**Program3:** ID3 Algorithm.

*# Algorithm : ID3-(Iterative Dichotomiser)***import** math  
**import** pandas **as** pd  
**import** numpy **as** np  
main\_df = pd.read\_csv(**"dataSet.csv"**)  
uniqueValues = main\_df.iloc[:, -1].unique()  
attributes = main\_df.columns[1:-1]  
*# -------------------------------------overall Entropy--------------------------------------***def** getEntropy(A,B):  
 total = A + B  
 **if** A == 0 **and** B == 0:  
 **return** -1  
 **if** B <= 0:  
 **return** 0  
 **if** A <= 0:  
 **return** 0  
 **else**:  
 **return** -(A / total) \* math.log2(A / total) - (B / total) \* math.log2(B / total)  
  
**def** getEntropyByDF(df):  
 *# get unique values and count* uniqueValues = df.iloc[:, -1].unique()  
 uniqueCount = df.iloc[:, -1].value\_counts()  
 *# print(uniqueValues,uniqueCount)* A = int(uniqueCount.iloc[0])  
 **if** len(uniqueValues)>1:  
 B = int(uniqueCount.iloc[1])  
 **else**:  
 B = 0  
 *# print(A,B)  
 # get overall main entropy* **return** getEntropy(A, B)  
  
  
*# -------------------------------------Info Gain--------------------------------------  
# get infogain***def** getInfoGain(attr, totalEntropy, dataFrame):  
 tempArray = dataFrame[[attr, dataFrame.columns[-1]]].to\_numpy()  
 *# print(attr,":")  
 # print(tempArray)* totalEle = len(tempArray)  
 *# get unique elements name* elements = np.unique(tempArray[:, 0])  
 *# print(elements)* eleDic = {}  
 eleAvg = {}  
 InfoGain = 0  
 **for** ele **in** elements:  
 countA = 1  
 countB = 1  
 eleCount = 0  
 **for** values **in** tempArray:  
 **if** ele == values[0]:  
 **if** values[1] == uniqueValues[0]:  
 eleDic.setdefault(ele, {})[uniqueValues[0]] = countA  
 countA += 1  
 **elif** values[1] == uniqueValues[1]:  
 eleDic.setdefault(ele, {})[uniqueValues[1]] = countB  
 countB += 1  
 eleCount += 1  
 eleDic.setdefault(ele, {})[**'count'**] = eleCount  
 **for** key, values **in** eleDic.items():  
 A = eleDic[key].get(uniqueValues[0], 0)  
 B = eleDic[key].get(uniqueValues[1], 0)  
 eleAvg[key] = (((A + B) / totalEle) \* getEntropy(A, B)).\_\_round\_\_(4)  
 Esum = sum(eleAvg.values()).\_\_round\_\_(4)  
  
 InfoGain = (totalEntropy - Esum).\_\_round\_\_(4)  
 print(**'in getInfoGain Fun for '**, attr,**"AvgEntropy & infogain:\n"**, Esum, InfoGain)  
 **return** InfoGain  
  
  
*# --------------------------------------------------------------------------------------------  
  
# get unique values and count  
# uniqueCount = main\_df.iloc[:, -1].value\_counts()  
# A = int(uniqueCount.iloc[0])  
# B = int(uniqueCount.iloc[1])  
# get overall main entropy*total\_e = getEntropyByDF(main\_df).\_\_round\_\_(4)  
infoGain = {}  
  
*# root-----------------------------------------------------------------------------*root = **None  
for** attr **in** attributes:  
 infoGain[attr] = getInfoGain(attr, total\_e, main\_df)  
root = max(infoGain, key=infoGain.get)  
infoGain.clear()  
*# -----------------------------------------------------------------------------*rootEle = main\_df[root].unique()  
subRoots = {}  
lstToRemove = [root]  
**for** ele **in** rootEle:  
 tempDf = main\_df.loc[main\_df[root]==ele].drop(lstToRemove,axis=1)  
 *# print(tempDf)* mainEn = getEntropyByDF(tempDf).\_\_round\_\_(4)  
 *# print(ele,mainEn)* **if** mainEn != 0:  
 **for** attr **in** tempDf.columns[1:-1]:  
 subRoots.setdefault(ele,{})[attr] = getInfoGain(attr,mainEn,tempDf)  
 **else**:  
 subRoots[ele] = 0  
 **if** subRoots[ele] != 0:  
 maximum = max(subRoots[ele],key=subRoots[ele].get)  
 lstToRemove.append(maximum)  
 subRoots[ele] = maximum  
  
print(**f'Root:{**root**}'**)  
print(subRoots)

Dataset :

***Day***,***Outlook***,***Temp***,***Humidity***,***Wind***,***PlayTennis  
D1***,***Sunny***,***Hot***,***High***,***Week***,***No  
D2***,***Sunny***,***Hot***,***High***,***Strong***,***No  
D3***,***Overcast***,***Hot***,***High***,***Week***,***Yes  
D4***,***Rain***,***Mild***,***High***,***Week***,***Yes  
D5***,***Rain***,***Cool***,***Normal***,***Week***,***Yes  
D6***,***Rain***,***Cool***,***Normal***,***Strong***,***No  
D7***,***Overcast***,***Cool***,***Normal***,***Strong***,***Yes  
D8***,***Sunny***,***Mild***,***High***,***Week***,***No  
D9***,***Sunny***,***Cool***,***Normal***,***Week***,***Yes  
D10***,***Rain***,***Mild***,***Normal***,***Week***,***Yes  
D11***,***Sunny***,***Mild***,***Normal***,***Strong***,***Yes  
D12*** ,***Overcast***,***Mild***,***High***,***Strong***,***Yes  
D13***,***Overcast***,***Hot***,***Normal***,***Week***,***Yes  
D14***,***Rain***,***Mild***,***High***,***Strong***,***No***

Output:

