4.Write a program to demonstrate the working of the decision tree based ID3 algorithm. Use an appropriate data set for building the decision tree and apply this knowledge to classify a new sample

```
import pandas as pd

import math

import numpy as np



data = pd.read_csv("Dataset/4-dataset.csv")

features = [feat for feat in data]

features.remove("answer")
```

```
class Node:

    def __init__(self):

        self.children = []

        self.value = ""

        self.isLeaf = False

        self.pred = ""

def entropy(examples):

    pos = 0.0

    neg = 0.0

    for _, row in examples.iterrows():
```

```
        if row["answer"] == "yes":
            pos += 1
        else:
            neg += 1
    if pos == 0.0 or neg == 0.0:
        return 0.0
    else:
        p = pos / (pos + neg)
        n = neg / (pos + neg)
        return -(p * math.log(p, 2) + n * math.log(n, 2))
def info_gain(examples, attr):
    uniq = np.unique(examples[attr])
    #print ("\n",uniq)
    gain = entropy(examples)
    #print ("\n",gain)
    for u in uniq:
        subdata = examples[examples[attr] == u]
        #print ("\n",subdata)
        sub_e = entropy(subdata)
```

```
        gain -= (float(len(subdata)) / float(len(examples))) *
sub_e

        #print ("\n",gain)

    return gain

def ID3(examples, attrs):

    root = Node()



    max_gain = 0

    max_feat = ""

    for feature in attrs:

        #print ("\n",examples)

        gain = info_gain(examples, feature)

        if gain > max_gain:

            max_gain = gain

            max_feat = feature

    root.value = max_feat

    #print ("\nMax feature attr",max_feat)

    uniq = np.unique(examples[max_feat])

    #print ("\n",uniq)

    for u in uniq:
```

```
        #print ("\n",u)
        subdata = examples[examples[max_feat] == u]
        #print ("\n",subdata)
        if entropy(subdata) == 0.0:
            newNode = Node()
            newNode.isLeaf = True
            newNode.value = u
            newNode.pred = np.unique(subdata["answer"])
            root.children.append(newNode)
        else:
            dummyNode = Node()
            dummyNode.value = u
            new_attrs = attrs.copy()
            new_attrs.remove(max_feat)
            child = ID3(subdata, new_attrs)
            dummyNode.children.append(child)
            root.children.append(dummyNode)

    return root
def printTree(root: Node, depth=0):
```

```
    for i in range(depth):
        print("\t", end="")
    print(root.value, end="")
    if root.isLeaf:
        print(" -> ", root.pred)
    print()
    for child in root.children:
        printTree(child, depth + 1)
def classify(root: Node, new):
    for child in root.children:
        if child.value == new[root.value]:
            if child.isLeaf:
                print ("Predicted Label for new example", new," 
is:", child.pred)
                exit
            else:
                classify (child.children[0], new)
root = ID3(data, features)
print("Decision Tree is:")
printTree(root)
```

```
print ("-------------------")



new = {"outlook":"sunny", "temperature":"hot",
"humidity":"normal", "wind":"strong"}

classify (root, new)
```

OUTPUT:

```
Decision Tree is:
outlook
      overcast ->  ['yes']

      rain
            wind
                  strong ->  ['no']

                  weak ->  ['yes']

      sunny
            humidity
                  high ->  ['no']

                  normal ->  ['yes']

------------------
Predicted Label for new example {'outlook': 'sunny',
'temperature': 'hot', 'humidity': 'normal', 'wind': 'strong'}
is: ['yes']
```