Darshan UNIVERSITY
योगः कर्मसु कौशलम्

Data Mining

Lab - 10

Enrollment No : 23010101014

Vishal Baraiya

Roll No : 137

Implement Decision Tree(ID3) in python

Uses Information Gain to choose the best feature to split.

Recursively builds the tree until stopping conditions are met.

1. Calculate Entropy for the dataset.
2. Calculate Information Gain for each feature.
3. Choose the feature with maximum Information Gain.
4. Split dataset into subsets for that feature.
5. Repeat recursively until:

All samples in a node have the same label.
No features are left.
No data is left.

Step 2. Import the dataset from this address.

import Pandas, Numpy

In [1]:
```
import pandas as pd
import numpy as np
```

Create Following Data

In [2]:
```
data = pd.DataFrame({
    'Outlook': ['Sunny', 'Sunny', 'Overcast', 'Rain', 'Rain', 'Rain', 'Overcast', '
    'Temperature': ['Hot', 'Hot', 'Hot', 'Mild', 'Cool', 'Cool', 'Cool', 'Mild', 'C
    'Humidity': ['High', 'High', 'High', 'High', 'Normal', 'Normal', 'Normal', 'Hig
    'Wind': ['Weak', 'Strong', 'Weak', 'Weak', 'Weak', 'Strong', 'Strong', 'Weak',
    'PlayTennis': ['No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'No', 'Yes', 'Yes'
})
```

In [3]:
```
data
```

Out[3]:

	Outlook	Temperature	Humidity	Wind	PlayTennis
0	Sunny	Hot	High	Weak	No
1	Sunny	Hot	High	Strong	No
2	Overcast	Hot	High	Weak	Yes
3	Rain	Mild	High	Weak	Yes
4	Rain	Cool	Normal	Weak	Yes
5	Rain	Cool	Normal	Strong	No
6	Overcast	Cool	Normal	Strong	Yes
7	Sunny	Mild	High	Weak	No
8	Sunny	Cool	Normal	Weak	Yes
9	Rain	Mild	Normal	Weak	Yes
10	Sunny	Mild	Normal	Strong	Yes
11	Overcast	Mild	High	Strong	Yes
12	Overcast	Hot	Normal	Weak	Yes
13	Rain	Mild	High	Strong	No

Now Define Function to Calculate Entropy

In [4]:
```
def entropy(y):
    unique_classes, counts = np.unique(y, return_counts=True)
    probability = counts / counts.sum()
    entropy = -np.sum(probability * np.log2(probability))
```

```
    return entropy
```

Testing of Above Function -

y = np.array(['Yes', 'No', 'Yes', 'Yes'])
Function Call - > entropy(y))

output - 0.8112781244591328

In [5]:
```
y = np.array(['Yes','No','Yes','Yes'])
entropy(y)
```

Out[5]:
```
0.8112781244591328
```

Define function to Calculate Information Gain

In [6]:
```
def information_gain(data, split_attribute, target):
    total_entropy = entropy(data[target])
    print("Total Entropy = ",total_entropy)
    unique_classes, counts = np.unique(data[split_attribute], return_counts=True)
    print(unique_classes)
    print(counts)
    weighted_entropy = 0
    for i in range(len(unique_classes)):
        subset = data[data[split_attribute] == unique_classes[i] ]
        print(subset)
        weighted_entropy += (counts[i]/counts.sum()) * entropy(subset[target])
        print(weighted_entropy)
    return total_entropy - weighted_entropy
```

Testing of Above Function-

data = pd.DataFrame({ 'Weather': ['Sunny', 'Sunny', 'Rain', 'Rain'], 'Play': ['Yes', 'No', 'Yes', 'Yes']
})

Function Call - > information_gain(data, 'Weather', 'Play')

Output - 0.31127812445913283

In [7]:
```
data1 = pd.DataFrame({ 'Weather': ['Sunny', 'Sunny', 'Rain', 'Rain'], 'Play': ['Yes
information_gain(data1, 'Weather', 'Play')
```

```
Total Entropy =  0.8112781244591328
['Rain' 'Sunny']
[2 2]
  Weather Play
2    Rain  Yes
3    Rain  Yes
0.0
  Weather Play
0   Sunny  Yes
1   Sunny   No
0.5
```

Out[7]: 0.31127812445913283

Implement ID3 Algo

In [8]:
```
def id3(data, features, target):
    # If all labels are same → return the label
    if(len(np.unique(data[target]))) == 1:
        return np.unique(data[target])[0]

    # If no features left → return majority label
    if len(features) == 0:
        return data[target].mode()[0]

    # Choose best feature
    gains = [information_gain(data,feature,target) for feature in features]
    best_feature =features[np.argmax(gains)]

    tree = {best_feature:{}}
    # For each value of best feature → branch
    for value in np.unique(data[best_feature]):
        sub_data = data[data[best_feature] == value].drop(columns=[best_feature])
        subtree = id3(sub_data,[f for f in features if f != best_feature],target)
        tree[best_feature][value]=subtree
    return tree
```

Use ID3

In [9]:
```
features = list(data.columns[:-1])
target = 'PlayTennis'
tree = id3(data,features,target)
```

```
Total Entropy =  0.9402859586706311
['Overcast' 'Rain' 'Sunny']
[4 5 5]
     Outlook Temperature Humidity    Wind PlayTennis
2   Overcast         Hot     High    Weak        Yes
6   Overcast        Cool   Normal  Strong        Yes
11  Overcast        Mild     High  Strong        Yes
12  Overcast         Hot   Normal    Weak        Yes
0.0
   Outlook Temperature Humidity    Wind PlayTennis
3     Rain        Mild     High    Weak        Yes
4     Rain        Cool   Normal    Weak        Yes
5     Rain        Cool   Normal  Strong         No
9     Rain        Mild   Normal    Weak        Yes
13    Rain        Mild     High  Strong         No
0.3467680694480959
   Outlook Temperature Humidity    Wind PlayTennis
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
0.6935361388961918
Total Entropy =  0.9402859586706311
['Cool' 'Hot' 'Mild']
[4 4 6]
    Outlook Temperature Humidity    Wind PlayTennis
4      Rain        Cool   Normal    Weak        Yes
5      Rain        Cool   Normal  Strong         No
6  Overcast        Cool   Normal  Strong        Yes
8     Sunny        Cool   Normal    Weak        Yes
0.23179374984546652
     Outlook Temperature Humidity    Wind PlayTennis
0      Sunny         Hot     High    Weak         No
1      Sunny         Hot     High  Strong         No
2   Overcast         Hot     High    Weak        Yes
12  Overcast         Hot   Normal    Weak        Yes
0.5175080355597522
     Outlook Temperature Humidity    Wind PlayTennis
3       Rain        Mild     High    Weak        Yes
7      Sunny        Mild     High    Weak         No
9       Rain        Mild   Normal    Weak        Yes
10     Sunny        Mild   Normal  Strong        Yes
11  Overcast        Mild     High  Strong        Yes
13      Rain        Mild     High  Strong         No
0.9110633930116763
Total Entropy =  0.9402859586706311
['High' 'Normal']
[7 7]
     Outlook Temperature Humidity    Wind PlayTennis
0      Sunny         Hot     High    Weak         No
1      Sunny         Hot     High  Strong         No
2   Overcast         Hot     High    Weak        Yes
3       Rain        Mild     High    Weak        Yes
7      Sunny        Mild     High    Weak         No
11  Overcast        Mild     High  Strong        Yes
```

```
13      Rain        Mild     High  Strong         No
0.49261406801712576
     Outlook Temperature Humidity    Wind PlayTennis
4       Rain        Cool   Normal    Weak        Yes
5       Rain        Cool   Normal  Strong         No
6   Overcast        Cool   Normal  Strong        Yes
8      Sunny        Cool   Normal    Weak        Yes
9       Rain        Mild   Normal    Weak        Yes
10     Sunny        Mild   Normal  Strong        Yes
12  Overcast         Hot   Normal    Weak        Yes
0.7884504573082896
Total Entropy =  0.9402859586706311
['Strong' 'Weak']
[6 8]
     Outlook Temperature Humidity    Wind PlayTennis
1      Sunny         Hot     High  Strong         No
5       Rain        Cool   Normal  Strong         No
6   Overcast        Cool   Normal  Strong        Yes
10     Sunny        Mild   Normal  Strong        Yes
11  Overcast        Mild     High  Strong        Yes
13      Rain        Mild     High  Strong         No
0.42857142857142855
     Outlook Temperature Humidity  Wind PlayTennis
0      Sunny         Hot     High  Weak         No
2   Overcast         Hot     High  Weak        Yes
3       Rain        Mild     High  Weak        Yes
4       Rain        Cool   Normal  Weak        Yes
7      Sunny        Mild     High  Weak         No
8      Sunny        Cool   Normal  Weak        Yes
9       Rain        Mild   Normal  Weak        Yes
12  Overcast         Hot   Normal  Weak        Yes
0.8921589282623617
Total Entropy =  0.9709505944546686
['Cool' 'Mild']
[2 3]
  Temperature Humidity    Wind PlayTennis
4        Cool   Normal    Weak        Yes
5        Cool   Normal  Strong         No
0.4
   Temperature Humidity    Wind PlayTennis
3         Mild     High    Weak        Yes
9         Mild   Normal    Weak        Yes
13        Mild     High  Strong         No
0.9509775004326937
Total Entropy =  0.9709505944546686
['High' 'Normal']
[2 3]
   Temperature Humidity    Wind PlayTennis
3         Mild     High    Weak        Yes
13        Mild     High  Strong         No
0.4
  Temperature Humidity    Wind PlayTennis
4        Cool   Normal    Weak        Yes
5        Cool   Normal  Strong         No
9        Mild   Normal    Weak        Yes
0.9509775004326937
```

```
Total Entropy =  0.9709505944546686
['Strong' 'Weak']
[2 3]
   Temperature Humidity    Wind PlayTennis
5         Cool   Normal  Strong         No
13        Mild     High  Strong         No
0.0
  Temperature Humidity  Wind PlayTennis
3        Mild     High  Weak        Yes
4        Cool   Normal  Weak        Yes
9        Mild   Normal  Weak        Yes
0.0
Total Entropy =  0.9709505944546686
['Cool' 'Hot' 'Mild']
[1 2 2]
  Temperature Humidity  Wind PlayTennis
8        Cool   Normal  Weak        Yes
0.0
  Temperature Humidity    Wind PlayTennis
0         Hot     High    Weak         No
1         Hot     High  Strong         No
0.0
   Temperature Humidity    Wind PlayTennis
7         Mild     High    Weak         No
10        Mild   Normal  Strong        Yes
0.4
Total Entropy =  0.9709505944546686
['High' 'Normal']
[3 2]
  Temperature Humidity    Wind PlayTennis
0         Hot     High    Weak         No
1         Hot     High  Strong         No
7        Mild     High    Weak         No
0.0
   Temperature Humidity    Wind PlayTennis
8         Cool   Normal    Weak        Yes
10        Mild   Normal  Strong        Yes
0.0
Total Entropy =  0.9709505944546686
['Strong' 'Weak']
[2 3]
   Temperature Humidity    Wind PlayTennis
1          Hot     High  Strong         No
10        Mild   Normal  Strong        Yes
0.4
  Temperature Humidity  Wind PlayTennis
0         Hot     High  Weak         No
7        Mild     High  Weak         No
8        Cool   Normal  Weak        Yes
0.9509775004326937
```

Print Tree

In [10]:
```
tree
```

Out[10]:
```
{'Outlook': {'Overcast': 'Yes',
  'Rain': {'Wind': {'Strong': 'No', 'Weak': 'Yes'}},
  'Sunny': {'Humidity': {'High': 'No', 'Normal': 'Yes'}}}}
```

Extra: Create Predict Function

In [11]:
```python
def predict(tree, sample):
    # If tree is a leaf node (not a dict), return the label directly
    if not isinstance(tree, dict):
        return tree

    # Extract the root feature
    root = next(iter(tree))

    # Get the value of this feature from the sample
    feature_value = sample[root]

    # If the value exists in the tree → follow that branch
    if feature_value in tree[root]:
        return predict(tree[root][feature_value], sample)
    else:
        # If unseen feature value → fallback (majority or None)
        return None
```

Extra: Predict for a sample

sample = {'Outlook': 'Sunny', 'Temperature': 'Cool', 'Humidity': 'High', 'Wind': 'Strong'}

Your Answer ?

In [12]:
```python
sample = {'Outlook': 'Sunny', 'Temperature': 'Cool', 'Humidity': 'High', 'Wind': 'S
predict(tree, sample)
```

Out[12]:
```
'No'
```

In [ ]: