16-Apr-21

classmate
Date
Page 1

03 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 numpy as np
import math

data = pd.read_csv("dataset.csv")
features = [f for f 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])
    gain = entropy(examples)
    for u in uniq:
        subdata = examples[examples[attr]==u]
        sub_e = entropy(subdata)
        gain -= (float(len(subdata))/float(len(examples)))*sub_e
    return gain

def ID3(examples, attrs):
    root = Node()

    max_gain = 0
    max_feat = ""
    for feature in attrs:
        gain = info_gain(examples, feature)
        if gain > max_gain:
            max_gain = gain
            max_feat = feature
    root.value = max_feature

    uniq = np.unique(examples[max_feat])
    for u in uniq:
        subdata = examples[examples[max_feat]==u]
        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
        dummyNode.value = u
        new_attr.remove(max_feat)
        child = ID3(subdata, new_attr)
        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)


root = ID3(data, features)
printTree(root).
```