Experiment-1.4

Student Name: Ashish Kumar **UID: 23MAI10008**
Branch: ME CSE AIML **Section/Group: 23MAI-1**
Semester: 02 **Date of Performance: 07/02/2024**
Subject Name: Machine Learning Lab **Subject Code: 23CSH-651**

Aim of the Experiment :

Implementing Decision Tree Algorithm using Python.

Theory :

Decision Tree is a tree-like structure that represents a set of decisions and their possible consequences. Each node in the tree represents a decision, and each branch represents an outcome of that decision. The leaves of the tree represent the final decisions or predictions.

Decision trees are created by recursively partitioning the data into smaller and smaller subsets. At each partition, the data is split based on a specific feature, and the split is made in a way that maximizes the information gain. It is used to address classification problems in statistics, data mining, and machine learning.

Example: Decision trees can be used in predicting the price of a house based on size and number of rooms.

Entropy:

Entropy is the measure of uncertainty of a random variable, it characterizes the impurity of an arbitrary collection of examples. The higher the entropy the more the information content.

For a dataset that has 'c' classes and the probability of randomly choosing data from class, i is p_i. Then entropy E(S) can be mathematically represented as:

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

Information Gain:

The Information Gain measures the expected reduction in entropy. Entropy measures impurity in the data and information gain measures reduction in impurity in the data. The feature which has minimum impurity will be considered as the root node.

For a dataset having many features, the information gain of each feature is calculated and then the feature having maximum information gain will be the most important feature which will be the root node for the decision tree.

$$\text{Information Gain} = \text{Entropy}_{\text{parent}} - \text{Entropy}_{\text{children}}$$

Where, Entropy $_{\text{parent}}$ is the entropy of the parent node.

Entropy $_{\text{children}}$ represents the average entropy of the child nodes that follow this variable.

Code for Experiment :

```
# Importing the required packages
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
import matplotlib.pyplot as plt

# Load the dataset
balance_data = pd.read_csv('balance-scale.csv')

# Displaying dataset information
print("Dataset Length: ", len(balance_data))
print("Dataset Shape: ", balance_data.shape)
print("Dataset First 5 rows: \n", balance_data.head())
```

```
# Separating the target variable
X = balance_data.values[:, 1:5]
Y = balance_data.values[:, 0]

# Splitting the dataset into train and test
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.3, random_state=100)

# Decision tree with entropy
clf_entropy = DecisionTreeClassifier(criterion="entropy", random_state=100,max_depth=3,
min_samples_leaf=5)

# Performing training
clf_entropy.fit(X_train, y_train)

# Plotting the Decision Tree
feature_names=['X1', 'X2', 'X3', 'X4']
class_names=['L', 'B', 'R']
plt.figure(figsize=(15, 10))
plot_tree(clf_entropy, filled=True, feature_names=feature_names, class_names=class_names,
rounded=True)
plt.show()

# Result of the Decision Tree Model
print("Results Using Entropy:")
y_pred_entropy = clf_entropy.predict(X_test)
print("Predicted values:")
print(y_pred_entropy)
print("Confusion Matrix: \n",confusion_matrix(y_test, y_pred_entropy))
print("Accuracy : ",accuracy_score(y_test, y_pred_entropy)*100)
print("Report : \n",classification_report(y_test, y_pred_entropy))
```

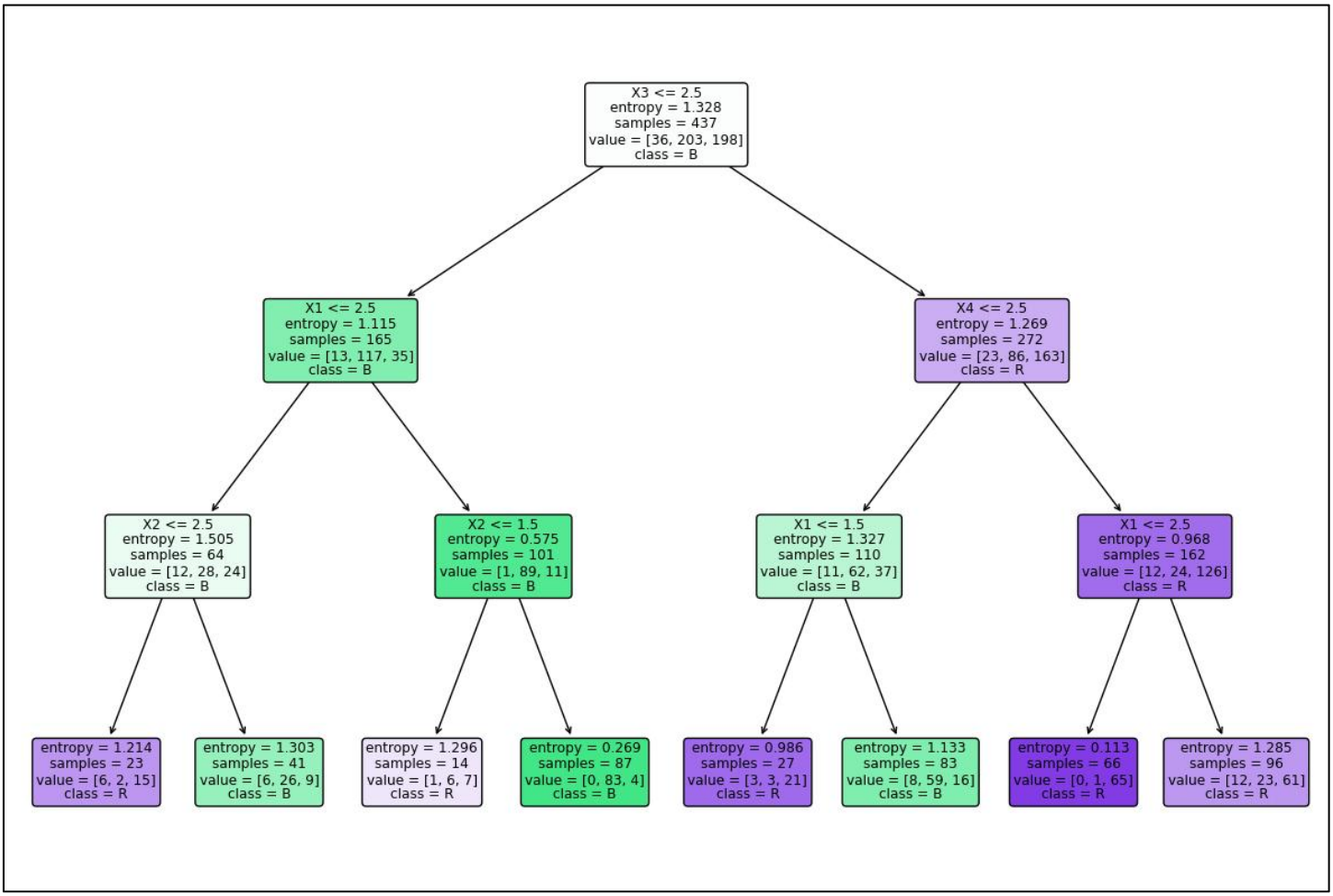
Result/Output :

jupyter ML_Experiment4_ASHISH_23MAI10008 Last Checkpoint: 10 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

```
Dataset Length:  625
Dataset Shape:  (625, 5)
Dataset First 5 rows:
   Class  L-Weight  L-Distance  R-Weight  R-Distance
0      B         1           1         1           1
1      R         1           1         1           2
2      R         1           1         1           3
3      R         1           1         1           4
4      R         1           1         1           5
```



jupyter ML_Experiment4_ASHISH_23MAI10008 Last Checkpoint: 11 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

```
Results Using Entropy:
Predicted values:
['R' 'L' 'R' 'L' 'R' 'L' 'R' 'L' 'R' 'R' 'R' 'R' 'L' 'L' 'R' 'L' 'R' 'L'
 'L' 'R' 'L' 'R' 'L' 'L' 'R' 'L' 'R' 'L' 'R' 'L' 'R' 'L' 'R' 'L' 'L' 'L'
 'L' 'L' 'R' 'L' 'R' 'L' 'R' 'L' 'R' 'R' 'L' 'L' 'R' 'L' 'L' 'R' 'L' 'L'
 'R' 'L' 'R' 'R' 'L' 'R' 'R' 'R' 'L' 'L' 'R' 'L' 'L' 'R' 'L' 'L' 'L' 'R'
 'R' 'L' 'R' 'L' 'R' 'R' 'R' 'L' 'R' 'L' 'L' 'L' 'L' 'R' 'R' 'L' 'R' 'L'
 'R' 'R' 'L' 'L' 'L' 'R' 'R' 'L' 'L' 'L' 'R' 'L' 'L' 'R' 'R' 'R' 'R' 'R'
 'R' 'L' 'R' 'L' 'R' 'R' 'L' 'R' 'R' 'L' 'R' 'R' 'L' 'R' 'R' 'R' 'L' 'L'
 'L' 'L' 'L' 'R' 'R' 'R' 'R' 'L' 'R' 'R' 'R' 'L' 'L' 'R' 'L' 'R' 'L' 'R'
 'L' 'R' 'R' 'L' 'L' 'R' 'L' 'R' 'R' 'R' 'R' 'R' 'L' 'R' 'R' 'R' 'R' 'R'
 'R' 'L' 'R' 'L' 'R' 'R' 'L' 'R' 'L' 'R' 'L' 'R' 'L' 'L' 'L' 'L' 'L' 'R'
 'R' 'R' 'L' 'L' 'L' 'R' 'R' 'R']
Confusion Matrix:
 [[ 0  6  7]
 [ 0 63 22]
 [ 0 20 70]]
Accuracy :  70.74468085106383
Report :
              precision    recall  f1-score   support

           B       0.00      0.00      0.00        13
           L       0.71      0.74      0.72        85
           R       0.71      0.78      0.74        90

    accuracy                           0.71       188
   macro avg       0.47      0.51      0.49       188
weighted avg       0.66      0.71      0.68       188
```

Learning outcomes (What I have learnt):

1. I learnt about various python libraries like pandas, sklearn.
2. I learnt about the concept of Decision Tree Classifier.
3. I learnt about the concept of Entropy and Information Gain.
4. I learnt about how to calculate the Accuracy and F1-score.
5. I learnt about how to split data at each node of Decision Tree.