```python
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
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree


def importdata():
  balance_data = pd.read_csv(
    'https://archive.ics.uci.edu/ml/machine-learning-' +
    'databases/balance-scale/balance-scale.data',
    sep=',', header=None)

  print("Dataset Length: ", len(balance_data))
  print("Dataset Shape: ", balance_data.shape)
  print("Dataset: ", balance_data.head())

  return balance_data


def splitdataset(balance_data):

  X = balance_data.values[:, 1:5]
  Y = balance_data.values[:, 0]

  X_train, X_test, y_train, y_test = train_test_split(
    X, Y, test_size=0.3, random_state=100)

  return X, Y, X_train, X_test, y_train, y_test


def train_using_gini(X_train, X_test, y_train):

  clf_gini = DecisionTreeClassifier(criterion="gini",
                random_state=100, max_depth=3, min_samples_leaf=5)

  clf_gini.fit(X_train, y_train)
  return clf_gini
```

main.py

```python
def train_using_entropy(X_train, X_test, y_train):

  clf_entropy = DecisionTreeClassifier(
    criterion="entropy", random_state=100,
    max_depth=3, min_samples_leaf=5)

  clf_entropy.fit(X_train, y_train)
  return clf_entropy


def prediction(X_test, clf_object):
  y_pred = clf_object.predict(X_test)
  print("Predicted values:")
  print(y_pred)
  return y_pred

def cal_accuracy(y_test, y_pred):
  print("Confusion Matrix: ",
    confusion_matrix(y_test, y_pred))
  print("Accuracy : ",
    accuracy_score(y_test, y_pred)*100)
  print("Report : ",
    classification_report(y_test, y_pred))


def plot_decision_tree(clf_object, feature_names, class_names):
  plt.figure(figsize=(15, 10))
  plot_tree(clf_object, filled=True, feature_names=feature_names, class_names=class_names, rounded=True)
  plt.show()


if __name__ == "__main__":
  data = importdata()
  X, Y, X_train, X_test, y_train, y_test = splitdataset(data)

  clf_gini = train_using_gini(X_train, X_test, y_train)
  clf_entropy = train_using_entropy(X_train, X_test, y_train)

  plot_decision_tree(clf_gini, ['X1', 'X2', 'X3', 'X4'], ['L', 'B', 'R'])
  plot_decision_tree(clf_entropy, ['X1', 'X2', 'X3', 'X4'], ['L', 'B', 'R'])

```

Run | Ask AI | 15m on 12:17:13, 04/25

```
Dataset Shape:  (625, 5)
Dataset:     0  1  2  3  4
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
```

Output

Figure 1