In [21]:
```
# Decision Tree

import numpy as np
import pandas as pd
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import OneHotEncoder, LabelEncoder
from sklearn.compose import ColumnTransformer
```

In [24]:
```
# Import Dataset
df = pd.read_csv(r"C:\Users\91830\Desktop\DUK\AIML\DT\classifierdata.csv",
                           sep=',', header=None)

# Printing the dataset shape
print("Dataset Length: ", len(df))
print("Dataset Shape: ", df.shape)
print("")

# Printing the dataset observations
print("Dataset: \n", df.head())
print("")
```

```
Dataset Length:  15
Dataset Shape:  (15, 5)

Dataset:
         0       1        2              3              4
0    age  income  student  credit rating  buys computer
1   <=30    high       no           fair             no
2   <=30    high       no      excellent             no
3  31-40    high       no           fair            yes
4    >40  medium       no           fair            yes
```

In [26]:
```
# Separating the target variable
X = df.iloc[:, 1:]
y = df.iloc[:, 0]

# Convert target variable to numeric using LabelEncoder
enc = LabelEncoder()
y = enc.fit_transform(y)

# Use one-hot encoding for all categorical features
categorical_cols = [0, 1, 2, 3]  # assuming all columns are categorical
encoder = ColumnTransformer(transformers=[("OneHot", OneHotEncoder(), categorical_cols)],
                            remainder='passthrough')
X_encoded = encoder.fit_transform(X)
```

In [28]:
```
# Splitting the dataset into train and test
X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size=0.3,
                                                    random_state=100)

# Train using Gini index
clf_gini = DecisionTreeClassifier(criterion="gini", random_state=100,
                                  max_depth=3, min_samples_leaf=5)
clf_gini.fit(X_train, y_train)

# Train using entropy
clf_entropy = DecisionTreeClassifier(criterion="entropy", random_state=100,
                                     max_depth=3, min_samples_leaf=5)
clf_entropy.fit(X_train, y_train)
```

Out[28]:
```
DecisionTreeClassifier(criterion='entropy', max_depth=3, min_samples_leaf=5,
                       random_state=100)
```

In [34]:
```
# Results Using Entropy
y_pred_entropy = clf_entropy.predict(X_test)
print("Predicted values using Information Gain:")
print(enc.inverse_transform(y_pred_entropy))  # Convert back to original labels
print("")

# Results Using Gini Index
y_pred_gini = clf_gini.predict(X_test)
print("Predicted values using Gini Index:")
print(enc.inverse_transform(y_pred_gini))  # Convert back to original labels
print("")
```

```
Predicted values using Entropy:
['31-40' '31-40' '>40' '31-40' '31-40']

Predicted values using Gini Index:
['31-40' '31-40' '>40' '31-40' '31-40']
```

In [37]:
```
# Calculate accuracy using Entropy
print("Information Gain")
print("******************")
print("Confusion Matrix (IG): \n", confusion_matrix(y_test, y_pred_entropy))
print("Accuracy (IG): ", accuracy_score(y_test, y_pred_entropy) * 100)
print("Report (IG): \n", classification_report(y_test, y_pred_entropy,
                                               zero_division=1))
print("")

# Calculate accuracy using Gini Index
print("Gini Index ")
print("*************")
print("Confusion Matrix (Gini): \n", confusion_matrix(y_test, y_pred_gini))
print("")
print("Accuracy (Gini): ", accuracy_score(y_test, y_pred_gini) * 100)
print("Report (Gini): \n", classification_report(y_test, y_pred_gini,
                                                 zero_division=1))
print("")
```

```
Information Gain
******************
Confusion Matrix (IG):
 [[1 0 1]
 [2 0 0]
 [1 0 0]]
Accuracy (IG):  20.0
Report (IG):
               precision    recall  f1-score   support

           0       0.25      0.50      0.33         2
           1       1.00      0.00      0.00         2
           2       0.00      0.00      0.00         1

    accuracy                           0.20         5
   macro avg       0.42      0.17      0.11         5
weighted avg       0.50      0.20      0.13         5


Gini Index
*************
Confusion Matrix (Gini):
 [[1 0 1]
 [2 0 0]
 [1 0 0]]

Accuracy (Gini):  20.0
Report (Gini):
               precision    recall  f1-score   support

           0       0.25      0.50      0.33         2
           1       1.00      0.00      0.00         2
           2       0.00      0.00      0.00         1
```

accuracy			0.20	5
macro avg	0.42	0.17	0.11	5
weighted avg	0.50	0.20	0.13	5