Assignment 5(b):

Problem Statement: Use balance-scale dataset for the prediction using Decision Tree classifier. Split the dataset into training and test dataset in 60:40 ratio. Train the model using by ginilndex and entropy. Perform the prediction on the test dataset.

Source Code (Decision Tree Classifier):

```
In [1]: #Importing Modules and Libraries:
    import pandas as pd
    import seaborn as sns
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
    from sklearn.tree import DecisionTreeClassifier
In [2]: #Dataset Loading and Preprocessing:
    data = pd.read_csv(r"D:\softwares\python\python programs\ML\Datasets\balance-scale.csv")
    data = data.dropna(axis=0, how='any', inplace=False)
    data
```

[2]:		Class	L-Weight	L-Distance	R-Weight	R-Distance
	0	В	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
	620	L	5	5	5	1
	621	L	5	5	5	2
	622	L	5	5	5	3
	623	L	5	5	5	4
	624	В	5	5	5	5

625 rows × 5 columns

```
In [3]: #Selecting [X1, X2...] and Y values for Classifictaion and Label Encoding:
    X = data.drop('Class', axis=1, inplace=False)
    Y = data.Class.astype('category').cat.codes
    x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.4, random_state=42)
    x_train
```

t[3]:		L-Weight	L-Distance	R-Weight	R-Distance
	149	2	1	5	5
	124	1	5	5	5
	465	4	4	4	1
	505	5	1	2	1
	185	2	3	3	1
	71	1	3	5	2
	106	1	5	2	2
	270	3	1	5	1
	435	4	3	3	1
	102	1	5	1	3

375 rows × 4 columns

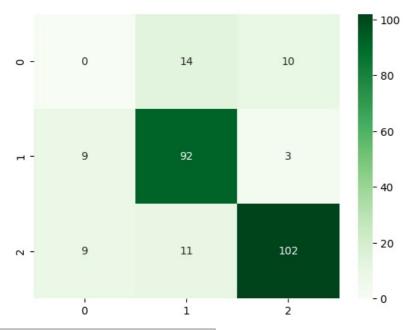
```
score = model.score(x_train, y_train)
             train_report['criterion'].append(c)
             train_report['score'].append(score)
        train_report = pd.DataFrame(train_report)
        print("Contrast between Training Performances of Gini and Entropy Impurity Indices:")
        train report
       Contrast between Training Performances of Gini and Entropy Impurity Indices:
Out[4]:
           criterion
                       score
               gini 0.994667
            entropy 0.992000
In [5]: #Model Testing and Testing Accuracy:
        test_report = x_test.copy()
        test report['Class'] = data.Class.astype('category').cat.categories[y test]
        for i in range(len(model_instances)):
             estimator = model instances[list(model instances.keys())[i]]
             y_pred = estimator.predict(x_test)
             accuracy = accuracy_score(y_test, y_pred)
             print(list(model_instances.keys())[i]+" Testing Accuracy:",accuracy)
             test_report['Class_'+list(model_instances.keys())[i]] = data.Class.astype('category').cat.categories[y_pred
        test_report
       gini Testing Accuracy: 0.776
       entropy Testing Accuracy: 0.776
Out[5]:
             L-Weight L-Distance R-Weight R-Distance Class Class_gini Class_entropy
        447
                    4
                              3
                                        5
                                                   3
                                                         R
                                                                   В
                                                                                 В
        485
                    4
                              5
                                        3
                                                         Ι
                                                                                 1
                                                                    L
                    2
                                        4
                                                         L
        215
                              4
                                                   1
                                                                    L
                                                                                 L
                    2
         212
                              4
                                        3
                                                   3
                                                         R
                                                                   R
                                                                                 R
         480
                    4
                              5
                                        2
                                                   1
                                                         L
                                                                   L
                                                                                 L
        439
                    4
                              3
                                        3
                                                   5
                                                         R
                                                                   В
                                                                                 В
         302
                    3
                              3
                                                   3
         497
                    4
                              5
                                        5
                                                   3
                                                         L
                                                                    L
                                                                                 L
                    2
                                                   5
                                                         R
                                                                   R
        249
                              5
                                        5
                                                                                 R
        277
                    3
                              2
                                        1
                                                   3
                                                         L
                                                                    L
                                                                                 L
        250 rows × 7 columns
In [6]: #Generating the Confusion Matrix and Classification Report:
        cm = pd.DataFrame(confusion_matrix(y test, y pred))
        report = classification_report(y_test, y_pred)
        print("Classification Report:",report)
        print("\nConfusion Matrix:")
       Classification Report:
                                              precision
                                                            recall f1-score support
                   0
                           0.00
                                      0.00
                                                0.00
                                                             24
                   1
                           0.79
                                      0.88
                                                0.83
                                                            104
                   2
                           0.89
                                      0.84
                                                0.86
                                                            122
                                                0.78
                                                            250
           accuracv
                           0.56
                                      0.57
                                                0.56
                                                            250
          macro avg
                           0.76
                                      0.78
                                                0.77
                                                            250
       weighted avg
```

Confusion Matrix:

```
Out[6]: 0 1 2
0 0 14 10
1 9 92 3
2 9 11 102
```

```
In [7]: #Visualizing the Confusion Matrix:
sns.heatmap(cm, annot=True, cmap='Greens', fmt='g')
```

```
Out[7]: <Axes: >
```



Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js