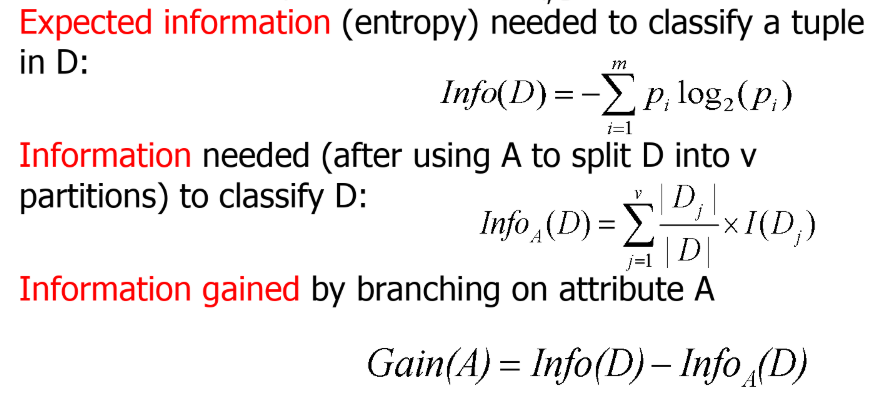
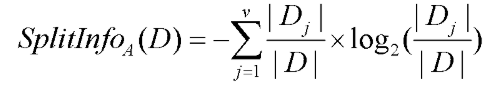
|  |  |
| --- | --- |
| Algorithm: C4.5 Decision Tree | |
| USN : 1MS17CS143 | NAME : Sathvik K P |
| USN : 1MS17CS148 | NAME : Sathvik B |

**Description of the Algorithm:**

C4.5 (a successor of ID3) uses gain ratio to overcome the problem (normalization to information gain)





**GainRatio(A) = Gain(A)/SplitInfo(A)**

C4.5 builds decision trees from a set of training data in the same way as ID3, using the concept of information entropy. The training data is a set S={s1,s2,...} of already classified samples. Each sample si consists of a p-dimensional vector.

At each node of the tree, C4.5 chooses the attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. The splitting criterion is the normalized information gain (difference in entropy). The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurses on the partitioned sublists.

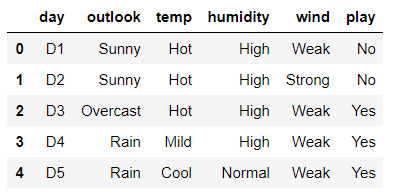
This algorithm has a few base cases.

1. All the samples in the list belong to the same class. When this happens, it simply creates a leaf node for the decision tree saying to choose that class.
2. None of the features provide any information gain. In this case, C4.5 creates a decision node higher up the tree using the expected value of the class.
3. Instance of previously-unseen class encountered. Again, C4.5 creates a decision node higher up the tree using the expected value.

**Algorithm Pseudocode:**

1. Check for the above base cases.
2. For each attribute *a*, find the normalized information gain ratio from splitting on *a*.
3. Let *a\_best* be the attribute with the highest normalized information gain.
4. Create a decision *node* that splits on *a\_best*.
5. Recurse on the sublists obtained by splitting on *a\_best*, and add those nodes as children of *node*.

**Data set Used: (Attach Screen shot of the few rows)**



Play-Tennis dataset

**Challenges faced during the implementation of the program:**

1. Exception cases where calculation of log(0) takes place.
2. Calculating split ratio each time.
3. Lack of resources to refer from.
4. Lack of good libraries to implement decision trees.

**Code:**

import pandas as pd

import numpy as np

from random import randint

from scipy import stats

from copy import deepcopy

from c45 import c45

df=pd.read\_csv('play\_tennis.csv')

df.info() ; df.head();

X=df[['outlook','temp','humidity','wind']]

Y=df.iloc[:, 5]

dt\_model = dt\_c45(Xdata = X, ydata = Y, cat\_missing = "missing", num\_missing = "mean", pre\_pruning = "impur", chi\_lim = 0.1, min\_lim = 5)

# Calling Prediction

testX = df.iloc[:, :5]

print('This is testX')

print(testX)

ans = prediction\_dt\_c45(dt\_model, testX)

ans.info()

y\_pred = ans.iloc[:, 0]

y\_pred.iloc

print(dt\_model)

c=0

for i in range(15):

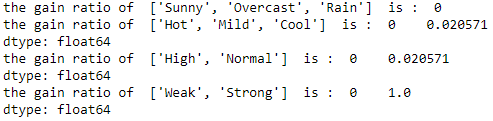
if y\_pred.iloc[i] == Y[i]:

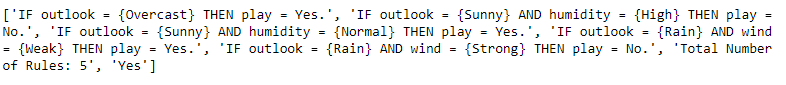
c+=1

print("Accuracy = ",(c/15))

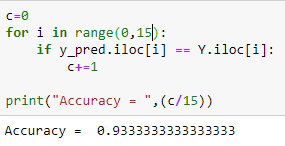
**Output: (Screen shots)**

Gain ratio:



Inference rules:  


Accuracy:



**References:**

1. <https://en.wikipedia.org/wiki/C4.5_algorithm>
2. <https://sefiks.com/2018/05/13/a-step-by-step-c4-5-decision-tree-example/>
3. PowerPoint presentations from last semester
4. https://www.quora.com/What-is-the-C4-5-algorithm-and-how-does-it-work