Music Genre Classification and Recommendation

Milestone : Project Report

Group 5
Anjali Kshirsagar
Saurabh Sonawane

Contact Numbers
857-313-1931 (Anjali Kshirsagar)
617-820-0820 (Saurabh Sonawane)

Email
kshirsagar.an@northeastern.edu
sonawane.sau@northeastern.edu

Percentage of Effort Contributed by Student 1 : 50 %

Percentage of Effort Contributed by Student 2 : 50 %

Signature of Student 1 : 

Signature of Student 2 : 

Submission Date : April 23, 2023

PROBLEM SETTING:

Music genre and recommendation are important for creating a personalized and enjoyable listening experience for users. By understanding an individual's preferences and recommending music based on those preferences, music services can create a unique listening experience for each user. Music genre and recommendation also help users discover new music that they may not have otherwise found, as well as organize their music library. Additionally, music genre and recommendation play an important role in marketing for the music industry by promoting new music and expanding audiences. Overall, music genre and recommendation benefit both users and the music industry by creating a personalized and organized listening experience while also promoting new music and expanding audiences.

PROBLEM DEFINITION:

Music is a powerful and universal form of communication that connects people from all walks of life, transcending language and cultural barriers. It has the ability to evoke strong emotions and bring back memories, serving as a source of comfort and escape. The vast and diverse musical landscape can be overwhelming, making it challenging for music lovers to discover new pieces to enjoy.

Music genre classification helps in navigating this landscape by grouping music into different genres, providing a roadmap for listeners. This allows them to explore new artists, styles, and compositions with ease and can also assist in narrowing down choices when searching for music that matches their current mood or taste.

In this project, we take a unique approach to music recommendations and genre classification, involving the use of acoustic features extracted from raw music through digital signal processing methods to make predictions, without considering the user's music profile. This approach aims to provide music lovers with a valuable framework for discovering new and exciting pieces to enjoy, no matter what their mood or taste. In conclusion, music genre classification plays a crucial role in helping music enthusiasts navigate the musical landscape and find the perfect fit.

DATA SOURCES:

Kaggle : https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification

DATA DESCRIPTION :

The first known application of a dataset in the realm of genre classification was in the landmark paper, "Musical Genre Classification of Audio Signals," by G. Tzanetakis and P. Cook, published in the IEEE Transactions on Audio and Speech Processing in 2002.

The dataset, known as the GTZAN Genre Dataset, encompasses 1000 audio tracks, each lasting 30 seconds. It encompasses 10 distinct genres: Blues, Classical, Country, Disco, Hiphop, Jazz, Metal, Pop, Reggae, and Rock, each with 100 tracks. The audio files are 22050 Hz Mono, 16-bit .wav format.

DATA EXPLORATION:

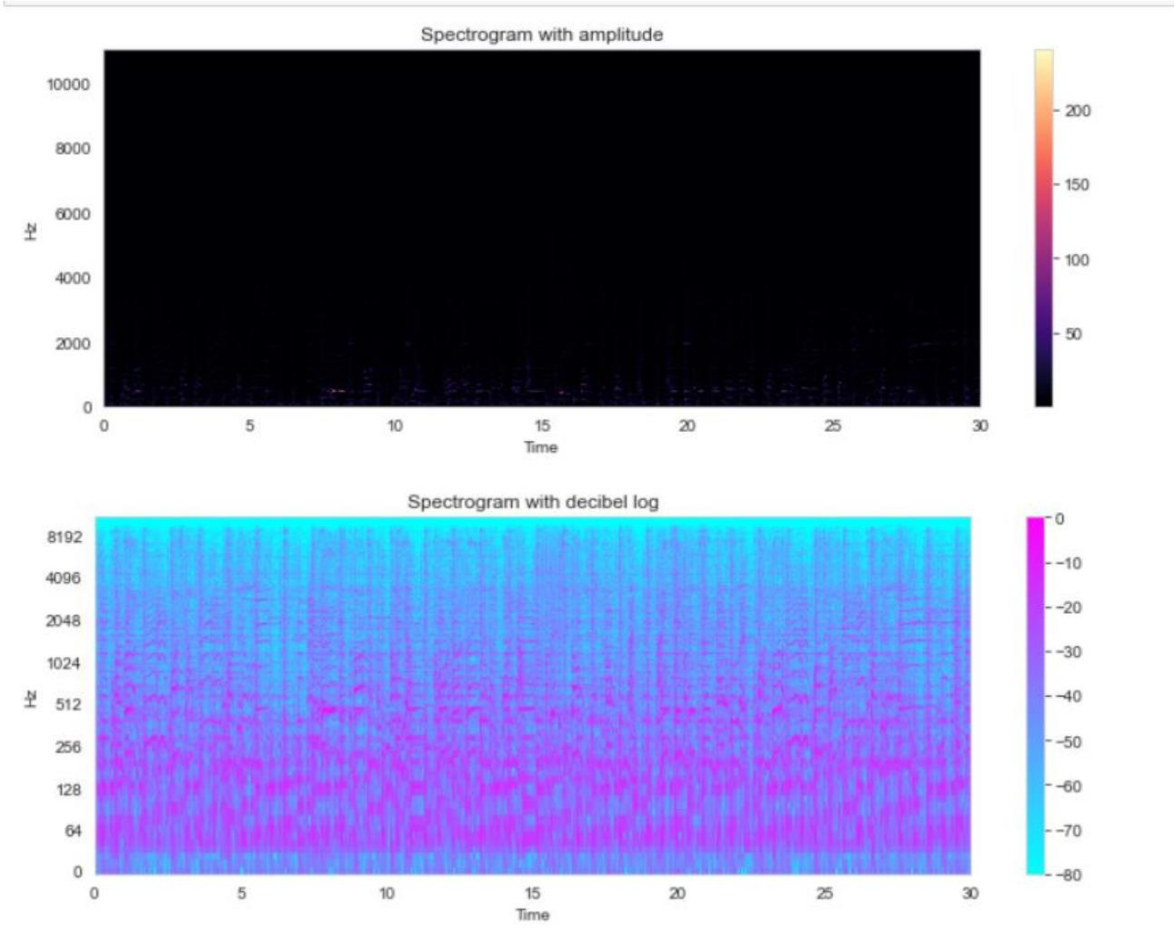
Dataset has (9990 rows, 60 columns) and no null values

Count of samples

Out[5]:

	index	label
0	blues	1000
1	jazz	1000
2	metal	1000
3	pop	1000
4	reggae	1000
5	disco	999
6	classical	998
7	hiphop	998
8	rock	998
9	country	997

Visualizing Sample Audio file:

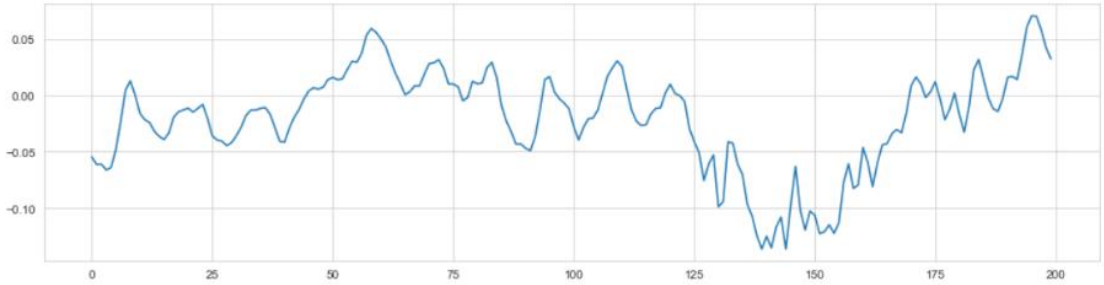


A mel spectrogram is a visual representation of the frequency content of an audio signal. It is obtained by applying a mathematical transformation called the Mel scale to the power spectrum of the audio signal. The Mel scale is a perceptual scale of pitches that is based on the way the human ear perceives sounds.

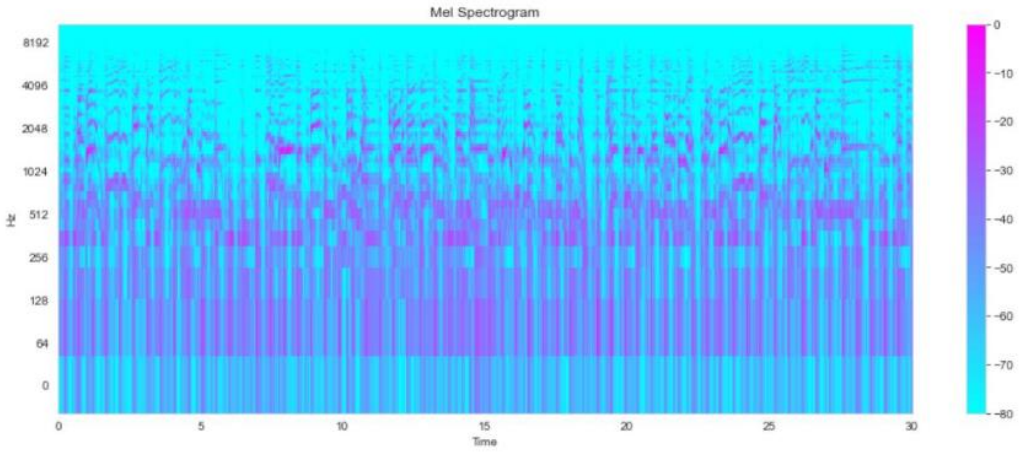
In a mel spectrogram, the x-axis represents time, while the y-axis represents frequency. The intensity of each point in the spectrogram is represented by a color, with brighter colors indicating higher amplitudes. This allows for a visualization of how the frequency content of an audio signal changes over time.

Mel spectrograms are commonly used in speech recognition, music analysis, and other audio processing tasks. They are often used as input to machine learning algorithms for tasks such as speech recognition, where the goal is to classify spoken words or phrases based on their spectrogram representation. Mel spectrograms are also useful for analyzing the frequency content of music, which can be used for tasks such as genre classification, beat detection, and music recommendation.

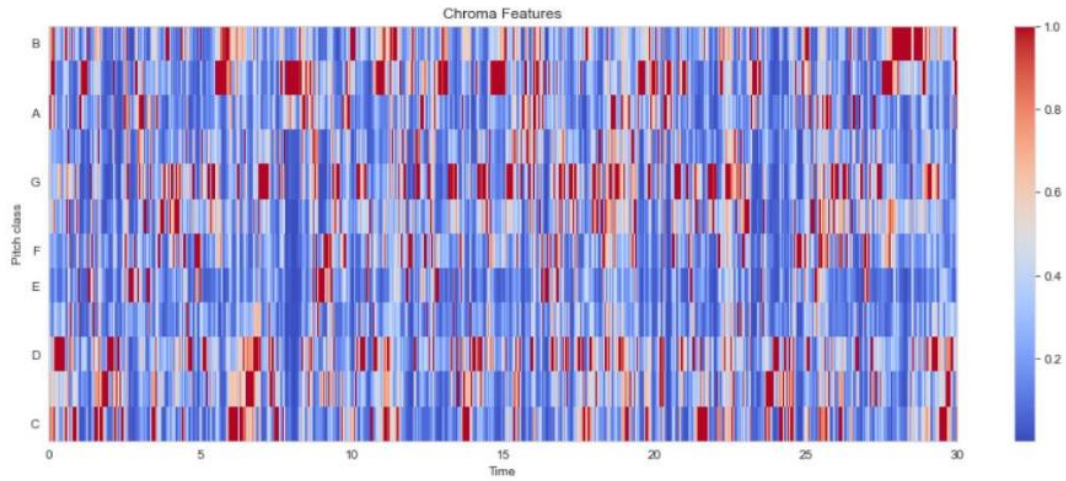
zoomed audio wave



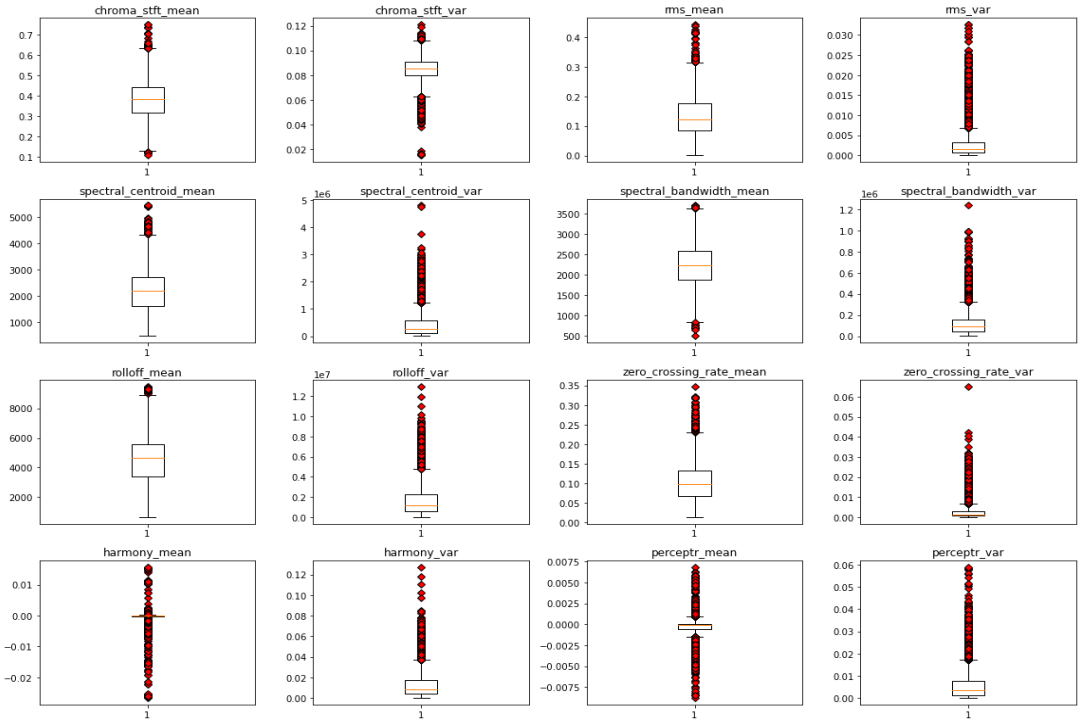
Spectrogram



Chroma features:

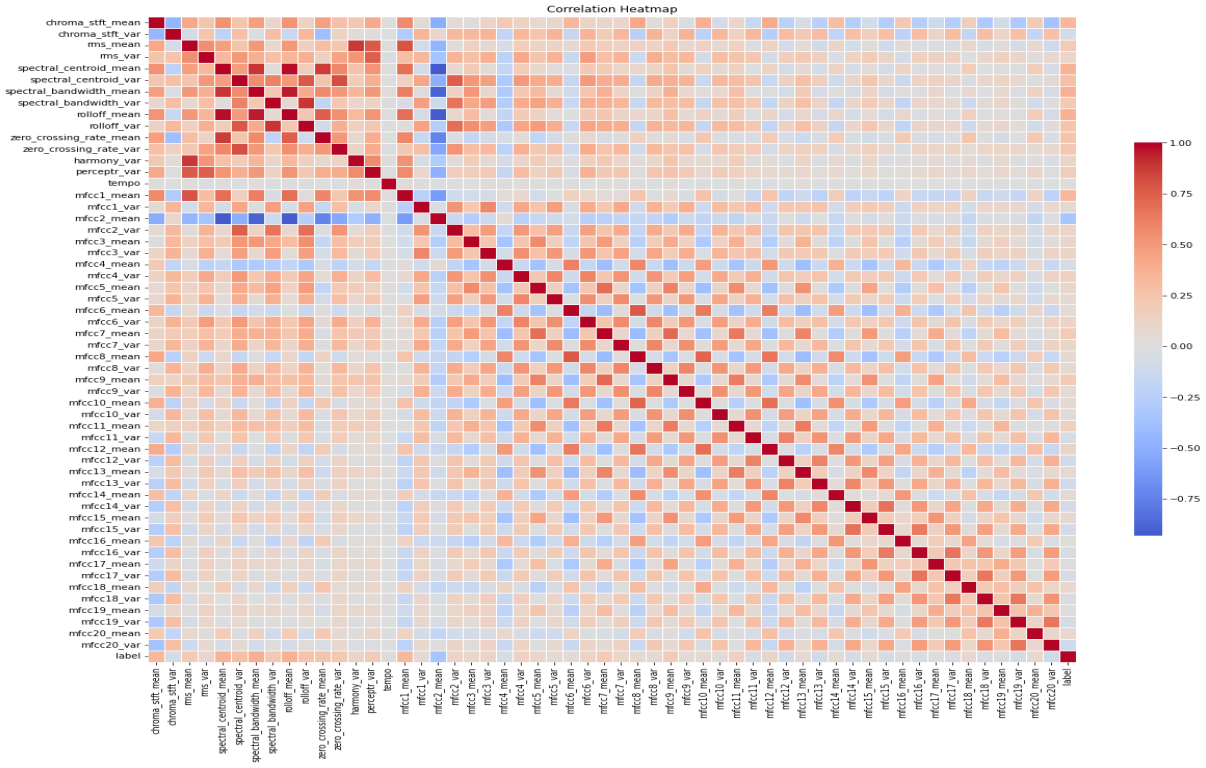


Outliers :



Harmony_mean and perceptr_mean has the most number of outliers.

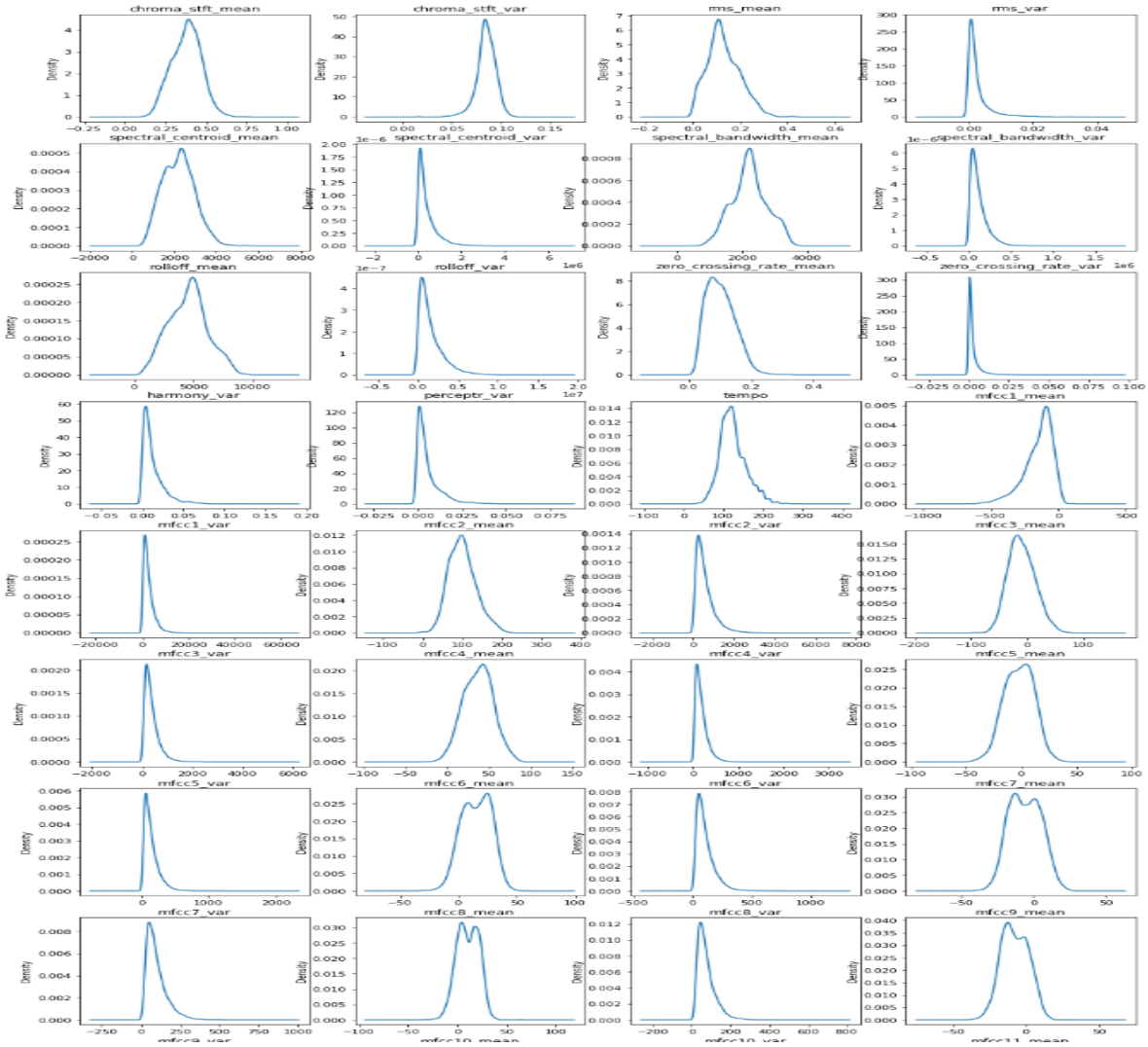
Computing the Correlation Matrix:

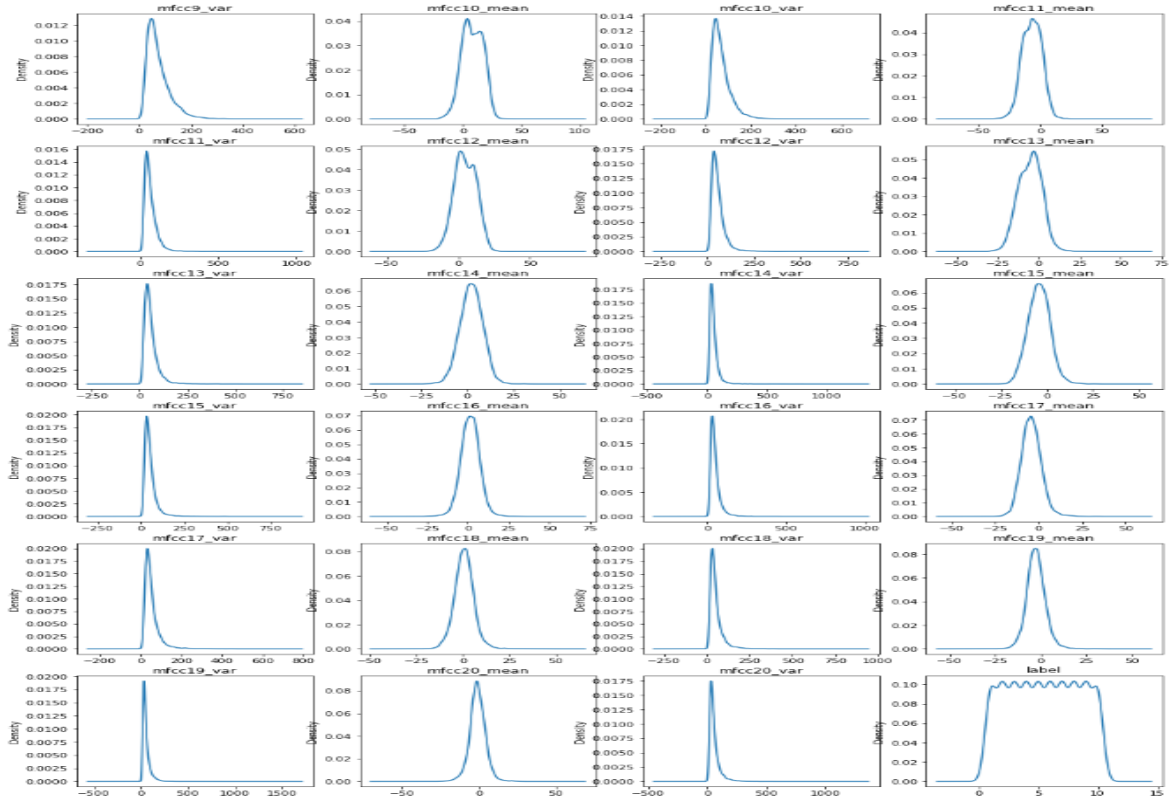


chroma_stft_mean	**chroma_stft_mean**	**1.000000**
spectral_centroid_mean	**rolloff_mean**	**0.974360**
spectral_bandwidth_mean	**rolloff_mean**	**0.951000**
spectral_bandwidth_var	**rolloff_var**	**0.891339**
spectral_centroid_mean	spectral_bandwidth_mean	0.890382
rms_mean	harmony_var	0.884846
spectral_centroid_mean	zero_crossing_rate_mean	0.865487
spectral_centroid_var	zero_crossing_rate_var	0.818348
rms_mean	mfcc1_mean	0.795000
spectral_centroid_var	rolloff_var	0.780308
mfcc6_mean	mfcc8_mean	0.769248
rms_mean	perceptr_var	0.766446
rolloff_mean	zero_crossing_rate_mean	0.755442
spectral_centroid_var	mfcc2_var	0.748612
rms_var	perceptr_var	0.744850

The highlighted features are the highly correlated features in the dataset.

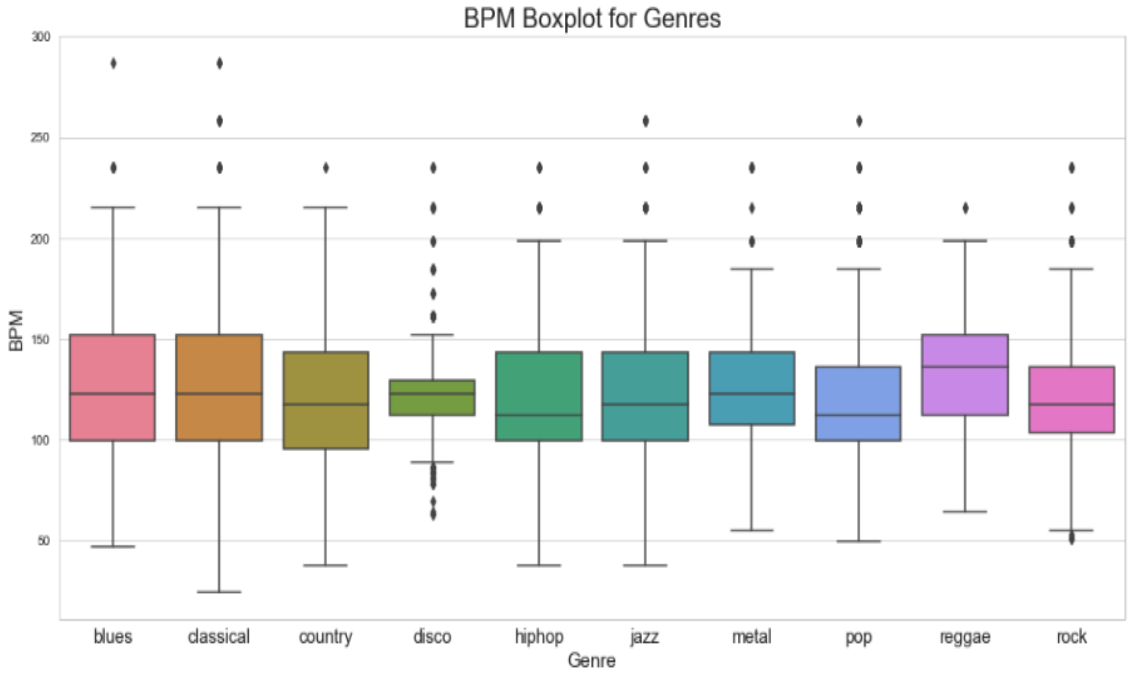
Distribution of variables:



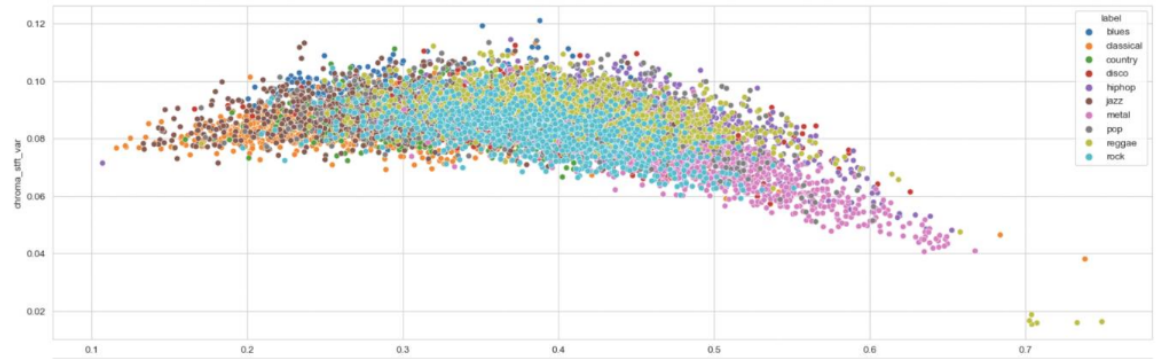


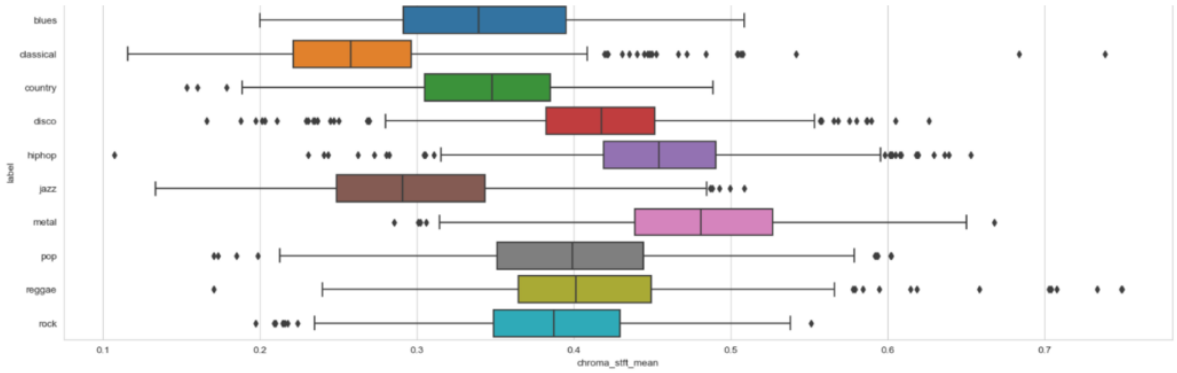
We can infer that all the features are uniformly distributed.

Boxplot for genres:



Distribution of Important Features:





DATA MINING TASKS:

Standardisation across features:

Standardization is important in features because it helps to ensure that the data used for machine learning models are on the same scale and have the same variance. This is critical because many machine learning algorithms are sensitive to the scale of the features, and having features that are on different scales can cause problems.

Out[27]:

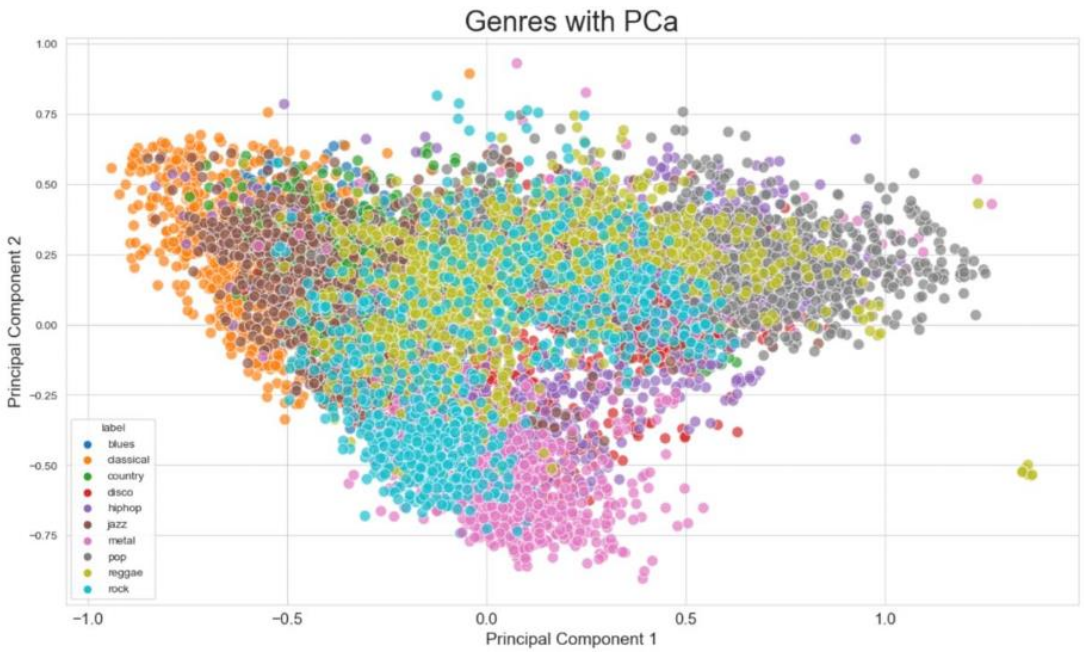
	chroma_stft_mean	chroma_stft_var	rms_mean	rms_var	spectral_centroid_mean	spectral_centroid_var	spectral_bandwidth_mean	spectral_ba
blues_mean	-0.399753	0.515638	0.078559	-0.005222	-0.619183	-0.390359	-0.564741	
classical_mean	-1.260167	-0.159735	-1.280308	-0.642441	-1.117054	-0.794661	-1.319966	
country_mean	-0.371519	0.047376	-0.072112	-0.364476	-0.399813	-0.198027	-0.260759	
disco_mean	0.413770	-0.152532	0.083252	0.031170	0.567600	0.123584	0.504504	
hiphop_mean	0.794292	0.166106	0.638290	1.041111	0.389764	0.606730	0.434806	
jazz_mean	-0.844938	0.286477	-0.612816	-0.416469	-0.473296	-0.481762	-0.349040	
metal_mean	1.065019	-1.231928	0.425991	-0.356458	0.560493	-0.421644	0.109178	
pop_mean	0.189476	0.374635	0.852578	1.017664	1.005701	0.992397	1.224505	
reggae_mean	0.333611	0.390567	-0.009850	0.051055	-0.070037	0.610099	0.070194	
rock_mean	0.105241	-0.311793	-0.129517	-0.419350	0.184378	-0.068944	0.143671	

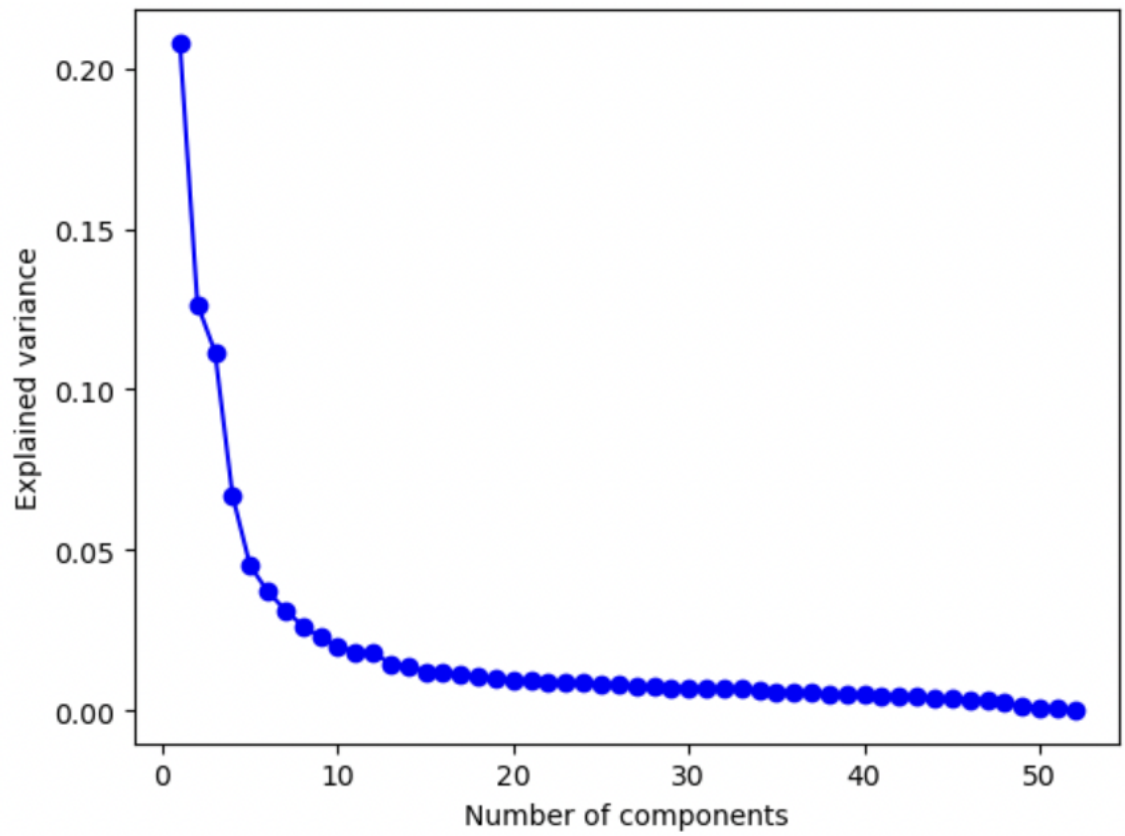
10 rows × 57 columns

Label Encoding for the response variable:

```
In [10]: label_encoding = {'blues': 1, 'classical': 2, 'country': 3, 'disco': 4, 'hiphop': 5, 'jazz': 6, 'metal': 7, 'p
         df['label'] = df['label'].replace(label_encoding)

In [11]: df['label'].unique()

Out[11]: array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10])
```

Principal Component Analysis:





Since the first 13 components contribute the most towards the target variable we reduce the dimension of data to 13 components.

```
df_pca.shape
```

```
8]: (9990, 13)
```

Splitting data into train and test set:

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(df_pca.drop('label', axis=1), df_pca['label'], test_size=0.25, random_state=42)

```
X_train Shape: (7492, 12)
X_test Shape: (2498, 12)
y_train Shape: (7492,)
y_test Shape: (2498,)
```

DATA MINING MODELS/ METHODS

We have implemented different machine learning models such as:

1. K Nearest neighbors
2. Gaussian Naive Bayes
3. Random Forest Classifier
4. Logistic Regression
5. Linear Discriminant Analysis
6. Support Vector Classifier

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.svm import SVC
from sklearn.metrics import classification_report


knn = KNeighborsClassifier()
nb = GaussianNB()
rf = RandomForestClassifier()
lr = LogisticRegression()
lda = LinearDiscriminantAnalysis()
svm = SVC()


knn.fit(X_train, y_train)
nb.fit(X_train, y_train)
rf.fit(X_train, y_train)
lr.fit(X_train, y_train)
lda.fit(X_train, y_train)
svm.fit(X_train, y_train)


knn_pred = knn.predict(X_test)
nb_pred = nb.predict(X_test)
rf_pred = rf.predict(X_test)
lr_pred = lr.predict(X_test)
lda_pred = lda.predict(X_test)
svm_pred = svm.predict(X_test)
```

KNN Metrics:

Accuracy: 0.7566052842273819

Precision: 0.7591552790612084

Recall: 0.7561072420723634

F1 Score: 0.7550222155862784

KNN Classification Report:

	precision	recall	f1-score	support
1	0.74	0.77	0.75	257
2	0.86	0.92	0.89	256

3	0.62	0.74	0.68	232
4	0.63	0.72	0.67	255
5	0.81	0.75	0.78	270
6	0.79	0.74	0.76	244
7	0.88	0.82	0.85	261
8	0.80	0.82	0.81	224
9	0.75	0.75	0.75	254
10	0.72	0.53	0.61	245
accuracy			0.76	2498
macro avg	0.76	0.76	0.76	2498
weighted avg	0.76	0.76	0.76	2498

Naive Bayes Metrics:

Accuracy: 0.5068054443554844

Precision: 0.5118378956995205

Recall: 0.5065222727321381

F1 Score: 0.4959297297818693

Naive Bayes Classification Report:

	precision	recall	f1-score	support
1	0.53	0.25	0.34	257
2	0.78	0.80	0.79	256
3	0.33	0.44	0.38	232
4	0.39	0.41	0.40	255
5	0.61	0.41	0.49	270
6	0.50	0.41	0.45	244
7	0.50	0.87	0.64	261
8	0.64	0.67	0.65	224
9	0.55	0.56	0.55	254
10	0.30	0.25	0.27	245
accuracy			0.51	2498

macro avg	0.51	0.51	0.50	2498
weighted avg	0.51	0.51	0.50	2498

Random Forest Metrics:

Accuracy: 0.7373899119295436

Precision: 0.7342208041605509

Recall: 0.736517978984267

F1 Score: 0.7332081573205491

Random Forest Classification Report:

	precision	recall	f1-score	support
1	0.75	0.67	0.70	257
2	0.84	0.93	0.88	256
3	0.64	0.65	0.65	232
4	0.65	0.68	0.66	255
5	0.83	0.75	0.79	270
6	0.70	0.73	0.71	244
7	0.79	0.88	0.83	261
8	0.75	0.83	0.79	224
9	0.72	0.72	0.72	254
10	0.67	0.52	0.59	245
accuracy			0.74	2498
macro avg	0.73	0.74	0.73	2498
weighted avg	0.74	0.74	0.73	2498

Logistic Regression Metrics:

Accuracy: 0.5536429143314652

Precision: 0.5375146597372649

Recall: 0.5535465618212347

F1 Score: 0.5400645858079806

Logistic Regression Classification Report:

	precision	recall	f1-score	support
1	0.47	0.51	0.49	257
2	0.85	0.88	0.86	256
3	0.37	0.33	0.35	232
4	0.37	0.27	0.31	255
5	0.63	0.47	0.54	270
6	0.58	0.61	0.59	244
7	0.62	0.82	0.71	261
8	0.61	0.81	0.69	224
9	0.54	0.58	0.56	254
10	0.34	0.26	0.29	245
accuracy			0.55	2498
macro avg	0.54	0.55	0.54	2498
weighted avg	0.54	0.55	0.54	2498

LDA Metrics:

Accuracy: 0.5124099279423538

Precision: 0.49770742752733665

Recall: 0.5129413949669543

F1 Score: 0.494099763757428

LDA Classification Report:

	precision	recall	f1-score	support
1	0.45	0.41	0.43	257
2	0.79	0.85	0.82	256
3	0.35	0.34	0.35	232
4	0.30	0.21	0.25	255
5	0.68	0.40	0.50	270
6	0.53	0.52	0.52	244

7	0.55	0.84	0.67	261
8	0.53	0.79	0.64	224
9	0.51	0.57	0.54	254
10	0.28	0.19	0.23	245
accuracy			0.51	2498
macro avg	0.50	0.51	0.49	2498
weighted avg	0.50	0.51	0.50	2498

SVM Metrics:

Accuracy: 0.6937550040032026

Precision: 0.6897122911535681

Recall: 0.6935901866195163

F1 Score: 0.6903376024342267

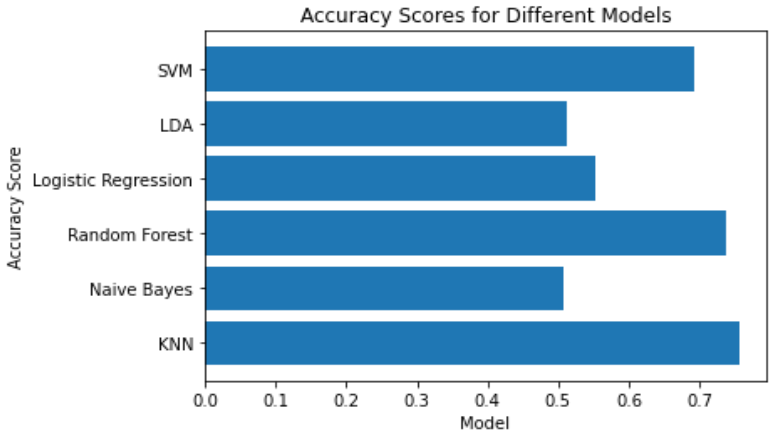
SVM Classification Report:

	precision	recall	f1-score	support
1	0.68	0.63	0.65	257
2	0.85	0.91	0.88	256
3	0.59	0.62	0.61	232
4	0.53	0.53	0.53	255
5	0.79	0.68	0.73	270
6	0.70	0.76	0.73	244
7	0.77	0.86	0.81	261
8	0.76	0.81	0.79	224
9	0.70	0.68	0.69	254
10	0.52	0.46	0.49	245
accuracy			0.69	2498
macro avg	0.69	0.69	0.69	2498
weighted avg	0.69	0.69	0.69	2498

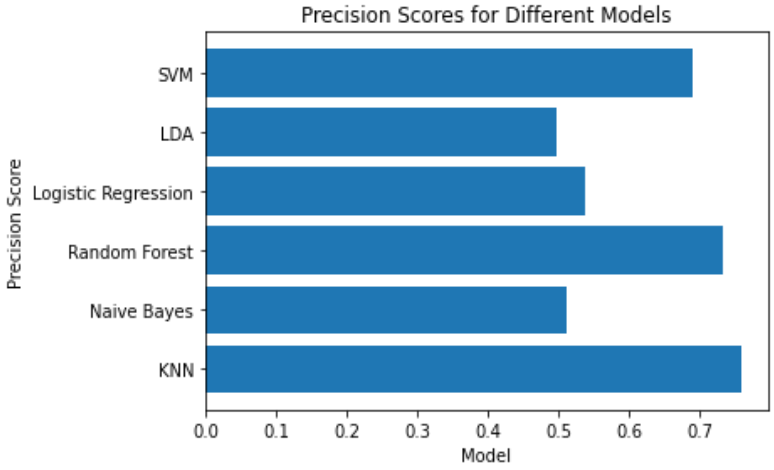
PERFORMANCE EVALUATION

We evaluated the implemented algorithms based on accuracy, precision F1 score and recall score

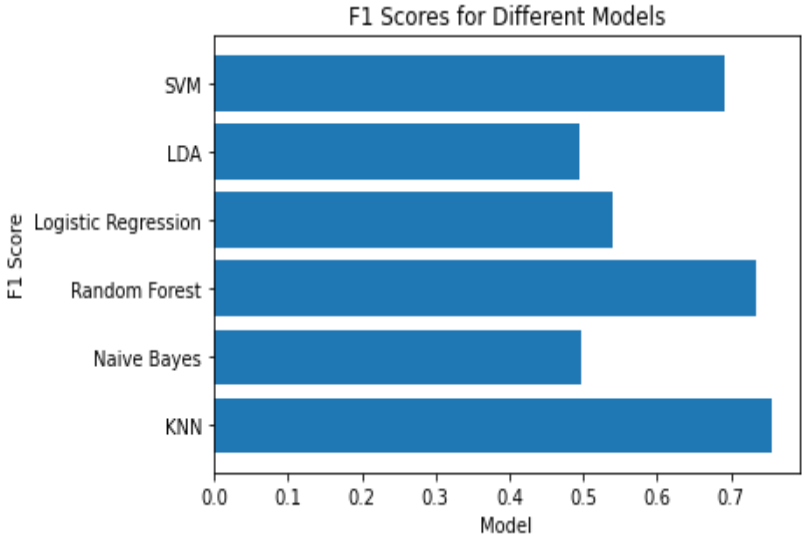
Accuracy



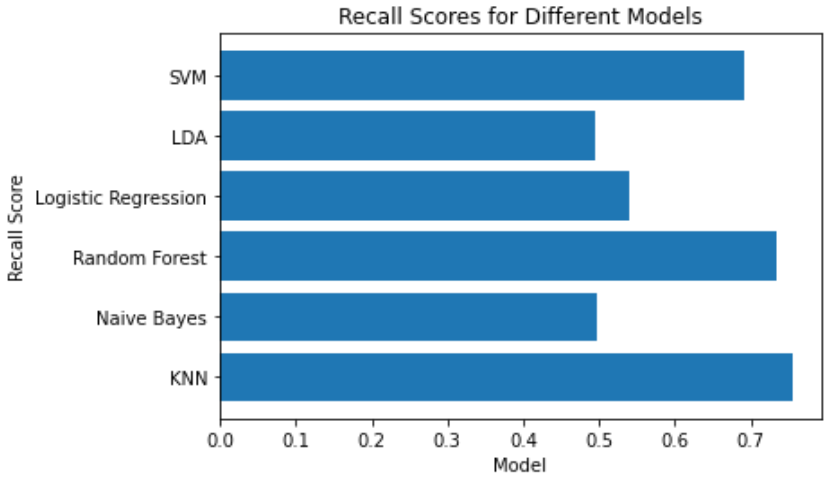
Precision



F1 - Score

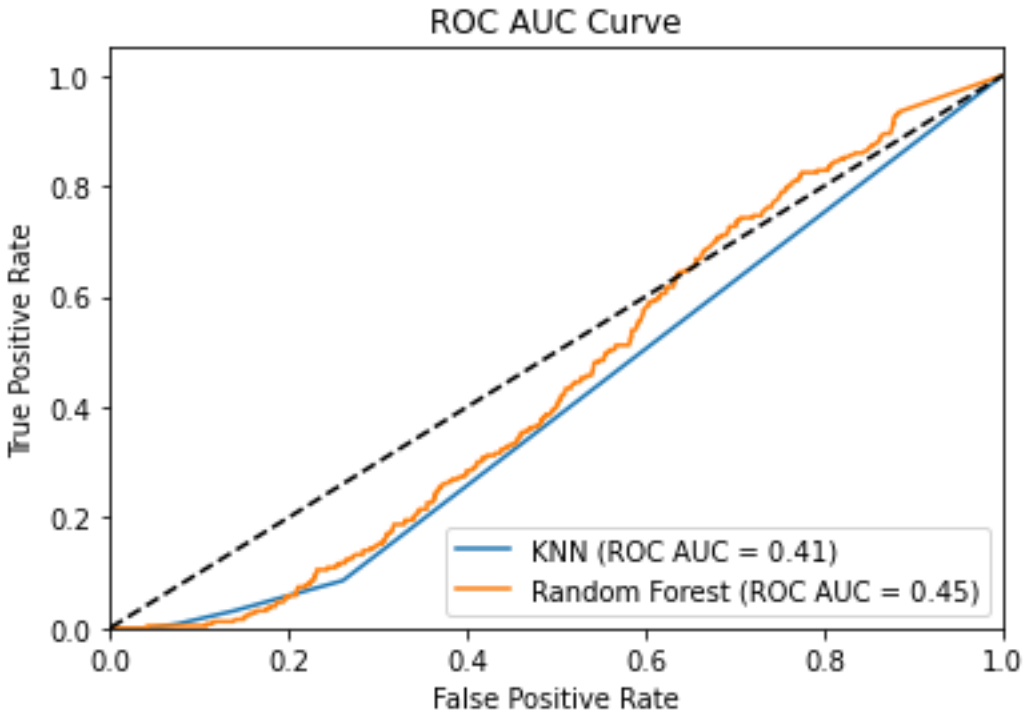


Recall



ROC AUC curve

```
from sklearn.multiclass import OneVsRestClassifier
from sklearn.metrics import roc_curve, auc
from sklearn.preprocessing import label_binarize

knn = KNeighborsClassifier()
rf = RandomForestClassifier()
ovr_knn = OneVsRestClassifier(knn)
ovr_rf = OneVsRestClassifier(rf)
ovr_knn.fit(X_train, y_train)
ovr_rf.fit(X_train, y_train)
y_prob_knn = ovr_knn.predict_proba(X_test)
y_prob_rf = ovr_rf.predict_proba(X_test)

y_test_bin = label_binarize(y_test, classes=[0, 1, 2])


fpr_knn = dict()
tpr_knn = dict()
roc_auc_knn = dict()
for i in range(3):
    fpr_knn[i], tpr_knn[i], _ = roc_curve(y_test_bin[:, i], y_prob_knn[:, i])
    roc_auc_knn[i] = auc(fpr_knn[i], tpr_knn[i])

fpr_rf = dict()
tpr_rf = dict()
roc_auc_rf = dict()
for i in range(3):
    fpr_rf[i], tpr_rf[i], _ = roc_curve(y_test_bin[:, i], y_prob_rf[:, i])
    roc_auc_rf[i] = auc(fpr_rf[i], tpr_rf[i])


import matplotlib.pyplot as plt
plt.figure()
plt.plot(fpr_knn[2], tpr_knn[2], label='KNN (ROC AUC = %0.2f)' % roc_auc_knn[2])
plt.plot(fpr_rf[2], tpr_rf[2], label='Random Forest (ROC AUC = %0.2f)' % roc_auc_rf[2])
plt.plot([0, 1], [0, 1], 'k--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC AUC Curve')
plt.legend(loc="lower right")
plt.show()
```



PROJECT RESULTS:

- Based on the provided metrics, both the KNN and Random Forest models seem to have performed similarly with the KNN model having slightly better accuracy, precision, recall, and F1 score values than the Random Forest model. However, the difference in performance is relatively small and may not be statistically significant.
- When choosing between the two models, other factors such as interpretability, scalability, and computational requirements should also be considered. KNN is a relatively simple and interpretable model that can be effective for small datasets, while Random Forest is a more complex and computationally intensive model that can handle larger datasets and high-dimensional feature spaces.
- Therefore, the choice between KNN and Random Forest should depend on the specific requirements of the problem at hand and the characteristics of the dataset being used.

IMPACT OF THE PROJECT OUTCOMES:

Machine learning has numerous applications in music genre classification. Music streaming platforms, such as Spotify and Pandora, use machine learning algorithms to recommend songs and playlists to users based on their listening history and preferences, which requires music genre classification to identify similar songs.

Music information retrieval focuses on developing algorithms to automatically extract information from music, including genre, tempo, and key, using machine learning to classify music based on audio features. In music production, machine learning can automate tasks such as genre classification, helping producers identify the right sounds and samples to use. Additionally, machine learning can be used in music analysis to study different aspects of music, such as the relationship between different genres or the evolution of musical styles over time. Overall, machine learning plays a critical role in music genre classification across various applications.