COMPARISON OF SUPERVISED MACHINE LEARNING ALGORTIHMS

In recent decades, the development of artificial intelligence has directly influenced many different industries. Undoubtedly, the music industry is one of these. The development of artificial intelligence and its connection with music has become a significant factor in encouraging creativity since the introduction of this technology. Artificial intelligence is also revolutionizing the field of music. Renowned composer Hans Zimmer collaborated on the musical score for the documentary series "Blue Planet II." In his documentary, he highlighted the potential of successful relationships between human creativity and the computational ability of algorithms. (Zimmer, 2018). It is remarkable that this famous composer made this statement. Creativity is an important component of human nature. Consequently, numerous innovations developed, inventors introduced new ideas, and the impact of knowledge gradually increased. Creativity is a crucial component of science. Nowadays, artificial intelligence has been programmed to be this creative. Multiple algorithms are currently being developed for this purpose, and the most suitable ones are selected. Exploring the potential impact of this development on science is fascinating, because it has the potential to be revolutionary because it allows individuals to answer previously unknown questions. Artificial intelligence started creating unique musical compositions. In short, this technology has started to acquire unique skills that are unique to humans. An instance of this phenomenon can be observed in the MuseNet company. Music recommendation systems use data analysis of consumer tastes and behaviours to offer customized playlists as suggestions. These systems employ methodologies like collaborative filtering and natural language processing to understand user preferences and offer customized music recommendations. The selected project started with the observation that artificial intelligence has now extended its capabilities into the field of music. The objective of this predictive research is to build a music recommendation system through the analysis of given song datasets. Therefore, it provides musicians with the opportunity to create revolutionary music, while also enabling listeners to discover unique melodies. Indeed, it provides passionate listeners with the opportunity to get to know themselves with the

melodies from many different perspectives. In this way, musicians have the ability to determine certain features that should be present in their songs.

PROBLEM DEFINITION

The developing music industry in recent years has led to the replacement of physical records by digital songs. Although this naturally limited the listenability of the song, it gave people the opportunity to listen to many songs. Innovative companies, like Spotify and YouTube, have the capability of quickly providing individuals with access to every song ever created. Furthermore, it provides individuals with the opportunity to select these songs. However, among an enormous number of millions of songs, how do we identify our personal likes and dislikes. Artificial intelligence algorithms are utilized to address this issue by developing recommendation systems. This enabled data analysts to examine huge data sets. It enabled him to describe the attributes of the songs (such as danceability, tempo, energy, etc.). Thus, it became possible to generate anticipatable song recommendations based on the music that every fan listens to. The objective of this project is to develop a music recommendation system by evaluating the accuracy of different algorithms using feature data collected from two Kaggle datasets containing hundreds of thousands of records. The objective is to categorize songs as either 'Hip-Hop' or 'Rock' by analyzing this dataset. During the development of these recommendation systems, the objective was to implement four machine learning algorithms. The accuracy results of these algorithms will be shown, and the algorithm that produces the highest level of accuracy will be chosen. The objective is to broaden and diversify the selection of songs that individuals listen to, and to introduce them to songs that they will find more enjoyable. This presents an ideal opportunity for musicians as it will enable them to gain a deeper understanding of the songs and exploring more genres will improve their creativity. By performing additional research, it may be possible to examine the music listening habits of various populations.

DATA COLLECTION AND PREPARATION

Initially, a dataset containing 9 different features of hundreds of thousands of songs, which was generated by Echo Nest, was obtained. Important features of these songs: These aspects can be observed, such as its acoustic attributes, danceability, and tempo. The second dataset contains the musical characteristics of each recorded feature, ranging from -1 to 1. The file types of these two files are distinct. One file is imported in CSV format, while the other file is imported in JSON format. Due to mismatch data types, the Python code removed all data from the columns that contained these features. This was done to avoid any problems with the dataset. The datasets in these two formats were later combined and merged into a unified file. The table presented below displays the data it contains along with their respective data types. The complexity grows directly with the increase in data volume. However, it is reasonable to anticipate that this will lead us into better results.

Variables of the Dataset

Data	Data Type
Track_id	Integer
Acousticness	Float
Danceability	Float
Energy	Float
Instrumentalness	Float
Liveness	Float
Speechiness	Float
Tempo	Float
Valence	Float
Genre_top	Object

EXPLORATORY DATA ANALYSIS

Exploratory data analysis is a collection of approaches mostly developed by John Wilder Tukey since 1970. The idea behind this strategy is to analyze the data prior to using a certain probability model. Tukey, J.W. states that exploratory data analysis shows similarities to detective work. Understanding and interpreting of the merge dataset of metrics and rock-hiphop datasets rely heavily on exploratory data analysis. Through EDA, project discovers pattern that may be present in the data and acquire useful insights. EDA in this assignment entails looking at number of variables including Track id, acoustiness, danceability, energy, instrumentalness, liveness, speechiness, tempo, valence and genre top.

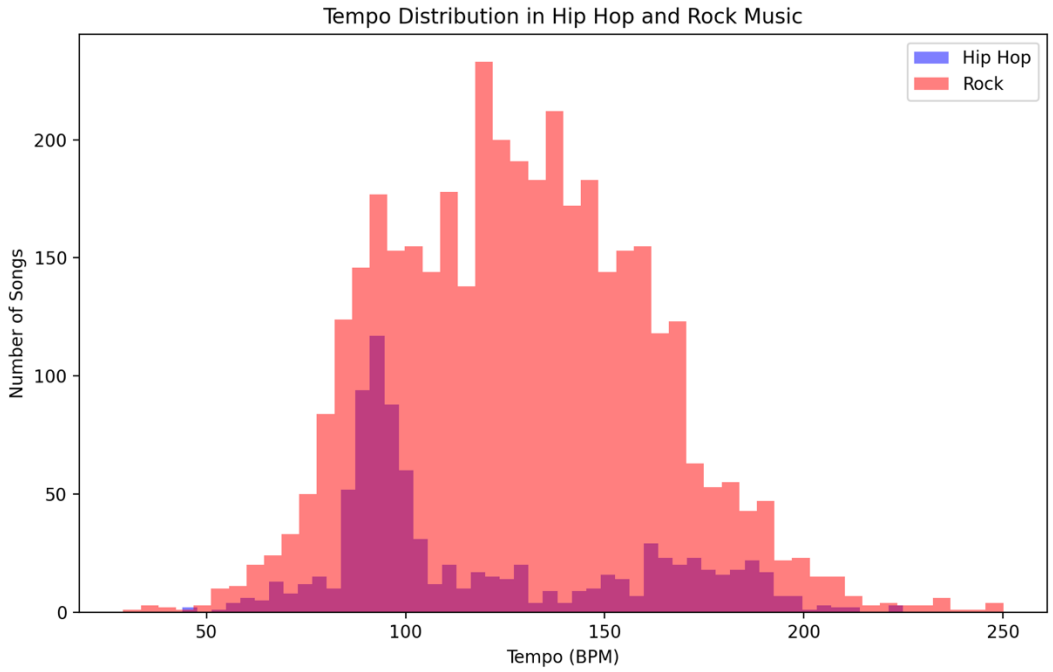
1) Initially, the datasets that were collected were assigned names and saved into the computer. Then, the two entities were combined and the resultant output was obtained using the information code. The dataset contains 8 data points of float type, 1 data point of integer type, and 1 data point of object type.

```
import pandas as pd
Songs = pd.read_csv('/Users/mustafaakgul/Downloads/fma-rock-vs-hiphop.csv')
echonest_metrics = pd.read_json('/Users/mustafaakgul/Downloads/echonest-metrics.json', precise_float=True)
echo_Songs = echonest_metrics.merge(Songs[['genre_top', 'track_id']], on='track_id')
echo_Songs.info()
```

```
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   track_id          4802 non-null   int64
 1   acousticness      4802 non-null   float64
 2   danceability      4802 non-null   float64
 3   energy            4802 non-null   float64
 4   instrumentalness  4802 non-null   float64
 5   liveness          4802 non-null   float64
 6   speechiness       4802 non-null   float64
 7   tempo             4802 non-null   float64
 8   valence           4802 non-null   float64
 9   genre_top         4802 non-null   object
dtypes: float64(8), int64(1), object(1)
memory usage: 375.3+ KB
(base) mustafaakgul@Mustafa-MacBook-Pro ~ %
```

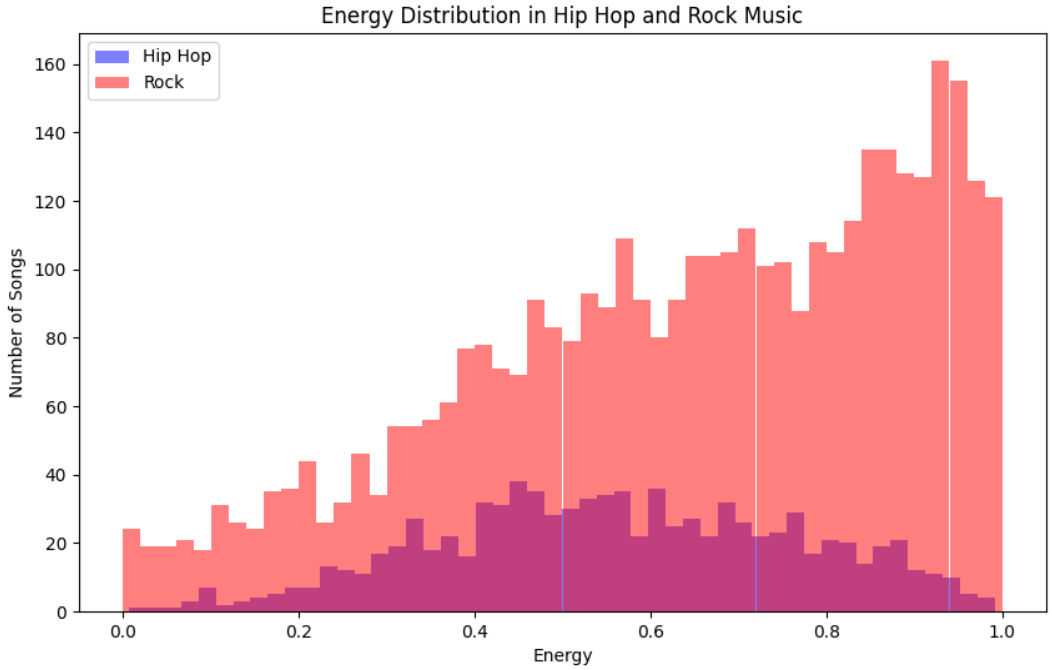
2) This graphic analysis analyzes the degree of tempo distribution present in rock and hip hop music genres. BPM means beats per minute. The research shows that the beats per minute (bpm) of rock songs mainly range within the range of 100 to 150 bpm. The hip hop tracks had the highest frequency of songs within the range of 90-100 bpm. The rock songs in this dataset have a higher tempo compared to the hip-hop tracks.

```python
import pandas as pd
import matplotlib.pyplot as plt
songs = pd.read_csv('/Users/mustafaakgul/Downloads/fma-rock-vs-hiphop.csv')
echonest_metrics = pd.read_json('/Users/mustafaakgul/Downloads/echonest-metrics.json', precise_float=True)
df = echonest_metrics.merge(songs[['genre_top', 'track_id']], on='track_id')
hip_hop = df[df['genre_top'] == 'Hip-Hop']
rock = df[df['genre_top'] == 'Rock']
plt.figure(figsize=(10, 6))
plt.hist(hip_hop['tempo'], bins=50, color='blue', alpha=0.5, label='Hip Hop')
plt.hist(rock['tempo'], bins=50, color='red', alpha=0.5, label='Rock')
plt.title('Tempo Distribution in Hip Hop and Rock Music')
plt.xlabel('Tempo (BPM)')
plt.ylabel('Number of Songs')
plt.legend()
plt.show()
```



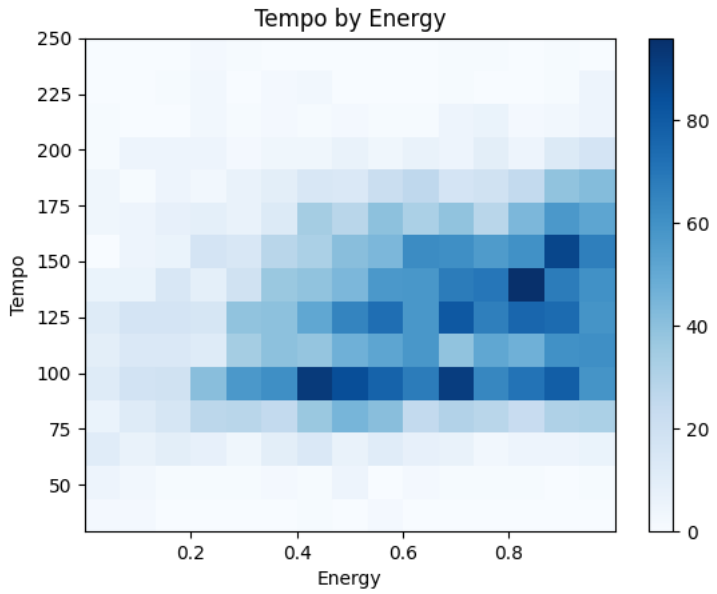
3) Here we may analyze the impact of energy distribution on hip hop and rock tunes. The intensity level of hip hop songs was predominantly moderate. Rock songs have greater energy levels compared to hip-hop music, as per observations.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
names= ['track_id','acousticness','danceability', 'energy', 'instrumentalness', 'liveness', 'speechiness', 'tempo', 'valence', 'genre_top']
Songs = pd.read_csv('/Users/mustafaakgul/Downloads/fma-rock-vs-hiphop.csv')
echonest_metrics = pd.read_json('/Users/mustafaakgul/Downloads/echonest-metrics.json', precise_float=True)
df = echonest_metrics.merge(Songs[['genre_top', 'track_id']], on= 'track_id')
hip_hop = df[df['genre_top'] == 'Hip-Hop']
rock = df[df['genre_top'] == 'Rock']
plt.figure(figsize=(10, 6))
plt.hist(hip_hop['energy'], bins=50, color='blue', alpha=0.5, label='Hip Hop')
plt.hist(rock['energy'], bins=50, color='red', alpha=0.5, label='Rock')
plt.title('Energy Distribution in Hip Hop and Rock Music')
plt.xlabel('Energy')
plt.ylabel('Number of Songs')
plt.legend()
plt.show()
```



4) The objective of this code was to achieve a correlation between tempo and energy. The task required the utilization of plt.color() to create a drawing. Focus was observed in the area characterized by a tempo ranging from 90 to 160 bpm and an energy level between 0.4 and 1. By combining two characteristics, understanding of the music in the dataset may be improved. The relationships between qualities may be observed with greater clarity.

```python
import numpy as np
import matplotlib.pyplot as plt
names= ['track_id','acousticness','danceability', 'energy', 'instrumentalness', 'liveness', 'speechiness', 'tempo', 'valence', 'genre_top']
Songs = pd.read_csv('/Users/mustafaakgul/Downloads/fma-rock-vs-hiphop.csv')
echonest_metrics = pd.read_json('/Users/mustafaakgul/Downloads/echonest-metrics.json', precise_float=True)
df = echonest_metrics.merge(Songs[['genre_top', 'track_id']], on= 'track_id')
hip_hop = df[df['genre_top'] == 'Hip-Hop']
rock = df[df['genre_top'] == 'Rock']
plt.hist2d(df['energy'], df['tempo'], bins=15, cmap= 'Blues')
plt.xlabel('Energy')
plt.ylabel('Tempo')
plt.title('Tempo by Energy')
plt.colorbar()
plt.show()
```



5) "LabelEncoder" code was used to digitize the data exactly. Each categorical data is assigned a numerical value and the result is available in the table below.

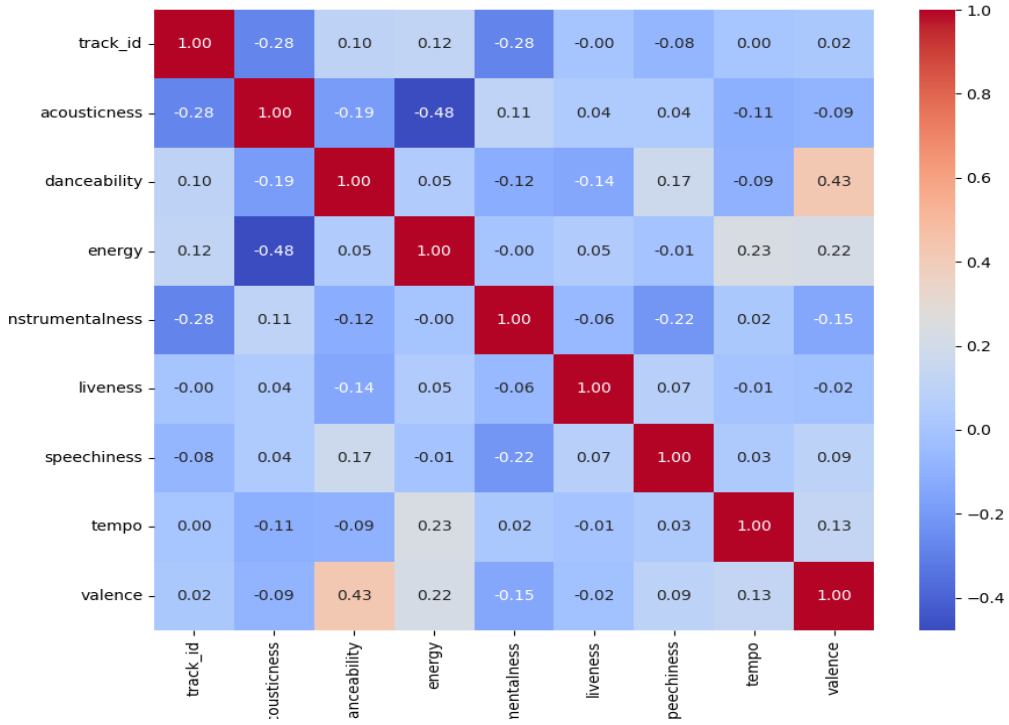
```python
import pandas as pd
from sklearn.preprocessing import LabelEncoder

Songs = pd.read_csv('/Users/mustafaakgul/Downloads/fma-rock-vs-hiphop.csv')
echonest_metrics = pd.read_json('/Users/mustafaakgul/Downloads/echonest-metrics.json', precise_float=True)
df = echonest_metrics.merge(Songs[['genre_top', 'track_id']], on= 'track_id')

categorical_features = ['track_id', 'acousticness','danceability', 'energy', 'instrumentalness', 'liveness', 'speechiness', 'tempo', 'valence', 'genre_top']
label_encoder = LabelEncoder()
for feature in categorical_features:
    df[feature] = label_encoder.fit_transform(df[feature])

print(df)
```

	track_id	acousticness	danceability	energy	instrumentalness	liveness	speechiness	tempo	valence	genre_top
0	0	2158	4228	2309	741	3316	4079	4035	3133	0
1	1	2054	3338	3476	574	1289	4589	2449	1456	0
2	2	990	4508	2752	486	4333	3935	1308	3375	0
3	3	2243	3214	1850	820	869	4658	1855	4535	0
4	4	4653	866	4676	4722	2331	2549	675	53	1
...	...	...	...	...	...	...	...	...	...	...
4797	4797	2151	4285	3691	6	4776	4471	1078	3731	0
4798	4798	1052	3944	2925	223	2648	4278	1087	2154	0
4799	4799	675	4122	2455	138	4718	4263	982	2332	0
4800	4800	1125	2518	3153	164	1213	4395	4203	3146	0
4801	4801	1435	4055	3141	5	3848	4383	212	3542	0

[4802 rows x 10 columns]

6) By plotting a correlation matrix, it is aimed to understand how all value pairs in the table are related to each other. Of great importance in large data sets, it is a useful tool for finding and showing patterns in data. Afterwards, visualization was helped with the heatmap function. For example, it can be said that there is a slight link between energy and acousticness and a strong link between danceability and valence.

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
echonest_metrics = pd.read_json('/Users/mustafaakgul/Downloads/echonest-metrics.json', precise_float=True)
corr_matrix = echonest_metrics.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, fmt=".2f", cmap='coolwarm')
plt.show()
```


track_id
acousticness
danceability
energy
nstrumentalness
liveness
speechiness
tempo
valence
1.00 -0.28 0.10 0.12 -0.28 -0.00 -0.08 0.00 0.02
-0.28 1.00 -0.19 -0.48 0.11 0.04 0.04 -0.11 -0.09
0.10 -0.19 1.00 0.05 -0.12 -0.14 0.17 -0.09 0.43
0.12 -0.48 0.05 1.00 -0.00 0.05 -0.01 0.23 0.22
-0.28 0.11 -0.12 -0.00 1.00 -0.06 -0.22 0.02 -0.15
-0.00 0.04 -0.14 0.05 -0.06 1.00 0.07 -0.01 -0.02
-0.08 0.04 0.17 -0.01 -0.22 0.07 1.00 0.03 0.09
0.00 -0.11 -0.09 0.23 0.02 -0.01 0.03 1.00 0.13
0.02 -0.09 0.43 0.22 -0.15 -0.02 0.09 0.13 1.00
track_id
:ousticness
anceability
energy
nentalness
liveness
peechiness
tempo
valence
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

MODEL SELECTION AND IMPLEMENTATION

Machine learning is the combination of computer science and artificial intelligence. The objective of this is to gain significant insights from the data. Software programs utilize a mathematical language to create algorithms through statistical methods. There are three types of learning that can be utilized: supervised learning, unsupervised learning, and reinforcement learning. Their applications vary. The outcome differs based on whether the input data is classified as a label or not. In the present time period, deep learning, an advanced artificial intelligence model that imitates the brain's functioning, is extensively employed. These algorithms facilitate the extraction of crucial information from intricate data sets and enable their analysis for the purpose of constructing models. The algorithms mentioned can be classified into four categories: supervised, unsupervised, semi-supervised, and reinforced. Supervised learning entails establishing a correlation between input data and output data, enabling the capability to make accurate predictions. Supervised Learning is a machine learning technique that focuses on understanding the relationship between input and output data by utilizing a designated set of paired training examples. An input-output training sample is frequently referred to as the result produced by a system that functions as a label for input data or control. This category of training data is alternatively referred to as labeled training data or supervised data. It is occasionally referred to as Teacher Learning (Haykin 1998), Learning from Labeled Data, or Inductive Machine Learning (Kotsiantis 2007). Unsupervised learning entails the implementation of clustering algorithms. The usage data does not possess any categorization. Semi-supervised learning entails the utilization of both unlabelled and labelled data. Unlabelled data is utilized widely, whereas labelled data is utilized rarely. Reinforcement machine learning employs past experiences to deliver the most favorable result in order to achieve the intended objective.

1) LOGISTIC REGRESSION

Logistic regression is a highly important analytical method applied in both social and natural sciences. The method is an useful supervised machine learning technique for analyzing the similarities between two characteristics. Logistic regression is a type of classifier that utilizes supervised machine learning and works based on probabilistic principles. It needs both input and aimed output. In this case, characteristics are shown in the picture below. Initially, logistic regression, a classification tool, was employed to determine the correlation between Rock and Hiphop music. The necessary data for training the classification algorithm may be obtained by examining the provided features and labels. The algorithm executed the following steps: data preparation, train-test split, feature scaling, Logistic regression modeling, algorithm training, prediction and evaluation. The outputs of each step were recorded accordingly.

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score, f1_score, precision_score

names= ['track_id','acousticness','danceability', 'energy', 'instrumentalness', 'liveness', 'speechiness', 'tempo',
        'valence', 'genre_top']
Songs = pd.read_csv('/Users/mustafaakgul/Downloads/fma-rock-vs-hiphop.csv')
echonest_metrics = pd.read_json('/Users/mustafaakgul/Downloads/echonest-metrics.json', precise_float=True)
df = echonest_metrics.merge(Songs[['genre_top', 'track_id']], on= 'track_id')
hip_hop = df[df['genre_top'] == 'Hip-Hop']
rock = df[df['genre_top'] == 'Rock']

#Preparing Data
features = df.drop(columns=['genre_top', 'track_id'], axis=1)
labels = df['genre_top']

#Train Test Split
features_train, features_test, labels_train, labels_test = train_test_split(features, labels, test_size=0.2, random_state=20)

# Feature Scaling
scaler = StandardScaler()
features_train = scaler.fit_transform(features_train)
features_test = scaler.transform(features_test)

#Modelling LogisticRegression
LogReg = LogisticRegression(random_state=20)

#Training and Predicting
LogReg.fit(features_train, labels_train)
labels_pred = LogReg.predict(features_test)
```

```
#Evaluating the Algorithm
ConfusionMatrix = confusion_matrix(labels_test, labels_pred)
ClassificationReport = classification_report(labels_test, labels_pred)
accuracy = accuracy_score(labels_test, labels_pred)
precision = precision_score(labels_test,labels_pred,pos_label='Rock')
f1= f1_score(labels_test, labels_pred,pos_label='Rock')

print("Confusion Matrix:" ,ConfusionMatrix)
print("Accuracy:", accuracy)
print("Precision:", precision)
print("F1 Score:",f1)
print(ClassificationReport)
```

2) DECISION TREE

The decision tree is a supervised algorithm which can implement for both classification and regression tasks. Because of there are inputs variable, this algorithm can be imlemented. First, datasets are created. Then, attributes and labels are created. Furthermore, train test split codes are wroten. For train and predict this algorithm import decisiontreeclassifier code from sklearn.tree code. Conclusion, by showing confusion matrix, results can be shown and be prepared to analyze the algorithm.

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score, f1_score, precision_score

#Preparing Datasets
songs = pd.read_csv('/Users/mustafaakgul/Downloads/fma-rock-vs-hiphop.csv')
echonest_metrics = pd.read_json('/Users/mustafaakgul/Downloads/echonest-metrics.json', precise_float=True)
df = echonest_metrics.merge(songs[['genre_top', 'track_id']], on='track_id')
hip_hop = df[df['genre_top'] == 'Hip-Hop']
rock = df[df['genre_top'] == 'Rock']

#Preparing Data
attribute = df.drop(columns=['genre_top', 'track_id'], axis=1)
labels = df['genre_top']

# Train Test Split
attribute_train, attribute_test, labels_train, labels_test = train_test_split(attribute, labels, test_size=0.3, random_state=21)

# Training and Predictions
from sklearn.tree import DecisionTreeClassifier
Classifier=DecisionTreeClassifier()
Classifier.fit(attribute_train,labels_train)
labels_pred = Classifier.predict(attribute_test)

#Evaluating the Algorithm
ConfusionMatrix = confusion_matrix(labels_test, labels_pred)
ClassificationReport = classification_report(labels_test, labels_pred)
accuracy = accuracy_score(labels_test, labels_pred)
f1 = f1_score(labels_test,labels_pred,pos_label='Rock')
precision = precision_score(labels_test,labels_pred,pos_label='Rock')
print("Confusion Matrix:",ConfusionMatrix)
print(ClassificationReport)
print("Accuracy:",accuracy)
print("F1 Score:", f1_score)
print("Precision:", precision)
```

3) K MEANS CLUSTERING

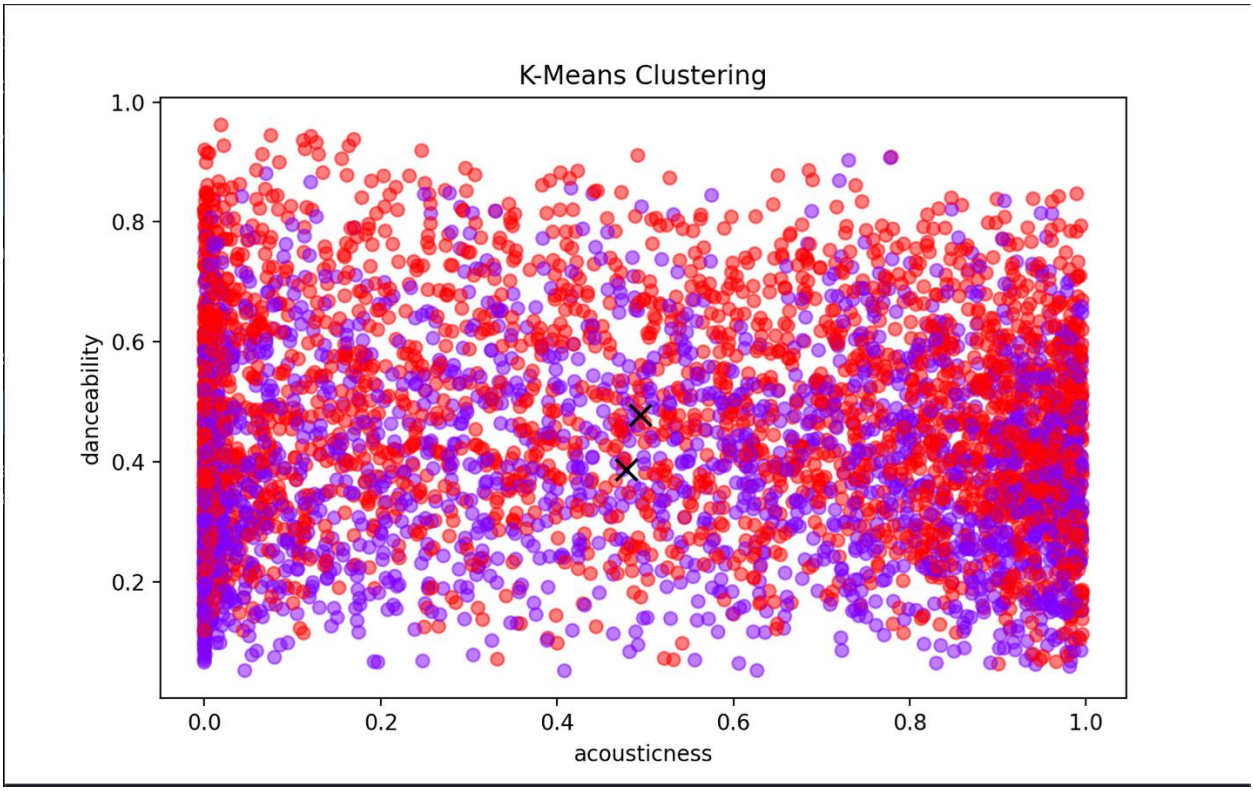
K means algorithm is used to solve the clustering problems. It determines the best value for K center points or centroids by an iterative process. It assigned each data point to its closest k center.

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.cluster import KMeans
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score, f1_score, precision_score

songs = pd.read_csv('/Users/mustafaakgul/Downloads/fma-rock-vs-hiphop.csv')
echonest_metrics = pd.read_json('/Users/mustafaakgul/Downloads/echonest-metrics.json', precise_float=True)
df = echonest_metrics.merge(songs[['genre_top', 'track_id']], on='track_id')
hip_hop = df[df['genre_top'] == 'Hip-Hop']
rock = df[df['genre_top'] == 'Rock']

# Preparing data for the K Means model
X = df.drop(columns=['genre_top', 'track_id'], axis=1)
Y = df['genre_top']

# Splitting the dataset into training and test sets
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=20)

# Creating Clusters
kmeans = KMeans(n_clusters=2)
kmeans.fit(X)
print(kmeans.cluster_centers_)
```

```
[[4.78400093e-01 3.87099826e-01 6.71209827e-01 6.15633755e-01
  1.87262125e-01 1.00230442e-01 1.57512412e+02 4.63475782e-01]
 [4.94332993e-01 4.78090983e-01 5.86423037e-01 5.94406329e-01
  1.88614841e-01 1.08779789e-01 1.00800114e+02 4.44962517e-01]]
```

Here, a table has been drawn based on the characteristics of the songs, danceability and acoustics. Black dots indicate the center of the clusters.

```python
plt.figure(figsize=(8, 5))
plt.scatter(X.iloc[:, 0], X.iloc[:, 1], c=kmeans.labels_, cmap='rainbow', alpha=0.5)
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], color='black', marker='x', s=100)
plt.title('K-Means Clustering')
plt.xlabel(X.columns[0])
plt.ylabel(X.columns[1])
plt.show()
```




```
35    #Predicting and Labelling
36    Y_pred = kmeans.predict(X_test)
37    Match = {0: 'Hip-Hop', 1: 'Rock'}
38    Y_train = np.vectorize(Match.get)(Y_pred)
39    kmeans_accuracy = accuracy_score(Y_test, Y_train)
40    print("K Means Accuracy:", kmeans_accuracy)
```

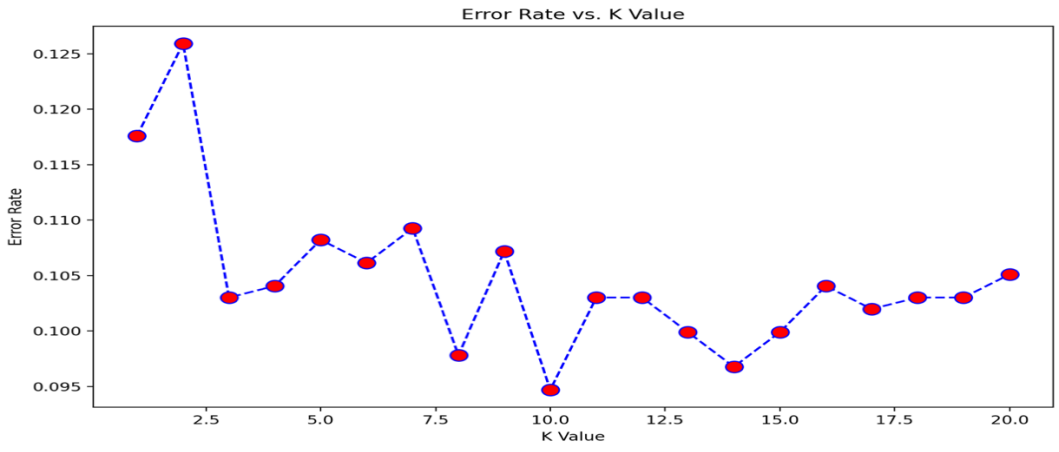
4) K NEAREST NEIGHBOURS

K nearest is a type of supervised algorithm used in classification and regression processes. In this algorithm, the first step to be carried out is to determine the number of neighbors close to an object, k, in order to determine which class it will belong to. The distance to these neighbors is calculated by a specified method. According to this number k, the object tested is considered a member of the class to which it is closest.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix

#Preparing Datasets
songs = pd.read_csv('/Users/mustafaakgul/Downloads/fma-rock-vs-hiphop.csv')
echonest_metrics = pd.read_json('/Users/mustafaakgul/Downloads/echonest-metrics.json', precise_float=True)
df = echonest_metrics.merge(songs[['genre_top', 'track_id']], on='track_id')
hip_hop = df[df['genre_top'] == 'Hip-Hop']
rock = df[df['genre_top'] == 'Rock']

X = df.drop(columns=['genre_top', 'track_id'], axis=1)
Y = df['genre_top']

# Train Test Split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=20)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
scaler.fit(X_train)
X_train=scaler.transform(X_train)
X_test=scaler.transform(X_test)

#Training and Predicting
from sklearn.neighbors import KNeighborsClassifier
classifier=KNeighborsClassifier(n_neighbors=5)
classifier.fit(X_train,Y_train)
Y_pred = classifier.predict(X_test)
```

```python
#Evaluating the Algorithm
from sklearn.metrics import classification_report,confusion_matrix
print(confusion_matrix(Y_test,Y_pred))
print(classification_report(Y_test, Y_pred))


#Comparing Error Rate with K
error_rates = []
for k in range(1, 21):
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, Y_train)
    pred_k = knn.predict(X_test)
    error = np.mean(pred_k != Y_test)
    error_rates.append(error)

# Plotting the error rates
plt.figure(figsize=(10, 6))
plt.plot(range(1, 21), error_rates, color='blue', linestyle='dashed', marker='o',
         markerfacecolor='red', markersize=10)
plt.title('Error Rate vs. K Value')
plt.xlabel('K Value')
plt.ylabel('Error Rate')
plt.show()
```

Error Rate vs. K Value



MODEL EVALUATION

The accuracy rates obtained for each model were compared and the algorithm with the best accuracy rate was determined. Data set separation rates and accuracy rates are given in the tables. It has been observed that accuracy rates decrease when the ratio of the training set to the test set is reduced. In the study, Tree Decision with KNN achieved over 89% accuracy in both test groups. It has been observed that the worst performing algorithm is k means. It has been observed that both of these algorithms predict rock songs more highly than hip-hop songs. This may be because rock songs are more prominent.

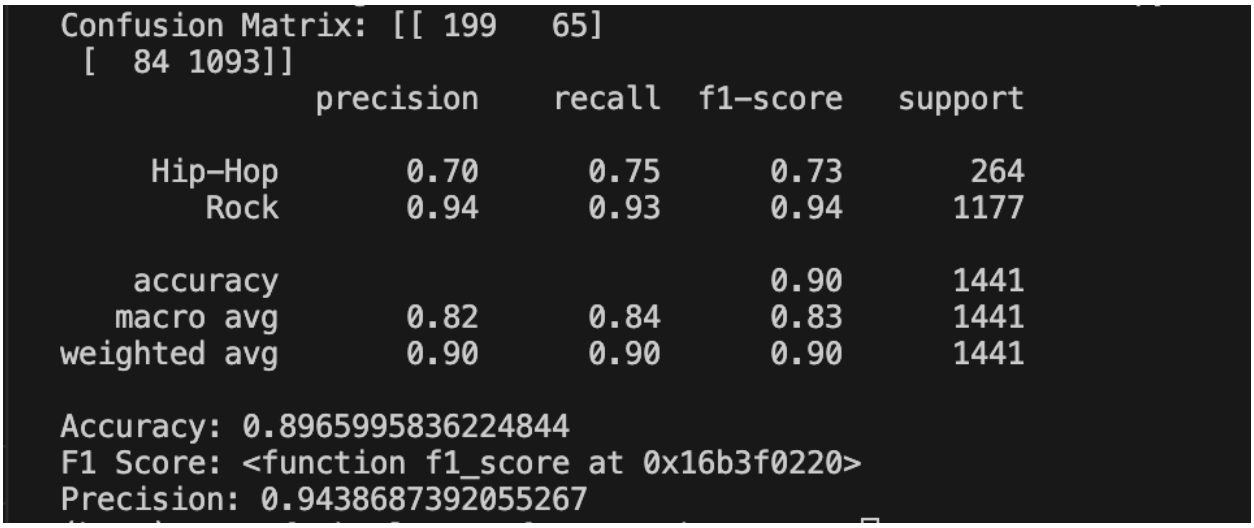
Logistic Regression:

```
Confusion Matrix: [[101  84]
 [ 30 746]]
Accuracy: 0.8813735691987513
Precision: 0.8987951807228916
F1 Score: 0.9290161892901618
              precision    recall  f1-score   support

     Hip-Hop       0.77      0.55      0.64       185
        Rock       0.90      0.96      0.93       776

    accuracy                           0.88       961
   macro avg       0.83      0.75      0.78       961
weighted avg       0.87      0.88      0.87       961
```

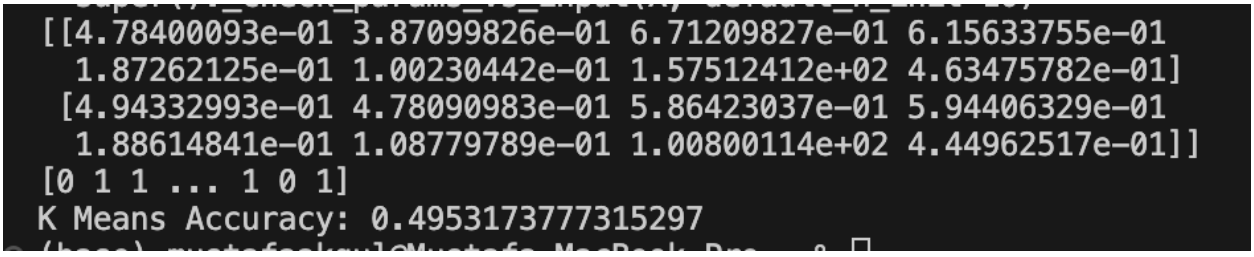
A classification report was generated, showing precision, recall, f1 score for each class. In all scores of Rock class are higher than Hiphop class. Also the Logistic Regression algortim's accuracy is 0.88 which is good.

Decision Tree:

A decision tree utilizes a hierarchical structure to construct classification models. The data is divided into decision nodes and leaf nodes, with each node representing a progressively decreasing percentage . A decision node includes many branches that represent the values associated with the attribute being evaluated.



K Means:



K Nearest Neighbors:

KNN algorithm, when implemented in music genre classification, looks at similar songs and assumes that they belong to the same category because they seem to be near to each other. Among various other techniques that prevail in this concept, the best results have been procured out of this technique. The KNN algorithm is a basic machine learning approach that analyzes data by comparing the similarity of characteristics. When new data is inputted, the computer automatically identifies and classifies it based on its similarity to existing data.

```
[[117  68]
 [ 36 740]]
              precision    recall  f1-score   support

     Hip-Hop       0.76      0.63      0.69       185
        Rock       0.92      0.95      0.93       776

    accuracy                           0.89       961
   macro avg       0.84      0.79      0.81       961
weighted avg       0.89      0.89      0.89       961
```

CONCLUSION AND RECOMMENDATIONS

Artificial intelligence has become common in various fields, aiming to improve the efficiency of people's daily lives. One option is to develop a music recommendation system. As technology improved, people began consuming more music, and the application of artificial intelligence tools in this context has become crucial for people's music preferences. Attaining a success rate of 90 percent shows a high likelihood that individuals are able to predict their future music preferences based on their current musical preferences. The classification of music genres is a crucial component of the recommendation system. The project took place in multiple phases, which included identifying datasets relevant to the problem, transforming the datasets into a format suitable for analysis, implementing the required algorithms, and obtaining both visual and statistical results. These results were then analyzed to determine the most suitable algorithm. The data was divided into an 80 percent training set and a 20 percent testing set, and only the models that produced accurate results were examined. The KNN and decision tree algorithms demonstrated an approximate accuracy rate of 90 percent. The logistic regression algorithm produced results that were highly similar to the other two algorithms. Four different algorithms have been used to develop a predictive model for music genre classification. Once the data set had been divided into training and test sets, the classifier was trained using the training data. Once the f1 score, accuracy, precision, and recall have been calculated. The collected data showed a significantly high overall accuracy rate of approximately 90%. The K-Nearest Neighbors (KNN) and Decision Tree algorithms exhibited excellent performance, achieving a high accuracy score.

PROJECT REFLECTION

First of all, I think it is a good result that the accuracy rate is close to ninety percent. However, if additional analysis is required, I think neural networks or support vector may be useful. Attachment units of these units. I would guess that higher accuracy results can be achieved. Where I have difficulty working on a project is where k means I have difficulty comparing the value of k to the error rate when running it.

REFERENCES

- Haykin, S. (1998). *Neural networks: A comprehensive foundation* (2nd ed.).

- Kotsiantis, S. (2007). Supervised machine learning: A review of classification techniques. *Informatica Journal*

- *Zimmer, Hans (2018)*