MAJOR PROJECT

NAME: K.V SRAVAN KUMAR

COLLEGE: SRM UNIVERSITY, RAMAPURAM

BRANCH: CSE-AIML

YEAR: 3RD YEAR

DEGREE: B.TECH

PHONE.NO: 9392769066

MAIL ID: kotamarthysravan2002@gmail.com

MAJOR PROJECT 1

Choose any dataset of your choice and apply a suitable CLASSIFIER/REGRESSOR .

PROCEDURE:

In [3]:
```
# IMPORT REQUIRED LIBRARIES
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
df=pd.read_csv('music.csv.csv')
df
```

Out[3]:

	age	gender	genre
0	20	1	HipHop
1	23	1	HipHop
2	25	1	HipHop
3	26	1	Jazz
4	29	1	Jazz
5	30	1	Jazz
6	31	1	Classical
7	33	1	Classical
8	37	1	Classical
9	20	0	Dance
10	21	0	Dance

In [23]:
```
# ASSUMING INPUT X AS INDEPENDENT VARIABLES
x=df.drop(columns=['genre'])
x
```

Out[23]:

	age	gender
0	20	1
1	23	1
2	25	1
3	26	1
4	29	1
5	30	1
6	31	1
7	33	1
8	37	1
9	20	0
10	21	0
11	25	0

In [24]:
```
# OUTPUT Y AS DEPENDENT VARIABLES
y=df['genre']
y
```

Out[24]:
```
0        HipHop
1        HipHop
2        HipHop
3          Jazz
4          Jazz
5          Jazz
6     Classical
7     Classical
8     Classical
9         Dance
10        Dance
11        Dance
12     Acoustic
13     Acoustic
14     Acoustic
15    Classical
16    Classical
17    Classical
Name: genre, dtype: object
```

[25]:
```
# BY USING DECISION TREE CLASSIFIER ALGORITHM
from sklearn.tree import DecisionTreeClassifier
model=DecisionTreeClassifier()
model.fit(x,y)
```

[25]:
```
DecisionTreeClassifier()
```

[26]:
```
# PREDICTIONS
pred=model.predict([[21,1],[22,0]])
pred
```

[26]:
```
array(['HipHop', 'Dance'], dtype=object)
```

[27]:
```
# TRAINGING THE BOTH X AND Y DATA
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=0)
```

[28]:
```
# FITTING THE DATA INTO THE ALGO
model.fit(x_train,y_train)
```

[28]:
```
DecisionTreeClassifier()
```

[29]:
```
# PREDICTING THE VALUES OF Y
y_pred=model.predict(x_test)
y_pred
```

[29]:
```
array(['HipHop', 'Classical', 'Classical', 'Dance', 'Acoustic'],
      dtype=object)
```

[31]:
```
# CALCULATING THE ACCURACY OF THE MODEL
from sklearn.metrics import accuracy_score
score=accuracy_score(y_test,y_pred)*100
score
```

[31]:
```
100.0
```

[41]:
```
# DUMPING THE ABOVE DATA INTO THE SEPERATE FILE
import joblib
joblib.dump(model,'music-rec.joblib')
```

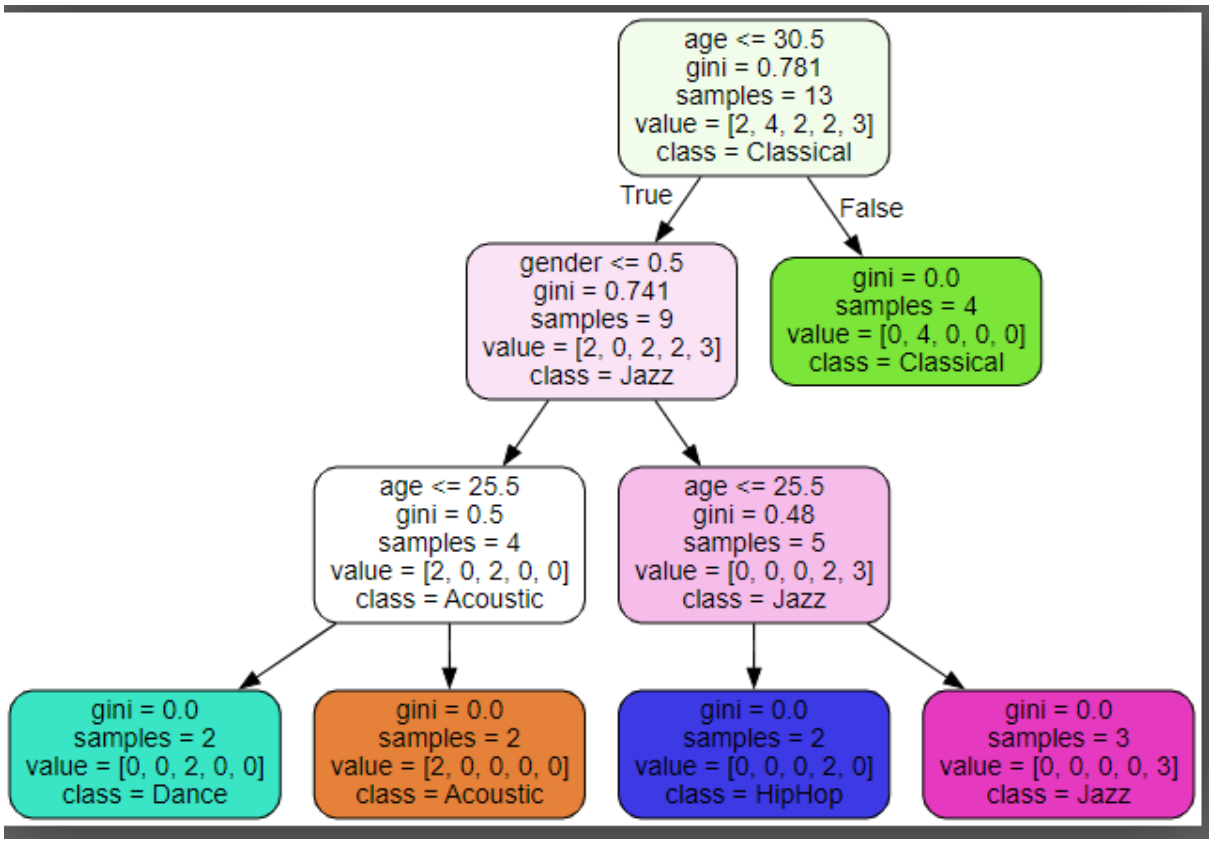
[41]:
```
['music-rec.joblib']
```

[42]:
```
# FINAL OUTPUT BY TREE MODEL
from sklearn import tree
tree.export_graphviz(model,out_file='music.rec.dot',feature_names=['age','gender'],class_names=sorted(y.unique()),label='all',
                     rounded=True,filled=True)
```

By opening the separate music.rec.dot file we get the below .DOT code

```
digraph Tree {
node [shape=box, style="filled, rounded", color="black", fontname=helvetica] ;
edge [fontname=helvetica] ;
0 [label="age <= 30.5\ngini = 0.781\nsamples = 13\nvalue = [2, 4, 2, 2, 3]\nclass = Classical", fillcolor="#f2fceb"] ;
1 [label="gender <= 0.5\ngini = 0.741\nsamples = 9\nvalue = [2, 0, 2, 2, 3]\nclass = Jazz", fillcolor="#fbe3f6"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="age <= 25.5\ngini = 0.5\nsamples = 4\nvalue = [2, 0, 2, 0, 0]\nclass = Acoustic", fillcolor="#ffffff"] ;
1 -> 2 ;
3 [label="gini = 0.0\nsamples = 2\nvalue = [0, 0, 2, 0, 0]\nclass = Dance", fillcolor="#39e5c5"] ;
2 -> 3 ;
4 [label="gini = 0.0\nsamples = 2\nvalue = [2, 0, 0, 0, 0]\nclass = Acoustic", fillcolor="#e58139"] ;
2 -> 4 ;
5 [label="age <= 25.5\ngini = 0.48\nsamples = 5\nvalue = [0, 0, 0, 2, 3]\nclass = Jazz", fillcolor="#f6bdea"] ;
1 -> 5 ;
6 [label="gini = 0.0\nsamples = 2\nvalue = [0, 0, 0, 2, 0]\nclass = HipHop", fillcolor="#3c39e5"] ;
5 -> 6 ;
7 [label="gini = 0.0\nsamples = 3\nvalue = [0, 0, 0, 0, 3]\nclass = Jazz", fillcolor="#e539c0"] ;
5 -> 7 ;
8 [label="gini = 0.0\nsamples = 4\nvalue = [0, 4, 0, 0, 0]\nclass = Classical", fillcolor="#7be539"] ;
0 -> 8 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
}
```

Output:



MAJOR PROJECT 2

Create any of the Image Processing Projects using NumPy and/or OpenCV. (Projects done in the class are not accepted)
(One can use the haarcasacde models if necessary)

PROCEDURE:

In [4]:
```
# IMPORT REQUIRED LIBRARIES
import cv2
import numpy as np
from sklearn.metrics import pairwise
```

In [6]:
```
# ALLOWING WEBCAM TO ACCESS
cap=cv2.VideoCapture(0)
ko=np.ones((5,5))
kc=np.ones((20,20))
lb=np.array([20,100,100])
ub=np.array([120,255,255])
# READING THE VALUES AND PROVIDING TO THE WEBCAM
while True:
    ret,frame=cap.read()
    flipped=cv2.flip(frame,1)
    flipped=cv2.resize(flipped,(500,400))
    imgseg=cv2.cvtColor(frame,cv2.COLOR_BGR2HSV)
    imgsegflipped=cv2.flip(imgseg,1)
    imgsegflipped=cv2.resize(imgsegflipped,(500,400))
    mask=cv2.inRange(imgsegflipped,lb,ub)
    mask=cv2.resize(mask,(500,400))
    mo=cv2.morphologyEx(mask,cv2.MORPH_OPEN,ko)
    mo=cv2.resize(mo,(500,400))
    mc=cv2.morphologyEx(mask,cv2.MORPH_CLOSE,kc)
    mc=cv2.resize(mc,(500,400))
```

```
    final=mc
    conts,h=cv2.findContours(mc,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_NONE)
    if(len(conts)!=0):
        b=max(conts,key=cv2.contourArea)
        west=tuple(b[b[:,:,0].argmin()][0])
        east=tuple(b[b[:,:,0].argmin()][0])
        north=tuple(b[b[:,:,1].argmin()][0])
        south=tuple(b[b[:,:,1].argmin()][0])
        centre_x=(west[0]+east[0])/2
        centre_y=(north[0]+south[0])/2

        cv2.drawContours(flipped,b,-1,(0,255,0),3)
        cv2.circle(flipped,west,6,(0,0,255),-1)
        cv2.circle(flipped,east,6,(0,0,255),-1)
        cv2.circle(flipped,north,6,(0,0,255),-1)
        cv2.circle(flipped,south,6,(0,0,255),-1)
        cv2.circle(flipped,(int(centre_x),int(centre_y)),6,(0,0,255),-1)

    cv2.imshow('video',flipped)
    if cv2.waitKey(0)==ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

Output:

Live video

Github link: https://github.com/kvsravan/MAJOR-PROJECT

CONCLUSION:

Therefore, I have successfully executed both the programs and learned about the Decision Tree Classifier and Opencv