UTKU KAYA -2019556039

CEN481 – INTRODUCTION TO DATA MINING

DECISION TREE

```
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
```

```
data = pd.read_csv("/content/drive/MyDrive/Acoustic_Features.csv")
data.head()
data.info()
```

#	Column	Non-Null Count	Dtype
0	Class	400 non-null	object
1	_RMSenergy_Mean	400 non-null	float64
2	_Lowenergy_Mean	400 non-null	float64
3	_Fluctuation_Mean	400 non-null	float64
4	_Tempo_Mean	400 non-null	float64
5	_MFCC_Mean_1	400 non-null	float64
6	_MFCC_Mean_2	400 non-null	float64
7	_MFCC_Mean_3	400 non-null	float64
8	_MFCC_Mean_4	400 non-null	float64
9	_MFCC_Mean_5	400 non-null	float64
10	_MFCC_Mean_6	400 non-null	float64
11	_MFCC_Mean_7	400 non-null	float64
12	_MFCC_Mean_8	400 non-null	float64
13	_MFCC_Mean_9	400 non-null	float64
14	_MFCC_Mean_10	400 non-null	float64
15	_MFCC_Mean_11	400 non-null	float64
16	_MFCC_Mean_12	400 non-null	float64
17	_MFCC_Mean_13	400 non-null	float64
18	_Roughness_Mean	400 non-null	float64
19	_Roughness_Slope	400 non-null	float64
20	_Zero-crossingrate_Mean	400 non-null	float64
21	_AttackTime_Mean	400 non-null	float64
22	_AttackTime_Slope	400 non-null	float64
23	_Rolloff_Mean	400 non-null	float64
24	_Eventdensity_Mean	400 non-null	float64
25	_Pulseclarity_Mean	400 non-null	float64
26	_Brightness_Mean	400 non-null	float64
27	_Spectralcentroid_Mean	400 non-null	float64
28	_Spectralspread_Mean	400 non-null	float64
29	_Spectralskewness_Mean	400 non-null	float64
30	_Spectralkurtosis_Mean	400 non-null	float64
31	_Spectralflatness_Mean	400 non-null	float64
32	_EntropyofSpectrum_Mean	400 non-null	float64
33	_Chromagram_Mean_1	400 non-null	float64
34	_Chromagram_Mean_2	400 non-null	float64
35	_Chromagram_Mean_3	400 non-null	float64
36	_Chromagram_Mean_4	400 non-null	float64
37	_Chromagram_Mean_5	400 non-null	float64
38	_Chromagram_Mean_6	400 non-null	float64
39	_Chromagram_Mean_7	400 non-null	float64
40	_Chromagram_Mean_8	400 non-null	float64
41	_Chromagram_Mean_9	400 non-null	float64
42	_Chromagram_Mean_10	400 non-null	float64
43	_Chromagram_Mean_11	400 non-null	float64
44	_Chromagram_Mean_12	400 non-null	float64
45	_HarmonicChangeDetectionFunction_Mean	400 non-null	float64
46	_HarmonicChangeDetectionFunction_Std	400 non-null	float64
47	_HarmonicChangeDetectionFunction_Slope	400 non-null	float64
48	_HarmonicChangeDetectionFunction_PeriodFreq	400 non-null	float64
49	_HarmonicChangeDetectionFunction_PeriodAmp	400 non-null	float64
50	_HarmonicChangeDetectionFunction_PeriodEntropy	400 non-null	float64

```
data.Class.value_counts(normalize=True)
```

```
relax    0.25
happy    0.25
sad      0.25
angry    0.25
Name: Class, dtype: float64
```

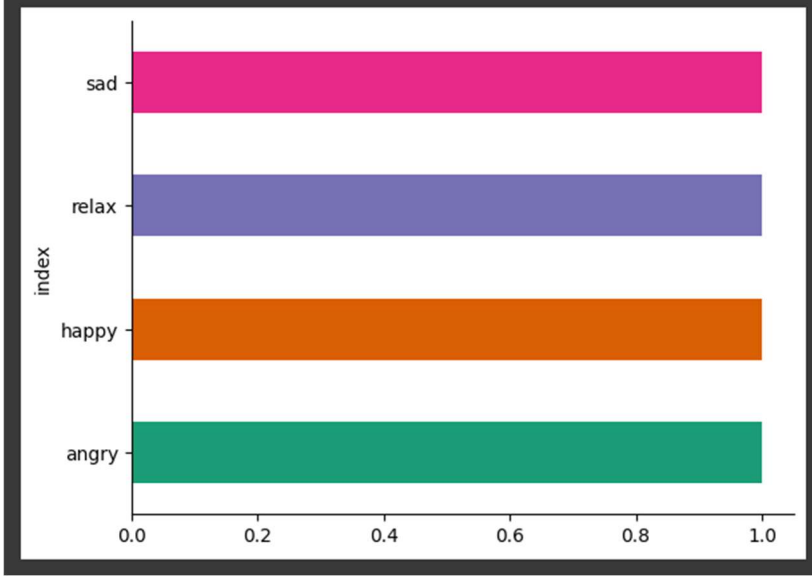
```
data.isnull().any().sum()
```

```
0
```

```
song_types=data["Class"].value_counts()
song_types_df=pd.DataFrame(song_types)
song_types_df=song_types.reset_index(level=0)
song_types_df
```

	index	Class
0	relax	100
1	happy	100
2	sad	100
3	angry	100

```
song_types_df.groupby('index').size().plot(kind='barh', color=sns.palettes.mpl_palette('Dark2'))
plt.gca().spines[['top', 'right',]].set_visible(False)
```



```
y = data['Class']
X = data.drop(columns=['Class'])

# Eğitim ve test setlerine ayırma
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1334)
le=LabelEncoder()
y_train=le.fit_transform(y_train)
y_test=le.fit_transform(y_test)

# Karar ağacı sınıflandırıcısını oluşturma
clf = DecisionTreeClassifier(max_depth=15)

# Çapraz doğrulama ile modelin performansını değerlendirme
cv_scores = cross_val_score(clf, X_train, y_train, cv=5)
```

```
print("Cross-validation Scores:")
print(cv_scores)
print("Mean Cross-validation Score:", cv_scores.mean())
```

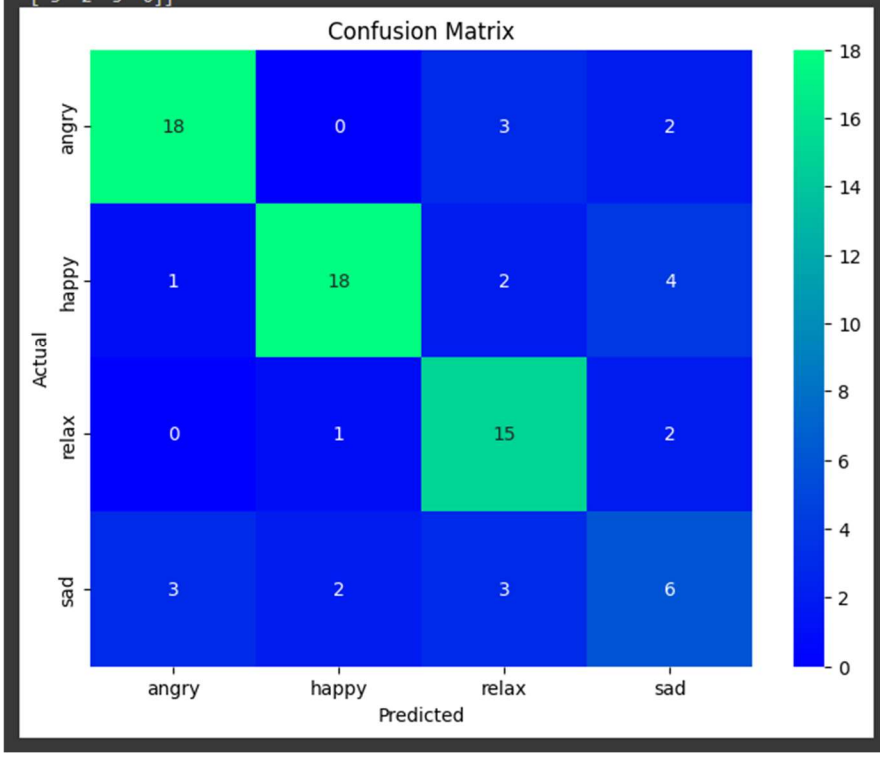
```
Cross-validation Scores:
[0.640625 0.671875 0.609375 0.765625 0.6875  ]
Mean Cross-validation Score: 0.675
```

```
clf.fit(X_train, y_train)

# Test seti üzerinde tahmin yapma
test_predictions = clf.predict(X_test)

# Test seti doğruluk ölçümü
test_acc = accuracy_score(y_test, test_predictions)
print("\nTest Accuracy:", test_acc)
```

```
Test Accuracy: 0.7125
```

```python
# Confusion matrix'i hesaplama
cm = confusion_matrix(y_test, test_predictions)
print("\nConfusion Matrix:")
print(cm)

# Confusion matrix'i görselleştirme
plt.figure(figsize=(8, 6))
sns.heatmap(cm,xticklabels=le.classes_,yticklabels=le.classes_,annot=True,cmap="winter")
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

```
Confusion Matrix:
[[18  0  3  2]
 [ 1 18  2  4]
 [ 0  1 15  2]
 [ 3  2  3  6]]
```



```python
# Etiketler ve değerler
labels = ['Validation Accuracy', 'Model Accuracy']
values = [(cv_scores.mean()),(test_acc)]

# Bar grafik oluşturma
plt.figure(figsize=(6, 8))
plt.bar(labels, values, color=['blue', 'green'])

# Değerleri gösterme
for i, value in enumerate(values):
    plt.text(i, value + 0.01, f"{value:.4f}", ha='center', va='bottom', fontsize=12)

# Eksenler ve başlık
plt.xlabel('Metrics')
plt.ylabel('Accuracy')
plt.title('Validation Accuracy vs Model Accuracy')

plt.show()
```