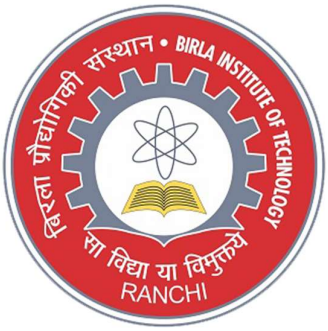
Birla Institute of Technology,Mesra,

Patna Campus



ML-Assignment

Name-Shubham Sourabh

Roll-Btech/15044/18

Sec-CSE 6th

#Assignment-3

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

Code:-

```
#Importing important libraries


import pandas as pd


from pandas import DataFrame


#Reading Dataset


df_tennis = pd.read_csv('data3.csv')


print( df_tennis)



#Function to calculate the entropy of probaility of observations

# -p*log2*p



def entropy(probs):
```

```
    import math
    return sum( [-prob*math.log(prob, 2) for prob in probs] )

#Function to calulate the entropy of the given Data Sets/List with respect to target attributes
def entropy_of_list(a_list):
    #print("A-list",a_list)
    from collections import Counter
    cnt = Counter(x for x in a_list)   # Counter calculates the propotion of class
   # print("\nClasses:",cnt)
    #print("No and Yes Classes:",a_list.name,cnt)
    num_instances = len(a_list)*1.0   # = 14
    probs = [x / num_instances for x in cnt.values()]  # x means no of YES/NO
    return entropy(probs) # Call Entropy :

# The initial entropy of the YES/NO attribute for our dataset.

total_entropy = entropy_of_list(df_tennis['PT'])

print("\n Total Entropy of PlayTennis Data Set:",total_entropy)

#Defining Information Gain Function
```

```
def information_gain(df, split_attribute_name, target_attribute_name,
trace=0):
    print("Information Gain Calculation of ",split_attribute_name)
    '''
    Takes a DataFrame of attributes, and quantifies the entropy of a target
    attribute after performing a split along the values of another attribute.
    '''
    # Split Data by Possible Vals of Attribute:
    df_split = df.groupby(split_attribute_name)
    for name,group in df_split:
        print("Name:\n",name)
        print("Group:\n",group)

    # Calculate Entropy for Target Attribute, as well as
    # Proportion of Obs in Each Data-Split
    nobs = len(df.index) * 1.0
    print("NOBS",nobs)
    df_agg_ent = df_split.agg({target_attribute_name : [entropy_of_list, lambda
x: len(x)/nobs] })[target_attribute_name]
    print([target_attribute_name])
    print(" Entropy List ",entropy_of_list)
    print("DFAGGENT",df_agg_ent)
    df_agg_ent.columns = ['Entropy', 'PropObservations']
    #if trace: # helps understand what fxn is doing:
        #print(df_agg_ent)

    # Calculate Information Gain:
```

```
    new_entropy = sum( df_agg_ent['Entropy'] *
df_agg_ent['PropObservations'] )
    old_entropy = entropy_of_list(df[target_attribute_name])
    return old_entropy - new_entropy



print('Info-gain for Outlook is :'+str(information_gain(df_tennis, 'Outlook',
'PT')),"\n")



#Defining ID3  Algorithm Function
def id3(df, target_attribute_name, attribute_names, default_class=None):

    ## Tally target attribute:
    from collections import Counter
    cnt = Counter(x for x in df[target_attribute_name])# class of YES /NO

    ## First check: Is this split of the dataset homogeneous?
    if len(cnt) == 1:
        return next(iter(cnt))  # next input data set, or raises StopIteration when
EOF is hit.

    ## Second check: Is this split of the dataset empty?
```

```
    # if yes, return a default value
    elif df.empty or (not attribute_names):
        return default_class  # Return None for Empty Data Set

    ## Otherwise: This dataset is ready to be devied up!
    else:
        # Get Default Value for next recursive call of this function:
        default_class = max(cnt.keys()) #No of YES and NO Class
        # Compute the Information Gain of the attributes:
        gainz = [information_gain(df, attr, target_attribute_name) for attr in
attribute_names] #
        index_of_max = gainz.index(max(gainz)) # Index of Best Attribute
        # Choose Best Attribute to split on:
        best_attr = attribute_names[index_of_max]

        # Create an empty tree, to be populated in a moment
        tree = {best_attr:{}} # Iniiate the tree with best attribute as a node
        remaining_attribute_names = [i for i in attribute_names if i != best_attr]

        # Split dataset
        # On each split, recursively call this algorithm.
        # populate the empty tree with subtrees, which
        # are the result of the recursive call
        for attr_val, data_subset in df.groupby(best_attr):
            subtree = id3(data_subset,
                        target_attribute_name,
                        remaining_attribute_names,
```

```
                default_class)
            tree[best_attr][attr_val] = subtree
        return tree



    # Get Predictor Names (all but 'class')
attribute_names = list(df_tennis.columns)
print("List of Attributes:", attribute_names)
attribute_names.remove('PT') #Remove the class attribute
print("\nPredicting Attributes:", attribute_names)

# Run Algorithm:
from pprint import pprint
tree = id3(df_tennis,'PT',attribute_names)
print("\n\nThe Resultant Decision Tree is :\n")
#print(tree)
pprint(tree)
attribute = next(iter(tree))
print("Best Attribute :\n",attribute)
print("Tree Keys:\n",tree[attribute].keys())



#classification accuracy
def classify(instance, tree, default=None): # Instance of Play Tennis with
Predicted
```

```
    #print("Instance:",instance)
    attribute = next(iter(tree)) # Outlook/Humidity/Wind
    print("Key:",tree.keys())  # [Outlook,Humidity,Wind ]
    print("Attribute:",attribute) # [Key /Attribute Both are same ]

    # print("Insance of Attribute :",instance[attribute],attribute)
    if instance[attribute] in tree[attribute].keys(): # Value of the attributs in  set of Tree keys
        result = tree[attribute][instance[attribute]]
        print("Instance Attribute:",instance[attribute],"TreeKeys :",tree[attribute].keys())
        if isinstance(result, dict): # this is a tree, delve deeper
            return classify(instance, result)
        else:
            return result # this is a label
    else:
        return default




df_tennis['predicted'] = df_tennis.apply(classify, axis=1, args=(tree,'No') )
    # classify func allows for a default arg: when tree doesn't have answer for a particular
    # combitation of attribute-values, we can use 'no' as the default guess

print(df_tennis['predicted'])
```

```
print('\n Accuracy is:' + str( sum(df_tennis['PT']==df_tennis['predicted'] ) / (1.0*len(df_tennis.index)) ))
```

```
df_tennis[['PT', 'predicted']]
```

```
training_data = df_tennis.iloc[1:-4] # all but last four instances
test_data  = df_tennis.iloc[-4:] # just the last four
train_tree = id3(training_data, 'PT', attribute_names)
```

```
test_data['predicted2'] = test_data.apply(classify,axis=1,args=(train_tree,'Yes') )
```

```
print ('\n\nAccuracy is : ' + str( sum(test_data['PT']==test_data['predicted2'] ) / (1.0*len(test_data.index)) ))
```

output:-

(note:- sir the output was very long so it was very difficult to post the screenshot of the output here so instead ive just copied the output and pasted it here hope you wont mind)

```
(mlenv) PS C:\Users\vampirepapi\Desktop\nowhere\6th-LABS\ML> python lab3.py
    Outlook Temperature Humidity   Windy   PT
0     Sunny       Hot     High    Weak   No
1     Sunny       Hot     High  Strong   No
2  Overcast       Hot     High    Weak  Yes
3     Rainy      Mild     High    Weak  Yes
4     Rainy      Cool   Normal    Weak  Yes
5     Rainy      Cool   Normal  Strong   No
6  Overcast      Cool   Normal  Strong  Yes
7     Sunny      Mild     High    Weak   No
8     Sunny      Cool   Normal    Weak  Yes
9     Rainy      Mild   Normal    Weak  Yes
10    Sunny      Mild   Normal  Strong  Yes
11 Overcast      Mild     High  Strong  Yes
12 Overcast       Hot   Normal    Weak  Yes
13    Rainy      Mild     High  Strong   No

 Total Entropy of PlayTennis Data Set: 0.9402859586706309
Information Gain Calculation of  Outlook
Name:
```

Overcast

Group:

Outlook Temperature Humidity Windy PT

2 Overcast Hot High Weak Yes

6 Overcast Cool Normal Strong Yes

11 Overcast Mild High Strong Yes

12 Overcast Hot Normal Weak Yes

Name:

Rainy

Group:

Outlook Temperature Humidity Windy PT

3 Rainy Mild High Weak Yes

4 Rainy Cool Normal Weak Yes

5 Rainy Cool Normal Strong No

9 Rainy Mild Normal Weak Yes

13 Rainy Mild High Strong No

Name:

Sunny

Group:

Outlook Temperature Humidity Windy PT

0 Sunny Hot High Weak No

1 Sunny Hot High Strong No

7 Sunny Mild High Weak No

8 Sunny Cool Normal Weak Yes

10 Sunny Mild Normal Strong Yes

NOBS 14.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Outlook

Overcast 0.000000 0.285714

Rainy 0.970951 0.357143

Sunny 0.970951 0.357143

Info-gain for Outlook is :0.2467498197744391

List of Attributes: ['Outlook', 'Temperature', 'Humidity', 'Windy', 'PT']

Predicting Attributes: ['Outlook', 'Temperature', 'Humidity', 'Windy']

Information Gain Calculation of Outlook

Name:

Overcast

Group:

	Outlook	Temperature	Humidity	Windy	PT
2	Overcast	Hot	High	Weak	Yes
6	Overcast	Cool	Normal	Strong	Yes
11	Overcast	Mild	High	Strong	Yes
12	Overcast	Hot	Normal	Weak	Yes

Name:

Rainy

Group:

	Outlook	Temperature	Humidity	Windy	PT
3	Rainy	Mild	High	Weak	Yes

4 Rainy Cool Normal Weak Yes

5 Rainy Cool Normal Strong No

9 Rainy Mild Normal Weak Yes

13 Rainy Mild High Strong No

Name:

Sunny

Group:

Outlook Temperature Humidity Windy PT

0 Sunny Hot High Weak No

1 Sunny Hot High Strong No

7 Sunny Mild High Weak No

8 Sunny Cool Normal Weak Yes

10 Sunny Mild Normal Strong Yes

NOBS 14.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Outlook

Overcast 0.000000 0.285714

Rainy 0.970951 0.357143

Sunny 0.970951 0.357143

Information Gain Calculation of Temperature

Name:

Cool

Group:

Outlook Temperature Humidity Windy PT

4 Rainy Cool Normal Weak Yes

5 Rainy Cool Normal Strong No

6 Overcast Cool Normal Strong Yes

8 Sunny Cool Normal Weak Yes

Name:

Hot

Group:

Outlook Temperature Humidity Windy PT

0 Sunny Hot High Weak No

1 Sunny Hot High Strong No

2 Overcast Hot High Weak Yes

12 Overcast Hot Normal Weak Yes

Name:

Mild

Group:

Outlook Temperature Humidity Windy PT

3 Rainy Mild High Weak Yes

7 Sunny Mild High Weak No

9 Rainy Mild Normal Weak Yes

10 Sunny Mild Normal Strong Yes

11 Overcast Mild High Strong Yes

13 Rainy Mild High Strong No

NOBS 14.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Temperature

Cool 0.811278 0.285714

Hot 1.000000 0.285714

Mild 0.918296 0.428571

Information Gain Calculation of Humidity

Name:

High

Group:

	Outlook	Temperature	Humidity	Windy	PT
0	Sunny	Hot	High	Weak	No
1	Sunny	Hot	High	Strong	No
2	Overcast	Hot	High	Weak	Yes
3	Rainy	Mild	High	Weak	Yes
7	Sunny	Mild	High	Weak	No
11	Overcast	Mild	High	Strong	Yes
13	Rainy	Mild	High	Strong	No

Name:

Normal

Group:

	Outlook	Temperature	Humidity	Windy	PT
4	Rainy	Cool	Normal	Weak	Yes
5	Rainy	Cool	Normal	Strong	No
6	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Cool	Normal	Weak	Yes
9	Rainy	Mild	Normal	Weak	Yes
10	Sunny	Mild	Normal	Strong	Yes

12 Overcast Hot Normal Weak Yes

NOBS 14.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Humidity

High 0.985228 0.5

Normal 0.591673 0.5

Information Gain Calculation of Windy

Name:

Strong

Group:

	Outlook	Temperature	Humidity	Windy	PT
1	Sunny	Hot	High	Strong	No
5	Rainy	Cool	Normal	Strong	No
6	Overcast	Cool	Normal	Strong	Yes
10	Sunny	Mild	Normal	Strong	Yes
11	Overcast	Mild	High	Strong	Yes
13	Rainy	Mild	High	Strong	No

Name:

Weak

Group:

	Outlook	Temperature	Humidity	Windy	PT
0	Sunny	Hot	High	Weak	No
2	Overcast	Hot	High	Weak	Yes
3	Rainy	Mild	High	Weak	Yes

4 Rainy Cool Normal Weak Yes

7 Sunny Mild High Weak No

8 Sunny Cool Normal Weak Yes

9 Rainy Mild Normal Weak Yes

12 Overcast Hot Normal Weak Yes

NOBS 14.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Windy

Strong 1.000000 0.428571

Weak 0.811278 0.571429

Information Gain Calculation of Temperature

Name:

Cool

Group:

Outlook Temperature Humidity Windy PT

4 Rainy Cool Normal Weak Yes

5 Rainy Cool Normal Strong No

Name:

Mild

Group:

Outlook Temperature Humidity Windy PT

3 Rainy Mild High Weak Yes

9 Rainy Mild Normal Weak Yes

13 Rainy Mild High Strong No

NOBS 5.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Temperature

Cool 1.000000 0.4

Mild 0.918296 0.6

Information Gain Calculation of Humidity

Name:

High

Group:

	Outlook	Temperature	Humidity	Windy	PT
3	Rainy	Mild	High	Weak	Yes
13	Rainy	Mild	High	Strong	No

Name:

Normal

Group:

	Outlook	Temperature	Humidity	Windy	PT
4	Rainy	Cool	Normal	Weak	Yes
5	Rainy	Cool	Normal	Strong	No
9	Rainy	Mild	Normal	Weak	Yes

NOBS 5.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Humidity

High 1.000000 0.4

Normal 0.918296 0.6

Information Gain Calculation of Windy

Name:

Strong

Group:

Outlook Temperature Humidity Windy PT

5 Rainy Cool Normal Strong No

13 Rainy Mild High Strong No

Name:

Weak

Group:

Outlook Temperature Humidity Windy PT

3 Rainy Mild High Weak Yes

4 Rainy Cool Normal Weak Yes

9 Rainy Mild Normal Weak Yes

NOBS 5.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Windy

Strong 0.0 0.4

Weak 0.0 0.6

Information Gain Calculation of Temperature

Name:

Cool

Group:

Outlook Temperature Humidity Windy PT

8 Sunny Cool Normal Weak Yes

Name:

Hot

Group:

Outlook Temperature Humidity Windy PT

0 Sunny Hot High Weak No

1 Sunny Hot High Strong No

Name:

Mild

Group:

Outlook Temperature Humidity Windy PT

7 Sunny Mild High Weak No

10 Sunny Mild Normal Strong Yes

NOBS 5.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Temperature

Cool 0.0 0.2

Hot 0.0 0.4

Mild 1.0 0.4

Information Gain Calculation of Humidity

Name:

High

Group:

Outlook Temperature Humidity Windy PT

0 Sunny Hot High Weak No

1 Sunny Hot High Strong No

7 Sunny Mild High Weak No

Name:

Normal

Group:

Outlook Temperature Humidity Windy PT

8 Sunny Cool Normal Weak Yes

10 Sunny Mild Normal Strong Yes

NOBS 5.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Humidity

High 0.0 0.6

Normal 0.0 0.4

Information Gain Calculation of Windy

Name:

Strong

Group:

Outlook Temperature Humidity Windy PT

1 Sunny Hot High Strong No

10 Sunny Mild Normal Strong Yes

Name:

Weak

Group:

Outlook Temperature Humidity Windy PT

0 Sunny Hot High Weak No

7 Sunny Mild High Weak No

8 Sunny Cool Normal Weak Yes

NOBS 5.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Windy

Strong 1.000000 0.4

Weak 0.918296 0.6

The Resultant Decision Tree is :

{'Outlook': {'Overcast': 'Yes',
'Rainy': {'Windy': {'Strong': 'No', 'Weak': 'Yes'}},
'Sunny': {'Humidity': {'High': 'No', 'Normal': 'Yes'}}}}

Best Attribute :

Outlook

Tree Keys:

dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Sunny TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Humidity'])

Attribute: Humidity

Instance Attribute: High TreeKeys : dict_keys(['High', 'Normal'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Sunny TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Humidity'])

Attribute: Humidity

Instance Attribute: High TreeKeys : dict_keys(['High', 'Normal'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Overcast TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Rainy TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Windy'])

Attribute: Windy

Instance Attribute: Weak TreeKeys : dict_keys(['Strong', 'Weak'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Rainy TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Windy'])

Attribute: Windy

Instance Attribute: Weak TreeKeys : dict_keys(['Strong', 'Weak'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Rainy TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Windy'])

Attribute: Windy

Instance Attribute: Strong TreeKeys : dict_keys(['Strong', 'Weak'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Overcast TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Sunny TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Humidity'])

Attribute: Humidity

Instance Attribute: High TreeKeys : dict_keys(['High', 'Normal'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Sunny TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Humidity'])

Attribute: Humidity

Instance Attribute: Normal TreeKeys : dict_keys(['High', 'Normal'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Rainy TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Windy'])

Attribute: Windy

Instance Attribute: Weak TreeKeys : dict_keys(['Strong', 'Weak'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Sunny TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Humidity'])

Attribute: Humidity

Instance Attribute: Normal TreeKeys : dict_keys(['High', 'Normal'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Overcast TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Overcast TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Rainy TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Windy'])

Attribute: Windy

Instance Attribute: Strong TreeKeys : dict_keys(['Strong', 'Weak'])

0 No

1 No

2 Yes

3 Yes

4 Yes

5 No

6 Yes

7 No

8 Yes

9 Yes

10 Yes

11 Yes

12 Yes

13 No

Name: predicted, dtype: object

Accuracy is:1.0

Information Gain Calculation of Outlook

Name:

Overcast

Group:

	Outlook	Temperature	Humidity	Windy	PT	predicted
2	Overcast	Hot	High	Weak	Yes	Yes
6	Overcast	Cool	Normal	Strong	Yes	Yes

Name:

Rainy

Group:

	Outlook	Temperature	Humidity	Windy	PT	predicted
3	Rainy	Mild	High	Weak	Yes	Yes
4	Rainy	Cool	Normal	Weak	Yes	Yes
5	Rainy	Cool	Normal	Strong	No	No
9	Rainy	Mild	Normal	Weak	Yes	Yes

Name:

Sunny

Group:

	Outlook	Temperature	Humidity	Windy	PT	predicted
1	Sunny	Hot	High	Strong	No	No
7	Sunny	Mild	High	Weak	No	No
8	Sunny	Cool	Normal	Weak	Yes	Yes

NOBS 9.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Outlook

Overcast 0.000000 0.222222

Rainy 0.811278 0.444444

Sunny 0.918296 0.333333

Information Gain Calculation of Temperature

Name:

Cool

Group:

	Outlook	Temperature	Humidity	Windy	PT	predicted
4	Rainy	Cool	Normal	Weak	Yes	Yes
5	Rainy	Cool	Normal	Strong	No	No
6	Overcast	Cool	Normal	Strong	Yes	Yes
8	Sunny	Cool	Normal	Weak	Yes	Yes

Name:

Hot

Group:

	Outlook	Temperature	Humidity	Windy	PT	predicted

1 Sunny Hot High Strong No No

2 Overcast Hot High Weak Yes Yes

Name:

Mild

Group:

Outlook Temperature Humidity Windy PT predicted

3 Rainy Mild High Weak Yes Yes

7 Sunny Mild High Weak No No

9 Rainy Mild Normal Weak Yes Yes

NOBS 9.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Temperature

Cool 0.811278 0.444444

Hot 1.000000 0.222222

Mild 0.918296 0.333333

Information Gain Calculation of Humidity

Name:

High

Group:

Outlook Temperature Humidity Windy PT predicted

1 Sunny Hot High Strong No No

2 Overcast Hot High Weak Yes Yes

3 Rainy Mild High Weak Yes Yes

7 Sunny Mild High Weak No No

Name:

Normal

Group:

	Outlook	Temperature	Humidity	Windy	PT	predicted
4	Rainy	Cool	Normal	Weak	Yes	Yes
5	Rainy	Cool	Normal	Strong	No	No
6	Overcast	Cool	Normal	Strong	Yes	Yes
8	Sunny	Cool	Normal	Weak	Yes	Yes
9	Rainy	Mild	Normal	Weak	Yes	Yes

NOBS 9.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Humidity

High 1.000000 0.444444

Normal 0.721928 0.555556

Information Gain Calculation of Windy

Name:

Strong

Group:

	Outlook	Temperature	Humidity	Windy	PT	predicted
1	Sunny	Hot	High	Strong	No	No
5	Rainy	Cool	Normal	Strong	No	No
6	Overcast	Cool	Normal	Strong	Yes	Yes

Name:

Weak

Group:

	Outlook	Temperature	Humidity	Windy	PT	predicted
2	Overcast	Hot	High	Weak	Yes	Yes
3	Rainy	Mild	High	Weak	Yes	Yes
4	Rainy	Cool	Normal	Weak	Yes	Yes
7	Sunny	Mild	High	Weak	No	No
8	Sunny	Cool	Normal	Weak	Yes	Yes
9	Rainy	Mild	Normal	Weak	Yes	Yes

NOBS 9.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Windy

Strong 0.918296 0.333333

Weak 0.650022 0.666667

Information Gain Calculation of Temperature

Name:

Cool

Group:

	Outlook	Temperature	Humidity	Windy	PT	predicted
4	Rainy	Cool	Normal	Weak	Yes	Yes
5	Rainy	Cool	Normal	Strong	No	No

Name:

Mild

Group:

	Outlook	Temperature	Humidity	Windy	PT	predicted

3 Rainy Mild High Weak Yes Yes

9 Rainy Mild Normal Weak Yes Yes

NOBS 4.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Temperature

Cool 1.0 0.5

Mild 0.0 0.5

Information Gain Calculation of Humidity

Name:

High

Group:

Outlook Temperature Humidity Windy PT predicted

3 Rainy Mild High Weak Yes Yes

Name:

Normal

Group:

Outlook Temperature Humidity Windy PT predicted

4 Rainy Cool Normal Weak Yes Yes

5 Rainy Cool Normal Strong No No

9 Rainy Mild Normal Weak Yes Yes

NOBS 4.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Humidity

High 0.000000 0.25

Normal 0.918296 0.75

Information Gain Calculation of Windy

Name:

Strong

Group:

Outlook Temperature Humidity Windy PT predicted

5 Rainy Cool Normal Strong No No

Name:

Weak

Group:

Outlook Temperature Humidity Windy PT predicted

3 Rainy Mild High Weak Yes Yes

4 Rainy Cool Normal Weak Yes Yes

9 Rainy Mild Normal Weak Yes Yes

NOBS 4.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Windy

Strong 0.0 0.25

Weak 0.0 0.75

Information Gain Calculation of Temperature

Name:

Cool

Group:

Outlook Temperature Humidity Windy PT predicted

8 Sunny Cool Normal Weak Yes Yes

Name:

Hot

Group:

Outlook Temperature Humidity Windy PT predicted

1 Sunny Hot High Strong No No

Name:

Mild

Group:

Outlook Temperature Humidity Windy PT predicted

7 Sunny Mild High Weak No No

NOBS 3.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Temperature

Cool 0.0 0.333333

Hot 0.0 0.333333

Mild 0.0 0.333333

Information Gain Calculation of Humidity

Name:

High

Group:

Outlook Temperature Humidity Windy PT predicted

1 Sunny Hot High Strong No No

7 Sunny Mild High Weak No No

Name:

Normal

Group:

Outlook Temperature Humidity Windy PT predicted

8 Sunny Cool Normal Weak Yes Yes

NOBS 3.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Humidity

High 0.0 0.666667

Normal 0.0 0.333333

Information Gain Calculation of Windy

Name:

Strong

Group:

Outlook Temperature Humidity Windy PT predicted

1 Sunny Hot High Strong No No

Name:

Weak

Group:

Outlook Temperature Humidity Windy PT predicted

7 Sunny Mild High Weak No No

8 Sunny Cool Normal Weak Yes Yes

NOBS 3.0

['PT']

Entropy List <function entropy_of_list at 0x000002204641BD08>

DFAGGENT entropy_of_list <lambda_0>

Windy

Strong 0.0 0.333333

Weak 1.0 0.666667

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Sunny TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Temperature'])

Attribute: Temperature

Instance Attribute: Mild TreeKeys : dict_keys(['Cool', 'Hot', 'Mild'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Overcast TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Overcast TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Outlook'])

Attribute: Outlook

Instance Attribute: Rainy TreeKeys : dict_keys(['Overcast', 'Rainy', 'Sunny'])

Key: dict_keys(['Windy'])

Attribute: Windy

Instance Attribute: Strong TreeKeys : dict_keys(['Strong', 'Weak'])

lab3.py:180: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.

Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
  test_data['predicted2'] =
test_data.apply(classify,axis=1,args=(train_tree,'Yes') )
```

Accuracy is : 0.75