In [29]:
```
import pandas as pd
```

In [30]:
```
file_path = 'ML2.csv'
```

In [31]:
```
df = pd.read_csv(file_path)
```

In [32]:
```
print(df.head())
```

```
   Number                                              title  \
0       1                        Put Your Head On My Shoulder
1       2  Whatever Will Be Will Be (Que Sera Sera) (with...
2       3                             Everybody Loves Somebody
3       4          Take Good Care Of My Baby - 1990 Remastered
4       5                                  A Teenager In Love

                 artist        top genre  year  bpm  nrgy  dnce  dB  live  val  \
0            Paul Anka  adult standards  2000  116    34    55  -9    10   47
1            Doris Day  adult standards  1948  177    34    42 -11    72   78
2          Dean Martin  adult standards  2013   81    49    26  -9    34   40
3            Bobby Vee  adult standards  2011   82    43    49 -12    12   66
4  Dion & The Belmonts  adult standards  1959   79    38    56  -9    13   62

   dur  acous  spch  pop
0  155     75     3   72
1  123     86     4   62
2  162     81     4   61
3  151     70     6   60
4  158     67     3   60
```

In [33]:
```
from sklearn.preprocessing import LabelEncoder, StandardScaler
```

Loading the data set

In [49]:
```
data = {
    'Number': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19
               21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 3
               38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 5
               55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 7
    'title': ['Put Your Head On My Shoulder', 'Whatever Will Be Will Be (Que Ser
    'artist': ['Paul Anka', 'Doris Day', 'Dean Martin', 'Bobby Vee', 'Dion & The
    'top genre ': ['adult standards', 'adult standards', 'adult standards', 'adu
    'year': [2000, 1948, 2013, 2011, 1959, 1962, 2000, 1991, 1993, 1961, 1990, 1
}
```

Spilting the data to find the Top Genre

In [42]:
```
from sklearn.model_selection import train_test_split
```

In [43]:
```
import pandas as pd
```

In [44]:
```
file_path = 'ML2.csv'
```

In [45]:
```
df = pd.read_csv(file_path, encoding='latin1')
```

In [50]:
```
print(df.head())
```

```
   Number                                              title  \
0       1                        Put Your Head On My Shoulder
1       2  Whatever Will Be Will Be (Que Sera Sera) (with...
2       3                             Everybody Loves Somebody
3       4          Take Good Care Of My Baby - 1990 Remastered
4       5                                   A Teenager In Love

                 artist        top genre  year  bpm  nrgy  dnce  dB  live  val  \
0             Paul Anka  adult standards  2000  116    34    55  -9    10   47
1             Doris Day  adult standards  1948  177    34    42 -11    72   78
2           Dean Martin  adult standards  2013   81    49    26  -9    34   40
3             Bobby Vee  adult standards  2011   82    43    49 -12    12   66
4  Dion & The Belmonts   adult standards  1959   79    38    56  -9    13   62

   dur  acous  spch  pop
0  155     75     3   72
1  123     86     4   62
2  162     81     4   61
3  151     70     6   60
4  158     67     3   60
```

In [53]:
```
df.columns = df.columns.str.strip()
```

In [54]:
```
target_variable = ' top genre'
```

In [56]:
```
X = df.drop(columns=['top genre'])  # Features
y = df['top genre']
```

In [57]:
```
train_data, test_data = train_test_split(file_path, test_size=0.2, random_state=
```

In [58]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
```

In [59]:
```
print("Training set shape:", X_train.shape, y_train.shape)
print("Testing set shape:", X_test.shape, y_test.shape)
```

```
Training set shape: (58, 14) (58,)
Testing set shape: (15, 14) (15,)
```

Decision trees classification model to predict the top genre

In [145…
```
from sklearn.tree import DecisionTreeClassifier
```

In [146…
```
from sklearn.metrics import accuracy_score
```

In [147…
```
from sklearn.model_selection import train_test_split
```

In [148…
```
from sklearn.impute import SimpleImputer
```

In [149…
```
df = pd.read_csv("ML2.csv")
```

In [150…
```
print(df.isnull().sum())
```

```
Number        0
title         0
artist        0
top genre    12
year          0
bpm           0
nrgy          0
dnce          0
dB            0
live          0
val           0
dur           0
acous         0
spch          0
pop           0
dtype: int64
```

In [151…
```
df.dropna(axis=0, inplace=True)
```

In [152…
```
X = df.drop(columns=["top genre", "title", "artist"])
```

In [153…
```
y = df["top genre"]
```

In [154…
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
```

In [155…
```
clf = DecisionTreeClassifier()
```

In [156…
```
import numpy as np
```

In [158…
```
imputer = SimpleImputer(strategy="mean")
X_train_imputed = imputer.fit_transform(X_train)
X_test_imputed = imputer.transform(X_test)
```

In [160…
```
clf.fit(X_train_imputed, y_train)
```

Out[160]:
```
▾ DecisionTreeClassifier

DecisionTreeClassifier()
```

In [161…
```
predictions = clf.predict(X_test_imputed)
```

In [162…
```
accuracy = accuracy_score(y_test, predictions)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.3076923076923077
```

In [ ]:
```
EVALUATING Decission Trees MODEL
```

In [283…
```
precision = precision_score(y_test, y_pred, average='micro', zero_division=1)
recall = recall_score(y_test, y_pred, average='macro', zero_division=1)
f1 = f1_score(y_test, y_pred, average='weighted', zero_division=1)
```

In [284…
```
print("Precision:", precision)
print("Recall:", recall)
print("F1-score:", f1)
```

```
Precision: 0.8461538461538461
Recall: 0.6666666666666666
F1-score: 0.8076923076923077
```

In [285…
```
print("\nClassification Report:")
print(classification_report(y_test, y_pred, zero_division=0))
```

```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         5
           5       0.00      0.00      0.00         1
           6       1.00      1.00      1.00         3
           7       1.00      1.00      1.00         2
           9       0.33      1.00      0.50         1
          10       0.00      0.00      0.00         1

    accuracy                           0.85        13
   macro avg       0.56      0.67      0.58        13
weighted avg       0.79      0.85      0.81        13
```

Random Forest classification model to predict the Top Genre

In [163…
```
from sklearn.ensemble import RandomForestClassifier
```

In [164…
```
data = {
    'Number': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19
               21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 3
               38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 5
               55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 7
    'title': ['Put Your Head On My Shoulder', 'Whatever Will Be Will Be (Que Ser
    'artist': ['Paul Anka', 'Doris Day', 'Dean Martin', 'Bobby Vee', 'Dion & The
    'top genre ': ['adult standards', 'adult standards', 'adult standards', 'adu
    'year': [2000, 1948, 2013, 2011, 1959, 1962, 2000, 1991, 1993, 1961, 1990, 1
}
```

In [168…
```
df = df.drop(columns=['Number', 'title', 'artist'])
```

In [169…
```
df['top genre'] = pd.factorize(df['top genre'])[0]
```

In [170…
```
X = df.drop('top genre', axis=1)
y = df['top genre']
```

In [171…
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.8, random_
```

In [172…
```
clf = RandomForestClassifier()
```

In [173…
```
clf.fit(X_train, y_train)
```

Out[173]:
```
▾ RandomForestClassifier

RandomForestClassifier()
```

In [174…
```
y_pred = clf.predict(X_test)
```

In [175…
```
print("Accuracy:", accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.631578947368421
```

In [ ]:
```
EVALUATING RANDOM FOREST MODEL
```

In [276…
```
precision = precision_score(y_test, y_pred, average='micro', zero_division=1)
recall = recall_score(y_test, y_pred, average='macro', zero_division=1)
f1 = f1_score(y_test, y_pred, average='weighted', zero_division=1)
```

In [277…
```
print("Precision:", precision)
print("Recall:", recall)
print("F1-score:", f1)
```

```
Precision: 0.8461538461538461
Recall: 0.6666666666666666
F1-score: 0.8076923076923077
```

In [278…
```
print("\nClassification Report:")
print(classification_report(y_test, y_pred, zero_division=0))
```

```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         5
           5       0.00      0.00      0.00         1
           6       1.00      1.00      1.00         3
           7       1.00      1.00      1.00         2
           9       0.33      1.00      0.50         1
          10       0.00      0.00      0.00         1

    accuracy                           0.85        13
   macro avg       0.56      0.67      0.58        13
weighted avg       0.79      0.85      0.81        13
```

SVM model to predict the Top Genre

In [176…
```
from sklearn.svm import SVC
```

In [177…
```
from sklearn.preprocessing import LabelEncoder
```

In [186…
```
df = pd.read_csv("ML2.csv")
```

In [196…
```
df.dropna(axis=0, inplace=True)
```

In [210…
```
X = df.drop(columns=['Number', 'title', 'artist', 'year', 'bpm', 'nrgy', 'dnce',
```

In [223…
```
label_encoder = LabelEncoder()
y = label_encoder.fit_transform(df['top genre'])
```

In [227…
```
X = pd.get_dummies(df.drop(columns=['Number', 'title', 'artist', 'year', 'bpm',
```

In [228…
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
```

In [229…
```
svm_classifier = SVC(kernel='linear')
```

In [230…
```
svm_classifier.fit(X_train, y_train)
```

Out[230]:
```
▾ SVC
SVC(kernel='linear')
```

In [231…
```
y_pred = svm_classifier.predict(X_test)
```

In [232…
```
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.8461538461538461
```

EVALUATING SVM MODEL

In [242…
```
from sklearn.metrics import precision_score, recall_score, f1_score
```

In [243…
```
y_pred = svm_classifier.predict(X_test)
```

In [281…
```
precision = precision_score(y_test, y_pred, average='micro', zero_division=1)
recall = recall_score(y_test, y_pred, average='macro', zero_division=1)
f1 = f1_score(y_test, y_pred, average='weighted', zero_division=1)
```

In [282…
```
print("Precision:", precision)
print("Recall:", recall)
print("F1-score:", f1)
```

```
Precision: 0.8461538461538461
Recall: 0.6666666666666666
F1-score: 0.8076923076923077
```

In [286…
```
print("\nClassification Report:")
print(classification_report(y_test, y_pred, zero_division=0))
```

```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         5
           5       0.00      0.00      0.00         1
           6       1.00      1.00      1.00         3
           7       1.00      1.00      1.00         2
           9       0.33      1.00      0.50         1
          10       0.00      0.00      0.00         1

    accuracy                           0.85        13
   macro avg       0.56      0.67      0.58        13
weighted avg       0.79      0.85      0.81        13
```