MODELS - UTILISED IN THE APPLICATION

Model 1. Genre Prediction

What are we doing?

We are predicting the **genre** of a song based on its audio features such as danceability, energy, valence, and more.

How are we doing it?

1. **Input:** Song name.
2. **Data Preparation:**
 - The target variable is genre (categorical).
 - Features include numerical attributes like danceability, energy, and others.
 - Genres are encoded as numerical labels.
3. **Model:**
 - A **Random Forest Classifier** is trained on the dataset to classify songs into genres.
4. **Prediction:**
 - When a user selects a song, its features are extracted.
 - The trained model predicts the genre based on these features.

Why is this useful? It helps users quickly identify the genre of a song, especially when they are exploring music and looking for recommendations.

Model 2. Mood Prediction

What are we doing?

We are predicting the **mood** of a song, such as "Happy," "Sad," "Energetic," or "Calm," based on its audio features.

How are we doing it?

1. **Input:** Song name.
2. **Data Preparation:**
 - Moods are defined based on features like valence, tempo, and danceability.
 - For example:
 - High valence and danceability = **Happy**.
 - Low valence and tempo < 100 = **Sad**.
 - High energy and tempo > 120 = **Energetic**.
 - Otherwise, **Calm**.
 - Moods are encoded as numerical labels.
3. **Model:**
 - A **Random Forest Classifier** is trained on the dataset with mood categories as the target variable.
4. **Prediction:**
 - For a given song, its features are extracted, and the model predicts the corresponding mood.

Why is this useful?

It enhances user engagement by helping them explore music based on emotional states and creating mood-based playlists.

Model 3. Similar Song Prediction

What are we doing?

We are generating a **playlist of songs** similar to a selected song based on its features.

How are we doing it?

1. **Input:** Song name and the number of similar songs to suggest.
2. **Data Preparation:**
 - Features include numerical attributes like acousticness, danceability, energy, etc.
3. **Similarity Calculation:**
 - **Cosine similarity** is used to compare the feature vector of the selected song with all other songs in the dataset.
 - Higher similarity scores indicate greater similarity.
4. **Recommendation:**
 - The songs are sorted by similarity score, and the top results (excluding the input song) are displayed as the playlist.

Why is this useful?

It helps users discover new songs that match the vibe or characteristics of their favorite tracks, enhancing music exploration.

Summary of Approaches

Model	Target	Algorithm/Approach	Input	Output
Genre Prediction	Genre (categorical)	Random Forest Classifier	Song name	Predicted genre
Mood Prediction	Mood (categorical)	Random Forest Classifier	Song name	Predicted mood
Similar Song Prediction	Similarity (cosine score)	Cosine Similarity	Song name, num results	Playlist of similar songs

Model 4: Popularity Prediction

What are we doing?

predicting the popularity of a song on a scale of 0–100, where higher scores indicate greater listener engagement and appeal.

How are we doing it?

1. Input - Song name: The user provides the name of the song for which popularity needs to be predicted.

2. Data Preparation:

Target Variable - The popularity score is derived from metrics such as the number of streams, play duration, and listener saves.

Features: Song audio features like acoustic ness, danceability, energy, tempo, and loudness are used as predictors.
Preprocessing: Songs with duplicate names are removed to ensure unique records.

Features are cleaned and checked for missing or inconsistent values.

3. Model: A Random Forest Regressor is trained on the dataset with `song_popularity` as the target variable. The model learns the relationships between audio features and popularity scores from historical data.

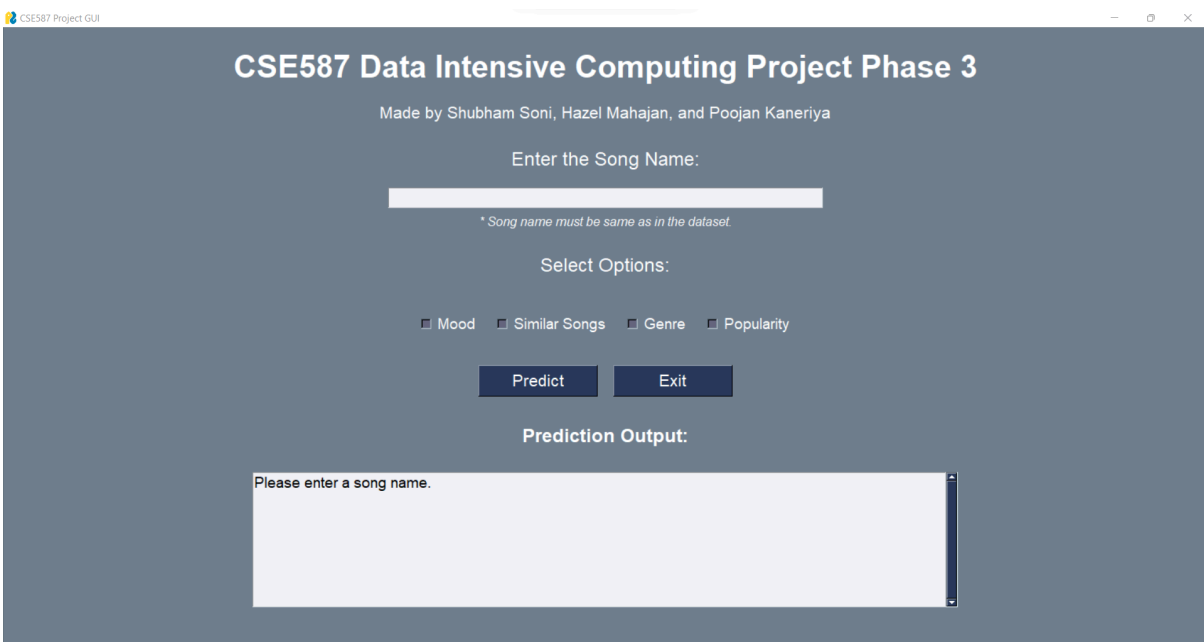
4. Prediction: For a given song, its features are extracted, and the trained model predicts its popularity score on a scale of 0–100.

RESULTS

app.py contains the source code to run the application

```
PS C:\Users\shubh\OneDrive\Desktop\DIC Project\src> python app.py
Initializing database...
Database initialized.
Initializing models...
Models initialized.
```

Edge Cases



CSE587 Project GUI

CSE587 Data Intensive Computing Project Phase 3

Made by Shubham Soni, Hazel Mahajan, and Poojan Kaneriya

Enter the Song Name:

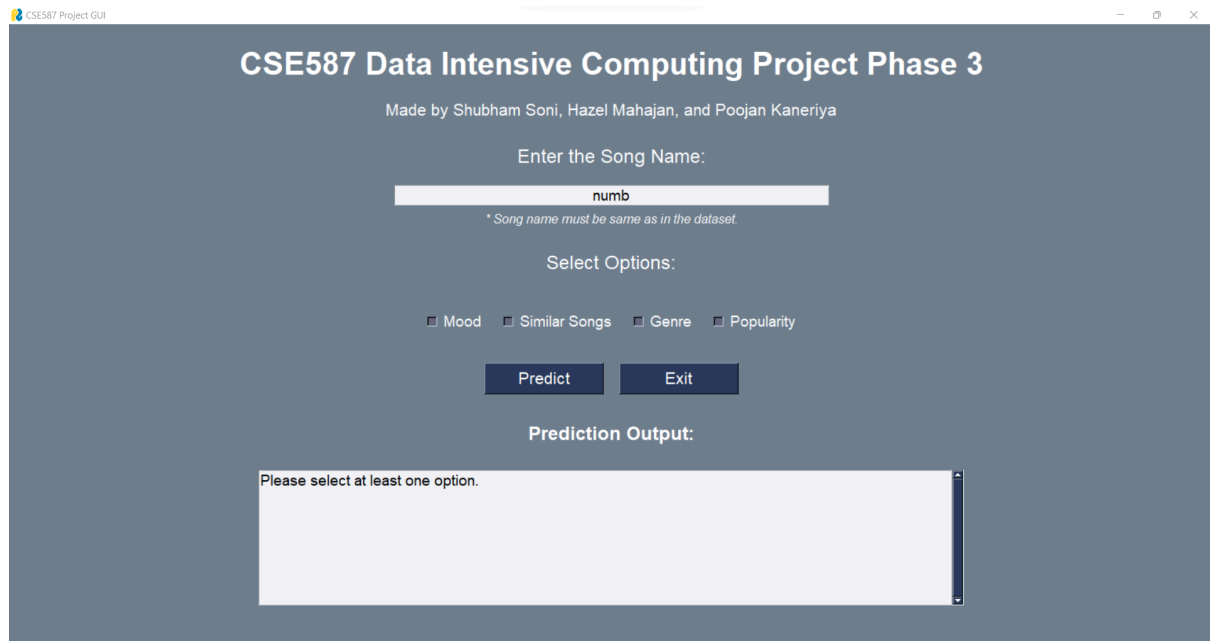
* Song name must be same as in the dataset.

Select Options:

☐ Mood ☐ Similar Songs ☐ Genre ☐ Popularity

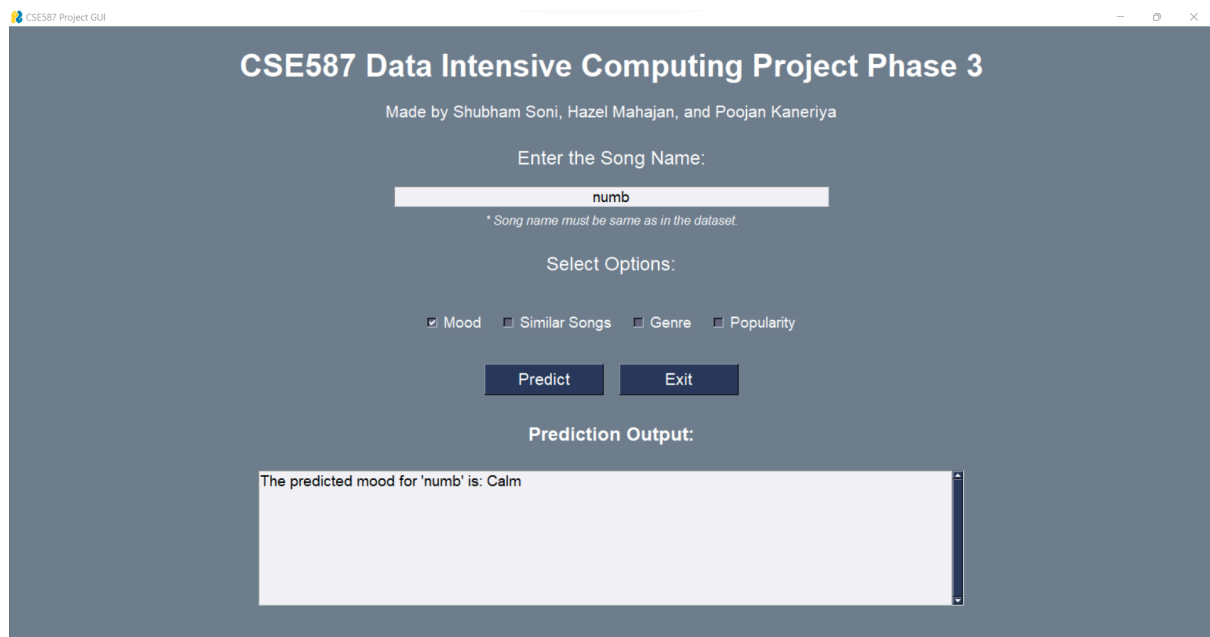
Prediction Output:

Please enter a song name.



Single SelectionApp Execution

Executed use case - Mood Prediction



Executed use case - Similar Song Suggestions

CSE587 Project GUI

CSE587 Data Intensive Computing Project Phase 3

Made by Shubham Soni, Hazel Mahajan, and Poojan Kaneriya

Enter the Song Name:

numb

* Song name must be same as in the dataset.

Select Options:

☐ Mood ☒ Similar Songs ☐ Genre ☐ Popularity

Predict Exit

Prediction Output:

Suggested Songs:
Swerve City
Life To Fix - Radio Edit
Déjà vu
Young Again
Dirty Little Secret

Executed use case - Genre Prediction

CSE587 Project GUI

CSE587 Data Intensive Computing Project Phase 3

Made by Shubham Soni, Hazel Mahajan, and Poojan Kaneriya

Enter the Song Name:

numb

* Song name must be same as in the dataset.

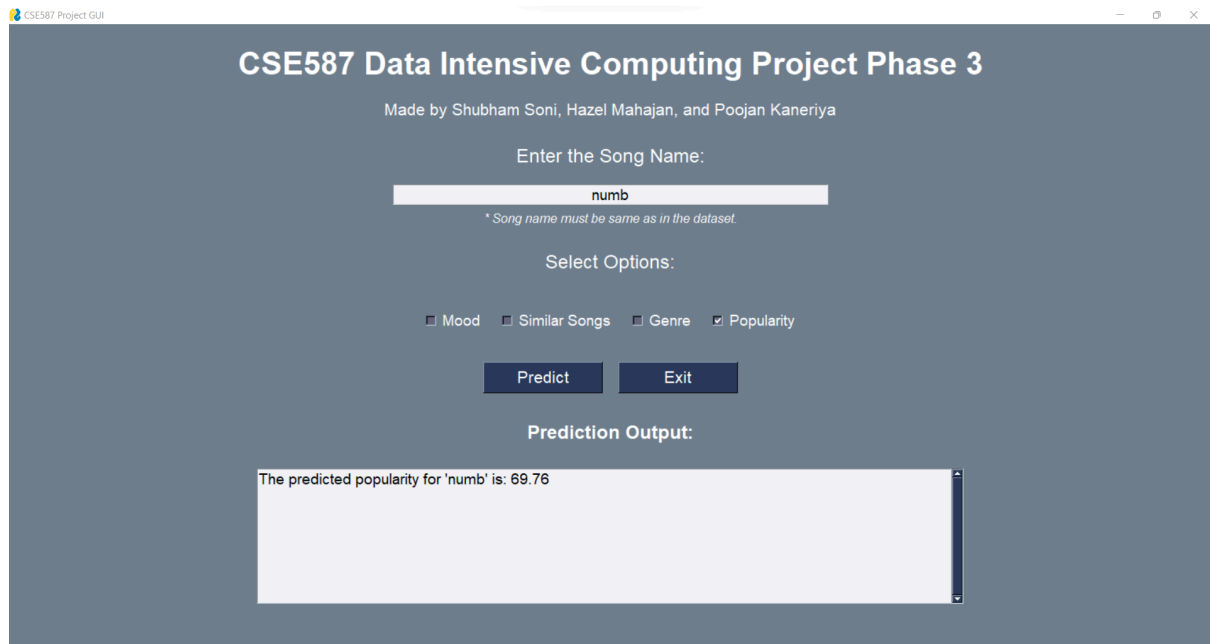
Select Options:

☐ Mood ☐ Similar Songs ☒ Genre ☐ Popularity

Predict Exit

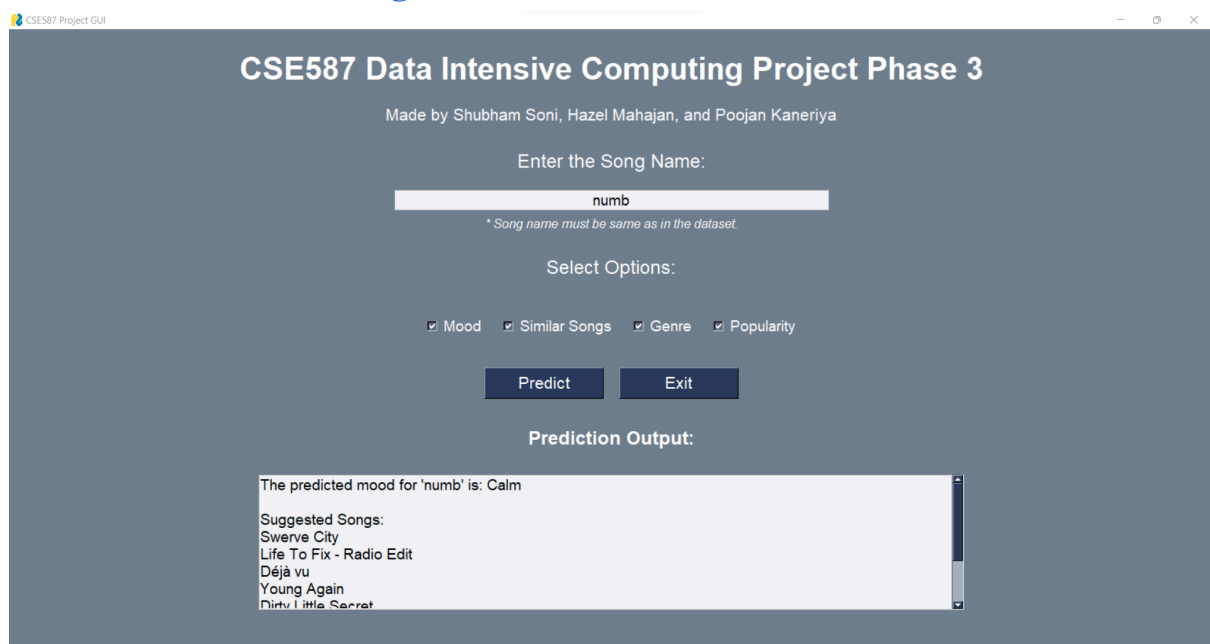
Prediction Output:

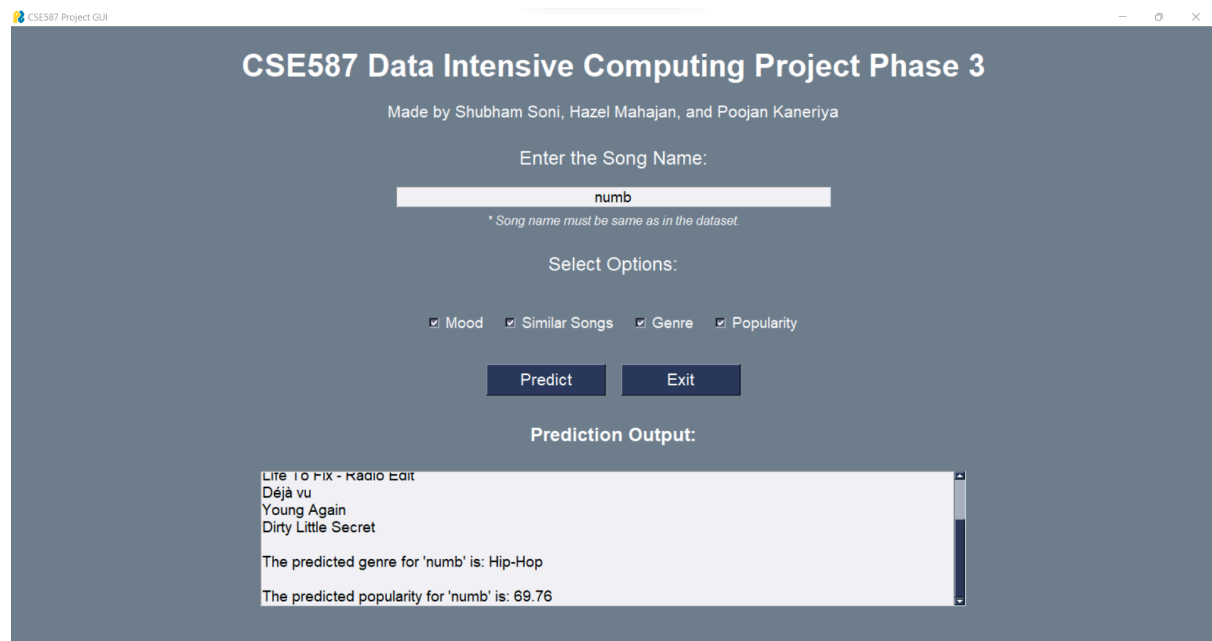
The predicted genre for 'numb' is: Hip-Hop



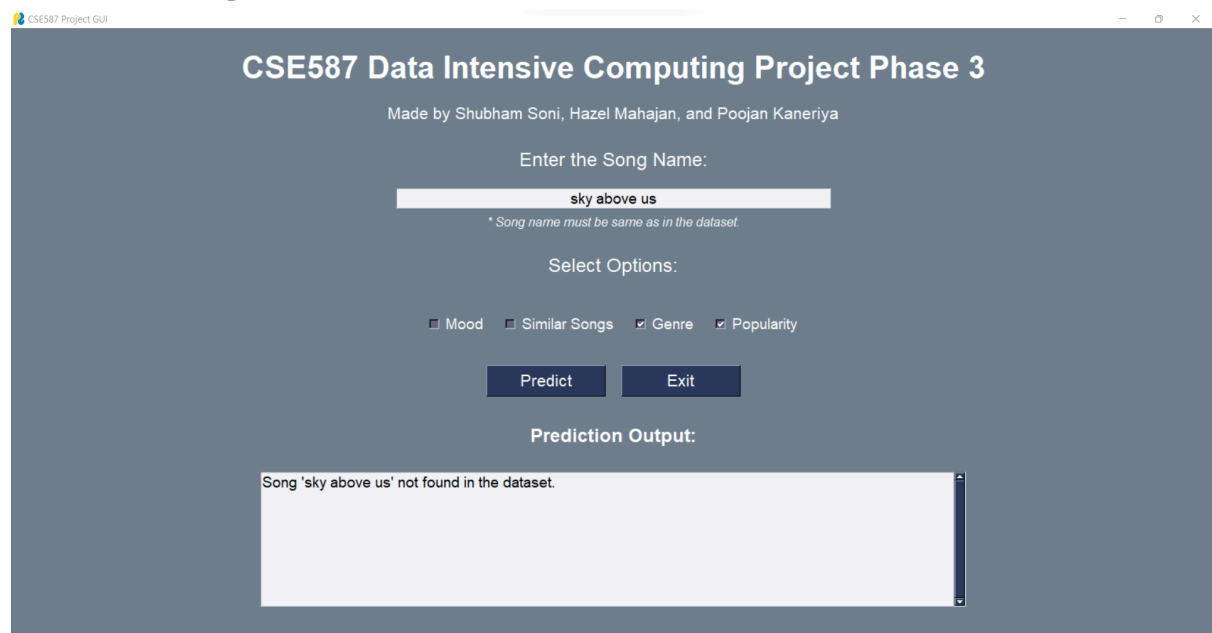
Multi Selection Execution

All 4 use cases executed together





Error Handling



FUTURE WORK

The future scope of this song analysis application can be expanded significantly to cater to more advanced use cases and a broader audience. Here are some potential directions:

1. Feature Enhancements

Advanced Recommendation System:

- Incorporate collaborative filtering or hybrid recommendation systems to provide more personalized song recommendations.
- Allow users to create dynamic playlists based on their preferences.

Multi-Language Support:

- Expand the dataset to include songs in multiple languages and allow the application to handle diverse genres and cultural music preferences.

Lyrics-Based Analysis:

- Integrate Natural Language Processing (NLP) to analyze song lyrics for sentiment, themes, or mood predictions.

Visualization Tools:

- Provide visualizations like mood graphs, genre trends, and popularity heatmaps for user engagement.

Voice Command Integration:

- Enable voice-based input for song names or commands using speech recognition libraries like SpeechRecognition.

2. User Personalization

User Profiles:

- Allow users to create profiles, save their preferences, and access historical predictions.
- Use these profiles for enhanced recommendations.

Music Sentiment Tracking:

- Enable users to upload or select songs to track their mood over time and suggest music accordingly.

Integration with Music Platforms:

- Integrate APIs from Spotify, YouTube, or Apple Music to directly fetch playlists or play songs.

3. AI and ML Enhancements

Real-Time Predictions:

- Implement real-time analysis and predictions using streaming data from live music platforms.

Improved Models:

- Use deep learning models like Convolutional Neural Networks (CNNs) for audio feature extraction and improved predictions.

Context-Aware Predictions:

- Introduce contextual features like time of day, weather, or user mood to influence song recommendations.

Generative Music Models:

- Integrate AI models like OpenAI's Jukebox to generate music recommendations or even compose new music based on user preferences.

4. Scalability and Platform Expansion

Mobile Application:

- Develop a mobile version of the application for Android and iOS for better accessibility.

Web-Based Deployment:

- Deploy the application on a web platform using frameworks like Flask, Django, or FastAPI.

Cloud Integration:

- Host the application on cloud platforms (AWS, Azure, GCP) for better scalability and multi-user support.

Cross-Platform Synchronization:

- Synchronize user data across devices using cloud storage or database solutions.

5. Research and Analytics

Music Industry Analytics:

- Provide insights into genre trends, popularity metrics, and audience preferences for music producers or platforms.

Music Therapy Applications:

- Collaborate with healthcare professionals to suggest music for mental well-being based on mood predictions.

Dataset Expansion:

- Build and use larger datasets with audio features, lyrics, and user feedback to improve accuracy and relevance.

Conclusion

This application has immense potential in the fields of music technology, data science, and entertainment. By incorporating these future enhancements, it can evolve into a sophisticated platform catering to music enthusiasts, researchers, and industry professionals.

REFERENCES

1. PySimpleGUI. (n.d.). PySimpleGUI Official Website. Retrieved from https://pysimplegui.com
2. OpenAI. (n.d.). ChatGPT: Conversations with AI. Retrieved from https://openai.com/chatgpt
3. Python Software Foundation. (n.d.). Python Downloads. Retrieved from https://www.python.org/downloads/
4. Kaggle. (n.d.). Kaggle: Your Machine Learning and Data Science Community. Retrieved from https://www.kaggle.com