Date 27/10/20

Expt. No. 03

Milind Bijukumar - 4MT17CS058

Page No. 05

* Write a program to demonstrate the working of the decision tree based ID3 algorithm. Use an appropriate dataset for building the decision tree & apply this knowledge to classify a new sample.

*

```
import pandas as pd
from pandas import DataFrame
df_tennis=pd.read_csv('C:/User/Mili/onedrive/Desktop/Mite/4MT17CS058_Milind/
                      Playtennis.csv')
attribute_name =list (df_tennis.columns)
attribute_name.remove('Play Tennis')
print(attribute_name)
def entropy_of_list (lst):
    from collections import counter
    count= counter(x for x in lst)
    num_instance=len(lst)*1
    probs =[x] num_instance for x in count.values()]
    return entropy(probs)
def entropy(probs):
      import math
      return sum([-prob * math.log (prob,2) for prob in probs])
total_entropy =entropy_of_list (df_tennis ['Play Tennis'])
```

Teacher's Signature

Milind Bijukumar - 4MT17CS058

```
def information_gain(df,split_attribute_name, target_attribute_name,
                                        trace=0):
    df_split= df.groupby (split_attribute_name)
    nobs = len( df.index) * 1
     df_agg_ent =df_split.agg({target_attribute_name :
                       [entropy_of_list, lambda x : len(x)/nobs]})
     df_agg_ent.column =['Entropy', 'propobservations']
    new_entropy =sum (df.agg_ent ['Entropy'] * def_agg_ent
                                  ['propobservation'])
    old_entropy  =entropy_of_list (df[target_attribute_name]))
    print [split_attribute_name, 'IG',old_entropy_new entropy)
    return old entropy_new_entropy

def id3 (df, target attribute name, attribute_name, default class=none):
    from collections import counter
    count =counter (x for x in df [target_attribute_name])
    if len (count) ==1:
            return next(item (count))
    elif df.empty or(not attribute_name):
            return default_class
    else:
        default_class = max (count.keys ())
        gain = [information_gain (df,attr, target_attribute_name) for
                    attr in attribute_name]
    index_of_max =gain.index [max(gain)]
    best_attr =attribute_name [index_of_max]

    tree ={ best_attr :{ }}
```

Teacher's Signature

Date

Expt. No.

Milind Bijukumar - 4MT17CS055

Page No. 07

```
remaining_attribute_names = [i for i in attribute_name if
                              i != best_attr]
for attr_val, data_subset in df.groupby(best_attr):
subtree = id3(data_subset, target_attribute_name, remaining_
              attribute_name, default_class)
tree[best_attr][attr_val] = subtree
return tree

from pprint import pprint
tree = id3 [df_tennis, 'Play Tennis', attribute_name)
print("\n The Resultant Decision Tree is :\n")
pprint(tree)
```

Teacher's Signature

## output

['outlook', 'Temperature', 'Humidity', 'wind']

outlook IG : 0.2467498197744391

Temperature IG = 0.0299222565658954647

Humidity IG: 0.15183550136234136

wind IG: 0.04812703040826927

Temperature IG : 0.01997309402197489

Humidity: 0.01997309402197489

wind IG: 0.01997309402197489

The Resultant Decision tree is:

{'outlook' : {'overcast': 'Yes',
'Rain': {'wind' : {'Strongly' :'No', 'weak' : 'Yes'}},
'Sunny': {'Humidity' :{'High':'No', 'Normal' : 'Yes'}}}}