Semester	T.E. Semester VI – Computer Engineering
Subject	Data Warehousing and Mining
Subject Professor In-charge	Prof. Kavita Shirsat
Assisting Teachers	Prof. Kavita Shirsat
Laboratory	Lab 312 A

Student Name	Deep Salunkhe	
Roll Number	21102A0014	
Grade and Subject Teacher's Signature		

Experiment Number	03	
Experiment Title	To perform the calculation of ID3 Algorithm	
Resources / Apparatus Required	Hardware: Computer system	Software: Python
Description	1. **Decision Tree Algorithm:** ID3 is a popular decision tree algorithm used for classification tasks. 2. **Objective:** It aims to build a decision tree that can be used to make decisions or predictions based on input features. 3. **Entropy-Based Approach:** ID3 uses an entropy-based approach to select the best attributes for splitting the data. It calculates the Information Gain for each attribute to determine the most informative one. 4. **Information Gain:** Information Gain measures the reduction in entropy (uncertainty) achieved by splitting the data on a particular attribute. The attribute with the highest Information Gain is chosen as the splitting criterion. 5. **Entropy:** Entropy is a measure of the impurity or randomness of data. Lower entropy indicates more ordered and predictable data, while higher entropy indicates randomness. 6. **Recursive Splitting:** ID3 recursively splits the dataset into subsets based on the selected attribute until a stopping criterion is met. The splitting continues until either all data points in a subset belong to the same class or a predefined depth limit is reached. 7. **Categorical Attributes:** ID3 is designed primarily for categorical (discrete) attributes, not continuous ones. It handles discrete attribute values well.	

Program

```
import pandas as pd
import math
```

```
data=pd.read_csv('Book1.csv')
data
```

	age	income	student	credit_rating	buys_computer
0	youth	high	no	fair	no
1	youth	high	no	excellent	no
2	middel_aged	high	no	fair	yes
3	senior	medium	no	fair	yes
4	senior	low	yes	fair	yes
5	senior	low	yes	excellent	no
6	middle_aged	low	yes	excellent	yes
7	youth	medium	no	fair	no
8	youth	low	yes	fair	yes
9	senior	medium	yes	fair	yes
10	youth	medium	yes	excellent	yes
11	middle_aged	medium	no	excellent	yes

```
# Define a function called calculate_entropy that takes a pandas Series
def calculate_entropy(data):
    # Count the occurrences of each unique value in the 'data' Series
    class_counts = data.value_counts()
    print(class_counts)

    # Get the total number of examples in the dataset
    total_examples = len(data)

    # Initialize the entropy variable to 0
    entropy = 0

    # Iterate through each count of unique values in 'class_counts'
    for count in class_counts:
        # Calculate the probability of a particular class occurrence
        p = count / total_examples

        # Calculate the entropy contribution of this class and subtract
        # The entropy formula: entropy = -p * log2(p)
        entropy -= p * math.log2(p)

    # Return the calculated entropy
    return entropy
```

```
initial_entropy = calculate_entropy(data["buys_computer"])
print(initial_entropy)
```

```
buys_computer
yes    9
no     5
Name: count, dtype: int64
0.9402859586706311
```

```
def calculate_information_gain(data, attribute, target_attribute):
    # Initialize 'info_gain' with the initial entropy of the entire dataset
    info_gain = initial_entropy

    # Initialize 'info_divergence' to 0, which will be used to calculate the weig
    info_divergence = 0

    # Get the total number of examples in the dataset
    total_examples = len(data)

    # Iterate through each unique value of the specified 'attribute' in the datas
    for value in data[attribute].unique():
        # Create a subset of the dataset where the 'attribute' has the current 'v
        subset = data[data[attribute] == value]

        # Get the size of the subset (number of examples with the current 'value'
        subset_size = len(subset)

        # Calculate the entropy of the subset based on the 'target_attribute'
        subset_entropy = calculate_entropy(subset[target_attribute])

        # Calculate the weighted average of subset entropies (info_divergence)
        info_divergence += (subset_size / total_examples) * subset_entropy

    # Subtract the weighted average entropy (info_divergence) from the initial en
    info_gain -= info_divergence

    # Return the calculated Information Gain
    return info_gain
```

```
# Calculate Information Gain and Information Divergence for each attribut
target_attribute = "buys_computer"
attributes = data.columns.drop(target_attribute)
```

```
ig_values = {}
```

```
for attribute in attributes:
    ig = calculate_information_gain(data, attribute, target_attribute)
    ig_values[attribute] = ig
```

<table>
<tr><td>Output</td><td>

```
# Print Information Gain
print("Information Gain:")
for x in ig_values:
    print(x,":",ig_values[x])
```

```
Information Gain:
age : 0.24674981977443933
income : 0.02922256565895487
student : 0.15183550136234159
credit_rating : 0.04812703040826949
```

</td></tr>
<tr><td>Conclusion:</td><td>ID3 employs an entropy-based approach to select the most informative attributes for splitting the dataset. It calculates Information Gain, which measures the reduction in uncertainty achieved by splitting the data based on a particular attribute.</td></tr>
</table>