Experiment NO 4: Implementation of ID3 algorithm

Name:Mushrifah Hasan

Roll no: CS116006

Date: 12 February 2019

**CODE:**

import math

def entropy(x, y):

if x == 0 or y == 0:

return 0

elif x == y:

return 1

a = x / (x + y)

b = y / (x + y)

ent = (-1) \* a \* math.log(a, 2) - b \* math.log(b, 2)

return ent

Income = ["Very High", "High", "Low", "High", "Very High", "Medium", "High", "Medium", "Low", "Low", "High", "Medium"]

Age = ["Young", "Medium", "Young", "Medium", "Medium", "Young", "Old", "Medium", "Medium", "Old", "Young", "Old"]

OH = ["Yes", "Yes", "Rented", "Yes", "Yes", "Yes", "Yes", "Rented", "Rented", "Rented", "Yes", "Rented"]

p = OH.count("Yes")

n = OH.count("Rented")

I = entropy(p, n)

print("Entropy:", I)

# Consider Income

print("\nIncome")

i = [0] \* 4

pi = [0] \* 4

ni = [0] \* 4

entinc = [0] \* 4

i[0] = Income.count("Very High")

i[1] = Income.count("High")

i[2] = Income.count("Medium")

i[3] = Income.count("Low")

unqInc = ["Very High", "High", "Medium", "Low"]

for j in range(0, len(i)):

pos = 0

pos = Income[pos:].index(unqInc[j])

if OH[pos] == "Yes":

pi[j] = pi[j] + 1

else:

ni[j] = ni[j] + 1

temp = 0

for k in range(0, i[j] - 1):

temp = Income[pos + 1:].index(unqInc[j])

pos = pos + temp + 1

if OH[pos] == "Yes":

pi[j] = pi[j] + 1

else:

ni[j] = ni[j] + 1

entinc[j] = entropy(pi[j], ni[j])

inc = 0

for i in range(0, len(unqInc)):

inc = inc + ((pi[i] + ni[i]) / 12) \* entinc[i]

print("Entropy Of Income: ", inc)

gainInc = I - inc

print("Gain Of Income: ", gainInc)

# Consider Age

print("\nAge")

i = [0] \* 3

pa = [0] \* 3

na = [0] \* 3

entage = [0] \* 3

i[0] = Age.count("Young")

i[1] = Age.count("Medium")

i[2] = Age.count("Old")

unqAge = ["Young", "Medium", "Old"]

for j in range(0, len(i)):

pos = 0

pos = Age[pos:].index(unqAge[j])

if OH[pos] == "Yes":

pa[j] = pa[j] + 1

else:

na[j] = na[j] + 1

temp = 0

for k in range(0, i[j] - 1):

temp = Age[pos + 1:].index(unqAge[j])

pos = pos + temp + 1

if OH[pos] == "Yes":

pa[j] = pa[j] + 1

else:

na[j] = na[j] + 1

entage[j] = entropy(pa[j], na[j])

inc = 0

for i in range(0, len(unqAge)):

inc = inc + ((pa[i] + na[i]) / 12) \* entage[i]

print("Entropy Of Age: ", inc)

gainAge = I - inc

print("Gain Of Age: ", gainAge)

# finding root

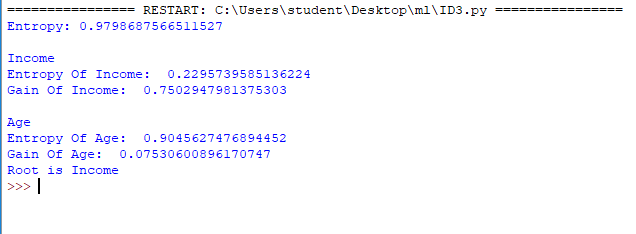
if(gainAge>gainInc):

print("Root is Age")

else:

print("Root is Income")

**OUTPUT:**

****