Developing a framework for Music Genre Classification using Machine Learning Algorithms

Chaitra C[1], P Nagaraju[2]

[1,2]*Department of Telecommunication Engineering, RV College of Engineering, Bengaluru*

---***---

Abstract— Due to the rapid variants in music tracks, music genre classification plays an intriguing part in today's globe. We'd want to index them in order to have better access to them. Music genres enable access to a huge range of music. To simplify its architecture and index the same with the preference of the user, Machine Learning (ML) is used in the majority of contemporary music genre categorization systems. Hence, in this paper, we present a music dataset which includes ten different genres consisting of 100 samples of each genre. Different Machine Learning approaches are used to train and classify the system such as Decision Tree Classifier (DTC), Random Forest (RF), K-Nearest Neighbor (KNN) and Support Vector Machine (SVM). The performance of the classifiers is compared and evaluated based on certain evaluation parameters. Feature Extraction is the most crucial process for audio analysis. After research we choose Mel Frequency Cepstral Coefficient (MFCC), Pitch and Tempo as the required feature vector for the audio samples. Each algorithm is tested with a dataset of 1000 & 2000 music samples respectively. The suggested method categorizes music into several genres based on the feature vector characterization. The four algorithms used are based on supervised learning, where the output is predicted based on the training of the developed classifier. From the results, it is observed that the accuracy of genre prediction using RF is 97% & outperformed the rest when compared with state of the art for 2000 music samples.

Index Terms— **Music Genre Prediction, Machine Learning, Audio Processing, Feature Extraction, MFCC, Pitch, Tempo, Decision Tree Classifier, K-Nearest Neighbors, Random Forest, SVM.**

I. INTRODUCTION

MUSIC browsing and purchase from online music businesses has become a part of many people's everyday life all around the world. Users commonly divide their likes into genres like hip hop, pop, and disco. The majority of contemporary music, on the other hand, is not immediately classified to a genre. Due to the huge amount of current collections, automatic genre categorization is crucial for music organization, search, retrieval, and recommendation. Music categorization is regarded as a tough task due to the selection and extraction of suitable audio parts. While unlabeled data is generally available, music songs with appropriate genre tags are few. Feature extraction and categorization are the two essential phases in music genre categorization. Various aspects of the waveform are retrieved in the first stage. The characteristics retrieved from the training data are used to construct a classifier in the second stage. There have been several ways of categorizing music into various genres. With so much music available in internet is required for the genre classification of music. Different kinds of feature extraction are used in each implementation. Some people use wood and MFCC as classification criteria, while others use beat, pitch, and beat, among other things. The process of extracting characteristics differs from person to person. We propose an approach for music genre classification using four machine learning algorithms such as Decision Tree Classifier (DTC), K-nearest neighbor (KNN), Random Forest (RF) and Support Vector Machine (SVM).

MFCC, Pitch, and Tempo are used to extract musical features. In the next sessions, a more complete explanation will be given. These retrieved characteristics are used as input to the training, which results in prediction. We examine music from 10 different genres as part of our research. The entire software is written in Python. The average accuracy derived by combining Pitch, MFCC, and Tempo are 97%.

Many academics have sought to understand the musical parameters as well as the methodologies for categorizing them into various genres. This section will largely look at a few of the previous analyses. Tzanetakis and Cook [1] were the first to use a machine learning algorithmic approach to classify expressive styles. They generated the GTZAN dataset, which is still used as a standard for genre categorization. In [2], Changsheng Xu et al. demonstrated how to use support vector machines (SVM) to accomplish this task. For expressive style categorization, the authors employed supervised learning algorithms. Scaringella et al. [3] provides a thorough examination of the many alternatives and categorization methodologies used in the expressive style categorization. Riedmiller et al. [4] used unsupervised learning to develop a choice vocabulary. Scheirer [5.] describes a time period beat following method for audio signals containing music. In this technique, a filter bank is paired with a network of combination filters to follow signal periodicities and offer a most beat and efficiency impact. [6] outlines a time period beat-following system with a multitudinous agent architecture that simultaneously follows many beat theories. By obtaining a lot of information out of the initial dataset, GTZAN, Tao in [7] illustrates how to employ constrained Boltzmann machines to get better outcomes than a generic multilayer neural network. In this work, a data distribution flaw in the dataset is discussed, and it is demonstrated that it makes it difficult to appropriately identify four categories using just the GTZAN dataset. This research recommends using MFCC spectrograms for song preparation as well. Aaron et al. [8] preprocess the tracks with MFCC spectrograms. This work does not focus on genre classification, but rather on song recognition and music suggestion. However, it was worth noting that he or she used convolutional neural networks with ReLU activation on music samples preprocessed as MFCC spectrograms.

The Organization of rest of the paper is as follows: Section II describes the feature vectors used for extraction. Section III gives details of the Machine learning algorithms incorporated for the classification of music genres. Section IV proposes the methodology used. Section V analyses the results achieved and evaluation of the developed classifiers based on their performance. Section VI presents the Conclusions & Future work regarding the proposed methodology

II. FEATURE EXTRACTION

The inputs to the algorithm are collected using 3 features namely MFCC, Pitch and Tempo. Each features are extracted using librosa: a python library.

A. MFCC

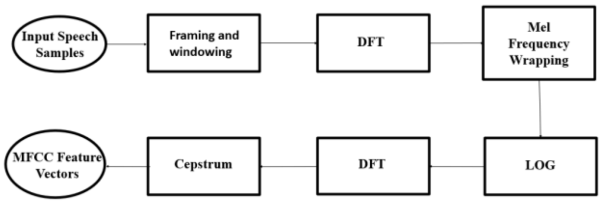


Fig. 1: Block diagram for MFCC extraction

Framing and Windowing: A continuous voice stream is divided into N-sample frames, with M samples between consecutive frames. Spectrum refers to the end outcome of this phase. The extraction of MFCC is shown in Fig. 1.
Mel Frequency Wrapping: Any tone with a frequency of f has its pitch evaluated on the Mel scale. This scale uses linear spacing for frequencies under 1000Hz and a logarithmic scale for frequencies beyond 1000Hz.
Cepstrum: Time conversion from log-mel scale. This yields MFCC features, which are an excellent representation of a signal's local spectral qualities.

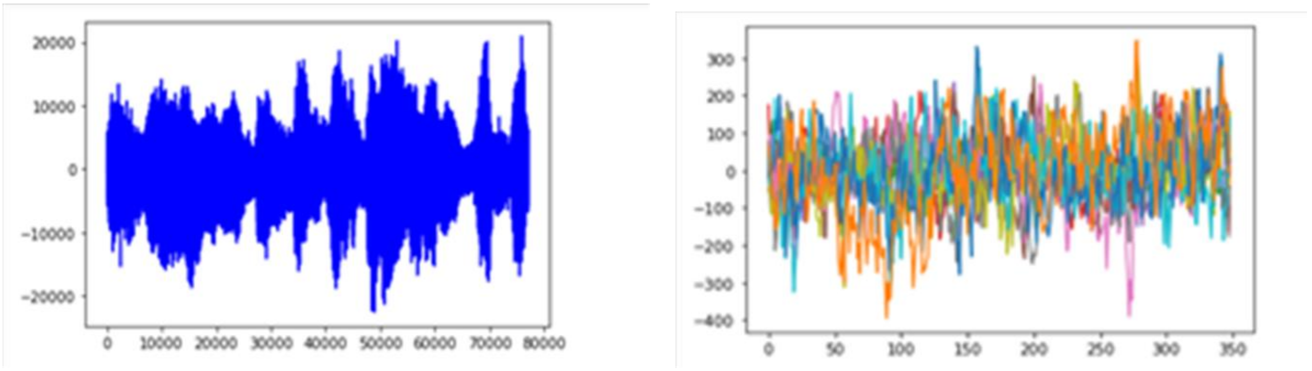
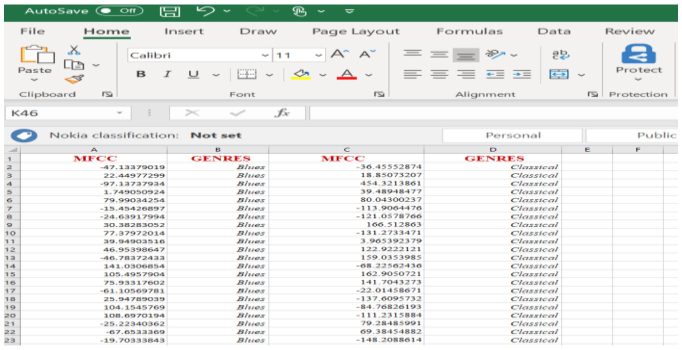


Fig. 2: (*from the left*) Spectrum of an example audio file and MFCC extraction of the same.

We can observe from Fig. 2, the diversity of each data point. The values are spread across the feature space and hence we need to choose an algorithm that provides good accuracy for randomized dataset. The results will be stored to Excel file as shown in Fig.3.



	A	B	C	D	E	F
1	MFCC	GENRES	MFCC	GENRES		
2	-47.13379019	Blues	-36.45552874	Classical		
3	22.44977299	Blues	18.85073207	Classical		
4	-97.13737934	Blues	454.3213861	Classical		
5	1.749050924	Blues	39.48948477	Classical		
6	79.99034254	Blues	80.04300237	Classical		
7	-15.45426897	Blues	-113.9064476	Classical		
8	-24.63917994	Blues	-121.0578766	Classical		
9	30.38283052	Blues	166.512863	Classical		
10	77.37972014	Blues	-131.2733471	Classical		
11	39.94903516	Blues	3.965392379	Classical		
12	46.95398647	Blues	122.9222121	Classical		
13	-46.78372433	Blues	159.0353985	Classical		
14	141.0306854	Blues	-68.22562436	Classical		
15	105.4957904	Blues	162.9050721	Classical		
16	75.93317602	Blues	141.7043273	Classical		
17	-61.10569781	Blues	-22.01458671	Classical		
18	25.94789039	Blues	-137.6095732	Classical		
19	104.1545769	Blues	-84.76826193	Classical		
20	108.6970194	Blues	-111.2315884	Classical		
21	-25.22340362	Blues	79.28485991	Classical		
22	-67.6533369	Blues	69.38454882	Classical		
23	-19.70333843	Blues	-148.2088614	Classical		

Fig.3. Extracted features of MFCC stored in excel file.

B. PITCH

The modified autocorrelation pitch detector based on clipping approach is used to detect pitch. Each audio segment is separated into smaller frames of 20 to 480 milliseconds using a low-pass filter with a cutoff frequency of 900 Hz. For each frame, a clipping threshold is computed by Eq. (1);

$$C_t = R \min[C_{t_1} - C_{t_2}] \qquad (1)$$

where R is typically equal to 0.64, & C_{t_1} and C_{t_2} are the maximum amplitude in the first and last third of the frame respectively. The total number of frames that have a pitch is calculated and the pitch ratio is calculated by Eq. (2);

$$Pitch\ Ratio = \frac{N_p}{N_f} \qquad (2)$$

where N_p is the total numbers of frames that have a pitch and N_f is the total number of the frames.

Every individual's human pitch range has a distinctive value. The larynx controls the pitch range. Males have a pitch range of 50 to 300 Hz, whereas females have a pitch range of 120 to 600 Hz. Stress, emotions, and intonation cause the pitch to fluctuate. The misclassification of various sorts of non-speech can be avoided by limiting the pitch search to the human range.

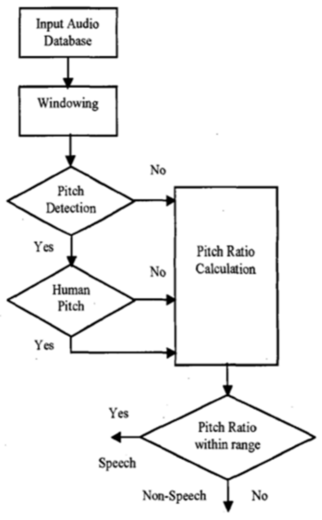
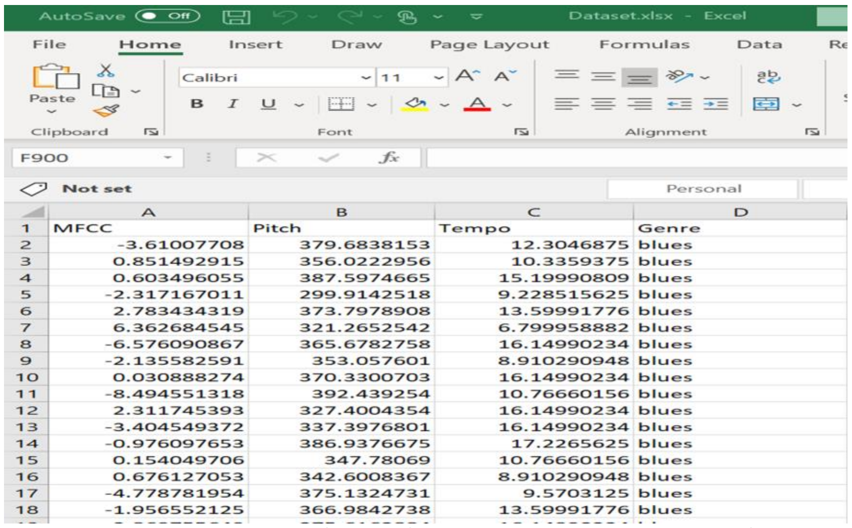


Fig. 4. Pitch algorithm



	A	B	C	D
1	MFCC	Pitch	Tempo	Genre
2	-3.61007708	379.6838153	12.3046875	blues
3	0.851492915	356.0222956	10.3359375	blues
4	0.603496055	387.5974665	15.19990809	blues
5	-2.317167011	299.9142518	9.228515625	blues
6	2.783434319	373.7978908	13.59991776	blues
7	6.362684545	321.2652542	6.799958882	blues
8	-6.576090867	365.6782758	16.14990234	blues
9	-2.135582591	353.057601	8.910290948	blues
10	0.030888274	370.3300703	16.14990234	blues
11	-8.494551318	392.439254	10.76660156	blues
12	2.311745393	327.4004354	16.14990234	blues
13	-3.404549372	337.3976801	16.14990234	blues
14	-0.976097653	386.9376675	17.2265625	blues
15	0.154049706	347.78069	10.76660156	blues
16	0.676127053	342.6008367	8.910290948	blues
17	-4.778781954	375.1324731	9.5703125	blues
18	-1.956552125	366.9842738	13.59991776	blues

Fig. 5. Extracted features of MFCC, pitch & tempo stored in excel file

Parameters specification

The following are the requirements for each audio signal in the database:

- 0.5 seconds in length.
- 16000 Hz sampling rate.
- 16 bits of resolution

The Fig. 4 is a flowchart that represents the pitch extraction from the audio samples.

C. TEMPO

The features extracted through tempo are extracted using librosa python package. The MFCC, pitch, tempo (features extracted for the audio samples) are stored in excel file as shown in Fig. 5.

III. MACHINE LEARNING ALGORITHMS

A. Decision Tree Classifier (DTC)

A decision tree uses a tree structure to create categorization models. It divides the data into decision nodes and leaf nodes, which are smaller and smaller portions [8]. Two or more branches reflect the values for the property being checked in a decision node. The ultimate choice is represented by the Leaf node. The uppermost decision node is known as the root node. Decision trees can evaluate both numerical and categorical data as shown in Fig. 6.

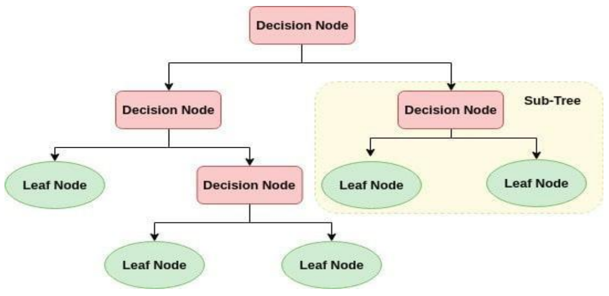


Fig. 6. Decision tree

There are two important parameters that are considered during node splitting namely Mean squared error and Gini index. Mean squared error (MSE) is the square of the deviation between the target value and the predicted value given by EQ. (3);

$$MSE = \frac{1}{N}\sum_i (y^i - y_t)^2 \qquad (3)$$

where y^i is the true target value, y_t is the predicted target value.
The developed model performs better with a lower MSE.

Minimum MSE improves the accuracy of the model. The Gini index is used to indicate the purity of the node in a decision tree. When more than one feature is engaged in the decision-making process, it is necessary to determine the relevance and importance of each feature on the node, assigning the most relevant feature to the node and dividing the nodes down. The amount of uncertainty diminishes as we move down the tree, resulting in a better categorization or best split at each node. The Gini index G is given by the Eq. (4)

$$G = \sum_{k=1}^{K} pm_k\,(1 - pm_k) \qquad (4)$$

where pm_k is the probability of an object being classified into a particular class. Gini index ranges from 0 to 1. Lower the value of Gini index, better is the node purity.

B. K-Nearest Neighbors

The K-Nearest Neighbor algorithm (KNN) is a classification approach that is non-parametric. The majority of its neighbors vote to classify an object. It gathers information from a training dataset and applies it to create predictions about new objects. A class or a category is the result. The item is placed in the class with the most members among its K closest neighbors, where K is a positive integer. KNN calculates the distance between data points. A common weighting scheme consists in giving each neighbor a weight of 1/d, where d is the distance to the neighbor. The distance is calculated based on Euclidian Formula [9] given by Eq. (5) and Eq. (6).

$$d(p,q) = d(q,p) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2} \qquad (5)$$

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \qquad (6)$$

The test data are categorized into a particular class based on the closest training samples present in the feature space. Fig. 7 represents a KNN classification.

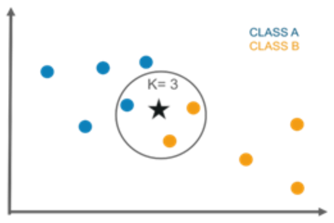


Fig. 7: KNN Classification, K = 3, Star is assigned to Class B as 2 neighbors belong to Class B in the given feature space.

C. *Random Forest*

Random Forest (RF) algorithm is a supervised classification algorithm. Larger the number of trees, more accurate is the result. Fig. 8 depicts a common representation of RF.

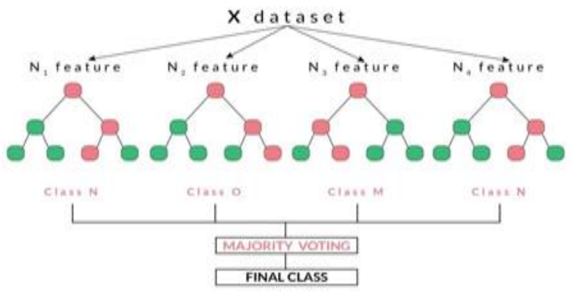


Fig. 8: RF Algorithm

The algorithm randomly selects ‘k’ features from total ‘m’ features where k << m. Among the ‘k’ features, it calculates the node ‘d’ using the best split point. Once the split point is known, it splits the node into daughter. It repeats the steps until ‘l’ number of nodes has been reached. A forest is built by repeating the entire process for ‘n’ number times to create ‘n’ number of trees. With the RF classifier created, the predicted output of each randomly created decision tree is stored individually. It then calculates the votes for each predicted output. The final class is then assigned to the highest voted predicted target [10].

For each decision tree, the algorithm calculates node importance using Gini Index, assuming it is a binary tree given by the Eq. (7)

$$ni_j = w_j C_j - w_{left(j)} C_{left(j)} - w_{right(j)} C_{right(j)} \qquad (7)$$

where ni_j is the importance of node j, w_j is the weighted number of samples reaching node j & $C_{(j)}$ gives the impurity value of node j. The importance for each feature on a decision tree is then calculated by the formula (8);

$$fi_i = \frac{\sum_{j:node\ j\ splits\ on\ feature\ i} ni_j}{\sum_{k \in all\ nodes} ni_k} \qquad (8)$$

where fi_i gives the importance of feature i. It then involves normalization of fi_i between 0 and 1 by dividing by the sum of all feature importance values; given by the formula;

$$norm\ fi_i = \frac{fi_i}{\sum_{j \in all\ features} fi_j} \qquad (9)$$

For Random Forest, it calculates the average over all the trees. It is given by the ratio of sum of the feature’s importance value on each trees and the total number of trees T.

$$RF\ fi_i = \frac{\sum_{j \in all\ trees} norm\ fi_{ij}}{T} \qquad (10)$$

D. *Support Vector Machine*

The Support Vector Machine (SVM) is a supervised machine learning methodology for solving classification and regression issues. It is, however, mostly used to tackle categorisation problems.

- Each data item is represented as a point in n-dimensional space (where n is the number of features you have), with the value of each feature being the SVM algorithm's value for a certain position.

- Then we accomplish classification by locating the hyper-plane that clearly distinguishes the two classes. The SVM algorithm is shown in Fig.9.

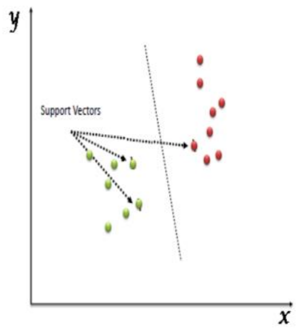
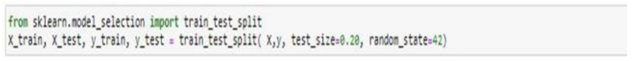


Fig.9. SVM Algorithm

IV. METHODOLOGY

A collection of audio tracks with comparable sizes and frequency ranges. The GTZAN genre categorization dataset is the most often used for determining music genres. The features are extracted from the above mentioned techniques using respective algorithm to aid the training of the classifiers. The database consists of the genre labels that is stored in an Excel file along with the MFCC, pitch and tempo. There are 2 sets of data each with 1000 & 2000 values respectively. These features will improve the model accuracy.

The features are given as input to the algorithms called as independent variables. The target output for each audio file is used for training the model which are termed as dependent variables (Genre labels). The classifiers are created by setting certain parameters and 80% of the database is used to train the model as shown in Fig. 10. The models are trained for each dataset separately.



```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split( X,y, test_size=0.20, random_state=42)
```

Fig. 10. Dataset division. 0.20 indicated the normalized value for 20% test data.

Remaining 20% database called test datasets are used to evaluate the model. The model is evaluated on the basis of Accuracy, Precision, Recall & F1 score. The accuracy of different models is compared and the best model for the prediction of the music genre is chosen. The Flowchart for the proposed methodology is shown in Fig. 11.

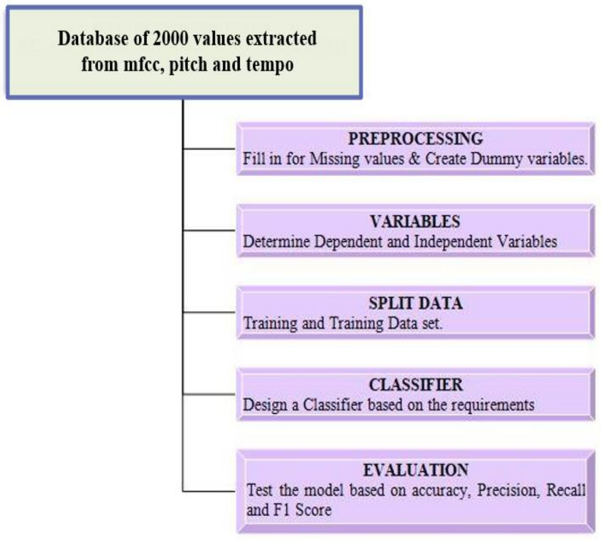

Database of 2000 values extracted from mfcc, pitch and tempo
PREPROCESSING
Fill in for Missing values & Create Dummy variables.
VARIABLES
Determine Dependent and Independent Variables
SPLIT DATA
Training and Training Data set.
CLASSIFIER
Design a Classifier based on the requirements
EVALUATION
Test the model based on accuracy, Precision, Recall and F1 Score

Fig. 11. Flowchart for the proposed methodology

V. RESULTS & ANALYSIS

Design the classifiers based on required parameters for the 2 datasets with 1000 & 2000 audio files. For the Decision tree classifier, we set the depth of the tree, minimum samples required to split a node, and minimum number of samples required in the leaf cell appropriately. Table I shows the parameters chosen for the given datasets. These parameters are set to avoid the complexity of trees and over-fitting. Tree pruning is an efficient way to remove the nodes that doesn't effectively contribute to the Accuracy of the model. Similarly, we also customize the K value required for the KNN Classifier. Table II shows the assigned K values to the respective datasets. The number of estimators is set to 1000 for the Random Forest algorithm for all the data sets and Support Vector Machine with C as parameter. These parameters control the accuracy of the classifiers.

TABLE I. PARAMETERS OF THE DECISION TREE CLASSIFIER.

Dataset (No. of users)	Maximum depth	Minimum samples to split the node	Minimum samples required in the leaf cell
1000	07	60	20
2000	07	90	30

TABLE II. K VALUE ASSIGNED FOR KNN CLASSIFIER.

Dataset (No. of users)	K
1000	07
2000	07

As discussed earlier, the evaluation parameters are Accuracy, Precision, Recall, and F1 score. Accuracy is the number of correctly predicted data points out of all the data points. In multiclass classification, Accuracy represents the proportion of correctly classified classes. The number of true positives and true negatives divided by the number of true positives, true negatives, false positives, and false negatives equals the number of true positives and true negatives. The equation for accuracy is given by Eq. (11)

$$Accuracy = \frac{True\ Negavtive+True\ positive}{True\ Negative+True\ Positive+False\ Positive+False\ Negative} \quad (11)$$

Precision is defined as the percentage of expected positive instances that are accurately detected. When the cost of False Positives is high, it is therefore beneficial. It is given by the Eq. (12)

$$Precision = \frac{True\ Positive}{True\ Positive+False\ Positive} \quad (12)$$

The recall score is a measurement of how many positive instances were properly identified out of all the positive instances. When the price of False Negatives is large, this is critical. It is given by the Eq. (13)

$$Recall = \frac{True\ Positive}{True\ Positive+False\ Negative} \quad (13)$$

The harmonic mean of accuracy and recall is defined as the F1 score. It's a statistical metric for evaluating performance. To put it another way, an F1-score is the average of a person's performance based on two factors: accuracy and recall.

$$F1 = \frac{2(Precision*Recall)}{(Precision+Recall)} \quad (14)$$

The maximum value for these parameters is 1 and lowest is 0. The results of the performance of the classifier based on the datasets are tabulated below in Table III and IV respectively.

TABLE III. THE EVALUATION OF THE MODEL FOR 1000 AUDIO

Classifier	Accuracy	Precision	Recall	F1
DTC	0.89	0.9049	0.89	0.8896
KNN	0.85	0.8580	0.85	0.8465
RF	0.88	0.8900	0.88	0.8782
SVM	0.88	0.8843	0.88	0.8778

TABLE IV. EVALUATION OF THE MODEL FOR 2000 AUDIO

Classifier	Accuracy	Precision	Recall	F1
DTC	0.9375	0.9389	0.9375	0.9366
KNN	0.8825	0.8845	0.8825	0.8827
RF	0.97	0.9710	0.97	0.9698
SVM	0.925	0.9266	0.925	0.9250

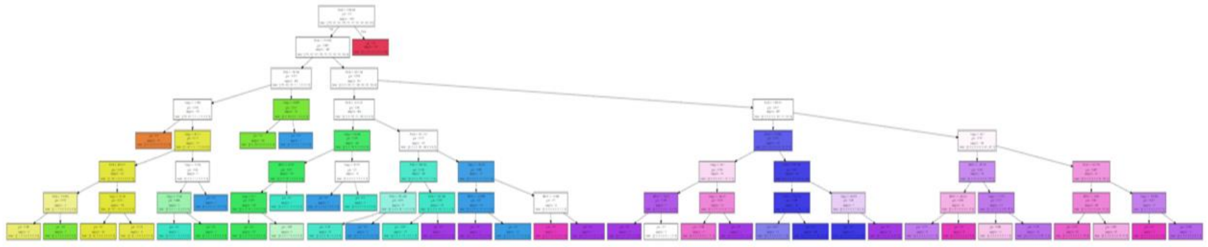


Fig. 12: Decision tree created for 1000 audio files.

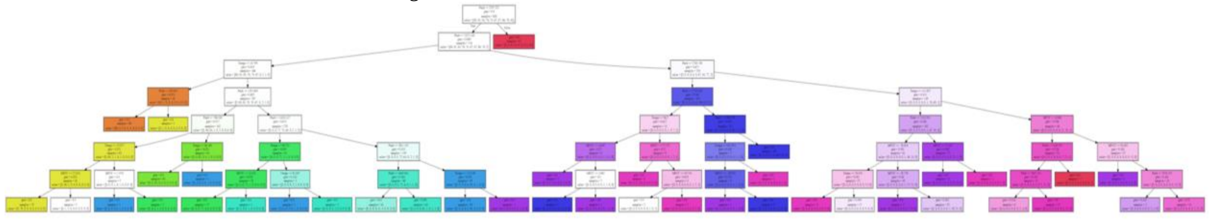


Fig. 13: Decision tree created for 2000 audio files.

Thus we infer from the above tables that, the evaluation parameters improve with the increase in the number of audio files in the dataset. The classifier best fits when it has larger training samples. The Music Genre Classification is based on the Feature Extraction and is given utmost importance. Among the 4 designed classifiers for 2000 users, RF classifier has an accuracy of 0.97 that outshines the rest. Fig. 12 and Fig. 13 represents the decision tree generated by the DTC classifier for 1000 and 2000 audio files respectively.

VI. CONCLUSION & FUTURE SCOPE

Classification of Music-Genres is crucial for a person to choose the preference based on their interest. Hence, in this project, a music classifier was developed and performance of four machine learning algorithms such as Decision Tree Classifier, K-Nearest Neighbor (KNN), Support Vector Machine (SVM) and Random Forest (RF) were compared. The four algorithms are based on supervised learning, where the output is predicted based on the training of the developed classifier. Each algorithm was tested for a database 1000 & 2000 users respectively.

The prediction of genres is based on the Feature-Extraction values and is given utmost importance. Among the 4 designed classifiers for 1000 music files, DTC classifier has highest accuracy of 0.89. And, for 2000 music files, RF classifier has highest accuracy of 0.97. The classifiers are also evaluated based on their accuracy, precision, recall & F1 scores.

Hence, we conclude that RF having an accuracy of 97% for 2000 items is the best algorithm among the other tested models for prediction of music-genres based on analyzing the feature-extraction values of the MFCC, pitch and tempo.

In this paper, evaluation is done on the algorithms based on maximum of 2000 audio signals. The database can be improvised further with relevant preprocessing. Algorithms like Convolution Neural Network (CNN) use a much larger database but provide much accurate classification. Also, we can increase the number of features for each genres and check for their dependency to improve our classification. Further, the database can be modelled and fit into different classifier such as *sklearn*, which will automatically predict the results. This can be used in production development.

REFERENCES

[1] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," in IEEE Transactions on Speech and Audio Processing, vol. 10, no. 5, pp. 293-302, July 2002, doi: 10.1109/TSA.2002.800560.
[2] Manuel Blum, Jost Tobias Springenberg, Jan Wulfing, ¨ and Martin Riedmiller. A Learned Feature Descriptor for Object Recognition in RGB-D Data. In IEEE International Conference on Robotics and Automation (ICRA), 2012.
[3] Kaichun K. Chang, Jyh-Shing Roger Jang, and Costas S. Iliopoulos. "Music genre classification via compressive sampling" In ISMIR, pages 387–392 in International Society for Music Information Retrieval, 2010.
[4] Adam Coates, Honglak Lee, and Andrew Y. Ng. "An analysis of single-layer networks in unsupervised feature learning". In Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS), 2011.
[5] Philippe Hamel and Douglas Eck. "Learning features from music audio with deep belief networks". In Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR), pages 339– 344, August 2010.
[6] Philippe Hamel, Simon Lemieux, Yoshua Bengio, and Douglas Eck. "Temporal pooling and multiscale learning for automatic annotation and ranking of music audio". In Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR11), 2011.
[7] Mikael Henaff, Kevin Jarrett, Koray Kavukcuoglu, and Yann LeCun. "Unsupervised Learning of Sparse Features for Scalable Audio Classification". In Proceedings of the International Society for Music Information Retrieval, 2011.
[8] Aapo Hyvarinen and Erkki Oja. "Independent component analysis: algorithms and applications". Neural networks, 13(4-5):411–30, 2000.
[9] Honglak Lee, Yan Largman, Peter Pham, and Andrew Y. Ng. "Unsupervised feature learning for audio classification using convolutional deep belief networks". In Advances in Neural Information Processing Systems 22, pages 1096–1104, 2009.
[10] Yannis Panagakis, Constantine Kotropoulos, and Gonzalo R. Arce. "Music genre classification using locality preserving non-negative tensor factorization and sparse representations". In ISMIR, pages 249–254. International Society for Music Information Retrieval, 2009.
[11] Andrew Saxe, Pang Wei Koh, Zhenghao Chen, Maneesh Bhand, Bipin Suresh, and Andrew Ng. "On random weights and unsupervised feature learning". In Proceedings of the 28th International Conference on Machine Learning (ICML-11), ICML '11, pages 1089– 1096, June 2011.
[12] Jan Schluter and Christian Osendorfer. Music Simi- ¨ larity "Estimation with the Mean-Covariance Restricted Boltzmann Machine". In Proceedings of the 10th International Conference on Machine Learning and Applications (ICMLA 2011), 2011.
[13] A. Schoerkhuber, C. and Klapuri. "Constant-Q transform toolbox for music processing".