3.1D3. 30.09.2020. Harsh R Jain 1BM17CS081

Program:

```
import numpy as np
import math
from data_loader import read_data

class Node:
    def __init__(self, attribute):
        self.attribute = attribute
        self.children = []
        self.ans = ''

    def __str__(self):
        return self.attribute

def subtables(data, col):
    dict = {}
    items = np.unique(data[:, col])
    count = np.zeros((items.shape[0], 1), dtype = int)
    for x in range(items.shape[0]):
        for y in range(data.shape[0]):
            if data[y, col] == items[x]:
                count[x] += 1

    for x in range(items.shape[0]):
        dict[items[x]] = np.empty((int(count[x]), data.shape[1]
dtype = 'S32')
        pos = 0
        for y in range(data.shape[0]):
```

TITLE : _____ Expt. No. : _____

Date : _____

Harish R
1BM17CS007

```
        if data[y, col] == items[x]:
            dict[items[x]][pos] = data[y]
            pos += 1
    dict[items[x]] = np.delete(dict[items[x]], col, 1)
    return items, dict

def entropy(s):
    items = np.unique(s)
    if items.size == 1:
        return 0
    count = np.zeros((items.shape[0], 1))
    sums = 0
    for x in range(items.shape[0]):
        counts[x] = sum(s == items[x]) / (s.size)
    for count in counts:
        sums += -1 * count * math.log(count, 2)
    return sums

gainratio(data, col):
    item, dict = subtables(data, col)
    total_size = data.shape[0]
    entropies = np.zeros((items.shape[0], 1))
```

Marks :
Staff :

P1

```
        for x in range(items.shape[0]):
            ratio = dict[items[x]].shape[0] / (total_size)
            entropies[x] = ratio * entropy(dict[items[x]])
    total_entropy = entropy(data[:, -1])
    for x in range(entropies.shape[0]):
        total_entropy -= entropies[x]
    return total_entropy
def create_node(data, metadata):
    if (np.unique(data[:, -1])).shape[0] == 1:
        node = Node("")
        node.answer = np.unique(data[:, -1])[0]
        return node
    gains = np.zeros((data.shape[1] - 1, 1))
    for col in range(data.shape[1] - 1):
        gains[col] = gain_ratio(data, col)
    split = np.argmax(gains)
    node = Node(metadata[split])
    metadata = np.delete(metadata, split, 0)
    items, dict = subtables(data, split)
    for x in range(items.shape[0]):
        child = create_node(dict[items[x]], metadata)
        node.children.append((items[x], child))
    return node
```

TITLE :

Expt. No. :

Date :

Harsh Pandey
1BM17CS031

```
def empty(size):
    s=""
    for x in range(size):
        s+=" "
    return s

def print_tree(node, level):
    if node.answer!="":
        print(empty(level), node.answer)
        return
    print(empty(level), node.attribute)
    for value, n in node.children:
        print(empty(level+1), value)
        print_tree(n, level+2)

metadata, traindata = read_data("training_examples.csv")
data = np.array(traindata)
node = create_node(data, metadata)
print_tree(node, 0)
```

Marks :
Staff :

Output 3:-

Sky
Rainy
b'No'

Sunny
b'Yes'