Homework2 - Zhihao Pei

Problem 1

Task

Load the Iris sample dataset from `sklearn` (using `load_iris()`) into Python with a Pandas DataFrame. Induce a set of binary decision trees with a minimum of 2 instances in the leaves (`min_samples_leaf=2`), no splits of subsets below 5 (`min_samples_split=5`), and a maximum tree depth ranging from 1 to 5 (`max_depth=1` to 5). You can leave other parameters at their default values. Answer the following questions:

- Which depth values result in the highest Recall? Why?
- Which value resulted in the lowest Precision? Why?
- Which value results in the best F1 score?

Also, explain the difference between the micro, macro, and weighted methods of score calculation.

Result Tables: Macro, Micro, Weighted

Macro			
max_depth	**Recall**	**Precision**	**F1 Score**
1	0.667	0.500	0.556
2	1.000	1.000	1.000
3	1.000	1.000	1.000
4	1.000	1.000	1.000
5	1.000	1.000	1.000

Micro			
max_depth	**Recall**	**Precision**	**F1 Score**
1	0.711	0.711	0.711
2	1.000	1.000	1.000
3	1.000	1.000	1.000
4	1.000	1.000	1.000
5	1.000	1.000	1.000

Weighted			
max_depth	**Recall**	**Precision**	**F1 Score**
1	0.711	0.567	0.615
2	1.000	1.000	1.000
3	1.000	1.000	1.000
4	1.000	1.000	1.000
5	1.000	1.000	1.000

Table 1: Macro, Micro, and Weighted performance metrics for different `max_depth` values.

Result Analysis

The maximum Recall value occurs at `max_depth` = 2, 3, 4, and 5, where the Recall values are all 1.000. Recall refers to the proportion of actual positive samples that are correctly predicted by the model. When `max_depth` is 2 or higher, the model's tree structure becomes more complex, allowing for better data segmentation.

The maximum Precision value also occurs at depths 2, 3, 4, and 5. As the complexity of the tree increases, it becomes more precise in distinguishing between positive and negative samples, reducing the number of false positives. Therefore, Precision improves to 1.000.

The maximum F1 Score occurs at depths 2, 3, 4, and 5. At `max_depth` = 1, although Recall is relatively good, Precision is low (0.500), resulting in a lower F1 Score (0.556). As

the depth of the tree increases, the model's classification performance improves significantly, causing the F1 Score to gradually increase.

Difference between Score Calculation Methods

- **Micro-average:** Combines the prediction results of all classes and then calculates the metrics for the entire dataset. This method reflects the global performance of the model.
- **Macro-average:** Calculates the metric for each class separately and then takes the arithmetic mean. This method gives equal weight to each class.
- **Weighted-average:** Calculates the metric for each class and then computes a weighted average based on the number of samples (support) in each class.

Problem 2

Task

Load the Breast Cancer Wisconsin (Diagnostic) sample dataset from the UCI Machine Learning Repository (the discrete version at: `breast-cancerwisconsin.data`) into Python using a Pandas DataFrame. Induce a binary Decision Tree with a minimum of 2 instances in the leaves, no splits of subsets below 5, and a maximum tree depth of 2 (using the default Gini criterion). Calculate the Entropy, Gini, and Misclassification Error of the first split. What is the Information Gain? Which feature is selected for the first split, and what value determines the decision boundary?

Result Table

First Split Feature	First Split Value	Gini Index of the Split	Entropy of the Split
concave_points1	0.05128	0.137367	0.377822

Misclassification Error of the Split	Entropy IG	Gini IG	Misclass. IG
0.73737	0.576288	0.33664	0.239955

Table 2: Metrics for the first split on the Breast Cancer Dataset (discrete version).

Result Analysis

Information gain is a metric used to measure the usefulness of a feature in a classification task. In decision tree algorithms, it is used to determine how to split the data. In this

problem, the first splitting feature selected is `concave_points1`, with 0.05128 chosen as the decision boundary.

Problem 3

Task

Load the Breast Cancer Wisconsin (Diagnostic) sample dataset from the UCI Machine Learning Repository (the continuous version at: `wdbc.data`) into Python using a Pandas DataFrame. Induce the same binary Decision Tree as above (now using the continuous data), but perform PCA dimensionality reduction beforehand. Using only the first principal component of the data for model fitting, determine the F1 score, Precision, and Recall of the PCA-based single factor model compared to the original (continuous) data. Then, repeat the process using the first and second principal components. Using the Confusion Matrix, provide the values for False Positives (FP) and True Positives (TP), as well as the False Positive Rate (FPR) and True Positive Rate (TPR). Is using continuous data beneficial for the model in this case? How?

Result Table (Third Table: Second Image)

Metric	**Original Data**	**PCA 1st Component**	**PCA 1st + 2nd Components**
F1 Score	0.904762	0.899225	0.885246
Precision	0.904762	0.878788	0.915254
Recall	0.904762	0.926635	0.857143

Table 3: Comparison of F1 Score, Precision, and Recall for PCA-based models (single and two-factor) vs. Original Data.

Result Table (Fourth Table: Third Image)

Metric	**Original Data**	**PCA 1st Component**	**PCA 1st + 2nd Components**
Confusion Matrix	$\begin{bmatrix} 102 & 6 \\ 6 & 57 \end{bmatrix}$	$\begin{bmatrix} 100 & 8 \\ 5 & 58 \end{bmatrix}$	$\begin{bmatrix} 103 & 5 \\ 9 & 54 \end{bmatrix}$
FP	6	8	5
TP	57	58	54
FPR	0.055556	0.074074	0.046296
TPR	0.904762	0.926635	0.857143

Table 4: Confusion Matrix metrics for PCA-based models vs. Original Data.

Result Analysis

The F1 Score, Precision, and Recall results for the single-factor model, two-factor model, and raw data are presented in the third table above.

Repeating the process using the first and second principal components, the results based on the confusion matrix (FP, TP, FPR, TPR) are shown in the fourth table. Based on these outcomes, using continuous data (raw data) yields better F1 scores, precision, and recall. Although PCA dimensionality reduction reduces computational complexity and minimizes redundant information, it also leads to a decrease in model performance. When using only the first principal component, both the F1 score and recall decreased. The two-factor model (PC1 + PC2) slightly improved precision but still did not perform as well as the model using the raw data.