# Kalyani Government Engineering College

Affiliated to

**Maulana Abul Kalam Azad University of Technology**

## Department of Computer Application

## Kalyani - 741235, Nadia, WB

# 

# Project report on

# Music Genre Classification using Machine Learning Techniques

Submitted by

**Deesha Adhikary University Roll No- 10271021011**

**Purboday Datta University Roll No- 10271021014**

**Sneha Roy University Roll No- 10271021016**

# Under the guidance of

Prof. Dr. Surya Sarathi Das

**Phone: 25826680 (PBX) e-mail:**

**Fax : 25821309**



***Certificate of Approval***

This is to certify that Project Report on “MUSIC GENRE CLASSIFICATION USING MACHINE LEARNING TECHNIQUES” is a record of project work under the curriculum of Maulana Abul Kalam Azad University of Technology(MAKAUT) for the 2nd year 4th semester Examination, 2023 for the subject “ Major Project & Viva-voce(MCAN-482) carried out by *Deesha Adhikary University Roll No- 10271021011* , *Purboday Datta University Roll No- 10271021014*, and *Sneha Roy University Roll No- 10271021016* , students of Kalyani Govt. Engineering College under the guidance of Prof. Dr. Surya Sarathi Das as a requirement for the partial fulfillment of the Degree of Master of Computer Application.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Head Supervisor

Department of Computer Application Department of Computer Application

Kalyani Government Engineering College Kalyani Government Engineering College

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Examiner

**Acknowledgement**

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success. We are grateful to our project guide Dr. Prof. Surya Sarathi Das for the guidance, inspiration and constructive suggestions that help us in the preparation of this project. We would also like to thank Dr. Surya Sarathi Das, Head, Department of CA, for his kind help towards the successful completion of the project. We would like to take this opportunity to thank Prof. Sourabh Kumar Das, Principal, Kalyani Government Engineering College, Kalyani for supporting us in our work.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DEESHA ADHIKARY

Roll Number: 10271021011

PURBODAY DATTA

Roll Number: 10271021014

SNEHA ROY

Roll Number: 10271021016

2nd year 4th semester

Department of Computer Application

|  |  |
| --- | --- |
| **Topic** | **Page Number** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Table Of Contents:**

**Introduction**

A music genre is a style or a type of music. There are numerous music genres, such as hip-hop, rock, country, pop, etc. Music is classified into genres based on various factors such as forms, styles and so on.

Classifying a song's genre allows music lovers to create a playlist of their favorite tracks, and it also helps music streaming services provide recommendations to users based on the genre of the songs they enjoy.

If one has to manually classify the songs or music, one has to listen to a whole lot of songs and then select the genre. This is not only time-consuming but also difficult. Automating music classification can help to find valuable data such as trends, popular genres and artists easily. Determining music genres is the very first step towards this direction. This is where the various techniques of Machine Learning step in.

A technology of Artificial Intelligence, Machine Learning is a concept wherein computers or machines learn from information (data) fed to them. Based on data that is entered into the machines, Machine Learning helps computers to build interpretive patterns and build analytical models automatically. Music Genre Classification using Machine Learning is a comparatively newer concept that has emerged on the surface in recent times. While music genres have been known to the world for decades, machines have been able to work along the lines of music genre classification in the contemporary world where every other person is listening to music.

**Objective**

The objective of this project is to build a predictive system or a machine learning model to classify the genre of a given song, i.e., if a song with a bunch of different attributes is fed into the machine learning model, the model should classify the category to which the song belongs. The music genre classification can be built using different approaches in which the top 4 approaches that are mostly used are listed below.

1. K-Nearest Neighbors
2. Decision Tree
3. Random Forest
4. Gaussian Naïve Bayes

K-Nearest Neighbors algorithm is used because various researches prove it is one of the best algorithms to give good performance and till time along with optimized models organizations uses this algorithm in recommendation systems as support. However, we have used all the algorithms listed above to present a comparative study of the efficacy of the various algorithms of Machine Learning.

The various genres into which music has been classified into are:

1. Blues
2. Classical
3. HipHop
4. Jazz
5. Pop
6. Rock
7. Metal
8. Country
9. Disco
10. Reggae

**Motivation**

Machine learning is a very important part of our life in this digital age. So as students of Computer Application, we want to employ various techniques of Machine learning in our everyday lives.

One such integral part of our lives is listening to music so we aim to create platform that will automatically classify the music we listen to in various genres. This way we can listen to a certain genre depending on our mood.

The project has been done using Jupyter notebook, which uses the Python programming language. Python is consistent and is anchored on simplicity, which makes it most appropriate for machine learning. Also, Python has many inbuilt libraries like Librosa, which makes the job much easier.

The most common used algorithm in such projects is KNN as it is highly efficient and easy to use. However, we also want to know the efficacy and difficulty level of the other widely used algorithms such as Decision Tree, Random Forest and so on, so as to present a comparative study of all the algorithms in addition to our primary objective of music genre classification.

In our project, we have used KNN both by manual implementation and by using classifiers so that we can compare the efficacy, accuracy and performance of the algorithm in both cases.

We also threw light on the feature extraction of audio files via audio processing libraries present in Python. According to different music sequence measurements, the feature sequence mechanism of music design feedback optimization is also investigated in this project. The features like Zero Cross-rate, Cepstral Coefficients, Spectral Bandwidth and so on are the basis for the audio processing.

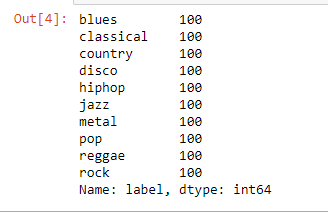
**Tools and Methods**

For the fulfilment of the aim of this project, various Supervised Machine Learning Algorithms have been implemented using Python Programming Language on the Jupyter Notebook Platform. Also, various Python Modules have been imported and further used for extracting features of audio files and implementing classifiers.

**Dataset overview**:

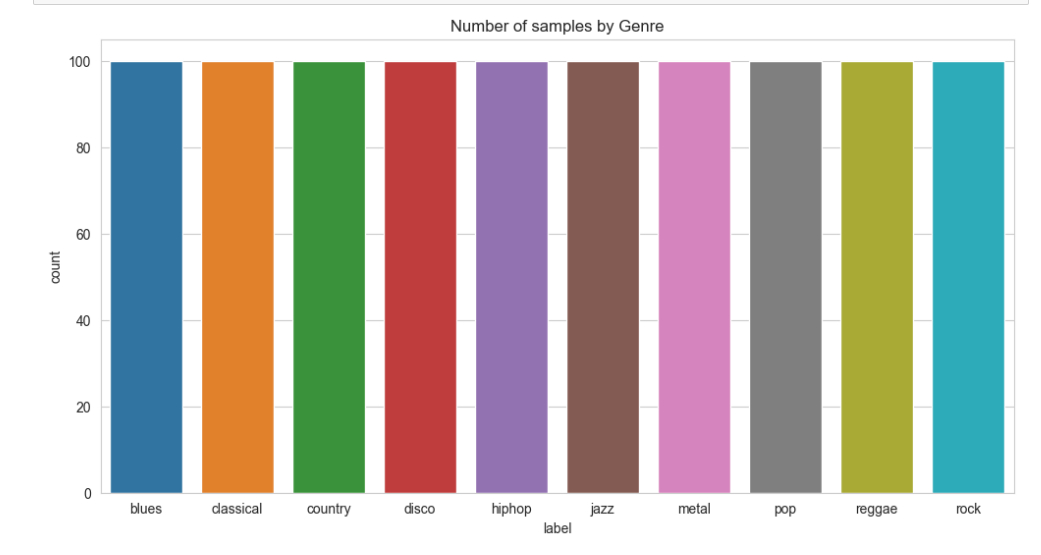
For doing this project, a dataset has to be used which will be required by the machine for training and testing purpose. The GTZAN dataset music collection has been used which has been downloaded from **www.kaggle.com**.

GTZAN dataset contains 1000 music samples (sampling frequency 22,500 Hz, 16 bits resolution, and 30 s duration) belonging to 10 different classes. All the samples are in .wav format. Genres in the GTZAN are blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae and rock as shown in Figure 1.

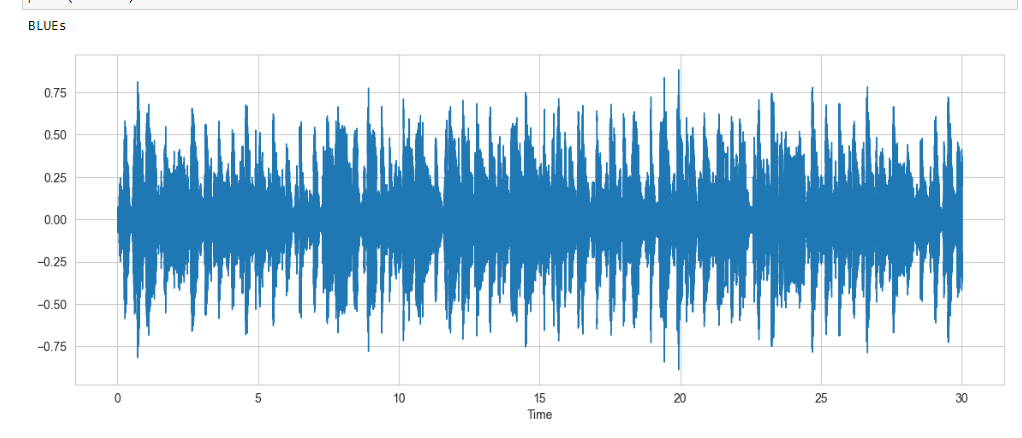


**Figure 1** – Genres in GTZAN dataset

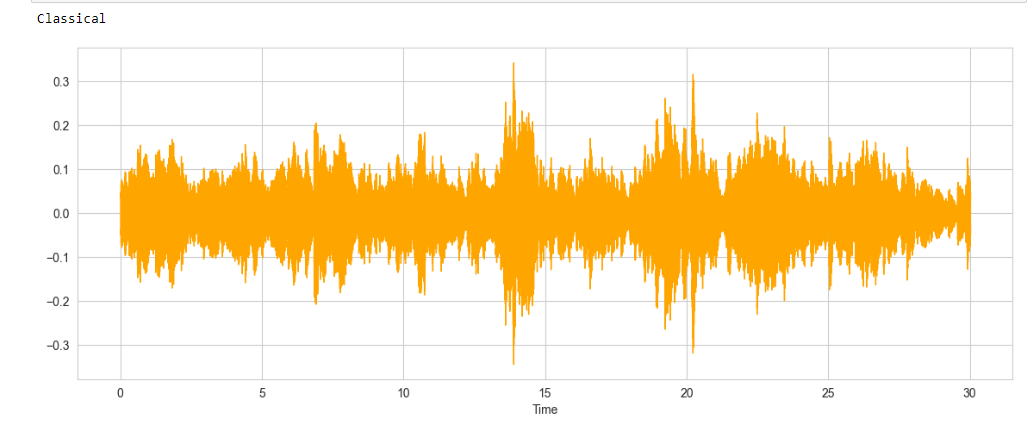
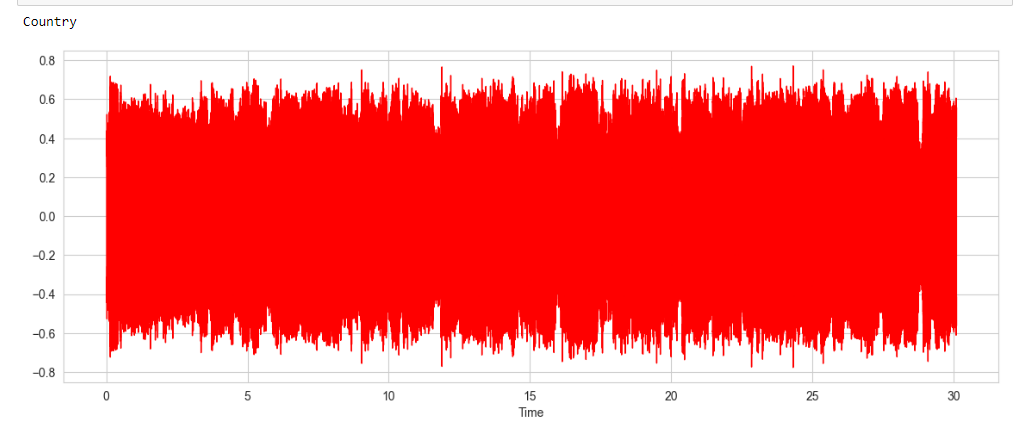
The data used is well balanced and the data is evenly distributed throughout each and every genre. The balanced distribution of the dataset can be evident in Figure 2.-



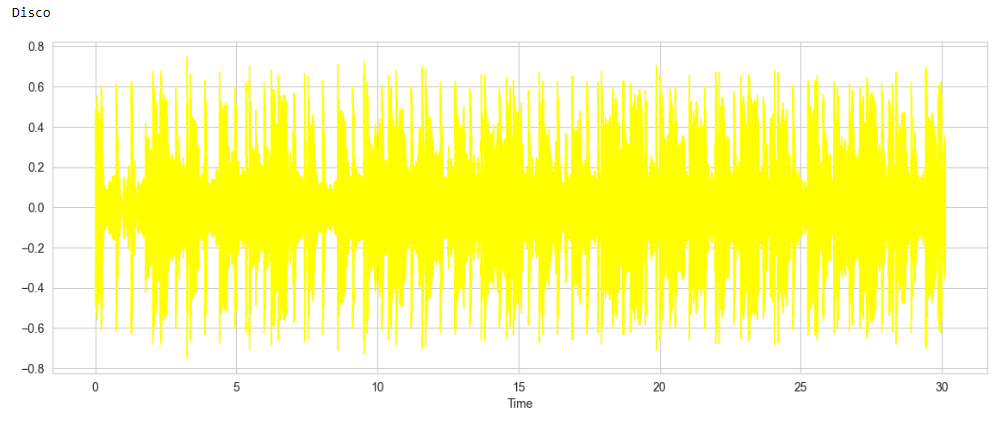
**Figure 2** – Number of samples in each genre

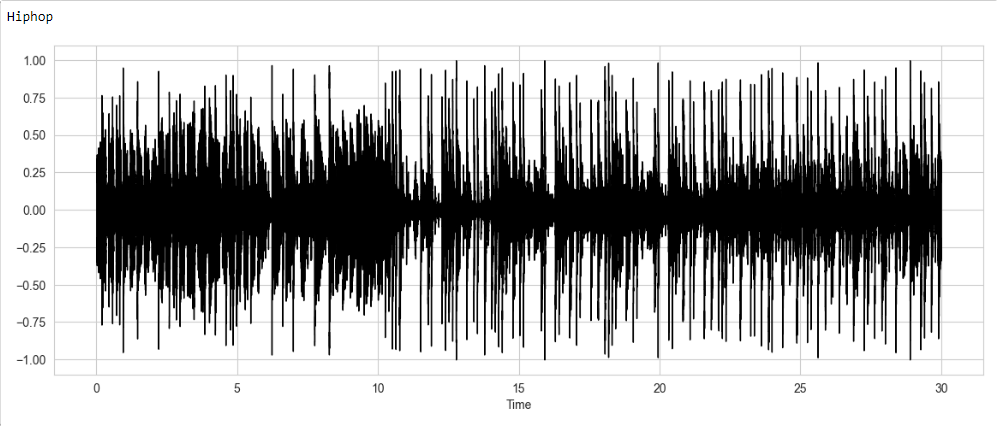
**Wave form of each genre:** Each genre of music sample in the set is different from the other by their characteristics as well as features. This can be evident if waveform of each genre is plotted. The waveform of each genre is shown from Figure 3 to Figure 12

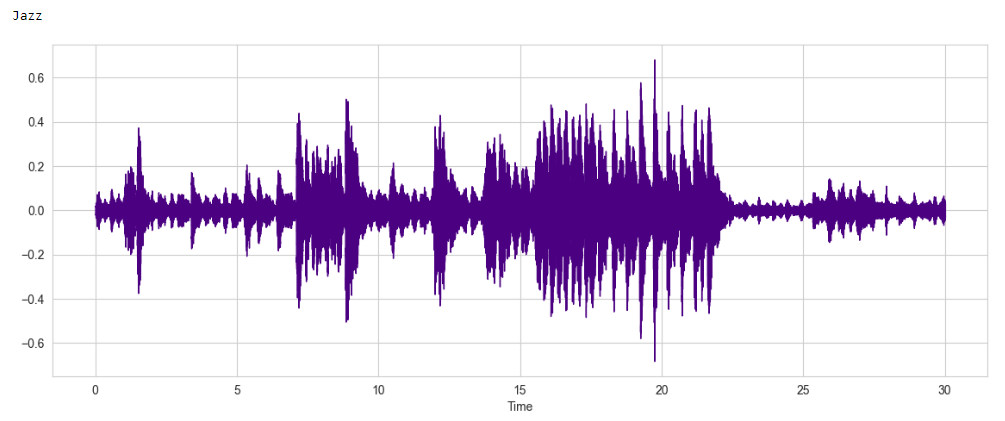
**Figure 3** – Wave form of the music genre ‘Blues’

**** **Figure 4** – Wave form of the music genre ‘Classical’

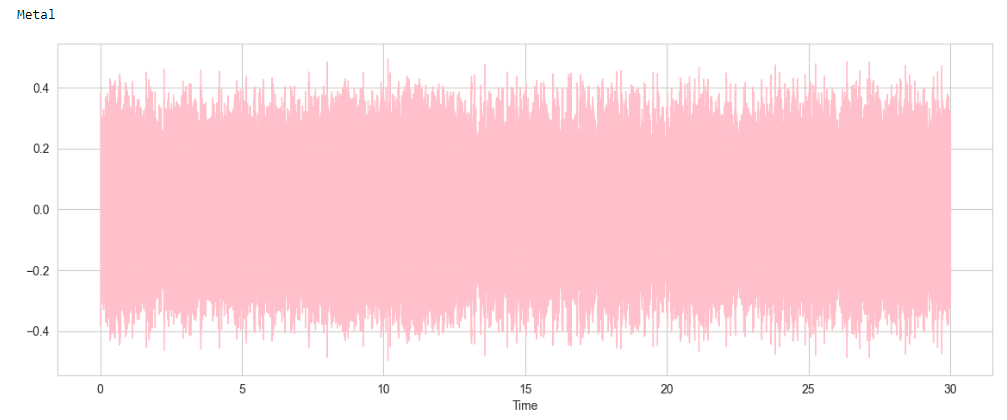
**Figure 5** – Wave form of the music genre ‘Country’

**Figure 6** – Wave form of the music genre ‘Disco’

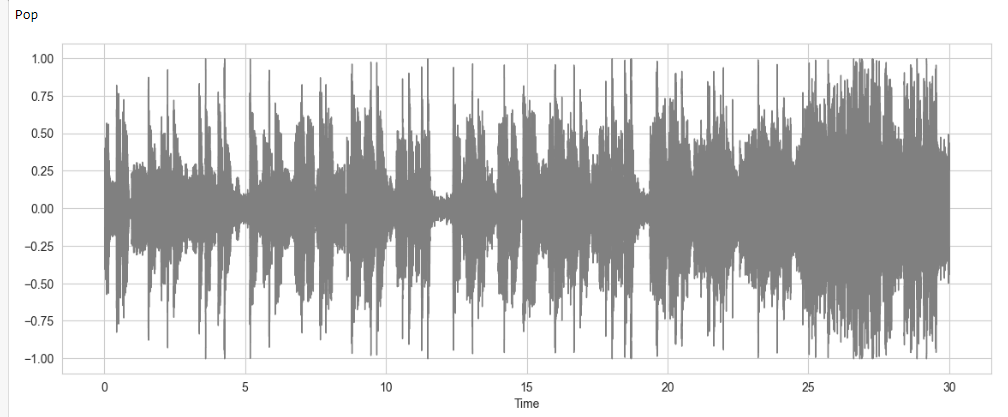
**Figure 7** – Wave form of the music genre ‘HipHop’



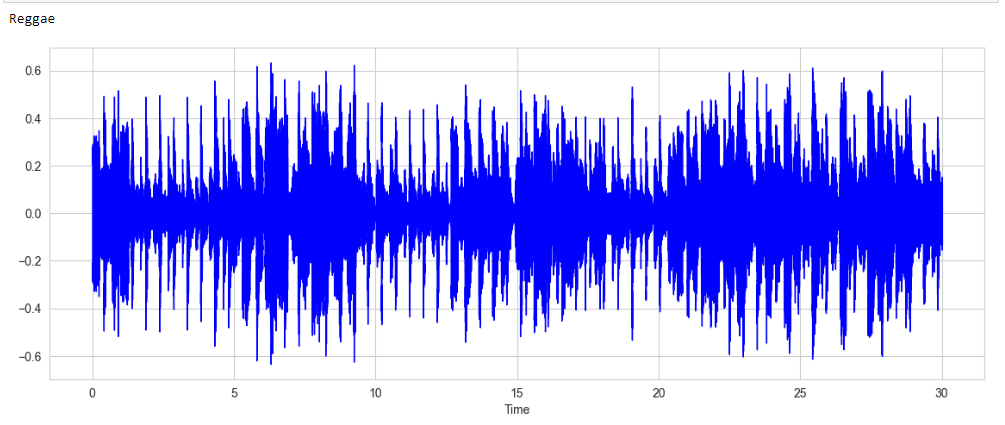
**Figure 8** – Wave form of the music genre ‘Jazz’



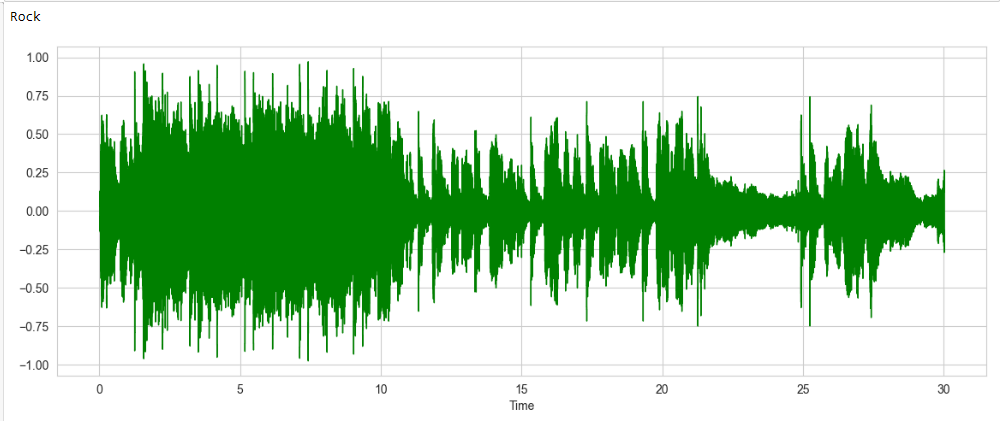
**Figure 9** – Wave form of the music genre ‘Metal’



**Figure 10** – Wave form of the music genre ‘Pop’



**Figure 11** – Wave form of the music genre ‘Reggae’



**Figure 12** – Wave form of the music genre ‘Rock’

**Libraries used:**

As the Jupyter Notebook platform has been used for implementation, before we move to load dataset and model building its important to install certain libraries. The following libraries have been used:

1. **Librosa** - Librosa is valuable Python music and sound investigation library that helps programming designers to fabricate applications for working with sound and music document designs utilizing Python.

2. **Python speech features -** This library provides common speech features for ASR including MFCCs andfilter bank energies**.**

3. **Pickle -** Pickle is a useful Python tool that allows to save ML models, to minimize lengthy re-training and allow to share, commit, and re-load pre-trained machine learning models.

4. **Scipy.io.wavfile** - SciPy is a scientific computation library that uses NumPy underneath. SciPy stands for Scientific Python. It provides more utility functions for optimization, stats and signal processing. Like NumPy, SciPy is open source so we can use it freely.

5. **Pandas** - Pandas is a Python library used for working with data sets. It has functions for analyzing, cleaning, exploring, and manipulating data.

6. **Sklearn** - Scikit-learn is an open-source data analysis library, and the gold standard for Machine Learning (ML) in the Python ecosystem.

7. **Seaborn** - Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics

8. **Matplotlib** - Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. It is used here to visualize the data frame.

9. **Numpy** - Numpy adds powerful data structures to Python that guarantee efficient calculations with arrays and matrices and it supplies an enormous library of high-level mathematical functions that operate on these arrays and matrices. It is used here to perform operations like scaling and correlation.

**Design of The Model**

**Feature Extraction:**

Feature extraction is a process to extract important features from data. Audio features are classified into 3 categories high-level, mid-level, and low-level audio features.

* High-level features are related to music lyrics like chords, rhythm, melody, etc.
* Mid-level features include beat level attributes, pitch-like fluctuation patterns, and MFCCs.
* Low-level features include energy, a zero-crossing rate which are statistical measures that get extracted from audio during feature extraction.

To generate these features a certain set of steps are used and are combined under a single name as MFCC (Mel Frequency Cepstral Coefficients) that helps extract mid-level and low-level audio features. Below are the steps discussed for the working of MFCCs in feature extraction.

1. Audio files are of a certain length(duration) in seconds or as long as in minutes. And the pitch or frequency is continuously changing so the audio files are divided into small-small frames which are near about 20 to 40 ms long.
2. After dividing into frames, different frequencies are identified and extracted from each frame.
3. Linguistic frequencies are separated from the noise
4. To discard any type of noise, discrete cosine transform (DCT)[1] of the frequencies are taken.

MFCC combines these steps which has already been imported from the python speech feature library. Each category folder is iterated through, audio file read, MFCC feature extracted, and dumped in a binary file using the pickle module.

Following are the main features, on the basis of which the classification is done:

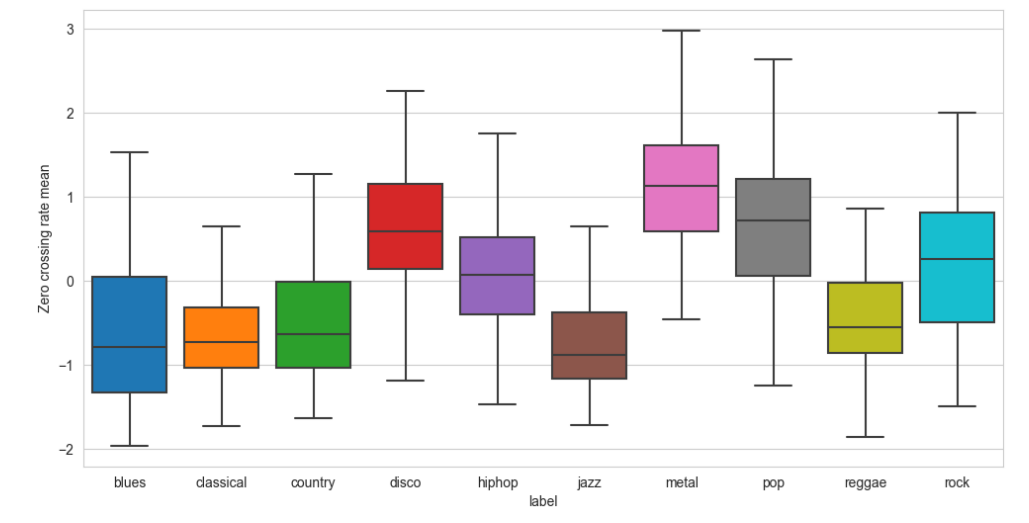
* **Mel-Scale Frequency Cepstral Coefficients**

Since the sound level perceived by the human ear is not linearly related to its frequency, researchers have proposed a new concept called Mel frequency. The Mel frequency scale is more in line with the auditory characteristics of the human ear.

* **Zero Crossing Rate**

The zero-crossing rate (ZCR), which is used in signal processing, is the rate at which a signal changes from positive to negative or vice versa.

The mean zero-crossing rate of the various genres has been analyzed using the box plotting in Figure 13.

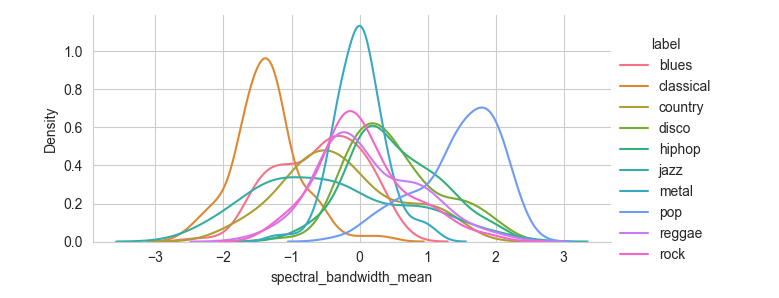
**Figure 13** – Zero Crossing Rate Mean of different genres

* **Spectral Bandwidth**

Bandwidth is the difference between the upper and lower frequencies in a continuous band of frequencies. As we know the signals oscillate about a point so if the point is the centroid of the signal then the sum of maximum deviation of the signal on both sides of the point can be considered as the bandwidth of the signal at that time frame. The spectral bandwidth can be computed by:

Where *p* is order and *t* is time.

The density graph for the spectral bandwidth is shown in Figure 14.

**Figure 14** – Density graph of spectral bandwidth of different genres

#### **Train-test split the dataset:**

Features have been extracted from the audio file which is dumped in binary format as a filename of the dataset. Now a function will be used that accepts a filename and copies all the data in form of a data frame. After that based on a certain threshold, the data will be split into train and test sets. There are different approaches to do train test split. In this case, a random module is being used and running a loop till the length of a dataset and generate a random fractional number between 0-1 and if it is less than 66 then a particular row is appended in the train test else in the test set.

**Implementation Of Algorithms**

As the objective is to automate the genre classification, techniques of machine learning are being used to perform the classification as implementing the task manually is cumbersome. In recent years, Machine Learning has been a major turning point in the field of Data Science to enact classifications or predictions.

* **Machine Learning** - Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. It enables computers to learn automatically from past data by using various algorithms for building mathematical models and making predictions. Currently, it is being used for various tasks such as image recognition, speech recognition, email filtering, recommendation system and many more.

In machine learning, classification is a predictive modelling problem where the class label is anticipated for a specific example of input data. To perform classification, one of the most uncomplicated ways is using the application of Supervised Learning Techniques.

* **Supervised Learning-** Supervised learning is the types of machine learning in which machines are trained using well "labelled" training data, and on basis of that data, machines predict the output. The labelled data means some input data is already tagged with the correct output. Here, the training data provided to the machines work as the supervisor that teaches the machines to predict the output correctly. The aim of a supervised learning algorithm is to find a mapping function to map the input variable(x) with the output variable(y).

We are applying a special category of Supervised Learning Algorithms for the genre classification, known as the Classification Algorithms.

We have used the following supervised machine learning algorithms for making predictions and comparing accuracy of the same:

* K-Nearest Neighbours
* Decision Tree
* Random Forest
* Gaussian Naïve Bayes

**Algorithms**

* **K Nearest Neighbours**

The k-nearest neighbours (KNN) algorithm is a supervised machine learning algorithm used for both classification and regression tasks. It is a non-parametric method that makes predictions based on the similarity of input samples to labelled training data.

Here's a step-by-step explanation of the KNN algorithm:

* 1. **Data Preparation**: The training data is collected and pre-processed. This typically involves cleaning the data, normalizing or scaling features, and splitting the data into a training set and a test set.
  2. **Choosing value of k:** The value of k is determined, which represents the number of nearest neighbours to consider when making predictions. This value can be chosen based on domain knowledge or through experimentation.
  3. **Distance Calculation:** The distance between the target sample and all samples in the training set is calculated. The most commonly used distance metric is the Euclidean distance[2], but other metrics like Manhattan distance[3] or Minkowski distance[4] can also be used.
  4. **Finding k Nearest Neighbours:** The k samples are selected from the training set that have the smallest distances to the target sample. These samples become the "neighbours" of the target sample.
  5. **Determining Class/Label:** For classification, the class label that is most frequent among the k nearest neighbours to the target sample is assigned. For regression[5], the average (or weighted average) of the target values of the k nearest neighbours is calculated.
  6. **Prediction**: The assigned class label or calculated value is considered the prediction for the target sample.
  7. **Evaluation of Performance:** The trained KNN model is applied to the test set and its performance is assessed using appropriate evaluation metrics such as accuracy, precision, recall, F1 score, or mean squared error (MSE)[6], depending on the problem type.
  8. **Optimizing Parameters:** Parameter tuning, such as selecting the optimal value of k or experimenting with different distance metrics to improve the performance of the model is performed.
  9. **Predicting New Samples:** Once the model is trained and tuned, it can be used to make predictions on new, unseen samples by following steps 3 to 6.

It's important to note that the KNN algorithm has some limitations, such as being sensitive to the choice of distance metric, requiring a large amount of memory to store the training data, and potentially being affected by irrelevant or noisy features. Pre-processing and feature selection techniques can help mitigate these challenges.

* **Decision Tree**

The decision tree algorithm is a popular machine learning algorithm used for both classification and regression tasks. The basic idea behind the algorithm is to recursively partition the input data based on certain features or attributes until a specific stopping criterion is met.

Here's a simplified version of the decision tree algorithm:

1. The entire dataset is considered as the root node of the tree.
2. An attribute is selected from the dataset to act as the root node, based on certain criteria (e.g., information gain, Gini index, or other impurity measures).
3. The dataset is split into subsets based on the selected attribute's possible values.
4. For each subset created in the previous step, steps 2 and 3 are repeated until a stopping criterion is met. This criterion could be reaching a maximum depth, reaching a minimum number of samples in a node, or other conditions.
5. A label or value to each leaf node is assigned based on the majority class or average value of the samples in that leaf node.
6. Optionally, pruning[7] techniques are applied to reduce overfitting[8], such as post-pruning or pre-pruning.
7. The decision tree is now trained and can be used for prediction on unseen data.

During the training process, decision trees use various metrics to determine the best attribute to split the data at each step. Some common metrics include:

- **Information Gain**: Measures the reduction in entropy or the amount of information gained by splitting the data using a specific attribute.

- **Gini Index**: Measures the impurity of a node by calculating the probability of misclassifying a randomly chosen element from the node.

- **Chi-Square Test**: Evaluates the independence between attributes and the class variable.

* **Random Forest**

Random Forest is an ensemble learning algorithm that combines multiple decision trees to make predictions. Random Forest is known for its ability to handle high-dimensional data, provide good generalization, handle missing values, and avoid overfitting. It is widely used for both classification and regression tasks.

The algorithm follows the following steps:

1. **Random sampling**: A subset of the original data (with replacement) is randomly selected from the training set. This subset is known as a bootstrap sample.
2. **Tree construction**: A decision tree is constructed using the bootstrap sample. At each node of the tree, a random subset of features is selected, and the best feature among them is chosen as the split criterion. The tree continues to split the data into smaller subsets based on the selected features until a stopping criterion is reached (e.g., a maximum tree depth or a minimum number of samples per leaf).
3. Steps 1 and 2 are repeated to create multiple decision trees, typically hundreds or thousands of trees.
4. **Voting or averaging**: For classification tasks, each tree in the forest independently predicts the class label of a new sample. The final prediction is determined by majority voting, i.e., the class label that receives the most votes among all the trees. For regression tasks, the individual tree predictions are averaged to obtain the final prediction.

* **Gaussian Naïve Bayes**

Gaussian Naive Bayes (GNB) is a classification technique used in Machine Learning (ML) based on the probabilistic approach and Gaussian distribution. Gaussian Naive Bayes assumes that each parameter (also called features or predictors) has an independent capacity of predicting the output variable. The combination of the prediction for all parameters is the final prediction, that returns a probability of the dependent variable to be classified in each group. The final classification is assigned to the group with the higher probability.

The algorithm for Gaussian Naive Bayes can be summarized in the following steps:

1. **Data Preparation:** A labelled dataset is taken where each instance is described by a set of features and associated with a class label.
2. **Feature Extraction:** Relevant features are extracted from the dataset that will be used to make predictions.
3. **Data Split:** The dataset is divided into training and testing subsets to evaluate the model's performance.
4. **Training Phase:**

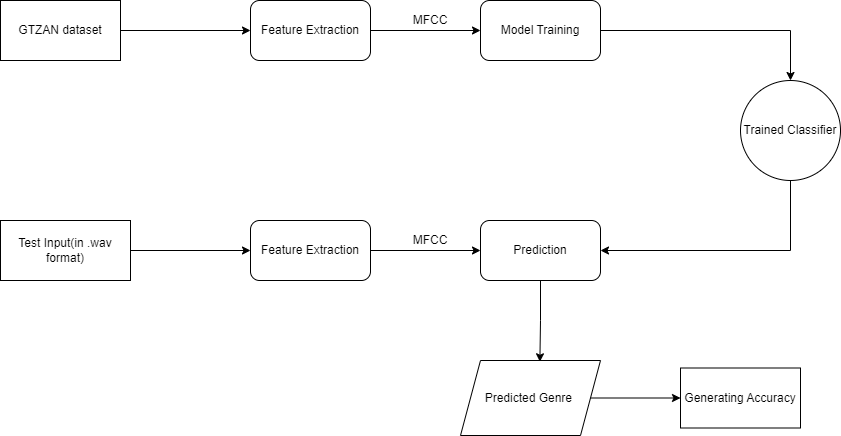
* The prior probability of each class is calculated by counting the frequency of each class label in the training set.
* For each feature, the mean and standard deviation of the feature values are calculated for each class.

1. **Prediction Phase:**

* Given a new instance with feature values, the posterior probability for each class is calculated using Bayes' theorem and the Gaussian probability density function.
* The class with the highest posterior probability is selected as the predicted class for the instance.

1. **Model Evaluation:** The performance of the model is evaluated by comparing the predicted class labels with the true class labels from the testing set, using metrics such as accuracy, precision, recall, or F1 score.

**Block Diagram of the Work Flow**

****

**Figure 15** – Block Diagram of the Work Flow

**Results and Analysis**

**Analysis of the parameter:**

Choosing the correct value of k in KNN (k-nearest neighbours) is important because it directly influences the performance and accuracy of the algorithm. The value of k determines the number of nearest neighbours that will be considered when making predictions.

If k is set too low, the algorithm may be more sensitive to noise or outliers, leading to overfitting. This means that the algorithm might make predictions based on a small number of neighbours, potentially leading to incorrect classifications.

On the other hand, if k is set too high, the algorithm may become overly generalized, resulting in underfitting. In such cases, the predictions might be biased towards the majority class, disregarding the local patterns and structure in the data.

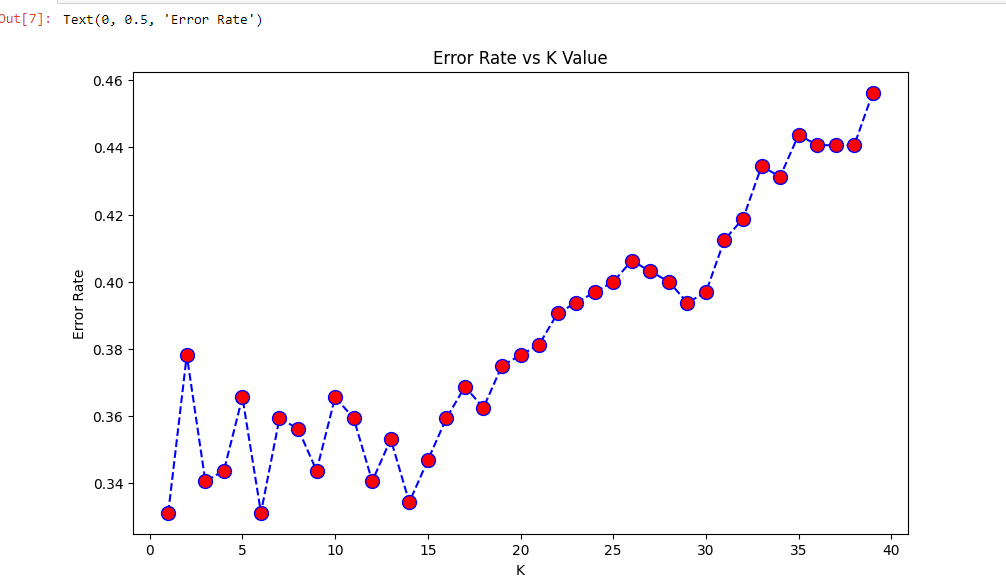
Therefore, selecting an appropriate value of k is crucial for finding the right balance between overfitting and underfitting, which ultimately affects the accuracy and performance of the KNN algorithm.

Also, plotting the error rate vs. k graph for KNN (k-nearest neighbours) is important because it helps in selecting the optimal value of k for the algorithm. By analysing Figure 16, we can observe how the error rate changes as the value of k varies. This allows us to identify the value of k that yields the lowest error rate or the best performance on the dataset.

The graph in Figure 16 provides insights into the bias-variance trade-off in KNN. A smaller value of k leads to a low bias but high variance, while a larger value of k leads to a high bias but low variance. The graph helps us visualize this trade-off and choose the appropriate value of k that strikes the right balance between bias and variance for the specific dataset.

Furthermore, the error rate vs. k graph in Figure 16 allows us to detect overfitting and underfitting. If the error rate is low for very small values of k but starts to increase as k becomes larger, it indicates that the model is overfitting. Conversely, if the error rate is high for small values of k but decreases as ‘k’ increases, it suggests that the model is underfitting.

In summary, plotting the error rate vs. k graph provides valuable insights into the performance and generalization ability of the KNN algorithm and aids in selecting the optimal value of k.

From Figure 16, we observe that the error rate is minimum when the value of k is around 5-7. So, we take the value of k to be 7 to maximize the accuracy rate of the KNN classifiers.

**Figure 16 –** Error rate vs K value

**Prediction:**

The prediction has been done using manual implementation of KNN algorithm as the other algorithms used here are extremely inconvenient for manually implementing predicting results. Figure 17 shows the predicted music genre of an input data.

**Figure 17** – Prediction of the music genre

The above shows us that the genre of the input music file has been correctly predicted by the machine.

**Accuracy rate:**

The accuracy rate of a classification algorithm is a metric that measures the proportion of correctly classified instances out of the total number of instances in the dataset. It is commonly expressed as a percentage. The accuracy rate can be calculated using the following formula: Accuracy Rate = (Number of correctly classified instances / Total number of instances) \* 100 For example, if a classification algorithm correctly classifies 900 out of 1000 instances in a dataset, the accuracy rate would be:

*Accuracy Rate = (900 / 1000) \* 100 = 90%*

The accuracy rate provides an overall measure of how well the classification algorithm is performing.

The following are the accuracy rates of the various algorithms that have been implemented in this project.

**Table 1** – Tabular Representation of accuracy of various Algorithms

|  |  |
| --- | --- |
| Algorithms | Accuracy |
| KNN | 69.21 |
| Decision Tree | 57.81 |
| Random Forest | 76.88 |
| Gaussian Naïve Bayes | 55.63 |

From Table 1, it is quite evident that Random Forest has the highest accuracy rate while Gaussian Naïve Bayes has the lowest.

**Confusion Matrix:**

A confusion matrix, also known as an error matrix, is a table that is often used to evaluate the performance of a classification algorithm. It provides a detailed breakdown of the predicted and actual class labels for a given dataset. The matrix summarizes the number of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) predictions made by the classification algorithm.

Here's a breakdown of the elements in the confusion matrix:

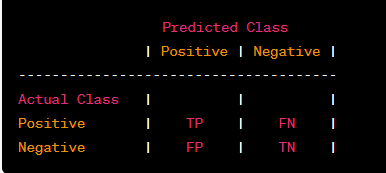
True Positive (TP): The algorithm correctly predicted a positive instance as positive.

True Negative (TN): The algorithm correctly predicted a negative instance as negative.

False Positive (FP): The algorithm incorrectly predicted a negative instance as positive. Also known as a Type I error.

False Negative (FN): The algorithm incorrectly predicted a positive instance as negative. Also known as a Type II error.

A confusion matrix has a tabular structure, typically organized as shown in Figure 18:



**Figure 18**- Tabular structure of Confusion Matrix

By examining the values in the confusion matrix, various evaluation metrics can be derived to assess the classification algorithm's performance, such as:

Accuracy: The overall accuracy of the classification algorithm, calculated as (TP + TN) / (TP + TN + FP + FN).

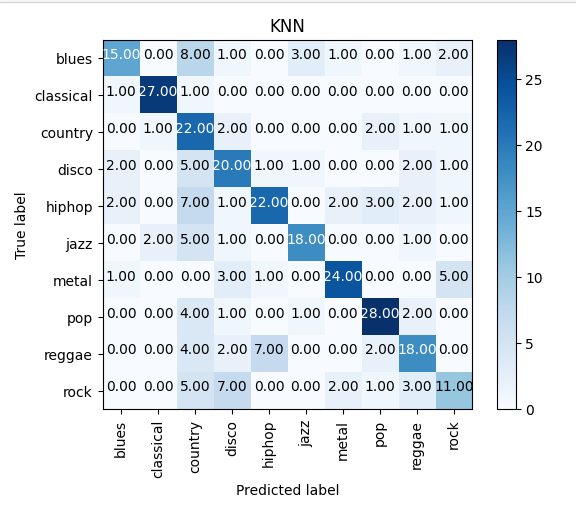
Precision: The proportion of correctly predicted positive instances out of the total instances predicted as positive, calculated as TP / (TP + FP).

Recall (Sensitivity or True Positive Rate): The proportion of correctly predicted positive instances out of the total actual positive instances, calculated as TP / (TP + FN).

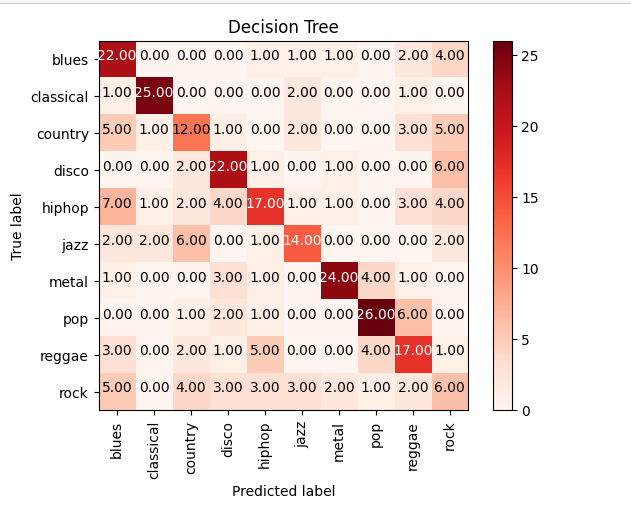
Specificity (True Negative Rate): The proportion of correctly predicted negative instances out of the total actual negative instances, calculated as TN / (TN + FP).

F1 score: A weighted average of precision and recall that balances the trade-off between them, calculated as 2 \* (Precision \* Recall) / (Precision + Recall).

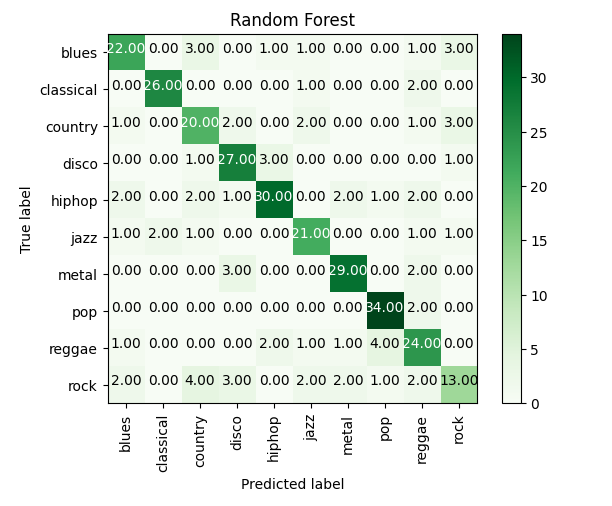
The confusion matrix provides a more detailed understanding of the classification algorithm's performance by revealing the types and quantities of prediction errors made. It is a valuable tool for evaluating and fine-tuning classification models

The confusion matrices for the various algorithms used are given below from Figure 19 to Figure 22 one after the other:

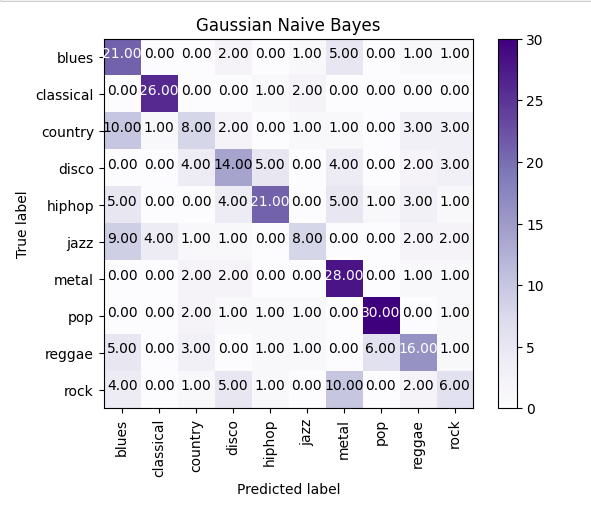
**Figure 19-** Confusion matrix of KNN algorithm (Manual Implementation)



**Figure 20-** Confusion matrix of Decision Tree algorithm



**Figure 21-** Confusion matrix of Random Forest algorithm

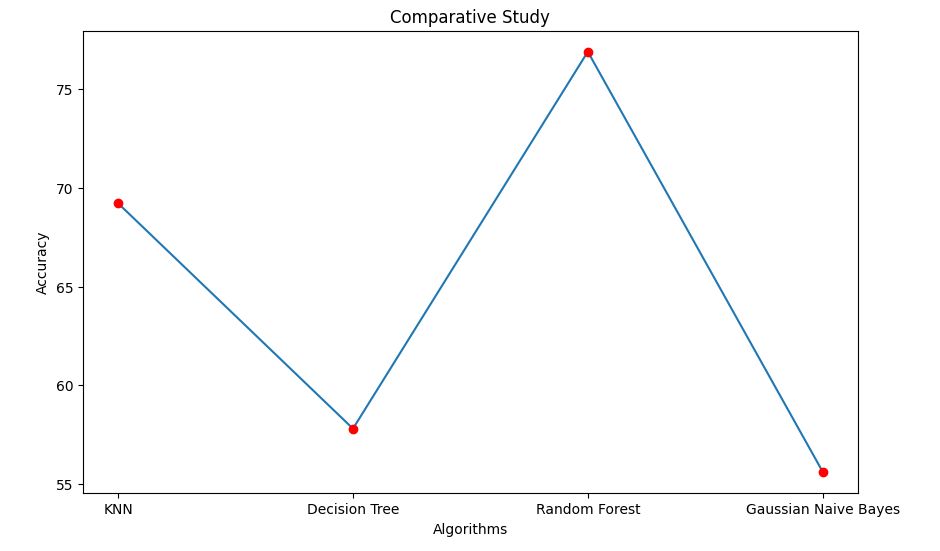


**Figure 22-** Confusion matrix of Gaussian Naïve Bayes algorithm

**Comparative Study:**

We observed above that the various algorithms have different accuracies while predicting the genre of the song. Among all the classification algorithms used, Random Forest is spotted to give the highest accuracy while Gaussian Naïve Bayes is noticed to have the minimum accuracy. Also, it is to be noted that the manual implementation of KNN (k nearest neighbors) algorithm yields a much better accuracy than that of the classifier.

Thus, by comparing all the results produced, it can be concluded that the classification must be done using Random Forest if we want to focus to get a better accuracy. It is to be noted that Random Forest is one of the most complicated algorithms and cannot be implemented manually to make predictions. Hence, manual implementation of KNN is the best solution to predict the genres, as it gives a decent accuracy as well as it is very easy to design manually for predicting genres. A comparative study of accuracy of each algorithm in graphical form is given in Figure 22.



**Figure 23-** Graphical representation of accuracy of different algorithms

**Future Scope**

At present, the application focuses only on classifying music which has been given as an input by a user into various genres. However, we plan to increase the scope of this project where it can recommend music to a user. This can be accomplished by using Recommendation System where it will suggest music based on various criteria, including past endeavors, search history, demographic information, and other factors.

We also plan to increase the scope of this platform by using Deep learning algorithms, such as convolutional neural networks (CNNs). Applying these models to music genre classification can capture intricate patterns and dependencies within audio signals, potentially leading to better performance compared to traditional supervised learning algorithms.

Furthermore, music genres often exhibit complex relationships and overlap. Extending supervised learning approaches to handle multi-label classification, where a track can be associated with multiple genres simultaneously, can provide more nuanced genre representations. Additionally, fine-grained genre classification, distinguishing subgenres or more specific descriptors within genres, can enhance the granularity of music genre classification systems.

Music genre classification often varies across cultures, regions, and time periods. Adapting supervised learning models to handle cross-domain and cross-cultural variations can improve their generalization and applicability in diverse musical contexts. Incorporating domain adaptation techniques and considering cultural perspectives can enhance the accuracy and relevance of genre classifications across different musical traditions.

**Limitations**

* This platform is only capable of classifying music into various genres. It cannot analyze the lyrics and classify the music on that basis. For lyrics analysis, Convolutional Neural Networks (CNN) and Natural Language Processing (NLP) comes into play which are outside the scope of supervised learning.
* The code for this platform has been written using Python language where we have made use of certain libraries like Librosa for feature extraction. However, Librosa does not work on all the versions of Python which may hamper the processing time of the code and in turn execution of the project.
* The dataset to be used contains features like zero cross rate, mel-frequency cepstral coefficient, etc. However, all the datasets do not contain the above features. Such datasets cannot be used. Deep Learning has to be used for feature extraction in such cases.
* This project focuses only on the various techniques of supervised learning. There is no such implementation of unsupervised learning. For this reason, we had to divide the dataset into a certain ratio for testing and training and also train the machine for correct result prediction.

**Conclusion**

In conclusion, music genre classification using various supervised learning techniques has proven to be an effective approach for categorizing music tracks into specific genres. The application of these techniques has led to significant advancements in automatic genre classification. The choice of the most suitable supervised learning algorithm depends on the characteristics of the dataset and the complexity of the genre classification problem. Moreover, evaluating and comparing the performance of different algorithms using appropriate evaluation metrics is essential for selecting the most effective approach.

However, it is important to note that music genre classification is a subjective task, as genre boundaries can be ambiguous and open to interpretation. The quality and diversity of the training data, along with the expertise and domain knowledge of the human annotators, can greatly influence the performance and reliability of the classification system. Pre-processing steps like data normalization, dimensionality reduction, and balancing class distributions may be necessary to improve the performance of the models. Cross-validation and hyperparameter tuning techniques help optimize the models and improve generalization on unseen data.

Overall, supervised learning techniques provide a robust framework for music genre classification, enabling accurate and automated categorization of music tracks. Continued research and advancements in these techniques will contribute to further improvements in the accuracy, scalability, and applicability of music genre classification systems.

**Bibliography**

Here are some references that can be consulted for music genre classification using supervised learning:

1. Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. IEEE Transactions on speech and audio processing, 10(5), 293-302.
2. Lu, L., Zhang, H. J., & Liu, L. (2006). Automatic musical genre classification using ensemble of classifiers. Multimedia systems, 11(3), 478-490.
3. Bergstra, J., & Bengio, Y. (2013). Music genre classification with the beat spectrum. In ISMIR.
4. <https://www.academia.edu/37729758/Music_Genre_Classification_Project_Report>.

**References**

1. DCT - https://www.geeksforgeeks.org/discrete-cosine-transform-algorithm-program/
2. Euclidean distance - https://www.sciencedirect.com/topics/mathematics/euclidean-distance
3. Manhattan distance – https://www.geeksforgeeks.org/maximum-manhattan-distance-between-a-distinct-pair-from-n-coordinates/
4. Minkowski distance - https://www.geeksforgeeks.org/how-to-calculate-minkowski-distance-in-r/
5. Regression- https://www.mygreatlearning.com/blog/what-is-regression/
6. MSE - https://en.wikipedia.org/wiki/Mean\_squared\_error
7. Pruning – https://en.wikipedia.org/wiki/Decision\_tree\_pruning
8. Overfitting – https://www.ibm.com/topics/overfitting.