Ex.no:2 **Decision Tree ID3 Algorithm In Python**

AIM:

A program to demonstrate the working of the decision tree based ID3 algorithm.

ALGORITHM:

```
ID3(Examples, Target_attribute, Attributes)

Examples are the training examples.
Target_attribute is the attribute whose value is to be
predicted by the tree.
Attributes is a list of other attributes that may be tested by
the learned decision tree.
Returns a decision tree that correctly classifies the given
Examples.

Create a Root node for the tree
If all Examples are positive, Return the single-node tree
Root, with label = +
If all Examples are negative, Return the single-node tree
Root, with label = -
If Attributes is empty, Return the single-node tree Root,
with label = most common value of Target_attribute in Examples

Otherwise Begin
   A ← the attribute from Attributes that best* classifies
Examples
   The decision attribute for Root ← A
   For each possible value, vi, of A,
      Add a new tree branch below Root, corresponding to the
test A = vi
      Let Examples vi, be the subset of Examples that have
value vi for A
      If Examples vi , is empty
         Then below this new branch add a leaf node with
         label = most common value of Target_attribute in
Examples
      Else
         below this new branch add the subtree
         ID3(Examples vi, Targe_tattribute, Attributes – {A}))
```

```
End
Return Root
```

(*NOTE: After completion of each block use the separate code IDE to perform the decision tree.

After completion on each block click on the code button to code the next block) Don't write the note in the observation note.

PROGRAM:

```
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)
```

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

        rain
                wind
                        strong ->  ['no']

                        weak ->  ['yes']

        sunny
                humidity
                        high ->  ['no']

                        normal ->  ['yes']
```

Dataset Link: CLICK HERE