A

SYNOPSIS

of

MINOR PROJECT

on

Classify Song Genres from Audio Data

तमसो मा ज्योतिर्गमय्

Darkness to Light

Submitted by

Rishabh Chaudhary (22EGICS119)

Project Guide **Head of Department**
Mr. Sandeep Bordia **Dr. Mayank Patel**

Geetanjali Institute of Technical Studies, Dabok , Udaipur (Raj.)
Department of Computer Science and Engineering
October,2023

Problem Statement: The classification of music into genres is essential for music lovers and streaming services to organize playlists and recommend tracks. Developing a machine learning model to classify the genre of a song based on various attributes will streamline this process.

Brief Description: In this guide, we aim to build, evaluate, and enhance machine learning models to classify the genre of songs. Music genres, such as hip-hop, rock, country, and pop, are defined by attributes like valence, liveness, and acousticness. Categorizing music by genre enables the creation of personalized playlists and enhances the recommendation systems of music streaming services. We will start with logistic regression and XGBoost models, compare their performance using log-loss as the evaluation metric, and then proceed to improve the model's performance through hyperparameter tuning. Finally, we will identify the most influential features using SHAP values and explore methods to generalize and optimize our model.

Objective and Scope:

Objective

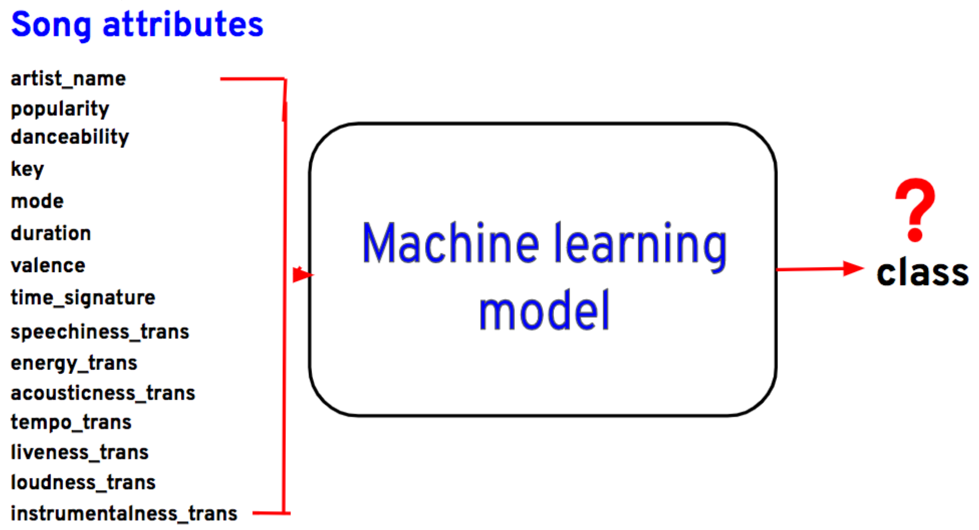
- To build a machine learning model that can classify the genre of songs accurately.
- To evaluate and compare different models to identify the best-performing one.
- To improve the model's performance and generalization for real-world applications.

Scope

- Utilize a public dataset from Kaggle containing multiple attributes of songs.
- Implement data exploration, cleaning, and analysis.
- Apply feature engineering techniques to prepare the data.
- Train and evaluate the model using various machine learning algorithms.
- Optimize the model for improved accuracy.

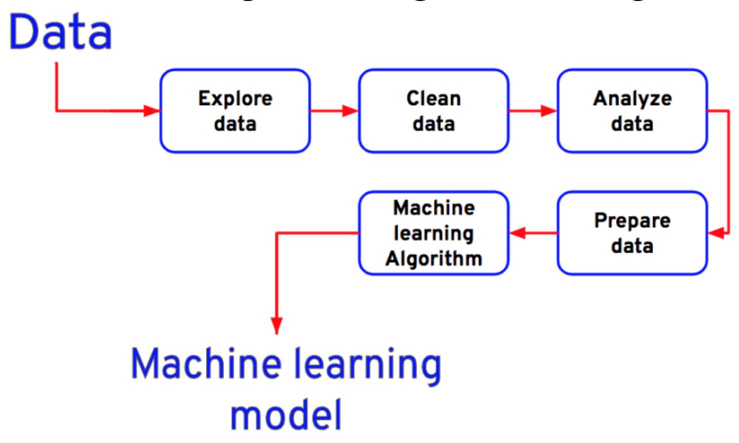
Methodology:

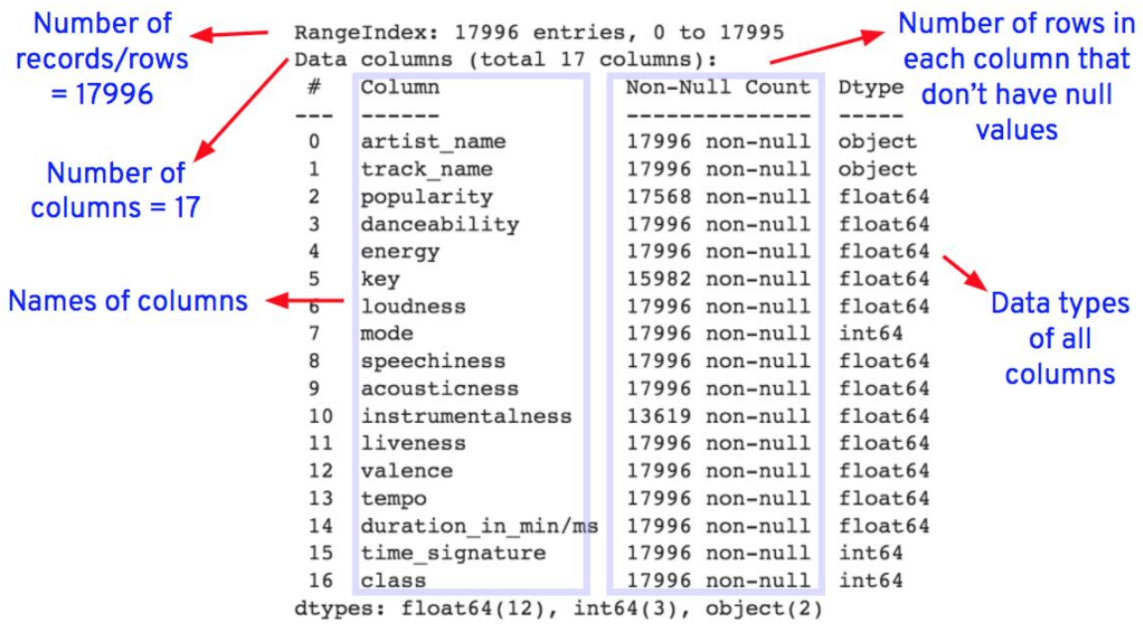
- **Import Libraries**: Use Python libraries such as Pandas, Numpy, Matplotlib, Seaborn, and Scikit-learn.



- **Data Preparation**:
  - Load the dataset, examine the structure, and understand the data types and target variables.
  - Edit column names, remove duplicates, and handle missing values.

- Scale the training and testing datasets using StandardScaler.



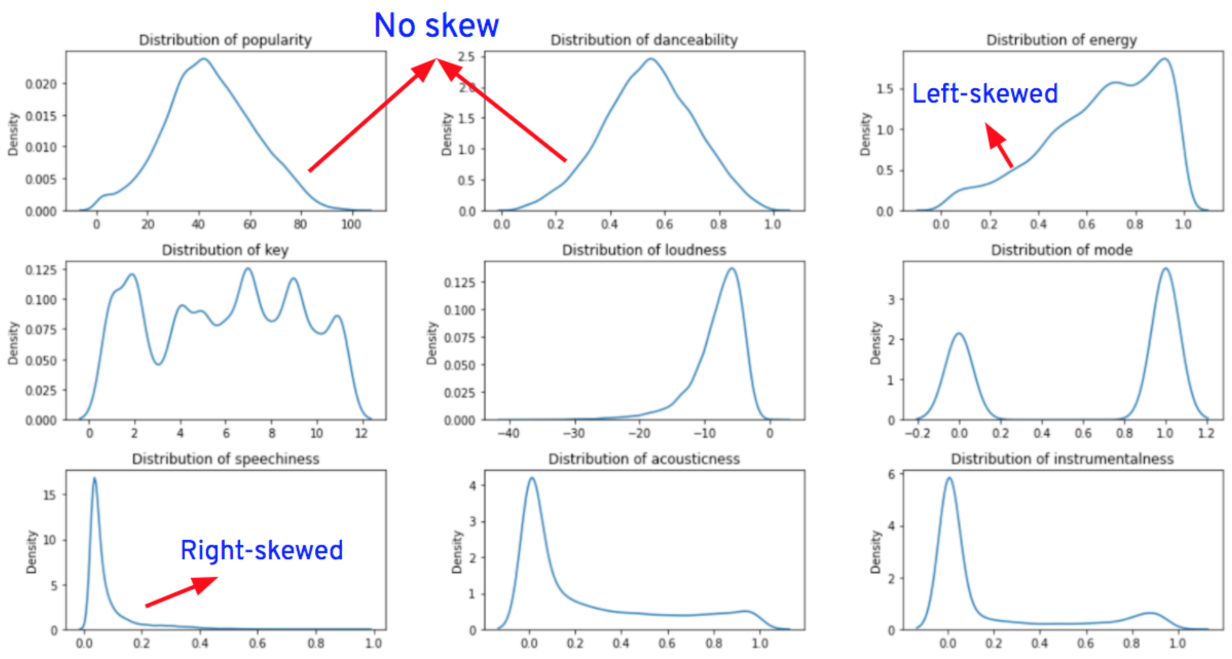

Number of records/rows = 17996
Number of columns = 17
Names of columns
Number of rows in each column that don't have null values
Data types of all columns

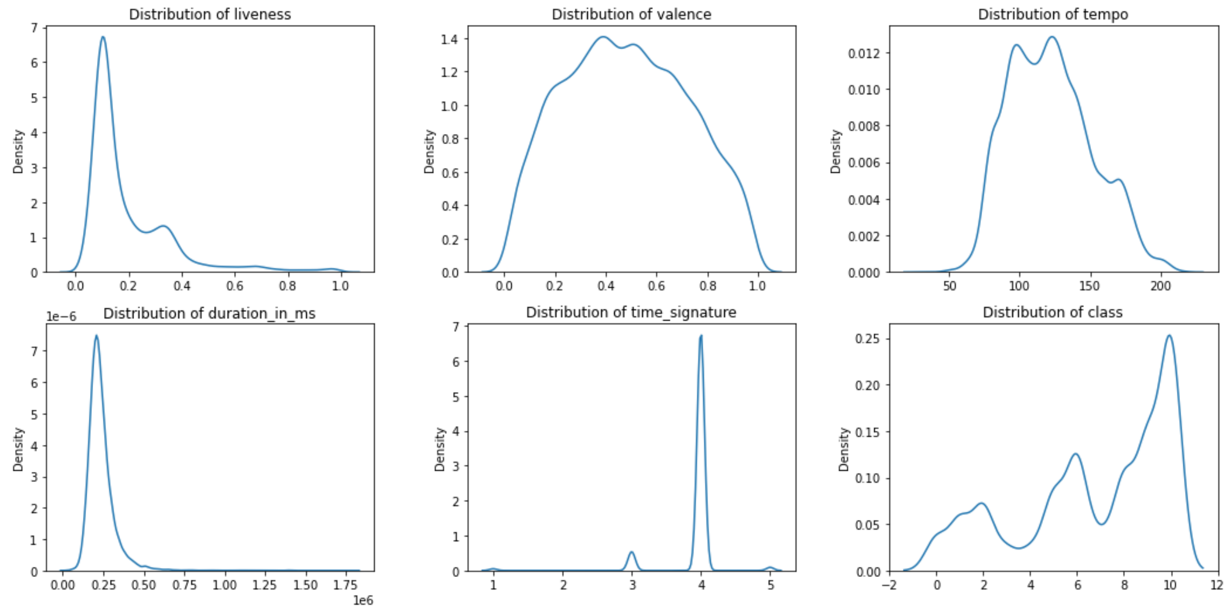
```
RangeIndex: 17996 entries, 0 to 17995
Data columns (total 17 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   artist_name         17996 non-null  object
 1   track_name          17996 non-null  object
 2   popularity          17568 non-null  float64
 3   danceability        17996 non-null  float64
 4   energy              17996 non-null  float64
 5   key                 15982 non-null  float64
 6   loudness            17996 non-null  float64
 7   mode                17996 non-null  int64
 8   speechiness         17996 non-null  float64
 9   acousticness        17996 non-null  float64
 10  instrumentalness    13619 non-null  float64
 11  liveness            17996 non-null  float64
 12  valence             17996 non-null  float64
 13  tempo               17996 non-null  float64
 14  duration_in_min/ms  17996 non-null  float64
 15  time_signature      17996 non-null  int64
 16  class               17996 non-null  int64
dtypes: float64(12), int64(3), object(2)
```

Data info.

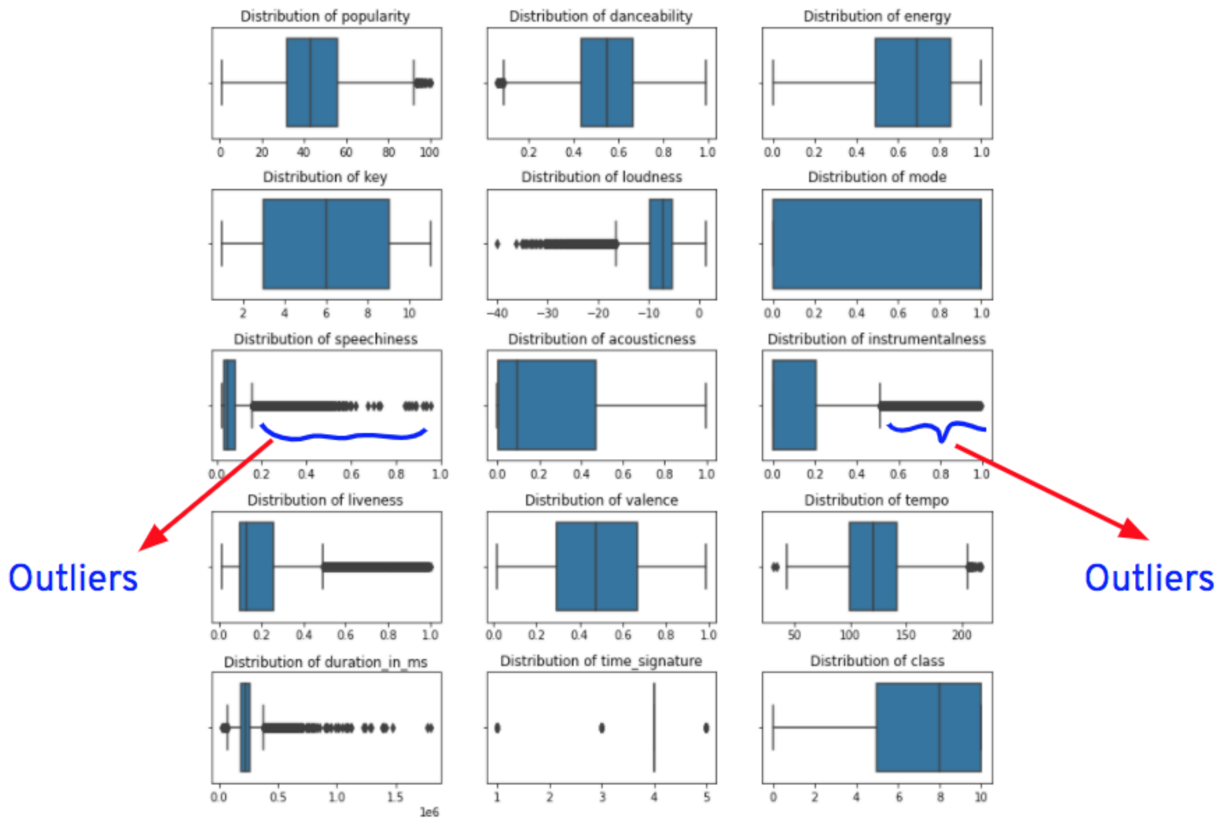
- **Model Building**:
  - Train logistic regression and XGBoost models on the standardized data.

- Split the data into training and test sets, standardize/normalize the data, and train the model using algorithms like Logistic Regression, Decision Tree, and Random Forest.





- **Model Evaluation**:
    - Evaluate models using log-loss as the primary metric.
    - Compare log-loss values to select the best model.
    - Use accuracy and other relevant metrics to assess model performance.



- **Generalization Check**:
    - Evaluate log-loss on training and test data to check for overfitting or underfitting.

- **Performance Improvement**:
    - Perform hyperparameter tuning to optimize the XGBoost model.

- **Feature Importance**:
    - Use the SHAP library to identify the most influential features.
- **Model Retraining**:
    - Retrain the model with optimized features and parameters.
- **Predictive System**:
    - Implement a system to predict music genres on real-world data.

Hardware and Software Requirements:

Hardware

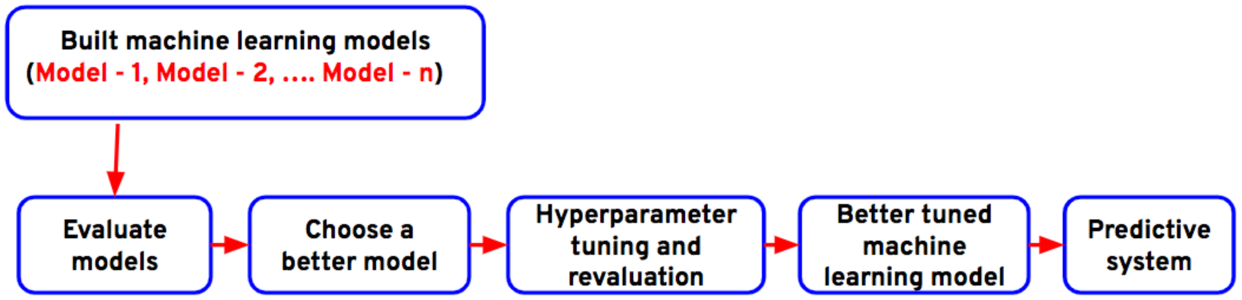
- Processor: Intel Core i5 or higher
- RAM: 8 GB or more
- Storage: 50 GB of free disk space

Software

- Operating System: Windows, macOS, or Linux
- Python 3.7 or higher
- Jupyter Notebook
- Libraries: Pandas, Numpy, Matplotlib, Seaborn, Scikit-learn, XGBoost, SHAP

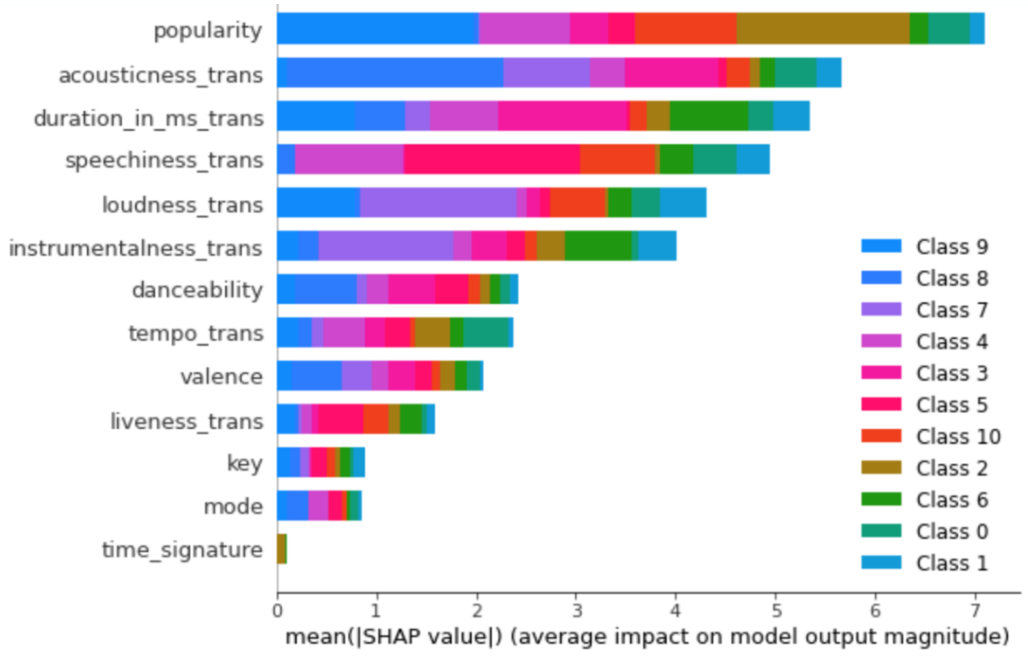
Technologies:

- **Python**: Programming language used for data manipulation, analysis, and machine learning.
- **Scikit-learn**: Machine learning library for implementing algorithms and model evaluation.
- **Pandas and Numpy**: Libraries for data manipulation and numerical operations.
- **Matplotlib and Seaborn**: Libraries for data visualization.
- **XGBoost**: Machine learning library for gradient boosting.

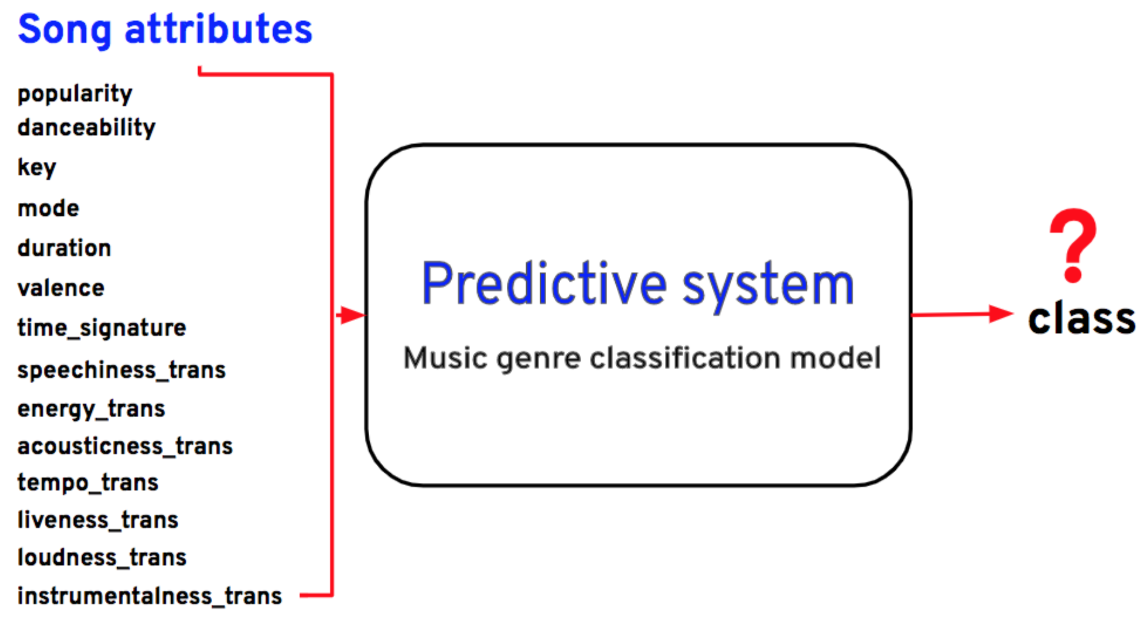

Built machine learning models
(Model - 1, Model - 2, ... Model - n)
Evaluate models
Choose a better model
Hyperparameter tuning and revaluation
Better tuned machine learning model
Predictive system

Note: In our case, the predictive system is the music genre classification system

- **SHAP**: Library for explaining model predictions and identifying feature importance.



Testing Techniques:



- **Cross-Validation**: To assess the model's performance on different subsets of the data.
- **Confusion Matrix**: To evaluate the accuracy of the classification model.
- **Hyperparameter Tuning**: To optimize the model parameters for better performance.
- **Model Evaluation**: Use log-loss to evaluate the performance on both training and test datasets.
- **Generalization Check**: Compare training and test log-loss to identify overfitting or underfitting.
- **Feature Importance Analysis**: Use SHAP values to determine the significance of each feature.

Project Contribution:

- Developed and evaluated multiple machine learning models for music genre classification.
- Improved model performance through hyperparameter tuning.
- Identified key features influencing genre classification using SHAP values.

- Implemented a system for predicting music genres on real-world data.
- Demonstrated the use of a no-code solution, Mage, for building efficient machine learning models with ease.