Journal of Applied Informatics and Computing (JAIC)
Vol.9, No.4, August 2025, pp. 1825~1834
e-ISSN: 2548-6861

Comparison of Hyperparameter Tuning in Decision Tree and Random Forest Algorithms for Song Genre Classification

Anindita A. Maitsa [1], Nurul A. S. Winarsih [2*]
Universitas Dian Nuswantoro
111202113551@mhs.dinus.ac.id [1], nurulanisasw@dsn.dinus.ac.id [2*]

Article Info

Article history:

Received 2025-07-08
Revised 2025-07-22
Accepted 2025-07-30

Keyword:

Decision Tree,
Hyperparameter Tuning,
Music Genre Classification,
Random Forest.

ABSTRACT

This research applies Decision Tree and Random Forest algorithms for music genre classification based on audio numerical features such as tempo, energy, loudness, and valence. The dataset used comes from Kaggle and consists of 7,958 song entries from eight genres. The data was processed through pre-processing stages that included duplication removal, empty value handling, normalization, outlier removal, and class balancing using the SMOTE technique. In the initial test, Random Forest showed an accuracy of 85%, higher than Decision Tree which recorded 76%. After hyper parameter tuning using GridSearchCV, Decision Tree's accuracy increased to 79%, while Random Forest experienced a slight decrease to 84%. This decrease does not reflect a decrease in performance, but rather a more balanced redistribution of predictions to minor classes, as reflected by the stable F1-score macro value at 0.84. In terms of efficiency, tuning the Random Forest took much longer (806.81 seconds) than the Decision Tree (17.42 seconds), indicating that model complexity has a direct impact on training time. These findings suggest that data quality, tuning strategy and time efficiency are important factors in building a reliable and balanced music genre classification system.

I. INTRODUCTION

Music genre classification approaches utilizing numerical attributes that represent musical characteristics have been widely used in machine learning systems. The accuracy in this classification process not only supports more relevant music recommendation systems, but can also be used in other fields, such as interactive music education and automatic music transcription [1].

This classification process is not free from a number of challenges. One issue that often arises is the uneven amount of data in each genre, where some genres have a much larger number of samples than others. In addition, the presence of outlier values as well as complex and not always linear relationships between features are obstacles for the model in recognizing patterns correctly. Some features also have very diverse value distributions and high correlations between each other, which can affect the overall performance of the model. To overcome this, appropriate data pre-processing steps are needed so that the model can learn relevant information and produce accurate genre predictions.

Another challenge in performing music genre classification lies in choosing the right algorithm. Decision Tree, for example, has the disadvantage of having a tendency to overfit the training data if the model parameters are not set correctly. Meanwhile, Random Forest, which is structurally more complex, can produce more stable predictions, but requires longer training time and more careful parameter tuning for optimal performance. Without a good tuning process, both algorithms can result in low classification accuracy, especially when the model is not properly tuned when dealing with data that is unbalanced or has a high variation in features.

In this research, the genres that are the focus of classification consist of eight categories, namely *blackmetal, comedy, gospel, grindcore, j-idol, samba, study, and tango.* Each genre has unique characteristics and a nonuniform amount of data, which adds to the difficulty of building an accurate model. Therefore, pre-processing steps such as removing duplicates, balancing class distribution, data normalization, and outlier detection become an important part of the model building process.

In addition to pre-processing, hyperparameter tuning plays a vital role in optimizing model performance. Hyperparameters, such as tree depth, minimum samples per split, and number of estimators, significantly influence how well a model learns from the data. Improper settings can lead to overfitting or underfitting, especially on datasets with class imbalance or high feature variation. Therefore, this study places particular emphasis on evaluating how tuning affects the performance of Decision Tree and Random Forest algorithms in genre classification tasks.

Taking these constraints into consideration, this research is directed to examine and compare the performance of Decision Tree and Random Forest algorithms in the task of music genre classification, both before and after hyperparameter tuning. The evaluation is conducted to observe the extent to which improvements in the preprocessing and parameter tuning stages can improve the accuracy of the model in recognizing genres based on numerical features. This approach is expected to provide a clearer picture of the effectiveness of each algorithm in dealing with complex and varied genre classification problems.

In line with the challenges in music genre classification, a number of previous studies have evaluated the effectiveness of machine learning algorithms in solving this problem. *Music Genre Classification Using Random Forest Model, Journal of Information and Computer Technology* used the Random Forest algorithm to predict genres based on numerical attributes and obtained an accuracy of 72%. This result shows that Random Forest is quite reliable in handling numerical feature variations, although its performance can still be improved through parameter optimization [2]

A Sparrow Search Algorithm (SSA) based optimization approach was developed to improve the performance of Random Forest. Test results showed an accuracy of 79.78% on test data, indicating that this optimization method is able to explore model parameters more effectively [3]. Meanwhile, a study combined using Short-Time Fourier Transform (STFT), and achieved an accuracy of 86%. This finding proves that spectral features can strengthen the representation of musical characteristics in the classification process [4].

In contrast, a study that employed MFCC features with the K-Nearest Neighbour algorithm for music genre classification achieved an accuracy of 80%, emphasizing that classification performance is highly influenced by feature selection and model choice. This also suggests that simpler models may require additional optimization steps to perform effectively on complex audio datasets [5]. The effectiveness of combining feature selection with hyperparameter tuning in improving Random Forest performance has been demonstrated, as reflected in the increase of the ROC-AUC score from 0.909 to 0.913 [6].

In comparison, a study evaluated the performance of Decision Tree and Random Forest algorithms in data classification using confusion matrix as a measurement tool. Although the context is not directly related to music genres, the results show that Random Forest produces higher accuracy than Decision Tree, which is 86.45% versus 84.30%. This difference is also reflected in the F1-score, where Random Forest recorded 90.2% while Decision Tree obtained 88.8%. These findings confirm that Random Forest tends to provide more stable and accurate classification results, especially on data with complex feature distributions [7].

From these studies, it can be concluded that Random Forest is a consistent and effective algorithm in genre classification tasks, especially when supported by relevant features and proper tuning. On the other hand, Decision Tree has advantages in terms of interpretability, but requires additional approaches in order to achieve sufficient accuracy. Therefore, this study attempts to review the effectiveness of both algorithms in the context of music genre classification based on numerical features.

II. METHODS

This research was conducted through a number of stages arranged systematically as shown in Figure 1, with the aim of classifying music genres using Decision Tree and Random Forest algorithms. The stages began with the data collection process, followed by preprocessing which included data cleaning, normalization, and class distribution balancing. Once the data is prepared, both models are built and trained, followed by hyperparameter tuning to improve model performance. Evaluation is done using metrics such as accuracy and f1-score. In the final stage, the model results before and after tuning are compared to see the impact of tuning on classification performance and determine the algorithm that gives the best results.

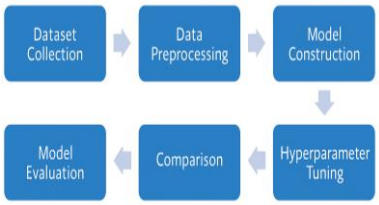

Figure 1 Research Overview

A. Data Collection

This research began with the data collection stage, which utilized a dataset titled Music Genre Classification obtained from the Kaggle platform (https://www.kaggle.com/datasets/purumalgi/music-genre-classification). The dataset contains over 114,000 song entries with various numerical features such as danceability, energy, loudness, tempo, and valence, along with genre labels as the classification target. For this study, a subset of the data consisting of eight specific genres was used, namely black-metal, comedy, gospel, grindcore, j-idol, samba, study, and tango. These genres were chosen as the sole focus of the classification task, and only entries belonging to these genres were included in the modeling process. Each genre contains approximately 993 to 997 songs, indicating that the data is relatively balanced across classes, although further balancing

techniques were still applied to ensure equal representation during training.

B. Data Preprocessing

The preprocessing stage is carried out to prepare the data before the model training process. The steps taken at this stage include:

1) *Removal of Irrelevant Columns*: In the initial stage of preprocessing, columns that are considered irrelevant to the classification process, such as *track_id*, *track_name*, *artists*, *album_name*, *album_release_date*, *popularity*, as well as time columns such as *duration_ms*, are removed because they do not have a direct contribution to the determination of song genres or are unique identities that cannot be generalized. For example, *track_name* and *artists* are string data that are specific to each song and do not reflect musical characteristics. Meanwhile, *track_id* only serves as a marker and cannot be used as a predictive feature. The removal of these columns aims to reduce data complexity, avoid redundant or noise features, and ensure that the model is only trained using attributes that have informative value for the music genre classification process. By leaving only numerical features that describe song characteristics such as *danceability*, *energy*, *acousticness*, and others, the model can be trained more focused and efficient.

2) *Checking and Handling Blank Values*: This check is important to ensure that there are no missing entries that could interfere with the model training process. In this research, the validation process is performed using the '?' value search method as well as the null check function on all data columns. The check results show that the dataset does not contain empty values or symbols that represent missing data. Therefore, no imputation or value replenishment process is required at this stage. The clean data condition from missing values allows the next preprocessing process to run more efficiently and ensures that the model is trained on complete and consistent data.

3) *Deleting duplicate values*: The presence of duplicates in the data can cause bias in model training, as machine learning algorithms can overadjust to repeated data, thus decreasing the generalizability of the model to new data. In this study, the duplication detection process is performed using the *drop_duplicates*() function on the DataFrame. As a result, 11,013 rows of duplicated data were found from a total of more than 114,000 initial entries. These entries were then removed to maintain data integrity and avoid overrepresentation of certain genres or features. The removal of duplicates is important so that the data distribution becomes more valid and is not dominated by the same samples, so that the model training process can take place fairly and accurately.

4) *Outlier Detection and Handling*: An outlier is defined as data that exhibits characteristics that differ significantly from the characteristics of other data in its group. Outliers may occur for a variety of reasons, including measurement errors, infrequent events, or other unanticipated factors [8]. In this study, outlier detection is performed using the Interquartile Range (IQR) method, where a value is considered an outlier if it is below Q1− 1.5× IQR or above Q3 + 1.5 × IQR. Figure 3 shows the boxplots for each feature before outlier handling. It can be seen that some features such as duration_ms, loudness, speechiness, tempo, and liveness have many outliers as indicated by dots outside the whisker boundaries on the diagram. For example, speechiness shows quite extreme outliers on the upper side, while loudness has extreme values on the lower side. To improve the distribution, outliers are handled by clipping to the lower and upper bounds of the IQR, so that values that exceed the threshold are replaced with the maximum or minimum allowed. With this method, the data is not discarded, but corrected so that the model can learn a more stable distribution. The results after outlier handling are shown in Figure 4, where the boxplots of each feature become more symmetrical and no longer have extreme outliers.

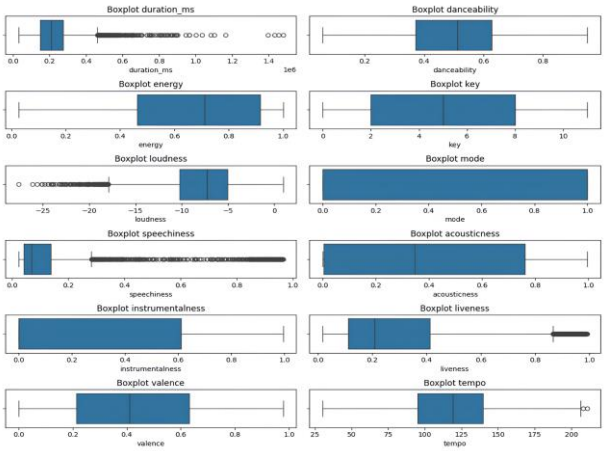

Figure 2 before outlier

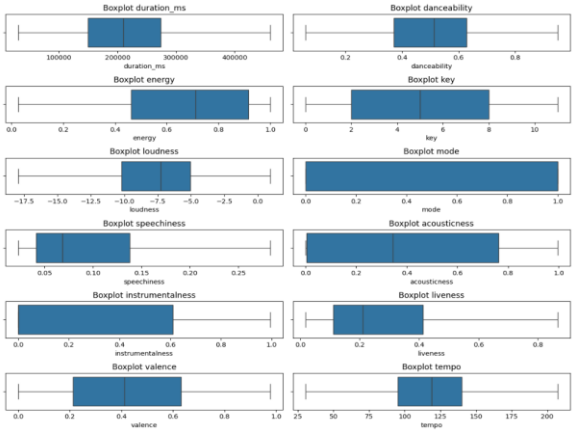

Figure 3 after outlier

A comparison between Figure 2 and Figure 3 shows that the data distribution has become more even and ready to be used in the model training process.

5) Genre Label Encoding: After the numerical data is prepared and the outliers are dealt with, the next step is to convert the target label (track_genre column) from categorical form to numerical form. This is done because machine learning algorithms such as Decision Tree and Random Forest can only process data in numeric form. This process is done using Label Encoding, where each genre is given a unique numeric representation. For example, the black-metal genre can be labeled 0, comedy becomes 1, and so on until tango becomes 7. Thus, genre labels that were originally text strings are converted into numerical representations between 0 and 7. This process is important so that the model can recognize each class as a distinct category without considering it a continuous variable.

6) *Normalization of Numeric Features:* This process aims to prevent large-scale features from dominating small-scale features [9]. Although algorithms such as Decision Tree and Random Forest are not very sensitive to scale, normalization is still performed to make the data distribution more uniform and the training process more stable. Normalization is performed using the StandardScaler method, which standardizes features in the dataset so that they have a uniform scale [10] by changing each feature to have a mean of 0 and a standard deviation of 1 [11]. This method is effective for equalizing the scale between features that were initially highly varied, such as the duration_ms feature, which has values in the hundreds of thousands, compared to the acousticness feature, which ranges from 0 to 1. After this process, each feature will be on a uniform scale, supporting a more balanced and efficient model learning process.

7) *Class Balancing:* The selected genres had a relatively balanced amount of data in the initial subset (approximately 993–997 songs per genre), but the process of dividing the data into training and test data using train_test_split caused an imbalance to reappear in the training data. This occurred because the data division process was performed randomly without explicitly maintaining class proportions. To address this issue, this study uses the SMOTE (Synthetic Minority Over-sampling Technique) technique. SMOTE is a resampling method that aims to balance class distributions. It takes samples from the minority class and adds synthetic samples to increase the amount of data in the minority class [12]. One advantage of SMOTE is that there is no loss of information within a class because there is no data reduction process. Additionally, SMOTE gives the minority class a greater chance of being studied, thereby improving its classification accuracy**.** Thus, classes with fewer instances are increased to achieve a balanced proportion with other classes, without the risk of overfitting due to data duplication. SMOTE was applied specifically to the training data, while the test data was left in its original state to ensure that the evaluation results reflect the model's generalization ability against the actual data distribution. After applying SMOTE, all genre classes in the training data have a balanced number of instances, enabling the Decision Tree and Random Forest algorithms to be trained fairly without bias toward the majority genre. This class balancing is an important step in improving the accuracy and stability of the classification model being built.

C. Model Construction

This stage explains the process of building a classification model using two supervised learning-based machine learning algorithms, namely Decision Tree and Random Forest. Both are methods that use a decision tree structure and are often used in solving classification problems.

Decision Tree Algorithms make decisions by splitting data into branches based on the most informative features[13]. Typically, feature selection is done using measures such as Gini Impurity or Information Gain to determine the best attributes for each split [14] This process of tree formation proceeds incrementally from the root of the tree until the data reaches a terminal state or a set depth limit. The main advantage of this algorithm is the ease of interpretation as the decision path can be traced from the root to the leaf nodes. However, a single decision tree has the risk of overfitting if parameters such as tree depth are not configured appropriately.

Random Forest is an extension of the decision tree algorithm that combines multiple decision trees randomly into a single prediction model that is more stable and accurate [15]. This ensemble technique builds hundreds of trees randomly, then generates a final prediction through voting for classification or averaging for regression. Each tree is constructed by selecting random samples from the training data set. Each split selection at each node of the tree uses a random subset of features [16]. This approach makes Random Forest more resistant to overfitting and more effective in recognizing patterns in data with complex and nonlinear structures. Therefore, the algorithm is widely used in various classification tasks, including music genre classification based on numerical data.

D. Hyperparameter Tuning

Hyperparameter tuning is the stage of adjusting parameters to get the best value [17]. In this research, the tuning performed using the GridSearchCV technique a hyperparameter optimization method used to determine the most optimal combination of hyperparameter values. The grid search method essentially involves testing all relevant parameter combinations in order to find the most effective parameters to improve model performance. For the Decision Tree model, the adjusted parameters *include max depth, min samples split, and min samples leaf*, with the best configuration found to be *max depth* = 10, *min samples split* = 2, and *min samples leaf* = 4. As for the Random Forest model, tuning was conducted on similar parameters, with the addition of *n_estimators*, and the best combination obtained

was *n_estimators* = 300, *max depth* = None, *min samples split* = 5, and *min samples leaf* = 1.

GridSearchCV works by trying each combination of values of these parameters and evaluating them using an accuracy metric on the validation data. This process is performed on the training data to avoid information leakage from the test data. Once the tuning is complete, the best model from each algorithm is selected based on the highest accuracy score obtained during the search process. The tuned model is then retested on the test data to find out how much the performance improvement is compared to the initial model. The results of the tuning process showed that parameter tuning had a significant impact on Decision Tree, which experienced a greater increase in accuracy than Random Forest. This confirms that simpler models such as Decision Tree rely heavily on proper parameter settings in order to provide optimal results.

E. Model Evaluation

Model evaluation aims to determine the extent to which the algorithm built is able to classify data correctly. In this research, the performance of the model is evaluated using a number of common classification metrics, namely accuracy, precision, recall, and f1-score. Each of these metrics provides a different perspective in assessing the quality of the model's predictions, especially when used on multi-class data such as music genres.

Before calculating the evaluation metrics, it is necessary to first understand four basic terms that are commonly used in the analysis of classification results. True Positive (TP) refers to the amount of data that is correctly predicted as a member of a positive class by the model, where the prediction matches the actual condition [18]. True Negative (TN) indicates the amount of data that is correctly predicted as not belonging to a class, in accordance with the actual conditions [19]. Meanwhile, false positive (FP) occurs when the model incorrectly predicts data as belonging to a class, even though the data is not actually part of that class [20]. Conversely, a false negative (FN) occurs when the model fails to detect data that should be included in the positive class, and instead classifies it as negative [21]. These four components form the basis of various model evaluation formulas, including the calculation of accuracy.

Accuracy measures the proportion of correct predictions to the entire amount of test data. The accuracy formula is written as:

$$Accuracy = \frac{TP + TN + FP + FN}{TP + TN}$$

However, accuracy alone is often not enough, especially when the amount of data between classes is not balanced. Therefore, precision and recall metrics are also used. Precision shows how precise the model is in classifying a class:

$$Precision = \frac{TP + FP}{TP}$$

Meanwhile, recall measures the model's ability to detect all data that should belong to that class:

$$Recall = \frac{TP + FN}{TP}$$

To provide a balanced assessment between precision and recall, f1-score is used, which is the harmonic mean of the two:

$$F1 - Score = 2x\frac{Precision + Recall}{Recall}$$

All these metrics are calculated for each class and then averaged using the macro average method to give equal weight to all classes. In addition to the numerical evaluation, the confusion matrix is also used as a visualization that illustrates the classification error between classes. Through this matrix, the distribution of correct and incorrect predictions can be seen more clearly, thus helping to evaluate the weaknesses and strengths of the model against the data structure used.

III. RESULT AND DISCUSSION

This research focused on eight music genres that were selectively chosen for classification purposes. These genres were chosen based on the relevance and diversity of audio characteristics between genres. The goal was to maintain an adequate balance of data in each class as well as ensuring the model was able to learn from the varied genre representations. Prior to the model building process, an initial analysis of the relationship between numerical features was conducted through the visualization of a correlation heatmap as shown in Figure 4.

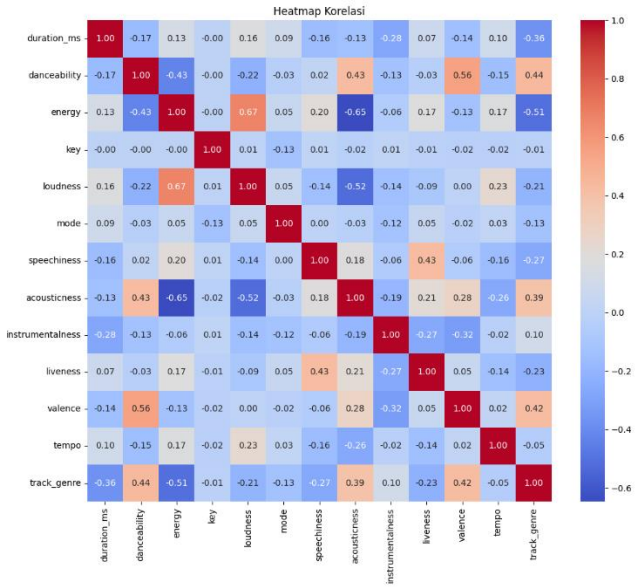

Figure 4 Heatmap Correlation

This heatmap provides an overview of the strength and direction of the relationship between features, including their relation to the genre label (track_genre). It can be seen that features such as energy, danceability, acousticness, and valence show a relatively strong correlation with genre, both

in positive and negative directions. Meanwhile, features such as key and mode have very weak correlations, indicating minimal information contribution in the classification process. This analysis serves as a starting point in understanding the role of each feature as well as a reference for the next preprocessing and modeling stages.Before building the model, the class distribution of the training data was analyzed. Following the preprocessing step, the class distribution of the training data was evaluated.

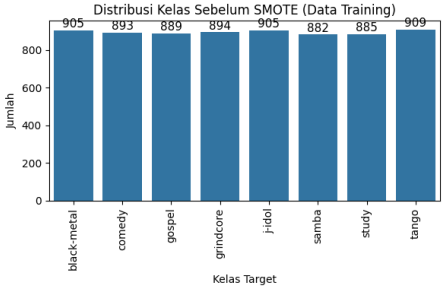

Figure 5 distribution before SMOTE

Figure 5 shows the number of samples for each genre before applying the SMOTE technique. In general, the eight genres selected have a fairly balanced amount of data, ranging from 882 to 909 entries. However, even though the class distribution appears quantitatively balanced, it does not mean that each genre is clearly separated in the feature space formed by audio characteristics such as danceability, energy, and loudness.

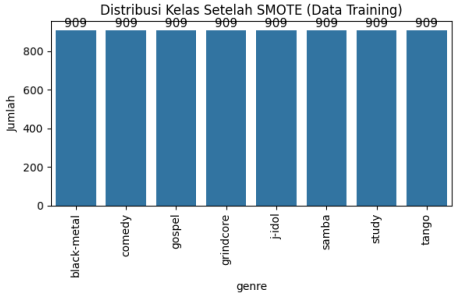

Figure 6 distribution after SMOTE

After the application of SMOTE, the class distribution in the training data becomes completely balanced, as shown in Figure 6. Each genre has the same number of samples, which is 909 entries. This balance aims to eliminate classification bias towards genres that previously had less data, so that the model can learn the representation of each class equally. With this uniform distribution, it is expected that the model training process will be fairer and able to increase the model's sensitivity to minority classes. However, it is important to note that even if the distribution is numerically balanced, the potential ambiguity between classes as described earlier can still affect the formation of optimal decision boundaries in the feature space.

Based on the data that has been thoroughly processed, including normalization, class balancing, and data cleaning, the initial stage of Decision Tree and Random Forest models were built using standard parameters with no additional tuning. Training is performed on data that has been fully pre-processed. The results of this process served as the baseline performance which was then used as a comparison to assess the performance improvement after tuning.

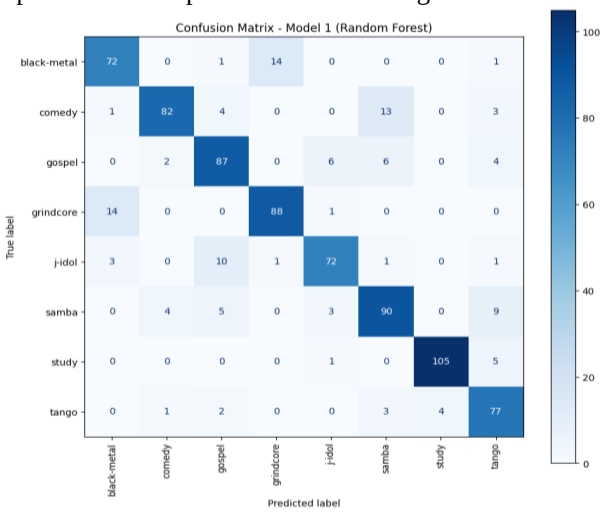

Figure 7 confusion matrix random forest before tuning

Figure 7 shows the confusion matrix for the Random Forest model. Most of the predicted values are on the main diagonal, indicating that the model is able to map song genres consistently. Genres such as *study* and *comedy* were classified correctly in almost all instances. However, there are still some errors, for example in the samba genre which is sometimes predicted as *gospel*. Meanwhile, Figure 6 shows the confusion matrix of the Decision Tree model.

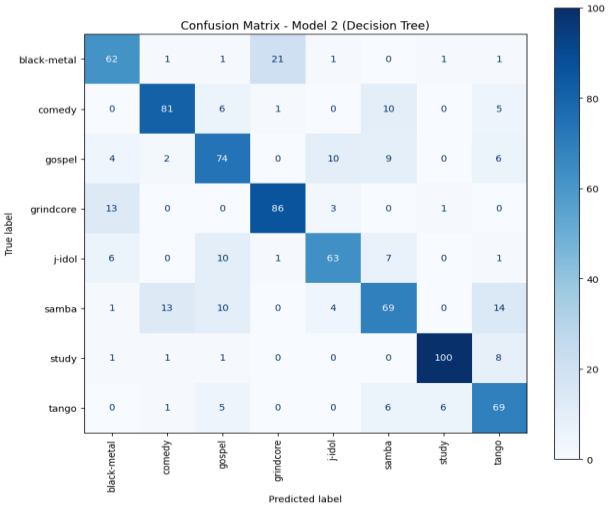

Figure 8 confusion matrix decision tree before tuning

Unlike the Random Forest, Figure 8 shows decision tree has more misclassification between genres. For example, the *blackmetal* genre is misclassified as grindcore several times, and the *tango* genre tends to be predicted as *samba*. This indicates that Decision Tree is more susceptible to complex

data variations because it relies on only one tree structure without a voting process as in Random Forest.

Classification Report:

	precision	recall	f1-score	support
black-metal	0.80	0.82	0.81	88
comedy	0.92	0.80	0.85	103
gospel	0.80	0.83	0.81	105
grindcore	0.85	0.85	0.85	103
j-idol	0.87	0.82	0.84	88
samba	0.80	0.81	0.80	111
study	0.96	0.95	0.95	111
tango	0.77	0.89	0.82	87
accuracy			0.85	796
macro avg	0.85	0.84	0.84	796
weighted avg	0.85	0.85	0.85	796

Figure 9 clasification report random forest before tuning

Figure 9 shows the classification report of the Random Forest model before hyperparameter tuning. The model recorded an overall accuracy of 85%, with a macro average f1-score of 0.84, and average precision and recall of 0.85 and 0.84 respectively. These results show that the model has a fairly balanced performance across almost all genre classes. Genre. The highest performance is shown by the study class, with an equally high precision and recall of 0.96 and 0.95, resulting in an f1-score of 0.95. Other genres that also performed well were comedy and j-idol, with f1-score of 0.85 and 0.84 respectively. Meanwhile, genres such as black-metal and gospel were in the f1-score range of 0.81, while the lowest scoring genres were samba (0.80) and tango (0.82).

Classification Report:

	precision	recall	f1-score	support
black-metal	0.71	0.70	0.71	88
comedy	0.82	0.79	0.80	103
gospel	0.69	0.70	0.70	105
grindcore	0.79	0.83	0.81	103
j-idol	0.78	0.72	0.75	88
samba	0.68	0.62	0.65	111
study	0.93	0.90	0.91	111
tango	0.66	0.79	0.72	87
accuracy			0.76	796
macro avg	0.76	0.76	0.76	796
weighted avg	0.76	0.76	0.76	796

Figure 10 classification report decision tree before tuning

Meanwhile, Figure 10 displays the classification report of the Decision Tree model. This model recorded an overall accuracy of 76%, with a macro average f1-score of 0.76, and the same average precision and recall values of 0.76. These figures show that Decision Tree's performance is still relatively moderate, especially when compared to Random Forest which has previously shown more stable results. The genre that showed the highest performance in this model was *study*, with a precision of 0.93, recall of 0.90, and f1-score of 0.91. Other genres such as *grindcore* and *comedy* also performed quite well, with f1-score 0.81 and 0.80, respectively. However, the model's performance tended to decline in some other genres. For example, *samba* only recorded an f1-score of 0.65 with a fairly low precision of 0.68, while *gospel* and *tango* recorded f1-score of 0.70 and 0.72 respectively. Overall, this classification report shows that while the Decision Tree is able to classify some genres quite well, it still struggles to handle genres with uneven feature distribution. This indicates that without additional parameter tuning, the single tree structure of the Decision Tree is not optimal for capturing the complexity of data across classes.

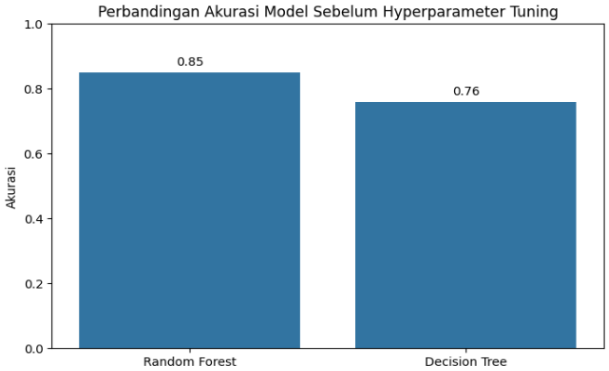

Figure 11 Model accuracy comparison before hyperparameter tuning

The accuracy comparison of the two models is visualized in Figure 11, Random Forest's performance is consistently higher, while Decision Tree is about eight percentage points below it, illustrating a significant performance gap before tuning is performed.

To optimize model performance, hyperparameter tuning was conducted using GridSearchCV with a predefined parameter grid. For the Decision Tree model, the tuning included max_depth, min_samples_split, min_samples_leaf, and criterion, while for the Random Forest model, it covered *n_estimators*, *max_depth*, *min_samples_split*, and *min_samples_leaf*. GridSearchCV performed an exhaustive search across all parameter combinations using 3-fold cross-validation. The tuning process required 17.42 seconds for the Decision Tree and 806.81 seconds for the Random Forest. The longer duration in Random Forest is due to its ensemble nature, which involves training multiple decision trees in parallel across parameter sets. This structure, while computationally heavier, offers higher stability through variance reduction and better generalization by averaging outputs from multiple trees, making it less sensitive to noise and outliers compared to a single-tree model.

After the hyperparameter tuning was completed, both models exhibited varying degrees of performance improvement. The Decision Tree model, in particular, showed notable gains, with accuracy increasing from 76% to 79%, and the macro average F1-score rising from 0.76 to 0.78. This suggests that even limited adjustments to parameters such as *max_depth*, *min_samples_split*, *and min_samples_leaf* can enhance the model's ability to handle more complex patterns. On the other hand, the Random Forest model, which had already demonstrated stable and high baseline performance, benefited less significantly from tuning. This indicates that the ensemble's initial configuration

was already well-suited for the classification task, possibly due to its inherent ability to generalize and reduce variance.

Classification Report:

	precision	recall	f1-score	support
black-metal	0.80	0.80	0.80	88
comedy	0.94	0.80	0.86	103
gospel	0.77	0.85	0.81	105
grindcore	0.84	0.85	0.85	103
j-idol	0.86	0.78	0.82	88
samba	0.79	0.80	0.79	111
study	0.96	0.95	0.95	111
tango	0.79	0.89	0.83	87
accuracy			0.84	796
macro avg	0.84	0.84	0.84	796
weighted avg	0.84	0.84	0.84	796

Figure 12 Classification report of random forest after hyperparameter tuning

After the hyperparameter tuning process, the Random Forest model produces the best parameter combination consisting of a max_depth value of None, n_estimators of 300, min_samples_split of 5, and min_samples_leaf of 1. The model accuracy decreased slightly from 85.1 percent to 84.0 percent, but the F1-score macro value remained stable at 0.84. This decrease in accuracy does not indicate a decrease in performance, but rather reflects a more even distribution of predictions across genres. Genres such as tango and gospel, which previously had lower evaluation scores, showed an improvement in recall. On the other hand, genres that had high performance from the beginning, such as study and comedy, still showed consistent performance. This balance of performance between classes can be observed through the classification report shown in Figure 13.

Classification Report:

	precision	recall	f1-score	support
black-metal	0.77	0.75	0.76	88
comedy	0.83	0.79	0.81	103
gospel	0.72	0.75	0.73	105
grindcore	0.80	0.83	0.82	103
j-idol	0.73	0.78	0.75	88
samba	0.72	0.66	0.69	111
study	0.94	0.94	0.94	111
tango	0.78	0.78	0.78	87
accuracy			0.79	796
macro avg	0.78	0.79	0.78	796
weighted avg	0.79	0.79	0.79	796

Figure 13 classification report of decision tree after hyperparameter tuning

In contrast, the Decision Tree model showed a more noticeable improvement after tuning. With optimized parameters (max_depth=10, min_samples_split=2, min_samples_leaf=4), the accuracy increased from 76% to 79%, and the macro F1-score rose from 0.76 to 0.78. This confirms that tuning has a greater positive impact on simpler models with higher sensitivity to parameter configuration. Genres like grindcore and comedy improved their F1-scores to 0.82 and 0.81 respectively.

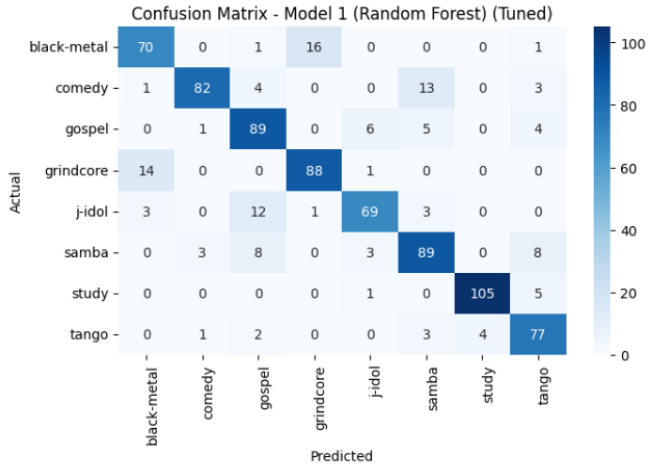

Figure 14 confusion matrix of random forest after hyperparameter tuning

Figures 14 and 15 display the confusion matrices after tuning for both models. In the Random Forest matrix, although minor misclassifications still occur in genres like j-idol and gospel, most predictions align closely with the true labels along the diagonal. Meanwhile, the Decision Tree matrix shows reduced misclassifications in genres such as samba and tango, indicating more refined decision boundaries, though some overlap between similar genres remains.

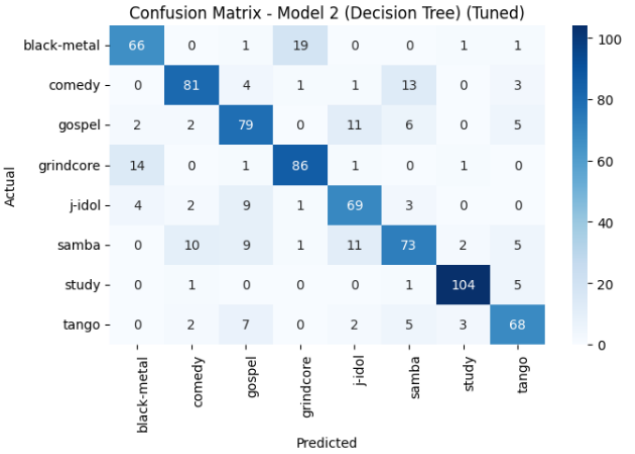

Figure 15 confusion matrix of decision tree after hyperparameter tuning

Figure 16 shows the accuracy comparison between Random Forest and Decision Tree after the hyperparameter tuning process. Random Forest still shows the highest accuracy of 84 percent, while Decision Tree increased to 79 percent. Although the difference in accuracy between the models is still visible, the tuning managed to narrow the gap. This suggests that parameter tuning has a greater impact on models with lower complexity. At the same time, the shift in Random Forest's performance also indicates a redistribution

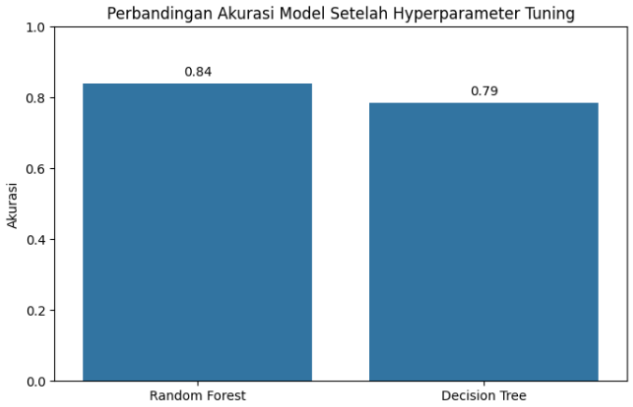

Figure 14 accuracy comparison model after hyperparameter tuning

Although tuning is generally aimed at improving performance, the decrease in Random Forest's accuracy from 85 percent to 84 percent can be explained through the redistribution of predictions that occurred after parameter tuning. Based on the classification reports and confusion matrix, tuning increased recall in low-performing genres such as tango and gospel, but slightly decreased precision in previously dominant genres such as j-idol. As a result, predictions became more balanced across classes, but overall accuracy decreased marginally. In the context of multi-class classification, this shows that tuning successfully reduces the dominance of major classes without compromising the overall performance of the model, which is reflected by the stable macro F1-score of 0.84.

IV. CONCLUSION

This research demonstrates the application of Decision Tree and Random Forest algorithms in audio numerical feature-based music genre classification. Data pre-processing, which includes duplication removal, empty value handling, normalization, outlier detection, and class balancing, successfully produces data that is ready to be used in modeling.

The results of the study showed that Random Forest consistently provided higher classification accuracy than Decision Tree. At the initial stage, Random Forest recorded an accuracy of 85 percent, while Decision Tree obtained 76 percent. After tuning with GridSearchCV, Decision Tree's accuracy increased to 79 percent, while Random Forest experienced a slight decrease to 84 percent. This decrease was due to the redistribution of predictions to low-performing classes, such as tango and gospel, which increased recall but slightly decreased precision in the dominant genres. This results in a more balanced distribution of predictions, which is reflected in the F1-score macro value that remains stable at 0.84.

In terms of time efficiency, the Random Forest tuning process takes significantly longer than Decision Tree, 806.81 seconds compared to 17.42 seconds. This shows that the complexity of the model greatly affects the training duration when a thorough parameter exploration is performed.

This finding confirms that the success of the model is not only determined by the complexity of the algorithm, but also by the data quality, tuning strategy, and training time efficiency. For further development, this research can be extended by exploring other algorithms and testing the model on real systems or other datasets to measure its robustness and generalization ability.

REFERENCES

[1] G. Ayu, V. Mastrika Giri, and L. Radhitya, "Musical Instrument Classification using Audio Features and Convolutional Neural Network," 2024. [Online]. Available: http://jurnal.polibatam.ac.id/index.php/JAIC

[2] J. Elektronik, I. K. Udayana, I. L. Simarmata, W. Supriana, and S. Kuta, "Music Genre Classification Using Random Forest Model," *Jurnal Elektronik Ilmu Komputer Udayana*, vol. 12, no. 1, pp. 2654–5101, 2023.

[3] Y. Wang and H. Chen, "An improved random forest model for music genre classification algorithm based on sparrow search algorithm," *Applied and Computational Engineering*, vol. 77, no. 1, pp. 84–90, Jul. 2024, doi: 10.54254/2755-2721/77/20240658.

[4] Merry Royanti Manalua, Made Agung Raharjaa, and F. Matematika dan Ilmu Pengetahuan Alam, "Analisis dan Klasifikasi Genre Musik Menggunakan Algoritma STFT dan Random Forest," *JNATIA*, vol. 3, no. 1, 2024.

[5] T. Pratiwi, A. Sunyoto, and D. Ariatmanto, "Music Genre Classification using K-Nearest Neighbor and Mel-Frequency Cepstral Coefficients," *Jurnal dan Penelitian Teknik Informatika*, vol. 8, no. 2, 2024, doi: 10.33395/v8i2.12912.

[6] F. Fakhriza *et al.*, "Optimalisasi Algoritma Random Forest Feature Selection Dan Hyperparameter Tuning Klasifikasi Genre Musik," 2025. [Online]. Available: https://www.kaggle.com/datasets/maharshipa

[7] E. Helmud, E. Helmud, F. Fitriyani, and P. Romadiana, "Classification Comparison Performance of Supervised Machine Learning Random Forest and Decision Tree Algorithms Using Confusion Matrix," *Jurnal Sisfokom (Sistem Informasi dan Komputer)*, vol. 13, no. 1, pp. 92–97, Feb. 2024, doi: 10.32736/sisfokom.v13i1.1985.

[8] G. Pu, L. Wang, J. Shen, and F. Dong, "A Hybrid Unsupervised Clustering-Based Anomaly Detection Method." [Online]. Available: http://creativecommons.org/licenses/by/4.0/

[9] Z. Jayidan, A. M. Siregar, S. Faisal, and H. Hikmayanti, "Improving Heart Disease Prediction Accuracy Using Principal Component Analysis (Pca) In Machine Learning Algorithms," *Jurnal Teknik Informatika (Jutif)*, vol. 5, no. 3, pp. 821–830, Jun. 2024, doi: 10.52436/1.jutif.2024.5.3.2047.

[10] N. T. Hastuti and F. Budiman, "Optimasi Klasifikasi Stunting Balita dengan Teknik Boosting pada Decision Tree," *Edumatic: Jurnal Pendidikan Informatika*, vol. 8, no. 2, pp. 655–664, Dec. 2024, doi: 10.29408/edumatic.v8i2.27913.

[11] M. Arif, maruf Setiawan, A. Dwi Hartono, M. Arif Ma, and ruf Setiawan, "Menggunakan Metode Machine Learning Untuk Memprediksi Nilai Mahasiswa Dengan Model Prediksi Multiclass," *Jurnal Informatika: Jurnal pengembangan IT*, vol. 10, no. 1, p. 2025, doi: 10.30591/jpit.v9ix.xxx.

[12] A. Nurhopipah and C. Magnolia, "Perbandingan Metode Resampling Pada Imbalanced Dataset Untuk Klasifikasi Komentar Program MBKM," *JUPIKOM*, vol. 1, no. 2, 2022.

[13] E. Oktavianti, M. Agustin, and R. Sari, "Implementasi Algoritma Decision Tree Dengan Fitur Seleksi Weight By Information Gain," 2023.

[14] R. N. Ramadhon, A. Ogi, A. P. Agung, R. Putra, S. S. Febrihartina, and U. Firdaus, "Implementasi Algoritma Decision Tree untuk

Klasifikasi Pelanggan Aktif atau Tidak Aktif pada Data Bank," 2024.

[15] R. Firdaus *et al*., "Implementasi Algoritma Random Forest Untuk Klasifikasi Pencemaran Udara di Wilayah Jakarta Berdasarkan Jakarta Open Data".

[16] R. Hidayat *et al*., "Implementasi Algoritma Random Forest Regression Untuk Memprediksi Penjualan Produksi di Supermarket," *SIMKOM*, vol. 10, no. 1, pp. 101–109, Jan. 2025, doi: 10.51717/simkom.v10i1.703.

[17] "Optimasi Hyperparameter Algoritma Support Vector Machine dalam Klasifikasi Penyakit β-Thalassemia." [Online]. Available: https://www.researchgate.net/publication/389288132

[18] H. Mahmud Nawawi, A. Baitul Hikmah, A. Mustopa, and G. Wijaya, "Model Klasifikasi Machine Learning untuk Prediksi Ketepatan Penempatan Karir," *Jurnal SAINTEKOM*, vol. 14, no. 1, pp. 13–25, Mar. 2024, doi: 10.33020/saintekom.v14i1.512.

[19] A. T. Setiawan, "Identifikasi Jenis Sampah Secara Otomatis Menggunakan Metode Convolutional Neural Network (CNN)."

[20] R. Fatmasari, W. Gata, N. Kusuma Wardhani, K. Prayogi, and M. Binti Husna, "Klasifikasi Sentimen Terhadap Kualitas Aplikasi Bahan Ajar Digital Akademik Universitas Terbuka di Google Play," *Jurnal SAINTEKOM*, vol. 14, no. 1, pp. 48–60, Mar. 2024, doi: 10.33020/saintekom.v14i1.591.

[21] A. Carolina Wibowo, S. Ardi Lestari, S. Informasi, F. Ilmu Komputer, and U. Duta Bangsa Surakarta, "Analisis Penggunaan Machine Learning Dalam Klasifikasi Penentuan Penyakit Jantung," vol. 9, no. 2, 2024.