ID: A20563448
Name: Zibo NIE **Data Mining HW2.0** **Date: 3/20/2025**

1 Problem 1: Decision Tree on the Iris Dataset

Dataset: The Iris dataset is a well-known multi-class classification dataset containing measurements of three types of iris flowers (*Setosa, Versicolor, and Virginica*). It includes **four numerical features** (sepal length, sepal width, petal length, petal width) and a target class.

Task:

- Load the Iris dataset into a Pandas DataFrame using `sklearn.datasets.load_iris()`.
- Train a **binary decision tree classifier** with the following parameters:
  - `min_samples_leaf = 2`: Each leaf node must have at least 2 instances.
  - `min_samples_split = 5`: Nodes require at least 5 samples to split.
  - `max_depth` ranging from **1 to 5** to evaluate the impact of tree depth on performance.
- Keep all other parameters at their default values.

Analysis:

- Identify which **tree depth** achieves the highest **Recall**, and explain why.
- Determine which depth results in the **lowest Precision**, and discuss possible reasons.
- Find the depth that provides the **best F1 score**, balancing Recall and Precision.
- Explain the differences between **micro, macro, and weighted** averaging methods in classification performance evaluation.

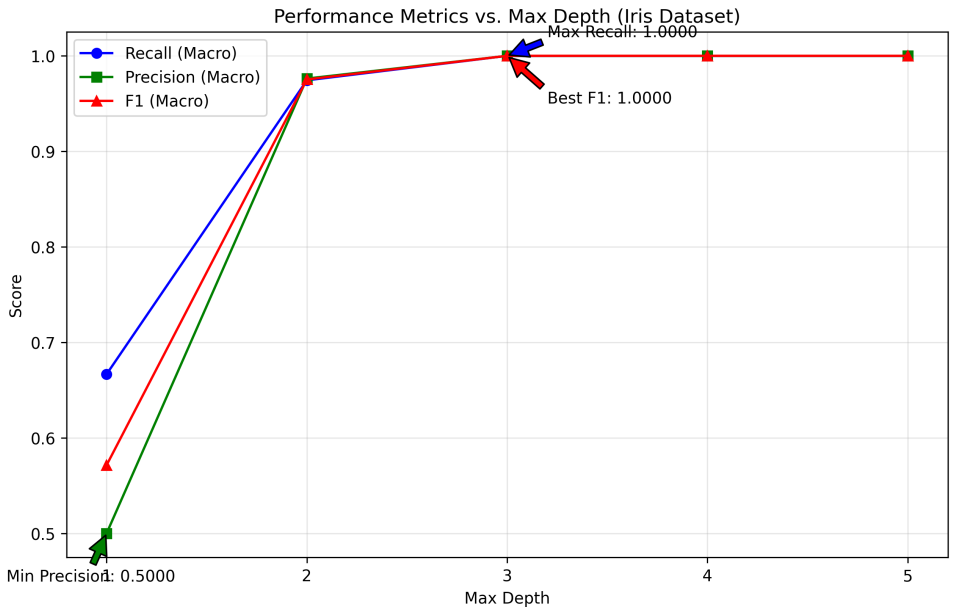

Figure 1: Values of Recall,Precision and F1 at different Max Depth

Figure 1 illustrates the values of the three indicators at different max_depth.It can be seen that when a increases, these three values also increase, and combining this with table 1 shows that when a reaches 3, the values of all three indicators increase to 1.00.

Table 1: Performance Metrics Across Different `max_depth` Values

	Micro			Macro			Weighted		
`max_depth`	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
1	0.7111	0.7111	0.7111	0.5000	0.6667	0.5714	0.5667	0.7111	0.6307
2	0.9778	0.9778	0.9778	0.9762	0.9744	0.9753	0.9794	0.9778	0.9786
3	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
4	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

From the data, it can be surmised that when the depth of the tree reaches 3, the model can completely differentiate between the different species with **complete accuracy**.

The lowest values of these three metrics are found when the depth of the tree is only 1. This is due to the fact that there are not **enough features** for training, resulting in a tree that cannot accurately distinguish between different classes.

- **Micro**: Combines all predictions into one set and computes the metric, emphasizing overall performance and favoring frequent classes.
- **Macro**: Calculates the metric for each class independently and takes an unweighted average, ensuring equal treatment of all classes regardless of size.
- **Weighted**: Computes per-class metrics and averages them, weighted by class size, balancing overall performance with class distribution.

As can be seen from Table 2, there is no difference between **Micro**, **Macro** and **Weighted**. This is due to the balanced distribution of the different categories of the dataset used in the question instead of the indication that the three are the same.

2 Problem 2: Decision Tree on Breast Cancer (Discrete Data)

Dataset: The Breast Cancer Wisconsin (Diagnostic) dataset (discrete version) consists of **categorical features** representing tumor characteristics and a **binary target** (benign or malignant).

Task:

- Load the dataset from the UCI Machine Learning Repository into a Pandas DataFrame.
- Train a **binary decision tree classifier** with:
  - `min_samples_leaf = 2`
  - `min_samples_split = 5`
  - `max_depth = 2`
  - Default `Gini` impurity criterion.
- Compute **Entropy, Gini impurity, and Misclassification Error** for the first split.
- Calculate **Information Gain** for the first split.
- Identify the **feature selected for the first split** and determine the **decision boundary** value.

Analysis:

- How does the **selected feature** impact the decision-making process?
- What does the **Information Gain** reveal about feature importance?
- How do **Gini, Entropy, and Misclassification Error** compare in terms of splitting criteria?

- **Entropy**: Measures the uncertainty in a node based on class probabilities.

$$H(D) = -\sum_{i=1}^{k} p_i \log_2(p_i) \tag{1}$$

where p_i is the proportion of instances in class i, and k is the number of classes.

- **Gini Index**: Measures the probability of misclassification.

$$Gini(D) = 1 - \sum_{i=1}^{k} p_i^2 \tag{2}$$

where p_i is the proportion of instances in class i, and k is the number of classes.

- **Misclassification Error**: Measures the error rate based on the majority class.

$$Misclass(D) = 1 - \max(p_i) \tag{3}$$

where p_i is the proportion of instances in class i, and the maximum is over all classes.

- **Information Gain**: Measures the reduction in uncertainty after splitting a dataset based on a feature.

$$IG(T, A) = Entropy(T) - \sum_{v \in Values(A)} \frac{|T_v|}{|T|} \cdot Entropy(T_v) \tag{4}$$

where $Entropy(T)$ is the entropy of the dataset T before splitting, $Values(A)$ is the set of possible values of feature A, T_v is the subset of T where feature A takes value v, and $|T_v|/|T|$ is the proportion of instances in T_v relative to T.

Table 2: Impurity Measures and First Split Details

	Entropy	Gini	Misclassification Error
Root Node	0.9340	0.4550	0.3499
After Split	0.3451	0.1294	0.0732
Information Gain		0.5889	
Split Feature		uniformity_size	
Threshold		2.5000	

Table 2 shows Entropy, Gini and Misclassification before and after node splitting. From the table, it can be seen that the **splitting boundary** of nodes is based on **unifromoty_size**. A set of data is split to the right child if the a feature is greater than 2.5 and vice versa for the left child.

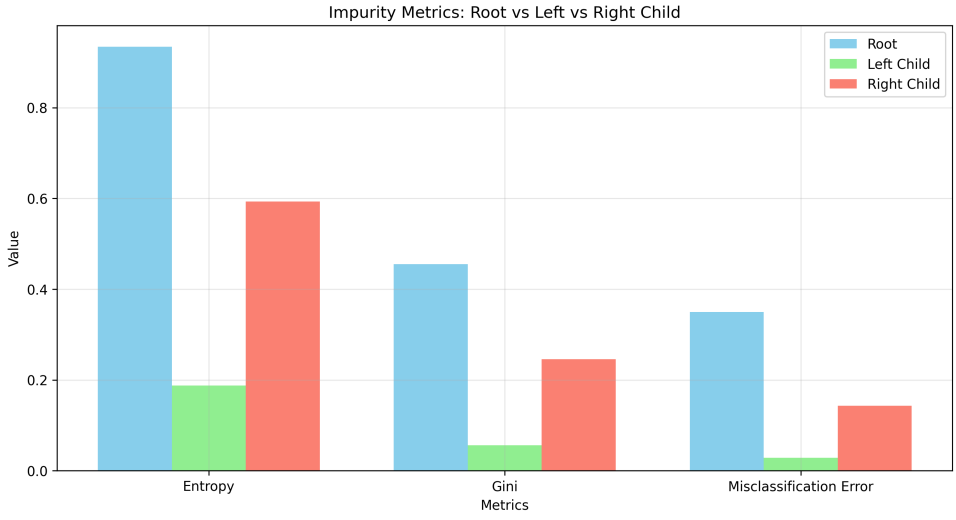

Figure 2: Values of entropy,gini and misclassification error before and after split

As can be seen in Figure 2, the root node has a significant decrease in the values of **Entropy**, **Gini Metrics** and **Misclassification Error** after the split.

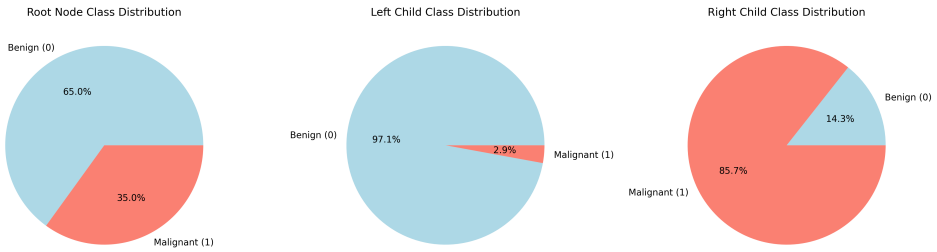

Figure 3: Percentage of content of each category in the three nodes

Figure 3 demonstrates the percentage of nodes in the two categories before and after the split, and it can be seen that after the split, the nodes have a purer composition.

3 Problem 3: Decision Tree with PCA on Breast Cancer (Continuous Data)

Dataset: The Breast Cancer Wisconsin (Diagnostic) dataset (continuous version) contains numerical features extracted from digitized images of cell nuclei, such as radius, texture, perimeter, and smoothness. The target variable is **binary** (benign or malignant).

Task:

- Load the dataset from the UCI Machine Learning Repository into a Pandas DataFrame.
- Apply **Principal Component Analysis (PCA)** for dimensionality reduction.
- Train a **binary decision tree classifier** (same settings as Problem 2) using:
  - **Only the first principal component** of the data.
  - **The first two principal components** for comparison.
- Compare model performance on **continuous data** versus **PCA-reduced data**.
- Compute **F1 score, Precision, and Recall** for both versions.
- Use a **Confusion Matrix** to extract:
  - **False Positives (FP)** and **True Positives (TP)**
  - **False Positive Rate (FPR)** and **True Positive Rate (TPR)**

Analysis:

- How does **dimensionality reduction** impact classification performance?
- Is the **PCA-based single-factor model** effective compared to using the full feature set?
- Does using continuous data provide better predictive power? Why or why not?

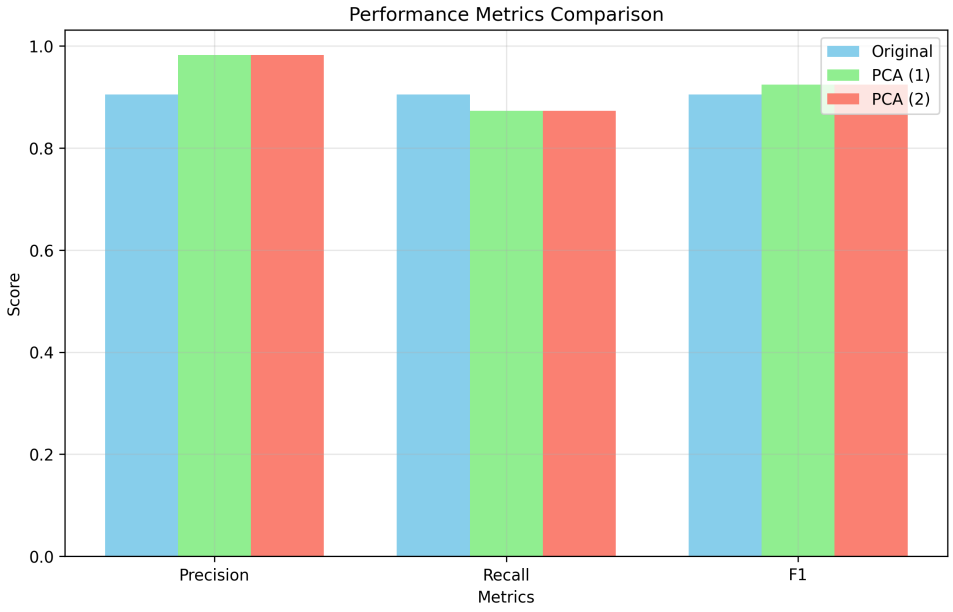

Figure 4: Values of precision, recall metrics and F1 in three distict dimensions

Table 3: Performance Metrics for Different Data Configurations

Data	**Precision**	**Recall**	**F1**	**FP**	**TP**	**FPR**	**TPR**
Original	0.9048	0.9048	0.9048	-	-	-	-
PCA (1 component)	0.9821	0.8730	0.9244	1	55	0.0093	0.8730
PCA (2 components)	0.9821	0.8730	0.9244	1	55	0.0093	0.8730

As can be seen in Figure 4 and Table 3, **PCA** increased precision and F1 and decreased Recall Metrics.When the principal components are 1 and 2, there is no difference between these three indicators, which suggests that the second principal component in PCA(2) has little or no effect on the model.

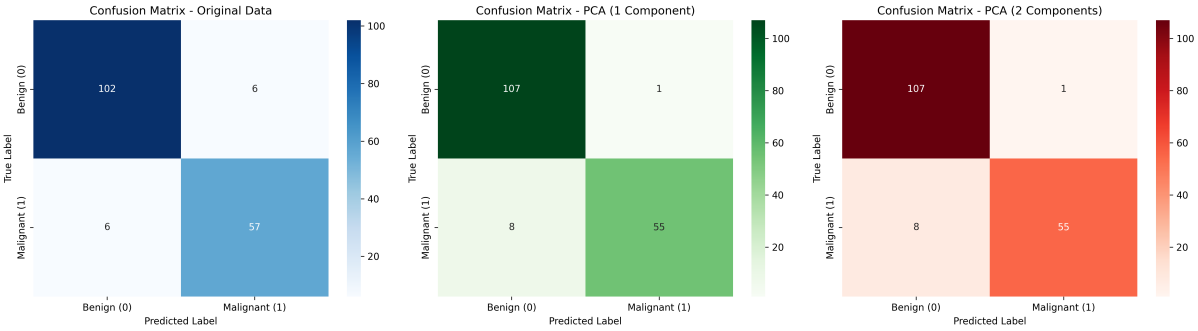

Figure 5: Values of FP,TP,FPR and TPR on three different dimensions

In this case, using **continuous data** is not optimal for the model. While directly using raw continuous data retains more information, it may **introduce noise and redundancy**, resulting in limited model performance. In contrast, **continuous data** processed through dimensionality reduction (e.g., PCA) is more beneficial as it effectively **reduces noise**, lowers the risk of **overfitting**, and improves the **classification performance and generalisation** of the model.